A Novel Framework for DNS Exfiltration Malware—Optimizing Neural Network Hyperparameters Using Swarm Intelligent Algorithm

G. Revathy, P. Suthanthiradevi, P. Sathishkumar, S. Sivakumar, L. Megalan Leo, and A. Rajasekar

Abstract Domain name service (DNS) is a well-liked approach that steals confidential data from corporate organizations and keeps a hidden channel open for interactions between a hostile website and control/command servers. Given the importance of DNS services, organizations frequently configure their gateways to allow DNS requests that enable attackers to transit encrypted messages to a hacked host under their influence. This research work proposes deep learning-based neural networks in which data are trained to identify low and slow data intrusions and tunneling through DNS. Moreover, the authors used Swarm Intelligent Algorithm, the most computational intelligent technique appropriate to resolve in identification of DNS malware from overall data samples. Our experimental results show that threats were blocked with detection rate around 99.9%, and loss accuracy is very less. Our work proves unequivocally that the flow of network system will not be impacted, also no lagging in performing operation in the system.

G. Revathy (✉)
Department of Computer Science and Engineering, Vels Institute of Science, Technology and Advanced Studies, Chennai, India
e-mail: revathy.se@velsuniv.ac.in

G. Revathy · P. Suthanthiradevi
SRM Institute of Science and Technology, Chennai, India
e-mail: suthantp@srmist.edu.in

P. Sathishkumar
Department of Computer and Communication Engineering, Rajalakshmi Institute of Technology, Chennai, India
e-mail: sathishkumar.p@ritchennai.edu.in

S. Sivakumar
Department of ECE, BIHER, Chennai, India

L. M. Leo
Department of ECE, Sathyabama Institute of Science and Technology, Chennai, India

A. Rajasekar
Department of AI and DS, Sri Sai Ram Institute of Technology, Chennai, India
e-mail: rajasekar.ai@sairamit.edu.in

A. Q. H. Badar et al. (eds.), *Smart Grid Security and Protection*, Lecture Notes in Electrical Engineering 1307, https://doi.org/10.1007/978-981-96-0824-9_5

Keywords Domain name system (DNS) · Optimization technique · Swarm intelligent algorithm · Hyperparameters

1 Introduction

Evolutionary algorithms have been successfully used in a wide range of contemporary techniques and domains in recent years, involving optimizations, feature engineering, analytical thinking, segmentation, as well as cluster analysis. One of the most popular optimization techniques utilized in challenging optimization situations is the particle swarm optimization (PSO) approach. The PSO is a widely used optimization and search algorithm that is member in the group of evolutionary computation that was provoked either by communal and helpful behavior of flocking birds. Hence, the authors applied optimization techniques for performing hyperparameter tuning of parameters to achieve optimal solution in detection of DNS exfiltration malware happened in website. Such tuning of parameters particularly learning rate, epochs, and batch size helps in enhancing the model performance in terms of validation accuracy, and validation loss as well.

The main contributions of this research work are mentioned as follows:

- Utilizing evolutionary deep learning optimization models such as RMSProp, SGD optimization, and Adam optimizer to detect malware, especially DNS exfiltration which spoil the website.
- Moreover, the author split DNS input samples into training and testing of 70:30 rations appropriate for validating the trained optimization models.
- To achieve greater accuracy, the hyperparameter tuning is executed to construct the model in optimized manner from DNS exfiltration metadata via optimization approaches like swarm intelligent, Cuckoo search, and Social Spider.
- The performance of optimization technique is validated in terms of training accuracy, training loss, validation accuracy, and validation loss.

2 Background

Several researchers implemented different kinds of machine learning algorithms in the identification of DNS malware by blocking various tunnel-based malware over DNS logs and DNS protocol. The authors of Al-Mashhadi et al. [1] implemented a hybrid approach for analyzing and detecting DNS traffic while undergoing robot network lifecycle. Using such hybrid approach, DNS malware was detected with an accurateness of 99.96%. Churcher et al. [2] developed machine-based K-nearest neighbor approach for identifying DNS malware based on Internet of Things with 99% efficiency in binary as well as multiclass classification. To categorize a particular DNS request, Chowdary et al. [3] utilized the very first strategy leverages cache misses in such domain name system server, and then applied various machine-based learning methods for DNS malware detection. Steadman and Scott-Hayward [4] used data mining methods for DNS malware detection with better outcomes. DNS firewall malware has been identified by Marques et al. [5] based on machine learning approach.

To overcome the limitations of traditional approach in website-based malware like computational complexity, execution time, and not getting optimal solution in malware identification, the author applied deep learning-based optimization technique for obtaining greater performance by performing hyperparameter tuning of parameters, especially learning rate and batch size. Few of the authors used deep learning models like deep-based LSTM approach for DNS malware recognition over DNS logs introduced by Sakarkar et al. [6]. Altuncu et al. [7] detected DNS-based exfiltration the threat by training the layers in network using several deep learning models. The malware detection rate was 99.9%, threat block time was around 0.923 ms. Moreover, to enhance the overall performance of malware detection based on DNS, the authors of Raju et al. [8] applied cuckoo search optimization algorithm and Ali [9] applied particle swarm approach to attain optimal solution. Here is the review of various existing research work on DNS exfiltration malware in terms of dataset, techniques used, and accuracy described as Table 1. From the Table 1 [1] attained accuracy as 99.97% without getting optimal but our proposed model used optimization techniques for tuning hyper-parameters with optimal solution as 99.99% accuracy.

3 Methodology

3.1 System Description

The proposed architectural diagram for DNS exfiltration malware detection and classification using various optimization techniques along with hyperparameter tuning to obtain greater convergence rate are described in Fig. 1. Proposed system comprises data preprocessing, splitting of data, training the network layers, optimization approach, and finally validation. During preprocessing, the data samples are balanced. Data preparation includes the crucial step of normalizing training data. Whenever data samples in a datasets are not really evenly distributed, this is known as data instability and it might result in possible dangers while constructing models.

3.2 Statistics of DNS Metadata

Data regarding DNS exfiltration malware gathered using the following link: https://www.unb.ca/cic/datasets/dns-exf-2021.html

Table 1 Review of existing DNS exfiltration malware based on machine, deep learning, and efficiency

Literature	Kinds of malware/ dataset	Dataset	Techniques used	Accuracy (%)
Al-Mashhadi et al. [1]	DNS-based botnet malware/	NIMS	Hybrid (JRIP + PART)	99.97
		CTU 13		99.96
Churcher et al. [2]	DNS-based malware	Bot-IOT	KNN	99
Chowdhary et al. [3]	DNS tunneling traffic	Github	KNN	93.9
Steadman and Scott-Hayward [4]	DNS exfiltration	–	Data mining	–
Marques et al. [5]	DNS firewall	OSINT resource	CART	96
Sakarkar et al. [6]	DNS tunneling malware	Data gathered from DNS (Suricata) logs	LSTM	97.2
Kozlenko and Tkachuk [10]	DNS spoofing	DNS data, TCP/ IP data	RNN	70
Ali et al. [9]	Android-based DNS	GENOME project	Genetic Algorithm and Particle Swarm Optimization	96

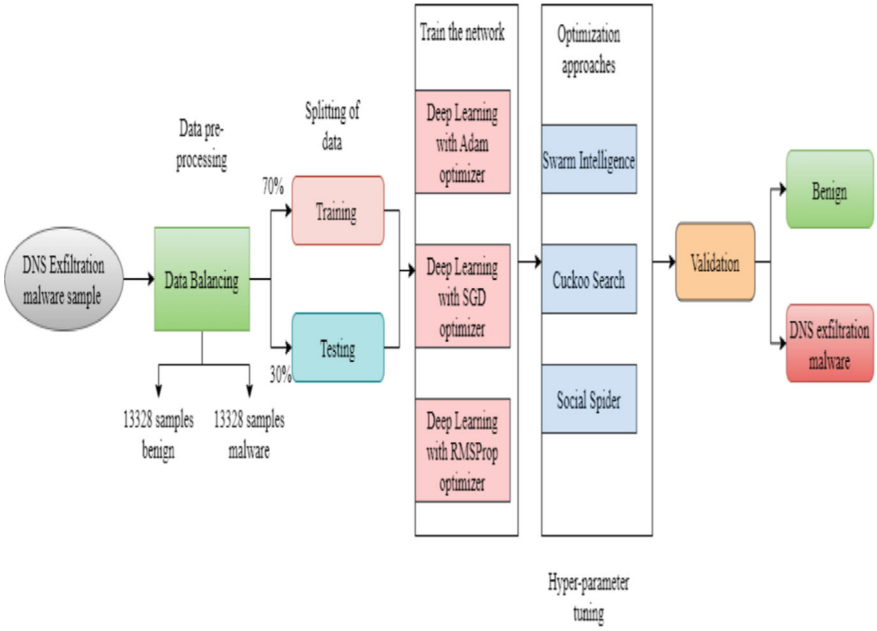
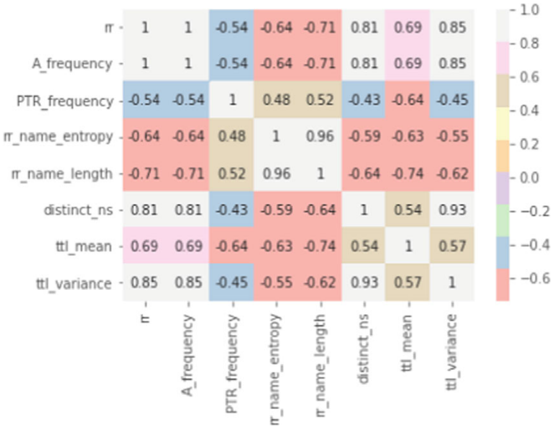


Fig. 1 Proposed framework for DNS exfiltration malware detection

Table 2 Description of dataset for DNS exfiltration

Total	Benign	Attack
93,430	80,102	13,328
After data balancing		
26,656	13,328	13,328



	rr	A_frequency	PTR_frequency	rr_name_entropy	rr_name_length	distinct_ns	ttl_mean	ttl_variance
rr	1	1	-0.54	-0.64	-0.71	0.81	0.69	0.85
A_frequency	1	1	-0.54	-0.64	-0.71	0.81	0.69	0.85
PTR_frequency	-0.54	-0.54	1	0.48	0.52	-0.43	-0.64	-0.45
rr_name_entropy	-0.64	-0.64	0.48	1	0.96	-0.59	-0.63	-0.55
rr_name_length	-0.71	-0.71	0.52	0.96	1	-0.64	-0.74	-0.62
distinct_ns	0.81	0.81	-0.43	-0.59	-0.64	1	0.54	0.93
ttl_mean	0.69	0.69	-0.64	-0.63	-0.74	0.54	1	0.57
ttl_variance	0.85	0.85	-0.45	-0.55	-0.62	0.93	0.57	1

Fig. 2 Covariance of DNS data samples

Table 2 describes the metadata of DNS exfiltration malware, with 80,102 benign samples and 13,328 attack samples—for a total of 93,430 samples. After performing data balancing, the number of samples available for DNS is 26,656. In From Table 2, the author mentioned that performing data balancing provides a similar quantity of data appropriate for predicting every class, and hence, it provides a good notion of how to react while testing DNS data.

3.3 Covariance

Subsequently, the covariance of data samples along with features such as rr, A_ frequency, PTR_frequency, rr_name entropy, rr name length, distinct_ns, ttl_mean, ttl_variance is shown in Fig. 2.

3.4 Train the Model

ConvNet comprises convolutional layer, pooling, and FC layer (Fully Connected) that allocate the features as well as weights of every feature. The outcome is generated from every convolutional layer supplied into ReLU function, max pooling, and activation function. The last convolutional layer, which is followed by a sigmoid function, generates a single outcome and is classified as either malware or benign.

3.5 SGD Optimizer

Stochastic gradient descent is perhaps the most well-known and often used optimization technique. It offers a quick and efficient way to determine the neural network's ideal parameters. Every optimizers aim to arrive somewhere at global optimal solution in which the computational complexity has the smallest value. Here, the gradient values are identified and revised the bias and weight values throughout every iteration of training the neural network. Moreover, the cost gets reduced and travel nearer to the least optimal value.

3.6 Adam Optimizer

In place of the conventional stochastic gradient descent method, Adam is an optimization technique which can be employed to iteratively upgrade connection weights depending on training examples. An extension of SGD optimizer is Adam optimizer. Such optimization algorithm appropriate for train the deep learning models based on detection of first-order and second-order moments as well as handling thin slope on noisy circumstances. The update rule by Adam optimization approach is given in Eq. (1).

$$\theta_{n+1} = \theta_n - \frac{\alpha}{\sqrt{\hat{v}_n} + \epsilon}\hat{m}_n \tag{1}$$

3.7 RMSProp Optimizer

Similar to gradient descent method, RMSProp optimizers are performed based on momentum. Hence learning rate might be enhanced with faster convergence. The main variation among GD and RMS optimizer is that the method of evaluating gradients only. Equations (2) and (3) demonstrate the method for estimating gradients.

$$v_{\mathrm{dw}} = \beta \cdot v_{\mathrm{dw}} + (1 - \beta) \cdot dw^2 \tag{2}$$

$$v_{\mathrm{db}} = \beta \cdot v_{\mathrm{dw}} + (1 - \beta) \cdot db^2 \tag{3}$$

Here β represents the momentum and the value fixed to 0.9.

$$W = W - \propto \cdot \frac{\mathrm{dw}}{\sqrt{v_{\mathrm{dw}}} + \epsilon} \tag{4}$$

$$b = b - \alpha \cdot \frac{\mathrm{db}}{\sqrt{v_{\mathrm{db}} + \epsilon}} \tag{5}$$

$v_{\mathrm{db}} = 0$, also provides a method called epsilon with a minimal amount in the denominator to stop the slopes from shooting up presented in Eqs. (4) and (5).

3.8 Hyperparameter Tuning

To quantify the discrepancy among our predicted results and indeed the desired classification, the author created a loss function before training a neural network model. Here, we are searching for a particular number of weights that will enable the neural net to estimate accurately and will consequently achieve a minor loss optimal solution. In this research work, the author utilizes three optimization techniques such as Particle Swarm, Cuckoo Search, and Social Spider algorithm for performing optimization tasks to obtain optimal solutions in detection of DNS exfiltration malware precisely. Moreover, learning rate and batch size are two parameters tuned in this study to attain greater accuracy and fewer losses.

3.9 Particle Swarm Optimization (PSO)

The swarm intelligence algorithm handles the problems that classical optimization based on individual agent and criterion cannot solve by replicating natural biological evolution and/or social behavior of species. The fundamental mechanism of these approaches is the understanding of the systematic and organizational principles governing animals' individual and/or in-group behavior. For instance, flocks and herds work together to find food or mates. Each member of the herd or flock gains knowledge from the experiences of the others as well as from their own, and they each modify their search method accordingly. A popular population-based optimization technique linked to evolutionary algorithms is the particle swarm optimization method (PSO) which is easier and quicker for getting optimal value. PSO has thus found extensive use in a variety of issues as well as various domains, including development, parameter extraction, clustering, detection of various insights, and classification.

PSO draws its inspiration from the swarming socioeconomic flock of organisms that cooperate and work together to locate prey. Similar to optimization computation, a PSO species (referred to as a swarming) is made up of alternative solutions or people (referred to as particles) that have been given randomized starts. Subsequently, in order to identify the ideal solution, every object moves with something like a velocity vv in the search process. Because of their own experiences as well as the knowledge of the remaining members of the swarm, the elements gain knowledge as they progress.

Let the current position of particle taken as xi = xi1, xi2, xi3…..xii, then velocity of certain particle taken as vi = v1, v2, v3,… vii and D denotes the dimensionality of search space. Every particle modifies its velocity to attain optimal solution as shown in Eqs. (6) and (7)

$$x_{\mathrm{id}}^{t+1} = x_{\mathrm{id}}^{t} + v_{\mathrm{id}}^{t+1} \tag{6}$$

$$v_{id}^{t+1} = w * v_{id}^{t} + c_1 * r_1 * \left(p_{id} - x_{id}^{t}\right) + c_2 * r_2 * \left(p_{gd} - x_{id}^{t}\right) \tag{7}$$

From the above equations, t denotes number of iterations “t” time, the value of d is 1, 2, 3,…. D, w represents the weight of inertia, p_{id} represents p_{best} in which preceding location of specific particle, p_{gd} represents g_{best} represents choosing best location of particle globally, c_1 and c_2 represent increased rate of features fixed as 2, r_1 and r_2 denote the random variables ranging between 0 and 1.

3.9.1 Cuckoo Search Algorithm

Cuckoo Search is a nature-inspired meta-heuristic algorithm appropriate for solving optimization issues. This optimization technique imitates cuckoo nesting parasitism. Additionally, the search procedure of the CS algorithm replicates the periodic search of the Le’vy fly, which is made up of a succession of straight moves as well as sudden 90 degree rotation. To apply such optimization tool, three rules are listed below:

- Every cuckoo lays a single egg, which is then incubated in a host nest randomly.
- The best nest with top-notch eggs is handed down to the subsequent generation.
- With a certain number of host nests accessible, the host has a p_2 probability of seeing the cuckoo’s egg (0, 1).
- The host either destroys the invader’s egg or throws away the nest after spotting it.
- The CS technique is well-known in social science since it can be optimized well with minimal parameters.

Epoch is defined as “the number of occasions a complete dataset is run through all the neural network model. One epoch denotes one-fourth and backpropagation of the experimental input through the neural net”.

3.9.2 Social Spider Algorithm

This study proposes the social spider optimization (SSO), a revolutionary swarm approach for tackling optimization problems. Such algorithm is constructed on a modeling of communal insects' cooperative behavior. In the suggested method, participants simulate a community of spiders that cooperate with one another in accordance with biological principles known as collaborative colony laws. The algorithm takes into account both male and female searching agents (spiders). Every organism is controlled by a variety of evolving operations that vary according to species and replicate cooperation between different behaviors that seem to be usual in a swarm. This fact makes it possible to mimic the social cooperation of such colony in a more accurate manner while also incorporating supercomputing mechanisms that prevent serious shortcomings that are frequently found in the renowned PSO algorithm, like optimal solution but also an inaccurate exploratory measure. The effectiveness and reliability of this algorithm are demonstrated through comparison to certain other well-known evolutionary techniques. The female and male spider can be calculated using the following Eqs. (8) and (9).

$$N_f = \text{floor}[(0.9 - r\,\text{and} \cdot 0.25) \cdot N] \tag{8}$$

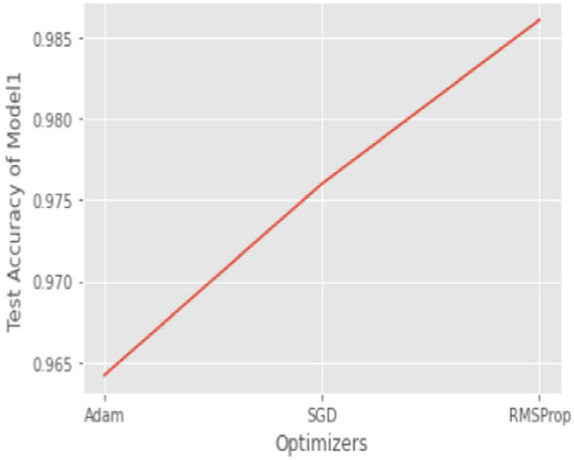
$$N_m = N - N_f \tag{9}$$

where rand indicates the random number and its value ranges from 0 to 1. Floor represents the mapping of real values into integer values. Where N denotes number of elements, and N_m and N_f signify male and female spider.

4 Experimental Outcomes

Initially, the authors implemented deep-based Adam optimizer, stochastic gradient descent optimizer, and RMSProp optimizer for training the layers in neural network by providing input data samples. An optimization technique is allocated after the nodes of a neural network have been assembled. In order to obtain the smallest possible loss function, the optimizers such as Particle swarm, Cuckoo Search, and Social Spider optimization techniques adjusted the learning rate along with weights of the neurons in NN. Either greater accuracy or least amount of loss is achieved by using all these optimizers in this research work. The learning rate serves as one of the optimizer's hyperparameters. In our work, the author adjusted the learning rate, epochs, batch size as well. The number of iterations required for a model to achieve the minimal error rate is controlled by network parameters such as learning rate.

Fig. 3 omparison on exfiltration malware detection using ADAM SGD and RMSProp optimizers



Although the model learns more quickly with an improved learning rate, this might ignore the minimum loss function and merely explore its surroundings. Finding a minimal prediction error has a greater chance with a low detection rate. Hence, our proposed optimization approaches attained greater accuracy with the least loss for ten epochs. Moreover, the computational complexity for tuning hyperparameters is lesser in intelligent-based particle swarm optimization approach when compared with other optimizations. This helps to attain global optimal solution and better convergence rate in detection of DNS exfiltration malware. Comparison has been made among models with three optimizers such as Adam, SGD, and RMSProp to attain convergence rate. Among three optimizers, RMSProp achieved greater convergence rate of 98.9% for ten epochs as illustrated in Fig. 3.

Based on the metrics such as validation accuracy and validation loss, the authors evaluated overall performance of optimization techniques such as cuckoo search and Social spider. Figures 4 and 5 depict the performance of optimization models (both CSO and SSA) along with hyperparameter tuning to get optimal solution for DNS malware detection and classification by tuning the learning rate, epochs, and batch size.

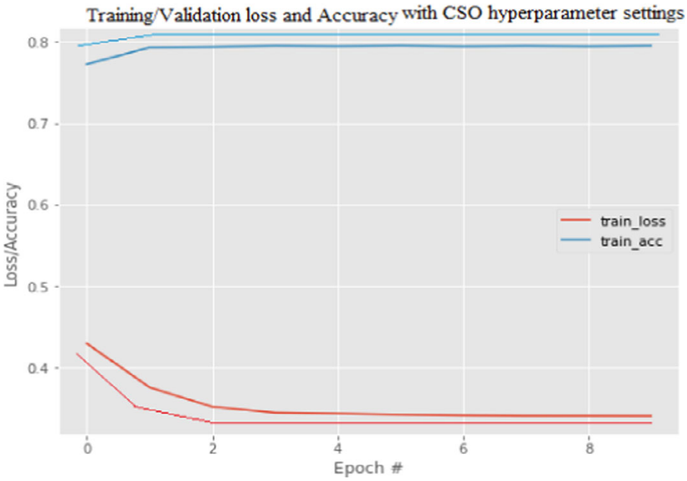


Fig. 4 Evaluation of validation accuracy/loss based on CSO approach

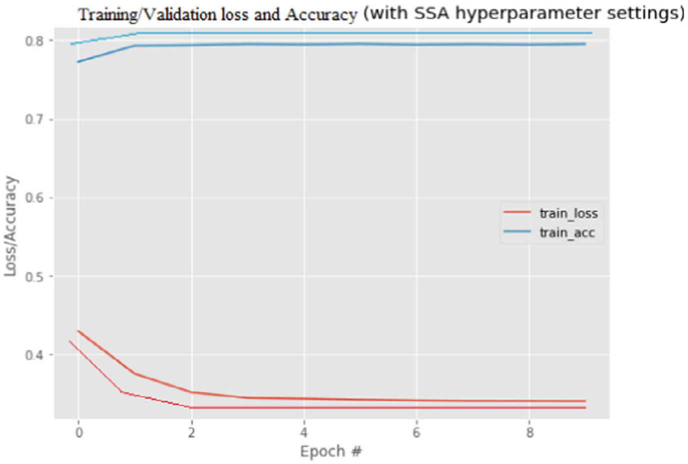


Fig. 5 Evaluation of validation accuracy/loss based on SSA optimization

5 Conclusion

In summary, the author's implemented deep learning-based CNN with optimizers such as Adam, SGD, and RMSProp in detection of website-based DNS Exfiltration malware along with its classification. Moreover, various optimization algorithms such as Cuckoo search and social spider optimization were developed for hyper-tuning of parameter to obtain better optimal solution in DNS malware detection. The convergence rate of optimization algorithm attained higher accuracy as well as higher detection rate of 99.9%. In future studies, the author will focus on hybridization approach with popular techniques in various applications.

References

1. Al-Mashhadi S, Anbar M, Hasbullah I, Alamiedy TA (2021) Hybrid rule-based botnet detection approach using machine learning for analysing DNS traffic. PeerJ Comp Sci 7:1–34
2. Churcher A, Ullah R, Ahmad J, Ur Rehman S, Masood F, Gogate M, Alqahtani F, Nour B, Buchanan WJ (2021) An experimental analysis of attack classification using machine learning in IoT networks. Sensors 21(2):1–32
3. Chowdhary A, Bhowmik M, Rudra B (2021) DNS tunneling detection using machine learning and cache miss properties. In: 2021 5th International conference on intelligent computing and control systems (ICICCS), pp 1225–1229. IEEE, Madurai, India
4. Steadman J, Scott-Hayward S (2018) Dnsxd: detecting data exfiltration over dns. In: 2018 IEEE Conference on network function virtualization and software defined networks (NFV-SDN), pp 1–6. IEEE, Verona, Italy
5. Marques C, Malta S, Magalhães J (2021) DNS firewall based on machine learning. Future Internet 13(12):1–18
6. Sakarkar G, Kolekar MKH, Paithankar K, Patil G, Dutta P, Chaturvedi R, Kumar S (2021) Advance approach for detection of DNS tunneling attack from network packets using deep learning algorithms. ADCAIJ: Adv Distrib Comput Artif Intell J 10(3):241–266
7. Altuncu MA, Gülağiz FK, Özcan H, Bayir ÖF, Gezgın A, Nıyazov A, Çavuşlu MA, Şahın S (2021) Deep learning based DNS tunneling detection and blocking system. Adv Electr Comput Eng 21(3):39–48
8. Raju PLN, Raju KS, Kalidindi A (2020) Feature selection and performance improvement of malware detection system using cuckoo search optimization and rough sets. Int J Adv Comput Sci Appl 11(5):708–714
9. Ali W (2019) Hybrid intelligent android malware detection using evolving support vector machine based on genetic algorithm and particle swarm optimization. IJCSNS 19(9):15–28
10. Kozlenko M, Tkachuk V (2019) Deep learning based detection of DNS spoofing attack. In: 2019 Scientific seminar on innovative solutions in software engineering, pp 10–11. Ivano-Frankivsk, Ukraine