Journal of Physics: Conference Series

PAPER • OPEN ACCESS

Comparison of simple feedforward neural network, recurrent neural network and ensemble neural networks in phishing detection

To cite this article: Gan Kim Soon *et al* 2020 *J. Phys.: Conf. Ser.* **1502** 012033

Comparison of simple feedforward neural network, recurrent neural network and ensemble neural networks in phishing detection

Gan Kim Soon[1], Chin Kim On[1], Nordaliela Mohd Rusli[1], Tan Soo Fun[1], Rayner Alfred[1], Tan Tse Guan[2]

[1] Knowledge Technology Research Unit, Faculty of Computing and Informatics, Universiti Malaysia Sabah, Jalan UMS, 88400 Kota Kinabalu, Sabah.
[2] Faculty of Creative Technology & Heritage, Universiti Malaysia Kelantan, Kelantan, Malaysia

kimonchin@ums.edu.my

Abstract. The internet has been one of the greatest advancements in technologies. It has brought many advantages to today's society in many domains such as e-commerce, entertainment and supply chain, amongst others. However, it is also a double-edged sword which has brought many threats to the computer systems and devices known as cyber-attack, and one of these threats would be phishing attack. A phishing attack is where the scammer tries to impose or clone the legitimate email or website in order to deceive the victim to key in their personal information such as username and password. Phishing attack has been one of the most common attacks that happens every day on the Internet especially through email. Many methods have been devised to encounter phishing attack, and one of approaches is through training and monitoring team. These manual approaches, however, are user's experience-dependent and cost-inefficient. Therefore, many have adopted AI approach instead to detect phishing attack. This paper is one of the many efforts to detect the phishing attack through email by adopting AI method. The objective of this paper is to investigate the performance of feedforward neural network, recurrent neural network and ensemble neural network in phishing email detection. The result of this comparison is empirically evaluated.

1. Introduction

Internet is one of the greatest inventions of the centuries that has brought many benefits to the world of technologies affecting the process of many domains [1]. Therefore, many have appreciated the existence of the Internet for the efficiency and effectiveness that it has brought about [2]. Nonetheless, everything has its pros and cons, and this applies to Internet as well. One of the drawbacks of the Internet is that it introduces a new kind of threat which is commonly referred to as cyber-attack [3]. Cyber-attack is any kind of malicious activity that is done through the Internet.

The phishing attack is one of the most common cyber-attacks anyone with access to the Internet would be vulnerable to. The phishing attack is where the scammer tries to deceive the victim through fake email or clone of legitimate website in order to obtain their personal credential information such as bank account details and other information [4,5]. The phishing attack is based on the social engineering attack method to play around with the victim's emotion and psychology in the attempt to deceive their credential information.

Many prevention approaches have been devised to encounter the phishing attack. One of the approaches is through the manual approach where it educates the user in distinguishing between phishing and legitimate emails through training campaign. Another method is by establishing the monitoring team where only large organisation has the allocation for this kind of resources. These types of manual approaches are user's experience-dependent and cost-inefficient. Moreover, it is also prone to human error since the approaches are user's experience-dependent.

Other approaches to detect phishing attack would be using the software system, and one of the methods in software detection is called blacklist method [6]. The blacklist method contains the list of phishing URLs which will detect the email whether it contains any of the blacklisted URLs in the list. The blacklist method is compensated with a whitelist that contains legitimate URLs to reduce the false-positive rate during phishing detection. This method, however, does not protect against zero-hour phishing, i.e. it cannot detect new phishing URL which does not exist in the blacklist. Good examples of blacklist method include Google Safe Browsing API, DNS-based blacklist, PhishNet and others. Another method in software detection is known as heuristic method. The heuristic method detects phishing attack by using data or characteristic of existing phishing attack. Therefore, this method is effective against zero-hour phishing since it detects phishing by utilizing the data and characteristic. The drawback of this approach, nevertheless, is that it has a high false-positive rate. Noteworthy examples of this method include SpoofGuard and PhishGuard, amongst others. Visual similarity is another method in software system approach where the detection is based on the visual similarity of the website [7]. The visual similarity method compares the appearance of the website mainly based on CSS of the website. However, this method is time and cost consuming. An example of this method is BaitAlarm. The software system approach is undoubtedly able to perform better in phishing attack detection compared to manual approaches, but with the continuously evolving phishing attack which could be hard to catch-up, a more intelligent approach is required.

This intelligent approach to detect a phishing attack is by adopting AI [8], and the branch of AI that is most commonly used to detect phishing attack is known as machine learning. Machine learning is a continuous learning algorithm from data/experience to improve the performance of computer systems for some tasks [9]. Artificial Neural Network (ANN) is amongst one of popular algorithm in machine learning that has shown to perform well in many domains including the phishing attack detection [10,11]. Therefore, the main objective of this paper is to investigate the performance of the three different ANN architectures, namely Feedforward Neural Network (FFNN), Recurrent Neural Network (RNN) and Ensemble Neural Network (ENN) in phishing email.

The next section briefly describes some of the related work. The third section explains the methodology of the experiment whilst the fourth section details out the experiment setup of this paper, follow by the result and discussion in the fifth section. The conclusion to conclude the finding of this paper is presented in the final section.

2. Related work

Machine learning has been widely applied in the study of phishing detection. Different machine learning algorithms have shown to perform well in the detection of the phishing email. This section discusses some of the studies related to this content.

Huang et al. used SVM to detect phishing URLs [12]. In his paper, Huang utilized 23 features to detect the phishing URLs. There are four structure features, nine lexical features and ten brand name features in those 23 features. Huang achieved a high accuracy of 99.0% in this study. Akinyelu and Adewumi used random forest to classify the phishing emails [13]. The number of decision trees depends on the subset of dataset being drawn. A total of 15 features are used in phishing detection experiment which yielded a significantly high accuracy of 99.7% in the study by using the specified dataset. Badadhe et al. used a two-step classification method for detecting phishing attack [14]. The first step is to detect the phishing URLs by using the K-means classifiers with URL features. In the second step, if certain threshold of the phishing detection is met, Naïves Bayes classifier is used to detect the webpage based on a set of webpage features to classify the phishing attack. This study achieves a true positive rate of 97.08% and a false positive rate of 1.15%.

Zhang and Yuan used multilayer FFNN to detect and filter phishing emails [15]. Different experimental setups, such as different transfer function and a different number of hidden neurons have been tested on phishing detection. The experimental result has shown that setup with sigmoid activation function and eight hidden neurons achieved a better result compared to other settings. In the experiment, FFNN achieved an accuracy of 95%.

Mohammad et al. used FFNN in predicting phishing websites [16]. The experimental setup varies in the number of hidden neurons and a number of hidden layers. The activation function used is sigmoid function, and 0.7 learning rate is used in the experiment. The experimental result concluded that one hidden layer is sufficient to achieve good performance result in the prediction problem.

FFNN is also used for phishing email detection in the study carried out by Jameel and George [17]. A different number of hidden neurons setup is tested on the domain problem in order to find the better-hidden number neurons of the experimental setting. The study achieved a 98.72% of accuracy and 0.005 false-positive rates.

Kathirvalavakumar et al. used multilayer FFNN for detecting phishing emails [18]. In the study, the pruning weight elimination algorithm is used to reduce the number of hidden neurons and features. The pruning algorithm has successfully reduced the number of features from 19 to 9 and reducing hidden neurons from 15 to 3 hidden neurons but still achieves a better accuracy of 99.9%.

Swetha Babu and Radha conducted a study on detecting phishing website link using NNNs and Firefly algorithm [19]. The Firefly algorithm adjusted the weights based on an objective function that helps to decrease the false error rate (FER). The experiment has shown to perform better than Phishield by achieving a lower false-positive rate of 0.05% and better accuracy rate of 99.52%.

Bahnsen et al. conducted a study on detecting phishing URL by using long short-term memory (LSTM) [20]. In this study, the LSTM is compared with random forest method. The result showed that LSTM performs better by achieving higher accuracy of 5% greater than random forest.

Based on the reviews, NN has shown to perform well in phishing detection. As a result, it is worth to perform a study in comparing the performance of different NN in detecting phishing email.

3. Methodology

Aforementioned, ANN is used in this study to investigate its performance in detecting phishing email. ANN is one of the machine learning algorithms that is inspired by biological NNs. ANN is made up of large collection of finest components of artificial neurons that mimics biological neuron. The artificial neuron is associated with weight and interacts with other artificial neurons to build a network. ANN is composed of different type of layers which are the input layer, hidden layer and output layer. The input layer receives the external input and pass to the hidden layer. The hidden layer then performs some calculation and pass the result to output layer. The output layer delivers the output of the result.

In this experiment, three types of NNs are used to conduct the experiment which is the FFNN, RNN and ENN [21]. FFNN is the NN where the signal only flows in one direction. There is no loop and feedback signal from the previous layer in FFNN. FFNN is fast in computation.

RNN is different from FFNN, where the signal flows in both directions. The RNN consists of loop or feedback from the previous layer. The feedback signal of the previous layer or loop simulates the internal memory that stores the previous state of the signal which is important to time series data.

ENN is the aggregation of multiple NNs trained on the same tasks to produce a better result. Single NN has limited capacity in exploring the search space of a problem domain. Therefore, aggregating different NNs trained on same tasks will be able to generalize the search space cover and hence, producing better result. There are different ways of aggregating the output of ENN which include averaging, maximum value, minimum value, majority votes and others [22].

In order to carry out the experiment, the first step is to determine the dataset for the experiment and in this case, the email dataset. Afterwards, some pre-processing process is required to transform the raw data into the input to be fed into the NN. This includes the feature extraction process and the binary encoding of the extracted feature. After obtaining the pre-process dataset, the dataset is partitioned into training and testing set. The training set is then feed to train neural network whilst the testing set is used for NN performance.

A GUI interface is developed to facilitate the experiment and the output of the classification result. The GUI provides the function to upload the input emails in .eml format to detect whether it is a phishing email.

4. Experimental setup

This section describes the experimental setup used for this study. The dataset used in this experiment is CSDMC2010 SPAM. This dataset consists of 4,327 messages which include 2,949 ham emails and 1,379 phishing emails.

4.1. Features extraction

The dataset is then pro-process through several processes. Each email data is parsed with printable ASCII or HTML. Then, the parsed data is further binary encode with value 1 if the feature exists or value 0 if the feature does not exist. The desired output of email data is also encoded, 0 for ham email and 1 for phishing email. The list of the features for classification is adopted from [17]. Table 1 listed the features used for classification in this study. All these features encoded is saved in a CSV file to feed into the neural network.

Table 1. List of features

	Features
1	HTML code embedded within the email
2	pictures used as link is more than two
3	number of different domains is more than
4	number of embedded links in the email is more than three
5	the message has HTML code included <form> tag
6	“From” domain is not equal to “ReplyTo” domain
7	the message size less than 25KB
8	the message has javascript code
9	nonmatching between target and appeared text of URLs in the email
10	nonmatching between target and appeared text of URLs in the email
11	message has one of the words “click here”, “click” or “here” or “login” in text part of links
12	number of dots in the domain is more than 3
13	the message has @ symbol in URL
14	the URL in the message has a port value other than 80 or 443
15	the domain of any embedded links in the HTML body is not equal to the sender’s domain
16	the binary vector if https:// is used instead of http://
17	there is a URL in the email with hexadecimal numeric representation
18	the email is classified as spam by SpamAssassin3.2.3.5 Win32

4.2. Parameter tuning

In this study, a set of preliminary experiments are carried out to fine-tune two parameters in order to achieve better experimental result. The fine-tuning parameters are the number of neurons and learning rate. In this experiment, the test number of hidden neurons is from 1 to 18. The learning rate can lower the errors, hence achieving better experimental result [24]. The range of learning rate tested in this experiment is ranged between 0.001 to 0.1.

In these preliminary experiments, only a subset of data is selected. 500 samples are selected from the dataset and then divided to 70% (350 messages) for training and 30% (150 messages) for testing. A set of fixed value is used for the other parameters in the experimental setting. Table 2 listed all the fixed value for other experimental parameters.

Table 2. Neural networks parameters values for preliminary testing.

Number of neurons in input layer	18
Number of neurons in hidden layer	(input neurons + output neurons) / 2 ≈ 10
Number of neurons in output layer	1
Transfer function	Sigmoid function
Performance function	Cross entropy (CE)
Number of epoch	500
Learning rate	0.1

Table 3. Parameter range of the parameter tuning.

Number of hidden neurons	1 - 18
Learning rate	0.001 – 0.1

4.3. Final experiments

The experimental result from the parameter tuning is adopted in this final experiment. In this experiment, the whole dataset is adopted instead of a subset. The dataset is divided to 70% (3,029 messages) for training data and 30% (1,298 messages) for testing data. The parameter setting is listed in the following section- the result and discussion section. Furthermore, feature analysis is carried out to determine the important and less important features for the features adopted for this classification experiment.

5. Result and discussion

This section discusses the experimental result for the different experimental setting discussed in the previous section.

5.1. Parameter tuning result

Table 4 shows the experiment results for different hidden neuron for different neural network architectures. Based on the result, the highest average accuracy achieved by FFNN is 96.444% with two neurons, RNN is 97.333% with 12 neurons and for ENN is 96.667% with two neurons. RNN required more hidden neurons due to the complexity required to store internal memory with context units required.

Table 4. Result for different number of hidden neurons for different neural network architecture.

	Average Accuracy (%)		
Number of hidden neurons	FFNN	RNN	ENN
1	96.000	96.444	96.000
2	**96.444**	96.889	**96.667**
3	96.444	96.667	96.667
4	96.000	96.889	96.222
5	96.222	96.667	96.444
6	96.000	96.889	96.222
7	96.000	96.889	96.444
8	96.000	97.111	96.222
9	96.000	96.889	96.000
10	95.778	96.889	96.222
11	96.000	97.111	96.667
12	96.000	**97.333**	96.444
13	96.000	97.111	96.444
14	95.778	96.889	96.667
15	96.000	96.667	96.222
16	96.222	97.111	96.444
17	96.000	96.889	96.667
18	96.000	97.111	96.444

Table 5 shows the experiment result for different learning rates for different neural network architectures. Based on the table result, the highest average accuracy for FFNN is 97.111% with 0.002 learning rate, RNN achieved an accuracy of 97.111% with 0.002 learning rate and ENN achieved an accuracy of 96.889% with 0.001 learning rate. As mentioned, the higher accuracy is achieved by lower error rate which is the same as the finding in [24].

Table 5. Result for different learning rate for different neural network architecture.

Learning Rate	Average Accuracy (%)		
	FFNN	RNN	ENN
0.001	96.444	96.889	**96.889**
0.002	**97.111**	**97.111**	96.889
0.003	96.444	97.111	96.667
0.004	96.000	96.444	96.000
0.005	96.000	96.000	96.000
0.006	95.778	96.222	96.000
0.007	95.778	96.222	96.000
0.008	95.778	96.667	96.000
0.009	95.778	96.889	96.000
0.01	95.778	96.889	96.222
0.02	95.778	96.667	96.444
0.03	96.000	96.222	96.222
0.04	96.444	96.000	96.222
0.05	96.444	95.556	96.000
0.06	96.444	94.444	95.778
0.07	96.667	93.333	96.000
0.08	96.444	94.667	95.333
0.09	96.222	94.000	95.556
0.1	96.222	93.333	95.556

5.2. Final experiment result

The experimental result from the parameter tuning is adopted in the final experiment result. The result obtained from parameter tuning is used in this experiment. Table 6 shows the experimental parameter setting for the final experiment whilst the result of final experiment is shown in table 7. From the experimental result, it can be concluded that the ENN performs better by achieving slightly higher accuracy than FFNN and RNN. The ENN is able to perform better due to its generality performance with its ability to aggregate multiple neural network output compared to single neural network. Furthermore, ENN is also able to cover a wider search space compared to single neural network. The fault tolerance in ENN is also one of the reasons for its better performance as even one of neural network has performed poorly, it will not affect much on the end result when the other neural networks are still performing well.

Table 6. Experimental parameter setting for final experiment.

Number of Neurons in Input Layer	18 Neurons	
Number of Neurons in Hidden Layer	FFNN	2
	Elman RNN	12
	ENNs	2
Number of Neurons in Output Layer	1 Neuron	
Transfer Function	Sigmoid Function	
Performance Function	Cross Entropy (CE)	
Number of Epoch	250	
Learning Rate	FFNN	0.002
	Elman RNN	0.002
	ENNs	0.001

Table 7. Final experiment result.

	FFNN	RNN	ENN
Best Accuracy for 100 Runs	94.299	94.222	**94.453**
Average Accuracy for 100 Runs	94.051	93.895	**94.155**

6. Conclusion

This paper investigates the performance of three neural network types which are feedforward neural network, recurrent neural network and ensemble neural network for detecting a phishing attack. A graphical user interface is developed to facilitate the experiments. The parameter tuning experiment showed that FFNN and ENN require less hidden neurons compared to RNN due to the complexity in the RNN. The lower learning rate does produce better result due to its ability to reduce the error in computation. ENN is able to achieve slightly better accuracy compared to FFNN and RNN due to the generality characteristic and other characteristics.

Acknowledgment

This project is partially supported by UMS research grant SBK0366-2017.

References

[1] Miller, D., & Slater, D. 2001. *The Internet: an ethnographic approach*.

[2] Comer, D. E. 2018. *The Internet book: everything you need to know about computer networking and how the Internet works*. Chapman and Hall/CRC.

[3] Kumar, V., Srivastava, J., & Lazarevic, A. (Eds.). 2006. *Managing cyber threats: issues, approaches, and challenges*. 5. Springer Science & Business Media.

[4] Gupta, B. B., Arachchilage, N. A., Psannis, K. E. 2018. Defending against phishing attacks: taxonomy of methods, current issues and future directions. *Telecommunication Systems* 67(2), 247-267.

[5] Vayansky, I., Kumar, S.L. 2018. Phishing–challenges and solutions. *Computer Fraud & Security* (1), 15-20.

[6] Prakash, P., Kumar, M., Kompella, R. R., Gupta, M. 2010. Phishnet: predictive blacklisting to detect phishing attacks. In *2010 Proceedings IEEE INFOCOM*, pp. 1-5. IEEE.

[7] Mao, J., Li, P., Li, K., Wei, T., Liang, Z. 2013 BaitAlarm: detecting phishing sites using similarity in fundamental visual features. In *2013 5th International Conference on Intelligent Networking and Collaborative Systems*, pp. 790-795. IEEE (2013).

[8] Dilek, S., Çakır, H., & Aydın, M. 2015. *Applications of artificial intelligence techniques to combating cyber crimes: A review*.

[9] Mohri, M., Rostamizadeh, A., & Talwalkar, A. 2018. *Foundations of machine learning*. MIT press.

[10] Abiodun, O. I., Jantan, A., Omolara, A. E., Dada, K. V., Mohamed, N. A., & Arshad, H. 2018. *State-of-the-art in artificial neural network applications: A survey*. Heliyon, 4(11).

[11] Gan, K. S., Chin, K. O., Anthony, P., & Chang, S. V. 2018. Homogeneous Ensemble FeedForward Neural Network in CIMB Stock Price Forecasting. In *2018 IEEE International Conference on Artificial Intelligence in Engineering and Technology (IICAIET)* (pp. 1-6). IEEE.

[12] Huang, H., Qian, L., & Wang, Y. 2012. A SVM-based technique to detect phishing URLs. *Information Technology Journal*, 11(7), 921-925.

[13] Akinyelu, A. A., & Adewumi, A. O. 2014. Classification of phishing email using random forest machine learning technique. *Journal of Applied Mathematics*.

[14] Nilam, B., Sneha, M., Puri, N.V. 2014. An Efficient Approach To Detecting Phishing A Web Using K-Means And Naïve-Bayes Algorithms With Results. *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET),* 3(5). pp. 1584-1589.

[15] Zhang, N., & Yuan, Y. 2013. *Phishing detection using neural network*. Department of Computer Science, Department of Statistics, Stanford University, CA.

[16] Mohammad, R., McCluskey, T. L., & Thabtah, F. A. 2013. Predicting phishing websites using neural network trained with back-propagation. *World Congress in Computer Science, Computer Engineering, and Applied Computing*.

[17] Jameel, N. G. M., & George, L. E. 2013. Detection of phishing emails using feed forward neural network. *International Journal of Computer Applications*, 77(7).

[18] Kathirvalavakumar, T., Kavitha, K., & Palaniappan, R. 2015. Efficient harmful email identification using neural network. *Journal of Advances in Mathematics and Computer Science*, 58-67.

[19] Swetha Babu, K. Radha, D. 2018. Phishing Detection in Websites Using Neural Networks and Firefly. *International Journal of Engineering and Computer Science*, 5(9). pp. 18197-18204.

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

[21] Demuth, H. B., Beale, M. H., De Jess, O., & Hagan, M. T. 2014. *Neural network design*. Martin Hagan.

[22] Tumer, K., & Ghosh, J. 2002. Robust Combining of Disparate Classifiers Through Order Statistics. *Pattern Analysis & Applications*. 5(2):189-200."

[23] Panchal, F. S., & Panchal, M. 2014. Review on methods of selecting number of hidden nodes in artificial neural network. *International Journal of Computer Science and Mobile Computing*, 3(11), 455-464.

[24] Wilson, D. R., & Martinez, T. R. 2001. The need for small learning rates on large problems. In *IJCNN'01. International Joint Conference on Neural Networks. Proceedings* (Cat. No. 01CH37222) (Vol. 1, pp. 115-119). IEEE.