AI-Based Two-Stage Intrusion Detection for Software Defined IoT Networks

Jiaqi Li, Zhifeng Zhao, Rongpeng Li, and Honggang Zhang

***Abstract*—Software defined Internet of Things (SD-IoT) networks profit from centralized management and interactive resource sharing, which enhances the efficiency and scalability of Internet of Things applications. But with the rapid growth in services and applications, they are vulnerable to possible attacks and face severe security challenges. Intrusion detection has been widely used to ensure network security, but classical detection methods are usually signature-based or explicit-behavior-based and fail to detect unknown attacks intelligently, which makes it hard to satisfy the requirements of SD-IoT networks. In this paper, we propose an artificial intelligence-based two-stage intrusion detection empowered by software defined technology. It flexibly captures network flows with a global view and detects attacks intelligently. We first leverage Bat algorithm with swarm division and binary differential mutation to select typical features. Then, we exploit Random Forest through adaptively altering the weights of samples using the weighted voting mechanism to classify flows. Evaluation results prove that the modified intelligent algorithms select more important features and achieve superior performance in flow classification. It is also verified that our solution shows better accuracy with lower overhead compared with existing solutions.**

***Index Terms*—Artificial intelligence (AI), intrusion detection, network security, software defined Internet of Things (SD-IoT).**

I. Introduction

INTERNET of Things (IoT) is an evolving technology which provides ubiquitous connectivity and interactions between the physical and cyber worlds [1]. However, the rapid increase in the number and diversity of smart devices connected to the Internet has raised the issues of flexibility, efficiency, and availability within the current IoT networks. As an essential trend of IoT networks, the emergence of software defined IoT (SD-IoT) networks [2] provide a manageable solution, which has drawn significant attention. Benefiting from the advantages of software defined network (SDN) [3], SDN-based approaches facilitate the supervision of network status and the collection of information under centralized control in an active manner. Moreover, they also optimize network management and resource allocation flexibly owing to software programmability in order to meet the diverse demands of IoT networks. Nevertheless, the development of SD-IoT does not totally eliminate various security issues and challenges. In order to address these problems, it is urgent to build up effective and intelligent algorithms for enforcing the security of SD-IoT networks [4].

As an indispensable technology in network security, intrusion detection dynamically monitor abnormal behaviors in a system and indicate whether some events are susceptible to an attack [5]. There are two main categories of intrusion detection techniques: 1) misuse detection and 2) anomaly detection. Misuse detection is usually signature-based, which can only detect known attacks by matching the behaviors of incoming intrusions with the historical knowledge and predefined rules. Anomaly detection automatically constructs a normal behavior of the systems and stubbornly detects incoming intrusions by explicitly computing deviations. It can recognize novel attacks but may raise false alarms as well.

To overcome the limitations of traditional intrusion detection, artificial intelligence (AI) has been taken into account for intelligent detection. AI-based schemes can automatically discover deep knowledge or patterns from the historical data and make wise judgments to predict network intrusions [6], [7]. Though, there have been a few researches on combining IDS and AI, they are still incapable of universally precise detection and robustly considering the evolution and development of SD-IoT networks.

In this paper, we propose an advanced intrusion detection technology using AI algorithms based on SD-IoT architecture. More specifically, we apply a combination of enhanced AI algorithms to perform feature selection and flow classification, which are two crucial steps in intrusion detection. In particular for feature selection, we take advantage of improved bat algorithm (BA) by splitting the whole swarm into subgroups using K-means method so that each subgroup can learn within and among different populations more efficiently. Besides, binary differential mutation is also employed to increase the diversity of individuals. For flow classification, we optimize random forest (RF) through updating the weight of each sample after building each tree iteratively and making the final decision by using a weighted voting mechanism.

The remainder of this paper is organized as follows. Section II discusses the related research works. Section III introduces the AI-based two-stage intrusion detection empowered by software defined technology. Section IV describes the proposed BA algorithm for feature selection. Section V

Manuscript received April 30, 2018; revised July 27, 2018 and September 29, 2018; accepted November 18, 2018. Date of publication November 26, 2018; date of current version May 8, 2019. This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFB0803702, in part by the National Natural Science Foundation of China under Grant 61701439 and Grant 61731002, and in part by the Zhejiang Key Research and Development Plan under Grant 2018C03056. *(Corresponding author: Zhifeng Zhao.)*

The authors are with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou 310027, China (e-mail: 21631097@zju.edu.cn; zhaozf@zju.edu.cn; lirongpeng@zju.edu.cn; honggangzhang@zju.edu.cn).

Digital Object Identifier 10.1109/JIOT.2018.2883344

presents the improved RF algorithm for traffic classification. Section VI presents the performance evaluation results. Section VII summarizes this paper and points out the potential future work.

II. Related Works

As a critical enabling technology, SDN radically revolutionizes the way network operators will architect and coordinate, and meets the demands of IoT networks in terms of performance and reliability. It dynamically manages network configurations and provides flexible service-provisioning mechanisms in a centralized control manner [8]. Accordingly, the combination of IoT and SDN has attracted tremendous research interests. In [9], an architecture with SDN-based IoT framework coupled with network functionality virtualization is introduced. A general implementation is provided by virtualizing the IoT gateway, which makes it possible for the IoT network to be more dynamic, scalable, and elastic. Another IoT architecture originating from SDN to overcome big data problems is proposed in [10]. By evaluating the usefulness of the sensed values in the lower layers (especially in the gateway layer) instead of in the application layer, the number of packets being sent to the Internet is reduced, which overcomes the problems related to huge data volumes.

The advancement of SDN has strengthened IoT security through supporting the supervision of network status and the collection of flow statistics. It also provides network-layer security services, such as packet routing, identity authentication, and automated security management from a global view, which facilitates the detection and prevention of attacks [11]. However, SD-IoT networks still face severe security challenges as new attacks quickly appear. Therefore, several previous studies have investigated the ability of SDN and introduces various solutions to improve IoT security. In [12], it proposes an identity-based authentication scheme for IoT based on SDN. The specific identity formats used by different communication protocols are mapped to a shared identity and a trusted certificate authority is implemented on the SDN controller. Nobakht *et al.* [13] also proposed a host-based intrusion detection and mitigation framework for IoT, in which the network visibility and flexibility properties of SDN are exploited and modules of intrusion detection and mitigation are implemented at the SDN controller. Moreover, remote security management is provided by the third-party entities that offer "security as a service." A flow-based security approach for IoT devices is proposed using an SDN gateway in [14]. It aims to mitigate distributed denial-of-service attacks that violates service availability by monitoring traffic flows and anomalous behaviors. These proposed schemes are applicable for solving specific types of security problems and can only perform well under certain conditions or in specific cases.

As the focus of network security, intrusion detection has gained intensive attentions and wide investigations in recent years [15]. Most existing methods depend on predefined rules, which are still unable to recognize uprising new attacks intelligently. In terms of this issue, various machine learning algorithms with flow-based classification have been adopted for solving such problems [16]. Most often, there are two stages in this process: 1) feature selection and 2) flow classification. The former stage addresses high dimensional data efficiently and decreases the number of features from a noisy dataset, which improves the learning efficiency and prediction accuracy of flow classification. Recently, a variety of swarm intelligence (SI) algorithms, such as ant colony optimization (ACO) [17] and particle swarm optimization (PSO) [18], have been applied for selecting optimal features as well. The latter stage distinguishes network flows by marking whether they belong to specific types of attacks or benign traffic. Le *et al.* [19] proposed a network-based intrusion detection and prevention system which performs C4.5 algorithm to build the decision tree for classifying the traffic. Kim *et al.* [20] proposed a three-layer recurrent neural network which is capable of automatically finding the correlations between flow records and acting as a neural classifier for misuse detection. However, current algorithms are still inadequate for effectively selecting optimal features and detecting new types of attacks with a lower cost under different circumstances owing to their inherent limitations. Therefore, we need to optimize existing algorithms for the two critical stages to detect network intrusions efficiently and adaptively.

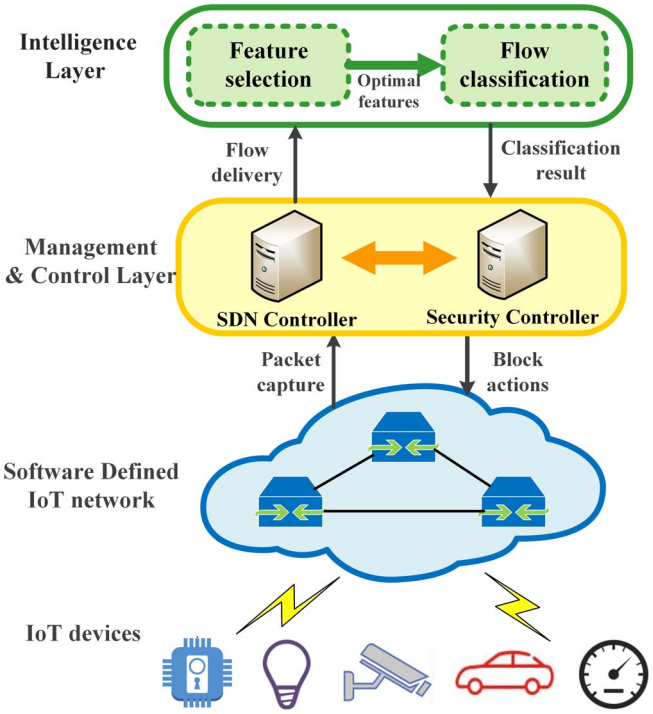

Fig. 1. AI-based two-stage intrusion detection for SD-IoT networks.

III. AI-Based Two Stage Intrusion Detection

In this paper, we propose a two-stage intrusion detection using AI algorithms under SD-IoT networks (see Fig. 1). Taking advantage of SDN, it captures network packets and collects status information with centralized control. The controllers partition network packets into flows and delivers them to the upper layer. In this way, flow-based intelligent intrusion detection can be implemented using AI algorithms in two

stages. It selects optimal flow features and detects network anomalies by classifying each flow into specific categories. Afterward, the controllers manage resource arrangement and organize specific actions for defending from attacks according to the classification results.

We apply two artificial intelligent algorithms in the two main stages of the intrusion detection process. SI algorithms have been widely adopted for global optimization through heuristic searching iteratively. As a promising novel SI algorithm, BA solves feature selection problems with outstanding performance and behaves better than traditional SI algorithms with simpler structure, fewer parameters, and stronger robustness. Therefore, in the first stage, we enhance BA to improve its ability for searching optimal features. As an ensemble method, RF can prevent overfitting and makes final decisions using majority voting. It has been validated that it outperforms other algorithms in various situations in terms of prediction accuracy with a tolerable time complexity, which gains much more popularities in classification. So in the second stage, we optimize RF algorithm to classify the network traffic into different classes of attacks using the selected features as input.

IV. Improved Bat Algorithm for Feature Selection

BA [21] is a metaheuristic algorithm for achieving global optimization. It was inspired by the echolocation behavior of micro-bats with varying pulse rates of emission and loudness. Population of identical bat individuals fly randomly searching for their target and use echolocation to perceive the environment and identify their prey. The solution to this kind of problem is given by the position of the bats and the method optimizes the problem by improving a candidate solution. The quality of the solution is measured by its fitness function.

For the feature selection process, we apply Binary BA (BBA) in this paper. It is a modified version of BA, which describes the bat's motion in a d-dimensional binary space in this paper. In order to adapt BBA for feature selection, we define the bat's position as the selected subset of features. Each coordinate with a value of 0 implies the absence of a feature, while a value of 1 reflects its presence in the selected subset of attributes. The feature selection process consists in iterations of position moving and target searching. The fitness function will be the evaluation metrics of the classifier after it has been trained with the selected features.

Although BA has achieved great performance, it still suffers from getting trapped into local minima easily. Since each bat is only strongly influenced by the global minima, it cannot exchange information with its neighbors efficiently. Moreover, the algorithm lacks a mutation mechanism, which leads to scarce positions of the swarm. In order to address these problems, we enhance the algorithm in the following two ways.

A. Swarm Division

At each iteration during the process, the whole swarm (the total number of individuals is N) is divided into several subgroups (e.g., K) using K-means algorithm referring to the distances between them (see Fig. 2). Members in each group are adjacent to each other with similar fitnesses. There are two levels in the mechanism. In the first level, general individuals within each subgroup only learn from their present local minima and direct to it through altering their velocity, which is influenced by both the local minima and their historical best positions. In the second level, the local minima with higher fitnesses likewise aim toward the global optima so that global optimal information can be exchanged. In this way, bats hierarchically learn from the superiority and move slightly toward a better position iteratively without being dominated by some specific bats. All the bats will be regrouped after each iteration until the algorithm terminates.

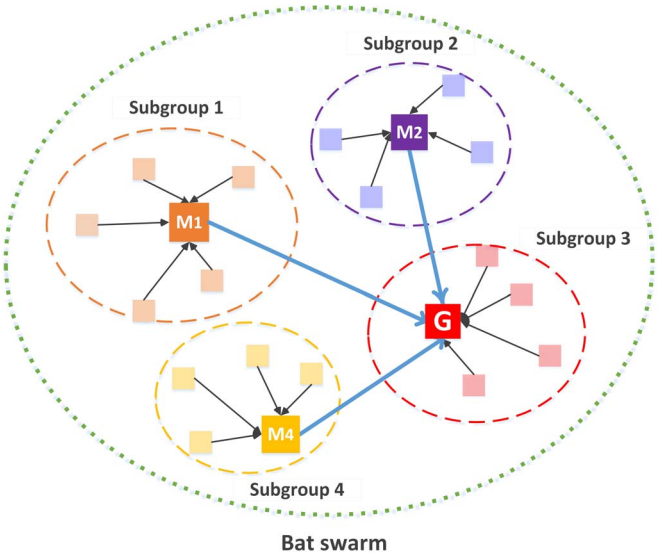

Fig. 2. Bat swarm division.

For each subgroup n ($n = 1, 2, \ldots, K$), we select the local minima, whose position is M_n. For all the subgroups, the global best bat is found located in G. Each bat retains its previous best position as P_i ($i = 1, 2, \ldots, N$). For the ith individual, its flight is described by its location in space $x_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,d})$ and velocity $v_i = (v_{i,1}, v_{i,2}, \ldots, v_{i,d})$, where d is the problem dimension. f_i is the frequency of the bat. At each iteration t, the individuals will update their internal variables f_i, v_i, and x_i as

$$f_i = f_{\min} + (f_{\max} - f_{\min}) \cdot \beta \tag{1}$$

where $\beta \in [0,1]$ is a random vector drawn from a uniform distribution. When applying the swarm division mechanism, the bat updates its velocity in two typical situations as follows.

For the nonlocal-minima individuals in each subgroup

$$v_{ij}^t = W^t \cdot v_{ij}^{t-1} + \left(x_{ij}^{t-1} - M_n^{t-1}\right) \cdot f_i + \left(x_{ij}^{t-1} - P_i^{t-1}\right) \cdot C^t. \tag{2}$$

For the local-minima individuals of each subgroup

$$v_{ij}^t = W^t \cdot v_{ij}^{t-1} + \left(x_{ij}^{t-1} - G^{t-1}\right) \cdot f_i + \left(x_{ij}^{t-1} - P_i^{t-1}\right) \cdot C^t \tag{3}$$

$$x_{ij}^t = \begin{cases} -1, & S(v_{ij}^t) > \delta \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

where $\delta \in [0, 1]$ is a random and S is a Sigmoid function. In each iteration, W^t is the inertia weight for each bat and C^t is the self-learning factor of each bat. $W_{\max}$, $W_{\min}$, $C_{\max}$, and $C_{\min}$ are the maximum and the minimum of W and C, respectively. N_T is the largest number of iterations. The two variables are calculated as

$$W^t = W_{\max} - \frac{(W_{\max} - W_{\min}) \cdot t}{N_t} \tag{5}$$

$$C^t = C_{\min} + (C_{\max} - C_{\min}) \cdot \left(1 - \frac{\arccos\left[\left(\frac{(-2)\cdot t}{N_t}\right) + 1\right]}{\pi}\right). \tag{6}$$

We introduce the linearly decreasing inertia weight W^t, which varies as the iterations progress to improve the optimization ability of the algorithm. In the early iterations, the bat with a higher inertia weight as well as a greater speed possess a strong global search ability. Later, a smaller inertia weight contributes to a more accurate local search, which accelerates the rate of convergence.

By employing the parameter C^t, we slightly enhance the velocity updating by taking advantage of the historical experience of the bats. The dynamically adjusting parameter indicates the influence of the previous best position of each bat on the current speed. The bat with larger C^t retains its own position with a better exploration ability initially. Afterward, the position of the bat tends to be greatly affected by the global best bat which elevates its exploitation ability.

Initially, each bat should have different values of loudness A_i and pulse emission rate r_i, which can be realized by randomization. As the iterations proceed, the bats have to be updated accordingly. When the bat comes closer to its target, it will decrease A_i and increase r_i

$$A_i^t = \alpha \cdot A_i^{t-1} \tag{7}$$

$$r_i^t = r_i^0 \cdot \left[1 - e^{-\gamma \cdot t}\right] \tag{8}$$

where $\alpha (0 < \alpha < 1)$ and $\gamma (> 0)$ are constants. In order to improve the variability of the discovered solutions, the bat performs random walk and generates a new solution through a local search

$$x_{\text{new}} = x_{\text{old}} + \delta \cdot A_t^* \tag{9}$$

where δ is a random number between -1 and 1. A_t^* is the average loudness of all bats at iteration t.

B. Binary Differential Mutation

After updating the location of bats at each iteration, we further apply the mutation mechanism of Differential Evolution [22] to BA algorithm, which enhances the diversity of the population and the ability of bats to jump out of local optima. It disturbs the target value by using the differences between randomly selected individuals in the swarm. Since the original method can only solve continuous optimization problems, we put forward a binary differential evolution algorithm using bit operations based on swarm division.

In the proposed new algorithm, the location and velocity of each bat in space are represented through binary strings. We leverage logical operations to implement the mutation process, where "+" represents the "XOR" operation, illustrated as "$\wedge$" and "$\oplus$" represents the "OR" operation, illustrated as "|." rand is a random number generated between 0 and 1. "$\cdot$" means that if the condition rand $< F^t$ is satisfied, the operations in the parentheses will be carried out. Suppose that vector $A = (a_1, a_2, \ldots, a_n)$, vector $B = (b_1, b_2, \ldots, b_n)$, and vector $C = (c_1, c_2, \ldots, c_n)$, then

$$A + B = (a_1 \wedge b_1, a_2 \wedge b_2, \ldots, a_n \wedge b_n) \tag{10}$$

$$A \oplus B = (a_1 | b_1, a_2 | b_2, \ldots, a_n | b_n) \tag{11}$$

$$C + (A \oplus B) \cdot F = \begin{cases} C \wedge (A|B), & \text{rand} < F^t \\ C, & \text{otherwise.} \end{cases} \tag{12}$$

At iteration t for the ith bat, if rand $< P^t$, we conduct the mutation on the velocity of the current bat as follows. Otherwise, we do nothing

$$v_{ij}^t = x_{r5}^{t-1} + \left(x_{r1}^{t-1} \oplus x_{r2}^{t-1}\right) \cdot F^t + \left(x_{r3}^{t-1} \oplus x_{r4}^{t-1}\right) \cdot F^t. \tag{13}$$

In the above equation, r_1, r_2, and r_5 are randomly selected bats in the same subgroup of the target, while r_3 and r_4 are bats selected from the distinctive subgroups. To implement the mechanism, we select a random individual within the same subgroup as the target bat and add some disturbance to it to generate a variation. This disturbance consists of the differences between both the individuals in the same group as well as that for individuals from different groups. The advance of the operation maintains the superiority of the original population without being destroyed by the unnecessary mutations. In addition, it also brings into the diversity of different subgroups so as to promote the evolution of the whole swarm.

P^t is the mutation probability that controls whether bats perform the mutation operations or not. It varies adaptively with the number of iterations to obtain better search abilities. N_T is the largest number of iterations. P^t is calculated as

$$P^t = \sqrt{\frac{t}{N_t - 1}}. \tag{14}$$

At early iterations, each bat can make full use of its ability to search within a larger space with a small mutation probability. As the number of iterations increases, it is more likely for bats to mutate, which breaks the constraints of local minima to avoid any premature convergence.

F^t is the shrinkage factor, which is randomly generated between 0 and 1. The shrinkage factor F regulates the variation of individuals by controlling the effect of the differential vectors. Larger values of F are conductive to maintaining the diversity of the population, whereas smaller values of F enables bats with better local search ability. $F_{\max}$ and $F_{\min}$ are the maximum and minimum of F, respectively, which is calculated as

$$F^t = F_{\min} + (F_{\max} - F_{\min}) \cdot \frac{N_t - t}{N_t}. \tag{15}$$

The modification of BA will ensure the algorithm with better exploration ability at early stage and higher exploitation at the later stage. The division of the swarm allows for efficient learning among similar individuals in the vicinity within

Input:
size of the swarm N
number of subgroups K
maximum number of iterations Max
fitness function $f(x_i)$
Output:
best solution of the swarm G^{Max}
Initialization:
generate initial population $(x_i, v_i, f_i, A_i, r_i)$;
compute fitness function $f(x_i)$
select global best solution G^0
Begin
For each iteration t, $t=1,...,Max$:
update controlling factors cf.eq.(5)(6)(14)(15) ;
swarm division using K-means ;
For each subgroup n, $n=1,...,K$:
select local minima bat at position Mn;
For each bat i, $i=1...N/K$ in the subgroup:
retain previous best position P_i ;
update fi cf.eq.(1), vi cf.eq.(2)(3), xi, *cf.eq.(4)* ;
local search for x_{new} cf.eq.(9) **if** $rand>r_i$;
update Ai, ri cf.eq.(7).(8) **if** rand$<Ai$ & $f(xi)<f(x_{new})$;
apply Binary Differential Evolution
if $rand<P'$ cf.eq.(13) ;
determine best fitness $f(x_i)$ and update current G';
End

Fig. 3. Main steps of improved BA.

each subgroup meanwhile sharing optimal information among the subgroups through the local minima. In this way, each bat can step toward the global best gradually without being trapped into the local minima in avoid of its drastic influence. By applying the binary differential mutation, the proposed approach also increases the diversity of the swarm and prevents it from getting trapped into local minima, which also accelerates the rate of convergence. The main steps of the new algorithm are given in Fig. 3.

V. Improved Random Forest for Flow Classification

As an ensemble machine learning algorithm, RF [23] has been widely used for processing high dimension dataset collaterally and preventing over-fitting to some extent. However, it degrades its performance in the minority classes of data when dealing with an imbalanced dataset. Since the samples are randomly selected with replacement when building trees, the minority classes with fewer samples are less likely to be selected and learned. In addition, we find that the cost of misclassifying minority classes is even higher than that for majority classes, which means that it is a crucial issue for improving detection rate. Therefore, we need to optimize the algorithm as in the following ways.

A. Weight Initialization

Usually, the weight of each sample is initialized similarly as $(1/N)$ (N is the total number of the samples in the dataset) and the sum of weights is 1. In this way, each sample has equal probability of being selected. In our mechanism, we initialize each training sample with a different weight according to the class that it belongs to. Corresponding to the original distribution of each class in our dataset, we initialize the weights of five classes as 0.3, 0.15, 0.35, 0.05, and 0.15. It reduces the weights of the majority class while boosting those of the minority class. Initially, the total number of samples in class j ($j = 0, 1, \ldots, 4$) is N_j, and the weight of each sample $w_{0,i}$ ($i = 1, 2, \ldots, N_j$) is calculated as

$$w_{0,i} = \frac{w_j}{N_j} \tag{16}$$

$$W_0 = \left(w_{0,1}, w_{0,2}, \ldots, w_{0,i}, \ldots, w_{0,N}\right) \tag{17}$$

where w_j is the weight of class j. $w_{0,i}$ represents the probability of each sample being selected from the N samples. In this way, the minority samples can be selected and paid more attention to instead of over-selecting the abundant samples in the majority class. We randomly select N samples using the roulette wheel selection scheme from the original dataset with replacement to train each tree.

B. Weight Updating

After building each tree, we update the weight of each sample according to the result it is classified. The weights of misclassified samples are increased while samples classified correctly are reduced. Consequently, we can stress more concern on the samples which are misclassified. Those samples with higher weights are more likely to be selected and learned in the next tree. We calculate the accuracy a_m using the error rate e_m of tree m ($m = 1, 2, \ldots, M$) under the whole dataset and update the weight of the sample $w_{m+1,i}$ as follows. We use e^a to boost the weights of misclassified samples and e^{-a} for those correctly classified. M is the predefined number of trees for training. Z_m is the scaling factor so that the total sum of weights remains 1

$$a_m = 0.5 \cdot \ln \frac{1 - e_m}{e_m} \tag{18}$$

$$w_{m+1,i} = \frac{w_{m,i}}{Z_m} \cdot \beta \cdot e^{\pm a} \tag{19}$$

$$Z_m = \sum_{i=1}^{N} w_{m,i} \cdot \beta \cdot e^{\pm a}. \tag{20}$$

We use a cost-sensitive way of learning by altering the weight of each sample. For the purpose of training, the misclassified samples in distinctive classes, specifically, we update the weight of each sample to a high extent by controlling the β factor in (19) considering the four situations are presented in Table I, where m and n are the sum of weights in the majority and minority classes, respectively.

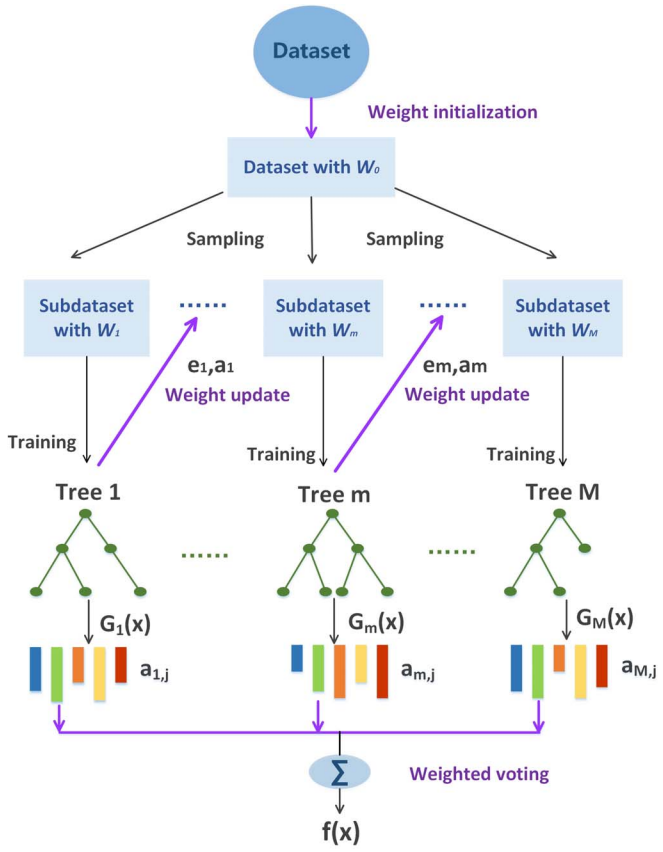

Fig. 4. Improved RF.

Input:
numbers of training data N
numbers of trees M
samples selected for training tree m N_m
the number of samples of N_m N_s
Output:
ensemble classification result $f(x)$
Initialization:
initialize weights of different classes w_j
compute weight of each sample $w_{0,i}$ in each class
cf.eq.(16)(17)
Begin
For each tree m, $m=1,...,M$:
building tree m using sampling data N_m;
calculate e_m and a_m cf.eq.(18) ;
For each selected sample i, $i=1,...,N_s$:
update the weight of each sample $w_{m+1,i}$
cf.eq.(19)(20) and Table I ;
determine the ensemble result $f(x)$ cf.eq.(21) and Table II ;
End

Fig. 5. Main steps of improved RF.

TABLE I
VALUES OF β CALCULATED FOR DIFFERENT CLASSES

Classes	Samples correctly classified	Samples mis-classified
Majority classes	2^{m-n}	2^{n-m}
Minority classes	2^{n-m}	2^{m-n}

TABLE II
ACCURACY MATRIX

Classes	Tree 1	Tree 2	...	Tree m	...	Tree M
Class0	$a_{1,0}$	$a_{2,0}$	...	$a_{m,0}$	...	$a_{M,0}$
Class1	$a_{1,1}$	$a_{2,1}$	...	$a_{m,1}$	...	$a_{M,1}$
Class2	$a_{1,2}$	$a_{2,2}$	...	$a_{m,2}$	...	$a_{M,0}$
Class3	$a_{1,3}$	$a_{2,3}$	...	$a_{m,3}$	...	$a_{M,3}$
Class4	$a_{1,4}$	$a_{2,4}$	...	$a_{m,4}$	...	$a_{M,4}$

From the table, we can see that, for the minority classes, we obviously increase the weights of misclassified samples, but slightly decrease those which are classified into the right class. For the majority classes, it is just the opposite. By using this method, the minority samples can be selected and trained consistently and iteratively without degrading their weights considerably when building trees. The weight update operation can also prevent samples from repetitively over-training.

C. Weighted Voting

Now that the classification ability of each tree varies among different classes, the traditional method using the majority votes of all the trees to obtain the final result cannot be used anymore. We introduce the weighted voting mechanism to the ensemble trees. More specifically, we compute the accuracy matrix of the classification accuracy $a_{m,j}$ of each tree m in each class j as shown in Table II.

The final result is computed according to (21). $f(x)$ is the ensemble result and $G_m(x)$ is the classification judgement of each tree m. Trees specialized in classifying different classes of samples can maximize their advantages when deciding the final result with higher weights

$$f(x) = \sum_{m=1}^{M} a_{m,j} \cdot G_m(x). \tag{21}$$

The whole process of the improved RF is illustrated in Fig. 4, including the three modifications which are highlighted in purple. The colorful histograms represent the accuracy of each tree for each class, which have different lengths. The figure shows the classification of a sample into class 2 as an example. The modification made to the RF helps to strike a balance between over-learning in the majority class and directing more emphasis on the minority class, which enhances its performance for imbalanced dataset. Furthermore, the cost-sensitive learning of misclassified samples also contributes to the overall accuracy. By using the weighted voting mechanism, various trees with distinguished abilities in classification can be strongly combined. The main steps of the algorithm are shown in Fig. 5.

TABLE III
DISTRIBUTION OF DATA USED IN OUR EVALUATION

Class	Training dataset		Testing dataset	
	Numbers	Percentage	Numbers	Percentage
Normal	17129	29.99%	12183	32.52%
Probe	3107	5.44%	1880	5.02%
DoS	35700	62.51%	21705	57.94%
U2R	52	0.09%	228	0.61%
R2L	1126	1.97%	1468	3.92%

TABLE IV
PERFORMANCE COMPARISON USING DIFFERENT ALGORITHMS

Algorithms	Accuracy (%)	FPR (%)
ACO	94.25	4.78
PSO	94.52	3.99
BA	94.93	3.68
Reference[26]	95.35	2.74
Reference[27]	95.64	2.06
Our proposed algorithm	96.03	1.18

VI. EVALUATION RESULT

In this section, we conduct several numerical experiments to evaluate the proposed intrusion detection mechanisms.

A. Dataset and Evaluation Metrics

As the KDD Cup 1999 dataset [24] has been widely used to evaluate various intrusion detection approaches, we perform a five-class flow classification using a subset of it after down-sampling in this paper. The distributions of both training and testing data marked by their attack type are summarized in Table III.

Generally, the performance of an intrusion detection is evaluated in the light of precision, recall, F-score, accuracy, and false alarm rate (FA). We desire a system with a higher detection rate and a lower false rate. Another comparative metric *Cost* is defined to measure the cost damage of misclassification for different attacks per sample. Relevant details of the dataset and evaluating metrics are introduced in our previous work [25]. Since the quality of the selected features contributes a lot to the final result of classification, we define the fitness function in the evaluations by giving each of the two previous performance metrics a certain weight as follows:

$$\text{fitness} = 0.6 \cdot R + 0.4 \cdot e^{-\text{FA}}. \tag{22}$$

B. Performance Analysis

We intend to evaluate the proposed mechanisms from three aspects. First, we assess the optimality and convergence of the proposed BA algorithm for feature selection. Second, we estimate the detection ability of the enhanced RF algorithm for flow classification on the overall dataset as well as in different classes of flows, respectively. Finally, we implement the combination of the above-mentioned algorithms in the proposed two-stage intrusion detection to make comparisons with existing solutions.

1) Evaluations on the Proposed Bat Algorithm: There have been several improvements made to the traditional BA to overcome its inherent shortcomings [26], [27]. We carry out a comparison with these methods as well as several typical SI algorithms for feature selection together with RF for classification. The results are shown in Table IV in terms of classification accuracy and false alarm rate. As noticed from the table, the proposed algorithm selects the features that contribute more to differentiate attack traffic, which gives rise to higher accuracy and lower false rate. It is proved that the proposed algorithm for optimal feature selection performs well, which is conductive to achieving better performance in classification.

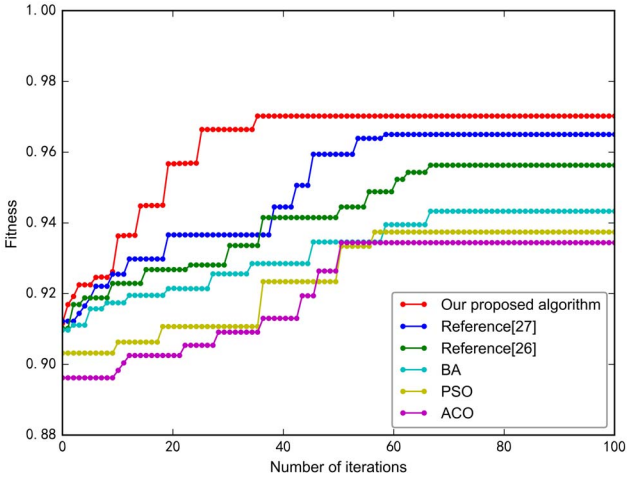

Fig. 6. Convergence comparison using different SI algorithms.

In Fig. 6, we verify the advantage of the proposed algorithm in finding a better subset of features with higher fitness within less iterations. It can be noticed that our algorithm converges faster than the others at approximately 40 iterations with a steeper slope, which lowers the time complexity. Furthermore, it obtains a higher fitness value after convergence and remains unchanged above the other curves. It validates that the improvement of swarm division and mutation mechanism prevent from getting trapped into local minima and search for a better solution. The linear time-varying parameters also strengthen individuals with a dynamic searching ability to adjust to the different phases of iterations.

Since the size of the swarm and the number of iterations are two critical parameters in the process of solving optimization problems, we evaluate their influence with various values in Fig. 7. It is illustrated that the accuracy of the algorithm enhances as the number of iterations increases with constant number of individuals in the swarm. It can be deduced to some extent that the more iterations the swarm goes through, the better they evolve in finding the optimal solutions. Moreover, the algorithm converges after a restricted number of iterations after finding the approximate optimal solution. At the same iteration, we can see that a larger swarm with more individuals performs better than those of smaller ones. It is because that a larger swarm with better population diversity can communicate more with each other interactively without gathering

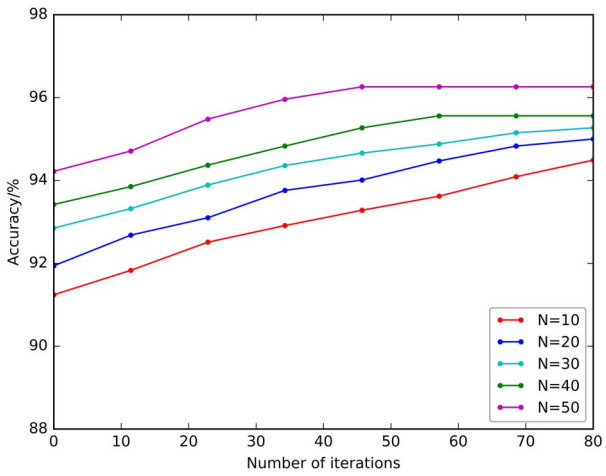

Fig. 7. Performance comparison using different numbers of individuals with various numbers of iterations.

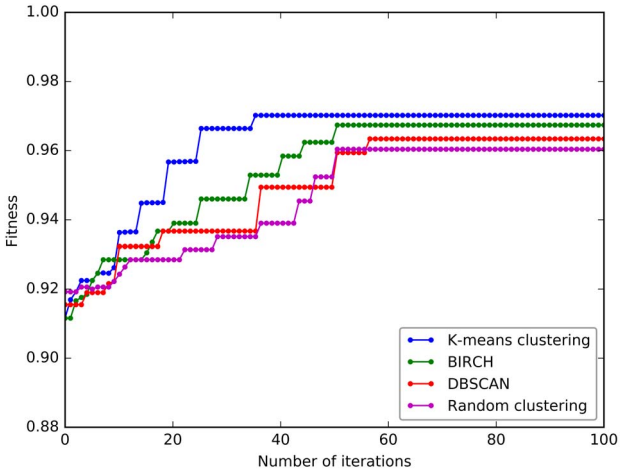

Fig. 8. Performance comparison using different methods of swarm division.

close to the local minima, which results in a greater searching ability over a larger area.

One of the most significant improvements of the proposed BA algorithm is the division of swarm into different populations. There are different kinds of methods to cluster individuals into populations. Some just randomly assign individuals into different clusters, whereas the others use clustering algorithms in the light of various metrics. Therefore, we examine the influence of different clustering algorithms on the final BA algorithm and present the convergence results in Fig. 8. It is apparent that *K*-means [28] clustering algorithm achieves better performance with a higher fitness and faster rate of convergence, since the adjacent individuals are clustered into same subgroup based on distances. On the one hand, the whole swarm moves toward the current best position by learning within each subgroup as well as by sharing knowledge between populations. On the other hand, each individual only learns partially from the local best in its subgroup and moves slightly in case of being badly affected by the local minima.

TABLE V
PERFORMANCE COMPARISON USING DIFFERENT ALGORITHMS

Algorithms	Precision (%)	Recall (%)	F_score (%)	FPR (%)	Cost
Decision Tree	96.59	92.84	95.42	4.79	0.2271
Adaboost	97.42	93.21	95.68	3.98	0.2032
RF	98.09	93.84	95.92	3.78	0.1738
SVM	98.74	94.36	96.55	2.75	0.1688
GBDT	99.17	94.84	96.72	1.68	0.1608
Proposed algorithm	99.51	95.17	97.29	0.98	0.1302

TABLE VI
DETECTION PERFORMANCE COMPARISON IN DIFFERENT CLASSES

Algorithms	Normal	Probe	DoS	U2R	R2L
Decision Tree	95.63	95.47	99.02	6.25	8.27
Adaboost	96.17	96.62	99.85	20.34	19.68
RF	96.28	95.85	100	12.19	14.51
SVM	97.72	97.55	99.78	12.49	12.72
GBDT	98.54	98.79	100	27.65	21.45
Proposed algorithm	99.02	99.63	100	57.46	23.84

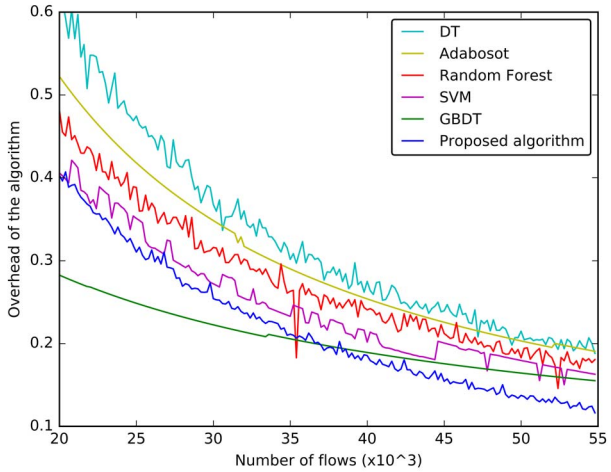

Fig. 9. Overhead produced by algorithms with different numbers of flows.

2) Evaluations on the Proposed Random Forest Algorithm: In Table V, we verify the advantages of the improved RF algorithm by comparing the performance of detection in contrast with that of the ordinary RF algorithm and other machine learning algorithms. All the classification algorithms use features selected by the proposed BA as input. By altering the weights of samples, each tree in the forest can be trained more effectively through picking the samples which are more frequently to be misclassified. By applying the weighted voting mechanism, the weight of each tree in a specific class is directly affected by its performance, which contributes distinctively to determining the final result. As we can see, it is obvious that the proposed method generates a better performance in every metric.

We observe the detection performance of the improved algorithm in five classes individually comparing with the above-mentioned algorithms in Table VI. In real network scenarios, some intrusions generate more connections than others,

TABLE VII
PERFORMANCE OF DIFFERENT COMBINATIONS OF ALGORITHMS

Combination of algorithms		No. of features	Accuracy (%)	FPR (%)
Proposed BA	Proposed RF	32	96.42	0.98
RF	GBDT	22	95.49	1.84
Tree	SVM	10	94.41	2.64
Fisher	RF	10	94.97	2.35
ReliefF	Adaboost	8	95.32	1.92
IG	Decision Tree	8	94.07	4.35
CFS	LR	18	93.22	6.75

TABLE VIII
PERFORMANCE COMPARISON OF DIFFERENT SYSTEMS

Proposed system (and related works)	Accuracy (%)	FPR (%)
Our system	96.42	0.98
Reference [25]	95.21	1.57
Reference [16]	94.56	1.83
Reference [29]	93.36	2.07
Reference [30]	92.42	2.82
Reference [19]	90.27	3.45

which leads to an extremely unbalanced dataset for classification. The detection rate varies remarkably in different classes. Usually, those intrusions with fewer flows generate higher costs when misclassified, so it is urgent to improve their performance for such cases. Thus, we solve the problem through altering the weights of samples accordingly to stress more attentions on those in the minority classes while avoiding over-fitting for the majority ones. It can be observed in Table VI, it is apparent that the proposed RF algorithm improves the detection accuracy of minority intrusions dramatically while slightly increases the detection rate of the majority ones. The result indicates that the algorithm adaptively balances its training for samples in different classes and decides the final result according to its learning ability, which accelerates the performance in each class.

We measure the overhead that the proposed algorithm causes when classifying using the above-mentioned concept cost. We compare the proposed algorithm with RF and the other machine learning algorithms in Fig. 9. The metric indicates that the more misclassified flows there are when classifying, the more overhead the algorithm generates for misclassification. As the number of flows grows, all of them become more well-trained for classifying with fewer errors. It can be seen that our algorithm produces less overhead than the others.

3) Evaluations on the Performance of Combined Algorithms in Two Stages: We evaluate different combinations of our improved algorithms by comparing them with several groups of traditional feature selection and machine learning algorithms in Table VII. Since we know that the selection of algorithms for feature selection and traffic classification possesses a mutual influence on each other, we care more about the performance of the combination of them. As we can see, it is obvious that the proposed methods achieve a better performance among all the combination alternatives in every metric.

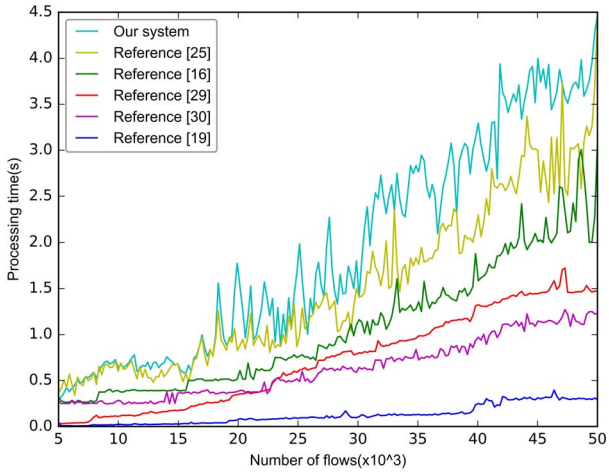

Fig. 10. Processing times of different systems.

There have been several intelligent architectures proposed to detect and prevent network intrusions under SDN environment [16], [19], [25], [29], [30]. We conduct a comparison with the previous results in terms of classification accuracy and error rate as described in Table VIII. The processing time of each approach using a portion of flows in the dataset is also illustrated in Fig. 10 to evaluate the efficiency. It can be noticed that the proposed two-stage intrusion detection improves the classification accuracy with a tolerable time consumption.

VII. CONCLUSION

This paper presents an AI-based two-stage intrusion detection implemented in SD-IoT networks. It leverages SDN contributing to status monitoring as well as traffic capturing under a global view. It integrates and coordinates two stages of IDS, including feature selection and flow classification, to detect novel intrusions with a self-learning ability. We improve BA to select optimal features and design network flow classification methods by enhancing RF algorithm. Evaluation results validate the optimality of our proposed algorithms in achieving higher accuracy and lower overhead. The experiments also reveal that the system improves its detection ability without much time consumption compared with the existing solutions.

In the future, we will implement this approach in a real network for traffic classification and evaluate the performance.

REFERENCES

[1] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A survey on enabling technologies, protocols, and applications," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 2347–2376, 4th Quart., 2015.

[2] M. S. Mahdavinejad *et al.*, "Machine learning for protect Internet of Things data analysis: A survey," *Digit. Commun. Netw.*, vol. 4, no. 3, pp. 161–175, 2017.

[3] H. Kim and N. Feamster, "Improving network management with software defined networking," *IEEE Commun. Mag.*, vol. 51, no. 2, pp. 114–119, Feb. 2013.

[4] M. B. Yassein, S. Aljawarneh, M. Al-Rousan, W. Mardini, and W. Al-Rashdan, "Combined software-defined network (SDN) and Internet of Things (IoT)," in *Proc. Int. Conf. Elect. Comput. Technol. Appl.*, Ras Al Khaimah, UAE, 2018, pp. 1–6.

[5] T. A. Tang, L. Mhamdi, D. Mclernon, S. A. R. Zaidi, and M. Ghogho, "Deep learning approach for network intrusion detection in software defined networking," in *Proc. Int. Conf. Wireless Netw. Mobile Commun.*, Fes, Morocco, 2018, pp. 258–263.

[6] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 2, pp. 1153–1176, 2nd Quart., 2016.

[7] R. Li *et al.*, "Intelligent 5G: When cellular networks meet artificial intelligence," *IEEE Wireless Commun.*, vol. 24, no. 5, pp. 175–183, Oct. 2017.

[8] Y. Zhang, M. Chen, and R. X. Lai, "Cloudified and software defined 5G networks: Architecture, solutions, and emerging applications," *Mobile Netw. Appl.*, vol. 21, no. 5, pp. 727–728, 2016.

[9] M. Ojo, D. Adami, and S. Giordano, "A SDN-IoT architecture with NFV implementation," in *Proc. GLOBECOM Workshops*, Washington, DC, USA, 2016, pp. 1–6.

[10] M. T. Kakiz, E. Özturk, and T. Çavdar, "A novel SDN-based IoT architecture for big data," in *Proc. Int. Art. Intell. Data Process. Symp.*, Malatya, Turkey, 2017, pp. 1–5.

[11] D. B. Rawat and S. R. Reddy, "Software defined networking architecture, security and energy efficiency: A survey," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 1, pp. 325–346, 1st Quart., 2017.

[12] O. Salman, S. Abdallah, I. H. Elhajj, A. Chehab, and A. Kayssi, "Identity-based authentication scheme for the Internet of Things," in *Proc. IEEE Symp. Comput. Commun.*, Messina, Italy, 2016, pp. 1109–1111.

[13] M. Nobakht, V. Sivaraman, and R. Boreli, "A host-based intrusion detection and mitigation framework for smart home IoT using OpenFlow," in *Proc. Int. Conf. Availability Rel. Security*, Salzburg, Austria, 2016, pp. 147–156.

[14] P. Bull, R. Austin, E. Popov, M. Sharma, and R. Watson, "Flow based security for IoT devices using an SDN gateway," in *Proc. IEEE Int. Conf. Future Internet Things Cloud*, Vienna, Austria, 2016, pp. 157–163.

[15] N. Farah *et al.*, "Application of machine learning approaches in intrusion detection system: A survey," *Int. J. Adv. Res. Artif. Intell.*, vol. 4, no. 3, pp. 9–18, 2015.

[16] A. S. D. Silva, J. A. Wickboldt, L. Z. Granville, and A. Schaeffer-Filho, "ATLANTIC: A framework for anomaly traffic detection, classification, and mitigation in SDN," in *Proc. IEEE/IFIP Netw. Oper. Manag. Symp.*, Istanbul, Turkey, 2016, pp. 27–35.

[17] T. Mehmood and H. B. M. Rais, "SVM for network anomaly detection using ACO feature subset," in *Proc. Int. Symp. Math. Sci. Comput. Res.*, Ipoh, Malaysia, 2015, pp. 121–126.

[18] N. Cleetus and K. A. Dhanya, "Multi-objective functions in particle swarm optimization for intrusion detection," in *Proc. Int. Conf. Adv. Comput. Commun. Informat.*, New Delhi, India, 2014, pp. 387–392.

[19] A. Le, P. Dinh, H. Le, and N. C. Tran, "Flexible network-based intrusion detection and prevention system on software-defined networks," in *Proc. Int. Conf. Adv. Comput. Appl.*, Ho Chi Minh City, Vietnam, 2015, pp. 106–111.

[20] J. Kim, J. Kim, H. L. T. Thu, and H. Kim, "Long short term memory recurrent neural network classifier for intrusion detection," in *Proc. Int. Conf. Platform Technol. Service*, 2016, pp. 1–5.

[21] A.-C. Enache and V. Sgârciu, "A feature selection approach implemented with the binary bat algorithm applied for intrusion detection," in *Proc. Int. Conf. Telecommun. Signal Process.*, 2015, pp. 11–15.

[22] J. Wang, J. Liao, Y. Zhou, and Y. Cai, "Differential evolution enhanced with multiobjective sorting-based mutation operators," *IEEE Trans. Cybern.*, vol. 44, no. 12, pp. 2792–2805, Dec. 2014.

[23] K. Singh, S. C. Guntuku, A. Thakur, and C. Hota, "Big data analytics framework for peer-to-peer botnet detection using random forest," *Inf. Sci.*, vol. 278, no. 19, pp. 488–497, 2014.

[24] H. Saxena and V. Richariya, "Intrusion detection in KDD99 dataset using SVM-PSO and feature reduction with information gain," *Int. J. Comput. Appl.*, vol. 98, no. 6, pp. 25–29, 2014.

[25] J. Li, Z. Zhao, and R. Li, "A machine learning based intrusion detection system for software defined 5G network," *IET Netw.*, vol. 7, no. 2, pp. 53–60, Mar. 2018.

[26] A.-C. Enache and V. Sgârciu, "Anomaly intrusions detection based on support vector machines with an improved bat algorithm," in *Proc. Int. Conf. Control Syst. Comput. Sci.*, Bucharest, Romania, 2015, pp. 317–321.

[27] A.-C. Enache and V. Sgârciu, "An improved bat algorithm driven by support vector machines for intrusion detection," in *Proc. Int. Joint Conf.*, 2015, pp. 41–51.

[28] T. Kanungo *et al.*, "An efficient *K*-means clustering algorithm: Analysis and implementation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 881–892, Jul. 2002.

[29] X. Ye *et al.*, "An anomalous behavior detection model in cloud computing," *Tsinghua Sci. Technol.*, vol. 21, no. 3, pp. 322–332, Jun. 2016.

[30] P. Wang, K.-M. Chao, H.-C. Lin, W.-H. Lin, and C.-C. Lo, "An efficient flow control approach for SDN-based network threat detection and migration using support vector machine," in *Proc. IEEE Int. Conf. E-Bus. Eng.*, vol. 24, 2016, pp. 56–63.

Jiaqi Li received the B.S. degree in electronic information engineering from Zhengzhou University, Zhengzhou, China, in 2016. She is currently pursuing the master's degree in information and communication engineering at Zhejiang University, Hangzhou, China.

Her current research interests include network intrusion detection, machine learning, and software defined networks.

Zhifeng Zhao received the bachelor's degree in computer science, the master's degree in communication and information system, and the Ph.D. degree in communication and information system from the PLA University of Science and Technology, Nanjing, China, in 1996, 1999, and 2002, respectively.

He is an Associate Professor with the Department of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, China, where he was a Post-Doctoral Researcher from 2002 to 2004, where his research was focused on multimedia next-generation networks and soft-switch technology for energy efficiency. From 2005 to 2006, he was a Senior Researcher with the PLA University of Science and Technology, where he performed research and development on advanced energy-efficient wireless router, ad hoc network simulator, and cognitive mesh networking test-bed. His current research interests include cognitive radio, wireless multihop networks (ad hoc, mesh, and WSN), wireless multimedia network, and green communications.

Dr. Zhao was the Symposium Co-Chair of ChinaCom in 2009 and 2010. He was the Technical Program Committee Co-Chair of IEEE ISCIT 2010 (10th IEEE International Symposium on Communication and Information Technology).

Rongpeng Li received the B.E. degree (Excellent Graduates) from Xidian University, Xi'an, China, in 2010, and the Ph.D. degree (Excellent Graduates) from Zhejiang University, Hangzhou, China in 2015.

From 2015 to 2016, he was a Researcher with the Wireless Communication Laboratory, Huawei Technologies Company Ltd., Shanghai, China. He is currently a Post-Doctoral Researcher with the College of Computer Science and Technologies, Zhejiang University. He has authored or/co-authored several papers. His current research interests include reinforcement learning, data mining and all broad-sense network problems (e.g., resource allocation, security, and IPv6).

Dr. Li serves as an Editor for *China Communications*.

Honggang Zhang is currently a Full Professor with the College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, China. He is also an Honorary Visiting Professor with the University of York, York, U.K. He was the International Chair Professor of Excellence with the Université Européenne de Bretagne, Rennes, France, and Supélec, Gif-sur-Yvette, France. He has co-authored and edited *Cognitive Communications-Distributed Artificial Intelligence (DAI), Regulatory Policy and Economics, Implementation* (Wiley) and *Green Communications: Theoretical Fundamentals, Algorithms and Applications* (CRC Press).

Prof. Zhang is active in research on green communications and was the leading Guest Editor of the *IEEE Communications Magazine* special issues on green communications. He is an Associate Editor-in-Chief of *China Communications* and the Series Editors of the *IEEE Communications Magazine* for its Green Communications and Computing Networks Series. He served as the Chair of the Technical Committee on Cognitive Networks of the IEEE Communications Society from 2011 to 2012.