Hindawi
Complexity
Volume 2020, Article ID 5287684, 18 pages
https://doi.org/10.1155/2020/5287684

WILEY | Hindawi

Research Article

Improved TLBO-JAYA Algorithm for Subset Feature Selection and Parameter Optimisation in Intrusion Detection System

Mohammad Aljanabi,[1,2] **Mohd Arfian Ismail**,[2] **and Vitaly Mezhuyev**[3]

[1]*College of Education, Aliraqia University, Baghdad, Iraq*
[2]*Faculty of Computing, College of Computing and Applied Sciences, Universiti Malaysia Pahang, Malaysia*
[3]*Institute of Industrial Management, FH Joanneum University of Applied Sciences, Graz, Austria*

Correspondence should be addressed to Mohammad Aljanabi; mohammad.cs88@gmail.com

Received 16 January 2020; Revised 2 May 2020; Accepted 4 May 2020; Published 31 May 2020

Academic Editor: Harish Garg

Many optimisation-based intrusion detection algorithms have been developed and are widely used for intrusion identification. This condition is attributed to the increasing number of audit data features and the decreasing performance of human-based smart intrusion detection systems regarding classification accuracy, false alarm rate, and classification time. Feature selection and classifier parameter tuning are important factors that affect the performance of any intrusion detection system. In this paper, an improved intrusion detection algorithm for multiclass classification was presented and discussed in detail. The proposed method combined the improved teaching-learning-based optimisation (ITLBO) algorithm, improved parallel JAYA (IPJAYA) algorithm, and support vector machine. ITLBO with supervised machine learning (ML) technique was used for feature subset selection (FSS). The selection of the least number of features without causing an effect on the result accuracy in FSS is a multiobjective optimisation problem. This work proposes ITLBO as an FSS mechanism, and its algorithm-specific, parameterless concept (no parameter tuning is required during optimisation) was explored. IPJAYA in this study was used to update the C and gamma parameters of the support vector machine (SVM). Several experiments were performed on the prominent intrusion ML dataset, where significant enhancements were observed with the suggested ITLBO-IPJAYA-SVM algorithm compared with the classical TLBO and JAYA algorithms.

1. Introduction

Recent advancements and popularisation of network and information technologies have increased the significance of network information security. Compared with conventional network defence mechanisms, human-based smart intrusion detection systems (IDSs) can either intercept or warn of network intrusion. However, most studies on information security have focused on the ways to improve the effectiveness of smart network IDSs. The use of smart IDSs is an effective network security solution that can protect against attacks. Nonetheless, machine learning (ML) methods and optimisation algorithms are often used for intrusion detection because the detection rate of existing IDSs is low when faced with audit data that have a high overhead [1].

The execution time can sometimes increase substantially when one attempts to rise a detection accuracy. Also, the execution time may be significantly reduced but at the cost of decreased accuracy. Therefore, the feature subset selection (FSS) problem can be considered as a multiobjective optimisation problem; it has more than one solution, from which the best may be chosen. Solutions that offer superior accuracy are selected by customers who prioritise precision. Other clients choose solutions that provide reduced execution times as the best solutions, even though accuracy is compromised to a certain extent.

The teaching-learning-based optimisation algorithm (TLBO), as a novel metaheuristic, has been recently applied to various intractable optimisation problems with considerable success. TLBO is superior to many other algorithms,

such as genetic algorithms (GAs), particle swarm, and ant colony. Moreover, TLBO needs fewer parameters for tuning during execution compared with other algorithms. Thus, the combination of improved multiobjective TLBO frameworks with supervised ML techniques was proposed in the present study for FSS in multiclass classification problems (MCPs) for intrusion detection. The selection of the least number of features without causing an effect on the result accuracy in FSS is a multiobjective optimisation problem. The first objective is the number of features, and the second is the detection accuracy. TLBO remarkably outperforms other metaheuristic algorithms. Thus, ITLBO and a set of supervised SVM were deployed in this study for the selection of the optimal feature subset. JAYA is a new metaheuristic optimisation algorithm proposed by Rao (2016), which was recently deployed in several intractable optimisation problems. JAYA differed from other optimisation algorithms by not requiring parameter tuning [2]. It has been used as a benchmark function for constrained and unconstrained cases, and despite being parameterless like TLBO, it requires no learning phase, making it different from TLBO [3]. The principle of JAYA is the establishment of the problem's solution by inclining towards the best result and keeping off from the bad one. This movement depends on certain control parameters like the number of design variables, the maximum number of generations, and the size of the population. It requires no tunable control parameter before the computation phase. Thus, IPJAYA is used to tune the parameters of the SVM. In order to improve the feature selection process and SVM parameter tuning, in this paper, we propose an improved algorithm for subset feature selection using an enhanced TLBO algorithm. It uses an additional phase in TLBO to increase the information exchange between teachers and learners. SVM parameter tuning is based on the improved parallel JAYA algorithm, which uses parallel processing to increase the speed of parameter tuning. The proposed algorithm is called ITLBO-IPJAYA-SVM.

The remaining part of this paper is presented in the following manner. Section 2 reviews work related to this study, and the FSS problem is introduced in Section 3. The ITLBO is discussed in Section 4, and Section 5 explains ML applied with ITLBO. Section 6 compares the results of the ITLBO and TLBO algorithms. Finally, Section 7 concludes this study.

2. Related Work

Intrusion detection is a prevalent security infrastructure topic in the era of big data. Combinations of different ML methods and optimisation algorithms have been developed and applied in the IDS to distinguish a normal network access from the attacks. Existing combinations include fuzzy logic, cuttlefish optimisation algorithm, K-nearest neighbour, artificial neural network, particle swarm algorithm, support vector machine (SVM), and artificial immune system approaches [4]. Most methods that combine ML with optimisation algorithms outperform conventional classification methods. Numerous researchers have also proposed ML and optimisation-based IDSs [5]. Louvieris et al. [6] proposed a novel combination of techniques (K-means clustering, naïve Bayes (NB), Kruskal–Wallis (KW), and C4.5) that pinpointed attacks as anomalies with high accuracy even within cluttered and conflicted cyber-network environments. Furthermore, the inclusion of the NB feature selection and the KW test in this method facilitates the classification of statistically significant and relevant feature sets, including a statistical benchmark for the validity of the method, while the detection of SQL injection in this method remains low. De la Hoz et al. [7] presented a method for NIDS that was based on self-organising maps (SOMs) and principal component analysis (PCA). Noise within the dataset and low-variance features were filtered by means of PCA and Fisher discriminant ratio. This procedure uses the most discriminative projections based on the variance explained by the eigenvectors. Prototypes generated by the self-organising process are modelled by a Gaussian, where d is the number of SOM units. Therefore, this system must be trained only once; however, the main limitation of this work is that the detection rate remains low. Bamakan et al. [8] proposed a chaos-particle swarm optimisation method to provide a new ML IDS based on two conventional classifiers: multiple-criteria linear programming and an SVM.

The proposed approach has been applied to simultaneously set the parameters of these classifiers and provide the optimal feature subset. The main drawback of this work is the long training time needed. Therefore, even though these combinations can improve the performance of IDSs in terms of learning speed and detection rate compared to conventional algorithms, further improvement is needed. The performance of most IDSs is affected in terms of classification accuracy and training time by an increase in the number of audit data features. The present paper proposes the use of the TLBO technology to address this issue through the supply of a fast and accurate optimisation process that can improve the capability of an IDS to find the optimal detection model based on ML. In the TLBO algorithm proposed by Rao et al. [9], the optimisation process for mechanical design problems does not need any user-defined parameter. This novel technique was tested on different benchmark functions, and the results demonstrated that the developed TLBO outperformed particle evolutionary swarm optimisation, artificial bee colony (ABC), and cultural DE. Das and Padhy [10] studied the possibility of applying a novel TLBO algorithm to the selection of optimal free parameters for an SVM regression model of financial time-series data by using multi-commodity futures index data retrieved from multicut crossover (MCX). Their experimental results showed that the proposed hybrid SVM-TLBO model successfully identified the optimal parameters and yielded better predictions compared to the conventional SVM. Das et al. [11] proposed an extension of the hybrid SVM-TLBO model by introducing a dimension reduction technique whereby the number of input variables can be reduced by using PCA, kernel PCA (KPCA), and independent component analysis (ICA) (three common dimension reduction methods). This

study also examined the feasibility of the proposed model using multicommodity futures index data retrieved from MCX. Rao et al. [12] confirmed the superiority of the model compared to some population-inspired optimisation frameworks. Rao and Patel [13] investigated the effect of sample size and number of generations on algorithmic performance and concluded that this algorithm can be easily applied to several optimisation cases. Črepinšek et al. [14] solved the problems presented in [9, 12] by using TLBO. Nayak et al. [15] developed a multiobjective TLBO in which a matrix of solutions was created for each objective. The teacher selection process in TLBO is mainly based on the best solution presented in the solution space, and learners are taught to merely maximise that objective. All the available solutions in the solution space were sorted to generate a collection of optimal solutions. Xu et al. [16] presented multiobjective TLBO based on different teaching techniques. They used a crossover operator (rather than a scalar function) between solutions in the teaching and learning phases. Kiziloz et al. [17] suggested three multiobjective TLBO algorithms for FSS in binary classification (FSS-BCP). Among the presented methods, a multiobjective TLBO with scalar transformation was found to be the fastest algorithm, although it provided a limited number of nondominated solutions. Multiobjective TLBO with nondominated selection (MTLBO-NS) explores the solution space and produces a set of nondominated solutions but requires a long execution time. Multiobjective TLBO with minimum distance (MTLBO-MD) generates solutions that are similar to those of MTLBO-NS but in a significantly shorter time. The proposed multiobjective TLBO algorithms have been evaluated in terms of performance using LR, SVM, and extreme learning machine (ELM). Wang et al. suggested a novel "alcoholism identification method from healthy controls based on a computer-vision approach." [18] This approach relied on three components—the proposed wavelet Renyi entropy, feedforward neural network, and the proposed three-segment encoded JAYA algorithm. The results showed the proposed method exhibits good sensitivity, but the accuracy still needs improvements; Migallón et al. [19] developed parallel algorithms and presented their detailed analysis. They developed a hybrid algorithm that exploited inherent parallelism at two different levels. The lower level was exploited by parallel shared-memory platforms, while the upper level was exploited by distributed shared memory platforms. The results of both algorithms were good, especially in scalability. Hence, the proposed hybrid algorithm successfully used a number of processes with near-perfect efficiencies. The experiments showed that the method used about 60 processes to achieve near-ideal efficiencies as analysed on 30 unconstrained functions. Gong [20] suggested a "novel E-JAYA algorithm for the performance enhancement of the original JAYA algorithm." The proposed E-JAYA used the average of the better and worse groups to derive the best solution. The solution provided by the proposed E-JAYA had better accuracy than that of the original JAYA. The swarm behaviours were considered in the E-JAYA rather than considering the best and worst individual behaviours. The performance of E-JAYA was assessed on 12 benchmark functions of varying dimensionality.

Another study proposed an effective demand-side management scheme for residential HEMS [21]. The system was proposed for peak creation prevention to reduce electricity bills. This study applied JAYA, SBA, and EDE to realise its objectives; it also deployed the TOU pricing scheme for electricity bill computation. From the result, JAYA was sufficient in reducing electricity bill and PAR, thereby achieving customer satisfaction. Furthermore, the SBA outperformed JAYA and EDE in achieving user comfortability as it related negatively with an electricity bill. Yu et al. [22] developed improved JAYA (IJAYA) for steady and accurate PV model parameter estimation by incorporating a self-adaptive weight for the adjustment of the propensity of reaching the best solution and avoiding the bad solution while searching. The weight helps in ensuring the framework achieves the possible search region early and to perform local search later. Furthermore, the algorithm contains a learning strategy derived from other individuals' experiences, which was randomly used for population diversity improvement. Table 1 shows the lacks and limitation of IDS studies mentioned in the related work.

3. Feature Subset Selection Problem

This section explains the representation of the features and the problem of choosing the best feature subset. FSS refers to the selection of feature subsets from a larger feature set. FSS reduces the number of features in a dataset, thereby preventing complex calculations and improving the speed and performance of classifiers. Several definitions of FSS exist in literature [23]; some definitions deal with the reduction in size of the selected subset, while others focus on the improvement of prediction accuracy. FSS is essentially a process of constructing an effective subset that represents the information contained in a dataset by eliminating redundant and irrelevant features. FSS mainly aims at finding the least number of features without having a significant influence on classification accuracy. Owing to the complicated nature of optimal subset feature extraction, as well as the nonexistence of a polynomial-time algorithm for addressing it, FSS has been classified as an NP-hard problem [24]. There are four steps in typical FSS [23]; the first step involves the selection of candidate features that will constitute the subsets, while the second step is the evaluation and comparison of these subsets with each other. In the third step, a check is made for the satisfaction of the termination condition; otherwise, the first and second steps will be repeated. The final step checks if the optimal feature subset has been established based on prior knowledge. With these two major aims, FSS can be considered a multiobjective problem. A formal definition of finding optimal solutions through the satisfaction of both objectives is given in the following equation:

Table 1: IDS existing work.

Ref.	Limitation
[6]	Detection of SQL injection is low
[7]	Detection rate is low
[8]	Long training time

$$\begin{aligned} &\text{Min} && (f1) \\ &\text{Max} && (f2) \\ &\text{subject to} && f1 = |k| \\ & && f2 = \text{accuracy}(k), \quad \text{where } k \subseteq K. \end{aligned} \tag{1}$$

where k is the subset of the original dataset K which optimises $f1$ and $f2$ (the objectives).

The establishment of the best solution or the decision on the improved condition of a new individual is a complicated task in a multiobjective optimisation process. This is due to the chances of enhancement in one objective, causing a reduction in the other.

4. Improved TLBO Algorithm

The ITLBO algorithm was executed at the FSS phase in this study. The ITLBO algorithm was initialised by randomly generated initial population, namely, the teacher and a set of students, which represents the set of solutions. To represent the features in the ITLBO algorithm, ITLBO borrowed the crossover and mutation operators from GA by representing the features as chromosomes (one of the GA properties). To update this chromosome, crossover and mutation operators were used. In the population (called a classroom), each solution is taken as an individual/chromosome (Figure 1). A feature gene of a chromosome with a value of 1 is considered as selected, while a value of 0 denotes otherwise. Figure 1 shows a sample of the dataset; regarding Figure 2, features A, B, C, D, E, I, K, and L were selected (their values are 1), while features F, G, H, and J were not (their values are 0). The TLBO algorithm runs through iterations where the teacher is the best individual in the population and the rest of the individuals become the students. Having selected the teacher, ITLBO works in three phases: Teacher, Best Classmates (Learner Phase 1), and Learner Phase 2. In the Teacher phase, the teacher enhances the knowledge of each student by sharing knowledge with them, but in the Best Classmates phase, two best students are selected and assigned the task of interacting with the other students. In the Learner phase, there is a random interaction among the students in a bid to enhance their levels of knowledge. New chromosomes are generated in the proposed ITLBO using "half-uniform crossover and bit-flip mutation operators" which are special crossover operators (Figures 3 and 4). Two-parent chromosomes (could be a teacher, a student, or two students) are needed for the crossover operator. The crossover operator relies on the information of the two-parent chromosomes; if both parents feature the same gene, the gene is kept, but whenever there are different feature genes in both parents, a parent's gene is randomly chosen. Only one new chromosome is generated from this operation.

Figure 1: Schematic representation of a chromosome: 1 = selected features; 0 = unselected features.

The "bit-flip mutation" works on a single chromosome when trying to manipulate a single gene based on a probabilistic ratio. If the gene has a zero value, it will be updated as one, or vice versa. In the proposed ITLBO algorithm, nondominated sorting and selection were used. The dominance of an individual over another individual is determined strictly on the basis of whether a minimum of one of its objectives is superior to that of the other while keeping all the other objectives the same.

A nondominated scenario arises when there is no possibility of an individual being dominated by another. The front line of the solution set is filled by the nondominated individuals. Those that are closest to the ideal point in the front line are chosen as the teachers. All the teachers teach all students discretely at the Teacher, Best Classmate, and Learner phases. The details of the ITLBO algorithm are presented in Figures 5 and 6. The detail steps of ITLBO are as follows:

(i) Step 1: initialise the population randomly with each population having a different set of features from 1 to a maximum number of features (41 in NSL-KDD). This step is captured in line 2 of Figure 5.

(ii) Step 2: choose the best individual as a teacher. The chosen teacher interacts with all other individuals separately, and a crossover is applied with each one, and then a mutation is applied to all the resulting individuals. The operators used are half-uniform crossover and bit-flip mutation operators (represented in lines 4 to 5 in Figure 5).

(iii) Step 3: check the population (chromosome) that results from the crossover and mutation; if the new chromosome is better than the old, then the new one is kept; otherwise, the old one is retained. All the aforementioned steps are collectively called the Teacher phase because all individuals learn from the best one (the teacher). This step is represented in lines 6 to 13 in Figure 5.

(iv) Step 4: after that, Learner Phase 1 or learning from the best classmates is started. This phase begins with the fifth step which is the selection of the best two individuals as students and applying a crossover between them followed by a mutation. If the new one is better than the previous two students, then the newer choice is kept; otherwise, the older best choice is kept. This process is repeated with all other individuals (students). At this point, Learner Phase 1 has terminated (viewed in lines 14 to 27 in Figure 5).

(v) Step 5: this step is Learner Phase 2 which involves choosing two random individuals (students) between whom a crossover is applied followed by a mutation on the new individual. If the new

	A	B	C	D	E	F	G	H	I	J	K	L
1	0	1	1	1	215	45076	0	0	0	0	0	1
2	0	1	1	1	162	4528	0	0	0	0	0	1
3	0	1	1	1	236	1228	0	0	0	0	0	1
4	0	1	1	1	233	2032	0	0	0	0	0	1
5	0	1	1	1	239	486	0	0	0	0	0	1
6	0	1	1	1	238	1282	0	0	0	0	0	1
7	0	1	1	1	235	1337	0	0	0	0	0	1
8	0	1	1	1	234	1364	0	0	0	0	0	1
9	0	1	1	1	239	1295	0	0	0	0	0	1
10	0	1	1	1	181	5450	0	0	0	0	0	1

Figure 2: Sample of the dataset.

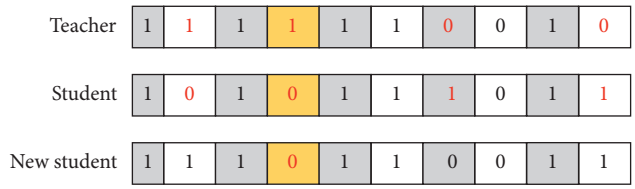

Figure 3: Crossover operator.

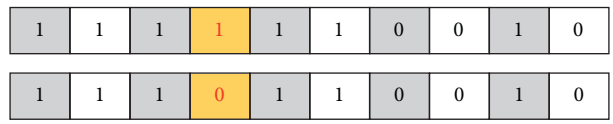

Figure 4: Mutation operator.

```
(1)  Start
(2)      Initialize population
(3)      Calculate_weighted_average_of
         _individuals (population);
(4)  for (k: =1 to number_of_generations) do
(5)      Xteacher: = Best_individual
(6)  Learning from Teacher */teacher phase
(7)      for (i: =1 to number_of_individuals) do
(8)      Xnew: = Crossover (Xteacher, Xi);
(9)      Xnew: = Mutation (Xnew);
(10)     if (Xnew is better than Xi) then
(11)         Xi: = [Xnew]
(12)     End if
(13) End for
(14) Learning from Best Classmates */learner phase 1
(15)     for (i: =1 to number_of_individuals) do
(16)     m: = Select_best_individual_from
         (population);
(17)     n: = Select_best_individual_from (population);
(18)     n ≠ m ≠ teacher*/
(19)     Xnew: = Crossover (Xm, Xn);
(20)     Xnew: = Mutation (Xnew);
(21)     if (Xnew is better than Xm) then
(22)         Xm: = Xnew
(23)     End if
(24)     if (Xnew is better than Xn) then
(25)         Xn: = Xnew;
(26)     End if
(27) End for
(28) Learning from Classmates */learner phase 2
(29) for (i: =1 to number_of_individuals) do
(30)     m: = Select_random_individual_from
         (population);
(31)     n: = Select_random_individual_from
         (population); n ≠ m ≠ teacher*/
(32)     Xnew: = Crossover (Xm, Xn);
(33)     Xnew: = Mutation (Xnew);
(34)     if (Xnew is better than Xm) then
(35)         Xm: = Xnew;
(36)     End if
(37)     if (Xnew is better than Xn) then
(38)         Xn: = Xnew;
(39)     End if
(40) End for
(41) Is termination criterion satisfied? if yes go to line
     42, else continue
(42) End for
(43) Show_the_pareto_optimal_set (population)
(44) End
```

Figure 5: ITLBO algorithm.

individual is better than the old two students, then the new one is kept; otherwise, the best old one is retained. This step is repeated with all other students. At this point, the main three stages of ITLBO have been completed, and a check should be carried out on whether the termination criteria have

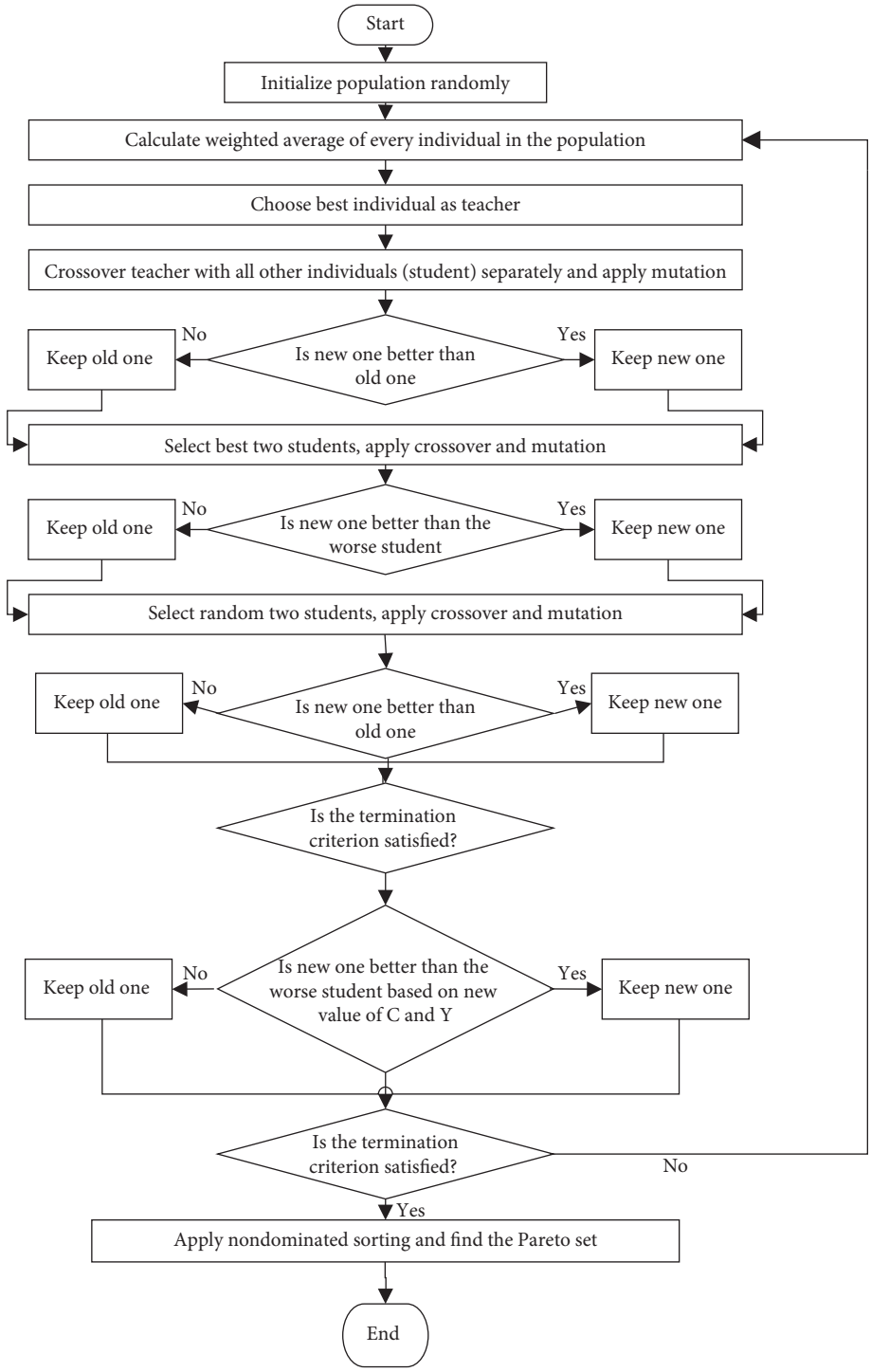

Figure 6: ITLBO flowchart.

been satisfied or not. If the termination criteria were satisfied, proceed to the next step; otherwise, the main three stages are repeated (Teacher phase, Learner Phase 1, and Learner Phase 2). This step is represented in lines 28 to 40 in Figure 5.

(vi) Step 6: the final step is the application of nondominated sorting to the result. Nondominated sorting means no result (individual) is better than all other individuals. This step can be viewed in line 43 in Figure 5.

5. Parameter Optimisation

After selecting the optimal subset feature, several SVM parameters will be tuned. The tuning of SVM parameters is a problem which can determine algorithm performance. The

radial basis function (RBF), kernel function of the SVM, is employed for the conversion of the completely nonseparable problem into a separable or approximate separable state. The RBF kernel parameter γ suggests data distribution to a new feature space, while parameter C suggests the level of penalty for the classification error in the linear nonseparable case. Equations (2) and (3) represent the cost and gamma, respectively. In the next section, the two parameters (C and γ) were tuned by using the IPJAYA algorithm.

$$w, b \frac{1}{2}\|w\| \frac{2}{2} + C \sum_{n} \zeta n s.t. yn\left(w^t x_n + b\right) \geq 1 - \zeta n, \tag{2}$$

$$k\left(x_n, x_m\right) = \exp\left(-\gamma \left\| x_n - x_m \right\| \frac{2}{3}\right). \tag{3}$$

6. Improved Parallel JAYA Algorithm

The JAYA algorithm needs improvements to work better. One of the observations on the JAYA algorithm is that if we sort the populations from best to worst and divide them into two groups, the best and the worst solutions. Obviously, the optimal solution is located in the best solution group [2]. Based on this observation, an improvement has been done in the JAYA algorithm; rather than selecting the best and worst cases from the whole solutions, which puts the worst solution further from the best solution and increase the iterations needed to reach the optimal solution, the solutions were divided into two groups. The best solution is chosen from the best solution group as "Best," and the best solution from the worst solution group is the "Worst." This procedure reserves the population's diversity and makes the solution start from a point closer to the optimal solution and decreases the number of iterations needed to reach the optimal solution. In the proposed work, JAYA algorithm was improved to optimise two parameters of the SVM classifier simultaneously. Figure 7 shows the flowchart of IPJAYA, while Figure 8 shows the IPJAYA algorithm followed by the detailed steps of IPJAYA.

The detailed steps of IPJAYA are shown as follows:

(i) Step 1: select the population size and the number of design variables, as well as initialise the termination condition. To explain the parameter optimisation in detail, we assume the following scenario: population size = 3, design variables = 2, and termination criterion = 2 iterations. The value of the population is the value of parameters C and γ; in this scenario, each one has 3 values. These values were initialised randomly for C between 0.001 to 100 and for γ between 0.0001 to 64. Table 2 shows the values of C and γ.

(ii) Step 2: SVM needs three things to classify any labelled data, i.e., features to choose, value of C parameter, and value of γ parameter. This step can be viewed in line 2 of Figure 8.

(iii) Step 3: the next step is to evaluate each value for both C and γ separately by using the SVM and on the first student from Learner Phase 2 after applying crossover and mutation as shown in Table 3.

To continue the optimisation process, the population is arranged from best to worst and split into two groups (Best and Worst groups) as shown in Table 4.

The same procedure is repeated for γ parameter, and this time, C is by default, and the new value of γ is 11.006.

(iv) Tables 5 and 6 show the details of γ parameter.

(v) Step 4: the result will be considered as the objective function for both C and γ and then compared with other populations and continued until the termination criterion is satisfied. This step can be viewed in line 7 of Figure 8.

These two new values for C and γ will be evaluated using the same subset feature at the same time as shown in Table 7. This step can be viewed in lines 5 to 6 of Figure 8.

7. The Proposed Method

This section describes the proposed combination of three different algorithms. Each algorithm has a different task to do, and these tasks complete the work of the model. The first algorithm is the ITLBO whose task is to choose the optimal subset feature from the whole features. The second algorithm is IPJAYA algorithm, and its task is to optimise the parameters of the SVM. The third algorithm is the SVM classifier which takes the outcome of the first two algorithms to determine if the processed traffic is intrusion or normal traffic. Figure 9 shows the flowchart of the proposed method. Figure 10 shows the pseudo-code of the proposed method, while Figure 11 details the proposed steps of the IPJAYA-ITLBO-SVM method.

The detail steps of ITLBO-IPJAYA-SVM are as follows:

(i) Step 1: initialise the population randomly. Each population is a different set of features from 1 to the maximum number of features (41 in NSL-KDD). This step can be viewed in line 2 of Figure 10.

(ii) Step 2: calculate the weighted average of every individual population. This step can be viewed in line 3 of Figure 10.

(iii) Step 3: choose the best individual as a teacher. The chosen teacher interacts with all other individuals separately. Apply crossover with each one, and then apply mutation to all resulted individuals. The crossover used is called the half-uniform crossover and bit-flip mutation operator. This step can be viewed in lines 4 to 5 of Figure 10.

(iv) Step 4: check the population (chromosome) resulting from crossover and mutation; if the new one is better than the old one, keep the new one; otherwise, retain the old one. The best and worst populations refer to the degree of accuracy during

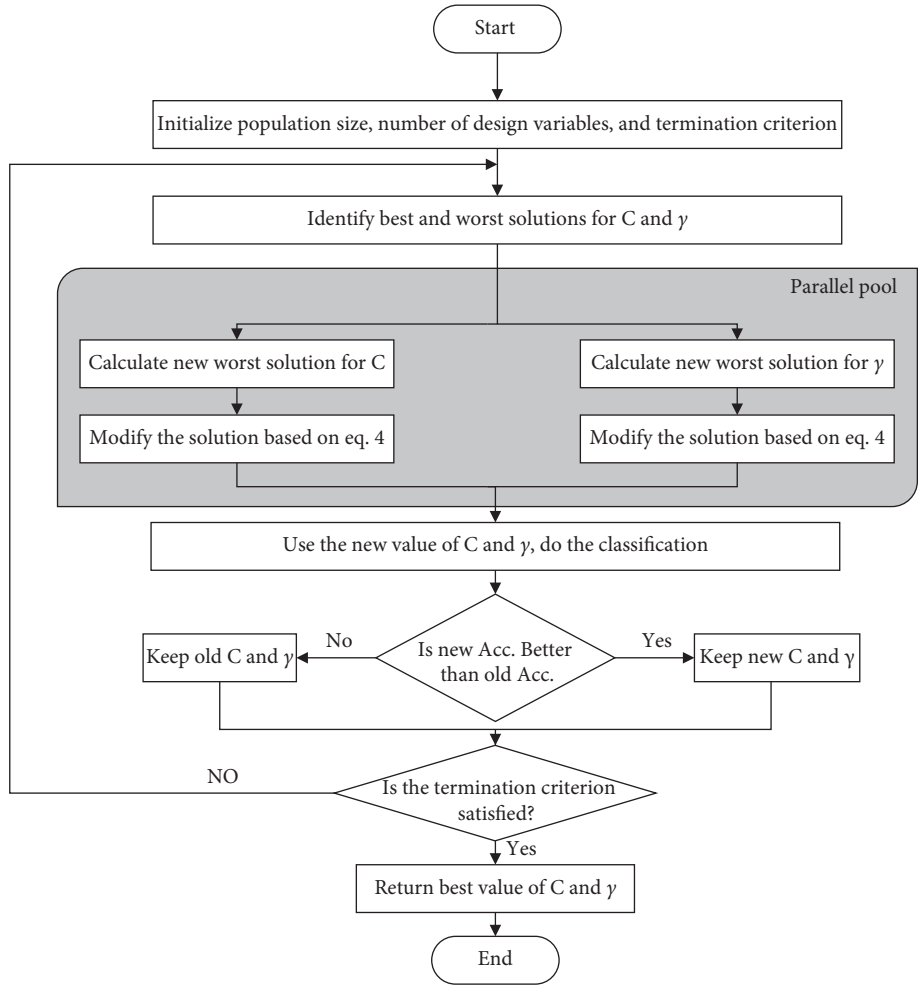

Figure 7: IPJAYA flowchart.

(1) Start
(2) Initialise the population size, number of designed variables, and termination criteria
(3) Repeat Steps 3–6 until the termination criteria are met
(4) Arrange the solutions from best to worst and split the solutions into two groups -best and worst solutions
(5) Make the best solution in best group as best, and make the best solution in worst group as worst
(6) Modify the solution based on the following equation:

$$Y'_{j,k,I} = Y_{j,k,I} + r_{1,k,I}\left(Y_{j,best,I} - \left|Y_{j,k,I}\right|\right) - r2,k,I\left(Y_{j,worst,I} - \left|Y_{j,k,I}\right|\right)$$

(7) Update the previous solution if $Y'j,k,I > Yj,k,I$, otherwise, do not update the previous solution
(8) Display the established optimum solution
(9) End

Figure 8: IPJAYA algorithm.

Table 2: C and γ values.

C	γ
20	10
1	2
10	0.7
0.1	1

Table 3: Evaluation of C.

C	γ	Subset feature	Accuracy (objective function)
20	Default	Fixed	0.97
1	Default	Fixed	0.899
10	Default	Fixed	0.994
0.1	Default	Fixed	0.99

Table 4: Best and Worst groups for C.

C	γ	Subset feature	Accuracy (objective function)	
10	Default	Fixed	0.994 best of best	Best group
0.1	Default	Fixed	0.99	
20	Default	Fixed	0.97 best of worst	Worst group
1	Default	Fixed	0.899	

Table 5: Accuracy based on γ.

C	γ	Subset feature	Accuracy (objective function)
20	Default	Fixed	0.97
1	Default	Fixed	0.899
10	Default	Fixed	0.994
0.1	Default	Fixed	0.99

Table 6: Best and Worst groups for γ.

C	γ	Subset feature	Accuracy (objective function)	
Default	0.7	Fixed	0.9941 best of best	Best group
Default	1	Fixed	0.99	
Default	2	Fixed	0.98 best of worst	Worst group
Default	10	Fixed	0.97	

Table 7: Evaluation of features based on new C and γ.

C	γ	Subset feature	Accuracy (objective function)
14.2	11.006	Fixed	Result

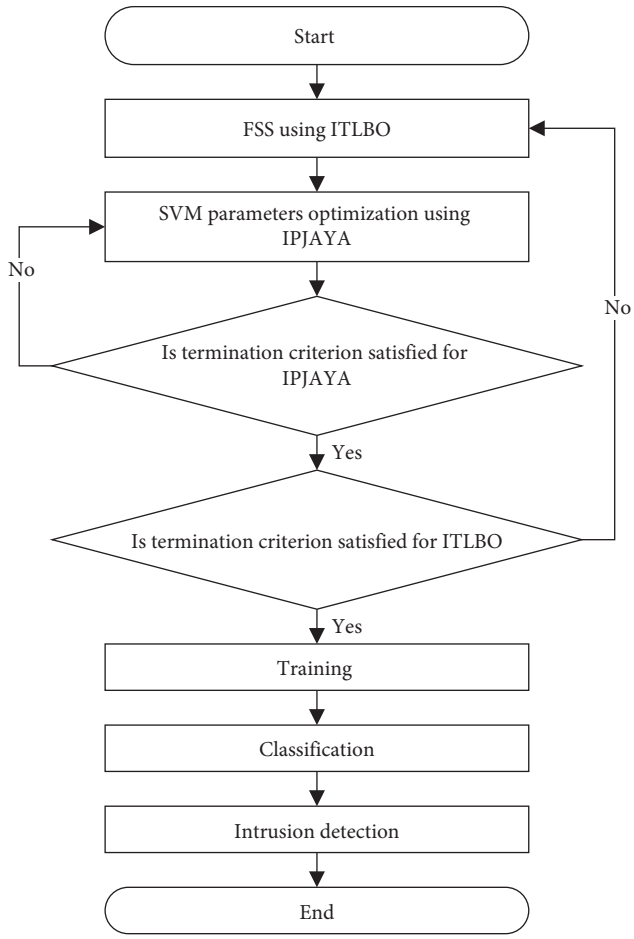

Figure 9: Proposed method flowchart.

classification. All the aforementioned steps are called the Teacher phase because all individuals learn from the best one (teacher). After that, Learner Phase 1 is started. This step can be viewed in lines 6 to 13 of Figure 10.

(v) Step 5: select the best two individuals as students, and apply crossover between these two students. Then, apply mutation on the new one. If the new one is better than the old two students, keep the new one; otherwise, keep the best old one, and apply this with all other individuals (students). The students are chosen once and will not be chosen again. At this point, Learner Phase 1 has ended. This step can be viewed in lines 14 to 27 of Figure 10.

(vi) Step 6: Learner Phase 2 is initiated with two objectives; one is to optimise the SVM parameters, and the other is to make students learn from each other. This phase starts with choosing two random individuals (students) and then applying crossover between these

```
(1)  Start
(2)      Initialize population
(3)      Calculate_weighted_average_of_individuals
(4)  for (k: =1 to number_of_generations) do
(5)      Xteacher: = Best_individual
(6)  Learning from Teacher */teacher phase
(7)      for (i: = 1 to number_of_individuals) do
(8)      Xnew: = Crossover (Xteacher, Xi);
(9)      Xnew: = Mutation(Xnew);
(10)     if (Xnew is better than Xi) then
(11)         Xi: = [Xnew]
(12)         End if
(13)     End for
(14) Learning from Best Classmates */learner phase 1
(15)     for (i: = 1 to number_of_individuals) do
(16)     m: = Select_best_individual_from
         (population);
(17)     n: = Select_best_individual_from (population);
(18)     n ≠m ≠ teacher*/
(19)     Xnew: = Crossover (Xm, Xn);
(20)     Xnew: = Mutation(Xnew);
(21)     if (Xnew is better than Xm) then
(22)         Xm: = Xnew
(23)         End if
(24)     if (Xnew is better than Xn) then
(25)         Xn: = Xnew;
(26)         End if
(27)     End for
(28) Learning from Classmates */learner phase 2
(29)     for (i: =1 to number_of_individuals) do
(30)     m: =Select_random_individual_from
         (population);
(31      n: =Select_random_individual_from
         (population); n ≠m ≠ teacher*/
(32)     Xnew: = Crossover (Xm, Xn);
(33)     Xnew: = Mutation (Xnew);
(34) IPJAYA algorithm for SVM parameter
(35) Initialize the population size, number of designed
     variables, and termination criteria (IPJAYA)
(36) Arrange the solutions from the best to the worst
     and Split the solutions into two groups best and
     worst solutions
(37) Modify the solutionY'j, k, I = Yj, k, I + r1, k, I
     (Yj, best, I – |Yj, k, I|) – r2, k, I (Yj, worst, I – | Yj, k, I|)
(38  Update the previous solution if Y'j, k, I > Yj, k, I,
     otherwise, do not update the previous solution.
(39)     return best value of C and Y
(40)     if ( Xnew is better than Xm) then
(41)         Xm: = Xnew;
(42)         End if
(43)     if (Xnew is better than Xn) then
(44)         Xn: = Xnew;
(45)         End if
(46)     End for
(47) Is termination criterion satisfied? if yes go to line
     42, else continue
(48)  End for
(49) Show_the_pareto_optimal_set (population)
(50)  End
```

Figure 10: Proposed method pseudo-code.

two students and applying mutation on the new individual. After that and before the classification process is initiated, check if the new student is better than the old two students. The SVM parameter optimisation is started using IPJAYA; this process starts at the 29^{th} step by initialising the population size, the number of design variables, and the termination criteria for IPJAYA. The population size can be set before the execution, and each population is generated randomly. The designed variables are the two parameters of the SVM which need to be optimised. The termination criteria can be the number of iterations; after that, each population for each parameter is evaluated separately (which one gives better accuracy) followed by a parallel poll for each parameter, sorting the population from best to worst (best accuracy to worst accuracy), and separating them into two groups (best and worst groups). The best population in the best group is chosen as best, and the best population in the worst group is chosen as worst. Then, the population is modified based on equation in Figure 8 and updated if the new one is better than the old one. IPJAYA is repeated until the termination criterion is satisfied. The final step of IPJAYA is to deliver the best value of the two parameters to be used by the SVM. At this point, the parameter optimisation has ended, and Learner Phase 2 continues in the next step. This step can be viewed in lines 28 to 39 of Figure 10.

(vii) Step 7: evaluate the individuals (chromosome) by using the outcome of IPJAYA. If the new individual is better than the old two students, keep the new one; otherwise, keep the best old one. Apply this step to all other students. At this step, the main three stages of the ITLBO have finished. The next step is to check for the satisfaction of the termination criteria; if satisfied, proceed to the next step. Otherwise, the main three stages are repeated. This step can be viewed in lines 40 to 48 of Figure 10.

(viii) Step 8: the last step is to apply nondominated sorting on the result. Nondominated sorting means no result (individual) is better than all the other individuals. This step can be viewed in line 49 of Figure 10.

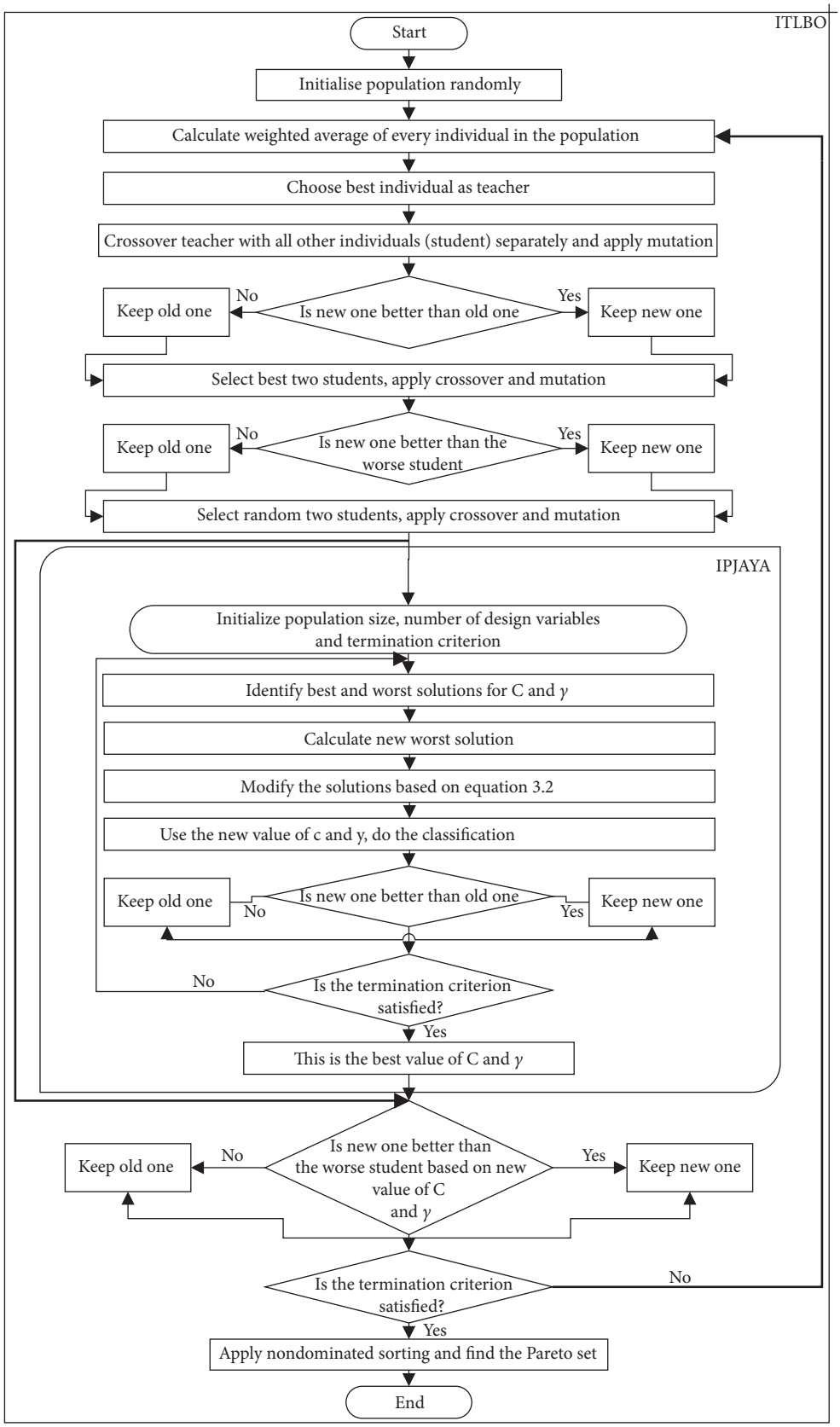

Figure 11: Details of the proposed method.

8. Evaluation Metrics

The metrics, measures, and validation procedures used in the evaluation of the experimental data were reviewed in this section. The literature review showed that most studies use overall accuracy as the major performance measure for ID systems. However, other metrics and validation measures have also been mentioned. Some works have detailed the information on FAR detections and missed detections which are all useful system performance evaluation measures. The following section details the analysis based on standard metrics for objective evaluation of the results achieved by various classification methods. The performance of the system was evaluated using several metrics based on the NSL-KDD and CICIDS 2017 datasets. A detailed description of learning performance measures has been provided by Singh et al. and Sokolova and Guy [25, 26], while Phoungphol et al. [27] detailed the imbalanced dataset issues. One of these metrics is the accuracy as given in the following equation:

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{TF} + \text{FP}}. \tag{4}$$

Accuracy is the capability of the classifier in predicting the actual class; here, TP = true positive, TN = true negative, FP = false positive, and FN = false negative.

Several metrics can be computed from the confusion matrix. The false-positive rate (FPR) is another metric; it is the percentage of the samples incorrectly predicted as positive by the classifier. It is calculated by using the following equation:

$$\text{false} - \text{positive rate}\,(\text{FPR}) = \frac{\text{FP}}{\text{FP} + \text{TN}}. \tag{5}$$

The false-negative rate (FNR) is the percentage of the data incorrectly classified by the classifier as negative. It is calculated by using the following equation:

$$\text{false} - \text{negative rate}\,(\text{FNR}) = \frac{\text{FN}}{\text{TP} + \text{FN}}. \tag{6}$$

The detection rate (DR) is the percentage of the samples correctly classified by the classifier to their correct class. It is calculated by using the following equation:

$$\text{detection rate} = \frac{\text{FP}}{\text{TP} + \text{FP}}. \tag{7}$$

The recall quantifies the number of correct positive predictions made out of all positive predictions. It is calculated by using the following equation:

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{8}$$

F-Measure provides a way to combine both detection rate and recall into a single measure that captures both properties. It is calculated by using the following equation:

$$F - M = \frac{(2 * \text{DR} * \text{recall})}{(\text{DR} + \text{recall})}. \tag{9}$$

The results were validated by using k-fold cross-validation technique [27–30]. This technique requires a random partitioning of the data into k different parts, and one part is selected from each iteration as testing data, while the other $(k-1)$ parts are considered as the training dataset. All the connection records are eventually used for training and testing. For all experiments, the value of k is taken as 10 to ensure low bias, low variance, low overfitting, and good error estimate [28].

9. Dataset Preprocessing and Partitioning

The whole dataset is preprocessed in this stage. It consists of two steps, i.e., scaling and normalisation. In the scaling step, the dataset is converted from a string representation to a numerical representation. For example, the class label in the dataset contains two different categories, "Normal" and "Attack." After implementing this step, the label is changed to "1" and "0," where "1" means normal case, while "0" means attack. The second step is normalisation [31]. The normalisation process removes noise from the dataset and decreases the differences in the ranges between the features. In this work, the Max-Max normalisation method was used as shown in the following equation:

$$Fi = \frac{Fi - \text{Min}i}{\text{Max}i - \text{Min}ii}, \tag{10}$$

where Fi represents the current feature that needs to be normalized and Min$i$ and Max$i$ represent the minimum and the maximum value for that feature, respectively. The objective function represents the accuracy of the SVM when it is evaluated on the validation set. The validation set is a part of the training set. In order to make the validation fairer, K-fold validation can be used. The value K is 10. The NSL-KDD and CICIDS 2017 datasets were used to evaluate the performance of the proposed models.

10. NSL-KDD Dataset

In this study, NSL-KDD datasets were used to evaluate the proposed method. This dataset was suggested in 2009 by Tavallaee et al. [32] due to the drawbacks of KDD CUP99. The NSL-KDD is a variant of the KDD CUP 99 dataset in which the redundant instances were discarded followed by the reconstitution of the dataset structure [29]. The NSL-KDD dataset is commonly used for evaluating the performance of new ID approaches, especially anomaly-based network ID. There are a reasonable number of testing and training records in the NSL-KDD. The training set (KDDTrain+) consists of 125,973 records, while the testing set (KDDTest+) contains 22,544 records. In this dataset, each traffic record has 41 features (six symbolic and 35 continuous) and one class label (Table 7). The features are classified into basic, content, and traffic types (Table 8). Attack classification in the NSL-KDD is based on the feature characteristics [33]. The NSL-KDD dataset can be downloaded from https://www.unb.ca/cic/datasets/nsl.html.

Table 8: NSL-KDD dataset.

Attack classes	22 types of attacks	No. of instances
Normal		67,343
DoS	smurt, neptune, pod, teardrop, back, land,	45,927
R2L	phf, ftp-write, imap, multihop, warezclient, warezmaster, spy, guess password	995
U2R	perl, loadmodule, buffer-overflow, rootkit	52
Probing	portsweep, ipsweep, satan, nmap	11,656

Table 9: CICIDS dataset.

Attack class	14 types of attacks	No. of instances
Benign (normal)		2,359,087
DOS	DDoS, slowloris, Heratbleed, Hulk, GoldenEye, Slowhttptest	294,506
PortScan	Portscan	158,930
Bot	Bot	1,966
Brute-Force	FTP-Patator, SSH-Patator	13,835
Web attack	Web attack XSS, web attack SQL injection, web attack brute force	2,180
Infiltration	Infiltration	36

11. CICIDS 2017 Dataset

The CICIDS 2017 dataset consists of benign and the most current common attacks, which mimic real-world data (PCAPs). It also contains the results of a network traffic analysis obtained by using a CICFlowMeter; the flows are labelled based on the timestamp, source and destination ports, source and destination IPs, protocols, and attack. The CICIDS 2017 dataset satisfies the 11 indispensable features of a valid IDS dataset, namely, anonymity, available protocols, feature set, attack diversity, complete capture, complete interaction, complete network configuration, complete traffic, metadata, heterogeneity, and labelling [34]. There are 2,830,540 rows in the CICIDS 2017 devised on eight files with each row containing 79 features. In the CICIDS 2017, each row is labelled as benign or as one of the 14 attack types. A summary of the distribution of different attack types and the benign rows is presented in Table 9.

12. Results of ITLBO-IPJAYA vs. ITLBO and ITLBO-JAYA

This section provides the results of the improved method-based ITLBO-IPJAYA algorithm. This method selects the best features and updates the value of SVM parameters. This work proposed the idea of "parallel execution" to update the SVM parameters. The parameters for ITLBO, ITLBO-JAYA, and ITLBO-IPJAYA methods used in this study are shown in Table 10.

The NSL-KDD dataset is used to evaluate the three methods, and the evaluation metrics used are maximum accuracy (Max. Acc.), average accuracy (AVR. Acc.), detection rate (DT), false alarm rate (FAR), false negative rate (FNR), F-measure (F-M), recall, and error rate (ER). Table 11 shows the comparison in results among ITLBO, ITLBO-JAYA, and ITLBO-IPJAYA.

The results show that ITLBO-IPJAYA performs better than ITLBO and ITLBO-JAYA in all metrics. Figure 11 shows the comparison results based on the accuracy of ITLBO, ITLBO-JAYA, and ITLBO-IPJAYA.

Table 10: Parameters used in this study; margin.

Parameter	Value
Population size for ITLBO	40
Number of generations for ITLBO	60
Population size for JAYA	40
Number of generations for JAYA	60
Population size for IPJAYA	40
Number of generations for IPJAYA	60
Population size for ITLBO	40
Number of generations for IPJAYA	60
Crossover type	Half-uniform
Mutation type	Bit-flip

Figure 12 shows a comparison between ITLBO-JAYA and ITLBO-IPJAYA based on the number of iterations. It shows that ITLBO-IPJAYA performs better than ITLBO-JAYA even with less number of iterations. The increase in rate of accuracy for ITLBO-IPJAYA is higher than ITLBO-JAYA. The figure shows that ITLBO-IPJAYA with 20 iterations performs better than ITLBO-JAYA with 30 iterations and that ITLBO-IPJAYA performs better than ITLBO-JAYA with less number of iterations. This means there is less complexity and less execution time for ITLBO-IPJAYA. Figure 13 shows the average FAR of the three methods, showing that ITLBO-IPJAYA performs better than ITLBO and ITLBO-JAYA even with less number of features, where ITLBO-IPJAYA with 19 features performs better than TLBO and ITLBO-JAYA with 21 and 22 features, respectively. The improvements shown in Sections 4 and 6 reduce the execution time for ITLBO-IPJAYA over ITLBO-JAYA. The parallel processing of each SVM parameter independently is the main factor that reduces the execution time for ITLBO-IPJAYA over ITLBO-JAYA, as shown in Figure 14.

The results of the CICIDS 2017 dataset are shown in Table 12.

Finally, statistical significance tests (T-test), T-test made on the distribution of values in both samples, showed their significant difference, which allowed us to reject null hypothesis H0. The test sh2ows the superiority of IPJAYA-ITLBO-SVM over JAYA-ITLBO-SVM. The P values and T-

Table 11: Comparison of ITLBO, ITLBO-JAYA and ITLBO-IPJAYA for the NSL-KDD dataset.

No. of features	Method	MAX. Acc	AVR. Acc	DR	FAR	FNR	F-M	Recall	ER
16	TLBO	0.9639	0.9630	0.9612	0.0449	0.0282	0.9664	0.9717	0.036
	ITLBO	0.9680	0.9678	0.9671	0.0379	0.0268	0.9701	0.9731	0.032
	ITLBO-JAYA	0.9688	0.9685	0.9676	0.0373	0.0258	0.971	0.9741	0.0312
	ITLBO-IPJAYA	0.9708	0.9705	0.9712	0.0331	0.0256	0.9727	0.9742	0.0292
18	TLBO	0.9713	0.971	0.9739	0.0299	0.0275	0.9731	0.9724	0.0286
	ITLBO	0.9718	0.9713	0.9744	0.0292	0.0273	0.9736	0.9726	0.0282
	ITLBO-JAYA	0.9735	0.9733	0.9752	0.0285	0.0247	0.9752	0.9752	0.0265
	ITLBO-IPJAYA	0.9747	0.9746	0.9753	0.0280	0.0221	0.9764	0.9779	0.0252
19	TLBO	0.9738	0.9735	0.9727	0.0313	0.0225	0.9755	0.9774	0.0261
	ITLBO	0.9751	0.9745	0.9737	0.0305	0.0189	0.9769	0.9811	0.0248
	ITLBO-JAYA	0.9759	0.9758	0.9758	0.0278	0.0178	0.9775	0.9791	0.0241
	ITLBO-IPJAYA	0.9772	0.9770	0.9786	0.0245	0.0162	0.9787	0.9787	0.0228
21	TLBO	0.9782	0.9780	0.9742	0.0299	0.0145	0.9797	0.9844	0.0217
	ITLBO	0.9787	0.9784	0.9756	0.0279	0.0144	0.981	0.9846	0.0212
	ITLBO-JAYA	0.9793	0.979	0.9789	0.0273	0.0132	0.9811	0.9867	0.0207
	ITLBO-IPJAYA	0.9802	0.980	0.9792	0.0263	0.0123	0.9812	0.9716	0.0198
22	TLBO	0.9801	0.979	0.9755	0.0284	0.0131	0.9814	0.9868	0.0199
	ITLBO	0.981	0.9805	0.9758	0.0277	0.0117	0.9823	0.989	0.0191
	ITLBO-JAYA	0.9816	0.9814	0.9794	0.0265	0.0114	0.9829	0.989	0.0183
	ITLBO-IPJAYA	0-.9823	0.9821	0.9798	0.0262	0.0102	0.9835	0.9898	0.0177

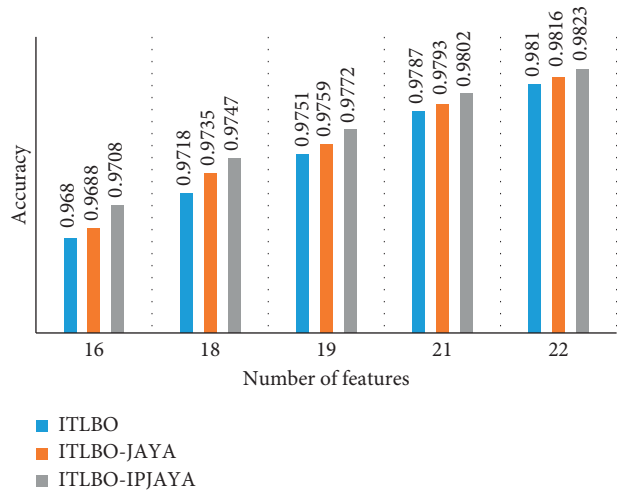

Figure 12: Accuracy based on the number of features for the NSL-KDD dataset.

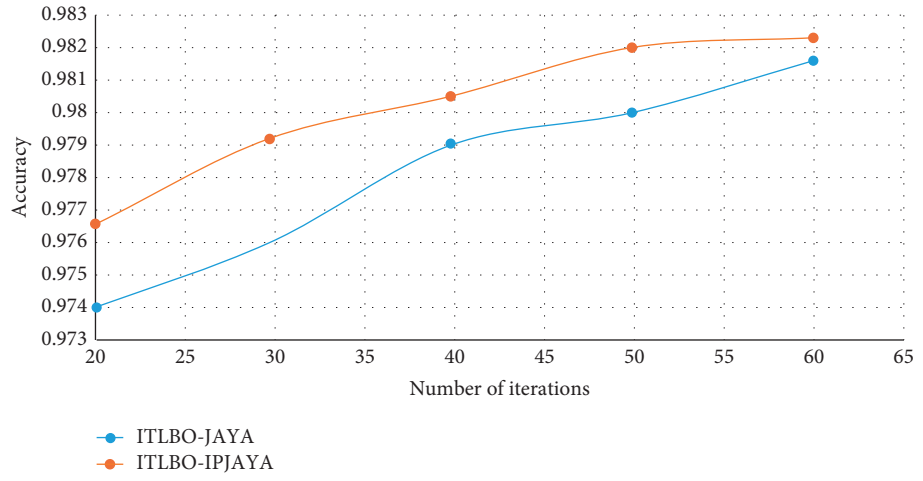

Figure 13: Accuracy comparison based on the number of iterations for the NSL-KDD dataset.

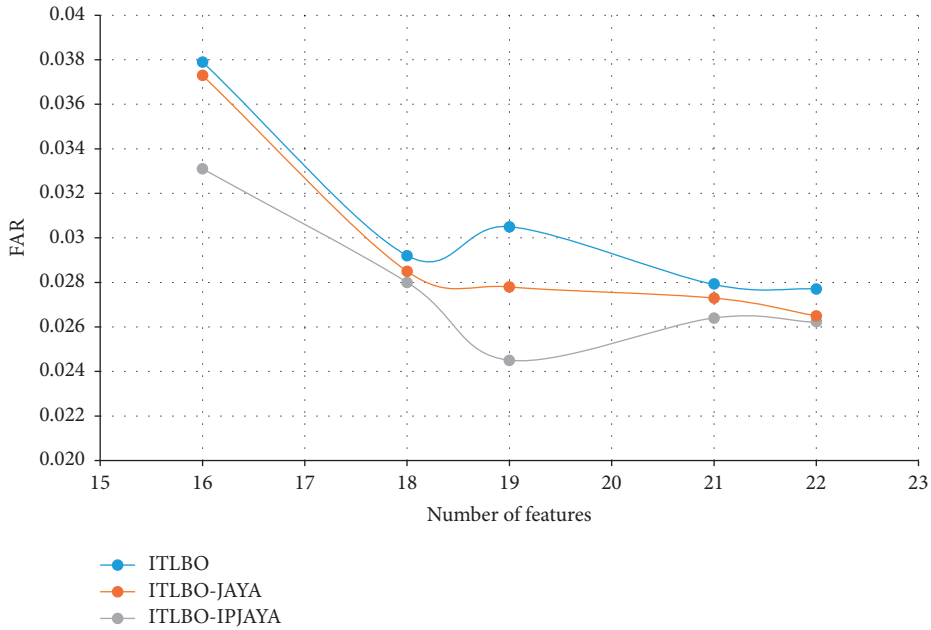

Figure 14: Comparison based on the number of features head for the NSL-KDD dataset.

Table 12: Comparison of ITLBO, ITLBO-JAYA, and ITLBO-IPJAYA for the CICIDS dataset.

No. of features	Method	MAX. Acc	AVR. Acc	DR	FAR	FNR	F-M	Recall	ER
12	ITLBO	0.9634	0.9631	0.9661	0.0389	0.0268	0.970	0.9721	0.0323
	ITLBO-JAYA	0.9685	0.9683	0.9682	0.0360	0.0267	0.9713	0.9722	0.0315
	ITLBO-IPJAYA	0.9704	0.9702	0.9701	0.0310	0.0265	0.9725	0.9724	0.0298
13	ITLBO	0.9712	0.9710	0.9724	0.0298	0.0273	0.9736	0.9726	0.0282
	ITLBO-JAYA	0.9745	0.9744	0.9728	0.0290	0.0264	0.9741	0.9794	0.0272
	ITLBO-IPJAYA	0.9768	0.9767	0.9732	0.0285	0.0260	0.9752	0.9787	0.0264
14	ITLBO	0.9776	0.9775	0.9737	0.0280	0.0189	0.9769	0.9811	0.0258
	ITLBO-JAYA	0.9789	0.9787	0.9742	0.0270	0.0174	0.978	0.986	0.0235
	ITLBO-IPJAYA	0.9801	0.980	0.9749	0.0265	0.0134	0.981	0.987	0.0210
16	ITLBO	0.9804	0.9803	0.9755	0.0271	0.011	0.9821	0.989	0.0190
	ITLBO-JAYA	0.981	0.9808	0.9773	0.0266	0.0109	0.9825	0.989	0.0183
	ITLBO-IPJAYA	0.9817	0.9815	0.9782	0.0264	0.0105	0.9831	0.9896	0.0170

values are shown in Table 13; the small values show that the IPJAYA-ITLBO-SVM method (MV1) is highly significant.

13. The Comparison of the Proposed Methods

To illustrate the effectiveness of our proposed IDS methods, the performance of the proposed methods is compared with six recently developed anomaly detection techniques. Table 14 demonstrates the result achieved by the proposed methods compared with other methods tested on the NSL-KDD dataset in terms of detection rate and false alarm rate. It is very clear that our proposed methods (ITLBO-JAYA and ITLBO-IPJAYA) obtained the best results with 0.9823 accuracy, 0.9798 detection rate, and 0.0102 false alarm rate for the ITLBO-IPJAYA model and 0.9816 accuracy, 0.9794 detection rate, and 0.0114 false alarm rate for the ITLBO-JAYA method, as shown in Table 11. However, Table 15 demonstrates the result achieved by the proposed methods compared with other methods tested on the CICIDS 2017 dataset in terms of detection rate and false alarm rate.

14. Discussion

This work in general contains 4 sections based on the proposed method. Furthermore, all methods proposed in this work were evaluated based on the NSL-KDD and CICIDS 2017 datasets.

Firstly, the proposed ITLBO-IPJAYA based on network intrusion detection and method results were compared with TLBO, ITLBO, and ITLBO-JAYA as shown in Tables 11 and 12. Additionally, the table shows the different features for the three algorithms to investigate the influence of the feature's increase on the performance, which represents a different algorithm structure. The ITLBO-IPJAYA results showed higher stability and better accuracy than ITLBO and ITLBO-JAYA algorithms.

Furthermore, Figure 13 shows that ITLBO-JAYA needs 60 iterations to reach accuracy of 0.9816 when the ITLBO-IPJAYA algorithm with 50 iterations achieved higher accuracy. Therefore, ITLBO-IPJAYA achieved better detection rate and less false alarm rate with less complexity of

Table 13: *T*-test results.

	NSL-KDD	CICIDS 2017
P value	0.0156	0.0068
T value	3.174	4.044

Table 14: Comparison with the existing work for the NSL-KDD dataset.

Ref.	Method	Dataset	Acc.	DR	FAR
[35]	Two-stage classifier	NSL-KDD	96.38	N.G	N.G
[36]	Hypergraph-based genetic algorithm and SVM	NSL-KDD	0.975	0.9714	0.83
[8]	PSO and SVM	NSL-KDD	0.9784	0.9723	0.87
[37]	Chi-square and SVM	NSL-KDD	0.98	N.G	0.13
[38]	SVM and hybrid PSO	NSL-KDD	0.7341	0.6628	2.81
[39]	SVM and feature selection	NSL-KDD	0.90	N.G	N.G
[40]	SVM and GA	NSL-KDD	0.975	N.G	N.G
TLBO-SVM	TLBO and SVM	NSL-KDD	0.9801	0.9755	0.0284
ITLBO-SVM	Improved TLBO and SVM	NSL-KDD	0.981	0.9758	0.0277
ITLBO-JAYA-SVM	Improved TLBO, improved JAYA and SVM	NSL-KDD	0.9816	0.9794	0.0265
ITLBO-IPJAYA-SVM	Improved TLBO, improved JAYA and SVM	NSL-KDD	0.9823	0.9798	0.0262

Table 15: Comparison with the existing work for the CICIDS 2017 dataset.

Ref.	Method	Dataset	Acc.	DR	FAR
[41]	Hybrid model	CICIDS	89.76	N.G	N.G
[42]	Wrapper-based feature selection	CICIDS	97.68	N.G	N.G
[43]	Feature selection technique and SVM	CICIDS	0.9803	N.G	N.G
TLBO-SVM	TLBO and SVM	CICIDS	0.9794	0.9745	0.0274
ITLBO-SVM	Improved TLBO and SVM	CICIDS	0.9804	0.9755	0.0271
ITLBO-JAYA-SVM	Improved TLBO, improved JAYA and SVM	CICIDS	0.981	0.9773	0.0266
ITLBO-IPJAYA-SVM	Improved TLBO, improved JAYA and SVM	CICIDS	0.9817	0.9782	0.0264

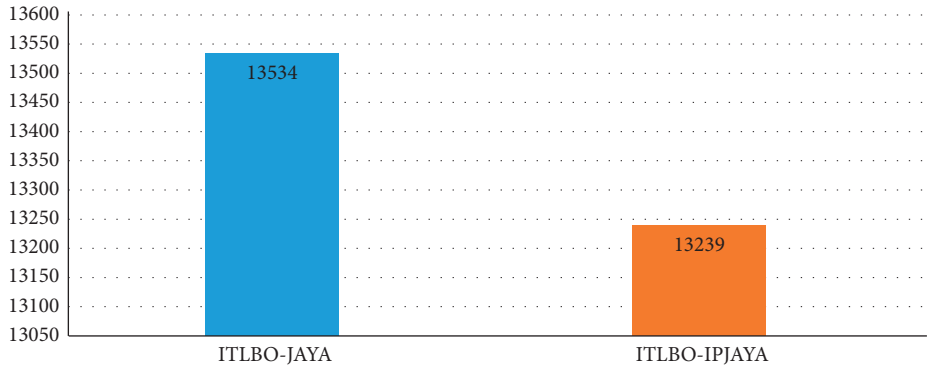

Figure 15: Execution time comparison for the NSL-KDD dataset.

iterations. Secondly, with all the improvement of ITLBO-SVM mentioned above, random selection of the main SVM parameters is considered as one of the algorithm limitations, which may not provide optimal parameter value and affect the model accuracy negatively.

The results above showed that the ITLBO-IPJAYA performance improved the basic SVM performance by providing the best parameter values as shown in the ITLBO-IPJAYA block diagram in Figure 11. In the end, the performance of ITLBO-IPJAYA is worth reducing the impact of selected parameters randomly.

As a result of the differences in the algorithm structure, the ITLBO structure contains three phases which should prevent the algorithm from being trapped in local and global optima. Also, teachers not only teach learners (students) but also teach other teachers. On the contrary, the TLBO structure contains two phases only, where teachers teach learners only.

Furthermore, the ITLBO algorithm achieved higher accuracy than TLBO because the knowledge exchange rate is higher in ITLBO since teachers teach learners and other teachers. Therefore, ITLBO achieved better detection rate and less false alarm rate with less complexity of iterations.

Dividing the solutions of the IPJAYA algorithm into two groups and choosing the best solution from the best solution group as "Best" and the best solution from the worst solution group as "Worst" cause IPJAYA to need less iterations than JAYA to reach better solutions, as shown in Figure 13. This also leads to improvement in accuracy and detection rate.

The parallel improvement done on the JAYA algorithm reduces the time needed for execution and hence reduces the total execution time for the ITLBO-IPJAYA-SVM model as shown in Figure 15.

Data Availability

The data used to support the findings of this study are available online at https://www.unb.ca/cic/datasets/nsl.html.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

Special appreciation is due to Universiti Malaysia Pahang for the sponsorship of this study approved by the Ministry of Higher Education (MOHE) for Fundamental Research Grant Scheme (FRGS) with Vot No. RDU190113.

References

[1] A. Sultana and M. Jabbar, "Intelligent network intrusion detection system using data mining techniques," in *Proceedings of the 2016 2nd International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT)*, pp. 329–333, IEEE, Bangalore, India, July 2016.

[2] R. Rao, "Jaya: a simple and new optimization algorithm for solving constrained and unconstrained opti- mization problems," *International Journal of Industrial Engineering Computations*, vol. 7, no. 1, pp. 19–34, 2016.

[3] M. Alsajri, M. A. Ismail, and S. Abdul-Baqi, "A review on the recent application of Jaya optimization algorithm," in *Proceedings of the 2018 1st Annual International Conference on Information and Sciences (AiCIS)*, Springer, Al-Fallujah, Iraq, pp. 129–132, November 2018.

[4] P. Tao, Z. Sun, and Z. Sun, "An improved intrusion detection algorithm based on GA and SVM," *IEEE Access*, vol. 6, pp. 13624–13631, 2018.

[5] A. S. Eesa, Z. Orman, and A. M. A. Brifcani, "A novel feature-selection ap- proach based on the cuttlefish optimization algorithm for intrusion detection systems," *Expert Systems with Applications*, vol. 42, no. 5, pp. 2670–2679, 2015.

[6] P. Louvieris, N. Clewley, and X. Liu, "Effects-based feature identification for network intrusion detection," *Neurocomputing*, vol. 121, pp. 265–273, 2013.

[7] E. De la Hoz, A. Ortiz, J. Ortega, and B. Prieto, "PCA filtering and probabilistic SOM for network intrusion detection," *Neurocomputing*, vol. 164, pp. 71–81, 2015.

[8] S. Mojtaba, H. Bamakan, H. Wang, Y. Tian, and Y. Shi, "An effective intrusion detection framework based on MCLP/SVM optimized by time-varying chaos particle swarm optimization," *Neurocomputing*, vol. 199, pp. 90–102, 2016.

[9] R. V. Rao, V. J. Savsani, and D. P. Vakharia, "Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems"" *Computer-Aided Design*, vol. 43, no. 3, pp. 303–315, 2011.

[10] S. P. Das and S. Padhy, "A novel hybrid model using teaching-learning-based optimization and a support vector machine for commodity futures index forecasting," *International Journal of Machine Learning and Cybernetics*, vol. 9, no. 1, pp. 97–111, 2018.

[11] S. P. Das, N. S. Achary, and S. Padhy, "Novel hybrid SVM-TLBO forecasting model incorporating dimensionality reduction techniques," *Applied Intelligence*, vol. 45, no. 4, pp. 1148–1165, 2016.

[12] R. V. Rao, V. J. Savsani, and J. Balic, "Teaching–learning-based optimization algorithm for unconstrained and constrained real-parameter optimization problems"" *Engineering Optimization*, vol. 44, no. 12, pp. 1447–1462, 2012.

[13] R. V. Rao and V. Patel, "An improved teaching-learning-based optimization algorithm for solving uncon- strained optimization problems," *Scientia Iranica*, vol. 20, no. 3, pp. 710–720, 2013.

[14] M. Črepinšek, S.-H. Liu, and L. Mernik, "A note on teaching–learning-based optimization algo- rithm," *Information Sciences*, vol. 212, pp. 79–93, 2012.

[15] M. R. Nayak, C. K. Nayak, and P. K. Rout, "Application of multi-objective teaching learning based optimization algorithm to optimal power flow problem," *Procedia Technology*, vol. 6, pp. 255–264, 2012.

[16] Y. Xu, L. Wang, S.-Y. Wang, and M. Liu, "An effective teaching-learning-based optimization algorithm for the flexible job-shop scheduling problem with fuzzy processing time," *Neurocomputing*, vol. 148, pp. 260–268, 2015.

[17] H. E. Kiziloz, A. Deniz, T. Dokeroglu, and A. Cosar, "Novel multiobjective TLBO algorithms for the feature subset selection problem," *Neurocomputing*, vol. 306, pp. 94–107, 2018.

[18] S. H Wang, K. Muhammad, Y. Lv et al., "Identification of Alcoholism based on wavelet Renyi entropy and three-segment encoded Jaya algorithm," *Complexity*, vol. 2018, Article ID 3198184, 13 pages, 2018.

[19] H. Migallón, A. Jimeno-Morenilla, and J.-L. Sanchez-Romero, "Parallel improvements of the Jaya optimization algorithm," *Applied Sciences*, vol. 8, no. 5, p. 819, 2018.

[20] C. Gong, "An enhanced Jaya algorithm with a two group Adaption," *International Journal of Computational Intelligence Systems*, vol. 10, no. 1, pp. 1102–1115, 2017.

[21] O. Samuel, N. Javaid, S. Aslam, and M. H. Rahim, "JAYA optimization based energy management controller for smart grid: JAYA optimization based energy management controller," in *Proceedings of the 2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, March 2018.

[22] K. Yu, J. J. Liang, B. Y. Qu, X. Chen, and H. Wang, "Parameters identification of photovoltaic models using an improved JAYA optimization algorithm," *Energy Conversion and Management*, vol. 150, pp. 742–753, 2017.

[23] M. Dash and H. Liu, "Feature selection for classification," *Intelligent Data Analysis*, vol. 1, no. 1-4, pp. 131–156, 1997.

[24] S. Dumais, J. Platt, D. Heckerman, and M. Sahami, "Inductive learning algorithms and representations for text categorization," in *Proceedings of the seventh international conference on Information and knowledge management-CIKM'98*, November 1998.

[25] R. Singh, H. Kumar, and R. K. Singla, "An intrusion detection system using network traffic profiling and online sequential extreme learning machine," *Expert Systems with Applications*, vol. 42, no. 22, pp. 8609–8624, 2015.

[26] M. Sokolova and L. Guy, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, vol. 45, no. 4, pp. 427–437, 2009.

[27] P. Phoungphol, Y. Zhang, and Y. Zhao, "Robust multiclass classification for learning from imbalanced biomedical data," *Tsinghua Science and Technology*, vol. 17, no. 6, pp. 619–628, 2012.

[28] A. Jahan, F. Mustapha, M. Y. Ismail, S. M. Sapuan, and M. Bahraminasab, "A comprehensive VIKOR method for material selection," *Materials & Design*, vol. 32, no. 3, pp. 1215–1221, 2011.

[29] L Aljarah and S. A. Ludwig, "Mapreduce intrusion detection system based on a particle swarm optimization clustering algorithm," in *Proceedings of the 2013 IEEE Congress on Evolutionary Computation*, pp. 955–962, IEEE, Cancun, Mexico, June 2013.

[30] K. Khaleel, M. A. Ismail, U. Yunan, and S. Kasim, "Review on intrusion detection system based on the goal of the detection system," *International Journal of Integrated Engineering*, vol. 10, no. 6, 2018.

[31] J. D. Rodriguez, A. Perez, and J. A. Lozano, "Sensitivity analysis of k-Fold cross validation in prediction error estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 3, pp. 569–575, 2010.

[32] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications*, pp. 1–6, IEEE, Ottawa, ON, Canada, July 2009.

[33] M. Al-Qatf, Y. Lasheng, M. Al-Habib, and K. Al-Sabahi, "Deep learning approach combining sparse autoencoder with SVM for network intrusion detection," *IEEE Access*, vol. 6, pp. 52843–52856, 2018.

[34] 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. 41525–41550, 2019.

[35] B.A. Tama, M. Comuzzi, and K.-H. Rhee, "TSE-IDS: a two-stage classifier ensemble for intelligent anomaly-based intrusion detection system," *IEEE Access*, vol. 7, pp. 94497–94507, 2019.

[36] M. R. Gauthama Raman, N. Somu, K. Kannan, R. Liscano, and V. S. Shankar Sriram, "An efficient intrusion detection system based on hypergraph - genetic algorithm for parameter optimization and feature selection in support vector machine"" *Knowledge-Based Systems*, vol. 134, pp. 1–12, 2017.

[37] I. S. Thaseen and C. Aswani Kumar, "Intrusion detection model using fusion of chi- square feature selection and multi class SVM," *Journal of King Saud University - Computer and Information Sciences*, vol. 29, no. 4, pp. 462–472, 2017.

[38] Y. Li, S. Yu, J. Bai, and X. Cheng, "Towards effective network intrusion detection: a hybrid model integrating Gini index and GBDT with PSO," *Journal of Sensors*, vol. 2018, Article ID 1578314, 9 pages, 2018.

[39] A. A. Aburomman and M. B. I. Reaz, "A novel weighted support vector machines multi- class classifier based on differential evolution for intrusion detection systems," *Information Sciences*, vol. 414, pp. 225–246, 2017.

[40] J. Esmaily and J. Ghasemi, "A novel intrusion detection systems based on genetic algorithms-suggested features by the means of different permutations of labels' orders," *International Journal of Engineering*, vol. 30, no. 10, pp. 1494–1502, 2017.

[41] S. Aljawarneh, M. Aldwairi, and M. B. Yassein, "Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model," *Journal of Computational Science*, vol. 25, pp. 152–160, 2018.

[42] 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.

[43] S. U. Jan, S. Ahmed, V. Shakhov, and I. Koo, "Toward a lightweight intrusion detection system for the internet of things," *IEEE Access*, vol. 7, pp. 42450–42471, 2019.