A feature selection algorithm for PNN optimized by binary PSO and applied to smart city intrusion detection system

Yanyan Shi[1], Gaoyuan Liu[1*], Boxiong Yang[1*], Yong Chen[2], Zhenbao Liang[2]

[1]*University of Sanya, Sanya, No. 191, Xueyuan Road, Jiyang District, Sanya City, Hainan Province, China*
[2] *Geely Automotive Research Institute (Ningbo) Co., Ltd, China*

Article Information

Received: 21-11-2024
Revised: 28-11-2024
Published: 05-12-2024

Keywords
Intrusion Detection System; smart city; binary particle swarm optimization, pnn

****Correspondence Email:***
boxiongyang@sanyau.edu.cn

Abstract

In the smart city construction and development process, network security is a vital link that must be addressed. As a critical technology to ensure the security of smart cities, intrusion detection systems play multiple roles, such as real-time monitoring, threat identification, and event response. By deploying and optimizing efficient intrusion detection mechanisms, smart cities can effectively resist various network attacks and ensure the safety and stability of urban operations. Therefore, this paper proposes a PNN feature selection model based on binary particle swarm algorithm optimization (BiPSO) to select features for network traffic data and remove a large amount of redundant information; then, a variety of classifiers (random forest, KNN, Adaboost, BPNN) are used to classify the data after feature selection. We use NSL-KDD for experiments. The experimental results show that the BiPSO-PNN proposed in this paper not only achieves accurate selection of data set features but also guarantees high accuracy on other classifiers, which has vital practical significance.

1. Introduction

With the continuous deployment of sensor networks, various entities are connected, realizing information transmission between objects and efficient interconnection, and finally developing into the Internet of Things. The Internet of Things technology provides many unprecedented opportunities for human interaction, significantly promotes data sharing, facilitates daily life, and provides a universal platform for sharing data for the environment, society and industry(Bhatt et al., 2024), which can effectively assist the construction of smart cities. Since smart cities rely on the construction of the Internet of Things, smart cities can also be divided into three parts: perception layer, network layer, and application layer.

(1) Perception layer: Various terminal devices and communication nodes are the core of the Internet of Things' perception layer. The perception layer can collect information from the physical world and send it to the network layer after being converted into a unified format by the data collection terminal.

(2) Network layer: The main function of the network layer is to connect the perception layer and the application layer. After the perception layer obtains information, it relies on the network layer for transmission, including broadband wireless networks, optical fiber networks, cellular networks, and

various dedicated networks. The network layer comprises various wireless/wired gateways, access networks, and core networks to achieve two-way transmission of perception data and control data.

(3) Application layer: The interface between the Internet of Things and users. It can provide corresponding management and operation platforms for different users and industries and combine with the professional knowledge and business models of different industries to achieve more accurate and sophisticated intelligent information management.

Smart cities cover smart city complexes, smart government affairs, smart education, smart grids, smart transportation, and other aspects as shown in Fig.1 . The number and scope of its smart networked devices have greatly increased compared to traditional cities, which has attracted illegal intruders to use these devices to spread malicious information. Intrusion behavior refers to destroying the target resources' confidentiality and integrity. Intruders may access and eavesdrop on personal or smart city core server information, causing direct economic losses or affecting the operation of the entire smart city system. There are various types of intrusion detection in smart cities, which can be mainly classified according to four methods: detection method, deployment location, response method, and data source. 1) Based on monitoring methods, intrusion monitoring mainly includes three types: feature (signature) detection(Masdari et al., 2020), which identifies intrusion behavior by using the features or signatures of known attacks, such as specific data packet structures and known malicious codes; based on anomaly detection(Hajj et al., 2021), by establishing a model of the normal behavior of the system, when activities that deviate from normal behavior are detected, it is judged that there may be an intrusion; and based on state analysis(Zhang et al., 2017), by monitoring changes in system status, such as file integrity and configuration changes, to detect unauthorized modifications or access; 2) Intrusion detection based on deployment location can be divided into Network Intrusion Detection System (NIDS)(Tu et al., 2023) and Host Intrusion Detection System (HIDS)6[(Martins et al., 2022)], which are deployed at key nodes at the edge of the network or on servers and terminal devices respectively, analyze data packets and monitor intrusion behaviors at the network level; 3) Intrusion monitoring based on response methods is divided into active(Naeem et al., 2023) and passive intrusion detection(Gui et al., 2023). The difference between the two is whether to alarm and isolate the user after detecting illegal intrusion; 4) Intrusion monitoring based on data sources is classified according to log analysis(Han et al., 2021) and traffic analysis(Zhao et al., 2023). Log analysis mainly detects abnormal activities by collecting and analyzing log information of systems, network devices, and applications. Traffic analysis to monitors network traffic, analyzes communication patterns and data content, and identifies suspicious behaviors. At present, intrusion detection technology represented by machine learning and artificial intelligence has gradually matured. By using AI technology, the patterns of normal and abnormal behaviors can be automatically learned to improve detection accuracy and real-time performance (Dina et al., 2021). For example, (Saba et al., 2022) proposed a hybrid deep learning network intrusion monitoring system, which achieved efficient monitoring performance by combining CNN and BiLSTM. Ajdani and Ghaffary designed an improved support vector machine algorithm considering time, user information, and scalability. The experimental results on the VIRUS TOTAL dataset showed that this method can achieve a recognition rate of 97%.

Fig. 1 Smart city construction

Optimization is often used to solve complex engineering problems (Pan et al., 2023). The initial hyperparameters often affect the performance of machine learning or deep learning. Many scholars have used optimization algorithms to optimize machine learning or deep learning hyperparameters to improve intrusion detection performance in intrusion monitoring. For example, (Keserwani et al., 2021) proposed a random forest algorithm optimized by GWO hybrid PSO, which achieved excellent performance on multiple intrusion detection data sets; (Pan et al., 2021) proposed a lightweight wireless sensor network intelligent intrusion detection model, which optimized KNN using an improved sine-cosine algorithm (PM-CSCA) and achieved satisfactory results in simulation tests on the NSL-KDD and UNSW-NB15 data sets. In addition, since the amount of intrusion detection data is large and contains a large number of irrelevant or redundant features, it will affect the accuracy of intrusion detection. To solve this problem, (Kunhare et al., 2020) used PSO to apply to the selective features of the NSL-KDD data set and classified the data after feature selection using different classifiers, and still achieved an accuracy of 99.26%; (Alawad et al., 2023)proposed an improved binary great white shark algorithm to conduct feature selection experiments on 12 public real-world IDS, proving that its performance is very significant. It can be seen that it is necessary and effective to use feature selection to process traffic data. Therefore, this paper proposes a binary PSO feature selection algorithm, which optimizes the probabilistic neural network (PNN) input feature subset through BiPSO to achieve feature selection of intrusion detection data. In addition, we also tested the data after feature selection with a variety of mainstream classifiers. The experiment shows that the proposed BiPSO-PNN intrusion detection feature selection model can still provide good classification performance for subsequent classifiers after removing many features.

The main contributions of this paper are as follows:

- A PNN feature monitoring model optimized by binary particle swarm algorithm is proposed;
- Feature selection of the NSL-KDD dataset is realized, and a large number of redundant features are removed;
- The data after feature selection is used in machine learning and deep learning models such as Random forest, KNN, AdaBoost, BPNN, etc., achieving good classification results.

2. Related works

2.1 Binary Particle Swarm Optimization (BiPSO)

The standard particle swarm optimization algorithm is a global optimization algorithm based on swarm intelligence derived from the simulation of bird's flock predation behavior. The algorithm first initializes a set of random solutions, as shown in the Eq. (1):

$$X_{i,j} = lb + (ub - lb) \times rand(0,1) \tag{1}$$

where $X_{i,j}$ is the position of individual i in dimension j, lb is the lower limit, and ub is the upper limit. Then, the speed and position are updated according to the Eq. (2) and Eq. (3):

$$v_{id}(t+1) = \omega \cdot v_{id}(t) + c_1 \cdot r\big(PBest_{id} - x_{id}(t)\big) + c_2 r(GBest_d - x_{id}(t)) \tag{2}$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1) \tag{3}$$

where, ω is the inertia weight, c_1 and c_2 are acceleration coefficients, and $r \in [0,1]$ is a random number. $x_{id}(t)$ represents the current position of the particle, $PBest_{id}(t)$ represents the historical optimal position of the particle, and $GBest_d(t)$ represents the global optimal position of the current population. In the binary particle swarm algorithm, the particle position is represented by a vector consisting of 0 or 1. In this paper, we use sigmoid to map it to the [0,1] interval, as shown in Eq. (4):

$$S(v_{id}) = \frac{1}{1+e^{-v_{id}}} \tag{4}$$

The position update is shown in Eq. (5):

$$x_{id}(t+1) = \begin{cases} 1 & , \ if\ S\big(v_{id}(t+1)\big) > r \\ 0, & else \end{cases} \tag{5}$$

where, $r \in [0,1]$ is a random number.

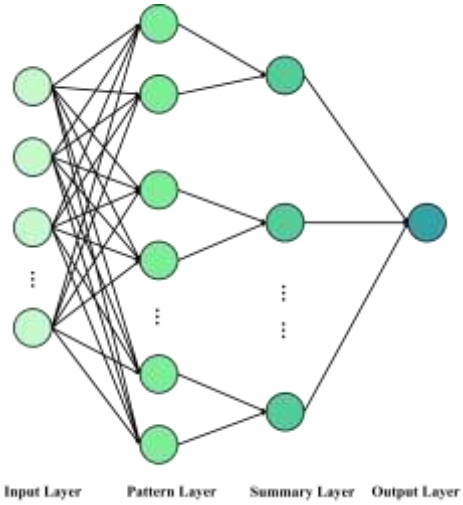

Fig. 2 Probabilistic neural network

2.2 Probabilistic Neural Networks (PNN)

Probabilistic neural network (PNN)(Nashed et al., 2022)is a feedforward neural network based on Bayesian decision theory and kernel density estimation. It is widely used in medical diagnosis, image recognition, fault detection and other fields due to its advantages such as fast training speed, high classification accuracy and low requirements on data distribution. PNN includes input layer, pattern layer, summary layer and output layer, as shown in the Fig. 2. Its core idea is to use the kernel density estimation method to estimate the probability density function of each category, and then select the category with the highest posterior probability as the prediction result according to the Bayesian criterion. For classification problems, as shown in Eq. (6):

$$C(x) = \arg\max_k P(C_k|x) \tag{6}$$

where x is the input vector, C_k is the probability of the category, and $P(C_k|x)$ is the posterior probability, as shown in Eq. (7):

$$P(C_k|x) = \frac{1}{N_k}\sum_{i=1}^{N_k} \phi(x; x_i^k, \sigma) \tag{7}$$

where N_k is the number of samples in category C_k, x_i^k is the i-th sample of category C_k, and $\phi(x; x_i^k, \sigma)$ is the Gaussian kernel function, as shown in Eq. (8), which is used to calculate the similarity between the input vector calculated by each neuron and its corresponding training sample.

$$\phi(x; x_i^k, \sigma) = \frac{1}{(2\pi)^{d/2}\sigma^d} \exp\left(-\frac{\left\|x - x_i^k\right\|^2}{2\sigma^2}\right) \tag{8}$$

where d is the feature dimension, σ is the variance, which is used to control the width of the Gaussian function. The sum of the similarities of each category will be calculated in the summary layer, as shown in Eq. (9):

$$S_k(x) = \sum_{i=1}^{N_k} \phi(x; x_i^k, \sigma) \tag{9}$$

where $S_k(x)$ represents the cumulative similarity of category C_k.

3. PNN feature selection with BiPSO optimization

In this section, the author presents brief conclusions from the results of research with suggestions for advanced BiPSO (Binary Particle Swarm Optimization) is mainly used for feature selection, that is, to optimize the classification performance of PNN by selecting the optimal feature subset. Specifically, BiPSO optimizes the following aspects:

(1) Initialization feature selection

BiPSO first generates a set of initialization populations. The position vector $X(i,:)$ of each particle is a binary vector, where 1 indicates that the feature is selected and 0 indicates that it is not selected. It is used to search for the optimal feature subset;

(2) One-hot encoding of the data set

Get all unique category labels, create a zero matrix Class of size [number of training samples x number of categories], traverse each training sample, and set the position of the corresponding category to 1;

(3) Fitness evaluation

Use the initialized population or $X(i,:)$ in the BiPSO algorithm iteration as the initial value of PNN feature selection, then train the one-hot encoded data, test the test set, and return the fitness value, as shown in formula (10):

$$finess = \alpha \times (1 - accuracy) + \beta \times (\frac{R}{N}) \tag{10}$$

where α and β are used to balance the importance of classification error and feature selection rate. In this paper, $\alpha = 0.3, \beta = 0.7, accuracy = (TP + TN)/(TP + TN + FP + FN)$, $\beta \times \left(\frac{R}{N}\right)$ is the feature selection rate, and the smaller $\left(\frac{R}{N}\right)$ is, the fewer features are selected.

(4) Until the iteration ends, the optimal value X_{gbest} is returned as the initial parameter for the best feature selection of PNN.

4. Experiment and analysis

4.1 Binary Particle Swarm Optimization (BiPSO)

In the field of network security, the research and development of intrusion detection systems (IDS) is crucial. To evaluate and compare the performance of different IDS, researchers need high-quality, representative datasets. The NSL-KDD dataset is an improvement and optimization of the classic KDD Cup 1999 dataset. It retains 41 features of the KDD Cup 1999 dataset, including four major attack types: DoS, R2L, U2R, and Probe, with a total of 805,049 instances. This paper uses T-SNE to visualize the data, as shown in Fig. 3.

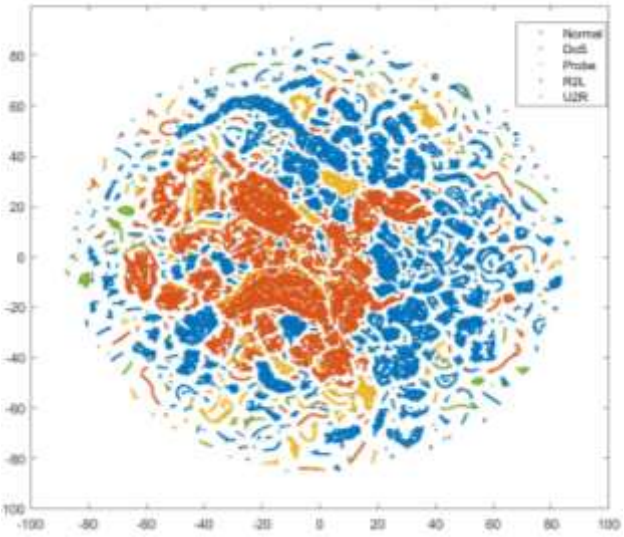

Fig. 3 NSL-KDD dataset visualized by T-SNE dimensionality reduction

4.2 Feature Selection Experiment and Analysis

First, the data set is preprocessed to remove irrelevant features such as ID and IP, and the string type features are mapped into numerical data so that PNN can be trained normally. Then, 10,000 samples are randomly selected and divided into train set and test set, where the training set: test set = 7:3, and normalized. The experimental results are shown in Fig. 4 and Fig. 5.

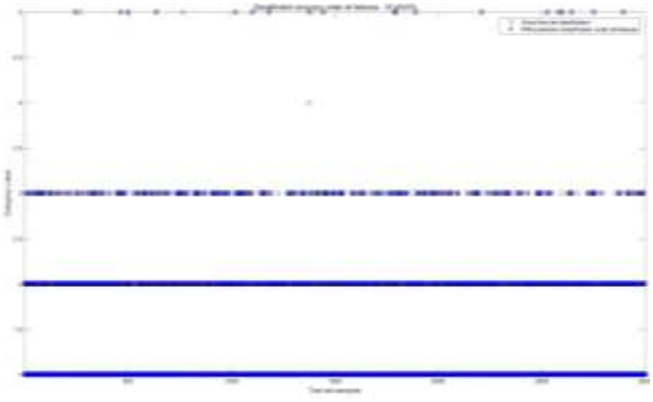

Fig. 4 Results before PNN feature selection

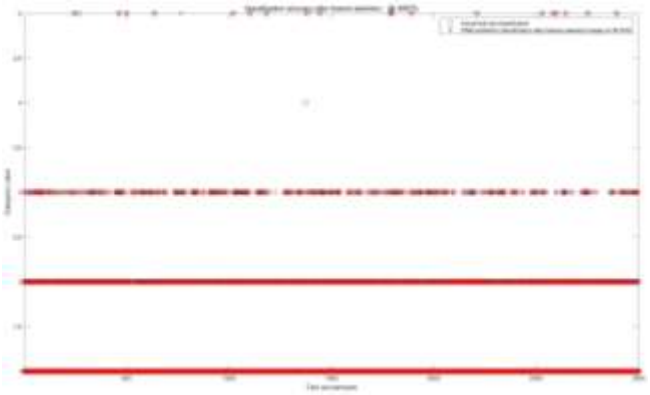

Fig. 5 BiPSO optimizes the results of PNN feature selection

It can be clearly seen from Fig. 4 and Fig. 5 that the features optimized using BiPSO-PNN have better classification effects. In addition, Fig. 6 also verifies this point. After feature selection, only the 5 most effective features are retained in the data set, and the classification accuracy is improved.

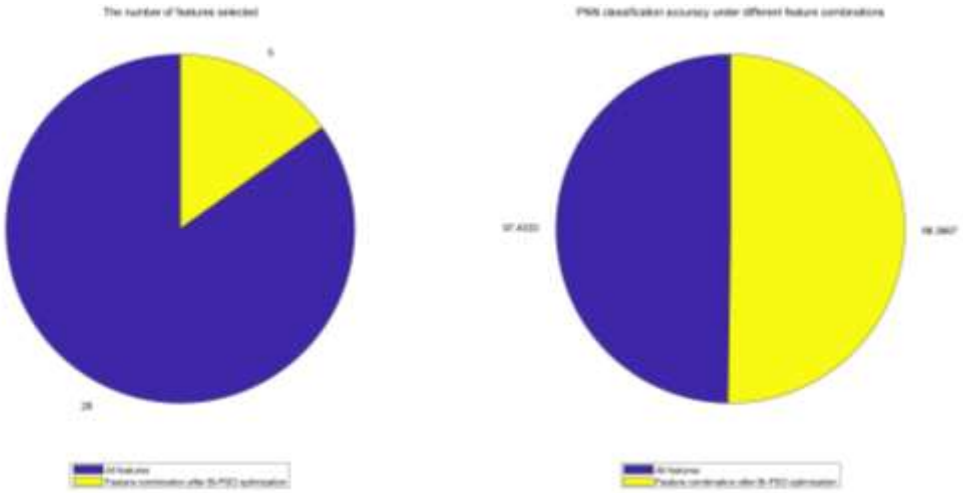

Fig. 6 Comprehensive comparison chart

4.3 Intrusion Detection Classifier

To verify the validity of the data after feature selection, we will further use other classifiers to test the dataset after feature selection.

4.3.1 Introduction to Classifiers

(1) Random Forest

Random Forest is an ensemble learning method that improves the accuracy and stability of the model by building multiple decision trees and combining their prediction results. Its core ideas include "bagging" and feature randomness, that is, randomly extracting multiple sub-sample sets from the original dataset with replacement, each sub-sample set is used to train a decision tree, and for classification tasks, the majority voting method is used.

(2) KNN

KNN is an instance-based supervised learning algorithm suitable for classification. Its core idea is to calculate the similarity between samples by measuring the distance between the new sample and all samples in the

training set. The Euclidean distance, Manhattan distance or other distance measurement methods are often used to calculate the similarity between samples, select the K nearest neighbors, and make predictions based on the labels of these neighbors.

(3) AdaBoost

AdaBoost is a boosting method that builds a strong classifier by combining multiple weak classifiers (usually decision tree stumps). The core idea is to gradually adjust the sample weights so that the subsequent classifiers pay more attention to the samples that were previously misclassified. In each round, a weak classifier is trained, and the sample weights are adjusted according to the error rate of the previous round to increase the weights of the misclassified samples. The final strong classifier combines the prediction results of all weak classifiers through weighted voting to achieve classification.

(4) BPNN

BPNN is a multi-layer feedforward neural network that adjusts the network weights through the back propagation algorithm to minimize the prediction error. Its core includes three steps: forward propagation, error calculation, and weight update. It usually includes an input layer, one or more hidden layers, and an output layer. Each layer consists of multiple neurons. In this article, softmax is used as the activation function to achieve multi-classification effects.

4.3.2 Experimental results

We use the dataset extracted by BPSO-PNN for verification. First, we randomly sample, divide and normalize the dataset after feature extraction, and then pass it into different classifiers for experiments. To better evaluate the intrusion detection classification effect, we also use recall rate, precision rate and F1-score for evaluation, as shown in Eq. (11)-Eq. (13).

$$Pre = TP \,/\, (TP + FP) \tag{11}$$

$$Recell = TP/TP + FN \tag{12}$$

$$F1 = \frac{2 \times Precision \times Recell}{Precision + Recell} \tag{13}$$

The experimental results of random forest, KNN, AdaBoost and BPNN are shown in the Table 1.

Table 1. Random Forest, KNN, AdaBoost, BPNN intrusion detection classification results

Model	**Evaluation Metrics**			
	ACC	***Pre***	***Recell***	***F1***
Random Forest	0.9890	0.9630	0.9391	0.9568
KNN	0.9633	0.6543	0.6110	0.6257
AdaBoost	0.9903	0.9876	0.9581	0.9719
BPNN	0.9393	0.5575	0.5362	0.5454

5. Conclusions

With the rapid development of information technology, the smart city concept has gradually become an important direction for the development of modern cities. Smart cities realize intelligent, refined, and efficient urban management by integrating advanced technologies such as the Internet of Things (IoT), big data, cloud computing, and artificial intelligence. However, the efficient operation of smart city depends on a large and complex information system, making it a potential target for network attacks. Therefore, the importance of intrusion detection system (IDS) in smart city is more prominent. This paper proposes a PNN feature selection algorithm based on binary particle swarm optimization. First, feature selection is performed on the NSL-KDD dataset to remove redundant data, and the detection accuracy is improved after removing redundant data. After that, the dataset is saved and verified by classifiers such as random forest, KNN, AdaBoost, and BPNN

after feature selection. The experiment shows that the data after feature selection can still maintain 93%~99% classification performance on other classifiers. Due to the sharp reduction in the number of features, the detection speed is greatly accelerated. In the future, we will use data-driven optimization methods to propose better intrusion detection models.

6. References

Alawad, N. A., Abed-alguni, B. H., Al-Betar, M. A., & Jaradat, A. (2023). Binary improved white shark algorithm for intrusion detection systems. *Neural Computing and Applications*, *35*(26), 19427–19451. doi: 10.1007/s00521-023-08772-x

Bhatt, K., Agrawal, C., & Bisen, A. M. (2024). A Review on Emerging Applications of IoT and Sensor Technology for Industry 4.0. *Wireless Personal Communications*, *134*(4), 2371–2389. doi: 10.1007/s11277-024-11054-x

Dina, A. S., & Manivannan, D. (2021). Intrusion detection based on Machine Learning techniques in computer networks. In Internet of Things (Netherlands) (Vol. 16). Elsevier B.V. doi: 10.1016/j.iot.2021.100462

Gui, L., Yuan, W., & Xiao, F. (2023). CSI-based passive intrusion detection bound estimation in indoor NLoS scenario. *Fundamental Research*, *3*(6), 988–996. doi: 10.1016/j.fmre.2022.05.015

Hajj, S., El Sibai, R., Bou Abdo, J., Demerjian, J., Makhoul, A., & Guyeux, C. (2021). Anomaly-based intrusion detection systems: The requirements, methods, measurements, and datasets. *Transactions on Emerging Telecommunications Technologies*, *32*(4). doi: 10.1002/ett.4240

Han, S., Wu, Q., Zhang, H., Qin, B., Hu, J., Shi, X., Liu, L., & Yin, X. (2021). Log-Based Anomaly Detection with Robust Feature Extraction and Online Learning. *IEEE Transactions on Information Forensics and Security*, *16*, 2300–2311. doi: 10.1109/TIFS.2021.3053371

Keserwani, P. K., Govil, M. C., Pilli, E. S., & Govil, P. (2021). A smart anomaly-based intrusion detection system for the Internet of Things (IoT) network using GWO–PSO–RF model. *Journal of Reliable Intelligent Environments*, *7*(1), 3–21. doi: 10.1007/s40860-020-00126-x

Kunhare, N., Tiwari, R., & Dhar, J. (2046). *Particle swarm optimization and feature selection for intrusion detection system*. doi: 10.1007/s12046-020-1308-5S

Martins, I., Resende, J. S., Sousa, P. R., Silva, S., Antunes, L., & Gama, J. (2022). Host-based IDS: A review and open issues of an anomaly detection system in IoT. *Future Generation Computer Systems*, *133*, 95–113. doi: 10.1016/j.future.2022.03.001

Masdari, M., & Khezri, H. (2020). A survey and taxonomy of the fuzzy signature-based Intrusion Detection Systems. In Applied Soft Computing Journal (Vol. 92). Elsevier Ltd. doi: 10.1016/j.asoc.2020.106301

Naeem, F., Ali, M., & Kaddoum, G. (2023). Federated-Learning-Empowered Semi-Supervised Active Learning Framework for Intrusion Detection in ZSM. *IEEE Communications Magazine*, *61*(2), 88–94. doi: 10.1109/MCOM.001.2200533

Nashed, M. S., Mohamed, M. S., Shady, O. T., & Renno, J. (2022). Using probabilistic neural networks for modeling metal fatigue and random vibration in process pipework. *Fatigue and Fracture of Engineering Materials and Structures*, *45*(4), 1227–1242. doi: 10.1111/ffe.13660

Pan, J. S., Fan, F., Chu, S. C., Zhao, H. Q., & Liu, G. Y. (2021). A Lightweight Intelligent Intrusion Detection Model for Wireless Sensor Networks. *Security and Communication Networks*, *2021*. doi: 10.1155/2021/5540895

Pan, J. S., Hu, P., Snášel, V., & Chu, S. C. (2023). A survey on binary metaheuristic algorithms and their engineering applications. *Artificial Intelligence Review*, *56*(7), 6101–6167. doi: 10.1007/s10462-022-10328-9

Saba, T., Rehman, A., Sadad, T., Kolivand, H., & Bahaj, S. A. (2022). Anomaly-based intrusion detection system for IoT networks through deep learning model. *Computers and Electrical Engineering*, *99*. doi: 10.1016/j.compeleceng.2022.107810

Tu, S., Waqas, M., Badshah, A., Yin, M., & Abbas, G. (2023). Network Intrusion Detection System (NIDS) Based on Pseudo-Siamese Stacked Autoencoders in Fog Computing. *IEEE Transactions on Services Computing*, *16*(6), 4317–4327. doi: 10.1109/TSC.2023.3319953

Zhang, Z., Zhu, H., Luo, S., Xin, Y., & Liu, X. (2017). Intrusion Detection Based on State Context and Hierarchical Trust in Wireless Sensor Networks. *IEEE Access*, *5*, 12088–12102. doi: 10.1109/ACCESS.2017.2717387

Zhao, H., Li, Y., Fan, F., Liang, Z., Lei, R., & Guo, Y. (2023). Encrypted Traffic Classification Method Based on GSK and BiLSTM. *Proceedings - 2023 China Automation Congress, CAC 2023*, 2191–2196. doi: 10.1109/CAC59555.2023.10452083