A Hybrid Convolution Neural Network with Binary Particle Swarm Optimization for Intrusion Detection

Hamza Turabieh

Department of Information Technology,Taif University, Taif, Saudi Arabia
h.turabieh@tu.edu.sa

***Abstract*—Many companies start using could systems and services to increase their productivity and decrease the cost by migrating their applications, infrastructures, and data to external cloud platforms. Using cloud systems leads to raise the number of attacks on such systems. Protecting these cloud platforms from different attacks becomes an essential task using Intrusion detection systems (IDS). In general, IDS is used to detect normal or abnormal network traffic packets. In this paper, we proposed a hybrid intelligent IDS system based on a one-dimensional Convolution Neural Network (1D-CNN) and Binary Particle swarm Optimization (BPSO). BPSO is employed as a wrapper feature selection to determine the most valuable features and reduce the high dimensionality of collected data. While 1D-CNN is employed as a binary classifier. We adopted a real dataset called UNSW-NB15 to evaluate the proposed hybrid IDS. The obtained results show the proposed system can detect normal and abnormal packets with an accuracy equals 94.3%.**

I. Introduction

Several companies start using cloud computing systems to operate their applications and manipulate their data over the internet. All could activities are accessed remotely over the internet. In general, could computing systems help companies and end users from several issues related to installing, maintaining, and securing their applications, infrastructures and data [1], [2]. Protecting cloud computing systems from abnormal traffic is needed. Several Cloud Service Providers (CSP) start developing intelligent IDS to prevent illegitimate entry to cloud computing systems. In simple, the main task of IDS is to distinguish normal and abnormal traffics [3].

There are two types of IDS: Host intrusion detection systems (HIDS) and network intrusion detection systems (NIDS) [4]. HIDS control all kind of attacks inside a local network (e.g., LAN) by monitoring and analyzing all traffics comes from local machines in order to detect abnormal traffic or behavior [5]. While NIDS monitors all traffic comes from outside network (e.g. WAN). Both types reports all abnormal behaviors and illegitimate activity to the system administrator and execute a set of protection activities to stop such attacks or abnormal behaviors [3].

To keep cloud computing systems healthy and protected, IDS examine each packet and classify it to normal or abnormal one [6]. In general, IDS works in two methods: Anomaly and misuse (signature)-based detection. Anomaly method tries to detect any abnormal traffic that is deviates from the normal one. While misuse method tries to detect any abnormal traffic based on previous patterns of abnormal traffic patterns [7]. Building IDS is a complex process since the process of detecting intrusion is considered as NP-Hard problem [8], [9]. As a result, building intelligent IDS based on Machine Learning (ML) methods and Soft Computing (SC) is needed.

Each IDS should have three attributes to secure any system which are: Data confidentiality, Data integrity, and Data availability [10]. Data confidentiality means that sensitive data cannot be accessed by untrusted user. Data integrity means that the data should be consistent and not tampered while transmission process. Data availability means that the data can be accessed any where any time securely. Figure 1 explores the detection and response processes that is proposed by Denning [11]. The detection process cannot be directly executing based on the data available when the main task of IDS it to classify all activity happen on the network due to several reasons such as: huge amount of traffic data, unequal distribution of data, shortage of available knowledge to recognize new types of attacks, and shortage of stability [11]. In addition to that, IDS did not have the ability to control a large number of alarms, which needs more computational time and reduce the detection rate [12]. Therefore, it is important to reduce the data dimensionality before building IDS. To achieved this, FS methods can reduce the data dimensionality and enhance the overall performance of IDS.

Several exits IDS try to build an intelligent classification system based on a set of historical data. Since network traffic data is considered a high dimensionality data, several researcher employed Feature Selection (FS) methods to enhance the data quality and reduce the dimensionality [13]. For example, Sarvari et al. [14] employed a modified Cuckoo Search Algorithm (CSA) as a wrapper FS and Evolutionary Neural Network (ENN) as a classification method. Thakkar and Lohiya [15] applied seven different ML classifiers(i.e., Support Vector Machine (SVM), Naïve Bayes (NB), Decision Tree (DT), Random Forest (RF), k-nearest neighbours (kNN), Logistic Regression (LR), and Artificial Neural Networks (ANN)) to classify intrusions. The authors employed two FS methods (i.e., Chi-Square, Information Gain (IG), and Recursive Feature Elimination (RFE)). Almomani [16] applied four warpper feature selection methods (i.e., particle swarm optimization (PSO), grey wolf optimizer (GWO), firefly optimization (FFA) and genetic algorithm (GA)) to select the most valuable features from intrusion detection dataset called UNSW-NB15. The author applied two ML classifiers (i.e., SVM and J48).

The motivation of this paper is to investigate the perfor-

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

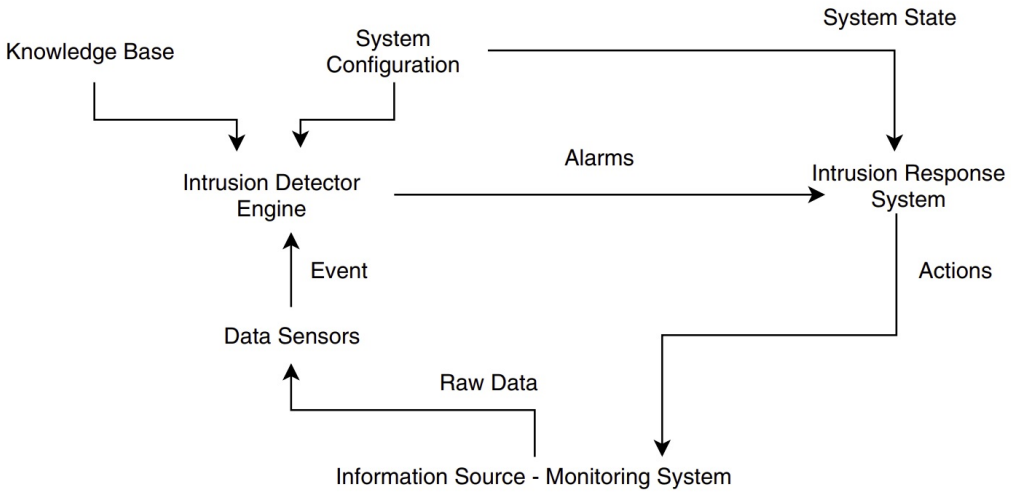

Fig. 1. Processes of intrusion detection and response system [11].

mance of BPSO as a wrapper feature selection method with CNN as a ML classifier to build an intelligent IDS.

The rest of this paper organized as following: Section II explores the related works of ML and FS in the area of IDS. SectionIII presents the proposed method. Section IV presents the dataset (i.e., UNSW-NB15) used in this paper. Section VI explores the obtained results and a deep discussion about it. Finally, Section VII presents the finding and future work of this paper.

II. Related works

FS and ML have been widely used in different domains successfully, such as software fault prediction [17], bioinformatics [18], text categorization [19], and intrusion detection [16]. There are many published papers that employ ML in classifying abnormal traffics in an IDS. In general, ML methods try to find a robust model based on training data to classify each packet. Figure 2 show a general pictorial diagram for IDS based on ML methods. In this section, we will highlight the main works that is related to intrusion detection. Aneetha et al. [20] applied an online intelligent analysis system that handles data collected from several devices. In general, IDS can provide a robust solution in order to provide a secure network against external attacks. IDS works as an elaborate device to control and monitor all network traffic and handle malicious traffic.

Most of the IDS is evaluated based on two criteria which are accuracy of detecting class and stability of detection in each class [21]. Detecting attacks is not an easy task due to large number of packets received. Therefore, IDS should be able to handle the coming data accurately and reduce the data dimensionally [22]. ML consists of several algorithms such as (SVM, DT, ANN, CNN, etc.) that makes computers to learn from data to identify patterns (e.g., identify attacks) . ML first tries to find out the best model that fit the training data by reducing the classification error between actual and estimated output. The trained model is simulated over a new data called testing data. Building a robust model depends on several attributes such as: learning algorithm, size of input data, and quality of input data [22]. To enhance the overall performance of ML model(s), removing irrelevant and redundant features will enhance the learning process which reflect on the overall performance of trained model.

Sivatha et al. [23] proposed a lightweight IDS based on a hybrid method between Genetic algorithm (GA), ANN and DT. GA is used as a wrapper feature selection method, while ANN and DT are hybridized in a single classifier called neurotree. The authors employed their proposed method over NSL-KDD datasets. The GA selects 14 valuable features out of 41. The proposed method shows a great performance with accuracy equals 98.38%.

Shahri et al.[22] proposed a ML model as an IDS based on a combination between SVM and GA . In this work, the authors employed GA as a wrapper feature selection, while SVM as a ML classifier. The obtained results show that GA can reduce 75.6% (i.e., 10 features out of 41) of the collected features. Moreover, the authors classified the selected features into three categories based on its importance (i.e., first priority, second priority, and third priority). The author's simulated their proposed model over KDD'99 dataset and the obtained results show a high accuracy (97.3%) of detecting abnormal packets with false alarm rate equals to 0.017.

Yang et al. [24] highlighted the importance of wrapper feature selection algorithm for building a lightweight IDS. The authors employed a modified haphazard mutation hill climbing as a search method. Yinhui et al.[25] applied a wrapper FS called gradually feature removal (GFR). The authors applied their FS method over KDD Cup dataset. The obtained results of GFR select 19 features out of 41. Moreover, the authors applied SVM as a ML classifier and the performance was 98.62%.

Selvakumar et al. [26] applied a firefly algorithm as a filter and wrapper FS methods. Moreover, the authors applied their

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

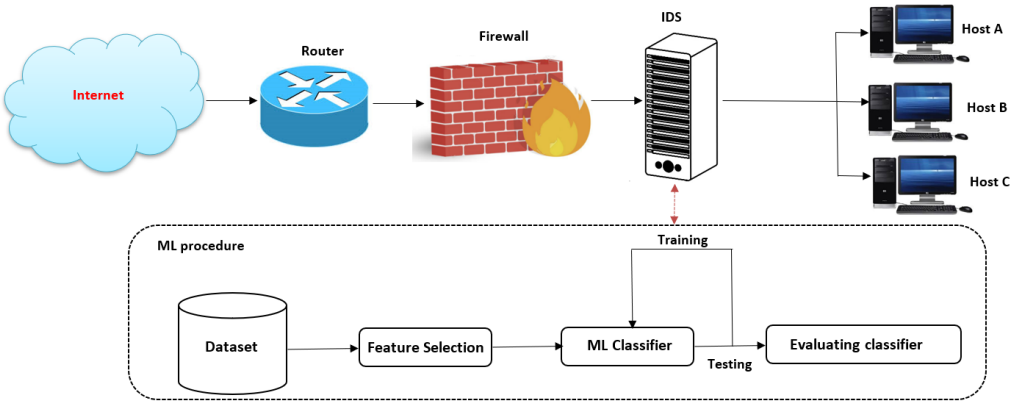

Fig. 2. IDS with ML procedure.

proposed method over the KDDCUP 99. The obtained results show that the performance of ML classifiers is improved with FS. Al-Yaseen [27] applied firefly algorithm as a wrapper FS method with SVM. The proposed system first removes the irrelevant features, which enhanced the performance of SVM. The obtained accuracy of the proposed IDS is 78.89%.

This paper tries to employ a wrapper feature selection based on BPSO and CNN as a classifier. It aims to employ the proposed method over UNSW-NB15 dataset.

III. Proposed Method

A. Binary PSO

Particle Swarm Optimization (PSO) algorithm proposed in 1995 by Kennedy and Eberhart [28]. PSO tries to simulate the social behavior of organisms (e.g., fish schooling). In simple, PSO consists of a set of solutions (particles) that are exploring the search space. Each solution has a position, which is adjusted based on two factors: internal memory for each particle and the location of the best particle in the search space. Each solution has two variables: position x_{id} and velocity v_{id} based on the position of the best obtained solution p_i and the position of best solution in the neighborhood p_g. Where i refers to the solution in the population $(i = 1, \ldots, Sn)$, n is the size of population, d refers to the dimension index of a solution $(d = 1, \ldots, m)$, and t refers to the number of iterations. The position and velocity for each solution is updated based on Eq. (1) and Eq. (2), respectively. Where the variable w refers to a positive inertia weight, r_1 and r_2 are two randomly generated numbers between [0,1] at each iteration, and c_1 and c_2 are the degree of influence of p_{id} and p_{gd} on the particles velocity, respectively. To control the velocity of each solution to be in the search space, the velocity variable v should be within range of $[v_{min}, v_{max}]$.

$$v_{id}(t+1) = wv_{id}(t) + c_1 r_1[p_{id}(t) - x_{id}(t)] + c_2 r_2[p_{gd}(t) - x_{id}(t)]. \quad (1)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1). \quad (2)$$

Given:
-Sn: swarm size.
-t: number of iterations.
-v: initial velocity.
-x: initial position.
-c1: degree of influence of p_{id}.
-c2: degree of influence of p_{gd}.
initialize particle( )
While $(current\_iteration \leq t)$
Evaluate each particle's position according to the fitness function.
Find the best solution of each particle so far.
update the global best solution.
update the velocity of each particle based on Eq. 1.
update the position of each particle based on Eq. 2.
end While
Output the global best solution

Fig. 3. The pseudo-code for Particle Swarm Optimization.

The particles in a continuous version of PSO move or update their positions based on Eqs. (1) and (2). While in BPSO, the variables values have either 0 or 1. Many research papers adopted BPSO to handle any binary optimization problems [29]. To convert continuous PSO to BPSO, a sigmoid transfer function (TF) is needed [29]. The continuous value is used as an input to TF in order to generate a probability value that switch each value to 0 or 1 in the position vector based on on Eq. (3):

$$S(v_{id}(t+1)) = \frac{1}{1 + e^{-v_{id}(t)}} \quad (3)$$

where V_i^d refers to the velocity value of the d^{th} dimension in the i^{th} vector, and t refers to the current iteration. Eq. (4) presents the updating procedure for the current particle based on probability value $S(v_{id}(t+1))$ that is obtained from Eq. (3):

$$x_{id}(t+1) = \begin{cases} 1 & \text{if } rand(0.0, 1.0) < S(v_{id}(t+1)) \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

where $x_i^d(t+1)$ refers to the element in the d^{th} dimension in the i^{th} position in the next iteration, $rand()$ refers to a

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

function that generates a random number between [0,1]. Table I explores the internal parameters setting used in BPSO. Figure 4 presents the overall performance of the BPSO for a single iteration. In this paper, we employed kNN as an internal classifier for selected features due to its simplicity and less execution time.

TABLE I
THE PARAMETERS SETTING FOR BPSO.

Parameters	Values
Number of iterations (t)	1500
Swarm size (Sn)	60
v_{max}	1
v_{min}	0
degree of influence (c1)	1.75
degree of influence (c2)	1.75
Inertia weight (w)	0.9

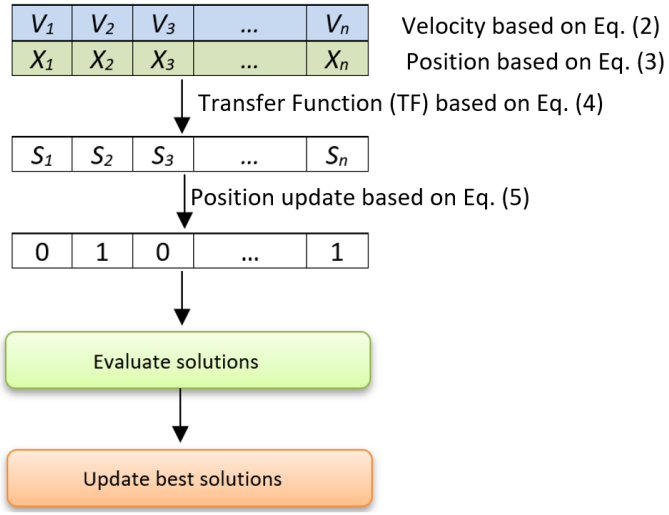

Fig. 4. An example of a BPSO for a single iteration.

B. CNN

One of the most well-known feedforward neural networks is Convolutional neural networks (CNNs), which mainly used in image processing [30]. CNN has an outstanding performance due to employing the convolutions concept. In simple, convolutions work by filtering the input data (e.g., image, row data, etc.) to small areas. CNN mainly used in image processing as two-dimensional (2D). However, 1D CNN exist can be employed successfully for row of data (e.g., time series processing). In this paper, we employed 1D CNNs to detect intrusions. Figure 5 explores the basic structure of 1D-CNN. The structure of 1D-CNN is quite similar to 2D CNN (i.e., convolutionReLU-MaxPooling). In 1D, the convolutions are 1D in order to handle each feature in a separate manner. For classification purpose, a fully connected layer is used to predict the output. Dropout layer is used to prevent overfitting [31].

IV. DATASET

In this paper, we used a public intrusion dataset called UNSW-NB1 that is collected by Moustafa et al.[32]. The dataset is a hybrid one that has actual current normal network operation and synthetic modified attack. The dataset has been collected using an attack generation tool called IXIA PerfectStorm. The dataset has nine types of modified attacks and real ones. The original dataset has 49 features. However, in this work we used only 43 input features and single output feature.

V. PERFORMANCE MEASURE

To evaluate the performance of proposed method, we employed four metrics which are: true positive (TP), true negative (TN), false positive (FP) and false negative (FN). Moreover, the confusion matrix as shown in Figure 6 is used to calculate true positive rate (TPR), true negative rate (TNR), false positive rate (FPR) and false negative rate (FNR). Based on these metrics, five factors are calculated which are: accuracy (See Eq.(5), Specificity (See Eq.(6), Precision (See Eq.(7), Recall (See Eq.(8), and F-Measure (See Eq.(9)).

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{5}$$

$$Specificity = \frac{TN}{TN + FP} \tag{6}$$

$$Precision = \frac{TP}{TP + FP} \tag{7}$$

$$Recall = \frac{TP}{TP + FN} \tag{8}$$

$$F - Measure = \frac{2 \times (Recall \times Precision)}{Recall + Precision} \tag{9}$$

VI. RESULTS AND ANALYSIS

In this paper, all experiments were performed on 3.80 GHZ, i7 CPU, 16.0 GB RAM, and Windows 10 operating system. All codes are implemented using MATLAB 2019a environment.

Table II explores the selected features for 11 independent runs. From the obtained results, we can see that BPSO can reduce the dimensionality of the data between 48% up to 82%. We believe that data reduction will enhance the overall performance of ML by eliminating the irrelevant features from the original dataset. While Table III evaluates the selected features in Table II based on Accuracy, Specificity, Precision, Recall, and F-Measure. It is clear that the performance of Run_{10} outperforms other experiments in terms of accuracy (i.e., 92%), and precision (i.e., 86%). Moreover, the BPSO reduces the data dimensionality 80% of the original data.

Table IV explores the performance of four ML classifiers (i.e., 1D-CNN, kNN, SVM, and DT) using the 10 selected features from BPSO. It is clear that the performance of CNN outperforms other classifiers based on the accuracy (i.e., 0.943) and precision (i.e., 0.930). While the worst performance comes from SVM with accuracy equals 0.850.

The obtained results show that the combination between 1D-CNN and BPSO can improve the performance of IDS in

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

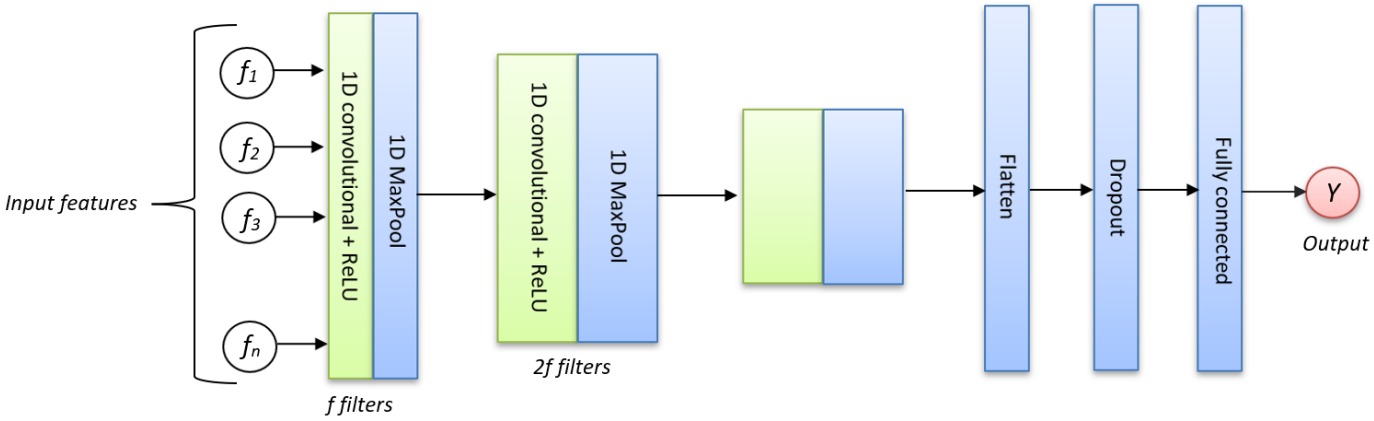

Fig. 5. Standard 1D-CNN.

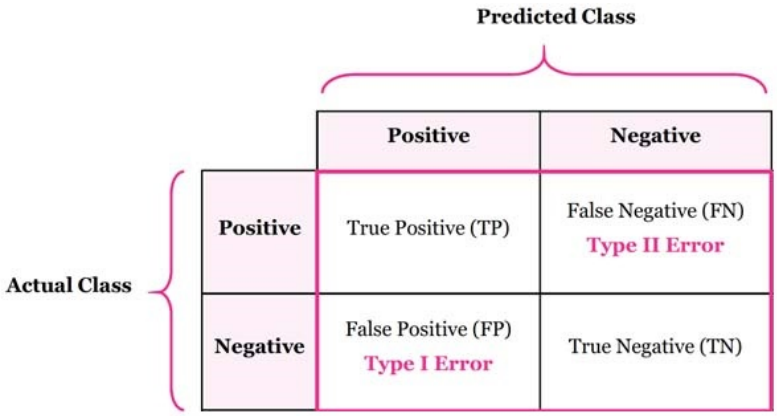

Fig. 6. Confusion matrix.

detecting abnormal packets in a simple, accurate, and robust manner.

VII. Conclusions and Future Work

In this paper, we employed a hybrid method between 1D-CNN and BPSO to detect the abnormal packets. In simple, BPSO is used as a wrapper features selection method, while 1-CNN is used as ML learning classifier. The proposed method is evaluated over a public dataset called UNSW-NB15. The obtained results show that the performance of the proposed method outperforms other ML classifiers (SVM, kNN, and DT) with an accuracy equals 94.3%.

In future works, we will examine the performance of different wrappers feature selection algorithms such as Binary Genetic Algorithm (BGA), Binary Gray Wolf Optimizer (BGWO), and Binary Ant Colony Optimization (BACO). Examining these methods will give us a better understanding about the most valuable features that affect on the performance of ML classifiers.

References

[1] S. Dey, Q. Ye, and S. Sampalli, "A machine learning based intrusion detection scheme for data fusion in mobile clouds involving heterogeneous client networks," *Information Fusion*, vol. 49, pp. 205 – 215, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1566253518306110

[2] R. A. Bridges, T. R. Glass-Vanderlan, M. D. Iannacone, M. S. Vincent, and Q. Chen, "A survey of intrusion detection systems leveraging host data," *ACM Computing Surveys (CSUR)*, vol. 52, no. 6, pp. 1–35, 2019.

[3] P. Ghosh, A. Karmakar, J. Sharma, and S. Phadikar, "Cs-pso based intrusion detection system in cloud environment," in *Emerging Technologies in Data Mining and Information Security*, A. Abraham, P. Dutta, J. K. Mandal, A. Bhattacharya, and S. Dutta, Eds. Singapore: Springer Singapore, 2019, pp. 261–269.

[4] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, "Deep learning approach for intelligent intrusion detection system," *IEEE Access*, vol. 7, pp. 41 525–41 550, 2019.

[5] B. Subba and P. Gupta, "A tfidfvectorizer and singular value decomposition based host intrusion detection system framework for detecting anomalous system processes," *Computers Security*, vol. 100, p. 102084, 2021. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167404820303576

[6] V. Jyothsna and V. Rama Prasad, "Fcaais: Anomaly based network intrusion detection through feature correlation analysis and association impact scale," *ICT Express*, vol. 2, no. 3, pp. 103 – 116, 2016, special Issue on ICT Convergence in the Internet of Things (IoT). [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2405959516300194

[7] S. Gamage and J. Samarabandu, "Deep learning methods in network intrusion detection: A survey and an objective comparison," *Journal of Network and Computer Applications*, vol. 169, p. 102767, 2020.

[8] U. Ravale, N. Marathe, and P. Padiya, "Feature selection based hybrid anomaly intrusion detection system using k means and rbf

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

TABLE II
SELECTED FEATURES IN 11 INDEPENDENT RUNS.

	Selected features																			
Run_1	2	6	8	10	14	17	25	32	34	35	36	38	39	40	42					
Run_2	2	3	4	9	10	13	24	25	29	30	31	34	35	40	42	43				
Run_3	6	8	26	32	33	36	39	40	41											
Run_4	2	4	5	7	9	15	20	21	22	26	31	33	40	41						
Run_5	4	6	7	9	10	11	14	15	17	19	21	25	27	33	35	36	40	41	42	43
Run_6	3	9	10	19	23	28	32	33	42	43										
Run_7	6	7	12	13	14	15	18	19	24	29	31	36	41	42						
Run_8	2	3	4	6	9	10	14	16	21	24	26	30	32	34	35	40	43			
Run_9	4	5	6	8	13	14	15	17	21	22	23	26	27	30	32	33	34	38	43	
Run_{10}	1	18	20	21	23	30	33	38	42	43										
Run_{11}	3	5	8	9	10	13	14	15	16	17	24	27	29	30	34	36	38	39	40	

TABLE III
OBTAINED RESULTS FOR BPSO USING KNN CLASSIFIER.

	Accuracy	**Specificity**	**Precision**	**Recall**	**F-Measure**
Run_1	0.87	0.81	0.80	0.86	0.81
Run_2	0.88	0.81	0.83	0.86	0.87
Run_3	0.88	0.81	0.83	0.83	0.88
Run_4	0.89	0.82	0.82	0.82	0.84
Run_5	0.89	0.84	0.82	0.86	0.87
Run_6	0.90	0.84	0.87	0.88	0.85
Run_7	0.90	0.85	0.83	0.83	0.81
Run_8	0.90	0.85	0.87	0.85	0.82
Run_9	0.91	0.86	0.83	0.84	0.83
Run_{10}	**0.92**	0.87	**0.86**	0.87	0.86
Run_{11}	0.92	0.90	0.83	0.86	0.87
Average	0.90	0.84	0.84	0.85	0.85
Best	0.92	0.90	0.87	0.88	0.88
worst	0.87	0.81	0.80	0.82	0.81
std.	0.02	0.03	0.02	0.02	0.02

TABLE IV
COMPARISON BETWEEN DIFFERENT ML CLASSIFIERS.

	1D-CNN		**kNN**		**SVM**		**DT**	
	Average	std.	Average	std.	Average	std.	Average	std.
Accuracy	**0.943**	0.0030	0.870	0.052	0.850	0.038	0.890	0.080
Specificity	0.920	0.0190	0.840	0.063	0.830	0.066	0.900	0.062
Precision	**0.930**	0.0320	0.840	0.020	0.810	0.031	0.820	0.079
Recall	0.910	0.0060	0.850	0.089	0.800	0.089	0.880	0.068
F-Measure	0.900	0.0030	0.850	0.027	0.816	0.033	0.843	0.032

kernel function," *Procedia Computer Science*, vol. 45, pp. 428 – 435, 2015, international Conference on Advanced Computing Technologies and Applications (ICACTA). [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050915004172

[9] W. Elmasry, A. Akbulut, and A. H. Zaim, "Evolving deep learning architectures for network intrusion detection using a double pso metaheuristic," *Computer Networks*, vol. 168, p. 107042, 2020.

[10] R. Patel, A. Thakkar, and A. Ganatra, "A survey and comparative analysis of data mining techniques for network intrusion detection systems," *International Journal of Soft Computing and Engineering (IJSCE)*, vol. 2, no. 1, pp. 265–260, 2012.

[11] D. E. Denning, "An intrusion-detection model," *IEEE Transactions on software engineering*, no. 2, pp. 222–232, 1987.

[12] R. Sahani, C. Rout, J. C. Badajena, A. K. Jena, H. Das *et al.*, "Classification of intrusion detection using data mining techniques," in *Progress in computing, analytics and networking*. Springer, 2018, pp. 753–764.

[13] S. Moustakidis and P. Karlsson, "A novel feature extraction methodology using siamese convolutional neural networks for intrusion detection," *Cybersecurity*, vol. 3, no. 1, pp. 1–13, 2020.

[14] S. Sarvari, N. F. M. Sani, Z. M. Hanapi, and M. T. Abdullah, "An efficient anomaly intrusion detection method with feature selection and evolutionary neural network," *IEEE Access*, vol. 8, pp. 70 651–70 663, 2020.

[15] A. Thakkar and R. Lohiya, "Attack classification using feature selection techniques: a comparative study," *Journal of Ambient Intelligence and Humanized Computing*, pp. 1–18, 2020.

[16] O. Almomani, "A feature selection model for network intrusion detection system based on pso, gwo, ffa and ga algorithms," *Symmetry*, vol. 12, no. 6, p. 1046, 2020.

[17] H. Turabieh, M. Mafarja, and X. Li, "Iterated feature selection algorithms with layered recurrent neural network for software fault prediction," *Expert systems with applications*, vol. 122, pp. 27–42, 2019.

[18] Y. Guo, F.-L. Chung, G. Li, and L. Zhang, "Multi-label bioinformatics data classification with ensemble embedded feature selection," *IEEE access*, vol. 7, pp. 103 863–103 875, 2019.

[19] K. Kim and S. Y. Zzang, "Trigonometric comparison measure: A feature selection method for text categorization," *Data & Knowledge Engineering*, vol. 119, pp. 1–21, 2019.

[20] A. Aneetha, T. Indhu, and S. Bose, "Hybrid network intrusion detection system using expert rule based approach," in *Proceedings of the Second International Conference on Computational Science, Engineering and Information Technology*, 2012, pp. 47–51.

[21] R. K. Cunningham, R. P. Lippmann, D. J. Fried, S. L. Garfinkel, I. Graf, K. R. Kendall, S. E. Webster, D. Wyschogrod, and M. A. Zissman, "Evaluating intrusion detection systems without attacking your friends: The 1998 darpa intrusion detection evaluation," MASSACHUSETTS INST OF TECH LEXINGTON LINCOLN LAB, Tech. Rep., 1999.

https://sites.google.com/site/ijcsis/
ISSN 1947-5500

[22] B. Aslahi-Shahri, R. Rahmani, M. Chizari, A. Maralani, M. Eslami, M. J. Golkar, and A. Ebrahimi, "A hybrid method consisting of ga and svm for intrusion detection system," *Neural computing and applications*, vol. 27, no. 6, pp. 1669–1676, 2016.

[23] S. S. Sivatha Sindhu, S. Geetha, and A. Kannan, "Decision tree based light weight intrusion detection using a wrapper approach," *Expert Systems with Applications*, vol. 39, no. 1, pp. 129 – 141, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0957417411009080

[24] Y. Li, J.-L. Wang, Z.-H. Tian, T.-B. Lu, and C. Young, "Building lightweight intrusion detection system using wrapper-based feature selection mechanisms," *Computers & Security*, vol. 28, no. 6, pp. 466–475, 2009.

[25] Y. Li, J. Xia, S. Zhang, J. Yan, X. Ai, and K. Dai, "An efficient intrusion detection system based on support vector machines and gradually feature removal method," *Expert Systems with Applications*, vol. 39, no. 1, pp. 424–430, 2012.

[26] B. Selvakumar and K. Muneeswaran, "Firefly algorithm based feature selection for network intrusion detection," *Computers & Security*, vol. 81, pp. 148–155, 2019.

[27] W. L. Al-Yaseen, "Improving intrusion detection system by developing feature selection model based on firefly algorithm and support vector machine," *IAENG International Journal of Computer Science*, vol. 46, no. 4, 2019.

[28] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Neural Networks, 1995. Proceedings., IEEE International Conference on*, vol. 4, Nov 1995, pp. 1942–1948 vol.4.

[29] H. Turabieh, M. Mafarja, and X. Li, "Iterated feature selection algorithms with layered recurrent neural network for software fault prediction," *Expert Systems with Applications*, vol. 122, pp. 27 – 42, 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0957417418308030

[30] A. Khan, A. Sohail, U. Zahoora, and A. S. Qureshi, "A survey of the recent architectures of deep convolutional neural networks," *Artificial Intelligence Review*, vol. 53, no. 8, pp. 5455–5516, 2020.

[31] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[32] N. Moustafa and J. Slay, "Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)," in *2015 Military Communications and Information Systems Conference (MilCIS)*, 2015, pp. 1–6.

https://sites.google.com/site/ijcsis/
ISSN 1947-5500