Design of Optimal Machine Learning based Cybersecurity Intrusion Detection Systems

Andino Maseleno

Institute of Informatics and Computing Energy, University Tenaga Nasional, Malaysia

Email: andino@uniten.edu.my

Abstract

Cybersecurity is the process of protecting critical systems and confidential data from digital attacks. With the advent of machine learning, cybersecurity systems can examine the patterns and learns them from preventing similar attacks and responds to fluctuating behavior. Cybersecurity intrusion detection system helps to detect the existence of intrusions in the network and achieves security in confidential data storage and transmission. In this view, this study designs an efficient cockroach optimization (CSO) with kernel extreme learning machine (KELM) model for cybersecurity intrusion detection. The proposed CSO-KELM model can accomplish cybersecurity by the detection and classification of intrusions. The proposed CSO-KELM technique encompasses a three-level process, namely preprocessing, classification, and parameter tuning. The design of the CSO algorithm for the appropriate selection of KELM parameters results in improved classification performance. For examining the betterment of the CSO-KELM technique, a series of experiments were performed on benchmark datasets. The experimental results pointed out the superiority of the CSO-KELM technique concerning several measures.

Keywords: Intrusion detection systems, Cybersecurity, Machine learning, Parameter tuning, CSO algorithm

1. Introduction

With the increasing in-depth integration of the Internet and social lives, the Internet is transforming how people work and learn and exposing ever more severe security risks. To detect different network attacks, mainly not previously seen attacks, is a significant challenge to be urgently resolved [1, 2]. Cybersecurity is a collection of processes and technologies developed for protecting data, computers, networks, and programs from unauthorized access, alteration, or destruction and attacks. A network security scheme is composed of a computer security system and network security system. Such systems consist of software, firewalls, antivirus, and intrusion detection system (IDS). IDS helps identify, discover and determine unauthorized system behaviors like copying, use, destruction, and modification.

The number of data created all the time surpasses petabytes, and this consists of the traces of those internet users leaves once they access a network, website, or mobile application. Such traces or "log data" are increasing daily since multiple sources are creating them. The efficient use of log information could benefit finding malicious connections, therefore securing the network from upcoming attacks. The requirement for a realtime detection scheme that could measure the number of information being consumed and acts rapidly according to the response time could provide an edge on these kinds of attacks. To signal and detect abnormal activity, an IDS method has been employed. [3] determined IDS as a scheme that delivers real-time, reliable network traffic analyses to determine whether the network is being hacked or not. Cloud computing provides applications, processing, power, storage, and services on the

DOI: https://doi.org/10.54216/JCIM.000103

Internet [4]. The more familiar cloud service provider consists of Microsoft Azure and Google Cloud Platform, Amazon (AWS). The usage of on-premises and public cloud platforms is increasing. To process and ingest large amount of information, big data tools like Apache spark [5] and Apache Hadoop [6] were utilized. Hadoop is a publicly available platform that was the director in the Big Data domain. It uses MapReduce for processing large amounts of information and stores it in the HDFS. Spark is shown to be 10× to 100× quicker than Hadoop since it uses in-memory computations. Spark supports graph processing, SQL queries, streaming data, and machine learning; it could also run on Hadoop or utilize HDFS to store information.

Breaches of security consist of internal intrusions and external intrusions. There are three significant network analyses for IDS: misuse-based, called anomaly-based, signature-based, and hybrid. Misuse based detection technique aims to identify attacks through the signature of this attack [7]. They are utilized for known types of attacks without making a huge amount of false alarms. But, the administrator must automatically upgrade the dataset signatures and rules. New (zero-day) attacks might not be identified according to misused technology. Anomaly-based technique studies the typical system and network behaviors and finds abnormalities as deviation from standard behaviors. They are attractive due to their ability to identify zero-day attacks. Another advantage is that the profile of usual activities is personalized for each network, system, or application, making it challenging for the hackers to know which activity could implement undiscovered. In addition, the information where anomaly-based technique alerts (new attack) are utilized for defining the signature for misuse detector. The anomaly-based technique's major drawback is the possibility for a higher false alarm rate since formerly hidden scheme behavior could be considered abnormalities. Hybrid detections combine misuse and anomaly detection [8]. It is utilized to increase the recognition rate of known intrusion and decrease the false-positive rates of unidentified attacks. Many ML/DL techniques are hybrid. The ML and AI methods are extensively utilized for constituting an intelligent and effective IDS scheme. But, authors usually train and develop their ML-based security scheme with network traces attained from open-source datasets. Because of malware changes and evolution in the attack strategy, this dataset fails to secure the system from novel kinds of attacks. Therefore, the standard dataset must be periodically upgraded.

This study designs an efficient cockroach optimization (CSO) with kernel extreme learning machine (KELM) model for cybersecurity intrusion detection. The proposed CSO-KELM model has the ability to accomplish cybersecurity by the detection and classification of intrusions. The proposed CSO-KELM technique encompasses a three-level process, namely preprocessing, classification, and parameter tuning. The design of the CSO algorithm for the appropriate selection of KELM parameters results in improved classification performance. For examining the betterment of the CSO-KELM technique, a series of experiments were performed on benchmark datasets.

2. Related works

Wu et al. [9] proposed a new data entropy-based system that uses several messages as a sliding window. By enhancing the sliding window approach and improving the decision condition, the false positive rate is reduced, and the recognition performance increases. In [10], DNN was used for predicting the attack on the N-IDS method. A DNN using 0.1 rates of learning is employed and operates for thousand numbers of the epoch as well as KDDCup-`99' datasets were utilized for training and benchmarking the networks. The training is based on similar datasets with various traditional ML models for comparison purposes, and the DNN layer ranges from one to five.

Gupta [11] utilized Apache Spark, a big data processing tool to process huge network traffic information. This work presented an architecture where a familiar FS model has been developed to select the essential characteristics. Later, the classification-based IDS system is utilized to efficiently and fast recognize intrusion in the huge network traffics. Alom and Taha [12] demonstrate a new network IDS for cybersecurity with an unsupervised DL technique. In this application, the input sample is numerically encoded and employed un-supervised DL methods named AE and RBM for dimensionality reduction and feature extraction. Next, iterative k-means clustering is used to cluster on low dimensional space with three characteristics. Additionally, UELM is utilized for network IDS in this work.

Wong et al. [13] make several contributions to progress study in this field. Initially, review the level of supports for SCADA protocol in famous publicly available IDS sources. Next, choose a specific IDS,

DOI: https://doi.org/10.54216/JCIM.000103

Suricata, and improve it to include support to detect threats against the SCADA system operating the EtherNet/IP (ENIP) industrial control protocol. Lin et al. [14] analyze the IDS framework and RA method in edge computing. Significantly, the presented method is developed for facilitating heterogeneous resource-demanding allocation and many assets sharing. A common edge computing IDS framework is proposed and employed as the source for this method to allocate resources. Next, an SDMMF allocation is utilized that is systematically proved to fulfill all hierarchical RA property and multi-layer RA schemes.

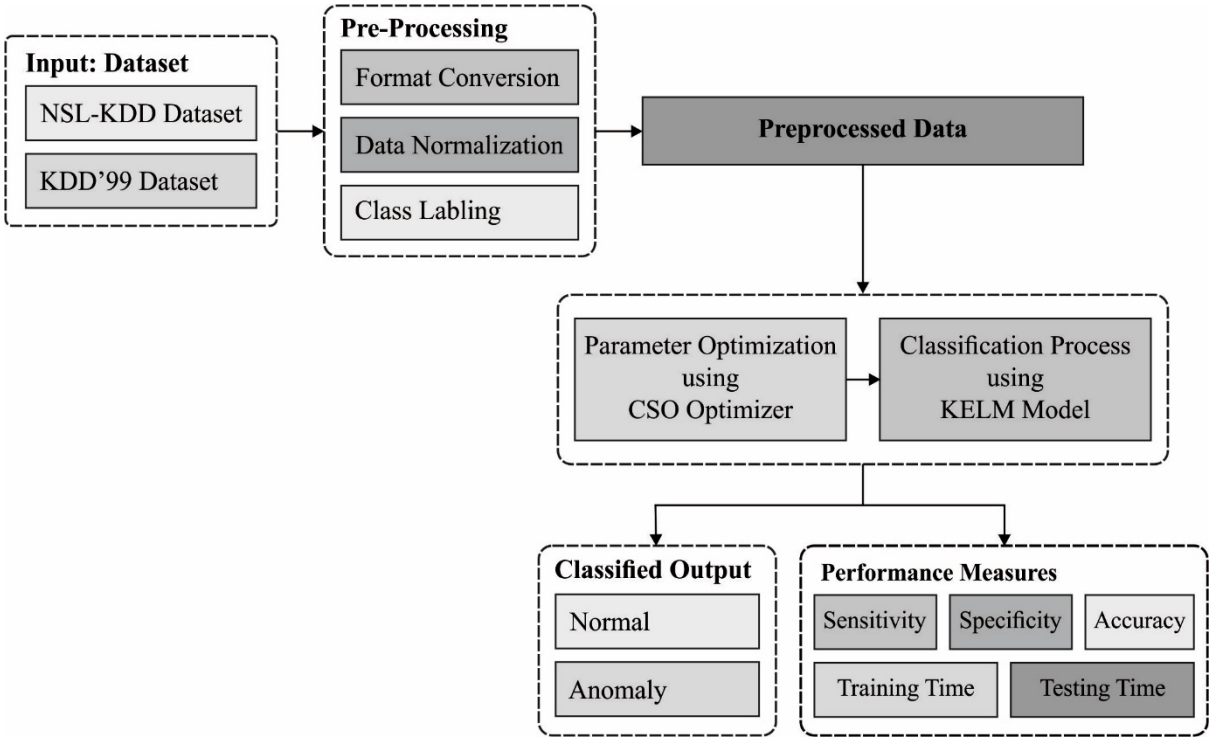

Fig. 1. The overall process of CSO-KELM model

3. The Proposed Model

This study has developed a CSO-KELM model for cybersecurity intrusion detection. The proposed CSO-KELM model can accomplish cybersecurity by the detection and classification of intrusions. The proposed CSO-KELM technique encompasses a three-level process: preprocessing, KELM-based classification, and CSO-based parameter tuning. Fig. 1 demonstrates the overall working process of the CSO-KELM model.

3.1 Level I: Process involved in KELM based Classification Model

The KELM model receives the preprocessed data as input and performs the classification process to determine the existence of intrusions in the network. KELM is a single hidden layer feedforward neural network (SLFN). In standard FFNN, trained speed was slow and fell as to local minimum, and chosen learning rate has been sensitive. The ELM arbitrarily makes the linked weight amongst the input and hidden layers and threshold of hidden layer source for obtaining unique optimum solutions [15, 16]. To N arbitrarily varying instances (x_i, o_i), where $x_i = [x_{i1}, x_{i2}, \cdots, x_{im}]^T \in R^n$ and $o_i = [o_{i1}, o_{i2}, \cdots, o_{im}]^T \in R^m$, the output of ELM with L hidden neuron has been demonstrated as:

$$\Theta(x_i) = \sum_{i=1}^{L} \beta_i g\left(a_i \cdot x_j + b_i\right) = o_j, j = 1,2, \ldots, N, \quad (1)$$

Where $g(\cdot)$ implies the activation function of hidden layers, $a_i = [a_{i1}, a_{i2}, \cdots, a_{im}]^T$ demonstrates the input weight vectors, $\beta_i = [\beta_{i1}, \beta_{i2}, \cdots, \beta_{im}]^T$ refers to the output weight vectors, and b_i represents the biases.

DOI: https://doi.org/10.54216/JCIM.000103

Eq. (1) has been simplified as

$$H\beta = T, \tag{2}$$

where

$$H = \begin{bmatrix} h(x_1) \\ \vdots \\ h(x_N) \end{bmatrix} = \begin{bmatrix} g(a_1 \cdot x_1 + b_1) & \cdots & g(a_L \cdot x_1 + b_L) \\ \vdots & \vdots & \vdots \\ g(a_1 \cdot x_N + b_1) & \cdots & g(a_L \cdot x_N + b_L) \end{bmatrix}_{N \times L}, \tag{3}$$

$$\beta = \begin{bmatrix} \beta_1^T \\ \vdots \\ \beta_L^T \end{bmatrix}_{L \times m} \quad and \; T = \begin{bmatrix} t_1^T \\ \vdots \\ t_L^T \end{bmatrix}_{L \times m}, \tag{4}$$

Where H refers to the ELM hidden layer outcome matrix. In the trained, a network of ELM is assumed as determining an appropriate set of $\hat{a}$, $\hat{b}$, and $\hat{\beta}$ sufficient:

$$\left\| H(\hat{a}, \hat{b})\hat{\beta} - T \right\| = \min_{a,b,\beta} \|H(a,b)\,\beta - T\|, \tag{5}$$

The regularization coefficients C has been established and the regularized least square solution has been attained [17]:

$$\hat{\beta} = H^T(I/C + HH^T)^{-1}T, \tag{6}$$

Therefore, the resultant function of the ELM method has been changed as to:

$$\Theta(x) = h(x)\hat{\beta} = H\hat{\beta}, \tag{7}$$

The KELM relates the ELM technique with kernel functions. The idea of kernel function has been to map the input spatial instance data to high dimension feature spaces and change the inner product function from the changed high dimension space with kernel function during the original input spaces. Fig. 2 illustrates the framework of the KELM model.

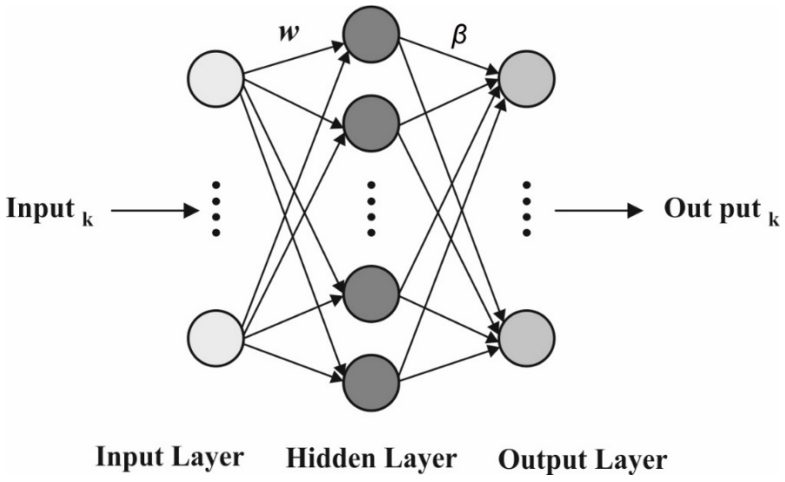

Fig. 2. Structure of KELM

During the KELM, HH^T of Eq. (6) has generated as:

$$HH^T(i,j) = K(x_i, x_j), \tag{8}$$

Afterward, it cloud be assume Eq. (9),

$$HH^T = \Omega_{ELM} = h(x_i) \cdot h(x_j) = K(x_i, x_j), \tag{9}$$

Where $K(\cdot,\cdot)$ represents the kernel function. It could be evident that KELM output function $\Theta(x)$ and resultant layer β as:

DOI: https://doi.org/10.54216/JCIM.000103

$$\begin{cases} \Theta(x) = h(x)\,.\,\beta = \begin{bmatrix} K(x,x_1) \\ \vdots \\ K(x,x_N) \end{bmatrix} (I/C + \Omega_{ELM})^{-1}T\,, \\ \beta = (I/C + \Omega_{ELM})^{-1}T \end{cases} \tag{10}$$

It can be worth noticing that Gaussian kernel function has been utilized during this work based on the Mercer theorem as:

$$K(x_i, x_j) = \mathrm{e}^{-\frac{\|x_i - x_j\|^2}{\gamma^2}}, \tag{11}$$

Where γ^2 implies the kernel function parameter, so there are two parameters required that are adjusted in KELM, and the accuracy of KELM is enhanced by altering C and γ.

3.2 Level II: Process involved in CSO based Parameter Tuning Technique

For optimally modifying the parameters involved in the KELM model, the CSO algorithm is introduced to it. The CSO technique is a population-based global optimized technique; it can be established with the summary of inertial weight [18, 19]. In the CSO techniques simulator, cockroach behaviors are chase-swarming, dispersing, and ruthless behaviors. The CSO technique is provided here. The chase swarming behavior can be defined as follows:

$$x_j = \{^{x_i + step \cdot rand \cdot (p_i - x_i)}_{x_i + step \cdot rand \cdot (p_g - x_i)}, x_i = p_i x_i \neq p_i, \tag{12}$$

where x_i refers to the cockroach place, the step has been set value, $rand$ signifies the arbitrary number within [0,1], p_i implies the personal optimum place, and p_g refers to the global optimum place. Assume

$$p_i = Opt_j\{x_j, |x_j - x_j| \leq visual\}, \tag{13}$$

where visual perception distance has been constants. $j = 1,2, \dots, N$ and $i = 1,2, \dots, N$

$$p_g = Opt_i\{x_i\}. \tag{14}$$

The dispersion behavior can be represented using Eq. (15):

$$x_i = x_i + rand(1, D), i = 1,2, \dots, N, \tag{15}$$

where rand $(1, D)$ represents the D-dimension arbitrary vector fixed in a particular range.

Ruthless behavior is defined as follows.

$$x_k = p_g, \tag{16}$$

where k demonstrates the arbitrary integer in [1, N] and p_g refers to the global optimum place. MCSO encompasses CSO is an overview of inertial weight from chase-swarming behavior as demonstrated under. The chase-swarming behavior

$$x_i = \{^{w \cdot x_i + step \cdot rand \cdot (p_i - x_i)}_{w \cdot x_i + step \cdot rand \cdot (p_g - x_i)}, x_i = p_i x_i \neq p_i, \tag{17}$$

where w refers to the inertial weight that is constant.

The constriction factor (CF) has been established to prevent swarm explosion from PSO; the PSO is the most present and popular SI technique. It can be presented as a constriction as

$$\chi = \frac{2}{|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}|}, \tag{18}$$

where $\varphi = c_1 + c_2, \varphi > 4.0$, $c1$ indicates the recognition element, and $c2$ denotes the social issue. Likewise, the CF was regarded as controlling cockroach movement in swarming procedures to avoid swarm explosions. It utilized a stochastic constriction factor (SCF) rather than a constant constriction. The SCF permits the generation of distinct values as CF from all iterations.

DOI: https://doi.org/10.54216/JCIM.000103

The SCF uses for maintaining the constancy of the swarm, improves local and global search, and increases the speed and convergence of the technique. This technique employed little CPU time in seconds for solving multi-dimensional issues and attain optimum outcomes. The chase-swarming behavior, (1) of CSO, has been altered with the overview of SCF ξ. ξ arbitrarily take values amongst 0 and 1 from all iterations. ξ controls complete cockroach movement, not only cockroach place as with inertial weight w of CSO. The chase-swarming formula present develops

$$x_i = \begin{cases} \xi(x_i + step \cdot rand \cdot (p_i - x_i)), x_i = p_i \\ \xi\left(x_i + step \cdot rand \cdot (p_g - x_i)\right), x_i \neq p_i \end{cases}, \quad (19)$$

The mathematical steps to SCCSO are demonstrated in Algorithm 1 and their computational phases are provided as follows.

1) Initialization cockroach swarm with uniform distributed arbitrary number and fixed every parameter with a value.
2) Determine p_i and p_g
3) Display chase-swarming behavior
4) Demonstration dispersion behavior
5) Show ruthless behavior
6) Repeat the loop till the end condition has been obtained.

The efficiency of the presented technique on famous higher dimensional benchmarks is estimated.

4. Performance Validation

The performance validation of the CSO-KELM technique takes place on the applied KDDCup 99 and NSL-KDD datasets.

Table 1 showcases the comparative result analysis of the CSO-KELM model under the KDDCup99 dataset in different measures with existing techniques.

Table 1 Results analysis of CSO-KELM technique on KDDCup 99 dataset

Method	Accuracy	Sensitivity	Specificity	Training Time (Min)	Testing Time (Min)
CSO-KELM	0.946	0.996	0.999	1.054	0.167
Logistic Regression	0.916	0.900	0.983	5.338	0.261
SVM	0.921	0.922	0.920	8.841	0.317
Naive Bayes	0.915	0.994	0.586	1.565	0.234
Random Forest	0.921	0.909	0.974	2.662	0.272
GB Tree	0.914	0.894	0.998	5.746	0.332

Fig. 3 depicts the analysis of the result of the CSO-KELM technique on the benchmark KDDCup99 dataset. The figure reported that the GBTree technique has resulted in ineffective outcomes with the accuracy, sensitivity, and specificity of 0.914, 0.894, and 0.998, respectively. Also, the NB model has gained slightly enhanced performance with the accuracy, sensitivity, and specificity of 0.915, 0.994, and 0.586, respectively. Followed by, the LR model has resulted in a moderate outcome with the accuracy, sensitivity, and specificity of 0.916, 0.90, and 0.983, respectively. Moreover, the SVM and RF model has resulted in a reasonable outcome with a closer accuracy of 0.921 and 0.921. But the proposed CSO-KELM technique has gained proficient results with the accuracy, sensitivity, and specificity of 0.946, 0.996, and 0.999, respectively.

DOI: https://doi.org/10.54216/JCIM.000103

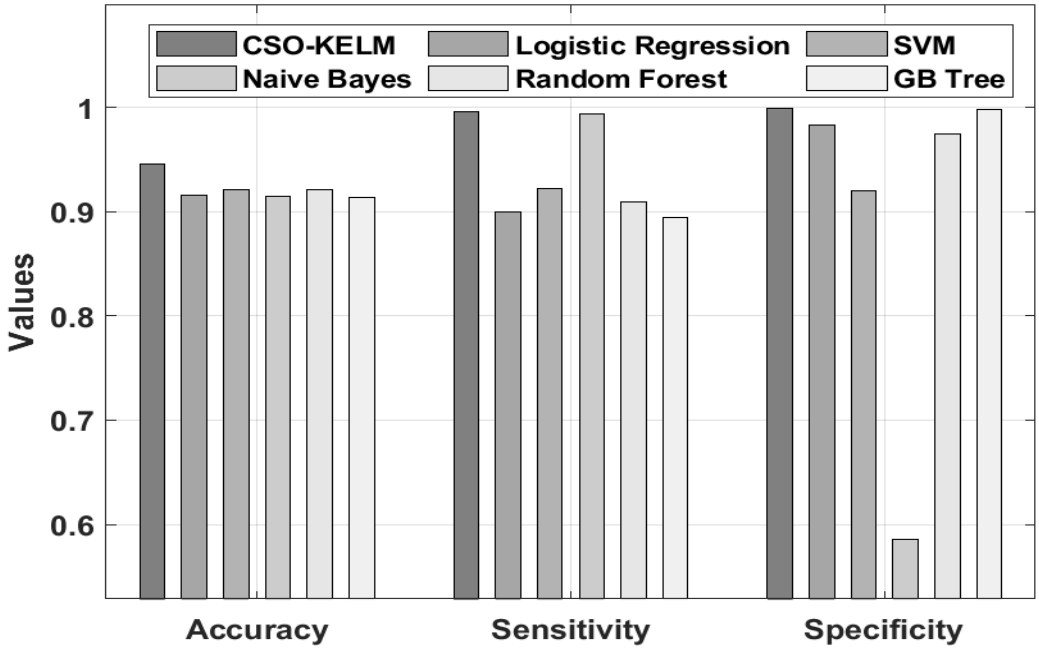

Fig. 3. Results analysis of CSO-KELM technique on KDDCup 99 dataset

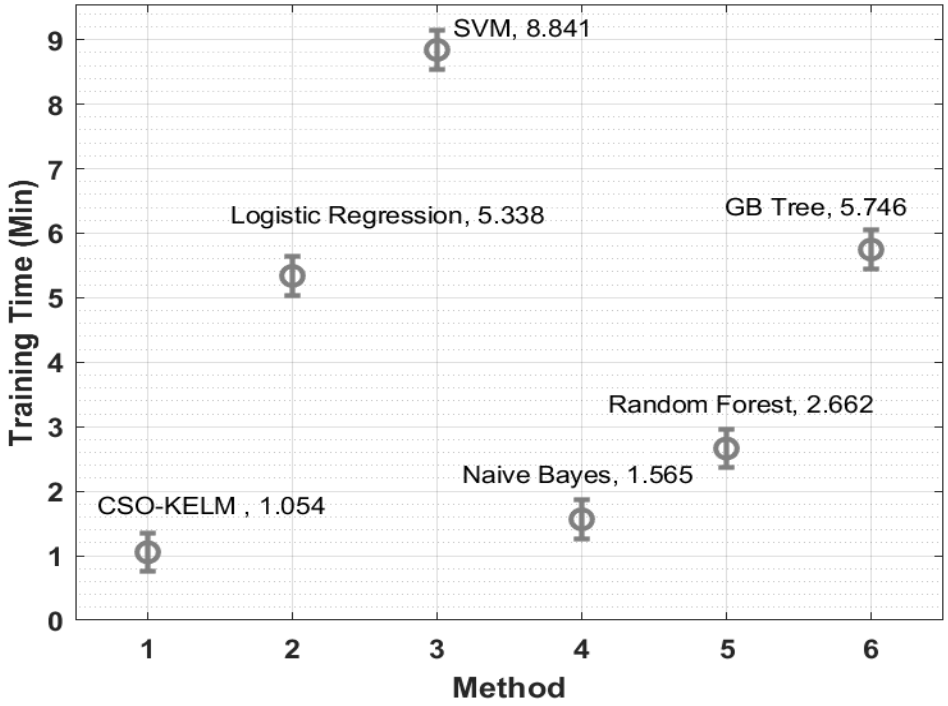

Fig. 4. TNT Analysis of CSO-KELM technique on KDDCup 99 dataset

Fig. 4 demonstrates the training time (TNT) analysis of the CSO-KELM technique with recent models. The results portrayed that the SVM technique has gained worse outcomes with the highest TNT of 8.841minutes, whereas the GBT, LR, and RF techniques have required a slightly reduced TNT of 5.746minutes, 5.338minutes, and 2.662minutes. In line with this, the NB model has accomplished a near-optimal TNT of 1.565minutes. However, the CSO-KELM technique has resulted in an effective outcome with the least TNT of 1.054minutes.

Fig. 5 depicts the testing time (TST) analysis of the CSO-KELM manner with recent approaches. The outcomes demonstrated that the GBT technique had reached least effect with the highest TST of

DOI: https://doi.org/10.54216/JCIM.000103

0.332minutes, whereas the SVM, RF, and LR techniques have required a slightly reduced TST of 0.317minutes 0.272minutes, and 0.261minutes. Along with that, the NB system has accomplished a near-optimum TST of 0.234minutes. But, the CSO-KELM manner has resulted in an effective outcome with a minimum TST of 0.167minutes.

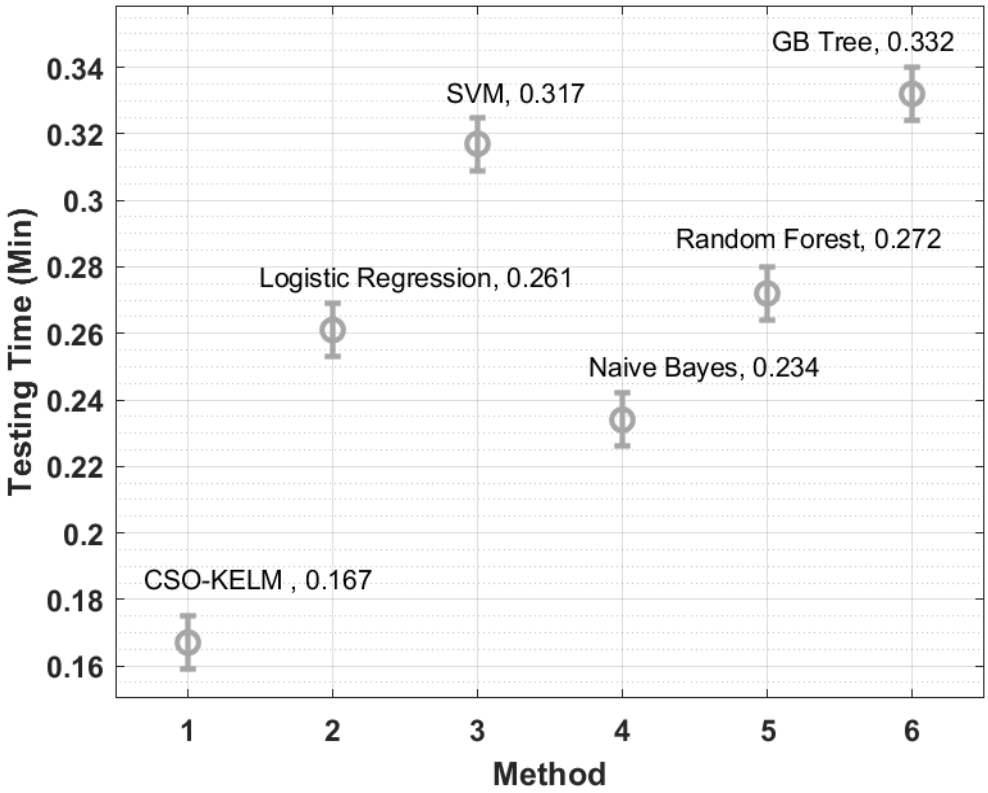

Fig. 5. TST analysis of CSO-KELM technique on KDDCup 99 dataset

Table 2 illustrates the comparative outcome analysis of the CSO-KELM technique under the NSL-KDD dataset concerning different measures with existing algorithms.

Fig. 6 showcases the outcomes analysis of the CSO-KELM manner on the benchmark NSL-KDD dataset. The figure stated that the NB technique has resulted in ineffective outcomes with the accuracy, sensitivity, and specificity of 0.354, 0.007, and 0.813. Also, the SVM method has attained somewhat increased performance with the accuracy, sensitivity, and specificity of 0.405, 0.585, and 0.169 correspondingly. At the same time, the LR system has resulted in a moderate outcome with the accuracy, sensitivity, and specificity of 0.744, 0.600, and 0.933 correspondingly. In addition, the RF and GBT technique has resulted in a certain reasonable outcome with the closest accuracy of 0.744 and 0.766. Finally, the projected CSO-KELM technique has obtained proficient outcomes with the corresponding accuracy, sensitivity, and specificity of 0.844, 0.757, and 0.992.

DOI: https://doi.org/10.54216/JCIM.000103

Table 2 Results analysis of CSO-KELM technique on NSL-KDD dataset

Method	Accuracy	Sensitivity	Specificity	Training Time (Min)	Testing Time (Min)
CSO-KELM	0.844	0.757	0.992	0.010	0.004
Logistic Regression	0.744	0.600	0.933	0.107	0.022
SVM	0.405	0.585	0.169	0.174	0.026
Naive Bayes	0.354	0.007	0.813	0.015	0.006
Random Forest	0.810	0.693	0.964	0.113	0.037
GB Tree	0.766	0.609	0.973	0.146	0.028

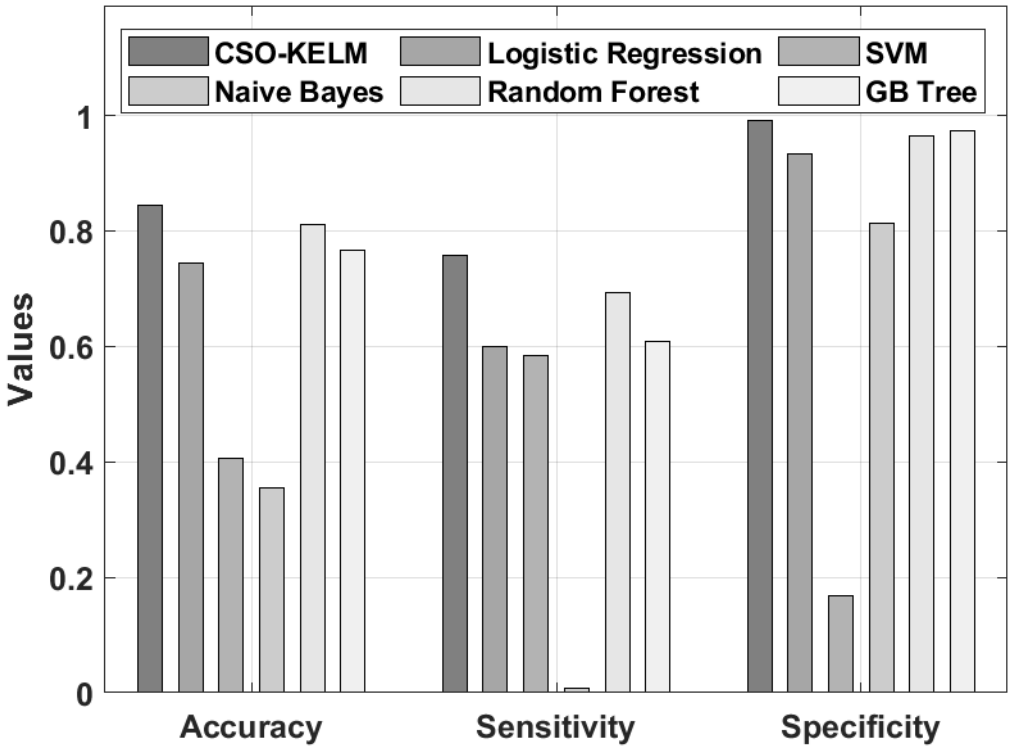

Fig. 6. Results analysis of CSO-KELM technique on NSL-KDD dataset

Fig. 7 depicts the TNT analysis of the CSO-KELM algorithm with state-of-art techniques. The outcomes demonstrated that the SVM technique has achieved minimal outcome with the highest TNT of 0.174minutes, whereas the GBT, RF, and LR techniques have required a slightly minimum TNT of 0.146minutes 0.113minutes, and 0.107minutes. Similarly, the NB model has accomplished a near-optimal TNT of 0.015minutes. At last, the CSO-KELM technique has resulted in an effective outcome with the least TNT of 0.01minutes.

Fig. 8 outperforms the TST analysis of the CSO-KELM manner with existing models. The results showcased that the RF technique has reached a minimum result with the maximum TST of 0.037minutes, whereas the GBT, SVM, and LR techniques have needed a somewhat decreased TST of 0.028minutes, 0.026minutes, and 0.022minutes. Also, the NB system has accomplished a near-optimal TST of 0.006minutes. However, the CSO-KELM methodology has resulted in an effective outcome with a minimum TST of 0.004minutes.

DOI: https://doi.org/10.54216/JCIM.000103

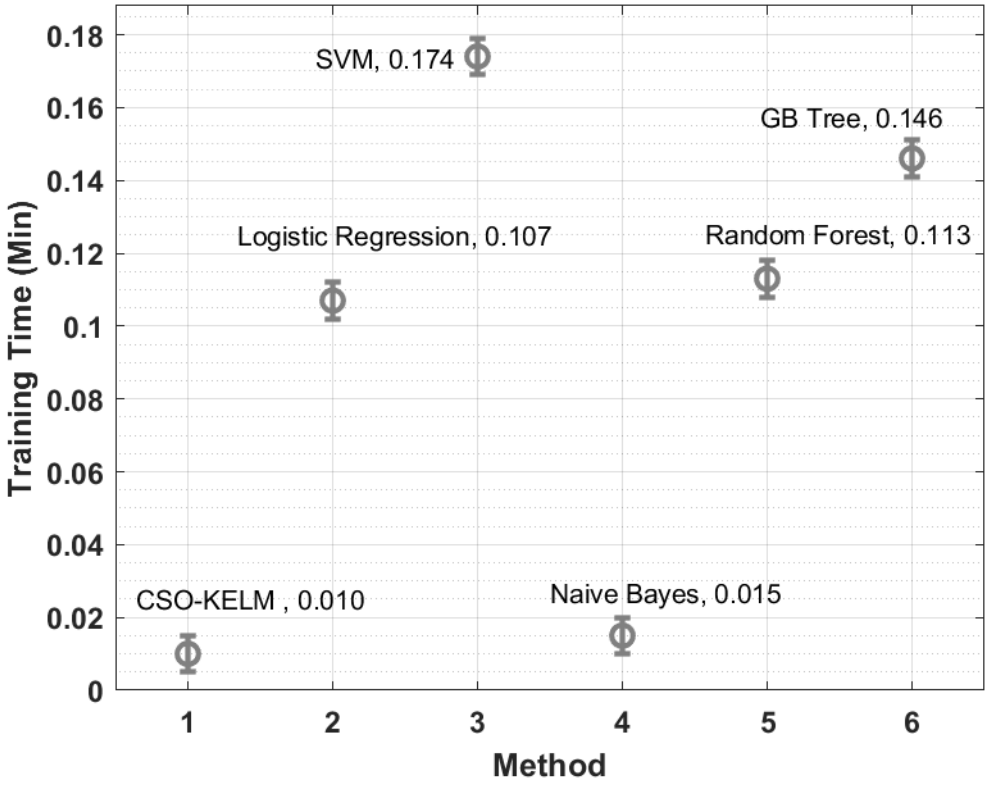

Fig. 7. TNT Analysis of CSO-KELM technique on NSL-KDD dataset

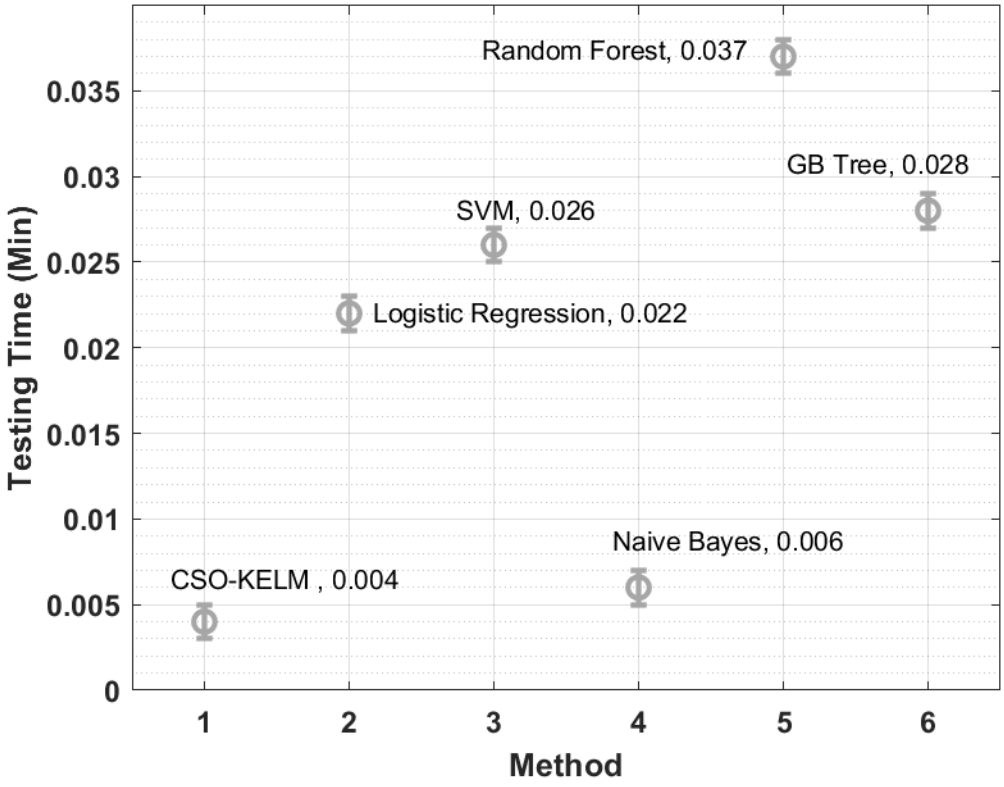

Fig. 8. TST analysis of CSO-KELM technique on NSL-KDD dataset

DOI: https://doi.org/10.54216/JCIM.000103

5. Conclusion

This study has developed a CSO-KELM model for cybersecurity intrusion detection. The proposed CSO-KELM model can accomplish cybersecurity by the detection and classification of intrusions. The proposed CSO-KELM technique encompasses a three-level process, namely preprocessing, classification, and parameter tuning. The design of the CSO algorithm for the appropriate selection of KELM parameters results in improved classification performance. For examining the betterment of the CSO-KELM technique, a series of experiments were performed on benchmark datasets. The experimental results pointed out the superiority of the CSO-KELM technique concerning several measures. In the future, feature selection and feature reduction techniques can be introduced to improve intrusion detection performance.

References

[1] Buczak, A.L. and Guven, E., 2015. A survey of data mining and machine learning methods for cyber security intrusion detection. IEEE Communications surveys & tutorials, 18(2), pp.1153-1176.

[2] Abubakar, A.I., Chiroma, H., Muaz, S.A. and Ila, L.B., 2015. A review of the advances in cyber security benchmark datasets for evaluating data-driven based intrusion detection systems. Procedia Computer Science, 62, pp.221-227.

[3] Myers, S.; Musacchio, J.; Bao, N. Intrusion Detection Systems: A Feature and Capability Analysis; Baskin School of Engineering: Santa Cruz, CA, USA, 2010.

[4] Mukkamala, S., Sung, A., Abraham, A. and Vemuri, V.R., 2005. Cyber security challenges: Designing efficient intrusion detection systems and antivirus tools. Vemuri, V. Rao, Enhancing Computer Security with Smart Technology.(Auerbach, 2006), pp.125-163.

[5] Abraham, A., Grosan, C. and Chen, Y., 2005. Cyber security and the evolution in intrusion detection systems. Journal of Engineering and Technology, ISSN, pp.0973-2632.

[6] Singh, P., Garg, S., Kumar, V. and Saquib, Z., 2015, August. A testbed for SCADA cyber security and intrusion detection. In 2015 International Conference on Cyber Security of Smart Cities, Industrial Control System and Communications (SSIC) (pp. 1-6). IEEE.

[7] Yang, Y., Xu, H.Q., Gao, L., Yuan, Y.B., McLaughlin, K. and Sezer, S., 2016. Multidimensional intrusion detection system for IEC 61850-based SCADA networks. IEEE Transactions on Power Delivery, 32(2), pp.1068-1078.

[8] Borkar, A., Donode, A. and Kumari, A., 2017, November. A survey on Intrusion Detection System (IDS) and Internal Intrusion Detection and protection system (IIDPS). In 2017 International conference on inventive computing and informatics (ICICI) (pp. 949-953). IEEE.

[9] Wu, W., Huang, Y., Kurachi, R., Zeng, G., Xie, G., Li, R. and Li, K., 2018. Sliding window optimized information entropy analysis method for intrusion detection on in-vehicle networks. IEEE Access, 6, pp.45233-45245.

[10] Vigneswaran, R.K., Vinayakumar, R., Soman, K.P. and Poornachandran, P., 2018, July. Evaluating shallow and deep neural networks for network intrusion detection systems in cyber security. In 2018 9th International conference on computing, communication and networking technologies (ICCCNT) (pp. 1-6). IEEE.

[11] Gupta, G.P. and Kulariya, M., 2016. A framework for fast and efficient cyber security network intrusion detection using apache spark. Procedia Computer Science, 93, pp.824-831.

[12] Alom, M.Z. and Taha, T.M., 2017, June. Network intrusion detection for cyber security using unsupervised deep learning approaches. In 2017 IEEE national aerospace and electronics conference (NAECON) (pp. 63-69). IEEE.

[13] Wong, K., Dillabaugh, C., Seddigh, N. and Nandy, B., 2017, April. Enhancing Suricata intrusion detection system for cyber security in SCADA networks. In 2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE) (pp. 1-5). IEEE.

[14] Lin, F., Zhou, Y., An, X., You, I. and Choo, K.K.R., 2018. Fair resource allocation in an intrusion-detection system for edge computing: Ensuring the security of Internet of Things devices. IEEE Consumer Electronics Magazine, 7(6), pp.45-50.

[15] Iosifidis, A., Tefas, A. and Pitas, I., 2015. On the kernel extreme learning machine classifier. Pattern Recognition Letters, 54, pp.11-17.

DOI: https://doi.org/10.54216/JCIM.000103

[16] Liu, X., Wang, L., Huang, G.B., Zhang, J. and Yin, J., 2015. Multiple kernel extreme learning machine. Neurocomputing, 149, pp.253-264.
[17] Lu, H., Du, B., Liu, J., Xia, H. and Yeap, W.K., 2017. A kernel extreme learning machine algorithm based on improved particle swam optimization. Memetic Computing, 9(2), pp.121-128.
[18] Kwiecień, J. and Pasieka, M., 2017. Cockroach swarm optimization algorithm for travel planning. Entropy, 19(5), p.213.
[19] Obagbuwa, I.C. and Abidoye, A.P., 2016. Binary cockroach swarm optimization for combinatorial optimization problem. Algorithms, 9(3), p.59.

DOI: https://doi.org/10.54216/JCIM.000103