Avrupa Bilim ve Teknoloji Dergisi
Sayı 25, S. 463-474, Ağustos 2021
© Telif hakkı EJOSAT'a aittir
Araştırma Makalesi



www.ejosat.com ISSN:2148-2683

European Journal of Science and Technology
No. 25, pp. 463-474, August 2021
Copyright © 2021 EJOSAT
Research Article

Network Intrusion Detection using Optimized Machine Learning Algorithms

Tahira Khorram[1*], Nurdan Akhan Baykan[2]

[1*] Astra Global Corporation, 6301 zug, Switzerland, (ORCID: 0000-0001-8736-5085), tahira-k@astra-bank-bvi.com
[2] Konya Technical University, Department of Computer Engineering, 42130, Konya, Türkiye, (ORCID: 0000-0002-4289-8889), nbaykan@ktun.edu.tr

(First received 29 December 2020 and in final form 26 June 2021)

(**DOI:** 10.31590/ejosat.849723)

ATIF/REFERENCE: Khorram, T. & Akhan Baykan, N. (2021). Network Intrusion Detection using Optimized Machine Learning Algorithms. *European Journal of Science and Technology*, (25), 463-474.

Abstract

Network intrusion detection mechanism is a primary requirement in the current fast-growing network systems. Data mining and machine learning approaches are widely used for network anomaly detection during past few years. Machine learning based intrusive activity detector is becoming more popular. The most commonly used machine learning algorithms for Intrusion Detection System (IDS) are K-Nearest Neighbor (KNN), Support Vector Machine (SVM) and Random Forest (RF). However, the performance of these methods is reliant upon the selection of the proper parameter values. This research focuses its aim to build an IDS model based on the most effective algorithms. The machine learning algorithms are used in this research are KNN, SVM and RF. To improve these algorithms classification accuracy, some parameters of the algorithms are optimized using Particle Swarm Optimization (PSO) and Artificial Bee Colony (ABC) optimization techniques, while other parameters are used with default values. The result of this experiment shows that optimized KNN, SVM and RF perform better than these algorithms with their default parameter values. Furthermore, the results the experiment shows that KNN is the most suitable algorithm for network anomaly detection regarding detection of known network attacks and unknown network attacks. NSL-KDD standard dataset is used for the experiments of this research. It has been proven that our proposed model performs better than what is provided in the state-of-arts models.

Keywords: Anomaly Detection, Intelligent Intrusion Detection System, Swarm Intelligence, Machine Learning Algorithms**.**

Optimize Edilmiş Makine Öğrenimi Algoritmaları Kullanarak Internet Ağı Saldırı Tespiti

Öz

Internet ağı saldırı tespit mekanizması, mevcutta hızlı büyüyen ağ sistemlerinde birincil gereksinimdir. Veri madenciliği ve makine öğrenimi yaklaşımları, son birkaç yıldır ağ anomali tespiti için yaygın olarak kullanılmaktadır. Makine öğrenimi tabanlı saldırı tespit sistemleri son zamanlarda daha popüler hale gelmektedir. Saldırı Tespit Sistemi (STS) için en yaygın olarak kullanılan makine öğrenimi algoritmaları K-En Yakın Komşu (KNN), Destek Vektör Makinesi (DVM) ve Rastgele Orman (RO) algoritmalarıdır. Ancak bu yöntemlerin performansı, uygun parametre değerlerinin seçimine bağlıdır. Bu araştırma, etkili makine öğrenme algoritmalarına dayalı bir STS modeli oluşturma amacına odaklanmaktadır. Bu araştırmada kullanılan makine öğrenme algoritmaları KNN, DVM ve RO'dır. Bu algoritmaların sınıflandırma doğruluğunu iyileştirmek için algoritmaların bazı parametreleri Parçacık Sürü Optimizasyonu (PSO) ve Yapay Arı Kolonisi (YAK) optimizasyon teknikleri kullanılarak optimize edilmiştir. Çalışmanın sonucu, parametreleri optimize edilmiş KNN, DVM ve RO'nın, orijinal parametre değerleri ile kullanımlarından daha iyi performans gösterdiğini göstermektedir. Ayrıca, deney sonuçları, hem bilinen ağ saldırılarının hem de bilinmeyen ağ saldırılarının tespiti ile ilgili olarak ağ anomali tespitinde KNN'nin en uygun algoritma olduğunu göstermektedir. Bu araştırma kapsamında çalışmalarda NSL-KDD standart veri seti kullanılmıştır. Çalışmada önerilen modelin, son teknoloji modellerde sağlanandan daha iyi performans gösterdiği kanıtlanmıştır.

Anahtar Kelimeler: Anomali Tespiti, Akıllı Saldırı Tespit Sistemi, Sürü Zekâsı, Makine Öğrenimi Algoritmaları.

[*] Corresponding Author: tahira-k@astra-bank-bvi.com

1. Introduction

During the past two decades, global networking and internet have become a requirement for the majority of the population in the world. The usage of social networking sites and several applications are part of daily routine. The percentage of internet users has rapidly increased over few years and it is expected to rise more in the future. The increase of devices connected to the internet has increased the risk of the unauthorized activity, and it is becoming defenseless from attacks whether they are internal or external. Cybercriminals can attempt to dodge the security of a computer system to reach the confidential data. Because of this, users need to manage the security of their information and data.

To prevent the sensitive information from cybercriminals, there are several types of security services such as firewalls, Intrusion Detection System (IDS), Intrusion Prevention System (IPS). In this study, our focus is on IDS. IDS monitors and analyzes all events occurring on a computer network, identifies intrusions and searches for a sign of security problems [1, 2]. In case of anomaly, IDS generate an alarm to aware the system administrators. IDS implementation can be either network-based to monitor all the events happening in the network or can be deployed host-based to record all the incidents occurring in that specific PC [3].

Enterprises deploy IDS as network-based and use two approaches of the IDS for their business namely misuse-based IDS and anomaly-based IDS. Misuse-based IDS functions on signature and generate an alarm when an activity match the signature. Anomaly-based IDS sends alert to the network management when an action deviates the normal behavior of the network system [3, 4]. Both of these IDS types are used to protect a network system. Computer system tends to be secure if confidentiality, integrity and availability of that setup are assured [5].

An enormous amount of network traffic is generated every day. Machine Learning (ML) and Data Mining (DM) are the best methods processing this traffic. ML-DM methods can identify patterns of regular and intrusive traffic therefore they are used to identify network traffic. A classification method can learn these patterns and detect present known attacks and future unknown traffic.

In this study, three machine learning algorithms are used to categorize the malicious traffic and the normal traffic. These algorithms are K-Nearest Neighbor (KNN), Support Vector Machine (SVM), and Random Forest (RF) to design an efficient IDS. Machine learning algorithms performance strongly depending on optimization of their parameters. To improve the performance of our anomaly detector, specific parameters of KNN, SVM and RF are optimized. For optimization task, Particle Swarm Optimization (PSO) and Artificial Bee Colony (ABC) are used. After PSO and ABC tune the algorithm parameters, the optimized algorithms are trained and tested on NSLKDD dataset and compared with the not optimized algorithms.

The rest of this paper has been organized as follows: section two presents some literature review based on intrusion detection. Section three provides brief information on methods and materials used in this research. Part four includes experiments and part five explains result analysis and discussion. In the last section, conclusion and information about the future works are given.

2. Related Works

A study that has been done in 2015, by Dhanabal et al. [6] applied SVM on normalized NSL-KDD dataset to detect intrusions in networks. The author uses CFS (Correlation based Feature Selection) method to select the effective features of the NSLKDD dataset. This technique reduces the time and resource utilization as well as increase the accuracy rate. This paper provides sufficient information about the NSL-KDD dataset. SVM and KNN are used to detect network anomalies. In [7] while the NSLKDD dataset is used for training and testing the detector model. The author did both binary and multi-class classification. The SVM accuracy on test dataset was 69%, and KNN accuracy was 92%. A survey paper was published in 2016 [8] provide information on different types of machine learning algorithms including SVM, to be used for anomaly detection. The complexity of ML/DM algorithms is addressed, discussion of challenges for using ML/DM for cybersecurity is presented, and some recommendations are provided. Farnaaz et al. [9] in 2016 used RF modeling for intrusion detection and RF modeling produces a better result than most of the classification methods in term of detecting anomalies. RF deals with multi-class classification and the performance of the model evaluated regarding accuracy (ACC), False Acceptance Rate (FAR), Detection Rate (DR), Matthews correlation coefficient (MCC). In [9], the RF is compared with Decision Tree according to the Symmetric uncertainty of attributes. Another study written in 2016 by Aburomman et al. [10] proposed a novel ensemble construction method that uses PSO generated weights to create an ensemble of classifiers (SVM and KNN) with better accuracy for intrusion detection. The authors stated that weights made by metaheuristic could yield improved accuracy for intrusion detection system. SVM is a state-of-the-art machine learning algorithm. However, the performance of SVM depends on the selection of the appropriate parameters. In [11], the author proposed an IDS model based on Information Gain for feature selection combined with the SVM classifier. The parameters for SVM will be selected by swarm intelligence algorithms (PSO and ABC). NSLKDD dataset is used for the experiments. The new optimized model accuracy rate is 98.6% by PSO optimization and 98.8% by ABC optimization.

3. Material and Method

In this section, a brief information is provided on the algorithms and techniques that are used for network intrusion detection in this study.

3.1. Machine Learning Algorithms

3.1.1. K-Nearest Neigbor (KNN)

K-Nearest Neighbors algorithm is one of the supervised machine learning algorithms that is very simple to understand and is the mostly commonly used algorithms for classification. KNN works based on minimum distance from the query instance to the training sample to determine the nearest neighbors of the unknown instance. After the k nearest neighbors of the new instance is gathered, the majority vote of the nearest neighbors determines what class the unknown instance is classified to [12].

3.1.2. Support Vector Machine (SVM)

Support Vector Machine is a supervised machine learning algorithm which is used in classification and regression problems. It is widely used in security software such as network anomaly detection. In this algorithm, each data item is plotted as a point in an n-dimensional space, each point shows a feature of the dataset. The classification is performed by a hyperplane that differentiate the classes of the instances. This algorithm is simple to apply and provides an excellent result if the hyperplane is placed correctly or the right hyperplane is identified [13].

3.1.3. Random Forest (RF)

Random Forest establishes by a different bootstrap pattern from the initial data formed. A new instance that needs to be classified is put down each the trees in the forest for classification. After then each tree gives a vote that indicates the tree's decision about the class of the object and the forest chooses the type with the most votes for the project [1].

3.2. Metaheuristic Algorithms

3.2.1. Particle Swarm Optimization (PSO)

PSO is a swarm-based optimization technique introduced by Dr. Eberhard and Dr. Kennedy in 1995 [14]. This method is animated from the social behavior of birds searching for a piece of food in a specific area, the birds are not aware of where the food is located but, in each iteration, they know how close the food is. The best way to catch the food is to follow the nearest birds to the food [15,16]. In PSO term, the birds are called "particles". Each particle has position and velocity, velocity is the speed and direction of a bird. In addition of these attributes the particles also have fitness value which is obtained by calculating the fitness function at particle's current position [15, 17, 18].

The particle's velocity is updated using Equation 1.

$$v_{id}^{t+1} = w * v_{id}^{t} + c_1 * r_{1i} * (p_{id} - x_{id}^{t}) + c_2 * r_{2i} * (p_{gd} - x_{id}^{t}) \quad (1)$$

And the particle's postion is updated using Equation 2.

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

In the equations, t shows the t'th iteration of PSO, d indicates search space dimension, w is initial weight, c_1 and c_2 are acceleration factors, r_1 and r_2 are random numbers between [0,1]. And p_{id} and p_{gd} are pbest and gbest respectively. p_{id} (pbest) is the best particle in the t′th iteration and p_{gd} (gbest) is the best particle in all iterations until then [19].

3.2.2. Artificial Bee Colony (ABC)

ABC is inspired by the social behavior of honeybee searching for a food origin. The honeybees can store and share the information they have, can memorize the search area, and can take decision based on the provided information. Based in changed in the environment they can move further by social learning and teaching. This intelligence behavior of them motivates Karaboga to develop an algorithm called Artificial Bee Colony (ABC) in 2005 [5]. The honeybee behavior can be summarized as follows [20]:

1. Food Source: The food source is exploit by honey bees which has information about the quality, distance and direction of the food origin. While honeybee searches for the food, it selects a particular food origin. From this food source, bee gathers information about, the quality of food source, the amount of nectar and the direction in which the food located from the hive. Bee stores all this information to share with other bees later.
2. Employed bees: The Food source is discovered by this group of bees. They keep information about the quality of the food distance and direction from the hive.
3. Unemployed bees: This group of are divided into two categories for onlooker bees and scouts' bees. The onlooker bees receive information about food source and choose the food source with higher quality. Whereas scouts are responsible for finding the new food origin when the existing food source is exhausted [21].

The artificial bee colony behavior is the same as real honeybee. ABC process requires the following steps:

1. Population initialization: ABC generates a distributed population of SN solutions (food source positions), where *SN* represents the size of employed or onlooker bees, where each solution x_m is a D-dimensional vector. In our experiment D is the number of parameters to be optimized. Equations 3 shows the bee swarm initialization:

$$x_m = l_i + rand(0,1) * (u_i - l_i) \quad (3)$$

Where x_m is the food source, u_i and l_i are the upper level and lower level of solution space. "rand (0, 1)" is a random number in range [0, 1].

2. Employed bee phase: The employee bees search for food sources in the neighborhood. This exploration is defined in Equation 4:

$$v_{mi} = x_{mi} + \varphi_{mi}(x_{mi} - x_{ki}) \quad (4)$$

Where i is a randomly selected parameter index, x_k is a randomly selected food source, and φ_{mi} is a random number in the range [-1, 1]. After v_{mi} is generated, we can obtain the fitness value for the food origin according to Equation 5.

$$\mathrm{fit_i} = \begin{cases} \dfrac{1}{\mathrm{f_i} + 1}, & \mathrm{f_i} \geq 0 \\ 1 + |\mathrm{f_i}|, & \mathrm{f_i} < 0 \end{cases} \quad (5)$$

Where f_i shows the objective value of the *i*'th solution.

3. Onlooker bee phase: After employee bee has found the food source, they will share the information about the food source and its quality with the onlooker bees. The probability of selecting that food source by onlooker bees is represented in Equation 6.

$$p_i = \frac{fit_i}{\sum_{n=1}^{SN} fit_i} \quad (6)$$

Where fit_i indicates the fitness, solution represented by food source *i* and SN indicate the total number of food sources.

4. Scouts bee phase: If the effectiveness of food sources cannot be improved, then the scout bee removes the existing solution and start searching for a new solution randomly using equation.

3.3. Dataset

In this study NSL-KDD dataset is used to create an intrusion detection model. It is a predictive model that could determine whether the traffic is normal or attack [6]. This dataset has two parts as NSL-KDD train, and NSL-KDD test set. The training dataset is made up of 21 various attacks out the 37 attacks in the test set. All these attack types are categorized into four classes of DoS, Probe, R2L, and U2R. For this study, the original NSL-KDD dataset which contains 125973 data records is selected for training. While for the test approaches, two different test datasets are used. The first test dataset contains 25192 data records of known attacks used for known attacks predictions. The other dataset that includes 22544 data records is used for unknown attacks prediction. Dataset features are given in Table 1.

Table 1. Dataset Attributes

No	Feature	No	Feature
1	Duration	*22*	Is_guest_login
2	Protocol_type	*23*	Count
3	Service	*24*	Srv_count
4	Flag	*25*	Serror_rate
5	Src_bytes	*26*	Srv_serror_rate
6	Dst_bytes	*27*	Rerror_rate
7	Land	*28*	Srv_rerror_rate
8	Wrong_fragment	*29*	Same_srv_rate
9	Urgent	*30*	Diff_srv_rate
10	Hot	*31*	Srv_dif_host_rate
11	Num_failed_logins	*32*	Dst_host_count
12	Logged_in	*33*	Dst_host_srv_count
13	Num_compromised	*34*	Dst_host_same_srv_rate
14	Root_shell	*35*	Dst_host_diff_srv_rate
15	Su_attempted	*36*	Dst_host_same_src_port_rate
16	Num_root	*37*	Dst_host_srv_dif_host_rate
17	Num_file_creations	*38*	Dst_host_serror_rate
18	Num_shells	*39*	Dst_host_srv_serror_rate
19	Num_access_files	*40*	Dst_host_rerror_rate
20	Num_outband_cmds	*41*	Dst_host_srv_rerror_rate
21	İs_hot_login		

4. Experimental Results

Objective of this study is to design a network intrusion detection system using machine learning algorithms. Since machine learning algorithms parameters values have a high impact on the performance of the algorithm, aim of the study is to find the suitable values for the parameters of these algorithms using PSO and ABC algorithms. There are two experiments where in the first experiment a detective model is made using the algorithms with default parameter values, and in the second experiment PSO and ABC are used to optimize the most critical parameters for KNN, SVM and RF. The classification approach is a multiclass classification implemented in Python.

Both experiments use the same NSL-KDD dataset. Thus, dataset preparation is a global setting. The dataset is noisy, contains some unnecessary and redundant features. Dataset records do not have the same scales; some of them are scaled with timescales while the other attribute has a byte scale. All these problems need to be addressed before the algorithms use the dataset. The following steps are taken to preprocess the datasets.

1. The dataset features are given a name. It makes easy to address one specific element.
2. The attack types are grouped into four classes DoS, Probe, R2L, U2R.
3. The symbolic feature values are converted to numeric values. For example, features “protocol type” has three values of “tcp, udp, icmp” and we grant tcp=1, udp=2, icmp=3.
4. The dataset is normalized between [0,1]
5. The dataset features are reduced from 41 features to 26 features by PCA.

4.1. Feature Reduction using PCA

Principal Components Analysis (PCA) is dimensionality reduction tool that is used to reduce a broad set of data records to a small number of data which is more meaningful and usable. PCA changes redundant feature into orthogonal features [23]. This means it combines correlated features into one feature space. So, no two elements contain the same information about the data record. It can be helpful to reduce the original feature space to a lower number of features before feeding the data as a training data or test data to the Machine learning classifier. Indeed, it can reduce the computational cost of the system tremendously. We used Python Sklearn library to implement PCA feature elimination method.

4.2. Intrusion Detection using ML with Default Parameters

In the first experiment; KNN, SVM and RF are used with default parameters to build a classifier model based on NSL-KDD training dataset.

The model that is made on KNN supposed to classify the dataset into five classes of Normal, DoS, Probe, R2L and U2R based on the training examples that was given to the algorithm. Sklearn Python tool is used to implement this classification method. Once the model is made, it is tested for its predictability power on both test datasets, the known and unknown attack datasets. For classifier validation, 5-fold cross-validation is used. For performance evaluation; detection rate (DR), accuracy rate (AR), time utilization (Time) and resource utilization (Resource)

are used. Accuracy rate is calculated based on the correctly classified data per total amount of data in the dataset. The detection rate is the ratio between the correctly classified attacks and the total number of attacks in the dataset.

For implementing the SVM and RF classifier, the approach is the same just the classifier is changed. The classifiers that build based on the SVM and also the RF algorithms, their performances will be tested on the both test datasets. The DR, AR, Time and Resource are the metrics that evaluate the performance of these algorithms. Algorithms with their default parameters are used in the experiment are given in Table 2.

Table 2. ML Parameters Default Values

KNN Parameter	SVM Parameters	RF Parameters
K = 5	Cost = 1	N-estimators = 10
	Gamma = 1/number of features	Random-state = 0
		Min-sample-leaf = 1

4.3. Intrusion Detection using ML with Optimized Parameters

Classification is a supervised learning method in which the computer program learns from the data input given to it and then uses this learning to classify new observation. Classification accuracy is a metric to evaluate a classifier model. Accuracy represents the number of correct predictions from all predictions made. By looking at the accuracy of a classifier model, it is determined that how accurate a classifier works. The aim of this study is to improve the performance of the classifiers by optimizing their parameters. For this purpose, some important parameters of the KNN, SVM, and Rf algorithms are optimized by PSO and ABC algorithms. Table 3 shows the parameters to be optimized and their optimization ranges.

For the KNN algorithm, choosing the right value for K improves the performance of the algorithm. The ABC and PSO algorithms choose the best value for K, an optimum value for K will improve the performance of the IDS based on KNN algorithm.

For SVM algorithm, the Cost and Gamma parameters are optimized. The parameter Gamma determines the influence of training examples in the classifier model that will be created. A low value shows that each training example does not have a high effect on the classifier and a higher value indicates that every training example has impact on the classifier model. Moreover, the Cost parameter determines the cost of misclassification on the training examples. Cost with a higher value make a strict classification, in this case the margin of error will be smaller, and the classifier is supposed to classify every sample correctly. A lower cost value makes the margin error loose and will cause misclassification. The optimal cost value is a value that leaves some space for errors while the intention must a correct classification.

For the RF algorithm, three important parameters that help the algorithm learn faster and predict more accurate are optimized. The n_esimators parameter is the number of trees that are built before the prediction. A higher number of trees improve the accuracy but utilize more time and resources. Min_sample_leaf parameter is the end node of a decision tree, by increasing this parameter's value each tree in the forest become more constrained as it has to consider more samples at each node. The last parameter that is optimized for RF is random_state. It makes a solution to replicate more comfortable and help the RF algorithm to learn faster.

Table 3. ML Parameter's Optimization Ranges

KNN Parameter	SVM Parameters	RF Parameters
K = 3	Cost = $[2^{-1}, 2^{3}]$	N-estimators = [1, 100]
	Gamma = $[2^{-6}, 2]$	Random-state = (0,100]
		Min-sample-leaf = [1, 10]

4.3.1. Parameter Optimization using PSO

In this experiment first PSO was used for the KNN, RF and SVM parameters optimization. For optimizing the K parameter, the fitness function was calculated by measuring the accuracy rate of the classifier on test datasets. The PSO parameters were set as the number of swarms was selected as 10 and 20 particles, maximum cycle was 30, and C1 and C_2 were set as 0.7 and 1, respectively. For optimizing the K parameter of KNN algorithm, PSO is initiated by a random swarm of particles in the optimization range and search for the best value for the K parameter. The KNN algorithm shows the best performance when K is 3. For optimizing the Cost (C) and Gamma parameters of SVM algorithm, the PSO algorithm searches for the best values for C and Gamma in the given range above. After 30 iterations, the algorithm found the best values for C and Gamma displays in Table 4. RF parameters optimization is a three-dimensional search area. The particles at the same time must find three best values for the three parameters of RF to improve the algorithm's accuracy. In iteration 30, the PSO algorithm returns the optimum values for n_estomators, min_sample_leaf, and random_state parameters show in Table 4. Table 4 shows the optimum values found by the PSO algorithm so far. And, Figure 1 shows parameter optimization steps by PSO.

Table 4. Best parameters values found by PSO

PSO with 10 particles		
KNN Parameter	**SVM Parameters**	**RF Parameters**
K = 3	Cost = 6.5	N- estimators = 73
	Gamma = 1.80	Random-state = 42
		Min-sample-leaf = 1
PSO with 20 particles		
KNN Parameter	**SVM Parameters**	**RF Parameters**
K = 3	Cost = 6.5	N- estimators = 70
	Gamma = 1.85	Random-state = 38
		Min-sample-leaf = 1

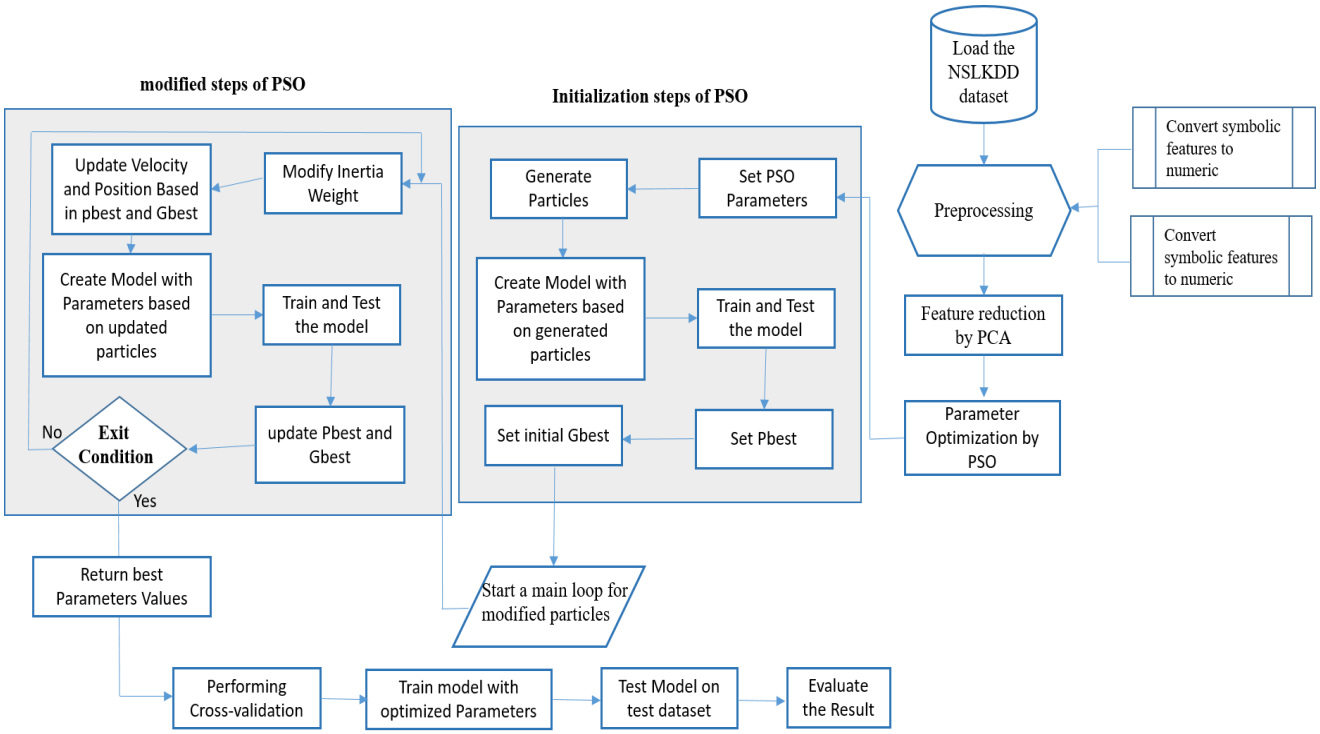


Fig.1. PSO Work Flow for Parameter Optimization

4.3.2. Parameter Optimization using ABC

Following PSO, ABC algorithm is used to optimize the parameter of KNN, SVM, and RF algorithms. The ranges of optimization are the same PSO. The ABC parameters are set as the colony size is selected as 10 and 20, maximum cycle is 30 and employed bee percentage is 50%. In each phase half of the bees forage the food sources and delivers the gathered information to the other bees which remains in the hive [24].

For K optimization, the ABC algorithm after 30 cycles stops the process and returns optimum values as 3 for the K parameter. For SVM parameter optimization, the algorithm tried to find the optimum values for C and Gamma parameters. Finally, for RF, ABC tried to find the optimum values for n-estimator, min-sample-leaf and random-state. The ABC algorithms follow the same steps for optimizing KNN, SVM and RF algorithms parameters. Figure 2 shows parameter optimization steps by ABC. Table 5 displays the values found by ABC for the algorithm's parameters.

Table 5. Best parameters values found by ABC

ABC with 10 bees		
KNN Parameter	**SVM Parameters**	**RF Parameters**
K = 3	Cost = 6.04	N- estimators = 34
	Gamma = 1.33	Random-state = 97
		Min-sample-leaf = 1
ABC with 20 bees		
KNN Parameter	**SVM Parameters**	**RF Parameters**
K = 3	Cost = 5.34	N- estimators = 33
	Gamma = 1.50	Random-state = 97
		Min-sample-leaf = 1

5. Results and Discussion

There are two different analyzes for the both known and unknown test dataset. In the first analysis; KNN, SVM, and RF algorithms are used with default parameters to build and anomaly detection model. The model is trained on NSL-KDD training dataset and teste on the NSL-KDD testing datasets, both known and unknown attack datasets. Tables 6 and 7 show the algorithm performances on datasets with default parameters. In Table 6, if we consider computational cost all the algorithms perform very well in term of memory usage and CPU usage, but RF consumes fewer resources and test time than SVM. In terms of accuracy rate and detection rate, KNN performs better than other algorithms. Table 7 shows the test results on unknown test dataset. Here KNN performs better than SVM and RF regarding classification accuracy, detection rate and train time. SVM regarding resource consumption does not perform very well but its accuracy rate is better than RF. Tables 8 and 9 show the results of optimized classifiers on known and unknown test datasets. Table 8, the success of the optimized algorithms is not very clear; but in the Table 9, the successes of classifiers are obvious.

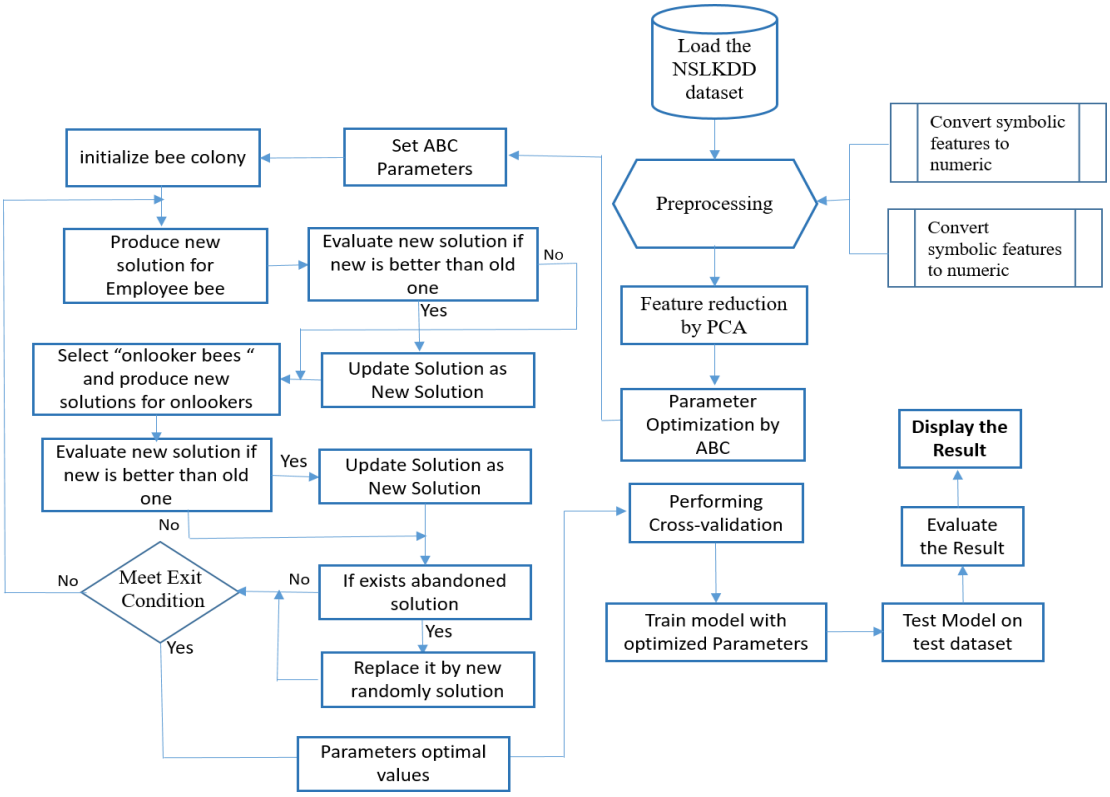


Fig.2. ABC Work Flow for Parameter Optimization

Table 6. Intrusion detection results using ML with default parameters (Known Attacks test set)

Classifiers	CV-Score	Train Time (Sec)	Test Time (Sec)	RAM Usage	CPU Usage	Overall Detection Rate	Overall Accuracy Rate
KNN	0.9978	0.44	2.25	0.25 GB	42.7	0.997	0.998
SVM	0.990	79.5	10.23	0.27 GB	26	0.992	0.990
RF	0.995	6.33	0.038	0.25 GB	31.10	0.994	0.996

Table 7. Intrusion detection results using ML with default parameters (Unknown Attacks test set)

Classifiers	CV-Score	Train Time (Sec)	Test Time (Sec)	RAM Usage	CPU Usage	Overall Detection Rate	Overall Accuracy Rate
KNN	0.997	0.36	3.34	0.25 GB	36.7	0.72	0.78
SVM	0.991	81.27	9.6	0.26 GB	26.4	0.66	0.76
RF	0.996	5.70	0.030	0.24 GB	30.20	0.66	0.75

Table 8. ML algorithms performance on known attacks dataset

Algorithms	CV-Score	Train Time (Sec)	Test Time (Sec)	RAM Usage	CPU Usage	Overall Detection Rate	Overall Accuracy Rate
KNN	0.9978	0.44	2.25	0.25 GB	42.7	0.997	0.998
KNN+PSO-10	0.998	0.83	0.46	0.21 GB	30.8	1.00	0.999
KNN+ABC10	0.998	0.67	0.57	0.20 GB	21.00	1.00	0.999
KNN+PSO-20	0.998	0.77	0.54	0.27 GB	21.1	1.00	0.999
KNN+ABC20	0.998	0.69	0.60	0.21 GB	28.1	1.00	0.999
SVM	0.990	79.5	10.23	0.27 GB	26	0.992	0.990
SVM+PSO-10	0.994	51.7	5.65	0.30 GB	29.2	0.995	0.994
SVM+ABC10	0.993	52.31	4.70	0.22 GB	22.00	0.995	0.994
SVM+PSO-20	0.993	54.4	5.65	0.30 GB	37.2	0.995	0.994
SVM+ABC20	0.993	45.8	4.33	0.29 GB	27.9	0.995	0.994
RF	0.995	6.33	0.038	0.25 GB	31.10	0.994	0.996
RF + PSO-10	0.998	49.5	0.24	0.21 GB	27.4	0.999	0.999
RF+ ABC-10	0.998	34.12	0.34	0.21 GB	29.8	0.999	0.999
RF + PSO-20	0.998	57.7	0.25	0.11 GB	37.4	0.999	0.999
RF + ABC-20	0.998	52.2	0.31	0.30 GB	42.8	0.999	0.999

Table 9. ML algorithms performance on unknown attacks dataset

Algorithms	CV-Score	Train Time (Sec)	Test Time (Sec)	RAM Usage	CPU Usage	Overall Detection Rate	Overall Accuracy Rate
KNN	0.997	0.36	3.34	0.25 GB	36.7	0.72	0.78
KNN+PSO-10	0.998	0.66	2.25	0.21 GB	25.4	0.749	0.795
KNN+ABC-10	0.998	0.68	1.42	0.20 GB	22.5	0.751	0.795
KNN+PSO-20	0.998	1.23	0.54	0.21 GB	51.1	0.749	0.795
KNN+ABC20	0.998	0.62	2.87	0.20 GB	25.4	0.751	0.795
SVM	0.991	81.27	9.6	0.26 GB	26.4	0.66	0.76
SVM+PSO-10	0.994	53.8	6.78	0.25 GB	25.2	0.683	0.77
SVM+ABC-10	0.994	65.86	5.71	0.23 GB	17.4	0.71	0.77
SVM+PSO-20	0.994	51.9	4.33	0.29 GB	27.9	0.686	0.765
SVM+ABC-20	0.994	110	16.2	0.30GB	13.2	0.684	0.765
RF	0.996	5.70	0.030	0.24 GB	30.20	0.66	0.75
RF + PSO-10	0.998	11.2	0.041	0.20 GB	53.2	0.696	0.76
RF + ABC-10	0.998	22.21	0.15	0.20 GB	24.20	0.709	0.779
RF + PSO-20	0.998	52.2	0.038	0.20 GB	39.6	0.705	0.773
RF + ABC-20	0.998	20.43	0.13	0.20 GB	25.9	0.709	0.778

Tables 10-11, show confusion matrices of classifiers with default parameters. Tables 12-19 show confusion matrices of classifiers with optimized parameters

Table 10. Confusion matrices for ML algorithms with default parameters on known test dataset

	Confusion Matrix for KNN						**Confusion Matrix for SVM**						**Confusion Matrix for RF**				
	Actual						Actual						Actual				
	Normal	Probe	DoS	R2L	U2R		Normal	Probe	DoS	R2L	U2R		Normal	Probe	DoS	R2L	U2R
Predicted	13434	7	8	6	1		13336	50	20	16	3		13428	14	16	14	3
	6	2275	1	0	1		70	2227	8	1	0		15	2273	5	0	1
	5	4	9225	3	0		27	10	9205	6	1		2	2	9213	2	0
	4	3	0	199	3		16	2	1	186	3		4	0	0	192	3
	0	0	0	1	6		0	0	0	0	4		0	0	0	1	4

Table 11. Confusion matrices for ML algorithms with default parameters on unknown test dataset

	Confusion Matrix for KNN						**Confusion Matrix for SVM**						**Confusion Matrix for RF**				
	Actual						Actual						Actual				
	Normal	Probe	DoS	R2L	U2R		Normal	Probe	DoS	R2L	U2R		Normal	Probe	DoS	R2L	U2R
Predicted	9446	318	1451	1720	28		9419	749	1149	2267	97		9432	782	1621	2027	101
	204	1783	145	301	133		209	1352	123	111	91		221	1453	94	243	65
	52	258	5862	221	2		78	320	6186	178	5		58	186	5743	260	7
	8	62	0	490	16		5	0	0	198	5		0	0	0	224	27
	1	0	0	22	21		0	0	0	0	2		0	0	0	0	0

Table 12. Confusion matrices for ML algorithms optimized by PSO-10 on known test dataset

	Confusion Matrix for KNN+PSO-10						**Confusion Matrix for SVM+PSO-10**						**Confusion Matrix for RF+PSO-10**				
	Actual						Actual						Actual				
	Normal	Probe	DoS	R2L	U2R		Normal	Probe	DoS	R2L	U2R		Normal	Probe	DoS	R2L	U2R
Predicted	13448	0	0	0	0		13387	32	9	15	2		13447	0	0	1	1
	0	2289	0	0	0		37	2255	1	1	1		1	2289	0	0	0
	0	0	9234	0	0		19	2	9223	3	0		0	0	9234	0	0
	0	0	0	209	0		6	0	1	189	1		0	0	0	208	0
	1	0	0	0	11		0	0	0	1	7		1	0	0	0	10

Table 13: Confusion matrices for ML algorithms optimized by PSO-10 on unknown test dataset

	Confusion Matrix for KNN+PSO-10						**Confusion Matrix for SVM+PSO-10**						**Confusion Matrix for RF+PSO-10**				
	Actual						Actual						Actual				
	Normal	Probe	DoS	R2L	U2R		Normal	Probe	DoS	R2L	U2R		Normal	Probe	DoS	R2L	U2R
Predicted	9465	367	1077	1730	42		9421	62	830	1344	23		9445	674	1297	1854	66
	183	1747	103	338	102		205	1960	126	716	140		208	1547	142	443	95
	58	256	6263	221	2		79	399	6408	189	8		56	200	6017	261	2
	5	50	0	417	13		6	0	94	498	4		2	0	2	104	12
	0	1	15	48	41		0	0	0	7	25		0	0	0	92	25

Table 14. Confusion matrices for ML algorithms optimized by PSO-20 on known test dataset

	Confusion Matrix for KNN+PSO-20						**Confusion Matrix for SVM+PSO-20**						**Confusion Matrix for RF+PSO-20**				
	Actual						Actual						Actual				
	Normal	Probe	DoS	R2L	U2R		Normal	Probe	DoS	R2L	U2R		Normal	Probe	DoS	R2L	U2R
Predicted	13448	0	0	0	0		13387	32	9	15	2		13447	0	0	1	1
	0	2289	0	0	0		37	2255	1	1	1		1	2289	0	0	0
	0	0	9234	0	0		19	2	9223	3	0		0	0	9234	0	0
	0	0	0	209	0		6	0	1	189	1		0	0	0	208	0
	1	0	0	0	11		0	0	0	1	7		1	0	0	0	10

Table 15. Confusion matrices for ML algorithms optimized by PSO-20 on unknown test dataset

	Confusion Matrix for KNN+PSO-20						**Confusion Matrix for SVM+PSO-20**						**Confusion Matrix for RF+PSO-20**				
	Actual						Actual						Actual				
	Normal	Probe	DoS	R2L	U2R		Normal	Probe	DoS	R2L	U2R		Normal	Probe	DoS	R2L	U2R
Predicted	9464	367	1077	1730	42		9420	696	1015	2236	70		9460	657	1116	1959	44
	183	1747	103	338	102		205	1380	229	143	96		197	1556	194	297	118
	59	256	6263	221	2		79	345	6214	126	9		51	208	6146	214	2
	5	50	0	417	13		6	0	0	238	10		3	0	2	240	12
	0	1	15	48	41		1	0	0	11	15		0	0	0	44	24

Table 16. Confusion matrices for ML algorithms optimized by ABC-10 on known test dataset

	Confusion Matrix for KNN+ABC-10						**Confusion Matrix for SVM+ABC-10**						**Confusion Matrix for RF+ABC-10**				
	Actual						Actual						Actual				
	Normal	Probe	DoS	R2L	U2R		Normal	Probe	DoS	R2L	U2R		Normal	Probe	DoS	R2L	U2R
Predicted	13448	0	0	0	0		13388	32	7	16	1		13447	0	0	1	1
	0	2289	0	0	0		36	2254	1	0	1		0	2289	0	0	0
	0	0	9234	0	0		19	3	9225	4	0		0	0	9234	0	0
	0	0	0	209	0		6	0	1	188	2		1	0	0	208	0
	1	0	0	0	11		0	0	0	1	7		1	0	0	0	10

Table 17. Confusion matrices for ML algorithms optimized by ABC-10 on unknown test dataset

	Confusion Matrix for KNN+ABC-10						**Confusion Matrix for SVM+ABC-10**						**Confusion Matrix for RF+ABC-10**				
	Actual						Actual						Actual				
	Normal	Probe	DoS	R2L	U2R		Normal	Probe	DoS	R2L	U2R		Normal	Probe	DoS	R2L	U2R
Predicted	9465	367	1057	1727	40		9451	445	1229	1973	56		9455	481	1223	1963	56
	183	1747	119	321	107		207	1767	135	266	108		203	1744	137	261	108
	58	254	6265	177	2		51	209	6092	250	2		50	196	6096	255	2
	5	52	2	419	13		2	0	237	237	20		3	0	2	246	10
	0	1	15	110	38		0	0	28	28	24		0	0	0	29	24

Table 18. Confusion Matrices for ML algorithms optimized by ABC-20 on known test dataset

	Confusion Matrix for KNN+ABC-20						**Confusion Matrix for SVM+ABC-20**						**Confusion Matrix for RF+ABC-20**				
	Actual						Actual						Actual				
	Normal	Probe	DoS	R2L	U2R		Normal	Probe	DoS	R2L	U2R		Normal	Probe	DoS	R2L	U2R
Predicted	13448	0	0	0	0		13388	32	7	16	1		13447	0	0	1	1
	0	2289	0	0	0		36	2254	1	0	1		0	2289	0	0	0
	0	0	9234	0	0		19	3	9225	4	0		0	0	9234	0	0
	0	0	0	209	0		6	0	1	188	2		1	0	0	208	0
	1	0	0	0	11		0	0	0	1	7		1	0	0	0	10

Table 19. Confusion Matrices for ML algorithms optimized by ABC-20 on unknown test dataset

	Confusion Matrix for KNN+ABC-20						**Confusion Matrix for SVM+ABC-20**						**Confusion Matrix for RF+ABC-20**				
	Actual						Actual						Actual				
	Normal	Probe	DoS	R2L	U2R		Normal	Probe	DoS	R2L	U2R		Normal	Probe	DoS	R2L	U2R
Predicted	9464	367	1057	1727	40		9406	57	919	2178	24		9449	517	1260	1979	49
	183	1747	119	321	107		227	1942	132	179	144		208	1730	102	251	115
	58	254	6265	177	2		73	422	6407	73	8		51	174	6094	234	2
	5	52	2	419	13		5	0	0	268	8		3	0	2	258	10
	1	1	15	110	38		0	0	0	56	16		0	0	0	32	24

In terms of detection known network attacks both optimized versions of the algorithms perform very well, there is only a very little difference between ABC and PSO. Whether regarding detecting unknown network attacks, the case is different KNN-ABC detection rate is 75% while in KNN-PSO is 74.9%. The accuracy rate of KNN-PSO is 79.5%, and the accuracy rate of KNN-ABC is 79.55%. In case of SVM-PSO and SVM-ABC, the

detection rate of SVM-PSO is 68.6% while SVM-ABC is 71.1% but in term of accuracy rate SVM-PSO performs better than SVM-ABC. RF-ABC with an accuracy rate of 77.8% and detection rate of 70.9% performs better than RF-PSO. When experimental results are compared to the literature, proposed method performs better than what is achieved in the novels. For example, in [6], Voldan achieved 92.47% accuracy for known attacks by KNN while our KNN result on the known dataset is 99.8%. Furthermore, [6] classification performance for SVM was 69% while we achieved 99.4% for known data samples and 77% for unknown data samples. Table 20 shows a comparison between our best results and the literature for the known dataset. Note that accuracy results given in the Table 20 are evaluated on known attacks test dataset. For unknown attacks, Table 21 shows the best accuracy results of our method. Here, we could not compare our results with the literature because, we could not find any study that has been done on the full unknown test dataset.

Table 20. A comparison of our proposed method with the literatures (for known dataset)

With Default Parameters		
Method	**Dataset**	**Accuracy**
RF [9]	NSL-KDD	RF = 99.8%
KNN and SVM [7]	NSL-KDD	KNN = 92.47 %, SVM = 69 %
RF and SVM [24]	NSL-KDD	SVM = 99.1%, RF = 99.5
Proposed method [27]	NSL-KDD	KNN = 99.8%, SVM = 99%, RF = 99.6%

With Optimized Parameters		
Method	**Dataset**	**Accuracy**
SVM optimized by PSO [25]	KDD CUP'99	SVM (default parameters) = 82.6% SVM + PSO-30 = 99.8%
SVM Optimized by ABC [26]	KDD CUP'99	SVM+ABC-20 = 92.7%
SVM Optimized by PSO and ABC [11]	NSL-KDD	SVM+PSO-20 = 98.6%, SVM+ABC-20 = 98.8%
Proposed method [27]	NSL-KDD	KNN+PSO-10/20 = 99.9% KNN+ABC-10/20 = 99.9% SVM+PSO-10/20 = 99.4% SVM+ABC-10/20 = 99.4 RF+PSO-10/20 = 99.9% RF+ABC-10/20 = 99.9%

Table 21. Optimized ML algorithms best performance on unknown attacks datasets [27]

KNN			**SVM**			**RF**	
KNN+PSO-20	79.54%		SVM+PSO-10	77%		RF+PSO-20	77.3%
KNN+ABC-20	79.55%		SVM+ABC-10	77%		RF+ABC-10	77.9%

6. Conclusions and Recommendations

The primary goal of this research is to implement network intrusion detection using machine learning algorithms. For this aim KNN, SVM and RF were used for this study. The performances of the machine learning algorithms are strongly depending on selection appropriate values for their parameters. For parameter optimization, PSO and ABC were used to choose the suitable values for the algorithms parameters and improve the performance. To prove the concept of parameter optimization, improve the algorithm's performance, performance of optimized KNN, SVM and RF were compared with standard KNN, SVM and RF. The experimental results show that optimized algorithms perform better than algorithms with default parameter values. All the experimental results showed that KNN algorithm has a better classification performance while consuming fewer amount of resources and time than SVM. The resource consumption of the RF is less than all other algorithms, but its classification performance is not better than SVM and KNN. Since KNN resource consumption and RF resource consumption is almost the same, so this is not a factor of comparison. The results gained from the experiments indicate that KNN classification performance is 99.8%, KNN-PSO is 99.9 % and KNN-ABC is 99.8% on known attack dataset; and KNN performance is 78%, KNN-PSO is 79.5%, and KNN-ABC is 79.55% on unknown attack test dataset. SVM algorithm performance has been improved from 66% detection rate to 68.6% by PSO optimization and 71% via ABC optimization. For the future work we are considering focusing on detecting

minority attacks (R2L and U2R) to improve their detection and accuracy rate.

7. Acknowledgement

This study is based on Tahira Khorram's Master Thesis [27].

References

[1] Ganapathy, S., Kulothungan, K., Muthurajkumar, S., Vijayalakshmi, M., Yogesh, P., & Kannan, A. (2013). Intelligent feature selection and classification techniques for intrusion detection in networks. *A survey. EURASIP Journal on Wireless Communications and Networking*, 913-921.

[2] Mukherjee, S. and Sharma, N., (2012). Intrusion detection using naive Bayes classifier with feature reduction. *Procedia Technology*, 119-128.

[3] Med, A., Lisitsa, A., & Dixon, C., (2011). A misuse-based network intrusion detection system using temporal logic and stream processing. *IEEE Network and System Security (NSS), 5th International Conference on*, Milan.

[4] Butun, I., Morgera., S., D., & Sankar., R., (2013). A Survey of Intrusion Detection Systems in Wireless Sensor Networks, *IEEE Communications Surveys and Tutorials*, 266-182.

[5] Karaboga, D., (2005). An idea on honey bee swarm for numerical optimization. Kayseri: Erciyes University,

[6] Dhanabal, L., & Shantharajah, S. (2015). A Study on NSL-KDD Dataset for Intrusion Detection System Based on Classification Algorithms. *International Journal of Advanced Research in Computer and Communication Engineering*, 446-451.

[7] Volden, H., H. (2016). Anomaly detection using Machine learning techniques. Oslo: University of Oslo.

[8] Buczak, A. L., & Guven, E. (2016). A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. *IEEE communication surveys and tutorials*,1153-1175.

[9] Farnaaz, N., & Jabbar, M. (2016). Random Forest Modeling for Network Intrusion Detection System, *Procedia Computer Science*, 213-217.

[10] Aburomman, A., & Bin Ibne Reaz, A. M. (2016). A novel SVM-kNN-PSO ensemble method for intrusion detection system. *Applied Soft Computing*, 2016, pp. 360-372.

[11] Enache, A.-C., & Patriciu, V. V. (2014). Intrusions Detection Based On Support Vector Machine Optimized with Swarm Intelligence. *9th IEEE International Symposium on Applied Computational Intelligence and Informatics*, Romania: IEEE.

[12] Liao, Y., & Vemuri, V. R. (2002). Use of K-Nearest Neighbor classifier for intrusion detection. *Computers and Security*, 439-448.

[13] Roughgarden,T., Algorithms, Retrieved from Coursera: http://class.coursera.org/algo-004/lecture/preview, July 30, 2017

[14] Eberhart, R.C., & Kennedy, J. (1999). A new optimizer using particle swarm theory. *In Proceedings of the 6th international symposium on micro machine and human science* (pp. 39-43). Nagoya, Japan.

[15] Yan, X., (2011). Metaheuristic Optimization Algorithms. http://www.scholarpedia.org/article/Metaheuristic_Optimization.

[16] Çiftçioğlu, A.Ö., Doğan, E. (2019). Çelik Çerçevelerin Stokastik Yöntemler Kullanılarak Optimum Boyutlandırılması, *Konya Mühendislik Bilimleri Dergisi 7(4)*, 847-861.

[17] Karakoyun, M., Baykan, N.A., Hacibeyoglu, M. (2017). Multi-Level Thresholding for Image Segmentation with Swarm Optimization Algorithms, *International Research Journal of Electronics & Computer Engineering*, Vol:30.

[18] Celtek, S.A., Durdu, A. (2020). An Operant Conditioning Approach For Large Scale Social Optimization Algorithms. *Konya Mühendislik Bilimleri Dergisi 8(SI)*, 38-45.

[19] Beşkirli, M., Tefek, M.F. (2019). Parçacık Sürü Optimizasyon Algoritması Kullanılarak Optimum Robot Yolu Planlama, *Avrupa Bilim Teknoloji Dergisi SI*, 201-213.

[20] Tefek, M.F., Beşkirli, M. (2019). Tesis Yerleştirme (p-Hub) Probleminin Yapay Arı Kolonisi Kullanılarak Çözülmesi. *Avrupa Bilim Teknoloji Dergisi SI*, 193-200.

[21] Bansal, J., C., Sharma, H., Jadon, S., S. (2013). Aritificial Bee colony Algorithm: A Survey. *Internation Journal of advanced Intelligence Paradigms.*

[22] Lakhina, S., Joseph, S., Verma, B. (2010). Feature reduction using PCA for effect anomaly intrusion detection on NSL-KDD. *International Journal of Science Engineering and Technology*.

[23] Yan, X. (2010). Metaheuristic Optimization, Nature-Inspired Algorithms and Applications. *Cambridge University Press*.

[24] Roy, S., S., Mittal, D., and Biba, M., (2016). Random Forest Support Vector Machine and Nearest Centriod Method for Classifying Network Intrusion. *Computer Science Series*, 9–17.

[25] Wang, J., Hong, X., and Ren, R., R., (2009). A Real-time Intrusion Detection System Based on PSO-SVM. *Proceedings of the 2009 International Workshop on Information Security and Applications*, Qingdao-china.

[26] Wang, J., Li, T., and Ron, R., R. (2010). A Real-time IDS Based on Artificial Bee Colony-Support Vector Machine Algorithm. *IEEE Third International Workshop in Advanced Computational Intelligence*.

[27] Khorram, T., Network Anomaly Detection Using Optimized Macine Learning Algorithms, Master Thesis, Graduate School of Natural Sciences, Selcuk University, Turkey.