An internet traffic classification method based on echo state network and improved salp swarm algorithm

Meijia Zhang[1], Wenwen Sun[2], Jie Tian[1], Xiyuan Zheng[1] and Shaopeng Guan[2]

[1] School of Data Science and Computer Science, Shandong Women's University, Jinan, Shandong, China
[2] School of Information and Electronic Engineering, Shandong Technology and Business University, Yantai, Shandong, China

ABSTRACT

Internet traffic classification is fundamental to network monitoring, service quality and security. In this paper, we propose an internet traffic classification method based on the Echo State Network (ESN). To enhance the identification performance, we improve the Salp Swarm Algorithm (SSA) to optimize the ESN. At first, Tent mapping with reversal learning, polynomial operator and dynamic mutation strategy are introduced to improve the SSA, which enhances its optimization performance. Then, the advanced SSA are utilized to optimize the hyperparameters of the ESN, including the size of the reservoir, sparse degree, spectral radius and input scale. Finally, the optimized ESN is adopted to classify Internet traffic. The simulation results show that the proposed ESN-based method performs much better than other traditional machine learning algorithms in terms of per-class metrics and overall accuracy.

INTRODUCTION

With the expansion of the Internet scale, the increasing kinds of network applications and huge amount of network traffic restrict the effective management of network equipment. Network security and congestion problems have become increasingly prominent. Internet traffic classification is the foundation of network security and management. According to different network application classes, the traffic generated by network communication is classified, which can optimize network configuration, reduce the risks of network security, and provide better service based on analysis of user behavior (*Callado et al., 2009*).

Traditional internet traffic classification schemes mainly comprise port-based approach and deep packet inspection-based approach (*Zhang et al., 2014*). According to the port mapping table stipulated by the Internet Assigned Numbers Authority (IANA), the port-based approach classifies the network traffic of a specific port into the corresponding network application (*Aceto et al., 2010*). However, with the popularity of novel network applications such as FTP and P2P, lots of random ports are used to transfer data. Hence, the port-based approach is rapidly eliminated (*Rezaei & Liu, 2019*). Since emerging services dynamically bind ports, the deep packet inspection (DPI)-based approach has

Corresponding author
Shaopeng Guan,
konexgsp@gmail.com

become a relatively mature technology recognized in the industry. This method identifies the application by analyzing the protocol signature in the payload (*Finsterbusch et al., 2013*). Nevertheless, due to the increase of the links in the backbone network, analyzing the complete payload not only has large computational overhead, but also may cause unnecessary privacy disputes. In addition, the payload encryption technology limits the identification capability of the DPI-based approach.

Machine learning (ML) does not rely on parsing protocol content or matching protocol port to classify network application, but distinguishes network applications according to different statistical features in the transmission process of the traffic generated by different applications. It is not affected by dynamic ports, load encryption or network addresses (*Nguyen & Armitage, 2008*; *Pacheco et al., 2018*). At present, the ML-based methods include Support Vector Machine (SVM) (*Yuan et al., 2010*; *Este, Gringoli & Salgarelli, 2009*; *Sun et al., 2018*; *Jing et al., 2011*; *Cao et al., 2017*), Decision Tree (*Tong, Qu & Prasanna, 2017*; *Li & Moore, 2007*), K-Nearest Neighbors (KNN) (*Qi et al., 2020*; *Ma, Du & Cao, 2020*), Bayes (*Dias et al., 2019*; *Zhang et al., 2012*; *Moore & Zuev, 2005*), and Neural Network, *etc.* Among them, Neural Network processes information by simulating the thinking mode of human brain and has the characteristics such as self-organization, self-learning and self-adaptation. It is applied to network traffic classification and has achieved satisfactory classification performance (*Pacheco et al., 2018*; *Liu et al., 2017*). *Liu et al. (2020)* proposed an HTTP traffic classification method based on the bidirectional Gated Recurrent Unit (GRU) Neural Network. This method utilized the bidirectional GRU to extract the forward and backward features of byte sequences in the session, and then employed the attention mechanism to assign the weight of the features according to their contributions. *Wang et al. (2017)* cut the original data traffic and input it into the Convolutional Neural Network (CNN) to classify network traffic. *Yang et al. (2018)* adopted CNN to extract high-dimensional features of the network traffic, and then extracted the representative features from these features based on the AutoEncoder (AE), which achieves network traffic identification. *Zeng et al. (2019)* proposed a framework of the network traffic classification based on Deep-Full-Range (DFR), which respectively utilizes CNN, LSTM and Stacked Auto Encoder (SAE) to extract the spatial, temporal and coding features of the original traffic, and combines these features to achieve comprehensive understanding of the original traffic. *Lotfollahi et al. (2020)* used header information and payload data to train CNN and SAE respectively. The experiment shows that the identification performance of CNN is slightly better than that of SAE. *Ren, Gu & Wei (2021)* proposed a tree structural recurrent neural network (Tree-RNN), which divides a large classification into small classifications by using the tree structure. A specific classifier is set for each small classification after division. With multiple classifiers employed, Tree-RNN can sovle the problem of relatively poor classification effect for multi-classification. *Li et al. (2021)* proposed a method of mobile service traffic classification based on joint deep learning with attention mechanism. In the first step, a joint deep learning model is designed as a basic classifier, which learns features of mobile service traffic from multiple time scales. In the second step, the attention mechanism is adopted to aggregates the basic predictions generated in the first step, which filters out useless information. Compared with traditional

Table 1 Summary of related work.

Work	Simple description	Comments
Aceto et al. (2010)	Classify the network traffic of a specific port into the corresponding network application	Be affected by dynamic ports
Finsterbusch et al. (2013)	Identify the application by analyzing the protocol signature in the payload	Have large computational overhead, and may cause unnecessary privacy disputes
Liu et al. (2020)	Utilize the bidirectional GRU to extract the forward and backward features of byte sequences in the session, and then employed the attention mechanism to assign the weight of the features according to their contributions	
Wang et al. (2017)	Cut the original data traffic and input it into the CNN to classify network traffic	
Yang et al. (2018)	Adopt CNN to extract high-dimensional features of the network traffic, and then extract the representative features from these features based on the AE	
Zeng et al. (2019)	Utilize CNN, LSTM and SAE to extract the spatial, temporal and coding features of the original traffic, and combine these features to achieve comprehensive understanding of the original traffic	
Lotfollahi et al. (2020)	Use header information and payload data to train CNN and SAE respectively	Have long training time and high computational cost
Ren, Gu & Wei (2021)	Divide a large classification into small classifications by using the tree structure	
Li et al. (2021)	Design a joint deep learning model as a basic classifier, and then adopt the attention mechanism to aggregates the basic predictions generated in the first step	
This paper	Utilize the advanced SSA to optimize the hyperparameters of the ESN, and then adopt the optimized ESN to classify Internet traffic	Simplify the training process, and have the characteristics of easy implementation and fast training speed

ML algorithm, the neural network-based methods have achieved better identification effect. However, the methods have some disadvantages such as long training time and high computational cost since the structure of adopted neural network is complex and their weights have to be determined through multiple iterations. With respect to the related researches, they are summarized in Table 1, including the simple descriptions on approach and comments.

Echo state network (ESN), as a kind of neural network, adopts the reserve pool composed of sparsely connected neurons as the hidden layer to perform high-dimensional and non-linear representation of the input data (*Grigoryeva & Ortega, 2018*). It only needs to train the weights from the reserve pool to the output layer, which simplifies the training process and solves problems of traditional neural networks such as complex training and difficult determination of network structure. ESN, with the characteristics of easy implementation and fast training speed, has good application prospects in time series forecasting

(*Wang et al., 2018*; *Long, Zhang & Li, 2019*; *Zhang et al., 2019*; *Duan & Wang, 2015*). In this paper, we propose an internet traffic classification method based on ESN.

Although the echo state network has advantages in solving the above problems, there are also some problems. For example, the stability of the reserve pool will affect the generalization ability of the network, which is easy to cause problems such as over-fitting. Hu et al. proposed an improved method of ESN, which combines ESN and deep learning, and used the efficient learning ability of deep learning to improve the stability of the reserve pool (*Hu, Wang & Lv, 2020*). When ESN is used to classify network traffic, hyperparameters such as the input scale and sparse degree are of great importance to the classification performance. The results obtained through different hyperparameter configurations vary greatly. Therefore, selecting the optimal hyperparameters are critical to the results of network traffic classification. The traditional hyperparameter optimization method finds out the hyperparameters of the ESN by manual setting. However, this method has the disadvantage of time-consuming and it is difficult to select the optimal hyperparameters. Swarm Intelligence (SI) Optimization algorithms are inspired by the collective behavior of creatures. Since the individuals in the population can interact and share information, the SI algorithm has the characteristics of strong flexibility and fast convergence, and can provide satisfactory solutions when applied to the automatic optimization of hyperparameters. At present, the Genetic Algorithm (GA) (*Zhong et al., 2017*), the Particle Swarm Algorithm (PSO) (*Chouikhi et al., 2017*), the Fruit Fly Optimization Algorithm (FOA) (*Tian, 2020*; *Zhang et al., 2020*), the Differential Evolution Algorithm (DE) (*Hu, Wang & Tao, 2021*) and the Grey Wolf Optimizer (GWO) (*Kohli & Arora, 2018*) algorithm have been adopted to automatically optimize hyperparameters of ESN. However, the reservoir of ESN contains many nodes and its search space is large, hence the above-mentioned algorithms are not suitable for optimizing hyperparameters with a large range of values. Salp Swarm Algorithm (SSA) is a new type of SI algorithm that simulates the foraging behavior of salps in the biological world. It guides search optimization by simulating the population behavior of salps sailing and foraging in ocean, and has the advantages of high optimization accuracy, strong search ability and good robustness. SSA has been employed to function optimization (*Rizk-Allah et al., 2019*), combination optimization (*Abualigah et al., 2020*), and shortest path solution (*Ateya et al., 2019*), *etc.*, and can obtain the optimal solution. *Mirjalili et al. (2017)* evaluated SSA on 19 well-known mathematical functions and 2 optimization problems, and compared it with state-of-art SI algorithms such as PSO, GA and FOA. The results show that SSA has better optimization performance than other SI algorithms. However, like other SI algorithms, SSA also has some shortcomings, such as inadequate spatial search at the early stage and reduced population diversity at the iterative process. To enable SSA to find the optimal solution more accurately, we introduce Tent mapping with reverse learning, polynomial operator and dynamic mutation strategy to improve SSA, and then adopt the advanced SSA to optimize the important hyperparameters of ESN automatically. Our main contributions are as follows:

(1) Network traffic classification plays a vital role in analyzing user behavior, enhancing network controllability, improving service quality and ensuring network security. Considering ESN has the advantages such as excellent classification performance, fast
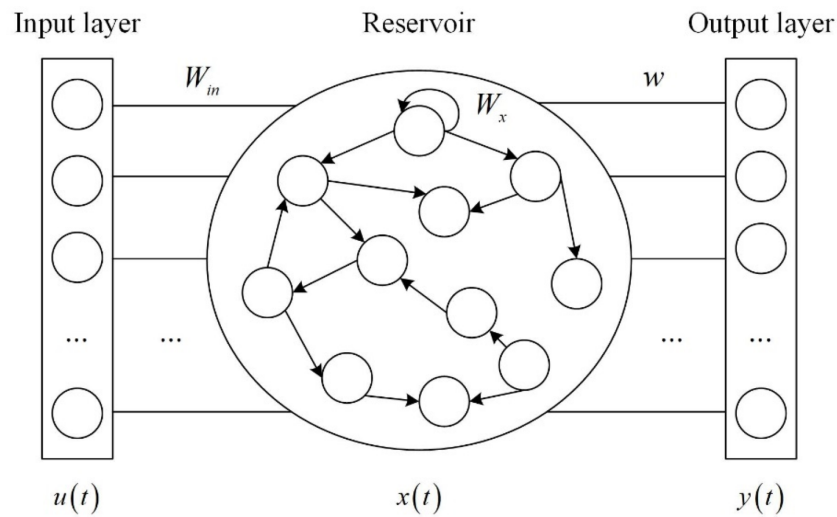
Figure 1 ESN structure.

training speed and easy implementation, we propose a classification method of Internet traffic based on ESN.

(2) We improve the SSA. Firstly, Tent mapping with reverse learning is introduced to initialize the population, which makes the distribution of the initial population position uniform and improves the search efficiency of the SSA. Secondly, polynomial operator is used to maximize the diversification of search domain and improve the global exploration ability of the SSA. Finally, a dynamic mutation strategy is adopted to increase the population diversity at the later stage and avoid the SSA falling into local optimum.

(3) Hyperparameters of ESN such as input scale and sparse degree are of great importance to the classification performance. The reservoir of ESN contains many nodes and large search space, thus the optimization effect of the traditional hyperparameter optimization method still needs to be improved. We use the advanced SSA to optimize important hyperparameters of ESN.

The rest of the paper is organized as follows: 'Materials & Methods' introduces the ESN and discusses its important hyperparameters. Then, we improves the SSA and describes the network traffic classification method. In 'Results', we perform experiments to verify the effectiveness of the proposed scheme. Finally, we summarizes our work in 'Conclusion'.

MATERIALS & METHODS

ESN

ESN is composed of input layer, reservoir and output layer. The reservoir contains hundreds of sparsely connected neurons, and the connection weights between neurons are randomly generated and fixed. Figure 1 shows the ESN structure.

The state equation and output equation of ESN are as follows (*Han & Xu, 2017*):

$$x(t) = \varphi\left(W_{in}u(t) + W_x x(t-1) + W_{back}y(t-1)\right) \tag{1}$$

$$y(t) = f_{out}\left(W_{out}\left(u(t), x(t), y(t-1)\right)\right). \tag{2}$$

where $u(t) \in R^{M \times 1}$ is the input vector, $y(t) \in R^{M \times 1}$ is the output vector, $b_x \in R^{N \times 1}$ is the input bias, and $b \in R^{M \times 1}$ is the output bias. The state $x(t) \in R^{N \times 1}$ at the current time is calculated from the input vector $u(t)$ at the current time t and the state of the reservoir $x(t-1)$ at the previous time $t-1$. $\varphi(\cdot)$ is the activation function of the neuron, which can select Sigmoid function or tanh function. The element of the input-reservoir connection weight matrix $W_{in} \in R^{N \times K}$ is in the interval $[-1,1]$. $W_x \in R^{N \times N}$ is the internal connection weight matrix of the reservoir. $W_{back} \in R^{N \times L}$ is the output-reservoir connection weight matrix. $W_{out} \in R^{L \times (K + N \times L)}$ is the output connection weight matrix. W_{in}, W_x and W_{back} are generated randomly and remain unchanged during the training phase of ESN. The network only needs to train the output connection weight matrix W_{out}, which reduces the computational complexity.

The core of ESN is the reservoir. Its performance depends on four crucial hyperparameters: the size of the reservoir N, spectral radius R, sparse degree D and input scale S. How to select these hyperparameters is very important (*Duan & Wang, 2015*).

(1) Size of the reservoir

The size of the reservoir N, as the most important hyperparameter affecting the performance of ESN, is the number of neurons in the reservoir. The larger the number of neurons is, the better the classification performance is. However, if the number of neurons is too large, overfitting will be caused.

(2) Spectral radius

The spectral radius R is the absolute value of the maximum eigenvalue of the internal connection weight matrix W_x of the reservoir. $R < 1$ is a necessary condition to ensure the stability of the network.

(3) Sparse degree

The sparse degree D indicates the sparsity of neuron connections. The neurons in the reservoir are sparsely connected rather than fully connected. The larger the value is, the stronger the nonlinear approximation ability is.

(4) Input scale

The input scale S refers to the scale factor before data is input into the reservoir, and represents the range of input connection weight. According to Eq. (1), it determines the working interval of activation function and the extent to which the input data affect the state of the reservoir. It's usually in the interval $[0,1]$.

The performance of ESN relies heavily on the above hyperparameters, and the results obtained through different hyperparameter configurations vary greatly.

The advanced SSA

SSA is a new type of SI algorithm with the advantages of high optimization accuracy, good robustness and high convergence rate. SSA divides the population into leaders and followers. They form a salp chain to perform population optimization. To enable it to find the optimal solution more accurately, we improved the SSA.

Population initialization

The uniform distribution of the population can effectively maintain the population diversity and improve the optimization performance. The initial population of SSA is generated randomly. Lack of prior knowledge leads to uneven distribution and poor initial population diversity.

To enhance population diversity and search efficiency, the Tent mapping with reverse learning is introduced to initialize the population. The Tent mapping has the characteristics of randomness, ergodicity and regularity. Thus, it can generate initial salp population with rich diversity. Then, the reverse learning strategy is adopted to optimize the population and generate the reverse population. At last, the population generated by the Tent mapping and its reverse population are merged and sorted. The salps with better fitness value are selected to form the initial population. Tent mapping with reverse learning is introduced to expand the search range of the population, reduce invalid search, and improve the search efficiency. The population generated by Tent mapping is expressed as follows (*Arora & An, 2019*):

$$x_{d+1} = \begin{cases} 2x_d, 0 \le x_d \le \dfrac{1}{2} \\ 2(1-x_d), \dfrac{1}{2} \le x_d \le 1 \end{cases} \tag{3}$$

where x_d and x_{d+1} are the respective values of the dth and $(d+1)$th dimensions of the population generated by the Tent mapping.

The reverse learning strategy is used for the population generated by Tent mapping, and the obtained reverse population is as follows:

$$x'_{d+1} = u_d + l_d - x_{d+1}. \tag{4}$$

where x'_{d+1} are the values of the $(d+1)$th dimension of the reverse population; u_d is the upper limit of the dth dimension; and l_d is the lower limit of the dth dimension.

Leader position update

Adequate global exploration is helpful for the algorithm to obtain better optimization results. Traditional SSA conducts global exploration by introducing random numbers into leader position update. However, the introduced random numbers have strong randomness and cannot fully perform global exploration. Polynomial operator can maximize the diversification of search domain and enhance the convergence speed of the SSA at the later stage. Therefore, we introduce polynomial operator into leader position update.

The formula of improved leader position update is as follows:

$$X_d^{l'} = F_d + \delta \cdot (u_d - l_d) \tag{5}$$

$$\delta = \begin{cases} \left[2u+(1-2u)(1-\delta_1)^{\eta_m+1}\right]^{\frac{1}{\eta_m+1}}-1, & u \le 0.5 \\ 1-\left[2(1-u)+2(u-0.5)(1-\delta_2)^{\eta_m+1}\right]^{\frac{1}{\eta_m+1}}, & u > 0.5 \end{cases} \tag{6}$$

$$\delta_1 = \left(X_d^l - l_d\right)/(u_d - l_d) \tag{7}$$

$$\delta_2 = \left(u_d - X_d^l\right)/(u_d - l_d). \tag{8}$$

where F_d is the food position (*i.e.,* the optimal position of salp in the population); u is the random number in the interval $[0,1]$; η^m represents the distribution index; $X_d^{l'}$ is the position of updated leader on the dth dimension; and X_d^l is the position of current leader on the dth dimension.

Follower position update

At the later stage of iteration, followers will gather near the current food source, which reduces the population diversity and the search ability of the SSA. To avoid the premature phenomenon at the later stage, we introduce dynamic mutation strategy into the follower position update, which increases the diversity of salp population at the later stage and improves the convergence accuracy of the SSA. At present, researchers have proposed a variety of mutation algorithms, such as Gaussian mutation and Cauchy mutation (*Li et al., 2017*). Compared with Gaussian operator, Cauchy operator has longer wings and can generate a large range of random numbers, so that the SSA has a greater chance to avoid local optimum. In addition, less time is needed to search the nearby area when the peak value is low. Therefore, we introduce Cauchy mutation into follower position update.

The formula of improved follower position update is as follows:

$$X_d^{m'} = \frac{1}{2}\left(X_d^m + X_d^{m-1}\right) + \eta * C(0,1)(u_d - l_d) \tag{9}$$

$$\eta = e^{-\lambda\frac{t}{T}}. \tag{10}$$

where X_d^m and X_d^{m-1} are the respective positions of the mth and $(m-1)^{\text{th}}$ followers on the $d$pth dimension before the update; $X_d^{m'}$ is the positions of the mth followers on the dth dimension after the update; η is the mutation weight which decreases with the increase of the number of iteration; T is the maximum number of iteration; t is the current number of iteration; λ is a constant and its value is 10; $C(0,1)$ is a random number generated by the Cauchy operator and its scaling parameter is 1.

If the mutation frequently occurs during the iteration process, it will not be conducive to the algorithm convergence. Therefore, the mutation trigger mechanism is introduced. If the fitness value of mutated follower position is better, the Cauchy mutation will be introduced into the follower position. Otherwise, the Cauchy mutation will not be introduced.

The steps of the advanced SSA (ASSA) are shown in Algorithm 1.

Algorithm 1: ASSA.

Input: population size NP, maximum number of iteration T

Output: the optimal position of salp in the population

1 Adopt the Tent mapping with reverse learning to initialize the salp population.

2 for $t = 1$ to T do

3 Calculate the fitness value of salps in the population

4 Sort salps in the population according to fitness value

5 Choose food. The salp position with the best fitness is regarded as the food position

6 Choose leaders and followers. After selecting the food, there are $N - 1$ salps remaining in the population. The salps with the first half of the fitness value are regarded as the leaders, and the others are regarded as followers

7 Update the leader position according to formula Eq. (5)

8 Update the follower position according to formula Eq. (9)

9 end for

In Algorithm 1, we at first employ the Tent mapping with reverse learning to initialize the salp population. Then, we calculate the fitness values of salps in the population and sort salps in the population according to the fitness values. The salp position with the best fitness value is regarded as food position. After the food position is selected, there are $N - 1$ salps left in the population. The salps with the first half of the fitness values are regarded as the leaders, and the others are regarded as followers. We update the position of leaders and followers respectively according to Eqs. (5) and (9). Repeat the above steps until meeting the stopping condition (*e.g.,* the maximum number of iterations). Finally, output the salp position with the best fitness value.

Performance analysis of ASSA

To verify the performance of the ASSA algorithm, two typical functions of Sphere and Griewank, are selected for function optimization and convergence test of the algorithm. Sphere is a unimodal function and Griewank is a multimodal function. We compare ASSA with the PSO, GA and SSA. The number of algorithm iterations is set to 500. The testing results of the four algorithms on the functions are shown in Figs. 2 and 3.

It can be seen from Figs. 2 and 3 that ASSA has obvious advantages in convergence speed and accuracy compared to the PSO, GA, and SSA. Therefore, the performance of the ASSA is significantly improved over the traditional optimization algorithms.

Scheme design

We use ASSA-optimized ESN for Internet traffic classification. The basic idea is as follows: The ASSA is utilized to find the salp position with the best fitness value. At the end of iteration, each dimension of the salp position is assigned to the corresponding hyperparameter of the reservoir of ESN, which establishes the network traffic classification
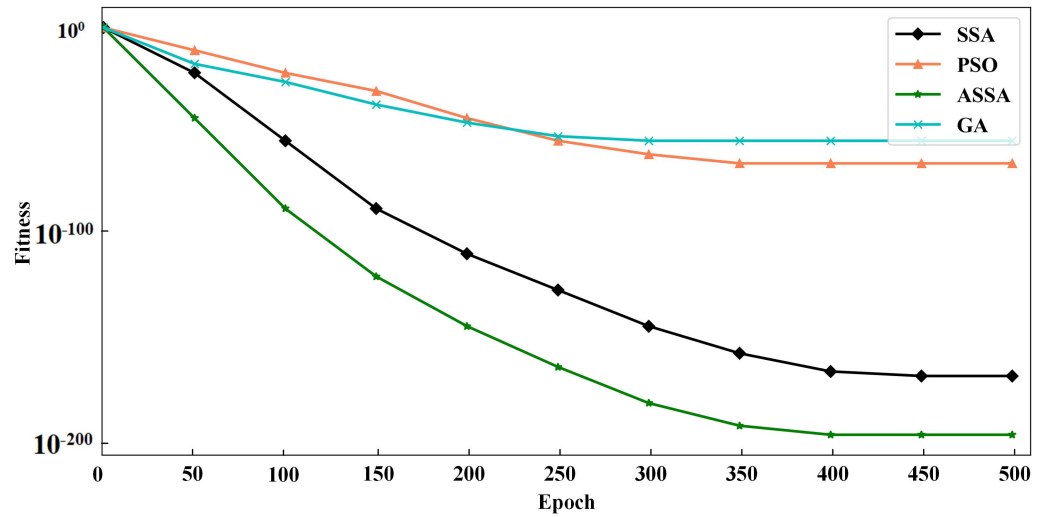
Figure 2 The testing results with Sphere function.
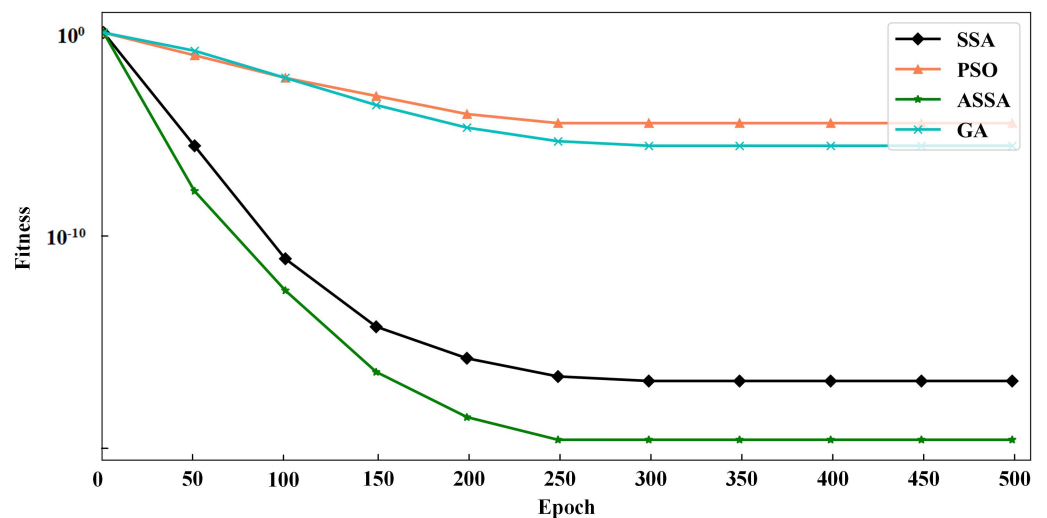
Figure 3 The testing results with Griewank function.

model. The flowchart of ESN-based network traffic classification is shown in Fig. 4. The steps of ESN-based network traffic classification are as follows:

Step 1: Pre-process the network traffic classification dataset. There are two pre-processing ways.

(1) One-hot coding. We implement one-hot coding for discrete features.

(2) Min-max normalization. The large difference between the data of the same attribute affects the training of the network. Therefore, we perform min-max normalization on continuous features.

Step 2: Divide the dataset into training set and testing set.
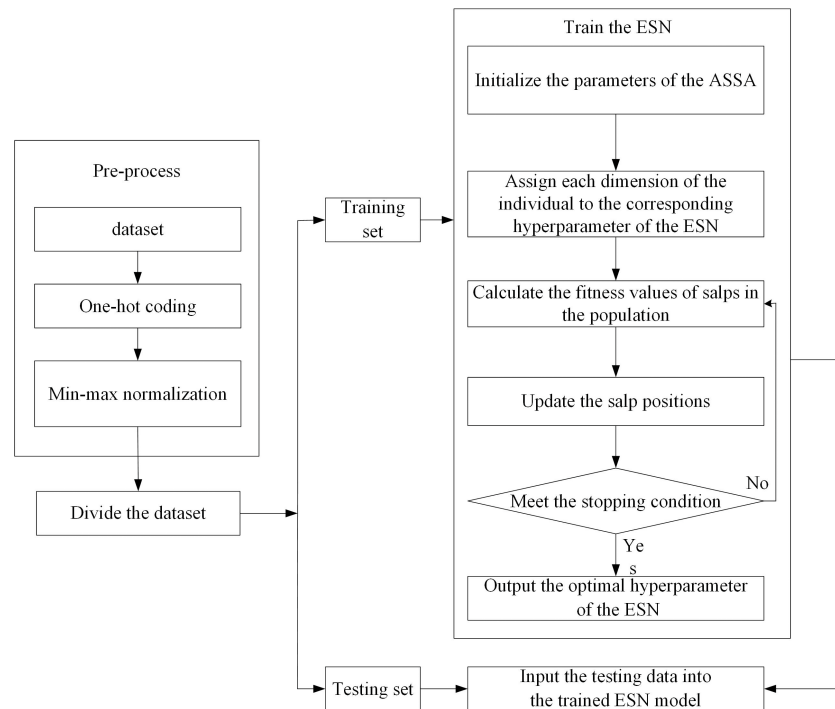
Figure 4 **The flowchart of ESN-based network traffic classification.**

Step 3: Train the ESN and adjust its hyperparameters.

(1) Initialize the parameters of the ASSA, such as the salp population size and the maximum number of iterations. Use the Tent mapping with reverse learning to initialize the salp population. Each dimension of the individual in the population represents a hyperparameter of the ESN, and different hyperparameters have different ranges. Therefore, each dimension of the individual is constrained.

(2) Assign each dimension of the individual to the corresponding hyperparameter of the ESN: the size of the reservoir N, spectral radius R, sparse degree D and input scale S.

(3) Calculate the fitness values of salps in the population according to the training samples and fitness function, and arrange the fitness values in ascending order to find the salp position with the optimal fitness value. If the stopping condition is met, go to **Step 4**. Otherwise, go to **Step 3**. The fitness function is the overall accuracy of network traffic classification.

Step 4: Input the testing data into the trained ESN model, and then get the classification result of each sample.

RESULTS

Experimental dataset

To verify the effectiveness of the proposed scheme, we conduct experiments using two public datasets called Moore dataset (*Lopez-Martin et al., 2017*) and NISM dataset (*Demertzis & Iliadis, 2016*), which are from raw traffic data. With a long interval between them, they

Table 2 Moore dataset statistics.			
Sample type	**The number of training samples**	**The number of testing samples**	**Proportion**
WWW	241,186	86,906	86.906%
Mail	21,000	7,567	7.567%
Ftp-data	4,261	1,536	1.536%
Ftp-pasv	1,976	712	0.712%
Ftp-control	2,245	809	0.809%
Services	1,543	556	0.556%
Database	1,947	701	0.701%
P2P	1,539	555	0.555%
Attack	1,318	475	0.475%
Mutimedia	423	153	0.153%
Interactive	82	28	0.028%
Games	6	2	0.002%
Totality	277,526	100,000	100%

have different data terminals and IP addresses, which enables the effective evaluation of the generality of the proposed scheme. Each dataset includes the training set and the testing set. The proportion of each category in the training set and testing set is consistent with that of the original dataset. 100,000 samples are randomly selected as the testing set, and the others are the training set.

(1) Moore dataset

The Moore dataset comes from the traffic flowing through the network outlet of a biological institute from 0 to 24 h on August 20, 2003. 377,526 network samples are obtained from the 24 h traffic by sampling algorithm. They are divided into 12 application classes. Each sample contains 249 attributes, among which the last one is the category corresponding to each sample. Table 2 reports the Moore dataset statistics.

(2) NISM dataset

The NISM dataset comes from the network traffic of the Information Technology Operations Center of the U.S. Military Academy in 2013. The dataset contains 713,851 network traffic samples, which are divided into 11 application classes. The NISM dataset statistics are shown in Table 3.

Evaluation index

We use the following evaluation indexes to evaluate the classification performance. The samples in the training set are divided into m application classes. TP_i represents the number of its samples that are correctly classified as belonging to class i. FN_i represents the number of its samples that are misjudged as other types. FP_i represents the number of the samples from other application classes that are misjudged as belonging to class i. The evaluation indexes are defined as follows.

(1) The accuracy rate of class i

$$Acc_i = TP_i/(TP_i + FN_i).$$ (11)

Table 3 NISM dataset statistics.

Sample type	The number of training samples	The number of testing samples	Proportion
DNS	32,691	5,325	5.325%
Ftp	1,486	242	0.242%
Http	10,236	1,668	1.668%
Telent	1,076	175	0.175%
Lime	555,738	90,533	90.533%
Local forwarding	2,199	358	0.358%
Remote forwarding	2,083	339	0.339%
Scp	2,102	342	0.342%
Sftp	2,074	338	0.338%
Shell	2,142	349	0.349%
X11	2,025	330	0.330%
Totality	613,851	100,000	100%

(2) The recall rate of class i

$$R_i = TP_i/(TP_i + FN_i). \tag{12}$$

(3) The F-measure of class i

$$F_i = 2 \times P_i \times R_i/(P_i + R_i.) \tag{13}$$

(4) The overall accuracy rate

$$OA = \sum_{i=1}^{m} TP_i / \sum_{i=1}^{m} (TP_i + FP_i). \tag{14}$$

Among the above indexes, the accuracy and recall rate of per class can reflect the classification performance of the proposed scheme for per class. The F-measure, as the harmonic average of the accuracy and the recall rate, gives a better comprehensive evaluation of the classification ability. In addition, the overall accuracy can reflect the proportion of correctly classified samples to all samples.

Experimental results

(1) Experiments of ESN hyperparameter optimization

The range of ESN hyperparameters is set as follows: The size range of the reservoir is set as [30, 300], the range of spectral radius [0.1, 0.99], the range of sparse degree [0.01, 1], and the range of input scale [0.1, 1]. Mirjalili et al. evaluated SSA on more than 20 test functions and 5 optimization problems, and compared them with typical swarm intelligence optimization algorithms such as PSO, GA, and ABC (*Rizk-Allah et al., 2019*). The experimental results show that, compared with above algorithms, SSA has better optimization performance. Therefore, referring to the literature (*Hu, Wang & Tao, 2021*), the population size is set as 21, and the maximum number of iterations 20. On the Moore dataset and the NISM dataset respectively, the optimal hyperparameter values of ESN selected by ASSA are shown in Table 4.

Table 4 Hyperparameters of ESN.

Dataset	Hyperparameter	Value
Moore dataset	the size of the reservoir	98
	spectral radius	0.78
	sparse degree	0.89
	input scale	0.95
NISM dataset	the size of the reservoir	77
	spectral radius	0.83
	sparse degree	0.78
	input scale	0.90
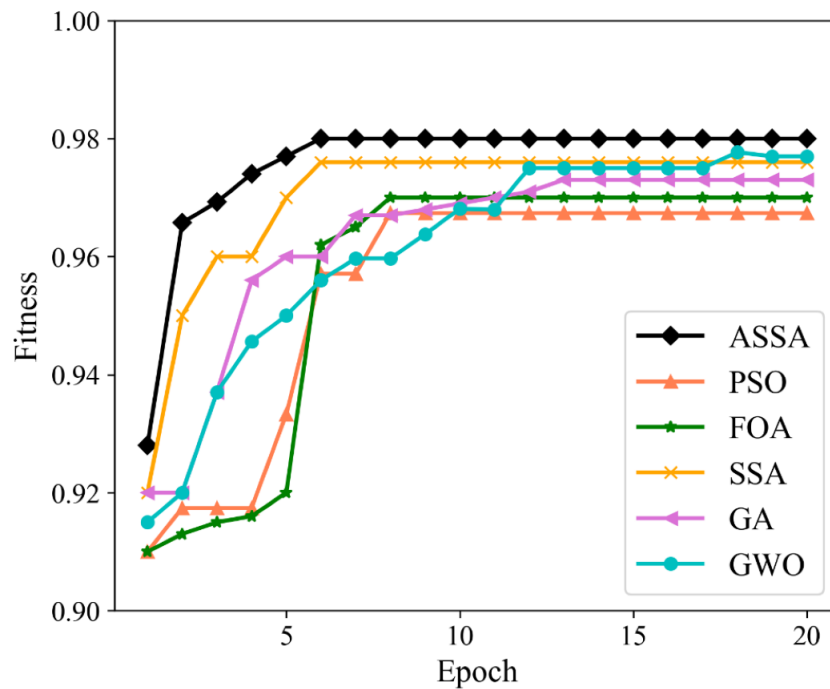
Figure 5 The changing curves of fitness values on the Moore dataset.

We utilize PSO, FOA, GA, GWO, SSA and ASSA to optimize the hyperparameters of the ESN. The fitness function, with a direct impact on the optimal solution of the model, is usually defined by the actual problem. For the network traffic classification problem, the overall accuracy is taken as the fitness function. On the Moore dataset and the NISM dataset, the changing curves of the fitness values of the six algorithms in the iteration process are shown in Figs. 5 and 6 respectively.

(2) Comparison of different machine learning algorithms

To verify the effectiveness of the proposed scheme, it is compared with different ML algorithms such as SVM, SAE, CNN, GRU and Deep Belief Networks (DBN) algorithms on Moore dataset and NISM dataset, respectively. On the Moore dataset and the NISM
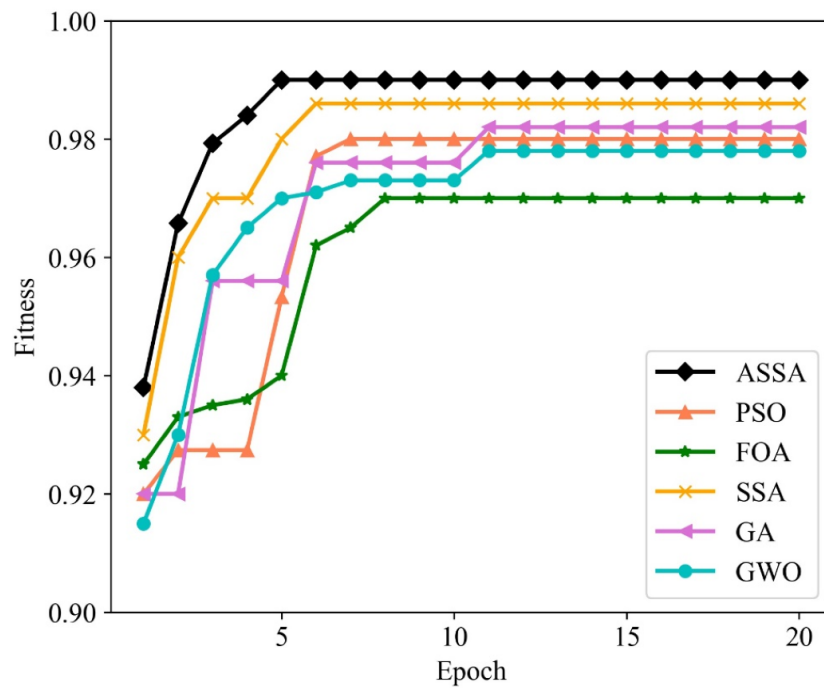
Figure 6   The changing curves of fitness values on the NISM dataset.

Table 5   The parameter values of the comparison algorithms.

Dataset	Method	Parameter values
Moore dataset	SVM	RBF kernel, $C = 1.0$, gamma $= 0.055$
	SAE	Hidden layers=2 (the node number of each layer is 50)
	CNN	Hidden layers=2 (the node number of each layer is 50)
	GRU	Hidden layers=2 (the node number of each layer is 50)
	DBN	Hidden layers=2 (the node number of each layer is 50)
NISM dataset	SVM	RBF kernel, $C = 1.0$, gamma $= 0.049$
	SAE	Hidden layers=2 (the node number of each layer is 40)
	CNN	Hidden layers=2 (the node number of each layer is 40)
	GRU	Hidden layers=2 (the node number of each layer is 40)
	DBN	Hidden layers=2 (the node number of each layer is 40)

dataset, the parameter values of the comparison algorithms are shown in Tables 5, 6, 7, 8, 9, 10 and 11 and Fig. 7 show the classification results of different algorithms on the two datasets.

For network traffic classification problems, classification time is also an important indicator. Classification time includes training time and testing time. The training time and testing time of each algorithm on the datasets are shown in Table 12.

Table 6   The class accuracy of different ML algorithms on the Moore dataset.

Method	WWW	Mail	F-D	F-P	F-C	SERV	DB	P2P	ATT	MULT	INT	Games
SVM	0.90	0.90	0.93	0.82	0.73	0.88	0.62	0.80	0.82	0.84	0.00	0.00
SAE	0.93	0.97	0.97	1.00	0.83	0.93	0.82	0.79	0.93	0.86	0.83	0.00
CNN	0.94	0.97	0.98	0.95	0.95	0.96	0.94	0.81	0.94	0.89	0.47	1.00
GRU	0.97	0.97	0.97	0.97	0.91	0.99	0.95	0.76	0.99	0.88	0.56	0.00
DBN	0.94	0.92	0.95	0.96	0.80	0.97	0.76	0.79	0.92	0.86	1.00	0.00
ESN	0.92	0.95	0.95	0.90	0.91	0.93	0.92	0.80	0.92	0.86	0.83	0.00
SSA-ESN	0.97	0.97	0.98	0.95	0.93	0.97	0.94	0.89	0.95	0.89	0.90	1.00
Ours	0.99	0.98	1.00	0.95	1.00	1.00	0.99	0.94	0.99	0.91	1.00	1.00

Table 7   The class accuracy of different ML algorithms on the NISM dataset.

Method	DNS	Ftp	Http	Telent	Lime	Local forwarding	Remote forwarding	Scp	Sftp	Shell	X11
SVM	0.90	0.94	0.98	0.97	0.76	1.00	1.00	0.66	0.68	0.91	0.92
SAE	0.90	0.94	0.99	0.93	0.81	1.00	1.00	0.70	0.76	0.93	1.00
CNN	0.95	1.00	1.00	1.00	0.81	1.00	1.00	0.99	0.95	0.97	1.00
GRU	0.85	0.98	0.99	0.96	1.00	0.99	1.00	0.98	0.94	0.97	0.99
DBN	0.91	0.94	0.99	0.99	0.82	1.00	1.00	0.72	0.81	0.94	0.99
ESN	0.89	0.94	0.99	0.94	0.89	1.00	1.00	0.85	0.80	0.94	0.99
SSA-ESN	0.97	1.00	0.99	1.00	0.98	1.00	1.00	0.98	0.98	0.98	1.00
Ours	0.99	1.00	1.00	1.00	0.99	1.00	0.99	0.99	0.99	0.99	1.00

Table 8   The class recall rate of different ML algorithms on the Moore dataset.

Method	WWW	Mail	F-D	F-P	F-C	SERV	DB	P2P	ATT	MULT	INT	Games
SVM	0.97	0.88	0.92	0.82	0.75	0.94	0.89	0.37	0.81	0.82	0.00	0.00
SAE	0.98	0.92	1.00	0.96	0.90	0.94	0.98	0.71	0.84	0.86	0.50	0.00
CNN	0.99	0.96	1.00	0.93	0.86	0.95	0.95	0.83	0.88	0.83	0.70	0.50
GRU	0.97	0.98	1.00	0.97	0.89	0.94	0.97	0.94	0.83	0.87	0.50	0.00
DBN	0.98	0.92	0.99	0.93	0.87	0.93	0.95	0.70	0.81	0.81	0.20	0.00
ESN	0.97	0.92	0.92	0.93	0.85	0.94	0.95	0.89	0.83	0.81	0.50	0.00
SSA-ESN	1.00	0.98	1.00	0.99	0.95	0.99	0.98	0.94	0.94	0.94	0.80	0.50
Ours	1.00	0.99	1.00	0.99	0.96	0.99	0.98	0.95	0.95	0.95	0.90	0.50

DISCUSSION

As shown in Figs. 5 and 6, the ASSA initializes the population using the Tent mapping with reverse learning, which promotes the uniform distribution of the initial population and improves the search efficiency. Then, it introduces polynomial operator to maximize the diversification of search domain, which improves the global exploration ability. Finally, the ASSA introduces a dynamic mutation strategy to increase the population diversity at the later stage and avoid local optimum. Therefore, the ASSA has better fitness value and

Table 9 The class recall rate of different ML algorithms on the NISM dataset.

Method	DNS	Ftp	Http	Telent	Lime	Local forwarding	Remote forwarding	Scp	Sftp	Shell	X11
SVM	0.72	1.00	0.96	0.93	0.90	0.97	0.98	0.67	0.67	0.92	0.97
SAE	0.79	1.00	0.99	0.93	0.91	0.99	0.98	0.76	0.68	0.97	0.98
CNN	0.78	1.00	0.99	1.00	0.96	0.99	0.98	0.97	0.99	1.00	0.99
GRU	1.00	1.00	0.99	0.98	0.81	0.99	0.98	0.94	0.97	0.99	0.99
DBN	0.80	1.00	1.00	0.93	0.91	0.96	0.99	0.81	0.71	1.00	0.98
ESN	0.80	1.00	1.00	0.93	0.91	0.97	0.98	0.87	0.85	0.95	0.98
SSA-ESN	0.95	1.00	1.00	1.00	0.96	0.99	0.99	0.97	0.97	0.99	0.98
Ours	0.99	1.00	1.00	1.00	0.99	0.99	0.99	0.99	0.99	0.99	0.99

Table 10 The class F-measure of different ML algorithms on the Moore dataset.

Method	WWW	Mail	F-D	F-P	F-C	SERV	DB	P2P	ATT	MULT	INT	Games
SVM	0.93	0.89	0.92	0.82	0.74	0.94	0.73	0.50	0.81	0.83	0.00	0.00
SAE	0.95	0.94	0.99	0.98	0.86	0.94	0.89	0.75	0.88	0.86	0.62	0.00
CNN	0.97	0.97	0.99	0.94	0.90	0.95	0.94	0.82	0.91	0.86	0.56	0.67
GRU	0.97	0.97	0.99	0.97	0.90	0.97	0.96	0.84	0.90	0.87	0.53	0.00
DBN	0.96	0.92	0.97	0.95	0.83	0.95	0.84	0.74	0.86	0.84	0.33	0.00
ESN	0.96	0.92	0.95	0.90	0.88	0.94	0.94	0.80	0.88	0.86	0.70	0.00
SSA-ESN	0.99	0.97	0.99	0.97	0.97	0.99	0.99	0.94	0.95	0.93	0.90	0.67
Ours	1.00	0.98	1.00	0.97	0.98	0.99	0.99	0.95	0.97	0.93	0.95	0.67

Table 11 The class F-measure of different ML algorithms on the NISM dataset.

Method	DNS	Ftp	Http	Telent	Lime	Local forwarding	Remote forwarding	Scp	Sftp	Shell	X11
SVM	0.80	0.97	0.97	0.95	0.83	0.99	0.99	0.67	0.68	0.91	0.95
SAE	0.84	0.97	0.99	0.93	0.86	0.99	0.99	0.73	0.72	0.95	0.99
CNN	0.86	1.00	1.00	1.00	0.88	0.99	0.99	0.98	0.97	0.98	0.99
GRU	0.92	0.99	0.99	0.97	0.90	0.99	0.99	0.96	0.96	0.98	0.99
DBN	0.85	0.97	0.99	0.96	0.86	0.98	0.99	0.76	0.76	0.97	0.99
ESN	0.80	0.97	0.97	0.92	0.88	0.99	0.99	0.89	0.85	0.95	0.99
SSA-ESN	0.98	0.99	1.00	0.97	0.98	0.99	0.99	0.98	0.99	0.99	0.99
Ours	0.99	1.00	1.00	1.00	1.00	0.99	0.99	0.99	0.99	0.99	0.99

converges faster during the iteration process compared with PSO, FOA, SSA, GA, and GWO.

It can be seen from Tables 6–11 and Fig. 7 that the proposed scheme has obvious advantages over traditional ML methods on the two datasets in the class accuracy, class recall rate, class F-measure and overall accuracy. The reasons for the better classification performance of the proposed scheme are as follows: (1) The ESN processes information by simulating the thinking mode of human brain and has the characteristics such as self-organization, self-learning and self-adaptation. It adopts the reservoir composed of
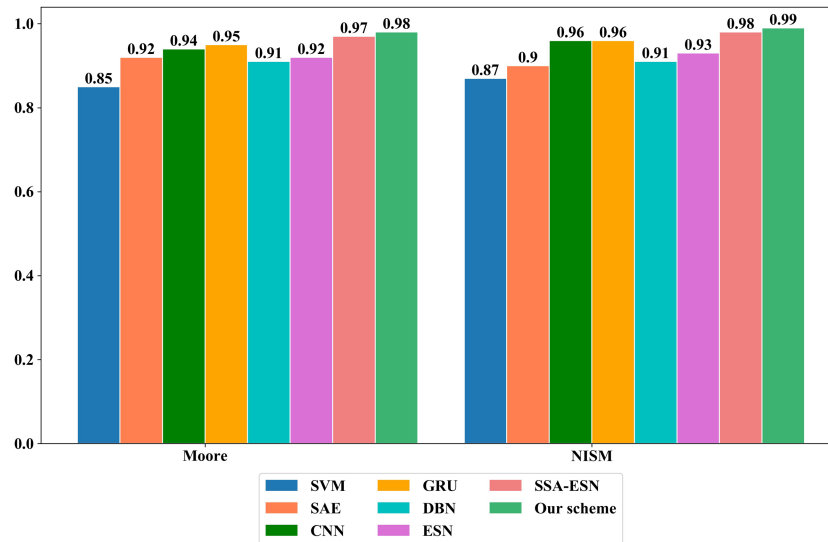
Figure 7 The overall accuracy of different machine learning algorithms on the Moore and NISM datasets.

Table 12 The training time and testing time of each algorithm.

Algorithms	Moore dataset		NISM dataset	
	Training time(s)	Testing time(s)	Training time(s)	Testing time(s)
SVM	35.613	0.2031	89.7042	0.9505
SAE	60.437	1.3562	113.489	2.369
CNN	38.1249	0.2504	56.3493	1.3795
GRU	31.8204	0.4143	52.2412	0.9124
DBN	33.8685	0.4361	52.1904	0.9579
ESN	9.892	0.1998	19.593	0.8237
SSA-ESN	250.078	0.4036	372.6851	0.9601
Ours	233.441	0.3983	345.7114	0.9582

sparsely connected neurons as the hidden layer to perform high-dimensional and non-linear representation of the input data. It only trains the weights from the reservoir to the output layer, which simplifies the training process. (2) The Tent mapping with reverse learning, polynomial operator and dynamic mutation strategy are introduced to improve the SSA, which improves global exploration ability of the algorithm and avoids the algorithm from falling into the local optimum. The ASSA is then used to automatically optimize the hyperparameters of the ESN and can accurately find the optimal hyperparameters of the ESN.

It can be seen from Table 12, due to use ASSA to optimize the hyperparameters of ESN, the training time is longer than other approaches. The trained model is tested on the testing set. The testing time of our method and other approaches is not much different. Especially

when the testing time is averaged to each sample, this difference is very small. In addition, our method has obvious advantages over other approaches in terms of per-class metrics and overall accuracy. Therefore, the gap between the classification time of our method and other methods is acceptable.

CONCLUSION

We propose a classification method of Internet traffic based on ESN. Firstly, Tent mapping with reverse learning, polynomial operator and dynamic mutation strategy are introduced to improve the SSA. Then, the advanced SSA is used to optimize the hyperparameters of the ESN such as the size of the reservoir and spectral radius. Finally, the optimized ESN is adopted to classify network traffic. We evaluate the performance of the proposed scheme on Moore and NISM datasets, and perform comparison experiments with SVM, SAE, CNN, GRU and DBN algorithms in terms of per-class metrics and overall accuracy, respectively. Experimental results show that our method has advantages in multiple evaluation metrics compared with these traditional ML algorithms and effectively improves the accuracy of network traffic classification. Our method needs further experimental verification before it can be deployed in the Internet environment. In the further, how to improve the real-time performance of the network traffic classification method still needs further research.

Funding
This work was supported by the Talent Project of Shandong Women's University under Grant 2020RCYJ21, 2018GSPGJ08, 2018RC34061, the National Science Foundation of China under Grant 62006143, and the National Science Foundation of Shandong Province (ZR2020MF152). There was no additional external funding received for this study. The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Grant Disclosures
The following grant information was disclosed by the authors:
Talent Project of Shandong Women's University: 2020RCYJ21, 2018GSPGJ08, 2018RC34061.
National Science Foundation of China: 62006143.
National Science Foundation of Shandong Province: ZR2020MF152.

Competing Interests
The authors declare there are no competing interests.

Author Contributions

- Meijia Zhang conceived and designed the experiments, performed the experiments, performed the computation work, prepared figures and/or tables, authored or reviewed drafts of the paper, and approved the final draft.

- Wenwen Sun performed the experiments, performed the computation work, prepared figures and/or tables, authored or reviewed drafts of the paper, and approved the final draft.
- Jie Tian analyzed the data, prepared figures and/or tables, and approved the final draft.
- Xiyuan Zheng analyzed the data, prepared figures and/or tables, and approved the final draft.
- Shaopeng Guan conceived and designed the experiments, authored or reviewed drafts of the paper, and approved the final draft.

Data Availability

The code is available at GitHub: https://github.com/melody213/network.

The third-party data is available at:

- Moore dataset: https://www.cl.cam.ac.uk/research/srg/netos/projects/archive/nprobe/data/papers/sigmetrics/index.html

- NISM dataset: https://projects.cs.dal.ca/projectx/Download.html

REFERENCES

Abualigah L, Shehab M, Alshinwan M, Alabool H. 2020. Salp swarm algorithm: a comprehensive survey. *Neural Computing and Applications* **32**(**15**):11195–11215 DOI 10.1007/s00521-019-04629-4.

Aceto G, Dainotti A, De Donato W, Pescape A. 2010. PortLoad: taking the best of two worlds in traffic classification. In: *INFOCOM IEEE conference on computer communications workshops*. Piscataway: IEEE, 1–5.

Arora S, An P. 2019. Chaotic grasshopper optimization algorithm for global optimization. *Neural Computing and Applications* **31**(**8**):4385–4405 DOI 10.1007/s00521-018-3343-2.

Ateya AA, Muthanna A, Vybornova A, Algarni AD, Abuarqoub A, Koucheryavy Y, Koucheryavy A. 2019. Chaotic salp swarm algorithm for SDN multicontroller networks. *Engineering Science and Technology, an International Journal* **22**(**4**):1001–1012 DOI 10.1016/j.jestch.2018.12.015.

Callado A, Kamienski C, Szabo G, Gero BP, Kelner J, Fernandes S, Sadok D. 2009. A survey on internet traffic identification. *IEEE Communications Surveys & Tutorials* **11**(**3**):37–52 DOI 10.1109/SURV.2009.090304.

Cao J, Fang Z, Qu G, Sun H, Zhang D. 2017. An accurate traffic classification model based on support vector machines. *International Journal of Network Management* **27**(**1**):e1962.

Chouikhi N, Ammar B, Rokbani N, Alimi AM. 2017. PSO-based analysis of Echo State Network parameters for time series forecasting. *Applied Soft Computing* **55**:211–225 DOI 10.1016/j.asoc.2017.01.049.

Demertzis K, Iliadis L. 2016. Ladon: a cyber-threat bio-inspired intelligence management system. *Journal of Applied Mathematics & Bioinformatics* **6**(**3**):45–64.

Dias KL, Pongelupe MA, Caminhas WM, Errico L. 2019. An innovative approach for real-time network traffic classification. *Computer Networks* **158**:143–157 DOI 10.1016/j.comnet.2019.04.004.

Duan H, Wang X. 2015. Echo state networks with orthogonal pigeon-inspired optimization for image restoration. *IEEE Transactions on Neural Networks and Learning Systems* **27(11)**:2413–2425.

Este A, Gringoli F, Salgarelli L. 2009. Support vector machines for TCP traffic classification. *Computer Networks* **53(14)**:2476–2490 DOI 10.1016/j.comnet.2009.05.003.

Finsterbusch M, Richter C, Rocha E, Muller J, Hanssgen K. 2013. A survey of payload-based traffic classification approaches. *IEEE Communications Surveys & Tutorials* **16(2)**:1135–1156.

Grigoryeva L, Ortega J. 2018. Echo state networks are universal. *Neural Networks* **108**:495–508 DOI 10.1016/j.neunet.2018.08.025.

Han M, Xu M. 2017. Laplacian echo state network for multivariate time series prediction. *IEEE Transactions on Neural Networks and Learning Systems* **29(1)**:238–244.

Hu H, Wang L, Lv S. 2020. Forecasting energy consumption and wind power generation using deep echo state network. *Renewable Energy* **154**:598–613 DOI 10.1016/j.renene.2020.03.042.

Hu H, Wang L, Tao R. 2021. Wind speed forecasting based on variational mode decomposition and improved echo state network. *Renewable Energy* **164**:729–751 DOI 10.1016/j.renene.2020.09.109.

Jing N, Yang M, Cheng S, Dong Q, Xiong H. 2011. An efficient SVM-based method for multi-class network traffic classification. In: *30th IEEE International performance computing and communications conference.* Piscataway: IEEE, 1–8.

Kohli M, Arora S. 2018. Chaotic grey wolf optimization algorithm for constrained optimization problems. *Journal of Computational Design and Engineering* **5(4)**:458–472 DOI 10.1016/j.jcde.2017.02.005.

Li C, Dong K, Niu K, Zhang Z. 2021. Mobile service traffic classification based on joint deep learning with attention mechanism. *IEEE Access* **9**:74729–74738 DOI 10.1109/ACCESS.2021.3081504.

Li W, Moore AW. 2007. A machine learning approach for efficient traffic classification. In: *15th International symposium on modeling, analysis, and simulation of computer and telecommunication systems.* 310–317.

Li C, Zhang N, Lai X, Zhou J, Xu Y. 2017. Design of a fractional-order PID controller for a pumped storage unit using a gravitational search algorithm based on the Cauchy and Gaussian mutation. *Information Sciences* **396**:162–181 DOI 10.1016/j.ins.2017.02.026.

Liu W, Wang Z, Liu X, Zeng N, Liu Y, Alsaadi FE. 2017. A survey of deep neural network architectures and their applications. *Neurocomputing* **234**:11–26 DOI 10.1016/j.neucom.2016.12.038.

Liu X, You J, Wu Y, Li T, Li L, Zhang Z, Ge J. 2020. Attention-based bidirectional GRU networks for efficient HTTPS traffic classification. *Information Sciences* **541**:297–315 DOI 10.1016/j.ins.2020.05.035.

Long J, Zhang S, Li C. 2019. Evolving deep echo state networks for intelligent fault diagnosis. *IEEE Transactions on Industrial Informatics* **16(7)**:4928–4937.

Lopez-Martin M, Carro B, Sanchez-Esguevillas A, Lloret J. 2017. Network traffic classifier with convolutional and recurrent neural networks for Internet of Things. *IEEE Access* **5**:18042–18050 DOI 10.1109/ACCESS.2017.2747560.

Lotfollahi M, Siavoshani MJ, Zade RSH, Saberian M. 2020. Deep packet: a novel approach for encrypted traffic classification using deep learning. *Soft Computing* **24(3)**:1999–2012 DOI 10.1007/s00500-019-04030-2.

Ma C, Du X, Cao L. 2020. Improved KNN algorithm for fine-grained classification of encrypted network flow. *Electronics* **9(2)**:1–28 DOI 10.3390/electronics9020324.

Mirjalili S, Gandomi AH, Mirjalili SZ, Saremi S, Faris H, Mirjalili SM. 2017. Salp Swarm Algorithm: a bio-inspired optimizer for engineering design problems. *Advances in Engineering Software* **114**:163–191 DOI 10.1016/j.advengsoft.2017.07.002.

Moore AW, Zuev D. 2005. Internet traffic classification using bayesian analysis techniques. In: *Proceedings of the 2005 ACM SIGMETRICS international conference on measurement and modeling of computer systems.* 50–60.

Nguyen TTT, Armitage G. 2008. A survey of techniques for internet traffic classification using machine learning. *IEEE Communications Surveys & Tutorials* **10(4)**:56–76 DOI 10.1109/SURV.2008.080406.

Pacheco F, Exposito E, Gineste M, Baudoin C, Aguilar J. 2018. Towards the deployment of machine learning solutions in network traffic classification: a systematic survey. *IEEE Communications Surveys & Tutorials* **21(2)**:1988–2014.

Qi H, Wang J, Li W, Wang Y, Qiu T. 2020. A blockchain-driven IIoT traffic classification service for edge computing. *IEEE Internet of Things Journal* **8(4)**:2124–2134.

Ren X, Gu H, Wei W. 2021. Tree-RNN: Tree structural recurrent neural network for network traffic classification. *Expert Systems with Applications* **167**:114363 DOI 10.1016/j.eswa.2020.114363.

Rezaei S, Liu X. 2019. Deep learning for encrypted traffic classification: an overview. *IEEE Communications Magazine* **57(5)**:76–81.

Rizk-Allah RM, Hassanien AE, Elhoseny M, Gunasekaran M. 2019. A new binary salp swarm algorithm: development and application for optimization tasks. *Neural Computing and Applications* **31(5)**:1641–1663 DOI 10.1007/s00521-018-3613-z.

Sun G, Chen T, Su Y, Li C. 2018. Internet traffic classification based on incremental support vector machines. *Mobile Networks and Applications* **23(4)**:789–796 DOI 10.1007/s11036-018-0999-x.

Tian Z. 2020. Echo state network based on improved fruit fly optimization algorithm for chaotic time series prediction. *Journal of Ambient Intelligence and Humanized Computing* **2020(10)**:1–20 DOI 10.1007/s12652-020-01920-4.

Tong D, Qu YR, Prasanna VK. 2017. Accelerating decision tree based traffic classification on FPGA and multicore platforms. *IEEE Transactions on Parallel and Distributed Systems* **28(11)**:3046–3059 DOI 10.1109/TPDS.2017.2714661.

Wang L, Hu H, Ai X, Liu H. 2018. Effective electricity energy consumption forecasting using echo state network improved by differential evolution algorithm. *Energy* **153**:801–815 DOI 10.1016/j.energy.2018.04.078.

Wang W, Zhu M, Zeng X, Ye X, Sheng Y. 2017. Malware traffic classification using convolutional neural network for representation learning. In: *International Conference on Information Networking (ICOIN)*. 712–717.

Yang Y, Kang C, Gou G, Li Z, Xiong G. 2018. TLS/SSL encrypted traffic classification with autoencoder and convolutional neural network. In: *IEEE 20th international conference on high performance computing and communications*. Piscataway: IEEE, 362–369.

Yuan R, Li Z, Guan X, Xu L. 2010. An SVM-based machine learning method for accurate internet traffic classification. *Information Systems Frontiers* **12(2)**:149–156 DOI 10.1007/s10796-008-9131-2.

Zeng Y, Gu H, Wei W, Guo Y. 2019. Deep-full-range: a deep learning based network encrypted traffic classification and intrusion detection framework. *IEEE Access. Piscataway, IEEE* **7**:45182–45190 DOI 10.1109/ACCESS.2019.2908225.

Zhang J, Chen C, Xiang Y, Zhou W, Xiang Y. 2012. Internet traffic classification by aggregating correlated naive bayes predictions. *IEEE Transactions on Information Forensics and Security* **8(1)**:5–15.

Zhang J, Chen X, Xiang Y, Zhou W, Wu J. 2014. Robust network traffic classification. *IEEE/ACM Transactions on Networking* **23(4)**:1257–1270.

Zhang Q, Qian H, Chen Y, Lei D. 2020. A short-term traffic forecasting model based on echo state network optimized by improved fruit fly optimization algorithm. *Neurocomputing* **416**:117–124 DOI 10.1016/j.neucom.2019.02.062.

Zhang S, Sun Z, Wang M, Long J, Bai Y, Li C. 2019. Deep fuzzy echo state networks for machinery fault diagnosis. *IEEE Transactions on Fuzzy Systems* **28(7)**:1205–1218.

Zhong S, Xie X, Lin L, Wang F. 2017. Genetic algorithm optimized double-reservoir echo state network for multi-regime time series prediction. *Neurocomputing* **238**:191–204 DOI 10.1016/j.neucom.2017.01.053.