IEICE TRANS. INF. & SYST., VOL.E102–D, NO.5 MAY 2019

INVITED PAPER *Special Section on the Architectures, Protocols, and Applications for the Future Internet*

AI@ntiPhish — Machine Learning Mechanisms for Cyber-Phishing Attack

Yu-Hung CHEN[†*a)] ***and*** **Jiann-Liang CHEN**[†]**,** ***Nonmembers***

SUMMARY This study proposes a novel machine learning architecture and various learning algorithms to build-in anti-phishing services for avoiding cyber-phishing attack. For the rapid develop of information technology, hackers engage in cyber-phishing attack to steal important personal information, which draws information security concerns. The prevention of phishing website involves in various aspect, for example, user training, public awareness, fraudulent phishing, etc. However, recent phishing research has mainly focused on preventing fraudulent phishing and relied on manual identification that is inefficient for real-time detection systems. In this study, we used methods such as ANOVA, X^2, and information gain to evaluate features. Then, we filtered out the unrelated features and obtained the top 28 most related features as the features to use for the training and evaluation of traditional machine learning algorithms, such as Support Vector Machine (SVM) with linear or rbf kernels, Logistic Regression (LR), Decision tree, and K-Nearest Neighbor (KNN). This research also evaluated the above algorithms with the ensemble learning concept by combining multiple classifiers, such as Adaboost, bagging, and voting. Finally, the eXtreme Gradient Boosting (XGBoost) model exhibited the best performance of 99.2%, among the algorithms considered in this study.

key words: *anti-phishing, machine learning algorithm, ensemble learning mechanism, cyber attack*

1. Introduction

With the rapid development of communication technologies, people have become increasingly dependent on information technology. In contrast, people have also become increasingly dependent on Internet services since the emergence of the Internet of Things. To make our life more convenient, all types of application services are gradually emerging, such as bank transfers, online credit cards, and online shopping. To ensure that a purchaser is a legitimate user, personal information needs to be uploaded to the application database for verification via the Internet. As user information must be uploaded via the Internet, many hackers engage in financial crimes and computer attacks. Among them, phishing is a method to steal personal data through the Internet [1].

Phishing is a cyber-crime based on social engineering. It uses the careless nature of human beings to achieve the purpose of committing crimes, which leads to the leakage of the victim's personal information. Phishing attacks may even extend to a company or an organization, causing the leakage of information within the company or organization. Once the data are leaked, the attacker can perform more serious internal attacks [2].

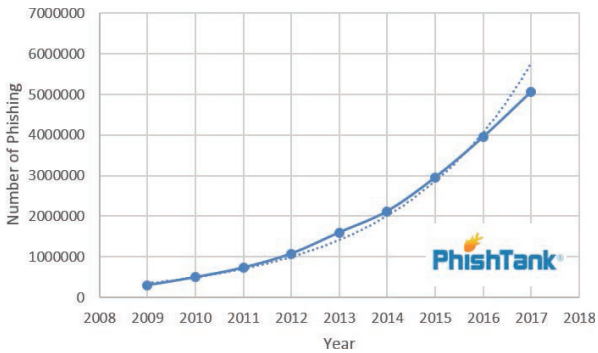

Fig. 1 Statistics of phishing websites from PhishTank

According to the analysis of the APWG in 2018, phishing attacks were found in more than 200 different top-level domain names, and the services of free domain names continued to be abused. In the third quarter of 2017, the number of phishing attacks counted by the APWG was 296,208, an increase of nearly 23,000 over the previous quarter. Among them, SAAS and webmail providers had a significantly increased the number of targeted phishing attacks [3], as shown in Fig. 1.

1.1 Motivation

In the existing detection technologies, blacklisting is mainly used for phishing detection. The blacklist method is mainly based on a comparison with the phishing links that were previously reported by victims [4]. However, this technology has some problems. Because the victim may not necessarily return the URL of the phisher and it is possible that the phishing URL has not appeared, the database is not recorded, and the user will be deceived.

In addition, there are many related studies that propose detection through image analysis and similarity techniques [5], [6]. However, these methods require a long run time, and the image database needs to include multiple pages for each page of a legitimate website; otherwise, it will be considered suspicious.

In recent years, many different machine learning algorithms have been developed for phishing detection. The main objective of the present study is to improve the overall training structure of these algorithms and improve the accu-

Manuscript received September 5, 2018.
Manuscript revised December 18, 2018.
Manuscript publicized February 18, 2019.
[†]The authors are with the National Taiwan University of Science and Technology, Taiwan.
[*]Presently, with the TREND MICRO Inc.
a) E-mail: M10507503@mail.ntust.edu.tw
DOI: 10.1587/transinf.2018NTI0001

racy of the architecture.

1.2 Contribution

The proposed architecture mainly uses machine learning algorithms to train phishing detection models. During the training period, this research also added other frameworks for preprocessing. The purpose of preprocessing was to filter out the unrelated or noise-related features in advance. This process improved the effectiveness of the training model. It also included feature extraction, evaluation, and coding modules that were pre-processed. After the design of the overall architecture, the stability and the accuracy of the original model were further improved, and the model was made more efficient and stable.

The contributions of this study can be summarized as follows.

- In this research, we extracted the characteristics of the previous literature and then, further analyzed and evaluated whether these characteristics had the same influence and relevance for the current phishing issue, which can help other threat experts understand the important characteristics of phishing research.
- We analyzed the phishing issue and trained from the extracted features by using different machine learning algorithms such as voting, bagging, and Adaboost, that is to understand what kind of algorithm is suitable for our extracted features in this study.
- To enhance the variability of our trained model, we further apply imbalanced learning method and evaluate the model performance for phishing detection.

The novelty of this paper is we mainly propose to use unbalanced learning algorithms, such as SMOTE, to achieve better performance for different integrated learning algorithms. We found that using the final extracted 28 features from feature evaluation module to train the model could achieve higher accuracy through the XGBoost algorithm with imbalanced learning. In addition to filtering out unrelated feature information from feature extraction module, the SMOTE method also improves the detection coverage of the model, making the detection model more robust and accurate.

The rest of this paper is organized as follows: The Sect. 2 introduces the background knowledge of phishing website detection and the research related to this study. The proposed intelligent learning architecture is described in Sect. 3. The performance analysis and the parameter settings of this study are presented in Sect. 4. Finally, we describe the conclusions and future work in Sect. 5.

2. Related Work

In the recent years, many methods to detect phishing, such as webpage similarity, blacklisting, and image processing, have been developed. These methods can be roughly divided into three categories, namely blacklisting, visual similarity analysis, and heuristics analysis. Some of these various technologies are described below.

2.1 Blacklist

Blacklisting is a traditional phishing detection technique. This technique mainly involves a comparison with the phishing URLs reported by previous victims. When a victim is deceived by clicking on a phishing email, the URL will be reported back and stored in the database after being verified by the analysis engineer [7], [8].

Therefore, when a user browses the webpage through the browser, this URL will be compared with the blacklist before being sent to the DNS. When this URL is successfully matched with the URL in the database, it will be identified as a suspicious or dangerous URL, and the user will be warned and blocked to ensure that the user can detect it and protect himself/herself in advance.

Most of the existing browsers or DNS, including Google, Firefox, and Quad9 [9], use the blacklisting technology for defense. However, the blacklist database is built via victim or user replies; this has the disadvantage of being unable to detect a URL that has not been replied to earlier.

2.2 Visual Similarity Analysis

In the recent years, an analysis of image similarity has also been used for phishing detection. This technology mainly captures the image of the page to be recognized and performs feature extraction through algorithms such as SIFT, SURF, and HOG. Finally, this image is compared with all the images in the image database, and the URLs of the simulated legitimate websites will be extracted and compared with each other.

Haruta et al. [10] proposed the use of an image similarity analysis for phishing detection. Their method combines an image similarity analysis and the CSS format to identify legitimate websites and phishing websites. In this mechanism, the authors prepared many image databases and images of new web pages, and then, stored the corresponding CSS format of each web page in another database for the searching process. If there is a similar web page in the image database, the given web page will be recognized as a phishing website. Afroz et al. [6] proposed the PhishZoo architecture. This architecture is mainly based on hierarchical concepts. First, the author compares the suspicious website with the proposed whitelist. If the identification URL is not on the whitelist, the SIFT technology is used to analyze the image similarity between the author proposed normal page screenshot and suspicious webpage, and compare other webpage features, such as HTML, tokens, and other feature information. In contrast, if the recognition result matches the URL listed in the whitelist, the given website will be identified as normal website.

Image based methods, such as image similarity analysis, logo analysis, or CSS format comparisons, have the disadvantage of could not recognize efficiently when the URL

to be identified is not in the already built image database or logo database.

2.3 Heuristics Analysis

A heuristic analysis is conducted to extract the unique characteristics of past legitimate websites and phishing websites, such as URL, HTML source code, and the who-is information. The extracted features are quantified, trained, and learned through artificial intelligence algorithms, thus enhancing the phishing identification capabilities of the detection model.

Subasi et al. [11] used different algorithms for the training of phishing detection models. These algorithms included Artificial Neural Networks (ANN) [1], [12], [13], K-Nearest Neighbor (KNN) [14], Support Vector Machine (SVM) [15], [16], C4.5 Decision Tree [17], [18], Random Forest (RF) [19], etc. According to the analysis results of this experiment, the authors proposed that the dataset provided by the UCI machine learning repository [20] was more suitable for training and prediction through a tree-structured algorithm. However, the accuracy and the stability of the model trained by a single machine learning algorithm were not excellent; therefore, some researchers proposed to train the model by using the method of ensemble learning [21]. Shiraz et al. [1] proposed a framework for automated phishing detection architecture based on the features defined by Mohammad et al. [21], which contained five types of feature information: URL, DNS, external statistics, HTML, and JavaScript. In the training process, the authors used different optimization methods for the deep learning architecture, such as adagrad, adadelta, and gradient descent; further, they used different kernel functions for the SVM algorithms. Verma et al. [22] used four different sources of datasets for training and analysis, including PhishTank.com, APWG, and the DMOZ Open Directory Project [23]. The authors used the logistic regression [24], J48 decision tree [18], and random forest algorithms [25], and analyzed the accuracy of the models through five cross-validations. Finally, the authors evaluated more relevant feature combinations and trained the most effective detection models. The most commonly used methods for its algorithms include Voting [21], [26]–[28], Stacking [21], Adaboost [29]–[33], etc. Based on the above description, Tahir et al. [21] propose a Hybrid Model for phishing detection via Ensemble Learning. The author mainly use the Bagging [32] method to train the phishing dataset from the UCI machine learning repository [20]. The author combined the results of several algorithms for phishing analysis. The algorithm includes Random Forest (RF), Naive Bayes (NB), Fuzzy, etc. However, the identification method is to vote the prediction results of all algorithms. If the prediction result is that the phishing website has a large number of votes, it is identified as a phishing website; otherwise, it is recognized as a legitimate website. In the process of model training, the author determines the timing of the model output by setting the threshold.

Although heuristic analysis could enhance the phishing identification capabilities, it has great correlation with the amount of data. The higher number of collected phishing websites, the wider the model can learn, which result in better recognition ability. In contrast, when the training data is insufficient, it's hard to accumulate enough experience to determine whether it is a phishing website or not and result in higher error rate. However, the lifecycle of phishing website is short, it is not easy to collect large number of phishing pages. Also, the number of normal websites is much larger than the number of malicious websites, which made the heuristic analysis model trend to give the normal prediction. Therefore, this research proposed Intelligent Learning Architecture that balanced the gap between the number of normal and malicious website, and generated more data automatically.

3. Intelligent Learning Architecture

In this study, we developed intelligent learning architecture, which is shown in Fig. 2.

Since the method of Heuristic analysis can solve the problem that black list and Image-based method would have, this research mainly focus on the extension of Heuristic analysis method and solved the problem of having insufficient data.

In this study, we used the Data processing layer to generate the insufficient data by SMOTE method and analyze the validity of each feature through ANOVA, X^2 and Information Gain method.

3.1 Training Dataset

In this component, we collect a large number of phishing sites and legitimate sites through the network. We then send the URL to the pre-processing unit and finally, extract the raw data as the training data.

In this study, we collected 24471 phishing sites from PhishTank in the collection model, with 3850 legitimate sites retrieved from the target column of the corresponding

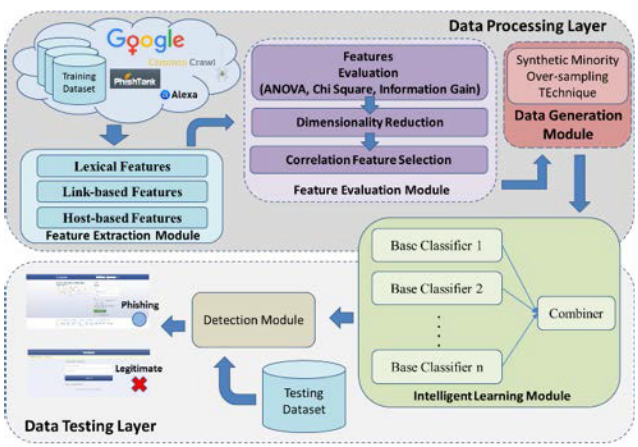

Fig. 2 Learning architecture

Table 1 Phishing Features

Symbol	Feature	Symbol	Feature
f_1	is_http_connection	f_{15}	null_a_tag
f_2	is_ip_address	f_{16}	script_block_rate
f_3	dots	f_{17}	style_block_rate
f_4	is_special_words	f_{18}	get_title_feature
f_5	url_linkin_num	f_{19}	is_login_form
f_6	url_traffic_rank	f_{20}	is_with_whois
f_7	get_kbytes	f_{21}	get_time
f_8	is_frame	f_{22}	is_redirect
f_9	is_meta_redirect	f_{23}	ipv4_numbers
f_{10}	is_meta_base64_redirect	f_{24}	ipv6_numbers
f_{11}	same_extern_domain_script_rate	f_{25}	organization
f_{12}	same_external_domain_link_rate	f_{26}	is_alias
f_{13}	same_external_domain_img_rate	f_{27}	is_weird_serial
f_{14}	external_a_tag_same_domain	f_{28}	get_day_age

phishing sites. Basically, the number of phishing sites was considerably larger than the number of legitimate sites, because hackers usually imitate a specific legitimate site and design multiple similar phishing sites.

3.2 Feature Extractor Module

When the collected URL, including the phishing and the normal website, is transferred to the Feature Extraction Module, the module will perform feature extraction.

Feature representation is essential as it provides the data matrix for training and detecting phishing web pages. This research represents the web pages by using 28 URL features used as suspicious features in previous research [21], [32], [35]; these features are listed in Table 1.

- is_http_connection (f_1): In the process of applying for https encryption, the certification authority is required to verify whether the content of the applied website has its legitimacy and certainty. Therefore, using https protocol exists to verify the validity of the website. If it uses http protocol, the value will be '1', otherwise '0'.
- is_ip_address (f_2): Some Phishing's URL had IP address. For example, https://192.168.0.159/index.php. If having IP address, the value will be '1', otherwise '0'.
- dots (f_3): The creation of phishing websites is to allow users to be deceived due to their lack of protection. Therefore, well-known domain names are often added to phishing URLs as sub-domains, which can be used to confuse victims and make them believe that this is the URL of a legitimate website. The value will be the number of dots in a url.
- is_special_words (f_4): Phishing websites usually add special symbols to the URL, such as @, —, *, which are mainly to confuse the user's vision, and thus allow the user to click the phishing link. If the url contain special words, the value will be '1', otherwise '0'.
- url_linkin_num (f_5): This feature counts the number of URLs that are linked to the current website.
- url_traffic_rank (f_6): The ranking of Alexa Rank as a feature has considerable influence. Since traditional phishing websites have low-life characteristics, their rankings are normally less than 100,000. If true, the value will be '1', otherwise '0'.
- get_kbytes (f_7): In general, phishing websites will reduce the risk of detection by using a large amount of CSS, that makes the appearance of web pages with high similarity. Therefore, this study uses the size of the webpage format as one of the features.
- is_frame (f_8): The Iframe is an HTML tag that causes the page content of the current web page to display the page content of other web pages. Hackers can use the "iframe" tag and close their borders, which will cause the browser to render a visual depiction. If true, the value will be '1', otherwise '0'.
- is_meta_redirect (f_9): Using the meta refresh to redirect the address. If true, the value will be '1', otherwise '0'.
- is_meta_base64_redirect (f_{10}): Encoding the address by base64 and the meta refresh to redirect this encoding variable (address). If true, the value will be '1', otherwise '0'.
- same_extern_domain_script_rate (f_{11}): Phishing usually use the JavaScript to reference many external resources. Therefore, it is possible to count the highest proportion of external websites in the JavaScript and to identify whether the website quotes many external links.
- script_block_rate (f_{12}): Since many phishing websites use JavaScript to steal the templates of legitimate websites, this study uses the ratio of the JavaScript usage as one of the features.
- style_block_rate (f_{13}): Phishing websites usually use a lot of CSS to imitate, so observe the proportion of CSS in the entire html, which can identify whether this site is imitate other websites.
- external_a_tag_same_domain (f_{14}): A phishing website is usually link to the outside by a tag, so observe the highest proportion of external links by a tag, which can identify whether this site is suspicious.
- null_a_tag (f_{15}): The remaining links in phishing websites are usually not clickable. Like Gmail's phishing website, its forgot email tag cannot be clicked. Therefore, this study will calculate how many null tags are in the entire web page as one of the reference features.
- same_external_domain_link_rate (f_{16}): In general, the referenced resource of the phishing websites and the linked domain name after clicking the links are not under its own domain name. This study counts the highest proportion of its external websites as one of its features.
- same_external_domain_img_rate (f_{17}): Phishing websites often reference external images, so observe the highest proportion of external links by img tag, which can identify whether this site is imitate other websites.
- get_title_feature (f_{18}): Some phishing sites want to mimic a particular site, so it sets the title to the name of the object to be mimicked. If the title mimicked the popular site from Alexa statistic, the value will be '1', otherwise '0'.

- is_login_form (f_{19}): The purpose of creating a phishing website is to steal personal data or confidential information. Therefore, by judging whether this webpage format contains a login form, it can doubt whether this website has the possibility of phishing websites. If true, the value will be '1', otherwise '0'.
- is_with_whois (f_{20}): DNS records the domain name information of each website. When a DNS record is not recorded or missing, it is identified as a suspicious URL. If true, the value will be '1', otherwise '0'.
- get_time (f_{21}): Phishing websites are usually set up on bad web servers or remote areas, so their response speed is slow. The value is the amount of time when download a webpage.
- is_redirect (f_{22}): A phishing website usually sends user's personal information to an externally constructed database and forwards it to other domains. Therefore, this study counts whether there is a behavior of header forwarding as one of the features. If true, the value will be '1', otherwise '0'.
- ipv4_numbers and ipv6_numbers (f_{23} and f_{24}): The number of phishing websites is usually not large, so the number of ipv4 and ipv6 is one of the characteristics. The value will be the number of ipv4 and ipv6 ip under the web server respectively.
- organization (f_{25}): The hostname of a normal URL is usually the same as the name of the registrar, such as www.apple.com, its registrar is APPLE. Therefore, distinguishing the registrar's name and hostname in the Whois information may also be a feature. If true, the value will be '1', otherwise '0'.
- is_alias (f_{26}): Phishing websites often create new subdomains and domain aliases in their DNS zones without the victim's knowledge. These aliases can be used for spam, phishing sites, or mail accounts. This study will identify whether there is an alias in the DNS record as a feature. If true, the value will be '1', otherwise '0'.
- is_weird_serial (f_{27}): Legitimate websites are regularly maintained and updated, and phishing websites are not regularly updated frequently. Therefore, one of the features can be viewed by identifying whether the website is regularly updated. If true, the value will be '1', otherwise '0'.
- get_day_age (f_{28}): Most phishing websites have a very short survival time. Therefore, it can be viewed via Whois command. There are many documents pointing out that the minimum age of its legal domain is 6 months. If the age of website is smaller than 6 months, the value will be '1', otherwise '0'.

3.3 Feature Evaluation Module

When the extracted feature URL is sent to the feature evaluation module for analysis, the unrelated and noise feature information will be filtered out.

There are three bases for the assessment here, which is to assess the degree of data cluster, data distribution, and data independence. This study mainly uses Information Gain, ANOVA, and X^2 methods [36] to analyze and evaluate each extracted feature. Each feature evaluation method has different characteristics, such as Information Gain is used to assess the degree of the data cluster and to understand the degree of confusion in the features. Then, ANOVA and X^2 are used to estimate the probability distribution of the data.

In this study, when the assessed feature is not related and influential, this feature is initially filtered to improve the stability and effectiveness of the detection model. Finally, the more relevant and influential features are extracted to next layer for training.

3.4 Data Generation Module

After the feature evaluation module, the final extracted feature can make the model more stable, and the training time can be greatly shortened. In this study, we used the extracted feature output from feature evaluation module for generating the insufficient data that solving the data imbalance issue and enhancing the detection coverage of the model.

Phishing websites typically imitate legitimate websites, so the number of phishing websites is much larger than the legitimate websites, which arise the issues of data imbalance between the phishing website and legitimate website during data collection. Also, data imbalance leads the model trending to the category with larger number of data. To solve the problem of model migration, this study utilizes SMOTE method to generate new instances, which complement the category with lesser data. Since the existing literature does not have much discussion on this issue, this study proposes to use the SMOTE method in this architecture.

The theory of SMOTE is to find K nearest neighbor data points of the same category and draw lines between each other, then generate similar data on lines. The formula is as follows:

$$x_{new} = x_i + (x' - x_i) \times \delta$$

x_i is a data point random select from the minority class dataset; x' is the other data point that is nearest to δ is between 0 and 1.

In the experiment, this study assessed the difference and accuracy of SMOTE. Finally, this study proposes a more stable and highly accurate generation framework for phishing research.

3.5 Intelligent Learning Algorithm

Finally, when the generated data and the original training data are sent to intelligent learning module, the module will train the detection model and optimize the weight of the model.

Once the features have been extracted, a binary classifier is trained using the presented 28 URL features. In this study, we used the traditional learning algorithms with ensemble learning [32] such as bagging, Adaboost, and voting.

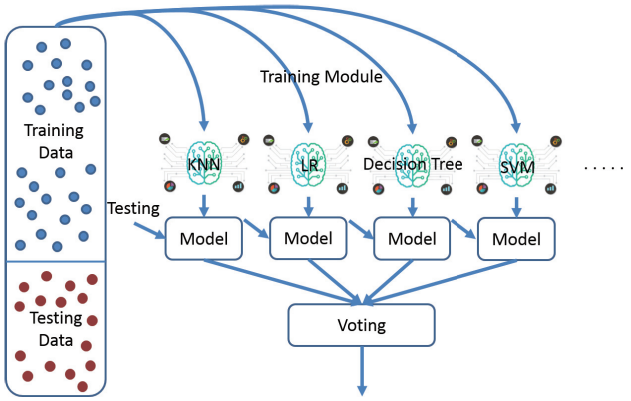

Fig. 3 Ensemble learning

Ensemble learning helps to improve the results of traditional machine learning algorithms by combining multiple detection models, as shown in Fig. 3. In addition, this study evaluated the use of the RF [21] method and of XGBoost [35] with imbalanced learning [37] to make up the problem of data imbalance.

Bagging or voting is a classification algorithm that utilizes many weaker models to build a stronger model that represents the average of the weaker models, as shown in Fig. 3. The Bagging mechanism is to sample the training data repeatedly, and generate multiple subsets, then create multiple models in sequence, and finally combine the results of all models together. In this method, bagging helps to reduce variance for training the model. The formula is as follows:

$$f_{prediction} = \frac{1}{n}\sum_{i=1}^{n} f(x)_i$$

$f_{prediction}$ is the final prediction result; x is a data point from testing dataset; $f(x)_i$ represents the prediction result of model i.

Adaboost is adaptive in the sense that subsequent weak learners are tweaked in favor of the instances misclassified by previous classifiers. By racing the weights of misclassified training data, those data are more likely to be classified correctly at the next classifier. Assuming that m of the n samples were classified incorrectly, the initial weight of each sample is 1/n, and n means the number of samples. Therefore, we have to calculate the error rate ε by error expression. The formula is as follows:

$$\varepsilon_t = \frac{\sum_{i=1}^{m} \omega_i}{\sum_{j=1}^{n} \omega_j}$$

ε_t means the error rate of the t-th classifier. ω means the weight of each sample.

After calculating the error rate, we can calculate the weight of this classifier for this training result. The formula is as follows:

$$\alpha_t = \frac{1}{2}\ln(\frac{1-\varepsilon_t}{\varepsilon_t})$$

α_t means the weight of the t-th classifier. Finally, according to the calculated error rate, the weight of the sample with the wrong classification is increased by the following formula.

$$D_{t+1}(i) = \begin{cases} D_t(i)\,, & if y_i = f_{prediction_t}(x_i) \\ D_t(i)\dfrac{1-\varepsilon_t}{\varepsilon_t}\,, & if y_i \neq f_{prediction_t}(x_i) \end{cases}$$

D_{t+1} represents the training sample of the next classifier; $f_{prediction_t}(x_i)$ means the predicted value for i-th sample x_i, and y_i means the actual value for i-th sample.

Therefore, we know that Adaboost is sensitive to noisy data and outliers. In some problems, it can be less susceptible to the overfitting problem than other learning algorithms.

Further, RF is mainly composed of the bagging and CART algorithms; training with multiple trees can enhance the diversity of the traditional decision tree method. Finally, XGBoost is an evolution of gradient tree boosting. It views the correctness of each decision tree in an RF as a probability; therefore, each tree is not necessarily correct.

Establishing a decision by CART's algorithm and use the Gini Index as an indicator to build a decision tree. Assuming that dataset D contains n categories, define the Gini Index of data set D as following formula:

$$Gini(D) = 1 - \sum_{j=1}^{n} p_j^2$$

p_j is the probability of belonging to category j in dataset D. The A attribute is used to cut out the two subsets D1 and D2 from the dataset D, it is defined as formula:

$$Gini_A(D) = \frac{|D_1|}{|D|}Gini(D_1) + \frac{|D_2|}{|D|}Gini(D_2)$$

During the training process, it is determined that the attribute is not suitable for classification according to the degree of noise reduction, as shown in formula:

$$\Delta Gini(A) = Gini(D) - Gini_A(D)$$

Training with multiple trees can enhance the diversity of the traditional decision tree method. Finally, XGBoost is an evolution of gradient tree boosting [35]. It views the correctness of each decision tree in an RF as a probability; therefore, the branches of each tree will be decided whether to be cut through the following formula:

$$L_{split} = \frac{1}{2}\left[\frac{(\sum_{i\epsilon I_L} g_i)^2}{\sum_{i\epsilon I_L} h_i + \lambda} + \frac{(\sum_{i\epsilon I_R} g_i)^2}{\sum_{i\epsilon I_R} h_i + \lambda} - \frac{(\sum_{i\epsilon I} g_i)^2}{\sum_{i\epsilon I} h_i + \lambda}\right] - \gamma$$

$$\begin{cases} g_i = \partial_{y^{(t-1)}_{predict_i}}(y_i, y^{(t-1)}_{predict_i}) \\ h_i = \partial^2_{y^{(t-1)}_{predict_i}}(y_i, y^{(t-1)}_{predict_i}) \end{cases}$$

Assume that I_L and I_R are the datasets of the left and right nodes after the separation. λ is the hyperparameter for this formula. g_i and h_i were the first derivative and

secondary derivative of the output result under the branch, which can find the weight for the detection model with the minimum error. Assuming $I = I_L \bigcup I_R$, the loss reduction after splitting is as shown in above formula.

4. System Performance Analysis

This section presents the verification of the accuracy and the stability of each module proposed in this paper. In the experiment, each model was analyzed separately, and the most suitable detection model algorithm was thus found. Finally, the performance analysis of the intelligent learning architecture will be discussed in the following section.

4.1 Experimental Environment

In this study, we crawled 3850 legitimate sites and 24471 phishing sites from PhishTank, Common Crawl, and Alex and extracted 28 features by using the feature extraction module. The machine learning algorithms used in this study were run on the TensorFlow architecture provided by Google. The detailed hardware configuration is shown in Table 2, and the software configuration of the analysis architecture is shown in Table 3.

4.2 Performance Analysis

This section describes the results of the proposed module implemented in the experimental environment and a detailed analysis of the results of each training module. In order to make the topic of this study more similar to a real-life case, the proportion of data between the phishing websites and the legitimate websites was unbalanced. In the early stage of this study, to explore the diversity of phishing websites, the amount of data of the phishing websites was larger than that of the legitimate websites. Therefore, if these data were directly sent to the machine learning algorithm, a model offset might occur. To solve this problem, the Synthetic Minority Over-sampling TEchnique (SMOTE) method was used to generate the corresponding to the minority class instances in the early stage of this study.

This study evaluated the performance of the XGBoost algorithm with imbalanced learning by comparing it with Extreme Learning Machine (ELM), SVM, LR, KNN, Linear Combination Extreme Learning Machine (LC-ELM), decision tree of C4.5, RF, and XGBoost alone. The optimal number of hidden nodes was the only parameter that could be determined for single ELM; therefore, we set the number of hidden nodes to 5-20 and constructed 100 simulations for each number by using a 10-fold cross-validation to determine this parameter. The results revealed that 10 was the optimal number of hidden nodes. In addition, the number of weaker classifiers were set to 5, 10, 15, 20, 25, 30, 35, 40, 45, and 50 for the bagging algorithm and implemented 10 simulations for each given estimator size. The testing accuracies for each estimator size are shown in Fig. 4.

In this study, five classification algorithms were combined into three weak classifiers. The weak classifier algorithm included decision tree C4.5, K-nearest neighbor, logistic regression, extreme learning machine, and RBF. The special symbols for each classifier are listed in Table 4.

The voting algorithm identified the results of the three weak classifiers by using a voting mechanism. Finally, the identification result was the category with the highest number of votes. This section describes the difference and effectiveness between various combinations for the voting mechanism. In this experiment, we used a 10-fold cross-validation method to train the model and then, to evaluate the stability and the accuracy of the models. The results of the performance analysis are shown in Fig. 5.

Furthermore, Fig. 6 shows the analysis result after the 10-fold cross-validation. It mainly sums up the confusion matrix of each fold and calculates the average value. Finally, precision, recall, accuracy, and standard deviation are calculated separately.

According to the analysis results of bagging as shown

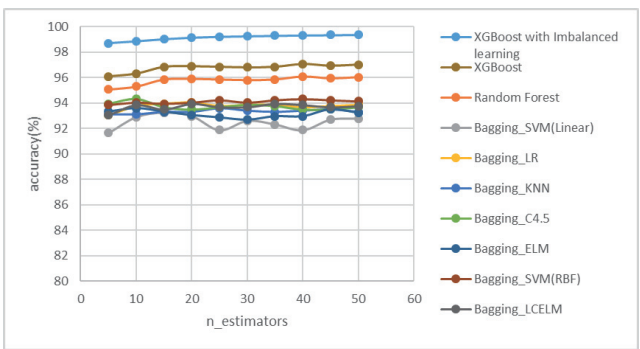

Fig. 4 Accuracy of bagging, XGBoost with imbalanced learning, and random forest

Table 2 Hardware configuration

Symbol	Feature
CPU	Intel(R)Xeon(R)CPUE5-2620 v4
Memory	32GB
Linux	Ubuntu 16.04
GPU	3*MSI GTX 1080Ti 11G
HDD	2TB
SSD	256GB

Table 3 Software configuration

Symbol	Feature
cuDNN	V6.0
CUDA	V8.0
TensorFlow	V1.1

Table 4 Special symbols for each classifier

Symbol	Mean
c	Decision Tree C4.5
k	K Nearest Neighbor
l	Logistic Regression
e	Extreme Learning Machine
s	Support Vector Machine (RBF)

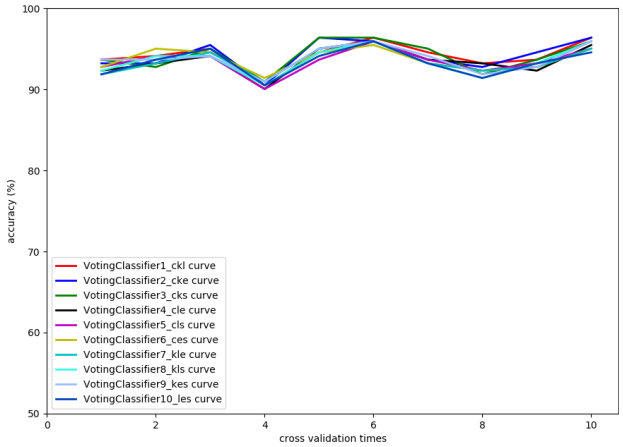

Fig. 5 Accuracy of voting algorithm with different combinations of classifiers

	Voting Classifier ckl	Voting Classifier cke	Voting Classifier cks	Voting Classifier cle	Voting Classifier cls	Voting Classifier ces	Voting Classifier kle	Voting Classifier kls	Voting Classifier kes	Voting Classifier les
Precision	95.37%	95.37%	95.32%	95.28%	96.15%	96.19%	95.28%	96.19%	96.19%	96.15%
Recall	93.63%	93.63%	93.57%	92.66%	91.74%	91.81%	92.66%	91.81%	91.81%	91.74%
F1 score	94.08%	94.08%	94.03%	93.34%	93.23%	93.5%	93.31%	93.3%	93.58%	93.09%
Accuracy	94.2%	94.2%	94.16%	93.52%	93.48%	93.71%	93.48%	93.52%	93.8%	93.34%
Standard deviation	1.6%	1.79%	1.8%	1.65%	1.51%	1.31%	1.51%	1.69%	1.54%	1.6%
Training time	1h 15m	1h 38m	1h 5m	1h 18m	1h 29m	1h 35m	1h 25m	1h 28m	1h 36m	1h 40m

Fig. 6 Analysis of voting algorithm with different combinations of classifiers after 10-fold cross-validation

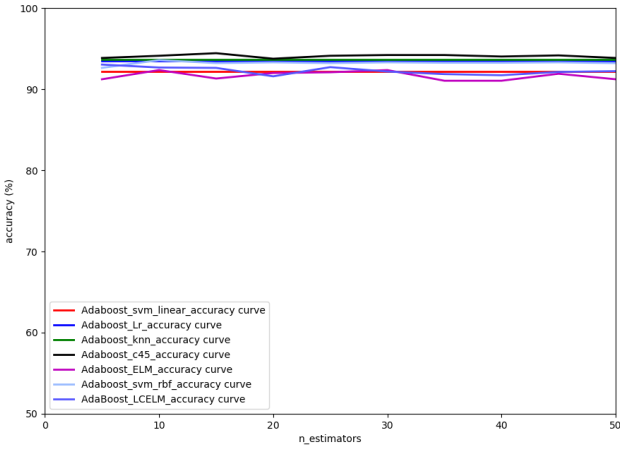

Fig. 7 Accuracy of adaboost algorithm with different estimators

	Adaboost SVM_Linear	Adaboost LR	Adaboost KNN	Adaboost C4.5	Adaboost ELM	Adaboost SVM_RBF	Adaboost LCELM
Precision	94.33%	96.15%	93.63%	94.44%	92.52%	94.44%	89.74%
Recall	90.9%	90.9%	94.49%	93.57%	90.82%	92.72%	96.33%
F1 score	92.02%	93.27%	93.58%	93.81%	91.11%	93.14%	92.41%
Accuracy	92.21%	93.48%	93.61%	93.84%	91.22%	93.21%	92.21%
Standard deviation	1.44%	1.21%	1.61%	1.47%	2.27%	1.51%	1.83%
Training time	2h 50m	2h 45m	1h 15m	1h 28m	2h 15m	2h 5m	2h 38m

Fig. 8 Analysis of adaboost algorithm with different estimators after 10-fold cross-validation

in Fig. 4, the performance of the RBF kernel for the SVM algorithms exceeded SVM algorithm with other kernels. Further, the algorithms with tree-structure got good classification result. We can conclude that the features used in this study were non-linear, which is not suitable for linear classification algorithms. Also, when the voting algorithm contained the SVM RBF kernel classifier or the tree structure classifier, the accuracy was fairly good. As shown in Fig. 6, the accuracy of the voting model trained by using different combinations of the classifiers was between 93.34% and 94.2%, and the standard deviation was between 1.31% and 1.8%.

Adaboost is one of the mechanisms of ensemble learning. The concept is also to train multiple weak classifiers of the same algorithm and combine these classifiers into a strong classifier. However, this algorithm is slightly different from the previous two methods. This method mainly enlarges the wrong data points that were identified in the previous classifier and hopes that the next classifier can distinguish them correctly during the classification process. The weak classifier algorithm includes decision tree C4.5, K-nearest neighbor, logistic regression, extreme learning machine, support vector machine, and linear-combination extreme learning machine. In this experiment, we used the 10-fold cross-validation method for the analysis and the evaluation. The analysis results are shown in Fig. 7.

In this experiment, we trained 5-50 classifiers for different weak classifier algorithms by using the Adaboost method and evaluated the model through a 10-fold cross-validation. Finally, the average of the confusion matrix for each fold was calculated, and precision, recall, accuracy, and standard deviation were calculated separately for the best number of classifiers. The results are shown in Fig. 8.

According to Fig. 8, the accuracy of the strong classifiers trained by the Adaboost mechanism was approximately between 91.22% and 93.84%, and the standard deviation was between 1.21% and 2.27%.

According to the analysis results of the three above-mentioned methods, the information of the 28 features used in this study was more suitable for the tree structure algorithm. According to the bagging mechanism, the accuracy of the algorithm of decision tree C4.5 was higher than that of the other algorithms, and both Adaboost and voting had the same benefits. However, the optimization method of the feed-forward neural network was relatively simplified and thus not suitable for this research. For example, ELM is much faster to train the detection model, but cannot encode more than one layer of abstraction. This method may do fine on relatively simple things like MNIST, but it's not going to be able to recognize complex things, without having an astronomically sized hidden layer. In this study, we have drawn a three-dimensional data distribution to analyze the distribution of two categories of data. However, we found the two categories of data points showed a serious overlap from the three-dimensional data distribution, as shown in Fig. 9. Red dots in Fig. 9 are data points of phishing websites, green triangles are data points of normal websites. The three-axis x, y, and z are the top 3 important features evaluated by ANOVA.

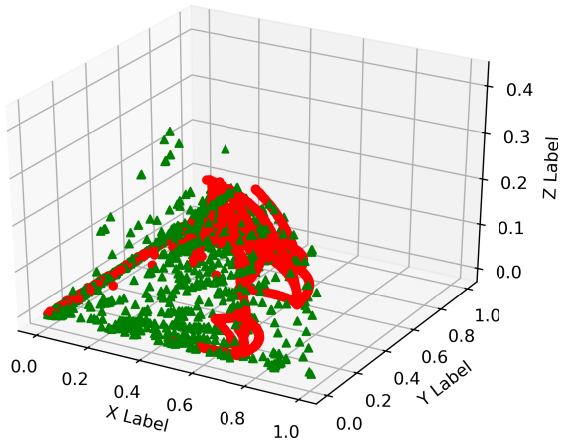

Fig. 9 Three-dimensional data distribution

We know that SVM with linear kernel or Logistic Regression classifies data points via linear equations, so when data overlaps in feature space, it is difficult to classify good results. In this study, our data distribution has a serious overlap as shown in Fig. 9, so we know the classification of linear methods, such as logistic regression and SVM algorithms with linear kernel, was not suitable for this study.

According to the above analysis, in this study, this study analyzed and evaluated the tree-structured classification algorithm in the later stage of this research. The algorithm included RF and XGBoost.

5. Conclusions and Future Work

Although various approaches have been used to mitigate phishing attacks, such attack techniques remain a serious Internet security problem. In this paper, we presented novel architecture using the techniques of machine learning, artificial intelligence, and deep learning to complete phishing detection in a software-defined networking environment. In the training process, we extracted 28 features to teach the model and tried a number of machine learning algorithms including bagging, RF, and XGBoost. We found that using the final extracted 28 features from feature evaluation module to train the model could achieve higher accuracy through the XGBoost algorithm with imbalanced learning. In addition to filtering out unrelated feature information from feature extraction module, the SMOTE method also improves the detection coverage of the model, making the detection model more robust and accurate. In this study, we used the SMOTE method to generate some uncovered or uncollected data, which can increase the coverage of this model detection. Experimental results showed that our proposed learning architecture with SMOTE method performed the best with respect to accuracy, precision, and recall. From this point of accuracy and feasibility, this system can be implemented in a network environment. The future development of the proposed learning algorithm architecture will include the application of deep learning [1], [12], [13] to update the detection model and its application to websites for testing. Since this research is still in the research stage, the volume between the testing traffic and real-word traffic have slightly differences. Will focus on exploring the prediction time and processing speed in the future. Further, we will cooperate with relevant research units or companies to provide higher network security for the users.

Acknowledgments

The authors would like to thank the Institute of Information Industry, Taiwan for financially supporting this research under Contract No. 105-2221-E-011 -078 -MY3 and 105-2221-E-011 -076 -MY3.

References

[1] H. Shirazi, K. Haefner, and I. Ray, "Fresh-Phish: A Framework for Auto-Detection of Phishing Websites," Proc. IEEE International Conference on Information Reuse and Integration (IRI), The San Diego, CA, USA, pp.137–143, Aug. 2017.

[2] A.J. Park, R.N. Quadari, and H.H. Tsang, "Phishing Website Detection Framework through Web Scraping and Data Mining," Proc. 8th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), The Vancouver, BC, Canada, pp.680–684, Nov. 2017.

[3] APWG, Unifying the global response to cybercrime, https://www.antiphishing.org/

[4] Z.A. Wen, Y. Li, R. Wade, J. Huang, and A. Wang, "What.Hack: Learn Phishing Email Defence the Fun Way," Proc. CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '17), The Denver, Co, USA, pp.234–237, May 2017.

[5] J. Mao, W. Tian, P. Li, T. Wei, and Z. Liang, "Phishing-Alarm: Robust and Efficient Phishing Detection via Page Component Similarity," Proc. IEEE, vol.5, pp.17020–17030, Aug. 2017.

[6] S. Afroz and R. Greenstadt, "PhishZoo: Detecting Phishing Websites by Looking at Them," Proc. IEEE Fifth International Conference on Semantic Computing, The Palo Alto, CA, USA, pp.368–375, Oct. 2011.

[7] J. Jung and E. Sit, "An Empirical Study of Spam Traffic and the Use of Dns Black Lists," Proc. 4th ACM SIGCOMM Conference on Internet measurement, The Taormina, Sicily, Italy, pp.370–375, Oct. 2004.

[8] L. Wenyin, N. Fang, X. Quan, B. Qiu, and G. Liu, "Discovering Phishing Target Based on Semantic Link Network," Proc. Future Generation Computer Systems, vol.26, no.3, pp.381–388, March 2010.

[9] How Quad9 works, Internet security & privacy in a few easy steps, https://www.quad9.net/

[10] S. Haruta, H. Asahina, and I. Sasase, "Visual Similarity-Based Phishing Detection Scheme Using Image and CSS with Target Website Finder," Proc. IEEE Global Communications Conference, The Singapore, pp.1–6, Dec. 2017.

[11] A. Subasi, E. Molah, F. Almkallawi, and T.J. Chaudhery, "Intelligent Phishing Website Detection Using Random Forest Classifier," Proc. International Conference on Electrical and Computing Technologies and Applications (ICECTA), The Ras AI Khaimah, United Arab Emirates, pp.1–5, Nov. 2017.

[12] A.C. Bahnsen, E.C. Bohorquez, S. Villegas, J. Vargas, and F.A. González, "Classifying Phishing URLs Using Recurrent Neural Networks," Proc. APWG Symposium on Electronic Crime Research (eCrime), The Scottsdale, AZ, USA, pp.1–8, April 2017.

[13] Z. Hu, R. Chiong, I. Pranata, W. Susilo, and Y. Bao, "Identifying Malicious Web Domains Using Machine Learning Techniques with Online Credibility and Performance Data," Proc. IEEE Congress

on Evolutionary Computation (CEC), The Vancouver, BC, Canada, pp.5186–5194, July 2016.
[14] S. Tayeb, M. Pirouz, J. Sun, K. Hall, A. Chang, J. Li, C. Song, A. Chauhan, M. Ferra, T. Sager, J. Zhan, and S. Latifi, "Toward Predicting Medical Conditions Using K-Nearest Neighbors," Proc. IEEE International Conference on Big Data (Big Data), The Boston, MA, USA, pp.3897–3903, Dec. 2017.
[15] K. Huang, H. Jiang, and X.-Y. Zhang, "Field Support Vector Machines," Proc. IEEE Transactions on Emerging Topics in Computational Intelligence, vol.1, no.6, pp.454–463, Dec. 2017.
[16] C.-C. Chang and C.-J. Lin, "LIBSVM: A Library for Support Vector Machines," Proc. ACM transactions on intelligent systems and technology (TIST), vol.2, no.3, pp.27:1–27:27, April 2011.
[17] D.R. Ibrahim and A.H. Hadi, "Phishing Websites Prediction Using Classification Techniques," Proc. International Conference on New Trends in Computing Sciences (ICTCS), The Amman, Jordan, pp.133–137, Oct. 2017.
[18] L. Rokach and O. Maimon, "Top-Down Induction of Decision Trees Classifiers - a Survey," Proc. IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), vol.35, no.4, pp.476–487, Nov. 2005.
[19] M. Denil, D. Matheson, and N.D. Freitas, "Narrowing the gap: Random forests in theory and in practice," Proc. International Conference on Machine Learning, The Beijing, China, pp.1–12, Oct. 2014.
[20] UCI Machine Learning Repository, "UCI machine learning repository: Phishing websites data set," https://archive.ics.uci.edu/ml/datasets/phishing+websites
[21] M.A.U.H. Tahir, S. Asghar, A. Zafar, and S. Gillani, "A Hybrid Model to Detect Phishing-Sites Using Supervised Learning Algorithms," Proc. International Conference on Computational Science and Computational Intelligence (CSCI), The Las Vegas, NV, USA, pp.1126–1133, Dec. 2016.
[22] R. Verma and A. Das, "What's in a URL: Fast Feature Extraction and Malicious URL Detection," Proc. 3rd ACM on International Workshop on Security and Privacy Analytics (IWSPA '17), The Scottsdale, Arizona, USA, pp.55–63, March 2017.
[23] Netscape Communications Corporation, Open directory RDF dump, http://rdf.dmoz.org/
[24] H. Sutrisno and S. Halim, "Credit Scoring Refinement Using Optimized Logistic Regression," Proc. International Conference on Soft Computing, Intelligent System and Information Technology (ICSIIT), The Denpasar, Indonesia, pp.26–31, Sept. 2017.
[25] R. Verma and K. Dyer, "On the Character of Phishing URLs: Accurate and Robust Statistical Learning Classifiers," Proc. 5th ACM Conference on Data and Application Security and Privacy, The San Antonio, Texas, USA, pp.111–122, March 2015.
[26] A. Saberi, M. Vahidi, and B.M. Bidgoli, "Learn to Detect Phishing Scams Using Learning and Ensemble Methods," Proc. 2007 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology - Workshops, The Silicon Valley, CA, pp.311–314, Nov. 2007.
[27] J.-H. Li and S.-D. Wang, "PhishBox: An Approach for Phishing Validation and Detection," Proc. 2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech), The Orlando, FL, pp.557–564, Nov. 2017.
[28] S.Y. Bhat, M. Abulaish, and A.A. Mirza, "Spammer Classification Using Ensemble Methods over Structural Social Network Features," Proc. 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT), The Warsaw, pp.454–458, Aug. 2014.
[29] X.-P. Tian, G.-G. Geng and H.-T. Li, "A framework for multi-features based Web harmful information identification," Proc. 2010 International Conference on Computer Application and System Modeling (ICCASM 2010), The Taiyuan, pp.V11-614–V11-618, Oct. 2010.
[30] H.V. Nath and B.M. Mehtre, "Ensemble learning for detection of malicious content embedded in PDF documents," Proc. 2015 IEEE International Conference on Signal Processing, Informatics, Communication and Energy Systems (SPICES), The Kozhikode, pp.1–5, Feb. 2015.
[31] M.H. Kamarudin, C. Maple, T. Watson, and N.S. Safa, "A LogitBoost-Based Algorithm for Detecting Known and Unknown Web Attacks," Proc. IEEE Access, vol.5, pp.26190–26200, Nov. 2017.
[32] Z.H. Zhou, "Ensemble Methods: Foundations and Algorithms," Chapman & Hall/CRC, USA, pp.1–234, May 2012.
[33] V. Ramanathan and H. Wechsler, "Phishing Website Detection Using Latent Dirichlet Allocation and AdaBoost," Proceedings of the IEEE International Conference on Intelligence and Security Informatics, Arlington, USA, pp.102–107, June 2012.
[34] Z. Wei, Q. Jiang, L. Chen, and C. Li, "Two-Stage ELM for Phishing Web Pages Detection using Hybrid Features," Proc. World Wide Web, vol.20, no.4, pp.797–813, July 2017.
[35] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," Proc. 22nd ACM Sigkdd International Conference on Knowledge Discovery and Data Mining, The San Francisco, California, USA, pp.785–794, March 2016.
[36] J. Tang, S. Alelyani, and H. Liu, "Feature selection for classification: A review," Data Classification: Algorithms and Applications, CRC Press, 2013.
[37] H. He, Y. Bai, E.A. Garcia, and S. Li, "ADASYN: Adaptive Synthetic Sampling Approach for Imbalanced Learning," Proc. IEEE International Joint Conference on Neural Networks, The Hong Kong, China, pp.1322–1328, June 2008.

Yu-Hung Chen was born in Taiwan in 1994. He received the M.S. degree in electrical engineering from National Taiwan University of Science and Technology, Taipei, Taiwan, in 2018. His main research interests include Software-Defined Network, Network Function Virtualization, Network Security, Phishing, Artificial Intelligence and Deep Learning.

Jiann-Liang Chen was born in Taiwan in 1963. He received the Ph.D. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, in 1989. Since 2008, he has been with the Department of Electrical Engineering, National Taiwan University of Science and Technology, where he is currently a Professor. His current research interests are directed at cellular mobility management and personal communication systems.