Genetic Search Wrapper-Based Naïve Bayes Anomaly Detection Model for Fog Computing Environment

John Oche Onah[1], Shafi'i Muhammad Abdulhamid[1], Sanjay Misra[2(✉)], Mayank Mohan Sharma[3], Nadim Rana[4], and Jonathan Oluranti[2]

[1] Department of Cyber Security, Federal University of Technology, Minna, Nigeria
shafii.abdulhamid@futminna.edu.ng
[2] Center of ICT/ICE Research, Covenant University, Ota, Nigeria
{Sanjay.misra,jonathan.oluranti}@covenantuniversity.edu.ng
[3] Zillow Inc., San Francisco, USA
[4] College of Computer Science and Information Technology, Jazan University, Jazan, Saudi Arabia

Abstract. Fog computing will provide low-latency connectivity between smartphone devices and the cloud as a complement to cloud computing. Fog devices can, however, face security related challenges as fog nodes are near to end users with restricted computing capabilities. Traditional network attacks break the fog node system. While the intrusion detection system (IDS) has been well studied in traditional networks, it may sadly be impractical to use it specifically in the fog environment. Fog nodes still produce large quantities of data and thus allowing the IDS in the fog context over big data is of the utmost importance. In order to counter some of these network attacks, a proactive security defense technology, Intrusion Detection System (IDS), can be used in the fog environment using data mining technique for network anomaly detection and network event classification attack has proven efficient and accurate. This research presents a Genetic Search Wrapper-based Naïve Bayes anomaly detection model (GSWNB) in Fog Computing environment that eliminates extraneous features to minimise time complexity as well as building an improved model that predict result with a higher accuracy using NSL-KDD dataset as benchmark dataset. From the experiment, the proposed model demonstrates a higher overall performance of 99.73% accuracy, keeping the false positive rate as low as 0.006.

Keywords: Fog computing · Cloud computing · Genetic search · Anomaly detection · Naïve Bayes · Wrapper approach

1 Introduction

Many preventative steps have been introduced in literature avoid cyber-attacks, however the nature and evolution of threats and attacks has an impact on the effectiveness of such preventive measures particularly in the areas of cloud infrastructure, the Internet

A. Abraham et al. (Eds.): ISDA 2020, AISC 1351, pp. 1371–1382, 2021.
https://doi.org/10.1007/978-3-030-71187-0_127

of Things (IoT) and a new paradigm called Fog Computing [1, 2] as depicted in Fig. 1. Cloud computing has been rendering worldwide and global services through central design that support real-time interaction [3, 4], but still faces problems with handling large IoTs and a number of connected devices increasing daily. Fog computing is a link between the cloud and "Things", a highly virtualizable application infrastructure that allows distributed services and data delivery in a cloud to be transferred or expanded near to networked edge devices [5, 6]. The possible security vulnerabilities have become a key challenge owing to the constant movement of fog nodes and their data in their environment.

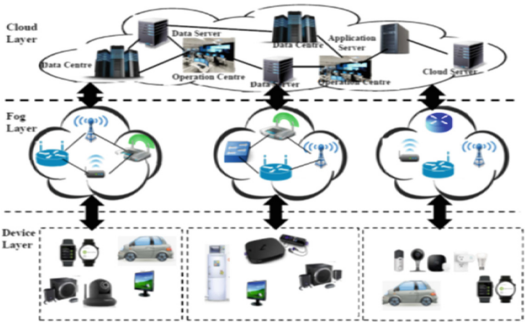

Fig. 1. Fog computing environment

Like any other computing and networking environment, an Intrusion Detection Systems (IDS) can be deployed in fog environment [7]. The anomaly detection highly involves various data mining techniques and machine learning algorithms such as Bayesian Network, Support Vector Machine, Artificial Neural Network, K-Means, Swarm Particle Optimisation (SPO), Genetic Algorithm [8]. Kai *et al.* [9] developed a Decision Tree (DT) dependent Intrusion Detection System for Big Data in Fog Environment without explicitly specifying the accuracy, the false positive and the negative rate. Xingshou *et al.* [10] designed an intrusion detection system based on sample selected extreme learning machine in fog computing and mobile edge computing with a high training computational time which is a problem when considering efficiency of a model or algorithm or technique in use, though their research recorded 99.07% accuracy. Farhoud *et al.* [11] used a smart technology method to develop an intrusion detection system for fog computing and IoT-based logistic applications. Their research work recorded 96.23% accuracy, 91.21% accuracy, and 3.51% false positive accuracy.

An anomaly is a deviation from a known behaviour, and profiles reflect the usual or predicted behaviours generated over a period of time from evaluating routine activities, network connections, hosts, or users [12]. Anomaly detection is also called Behaviour-based Detection since it pays attention to the behaviour of a novel traffic though its drawback is low profiles accuracy due to constant changes in observed events, unavailable during the reconstruction of behavioural profiles and difficult to trigger alerts at the right time [13, 14]. Levent et al. [15] classified feature selection model into Filter, Wrapper and Embedded method. In this technique learning and classification are two

steps for data classification. Every data in the classification technique in the dataset has the class-defining attribute value, and every class is predefined to give an analyst prior knowledge [16]. Classification can also be used for labelling each record in the data set and classifying the records in a predetermined set. The key contributions of this research study are laid out as follows:

- To reduce dimension of NSL-KDD datasets using Genetic Search Wrapper-Based algorithm for anomaly detection for Fog computing environment.
- To classify the reduced dataset using Naïve Bayes classification algorithm.
- To evaluate the performance of the model with accuracy, precision and computational time as standard parameters.

This research work proposed a Genetic Search Wrapper-based Naïve Bayes Anomaly Detection model for Fog computing environment. The remaining parts of the paper are arranged as follows: Section 2 describes the previous relevant literature. Section 3 points out the proposed GSWNB anomaly detection model. Section 4 outlines the experimental setup and Sect. 5 discusses the results and discussions. Finally, the conclusion and future works are chronicled in Sect. 6.

2 Related Works

Fog computing inherits a few cloud computing related challenges with about 76% studies exposing the security weakness in fog devices [17–19]. With regards to the use case of fog computing, interfering with the setup devices in public places is almost impossible as a result of little or no surveillance. Further, there is a potential threat involving the hardware equipment of third-party vendor as computation logic is moved close to the edge of the network [6]. Dastjerdi and Buyya [20, 21] offered a possible solution via the use of public-key infrastructures plus strong trustworthy executing systems in fog. Shi *et al.* [22] delivered in their work, a security framework based on cloudlet mesh that detects intrusion to distance cloud within mobile devices, cloudlet and cloud aiding secured communication. Technique called Extreme Learning Machine (ELM) was first proposed and used by Cheng et al. [23] to detect intrusion with a greater accuracy then SVM when compared. Ye and Yu [24] proposed a method of detecting intrusion by combing each class into an Ensemble Classifier with one-to-all strategy. For intrusion detection, an ELM weight has also been suggested [25]. Cai et al. [26] suggested a modern model of fusion that would incorporate a Bal Vector Machine (BVM), an ELM, and a Back Propagation (bp) neural network for intrusion detection. This method recorded strong performance in the detection of accuracy and false positive rate. Muniyandi et al. [27] established a hybridized method called cascading using clustering of k-mean and C.45 decision tree, to reduce the supremacy of k-mean and forced assignments. The k-means splitting the trainings into k sub-sets, followed by C.45 for the disassembled sub-sets. Throughout their study, Natesan et al. [28] have indicated an enhanced single weak classification with AdaBoost. As weak as better than AdaBoost, Bayes Net, Naïve Bayes and Decision Tree (DT) have been used. The key concern, though, is that there is a lack of mechanism for detecting novel attacks with a signature close to established

attacks, which could contribute to low detection. Govindarajan and Chandrasekaran [29] implemented a hybrid detection architecture that involves an ensemble and base classifiers for detection system. The ensemble module was designed utilizing both Radial Base Function (RBF) neural network and Multilayer Perceptron (MLP). There are several other works available in literature on intrusion detection [30–32] and algorithms but due to limitation and scope of study we are not including them.

3 Proposed GSWNB Model

The operation of this proposed GSWNB model is in two stages. Stage 1 involves the feature selection process using a wrapper approach with Genetic Search algorithm while stage 2 is about the classification of Test instances using Naïve Bayes. Process involved in stage 1 is screening and removing redundant features and a wrapper feature selection is proposed for the purpose of getting a better accuracy. Genetic search as the search algorithm used for searching through the space of possible features and Naïve Bayes based model employed on each subset for evaluation. At the end, feature subset is been selected based on the performance while, stage 2 entails building a classification model using a Naïve Bayes algorithm. Finally, an instance of a test is by the new Naïve Bayes based built classification model as shown by Fig. 2 followed by the algorithm.

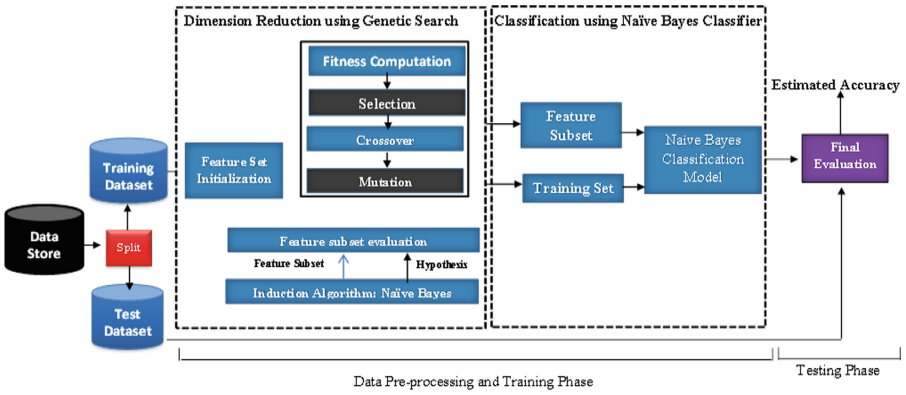

Fig. 2. Genetic search wrapper based Naïve Bayes anomaly detection mode

3.1 Dataset Description

The proposed model uses the NSL-KDD benchmark dataset as evaluation data which includes three separate datasets: the entire dataset, 20% of the entire training dataset and the complete KDD test dataset. A total of 25192 attacks and normal instances constitute 20% training data. Each instance consists of a set of 41 features and a label distinguishing each record as either a normal or a particular type of attack. Such features involve all sorts of continuous, discrete, and symbolic variables. The research is performed by adding a

10-fold cross validation method to the suggested intrusion detection pattern. The total number of normal and attack instances for each sample of training and testing dataset are displayed in Tables 1 and 2.

Table 1. Number of attack instances in the training

Attack types	Number of records
Normal	9711
DoS	7456
Probe	2421
R2L	2756
U2R	200
Total	**22544**

Established forms of attacks are those found in the training data set, while novel attacks are new attacks in the test data set, i.e. not included in the training data set. The kind of attacks are present are classified into: DoS, R2L, Probing and U2R.

Table 2. Number of attack instances in the testing dataset

Attack types	Number of records
Normal	67343
DoS	45927
Probe	11656
R2L	995
U2R	52
Total	**125973**

3.2 Genetic Search Algorithm Implementation Phase

Genetic algorithm begins by initiating a random number of possible solutions called population 20. Chromosome genes are described by a bit, character or number that are defined based on the NSL-KKD dataset's structure and properties. Next, the fitness of the individuals is assessed based on fitness function. The 0.6 probability selection and recombination operators are then used to study new solutions in the search space for the whole population. Ultimately, the mutation operator with the probability of 0.033 mutation conducts random adjustment in order to optimize the solutions. The process develops to a maximum of 20 generations, in this case as shown in Algorithm 1. The features of the NSL dataset were reduced to the amount of 19 at the conclusion of the

dimensional reduction. The final classification was then provided for by Naïve Bayes Classifier. See Appendix A for the final reduced features after genetic search.

Algorithm 1: Genetic Search Algorithm

Input: Dataset
Output: Subset features
Generate randomly, an initial population, P. //$P = 20$
Compute $e(x)$ for each member $x \in P$.
Define a probability distribution p over the member of P where $p(x) \propto e(x)$.
Select two population members x and y with respect to p.
Apply crossover to x and y to produce new population members x' and y'.
Apply mutation to x' and y'.
Insert x' and y' into P' // *The next generation.*
  if $|P'| < |P|$, goto 4
  Let $P \leftarrow P'$
  if there are generations to still process, *goto* 2.
Return $x \in P$ where $e(x)$ is highest.

3.3 Classification Phase – Naïve Bayes Algorithm

The Naïve Bayes is implemented as a classifier based on the Bayes concept, which is naive as the features are mutually exclusive.

Given a feature vector $X = \{x1, x2, \ldots, xn\}$ and a class variable Ci, Bayes theorem states that:

$$P(Ci/X) = \frac{P(X/Ci) * P(Ci)}{P(X)} \tag{1}$$

For $k = 1, 2, \ldots, i$
We call:
$P(Ci/X)$, the posterior probability,
$P(X/Ci)$ the likelihood,
$P(Ci)$, the prior probability of class, and;
P(X) the prior probability of predicator.
Therefore, using chain rule, the likelihood P(X/Ci) can be decomposed as:

$$\mathrm{P}\left(\frac{\mathrm{X}}{\mathrm{Ci}}\right) = \mathrm{P}(\mathrm{x1}, \ldots, \mathrm{xn}|\mathrm{Ci}) = \mathrm{P}(\mathrm{x1}|\mathrm{x2}, \ldots, \mathrm{xn}, \mathrm{Ci})\mathrm{P}(\mathrm{x2}|\mathrm{x3}, \ldots, \mathrm{xn}, \mathrm{Ci}) \ldots$$
$$\mathrm{P}(\mathrm{xn} - 1|\mathrm{xn}, \mathrm{Ci})\mathrm{P}(\mathrm{xn}|\mathrm{Ci}) \tag{2}$$

Equation 2 can be hard to calculate but using the naïve independence assumption which state that:

$$\mathrm{P}(\mathrm{xj}|\mathrm{xj} + 1, \ldots, \mathrm{xn}|\mathrm{Ci}) = \mathrm{P}(\mathrm{xj}|\mathrm{Ci}) \tag{3}$$

We can get:

$$P(X/Ci) = P(x1, ..., xn|Ci) = \prod_{j=1}^{n} * P(X_j/c_i) \tag{4}$$

$$P(X/Ci) = P(x1, \ldots, xn|Ci) = P\frac{(Ci/X) * \prod_{j=1}^{n} *P(X_j/c_i)}{P(X)} \tag{5}$$

Since the priority probability of predicator P(X) is constant given the input, we have:

$$P(X/Ci)\alpha P(Ci) * \prod_{j=1}^{n} *P(X_j/c_i)$$

The posterior probability can be written as:
Now, different class of values of Ci is obtained by finding the maximum of:

$$P(Ci) * \prod_{j=1}^{n} *P(X_j/c_i)$$

as:

$$Cm = \text{argmax}P(Ci) * \prod_{j=1}^{n} *P\left(\frac{X_j}{c_i}\right) \tag{6}$$

$$ci \in C$$

The priori probability of class $P(Ci)$ could be determined as the relative frequency of class Ci in the training data. Equation 6 is the model's Naïve Bayes classifier which implements the model.

Algorithm 2: Proposed GSWNB Algorithm

Input: Dataset
Output: Class labelled test instance
Generate randomly, an initial population, *P*.
Compute *e(x)* for each member $x \in P$.
Define a probability distribution *p* over the member of *P* where *p(x)* α *e(x)*.
Select two population members *x* and *y* with respect to *p*.
Apply crossover to *x* and *y* to produce new population members *x'* and *y'*.
Apply mutation to *x'* and *y'*.
Insert *x'* and *y'* into *P' // The next generation*.
if $|P'| < |P|$, goto 4
Let $P \leftarrow P'$
if there are generations to still process, goto 2.
Return $x \in P$ where *e(x)* is highest.
Given a training set, for each Class $c_i \in C$

i. Estimate the prior probability: $P(c_i)$
ii. For each feature *x*, estimate the probability of that feature value given Class c_i: $P(x_j/c_i)$

for each Class $c_i \in C$,
compute:

$$P(ci) * \prod_{j=1}^{n} P(xj/ci)$$

Select the most probable Class

$$Cm = \text{argmax}P(Ci) * \prod_{j=1}^{n} * P\left(\frac{X_j}{c_i}\right)$$

4 Experimental Setup

The experiment runs on a computer system that has a 64-bit Windows 10 Intel® CoreTM i5-2410M CPU @2.45 Ghz–2.4 GHz, 4.00 GB Random Access Memory as specification. The JAVA programming language was used to perform the experiment with the aid of WEKA 3.8 machine learning apparatus and WEKA Library functions with dozens of various feature selection techniques to pick the correct features. For the experiment, we used a well-known NSL-KDD benchmark dataset produced by MIT Lincoln Lab with the goal of juxtaposing the performance of various intrusion detection techniques. NSL-KDD data set containing classes grouped into five, namely: normal and four kinds of attacks including, DoS, Probing, U2R, and R2L.

5 Results and Discussion

5.1 Testing and Performance Evaluation

The experiment was carried out on a portion of the famous NSL-KDD benchmark dataset where besides the normal class classification it had, the entire dataset contained 25,192 instances with 41 features and 4 other class attack types such as DoS, Probing, U2R and R2L. Each experiment with 10-fold cross-validation was conducted as k-fold cross validation is a popular method of implementing any kind of classification scheme, as it avoids the possibility of creating an overfit classification model. The overall success of the proposed model as seen in a glimpse as the proposed model indicates a higher true positive rate of 98.1% and a very low false positive rate of 0.6% as described in depth in this section.

5.2 Overall Performance of the Proposed Model

The specifics of the average results of the proposed model in terms of performance are displayed in Table 3. The model provided 98.1% and 0.6% of an overall high true positive rating and false positive rating respectively. The model appeared to perform better against DoS among the sets of attacks with 96.9% true positive rate and 77.1% as the least rating on R2L attack. Once, the proposed model showed a rather large 99.7% ROC area indicating excellent results.

Table 3. Overall performance of the proposed model

Algorithms	Bayesian network	J48	SMO	Proposed GSWNB algorithm
Accuracy (%)	85.76	96.43	95.99	99.73
Precision (%)	91.40	96.5	97.6	99.10
Execution time in training phase (sec)	0.2	1.73	13.01	0.18

5.3 Comparison of the Proposed Model With Other Algorithms

The findings indicate that a more reliable 99.73% accuracy rating of the proposed model is obtained while others, such as the Bayesian Network, have an accuracy rate of 85.76%, 96.43% with J48 and another common SMO algorithm, 95.99%. In contrast with other popular algorithms, the time during the training phase of the proposed model was fairly short, with the proposed approach taking just 0.18 s whereas it took SMO 13.01 s during the classification model training process. See Fig. 3.

Table 4. Comparison of the proposed model with other algorithms

Class	True Positive Rate (TPR) (%)	False Positive Rate (FPR) (%)	ROC area (%)
Normal	97.5	0.6	99.7
U2R	73.5	0.2	93.5
R2L	77.1	0.1	99.1
DoS	96.9	0.6	99.7
Probing	93.4	0.4	99.2
Average weight	98.1	0.6	99.7

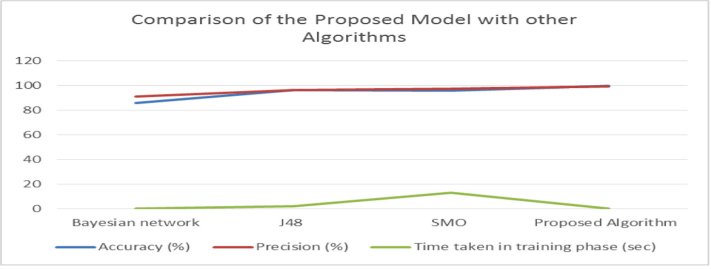

Fig. 3. Comparison of the proposed model with other algorithms

5.4 Proposed GSWNB Approach and other Feature Selection Techniques

Table 4 indicted the efficiency of the proposed GSWNB method in terms of performance relative to certain well-known selection techniques. The proposed wrapper method picked only 19 significant features in 41 showing stronger performance in terms of accuracy, which was better with 99.73% than the CFS, consistency type features selection techniques whereby 94.88% accuracy was reported with CFS type filter methodology and consistency type features selection method showed 93.13% accuracy via the use of rank search technique. The successful 91.13% accuracy of the CFS type genetic search was significantly less than the proposed wrapper method, which still used genetic search to search the space feature.

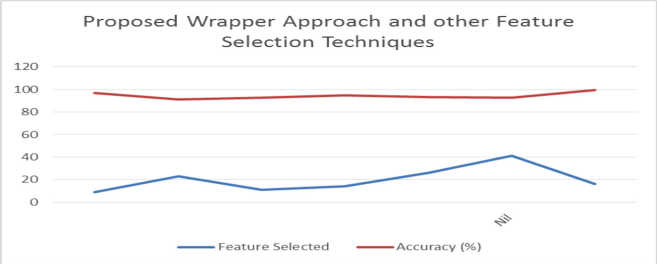

Fig. 4. GSWNB approach and other feature selection techniques

5.5 Comparison of Full Dataset Features with Selected Dataset Features

The model's performance in full data set and the 19 characteristics selected in terms of accuracy and computational time. Figure 4 has shown 92.68% of accuracy for complete datasets but has improved accuracy by 99.73%, with a lesser computational time of 0.18 s, when irrelevant data has been successfully reduced into 19 features using the proposed GSWNB-based dimension reduction approach.

5.6 Comparison of the GSWNB Model with Results from Previous Research Papers

The accuracy of this research is seen to be better than the results of other research papers as shown in Table 5.

Table 5. Comparison of the proposed GSWNB model with other research papers

Research Papers	Accuracy (%)	False positive rate (%)
Xingshou *et al.* [10]	99.07	-
Farhoud *et al.* [11]	98.35	3.51
Adel *et al.* [15]	91.97	3.44
Proposed GSWNB	**99.73**	**0.6**

6 Conclusion and Future Work

In this research, a novel model termed genetic search Wrapper-based Naïve Bayes anomaly detection model (GSWNB) for intrusion detection in fog computing environment is proposed. GSWNB is based on wrapper approach for feature selection and Naïve Bayes Classifier. The process included preparing an appropriate NSL-KDD train dataset with features 19 out of 41 chosen as final features then followed with the test

instances classification using Naïve Bayes Classifier. A 0.006 False Positive Rate (FPR) and True Positive Rate (TPR) of 98.1%, was observed in the proposed GSWNB model. The findings of the proposed model appeared accurate and outdone other classifiers in terms of their efficiency and accuracy success.

References

1. Mukherjee, M., Matam, R., Shu, L., Maglaras, L., Ferrag, M.A., Choudhury, N., Kumar, V.: Security and privacy in fog computing: challenges. IEEE Access **5**, 19293–19304 (2017)
2. Abdullah, A., Tariq, A.A.: Fog computing and security issues: a review. In: International Conference on Computers Communications and Control, vol. 13, pp. 237–239 (2018)
3. Muhammad, R.A., Shangguang, W., Muhammad, A.Z., Ahmer, K.J., Umair, A., Salman, R.: Fog computing: an overview of big IoT data analytics. Wirel. Commun. Mob. Comput. **2**(2), 1–22 (2018)
4. Hua-Jun, H.: From cloud computing to fog computing: unleash the power of edge and end devices. In: 2017 IEEE 9th International Conference on Cloud Computing Technology and Science, pp. 331–334 (2017)
5. Bonomi, F., Milito, R., Zhu, J., Addepalli, S.: Fog computing and its role in the internet of things. In: Proceedings of the 1st ACM Mobile Cloud Computing Workshop, MCC 2012, pp. 13–15. ACM (2012)
6. Vaquero, M.L., Luis, R.M.: Finding your way in the Fog: towards a comprehensive definition of fog computing. ACM SIGCOMM Comput. Commun. Rev. **44**(5), 27–32 (2014)
7. Modi, C., Patel, D., Borisaniya, B., Patel, H., Patel, A., Rajarajan, M.: A survey of intrusion detection techniques in cloud. J. Netw. Comput. Appl. **36**(1), 42–57 (2013)
8. Simon, F.: Big data mining algorithms for fog computing. Assoc. Comput. Mach. **3**(2), 57–61 (2017)
9. Kai, P., Victor, C.M., Lixin, Z., Shangguang, W., Chao, H., Tao, L.: Intrusion detection system based on decision tree over big data in fog environment. Wirel. Commun. Mob. Comput. **1**(2), 1–10 (2018)
10. Xingshou, A., Lin, F., Xu, S., Miao, L., Gong, C.: A novel differential game model-based intrusion response strategy in fog computing. Sec. Commun. Net. **2018**, 9 (2018). https://doi.org/10.1155/2018/1821804
11. Farhoud, H., Payam, V.A., Juha, P., Timo, H., Hannu, T.: An Intrusion detection system for fog computing and IoT based logistic systems using a smart data approach. Int. J. Digit. Content Technol. Appl. (JDCTA) **10**(6), 34–46 (2016)
12. Ismail, B., Salvatore, D.M., Ravi, S.: A survey of intrusion detection systems in wireless sensor networks. Commun. Surv. Tutor. **16**(1), 266–282 (2014)
13. Liao, H.-J., Lin, C.-H.R., Lin, Y.-C., Tung, K.-Y.: Intrusion detection system: a comprehensive review. J. Netw. Comput. Appl. **36**(1), 16–24 (2013)
14. Ming-Yang, S.: Real-time anomaly detection systems for Denial-of-Service attacks by weighted K-nearest-Neighbor classifiers. Exp. Syst. Appl. **38**, 3492–3498 (2011)
15. Levent, K., Thomas, A.M., Shahram, S.: A Network intrusion detection system based on a hidden naïve Bayesian multiclass classifier. Exp. Syst. Appl. **39**, 13492–13500 (2012)
16. Adel, S.E., Zeynep, O., Adnan, M.A.B.: A novel feature-selection approach based on the cuttlefish optimization algorithm for intrusion detection systems. Exp. Syst. Appl. **42**, 2670–2679 (2015)
17. Li, J., Jin, J., Yuan, D., Palaniswami, M., Moessner, K.: EHOPES: data-centered fog platform for smart living. In: 2015 International Telecommunication Networks and Applications Conference (ITNAC), pp. 308–313. IEEE (2015)

18. Stojmenovic, I., Wen, S.: The Fog computing paradigm: scenarios and security issues. In: 2014 Federated Conference on Computer Science and Information Systems, vol. 3, pp. 1–8 (2014)
19. Wen, Z., Yang, R., Garraghan, P., Lin, T., Xu, J., Rovatsos, M.: Fog orchestration for internet of things services. IEEE Internet Comput. **21**(2), 16–24 (2017)
20. Dastjerdi, A., Buyya, R.: Fog computing: helping the internet of things realize its potential. Computer **49**(8), 112–116 (2016)
21. Botta, A., de Donato, W., Persico, V., Pescapé, A.: Integration of cloud computing and internet of things: a survey. Fut. Gener. Comput. Syst. **56**, 684–700 (2015)
22. Shi, Y., Abhilash, S., Hwang, K.: Cloudlet mesh for securing mobile clouds from intrusions and network attacks. In: Proceedings of the 3rd IEEE International Conference of Mobile Cloud Computing, Services, and Engineering, pp. 109–118 (2015)
23. Cheng, C., Tay, W. P., Huang, G.B.: Extreme learning machines for intrusion detection. In: Proceedings of the Annual Inter Joint Conference on Neural Networks (IJCNN) (2012)
24. Ye, Z., Yu, Y.: Network intrusion classification based on extreme learning machine. In: Proceedings of the IEEE International Conference on Information and Automation, ICIA 2015, pp. 1642–1647. IEEE (2015)
25. Srimuang, W., Intarasothonchun, S.: Classification model of network intrusion using weighted extreme learning machine. In: Proceedings of the 12th International Joint Conference on Computer Science and Software Engineering, JCSSE 2015, pp. 190–194 (2015)
26. Cai, C., Pan, H., Cheng, G.: Fusion of BVM and ELM for anomaly detection in computer networks. In: Proceedings of the International Conference on Computer Science and Service System, CSSS, vol. 4, no. 2, pp. 1957–1960 (2012)
27. Muniyandi, P., Rajeswari, R., Rajaram, R.: Network anomaly detection by cascading K-means clustering and C.45 decision tree algorithm. Procedia Eng. **30**, 174–182 (2012)
28. Natesan, P.B., Gowrison, G.: Improving the attack detection rate intrusion detection using AdaBoost algorithm. J. Comput. Sci. **8**, 1041–1048 (2012)
29. Govindarajan, M., Chandrasekaram, R.M.: Intrusion detection using neural based hybrid classification methods. Comput. Net. **55**, 1662–1671 (2011)
30. Azeez, N.A., Bada, T.M., Misra, S., Adewumi, A., Van der Vyver, C., Ahuja, R.: Intrusion detection and prevention systems: an updated review. In: Sharma, N., Chakrabarti, A., Balas, V.E. (eds.) Data Management, Analytics and Innovation: Proceedings of ICDMAI 2019, vol. 1, pp. 685–696. Springer, Singapore (2020)
31. Azeez, N.A., Ayemobola, T.J., Misra, S., Maskeliūnas, R., Damaševičius, R.: Network intrusion detection with a hashing based Apriori algorithm using Hadoop MapReduce. Computer **8**(4), 86 (2019)
32. Alfa, A.A., Yusuf, I.O., Misra, S., Ahuja, R.: Enhancing stock prices forecasting system outputs through genetic algorithms refinement of rules-lists. In: Proceedings of 1st International Conference on Computing, Communications, and Cyber-Security, pp. 669–680 (2020)
33. Abayomi-Alli, A., Misra, S., Fernández-Sanz, L., Abayomi-Alli, O., Edun, A.R.: Genetic algorithm and Tabu search memory with course sandwiching for university examination timetabling. Intell. Autom. Soft Comput. **26**(3), 385–396 (2020)
34. Alfa, A.A., Misra, S., Bumojo, A., Ahmed, K.B., Oluranti, J., Ahuja, R.: Comparative analysis of optimisations of antecedents and consequents of fuzzy inference system rules lists using genetic algorithm operations. In: Chillarige, R.R., Distefano, S., Rawat, S.S. (eds.) Advances in Computational Intelligence and Informatics: Proceedings of ICACII 2019, pp. 373–379. Springer, Singapore (2020). https://doi.org/10.1007/978-981-15-3338-9_42