Computers & Security 156 (2025) 104508

Contents lists available at ScienceDirect

Computers & Security

journal homepage: www.elsevier.com/locate/cose

Full length article

Light-weight slow-rate attack detection framework for resource-constrained Industrial Cyber–Physical Systems

Farzana Zahid [a,*], Matthew M.Y. Kuo [b], Roopak Sinha [c]

[a] School of Computing and Mathematical Sciences, University of Waikato (UoW), 100 Knighton Road, Hillcrest, Hamilton 3216, New Zealand
[b] School of Engineering, Computer and Mathematical Sciences, Auckland University of Technology, Auckland, New Zealand
[c] School of Information Technology, Deakin University, Melbourne, Australia

ARTICLE INFO

Dataset link: https://www.unb.ca/cic/datasets/ids-2018.html

Keywords:
Industrial Cyber–Physical System
Resource-constrained
Slow-rate attacks
Online Sequential-Extreme Learning Machine
PLC

ABSTRACT

Industrial Cyber–Physical Systems (ICPS) are heterogeneous computer systems interacting with physical processes in an industrial environment. The presence of numerous interconnected components poses significant security threats to ICPS. Slow-Rate Attacks (SRA), in which attackers attack a system constantly at low volumes, are difficult to detect for resource-constrained ICPS computers like programmable logic controllers (PLC). We propose an optimised light-weight active security framework for SRA detection based on Online Sequential Extreme Learning Machine (OSELM). We optimise the memory and space footprint of OSELM for deployment in resource-constrained ICPS. Additionally, a simple stratified k-fold cross training method improves the performance and accuracy of binary and multi-class SRA detection. Compared to existing methods, our technique requires less space and reduces attack detection time by at least 95%.

1. Introduction

In Low-rate Denial of Service (LDoS) attacks, attackers exploit system vulnerabilities through continuous low volume attacks (Drinić and Čiča, 2024; Zhijun et al., 2020). LDoS attacks differ from traditional Distributed Denial of Service (DDoS) attacks in three ways: (1) attack traffic has similar characteristics to benign traffic, making it highly concealed and challenging to detect, (2) traditional DDoS detection mechanisms cannot detect low volume LDoS attacks, and (3) LDoS attacks utilise TCP/IP protocol's three-way handshake mechanism where control measures look normal but reduce the quality of service (Zhijun et al., 2020; Chen et al., 2021). LDoS attacks are categorised into SRA, Quality of Services (QoS) attacks, and Service-Queue attacks depending on the protocols it damages (Rios et al., 2022). This study focuses on SRA.

In SRA, attackers mimic a legitimate user or device with a slow connection or low data transmission capacity (Tripathi and Hubballi, 2021; Reed et al., 2021) causing degraded performance, sensor or actuator data manipulation, system instability, and/or operational disruption (Zahid et al., 2022). Mitigating these attacks requires modifications to Request For Comments (RFC) protocols, which is cumbersome and involves extensive deliberations and discussions between stakeholders (Tripathi and Hubballi, 2021). Current solutions to detect SRA include Machine learning (ML), Deep learning (DL), and statistical and cognitional methods (Mittal et al., 2022). ML and DL are designed to learn relevant features and identify complex and subtle abnormal patterns from network traffic (Gaba et al., 2024; Bocu and Iavich, 2024; Ahmad et al., 2021; Saghezchi et al., 2022; Khraisat et al., 2019).

SRA are a growing concern in ICPS as these systems host critical infrastructure and services. SRA are particularly difficult to detect in ICPS due to large attack surfaces, the high degree of distribution, and the limitations of resource-constrained computers used in such systems. These constraints prevent us from deploying mainstream SRA detection techniques, such as ML and DL solutions, in ICPS. These techniques employ many features, multi-layers with many hidden nodes, and/or need backward and forward propagation to set the optimal hyperparameters, causing high processing and computation overheads beyond the capacity of typical resource-constrained ICPS hardware. Little emphasis has been given to optimising existing model sizes and managing performance overheads in resource-constrained environments. In addition, current works also do not cover the proactive and timely detection of slow-rate attacks in ICPS.

Mainstream approaches for SRA detection require further research to become applicable in ICPS; this article focuses on this gap by exploring the following research questions:

RQ1 Which existing ML/DL based SRA detection mechanisms are most suitable for ICPS applications?

* Corresponding author.
E-mail addresses: farzana.zahid@waikato.ac.nz (F. Zahid), matthew.kuo@aut.ac.nz (M.M.Y. Kuo), roopak.sinha@deakin.edu.au (R. Sinha).

https://doi.org/10.1016/j.cose.2025.104508
Received 22 August 2024; Received in revised form 15 February 2025; Accepted 24 April 2025
Available online 9 May 2025

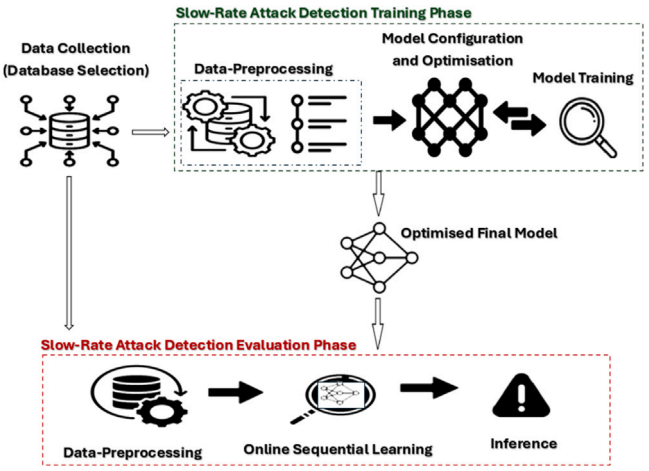

Fig. 1. Overview of the proposed light-weight slow-rate attack detection framework.

RQ2 What are the limitations of current SRA detection mechanisms identified in RQ1 for actively securing resource-constrained ICPS applications?

RQ3 How can ML/DL be used to actively secure ICPS against SRA?

RQ4 How can the effectiveness of the solution proposed in RQ3 be evaluated and compared to existing solutions identified in RQ1?

RQ1 and RQ2 are addressed through a systematic literature review presented in Section 2. RQ3 and RQ4 are answered using an adapted design science research methodology to develop and evaluate a light-weight mechanism to detect SRA in resource-constrained ICPS applications (Offermann et al., 2009) (Sections 3–5) .

Fig. 1 shows an overview of the proposed light-weight SRA detection framework for ICPS. It utilised an optimised OSELM model (Huang et al., 2005; Liang et al., 2006) that has enhanced efficiency, low resource usage and smaller size and complexity as compared to a traditional OSELM-based technique for SRA detection. We also propose an adapted stratified k-fold cross training method to reduce training time and mitigate any issues related to imbalanced datasets.

This framework comprises three primary components typical of machine-learning workflows: *data collection, training (slow-rate attack detection training)*, and *prediction (slow-rate attack detection evaluation)*. These are described later in Sections 3 and 4. Our technique carries out both binary and multi-class detection. Binary detection categorises incoming traffic into normal or attack traffic. In multi-class detection, the optimised OSELM is trained to classify traffic into normal or different slow-rate attack types: Slowloris, Golden Eye, SlowHTTPTest, and Hulk (Rios et al., 2022). The optimised OSELM is used to predict new and previously unseen SRA attacks in the prediction component during slow-rate attack detection evaluation phase (online-sequential learning).

The primary contributions of this article are:

1. A contemporary evaluation of existing SRA detection mechanisms (Section 2).
2. Design and implementation of a general light-weight active security framework based on optimised OSELM to detect binary and multi-class SRA for resource-constrained ICPS (Sections 3 and 4).
3. A novel and simple adapted stratified k-fold cross-training method with a significant focus on performance and accuracy of attack detection (Section 3).
4. Experimental validation of the proposed approach to show a reduction of the model's size, resource-utilisation, and attack detection time, enhancing the self-protection abilities of the resource-constrained ICPS (Section 5).
5. An evaluation of the effectiveness of the proposed framework through PLCs and the publicly available CIC-IDS2018 dataset (Sharafaldin et al., 2018) (Section 5).

2. Related works

Existing secondary studies related to LDoS attacks (Gaba et al., 2024; Jamal et al., 2023; Drinić and Čiča, 2024; Oluwatobi et al., 2020; Zhijun et al., 2020; Jaafar et al., 2019; Rios et al., 2022; Malliga et al., 2022) do not cover resource-constrained ICPS. Similarly, some surveys review security concerns in ICPS using ML/DL (Jamal et al., 2021; Haque et al., 2021; Zhang et al., 2021; Almajed et al., 2022; Umer et al., 2022; Luo et al., 2021) but do not cover LDoS detection. We surveyed works that included generic and ICPS-specific mechanisms for detecting SRA attacks using ML/DL (Table 1). All data related to our survey is available for download, as an Excel worksheet, via `www.removedforblindreview.com`. Please note: federated learning, and generative AI based detection models are out of the scope.

Our analysis indicates that ML-based solutions are easy to design and construct, based on one or no layers, and prove to be good at detecting SRA, as shown in Table 1. However, these models are not suited for resource-constrained ICPS in two ways: ML models cannot deal with diverse random attack traffic profiles while operating efficiently, and their dependence on a large number of features for training that causes high computational overheads (Wang et al., 2021a).

DL solutions are considered more accurate in detecting SRA because they can automatically extract complex features from incomplete and high dimensional data and maintain the history of patterns, which helps to detect different types of SRA (Zhang et al., 2021). However, these mechanisms exhibit slow processing speeds due to forward and backward propagation involving multiple hidden layers and complex mathematical operations. These models are sensitive to hyperparameters like epochs, a large number of hidden nodes, optimisers, learning rate, and stopping criteria. Moreover, these models required a large amount of data for training and generalisation. These factors combined result in high computational complexities and memory overheads (Asad et al., 2020; Liang et al., 2006) hindering practical implementation in resource-constrained environments.

Several works utilise shallow deep learning (SDL) models for classification because of their proven online learning detection capabilities, less training time and computational efficiency (Wang et al., 2021b; ElDahshan et al., 2022; Gunjal et al., 2022). The works (ElDahshan et al., 2022; Wang et al., 2021b) are based on traditional Extreme Learning Machine (ELM). ELM has certain limitations regarding training efficiency, memory requirements, and applicability. It is suitable for generating the best predictor by batch learning, where all the data is available upfront. ELM performs all data training at once and requires all the training samples to be stored in memory. These requirements result in exhaustive computational time memory overhead and require re-training each time new data arrives. Furthermore, existing ELM-based works are unsuitable for resource-constrained ICPS because of the involvement of many hidden nodes. Over-fitting, computational complexity, and memory required to store the node parameters increases as the number of hidden nodes increases. Additionally, the model does not generalise well on the new test data. Currently, ELM-based methods are unsuitable for application in real-time environments for attack detection (Al-Haija et al., 2024; Guo, 2019). Therefore, in this study, we propose to adopt OSELM, an extended version of ELM, to meet the continuous or sequential learning needs of real-time applications in ICPS (see Section 3.2).

Reinforcement learning (RL) is also used in some studies (Yu et al., 2024b; Shen et al., 2024b; Yu et al., 2024a; Shen et al., 2024a). RL is based on an adaptive learning approach (reward and penalty mechanism) that improves detection over time; however, these models are computationally extensive for resource-constrained ICPS. In Yu et al. (2024a), researchers have proposed a novel intrusion detection

Table 1
Comparison of slow-rate attack detection mechanisms. Metrics used: Accuracy (Acc), Precision (Pre), F1-score (F1).

Rf[a]	Detection model	Algo[b]	App[c]	DF[d]	F[e]	L[f]	H[g]	D.T[h]	RC[i]	Metrics	Datasets used
Abbas et al. (2024)	CNN1;CNN2	DL	S	Misc	47	10; 14	34; 32	NG	No	Acc = 0.95; Acc = 0.94	CICDIoT2023
Alslman et al. (2024)	DAE	DL	Semi	Misc	34	5	72	300; 200	No	Acc = 0.96; 0.85	SNMPMIB; UNSWNB15
Alabdulatif et al. (2024)	MLP;XGB;RF	ML; DL	S	27	NG	NG	NG	256	No	Acc = 0.99; Acc = 0.99	CICDIoT2023
Bocu and Iavich (2024)	Bi-LSTM	DL	S	NG	NG	Misc	128	0.79	No	Acc = 0.98	No
Hnamte and Hussain (2023)	CNN;BiLSTM	DL	S	NG	77	3	Misc	1712	No	Acc = 0.99	CIC-IDS2018
Almaraz-Rivera et al. (2022)	MLP	DL	S	Misc	15	4	51	NG	Yes	Acc = 0.965; Pre = 0.97	Bot-IoT
ElDahshan et al. (2022)	HBA-ELM;AoA-ELM	SDL[k]	S	Misc	38	1	77;79 84;883	NG	No	Acc = 0.97; Acc = 0.84	CIC-DoS2017
Muraleedharan and Janet (2021)	GRU	DL	S[j]	srcIP;dstIP;src-port; dstport;Pro	80	4	320	NG	No	Acc = 0.99; Pre = 0.99;F1 = 0.99	CIC-DoS2017
Wang et al. (2021b)	DBN-EGWO-KELM	DL	S	packets rate;conn	49	80	100	120	No	Acc = 0.97; Pre = 0.96	CIC-DoS2017
Fu et al. (2022)	LSTM	DL	S	NG	32	6	Misc	NG	No	Acc = 0.944;Pre = 0.92; F1 = 0.94	CIC-DoS2019; UTSA2021
Asad et al. (2020)	ANN	DL	S	Misc	66	7	Misc	NG	No	Acc = 0.98; F1 = 0.98	CIC-DoS2017
Pratomo et al. (2018)	AutoEncoder	DL	US[l]	bytes	NG	1;3;5	200; Misc	NG	No	NG	UNSW-NB15
Wang et al. (2022a)	Triplet CNN	DL	S	NG	NG	Misc	Misc	NG	No	Acc = 0.98; Pre = 0.99;F1 = 0.99	CIC-DoS2017
Gogoi and Ahmed (2022)	LSTM	DL	S	request;times-tamp	NG	3	Misc	NG	No	Acc = 0.99	CIC-DoS2017
Shaik and Kataoka (2021)	Multi-Autoencoder	DL	US	Misc	35	3	NG	200	No	Acc = 0.87;Acc = 0.95; Acc = 0.99;Pre = 0.99; F1 = 0.94	No
Xu et al. (2021)	1DCNN; GRU	DL	S	packets rate		10	64	NG	No	Acc = 0.97; Acc = 0.99; Pre = 0.98; Pre = 0.97	No
Sambangi et al. (2022)	Logistic Regression	ML	S	NG	25			NG	No	Acc = 0.90; Pre = 0.90; F1 = 0.94	CIC-DoS2017; CIC-DoS2019
Tang et al. (2021)	K-Means	ML	US	packets rate				NG	No	Acc = 0.98	WIDE2018
Hussain et al. (2022)	K-means	ML	US	Misc	NG			NG	No	NG	WIDE2018
Yan et al. (2019)	LR	ML	S	Misc	NG			1800	No	Acc = 0.94	No
Lima Filho et al. (2019)	RF; AdaBoost;DT;LR	ML	S	srcIP;dstIP;src-port; dstport;Pro	26			NG	No	Pre = .99; F1 = 0.99	CIC-DoS2017; CIC-IDS2018
Aamir and Zaidi (2021)	KNN; SVM; RF	ML	Semi[m]	traffic rate;delay; CPU utilisation	NG			NG	No	Acc = 0.966; Acc = 0.92; Acc = 0.95	CIC-DoS2017
Jain et al. (2022)	K-Means; SVM	ML	Semi	Misc	11			No	No	Acc = .89; Pre = 0.10	CIC-DoS2017
Kemp et al. (2020b)	RF;DT;NB;SVM;5NN; MLP	ML; DL	Semi	session data	12			NG	No	NG	No
Kemp et al. (2020a)	JRP;RF;C4.5D;C4.5N; NB; SVM; 5NN;MLP	ML; DL	Semi	Misc	12			NG	No	Acc = 0.99; Acc = 0.94; Acc = 0.81	No
Gunjal et al. (2022)	MLP;DCN	DL	Semi	NG	42;36	3	9	NG	No	Acc = 0.93; Acc = 0.70	CIC-IDS2018; UNBW2015
Li et al. (2022)	K-means	ML	Semi	net flow data	NG			NG	No	F1 = 22.7	CICDoS2017; ISCX2012
Sourbier et al. (2022)	Tangled program graph	ML	S	NG			NG	3847	No	Acc = 0.91; Pre = 0.953	CIC-DoS2017-2018
Kemp et al. (2023)	C4.5;KNN;NB;RF;JRip		Semi	net flow data		NG		NG	No	Acc = 0.98;Acc = 0.96	No
Vedula et al. (2021)	LSTM; RF	ML; DL	Semi	interarrival times of net flows	2			NG	Yes	Acc = 0.94; F1 = 0.93	UTSA-2021
Yu et al. (2024b)	DQN,CVAE	DL, RL	US, RL	Misc	105	Misc	Misc	NG	No	Acc = 1, 0.98	TON-IoT
Shen et al. (2024b)	CNN,D3QN	DL, RL	US, RL					2.414	No	NG	No
Yu et al. (2024a)	DDQN-LP,stochastic games	DRL	RL			2			No	NG	No
Shen et al. (2024a)	DQN, LSTM	DL, RL	US, RL	Misc	49		200; 128		No	Acc = 0.852;0.86;0.856, F1 = 0.90;0.856;0.827	UNSW-NB15
our work	*Optimised OSELM*	*SDL*	*S*	*Total forward packet;max inter-arrival time;total bytes used for headers in Fwd direction*	*3*	1	4; 5	*0.04; 0.05*	*Yes*	*Acc = 0.975;Acc = 0.96; Pre = 0.975;Pre = 0.97; F1 = 0.98;F1 = 0.965*	*CIC-IDS2018*

[a] References.
[b] Algorithm.
[c] Approach.
[d] Detection Feature.
[e] Number of Features.
[f] Number of Hidden Layers.
[g] Number of Hidden Nodes.
[h] Detection Time in sec.
[i] Resource-Constrained.
[j] Supervised.
[k] Shallow Deep Learning.
[l] Unsupervised.
[m] Semi-Supervised.

system with optimised hyperparameters. Nonetheless, extensive tuning and complexity made this model unsuitable for resource-constrained ICPS applications, where a delay in attack detection could result in catastrophic impacts.

Overall, we find that the research on detecting SRA in ICPS is still in its infancy. Existing works have prioritised attack detection accuracy, but pay negligible attention to performance and resource use. Our work aims to overcome these current gaps by focussing on the accuracy–efficiency tradeoff in the context of resource-constrained ICPS.

3. Framework design

Throughout design of the proposed framework, presented in Fig. 1, prioritises achieving a balance between accuracy and efficiency.

3.1. Dataset selection

The ideal dataset should contain a large amount of real-time network traffic of numerous interacting components as well as traffic data on contemporary SRA attacks (Farhan and Jasim, 2022). The CIC-IDS2018 dataset was chosen (Sharafaldin et al., 2018) from the list of datasets covered in our literature survey shown in Table 1. This dataset comprises benign and modern-day attacks, including SRA like GoldenEye, Slow-loris, SlowHTTPTest, and Hulk. The CIC-IDS2018 dataset contains real-world PCAP data gathered by the Canadian Institute for Cybersecurity, featuring more than 16 million rows and 80 features. Attacks are modelled on 50 machines, targeting an organisation with 5 departments with 420 machines and 30 servers.

From CIC-IDS2018, we selected data from Thursday, February 15, 2018, and Friday, February 16, 2018, during which a number of slow-rate attacks happened. The information about the labels used to represent slow-rate attack traffic in the dataset is given in Table

Table 2
Information about benign and slow–rate attacks types in CIC-IDS2018 dataset.

Label	Date	Attack start-time	Attack end-time	Instances
GoldenEye	Thur-15-2-18	9:26	10:09	41,508
Slowloris	Thur-15-2-18	10:59	11:40	10,990
SlowHTTPTest	Fri-16-2-18	10:12	11:08	461,912
Hulk	Fri-16-2-18	13:45	14:19	1,39,890
Benign	15-2-18; 16-2-18			1,442,849

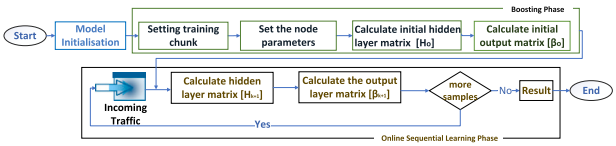

Fig. 2. Phases of online sequential extreme learning machine.

2. The network traffic data contains many features like source and destination IP addresses, ports number, timestamps, and many more. Further information about the CIC-IDS2018 dataset and attack types can be found in Sharafaldin et al. (2018). The csv files in CIC-IDS2018 containing benign and SRA network traffic were merged into a single file for training.

3.2. Model selection

Model (re-)training and evaluation can cause delays and impact timely decision-making, especially when dealing with previously unseen attacks (Zahid et al., 2024). Comparing the contemporary candidate models listed in Table 1, we choose OSELM for our framework. OSELM, an online extension of ELM, is better suited for streaming or sequential data and uses single hidden layer feedforward neural networks (Huang et al., 2006). OSELM's computational efficiency, small memory footprint, and faster and simpler learning capability make it suitable for resource-constrained ICPS. OSELM is time and memory-efficient as it neither requires re-training upon the arrival of new data nor does it store all the training samples in memory (Guo, 2019; Wang et al., 2022b). It can process training data in chunks or single samples and does not require tuning learning rates, epochs, and stopping criteria, making the learning phase significantly faster than traditional neural networks (Liang et al., 2006). OSELM has also shown good accuracy (Qaiwmchi et al., 2020; Li et al., 2018) and tolerance for noisy data (Guo, 2019; Lima et al., 2017), making it even more suitable for ICPS.

OSELM can efficiently learn the non-linear relationships between incoming data and their deviations or complex patterns, even if they fall within the linear range, which linear models miss (Li et al., 2018). SRA manipulate data or system parameters and force deviations from expected behaviour over time; for example, a slow-rate attack could include the alteration of packet header size, which is not linearly correlated with normal behaviour (discussed in more detail in Section 5). OSELM can reveal such hidden patterns in ICPS network traffic efficiently.

The *general architecture of OSELM* has three layers: an input layer, an output layer, and one hidden layer between them. *OSELM deployment* has two phases: boosting and online sequential-learning phases, as shown in Fig. 2.

In the *boosting phase*, some chunk of training data (number of samples equal to or greater than the number of hidden nodes) is used to train the model using these basic steps (Huang et al., 2005): (1) Initialise the parameters of the model randomly, i.e., the random assignment of input weights and biases (2) Calculate hidden layer output matrix (3) Calculate output weight matrix (4) Predict on a new sample.

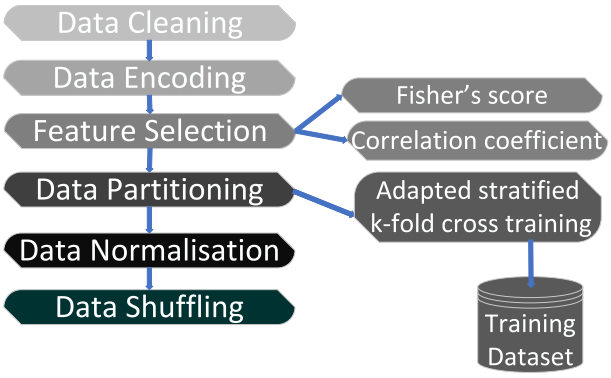

Fig. 3. Data pre-processing during training phase.

The *online sequential learning phase* follows the boosting phase where the model trains on incoming data in chunks by updating the weight matrix incrementally based on the previous weights and new data samples. Further details of how OSELM works are presented in Huang et al. (2005) and Liang et al. (2006).

3.3. Data pre-processing

Data pre-processing includes data cleaning, data encoding, feature selection, data partitioning, data normalisation and shuffling (Famili et al., 1997), as shown in Fig. 3.

Data cleaning improves data consistency and quality by correcting for missing, undefined, incorrect, or irrelevant information. We removed any columns containing only zeros, containing categorical values except for labels, and missing input values and infinite values. Duplicate rows were retained as such patterns can emerge when an attacker sends multiple identical or highly similar packets.

Data encoding converts categorical features into numerical values for machine learning models. We initially applied label encoding on the target label classes (containing the attack names), so the classifier (model) can learn the relationship and patterns according to their class number. Later, we applied a one-hot encoding scheme that converts multi-class numerical values into binary values (Hnamte and Hussain, 2023). For binary detection, we experimented with and without a one-hot encoding scheme. A label-only encoding scheme was selected for binary detection because it is more space efficient.

Feature selection identifies relevant features to retain in the data to enable accurate decision making, improve model performance, reduces the prediction time of the model and addresses the overfitting problem (Wang et al., 2022a). We used a hybrid selection approach to determine discriminative features with a strong relationship with the target label classes to understand attack patterns. First, we used Fisher's score (Gu et al., 2012) to rank features. We then applied the Pearson correlation coefficient technique to retain those features that best correlate with the target (Cohen et al., 2009). This technique assigns a value in a range of $[-1, 1]$, where 0 shows no correlation, 1 shows a perfect positive correlation, and -1 shows a total negative correlation. The purpose of using a hybrid approach is to reduce the impact of one technique being overly sensitive to specific data characteristics; thus, the hybrid approach increases the stability and generalisation of the selected features. The output of this step is used as an input to the attack detection phase.

Data partitioning employs a novel adaptation of stratified k-fold cross-training to address challenges arising from having an imbalanced dataset that can cause overfitting (Wang et al., 2022a), and to reduce training time. We created a *training dataset* (in a `csv` file) containing

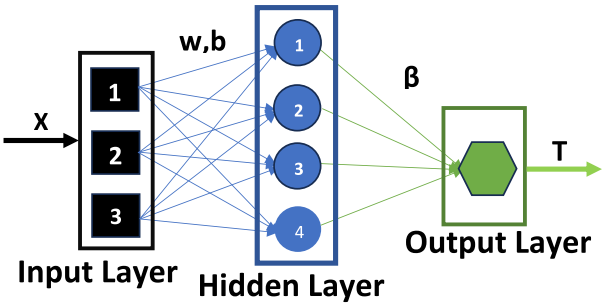

Fig. 4. Architecture of an optimised OSELM for binary detection with hidden nodes (L), input nodes (k), output nodes (o), input weight (w), bias (b), and output weight (β).

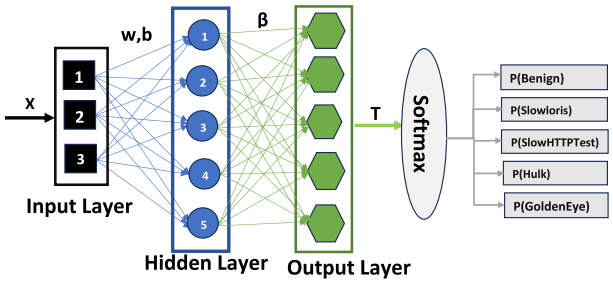

Fig. 5. Architecture of an optimised OSELM for multi-class detection with hidden nodes (L), input nodes (k), output nodes (o), input weight (w), bias (b), output weight (β), probability of target classes (P).

approximately the same distribution of each target class. Our method splits the training dataset into k folds equally in k iterations. During each iteration, one fold is used as a *validation set*, and the remaining $k-1$ folds are contained in the *training set*. After each iteration, performance metrics like the accuracy of the validation set and the iteration completion time (training time) are monitored and compared with previous iterations. Training stops if the accuracy drops or remains the same for consecutive iterations. This adapted method often completes training in fewer iterations than the traditional stratified k-cross validation method (Wang et al., 2022a) that requires a fixed number of iterations. The rationale for our proposed method is discussed further in Section 5.2.6. The final model is evaluated using a separate *testing dataset* with unequal sample distribution.

Data normalisation transforms data values into a common scale (similar range and mean value) for analysis. We used Min–Max Normalisation (scaling) method following formula (Sambangi et al., 2022):

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{1}$$

where x is the scaling data point. In this method, values for each feature are normalised between 0 (minimum) and 1 (maximum).

Data shuffling is applied to randomly shuffle the data to reduce bias and improve the model's performance during training and validation.

3.4. Model configuration and optimisation

Model configuration requires selecting an appropriate number of input nodes, hidden nodes, output nodes, and activation functions. The number of input nodes in the input layer of OSELM is equal to the optimal number of features identified during the feature selection step of the previous phase. The primary parameter for OSELM is therefore determining the optimal number of the hidden nodes. Further detail on the selection of hidden nodes is in Section 5. The additive hidden nodes with *sigmoid* activation function are generally used in the hidden layers. The number of output nodes in the output layer is based on the type of attack classification. For binary classification, only one output node and *sigmoid* activation function are used to produce the binary output (benign or attack), as shown in Fig. 4.

For multi-class classification, *softmax* activation is used and the number of output nodes is set equal to the number of target classes. In our work, we have five target classes; therefore, we have five nodes in the output layer of our model, shown in Fig. 5.

3.5. Training the model

The training phase uses the adapted stratified k-fold cross training method, shown in Fig. 6. During each iteration, the training process starts with the boosting phase (Fig. 2). In the boosting phase, a chunk of training data is selected for initial learning depending on the optimal

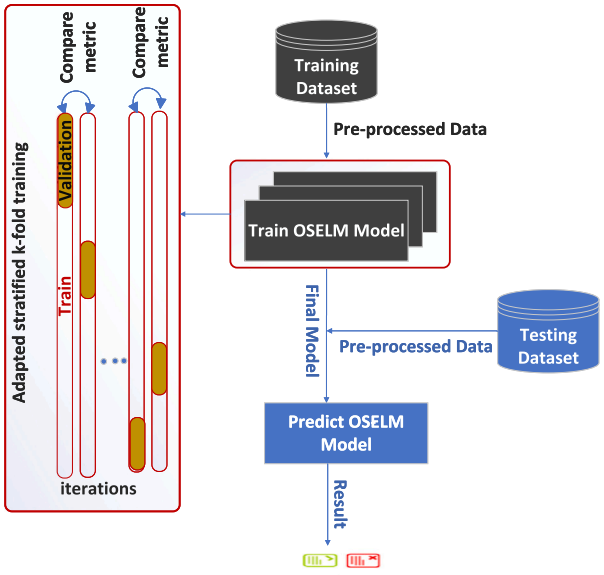

Fig. 6. Optimised OSELM model training and prediction.

number of hidden nodes. At first, the weight matrix and biases are initialised randomly to complete the initial training phase. Then, the initial hidden layer output matrix and initial output weight matrix are computed. Details of the OSELM training process can be found in Huang et al. (2005) and Liang et al. (2006).

Let N be the training dataset with X distinct samples pairs (x_i, t_i) such that $N = \{(x_i, t_i)$: $x_i = [x_{i1}, x_{i2} \cdots, x_{ik}]^T \in R^k$ is an input matrix, $t_i = [t_{i1}, t_{i2} \cdots, t_{io}]^T \in R^o$ is a target matrix, $i = 1, \ldots, X$, k is the number of input nodes, and o is the number of output nodes$\}$. Given the chunk of initial training dataset (N_o) with X_o distinct samples pairs selected from N such that: $N_o = \{(x_i, t_i)$: $x_i \in R^k, t_i \in R^o, i = 1, \ldots, X_o$; $N_o \geq L, L$ is number of hidden nodes$\}$, the steps involved in boosting phase are as follows (Huang et al., 2005)

Step 1. First, randomly assign the input weight (w_j) and bias (b_j) for input layer ($j = 1, \ldots, L$), w_j is a weight matrix ($[w_{j1}, w_{j2} \cdots, w_{jk}]^T \in R^k$) between input and ith hidden nodes and $b_j = [b_1, b_2 \cdots, b_k]^T \in R^k$ is a bias matrix.

Step 2. The initial hidden layer output matrix H_o is calculated as follows:

$$H_o = \begin{bmatrix} g(x_1.w_1 + b_1) & \cdots & g(x_1.w_L + b_L) \\ \vdots & \vdots & \vdots \\ g(x_{X_o}.w_1 + b_1) & \cdots & g(x_{X_o}.w_L + b_L) \end{bmatrix} \tag{2}$$

When the additive hidden nodes with the activation function $g(x)$: $R \rightarrow R$ are used, $G(w, b, x)$ is represented by the following equation:

$$G(w, b, x) = g(x.w + b) \tag{3}$$

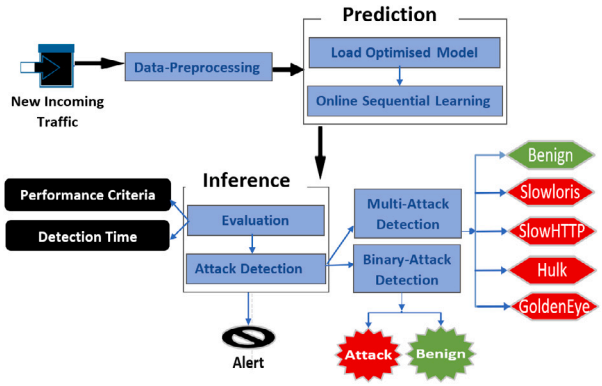

Fig. 7. Slow-rate attack detection evaluation phase.

where g is an activation function/sigmoid function commonly defined as:

$$g(x.w+b) = \frac{1}{1+e^{-(x.w+b)}} \tag{4}$$

Step 3. The output weight matrix (β_o) is calculated as follows:

$$\beta_o = M_o H_o^T T_o \tag{5}$$

where $M_o = (H_o^T H_o)^{-1}, T_o = [t_1, \ldots, t_L]^T$.

Step 4. Set $k' = 0$ where k' shows the first chunk of data.

After the boosting phase, the online sequential training phase starts (Step 5–8). Upon the arrival of new samples, the hidden layer output matrix and output weight matrix are updated. This phase is repeated until all the training samples have been processed. This is to ensure that the model is continuously adapting and learning from the new samples.

Let N_{k+1} is the arriving chunk of data having X_{j+1} number of samples in it such that: $N_{k+1} = \{(x_i, t_i)\colon x_i \in R^k, t_i \in R^o, i = (\sum_{j=0}^{k} X_j) + 1, \ldots, \sum_{j=0}^{k+1} X_j\}$. Let $y = (\sum_{j=0}^{k} X_j) + 1$ and $z = \sum_{j=0}^{k+1} X_j$.

Step 5. The hidden layer output matrix (H_{k+1}) is computed as:

$$H_{k+1} = \begin{bmatrix} g(x_y.w_1 + b_1) & \cdots & g(x_y.w_L + b_L) \\ \vdots & \cdots & \vdots \\ g(x_z.w_1 + b_1) & \cdots & g(x_z.w_L + b_L) \end{bmatrix} \tag{6}$$

Step 6. Set T_{k+1} by using following equation:

$$T_{k+1} = [t_y, \ldots, t_z]^T \tag{7}$$

Step 7. The output weight β_{k+1} is computed as follows:

$$\beta_{k+1} = \beta_k + M_{k+1} H_{k+1} (T_{k+1}^T - H_{k+1}^T \beta_k) \tag{8}$$

where $M_{k+1} = M_k - \frac{M_k H_{k+1} H_{k+1}^T M_k}{1 + H_{k+1}^T M_k H_{k+1}}$.

Step 8. Update k'=k'+1 and repeat from step 5.

After each iteration, the performance of the model is evaluated. When the model achieves the highest accuracy, the model with optimal parameters is saved and subsequently used for the prediction during deployment .

4. Deploying the framework

During the slow-rate attack detection evaluation phase, the trained model is used to detect slow-rate attacks in new and unseen traffic samples from the dataset. The overall process of prediction is shown in Fig. 7.

First, incoming traffic will be pre-processed by removing any missing values and noises, encoding and normalising, and shuffling data. These steps were described earlier in Section 3.3.

After data-preprocessing, the online sequential learning phase will be executed. In this phase, the parameters of the optimised OSELM will be restored and used. These parameters will be utilised to calculate the hidden layer output matrix and output matrix using Eqs. (6) and (8), as shown in Fig. 2. Finally, the model's predicted output (target score) is calculated as follows (Huang et al., 2005):

$$H\beta = T \tag{9}$$

where β and T of sizes is $L \times o$ and $X \times o$, respectively, are calculated as:

$$\beta = \begin{bmatrix} \beta_1^T \\ \beta_2^T \\ \vdots \\ \beta_L^T \end{bmatrix} = \begin{bmatrix} \beta_{11} & \beta_{12} & \cdots & \beta_{1o} \\ \vdots & \vdots & \cdots & \vdots \\ \beta_{L1} & \beta_{L2} & \cdots & \beta_{Lo} \end{bmatrix} \tag{10}$$

$$T = \begin{bmatrix} t_1^T \\ t_2^T \\ \vdots \\ t_X^T \end{bmatrix} = \begin{bmatrix} t_{11} & t_{12} & \cdots & t_{1o} \\ \vdots & \vdots & \cdots & \vdots \\ t_{X1} & t_{X2} & \cdots & t_{Xo} \end{bmatrix} \tag{11}$$

The model's output predicts whether the input samples represent benign or attack traffic in the case of binary classification. The Sigmoid function maps the target score to the range $[0, 1]$. If the probability is less than 0.5, the prediction states the value 0 (traffic is benign); otherwise, the traffic is an attack traffic. In the case of multi-class classification, the model classifies the traffic as normal or predicts the attack class type. The predicted target score is mapped using the Softmax function that converts target scores into probability distributions between $[0, 1]$. Each relative probability represents that the target score belongs to a specific target class (labels), and finally, the sum of the probabilities is equal to 1. The output with the highest probability is considered as the final output.

Table 3
Training and testing dataset information (number of samples, labels and values) for binary detection.

Label	Value	Training dataset	Testing dataset
Benign	0	36,000	1,442,849
Attack	1	36,000	654,300

5. Experimental analysis and discussion

5.1. Experimental setup

The model was trained on a virtual machine cluster configured with 8 CPUs, 31880 MB memory, and 64-bit Ubuntu operating system. For data preprocessing, we used Pandas and Sci-kit-learn Python libraries. For model implementation, TensorFlow, Keras, and Python[1] were used. The optimised model was evaluated by deploying on Raspberry PLC 50RRA with 4 GB RAM.

5.2. Analysis of training and testing datasets

Based on the adapted stratified k-fold cross training method, described earlier in Section 3.3, the training dataset contains almost equal distribution of each label set. For binary detection, the training dataset has two labels: Benign (0) and Attack (1). For multi-class detection, five labels (Benign, SlowHTTP-Test, Hulk, GoldenEye, Slowloris) indicate the types of SRA attacks. Respective values are shown in Tables 3 and 4. These tables also show the number of samples and values of each label used in the training dataset for binary and multi-class detection.

[1] The code will be available at https://github.com/FarzanaZahid/Slowrate-AttackDetection.

Table 4
Training and testing dataset information (number of samples, labels and values) for multi-class detection.

Label	Value	Training dataset	Testing dataset
Benign	0	9000	1,442,849
SlowHTTPTest	1	9000	461,912
Hulk	2	9000	1, 39, 890
GoldenEye	3	9000	41,508
Slowloris	4	9000	10,990

Table 5
Distribution of training and validation set for binary and multi-class detection.

Training dataset	Ratio (%)	Training set	Validation set
Binary detection	85 : 15	61,200	10,800
Multi-class detection	85 : 15	38,250	6750

Table 6
Features' ranking based Fisher's algorithm.

Feature name	Description	Rank
fl_dur	Flow Duration	1
tot_fw_pk	Total forward packets	2
fw_pkt_l_max	Maximum size of packet in forward direction	3
fw_iat_max	Maximum time between two packets sent in the forward direction	4
fw_hdr_len	Total bytes used for headers in the forward direction	5
pkt_size_avg	Average size of packet	6
bw_iat_avg	Mean time between two packets sent in the backward direction	7
fw_seg_min	Minimum segment size observed in the forward direction	8
bw_iat_min	Minimum time between two packets sent in the backward direction	9
fw_seg_avg	Average size observed in the forward direction	10

corr_matrix	fl_dur	tot_fw_pk	fw_pkt_l_max	fw_iat_max	fw_hdr_len	pkt_size_avg	label
fl_dur	1	0.16	0.11	0.36	-0.14	0.29	0.45
tot_fw_pk	0.16	1	0.11	-0.08	0.97	0.25	0.65
fw_pkt_l_max	0.11	0.11	1	0.11	0.13	0.61	0.26
fw_iat_max	0.36	-0.08	0.11	1	0.7	0.22	0.92
fw_hdr_len	-0.14	0.97	0.13	0.7	1	-0.24	0.78
pkt_size_avg	0.29	0.25	0.61	0.22	-0.24	1	0.37
label	0.45	0.65	0.26	0.92	0.78	0.37	1

Fig. 8. Features selection based on Pearson correlation.

Table 7
Initial training chunk size determination for binary detection.

Initial chunk size	Accuracy	Iterations = number	Training time in s
4	0.2	Iteration 1	0.7
	0.4	Iteration 2	0.68
	0.2	Iteration 3	0.68
	0.4	Iteration 4	0.67
	0.2	Iteration 5	0.60
	0.2	Iteration 6	0.55
	0.2	Iteration 7	0.5
6	0.5	Iteration 1	0.66
	0.6	Iteration 2	0.62
	0.5	Iteration 3	0.61
	0.7	Iteration 4	0.58
	0.7	Iteration 5	0.55
	0.7	Iteration 6	0.49
8	0.79	Iteration 1	0.6
	0.98	Iteration 2	0.5
	0.98	Iteration 3	0.3
	0.98	Iteration 4	0.25
10	0.1	Iteration 1	0.6
	0.6	Iteration 2	0.53
	0.6	Iteration 3	0.52
	0.8	Iteration 4	0.44
	0.7	Iteration 5	0.37
	0.8	Iteration 6	0.25
	0.8	Iteration 7	0.2

The total number of samples for the training dataset and testing dataset in case of binary detection are 72,000 and 2, 097, 149 (Table 3). For multi-class detection, the training and testing datasets have 45,000 and 2, 097, 149 samples (Table 4), respectively.

The distribution of samples in the training dataset as training and validation sets are given in Table 5.

5.2.1. Feature selection

The selection of the most relevant and informative features is critical for accurate SRA detection, to reduce the computational complexity, and to improve performance.

The CIC-IDS2018 dataset has 80 features, that we rank using Fisher's scoring technique. Then, based on these scores/ranks, we selected the top 10 features, shown in Table 6. To yield more accurate and efficient results, we applied Pearson correlation (Fig. 8) and identified three optimal features used for binary and multi-class detection. The final features selected are based on a strong positive correlation with the labels and strong positive or negative correlations with each other. This ensures that the framework is robust and effective at capturing not only common and recurring attack patterns but also diverse, unseen attacks. Fig. 8 shows that maximum inter-arrival time (*fw_iat_max*), total bytes used for headers in the forward direction (*fw_hdr_len*), and total forward packets (*tot_fw_pk*) have the strongest correlation with target class label.

The feature *Total forward packets (tot_fw_pk)*, is a potential indicator of SRA. In SRA, an attacker sends a small number of packets over an extended period; thus, an unusually low rate of packets compared to benign traffic may indicate an attack. Likewise, if the maximum time interval between two consecutive packets is abnormally high, it indicates a slow traffic rate compared to benign traffic. Therefore, *maximum inter-arrival time (fw_iat_max) between two packets* is another significant factor. In SRA, due to low volume, packet headers constitute a significant portion of the overall packet sizes (over time), resulting in reduced payloads. Hence, we also detect SRA by considering the *total bytes used for headers in the forward direction (fw_hdr_len)*. These three critical features are targetted in our approach.

5.2.2. Determination of hidden nodes

According to ELM learning theory, the number of hidden nodes (L) should be greater than the feature dimensions (Leng et al., 2015). In our context of resource-constrained ICPS, we determined the optimal number of hidden nodes as the configuration under which the validation set gives the maximum accuracy with the least training time (Section 3). For binary detection, we obtained optimal accuracy/training time when $L = 4$, as shown in Fig. 10. For multi-class detection, optimal accuracy was obtained with $L = 5$ (Fig. 9).

5.2.3. Initial training chunk size determination

Determining the initial chunk of data during the boosting phase sets a foundation for the online sequential learning phase and thus enables OSELM to adapt and learn from new data efficiently. Chunk sizes should equal or exceed the number of selected hidden nodes. For our work, the accuracy of the validation set and training time help determine the chunk sizes for binary detection (Table 7) and multi-class detection (Table 8).

Table 8
Initial training chunk size determination for multi-class detection.

Initial chunk size	Accuracy	Iterations number	Training time in s
	0.2	Iteration 1	0.6
	0.3	Iteration 2	0.58
	0.35	Iteration 3	0.3
	0.5	Iteration 4	0.27
5	0.6	Iteration 5	0.25
	0.65	Iteration 6	0.25
	0.72	Iteration 7	0.15
	0.72	Iteration 8	0.12
	0.72	Iteration 9	0.11
	0.3	Iteration 1	0.6
	0.49	Iteration 2	0.47
7	0.7	Iteration 3	0.23
	0.97	Iteration 4	0.19
	0.97	Iteration 5	0.17
	0.97	Iteration 6	0.15
	0.54	Iteration 1	0.5
	0.8	Iteration 2	0.21
9	0.97	Iteration 3	0.13
	0.97	Iteration 4	0.10
	0.97	Iteration 5	0.10
	0.9	Iteration 1	0.5
	0.7	Iteration 2	0.3
	0.7	Iteration 3	0.25
11	0.97	Iteration 4	0.19
	0.8	Iteration 6	0.11
	0.9	Iteration 7	0.10
	0.75	Iteration 8	0.10

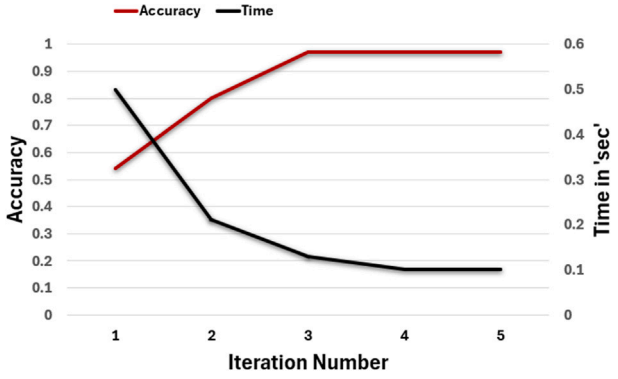

Fig. 9. Multi-class detection results during training phase with $L = 5$ and initial chunk size = 10.

In binary detection, we chose an initial chunk size equal to the number of hidden nodes and then gradually increased it. Table 7 shows the experimental results for choosing different chunk sizes. Finally, when we choose an initial chunk size equal to 8, the performance of the proposed framework reaches close to optimal.

The same pattern was applied for multi-class detection. We achieved the best results with both initial chunk sizes equal to 7 and 9, as shown in Table 8. A further increase to a chunk size of 11, we got 90% accuracy in the first iteration, but the result fluctuated as we did further iterations. Therefore, we opted for the chunk size that involved the fewest iterations and training time, i.e., 9.

5.2.4. Selection of the final optimised model for binary and multi-class detection

Figs. 9 and 10 show the accuracy, and training time in seconds for the iterations performed during the adapted stratified k-fold cross training. We evaluated the OSELM model performance after each training

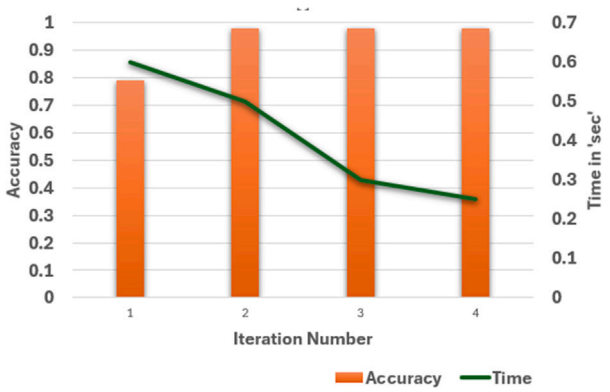

Fig. 10. Binary detection results during training phase with $L = 4$ and initial chunk size = 8.

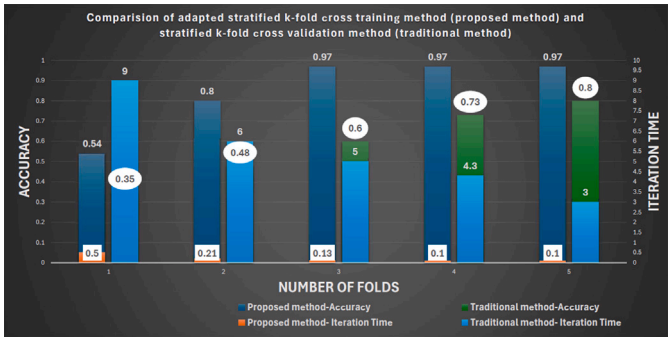

Fig. 11. Comparison of adapted stratified k-fold cross training method (proposed method) and stratified k-fold cross validation method (traditional method).

iteration. For example, in the case of binary detection, iteration 1 and 2 produced accuracy of 79% and 98%, respectively. This increase meant that another iteration was executed. We stopped at iteration 4 when the gains in accuracy stopped. The same dynamic procedure was applied for multi-class detection and we saved the model when the accuracy reached close to the optimal (97%).

5.2.5. Evaluation

The performance of the trained model was evaluated on the testing dataset containing new and unseen data samples. The prediction was carried out on the raspberry PLC 50RRA. The result of the trained model was evaluated with performance criteria accuracy, recall, precision, F1-score, shown in Table 9. We also considered average detection time (including data-preprocessing time) and system overhead (memory and CPU overhead), to ensure that the proposed model is lightweight and feasible for resource-constrained ICPS. Table 9 shows the average attack detection time, memory and CPU usage of the PLC when OSELM performs the binary and multi-class detection over the unseen and new data.

5.2.6. Justification for adapted stratified k-fold cross-training

The key objective of the adapted stratified k-fold cross-training method is to optimise the selected model's performance by reducing training time and addressing the imbalanced dataset challenge. The rationale for this method becomes clearer by comparing it with the traditional stratified k-cross validation method (Wang et al., 2022a). In contrast to the traditional method, the adapted stratified k-fold cross training method features a variable number of iterations. Stopping when no further accuracy gains happen in an iteration, helps minimise computational resource usage and prevents overfitting, as shown Fig. 11.

Table 9
Performance evaluation of proposed framework.

Proposed framework	Accuracy	Precision	F1-score	Recall	Detection time (s)	CPU usage	Memory usage
Binary detection	0.975	0.975	0.98	0.975	0.03	10%	2.6 MB
Multi-class detection	0.96	0.97	0.965	0.97	0.04	12%	3 MB

Table 10
Comparison of proposed method and traditional method.

Folds	Proposed method		Traditional method	
	Accuracy	Time	Accuracy	Time
1	0.54	0.5	0.35	9
2	0.8	0.21	0.48	6
3	0.97	0.13	0.60	5
4	0.97	0.10	0.73	4.3
5	0.97	0.10	0.80	3
Result	0.97		Average = 0.59	

The performance monitoring criteria in our training method differ from the traditional method. Our approach compares metrics from the previous iteration (accuracy and iteration time) rather than averaging the accuracy results from all k iterations. To resolve the imbalanced dataset issues the training dataset is created with an equal distribution of each target class, including majority and minority target classes. However, unlike the traditional method, the proportion of target classes in each iteration is selected randomly to reduce bias.

Table 10 compares the accuracy and training time of our approach with traditional methods. For instance, we obtained an optimal accuracy of 97% at the third iteration in the case of binary detection. Further iterations were used to confirm the result. The traditional method with a fixed number of iterations $k = 5$ achieved an accuracy of 59% and was computationally more expensive.

5.3. Comparison with selected existing works

A general and qualitative comparison of our proposed framework with existing literature was presented in Section 2. As numerous approaches proposed in literature vary significantly in datasets used and maturity of available implementations, conducting a comprehensive quantitative comparison among approaches is challenging. Therefore, we compare our approach quantitatively with selected existing binary and multi-class SRA detection approaches using the criteria shown in Tables 11 and 12.

A denoising autoencoder (DAE) with four hidden layers and seventy-six hidden nodes for binary and multi-class attack detection is proposed in Alslman et al. (2024). In contrast to this work, our approach uses forward propagation within a single hidden layer over only three features. Our approach also outperforms this work by showing better precision (97% over 85%), F1-score (96.5% over 87%), and recall (97% over 80%). In multi-class detection (Table 12), our model has the same accuracy but performs better with respect to other metrics. Our approach carries out binary and multi-class detections in 0.03 and 0.04 s, respectively, compared to 200 and 300 s, respectively, for (Alslman et al., 2024).

Another approach uses the Multi-Layer Perceptron (MLP) model (Almaraz-Rivera et al., 2022) to detect application layer attacks for secure communication between devices using fifteen features. Similar to our work, their detector (MLP) is categorised as a feed-forward neural network (Almaraz-Rivera et al., 2022) and is used for only binary detection with a balanced dataset. In contrast, our approach also carries out multi-class detection.

In ElDahshan et al. (2022), the ELM model is fine-tuned to carry out multi-class attack detection using several hidden nodes. Table 11 shows that our approach has a slightly lower accuracy (96%) than HBA-ELM (97%) (ElDahshan et al., 2022). Nonetheless, our approach balances performance and accuracy by significantly outperforming this work in detection time and space complexity.

The selected works we compared with do not describe their algorithms' memory and CPU usage. They typically use many more features than our work, and most do not consider the imbalance dataset issue, which can affect the accuracy of attack detection and result in overfitting. In contrast, a PLC-based deployment of out method shows drastic reduction in model size, high efficiency and effectiveness.

5.3.1. Space Complexity of Optimised ***OSELM!***

We have present the space complexity of our optimised OSELM to show that it can run efficiently in most resource-constrained environments. We also compare with the space complexity of other models (note: We have estimated the space complexity of each model based on the partial information in the published articles). The space complexity of a model depends upon the parameters such as the number of weights and biases of each layer in a network, gradient, and intermediate activation during computation - `(# of parameters)` (Deonatan, 2023).

For each hidden node in the optimised OSELM, the parameters involve: weights (connecting each node in the input layer to each hidden node in the hidden layer), and biases (associated with each hidden node). For binary SRA detection, shown in Fig. 4, we have three input nodes, four hidden nodes, and one output node. For each hidden node, connected to the three input nodes, we have 4 weights and 1 bias.

The output layers do not contribute significantly to the space complexity. Storing 12 weights and 4 biases (16), is extremely efficient. Multi-class SRA detection (Fig. 5) involves three input nodes, five hidden nodes, and five output nodes; this requires storing only 15 weights and 5 biases (20), which is also very efficient.

Tables 11 and 12 show the space complexities of the models used in existing studies. For example, the DAE model in Alslman et al. (2024) has an input size 34, five hidden layers, twenty hidden nodes in the first and second layer, sixteen hidden nodes in the third layer, and eight in the fourth and fifth layers (in total hidden nodes are 72). Thus, the proposed model in Alslman et al. (2024) needs storage for a total of 2744 parameters. This figure indicates that such storage requirements can pose significant challenges in resource-constrained applications. Therefore, our model has superior efficiency.

5.4. Threats to validity

Our study uses a balanced dataset for the model's training and focuses on accuracy and faster training time as the critical metrics for selecting the final optimised model. In real-world applications, the focus on accuracy and training time are essential metrics to ensure the effectiveness and efficiency of the model. The inclusion of the training time metric helps us to ensure that the optimised model is not only performing well in terms of accuracy but also meet the operational constraints. The combination of two metrics helps us to balance between model complexity and computational efficiency. However, in the future, additional performance metrics could be included for a more holistic assessment of our approach.

Additionally, our focus was to make PLC-based ICPS security-aware. We selected only three features and received good results. We can further incorporate more negatively correlated features with existing ones, providing us with broader coverage of unknown attacks.

There is also a possibility of drastic changes in data distribution or the availability of new features, and so it may be necessary to perform

Table 11
Comparison of proposed binary SRA detection framework with existing approach.

Proposed framework	Our work	Abbas et al. (2024)	Almaraz-Rivera et al. (2022)
Model used	Optimised OSELM	DAE	MLP
Number of features	3	34	15
Hidden layers	1	5	4
Number of hidden nodes	4	72	51
Balanced dataset	Yes	No	Yes
Accuracy	0.975	0.85	0.965
Precision	0.975	0.85	0.97
F1-score	0.98	0.75	0.97
Recall	0.975	0.65	0.978
Detection time (s)	0.03	200	NP
CPU usage	10%	NP	NP
Memory usage	2.6 MB	NP	NP
Space complexity	16	2744	2987

Table 12
Comparison of proposed multi-class SRA detection framework with existing approach (NP: Not Provided).

Proposed framework	Our work	Alslman et al. (2024)	Abbas et al. (2024)	ElDahshan et al. (2022)
Model used	Optimised OSELM	DAE	CNN1	HBA-ELM
Number of features	3	34	47	38
Hidden layers	1	4	10	1
Number of hidden nodes	5	72	34	77
Balanced dataset	Yes	No	No	No
Accuracy	0.96	0.96	0.95	0.97
Precision	0.97	0.85	0.95	0.97
F1-score	0.965	0.87	0.94	0.98
Recall	0.97	0.8	0.95	NP
Detection time (s)	0.04	300	NP	NP
CPU usage	12%	NP	NP	NP
Memory usage	3 MB	NP	NP	NP
Space complexity	20	2744	11 522	3003

feature selection over time to better adapt the model. Unreliable performance can occur if there is significant difference in feature composition between the training and testing datasets. We consider this aspect as a promising future direction.

6. Conclusions and future directions

We propose an efficient and accurate approach to detect SRA in resource-constrained ICPS. Our light-weight active security framework involves OSELM based SRA detection and we deployed this framework onto a PLC for evaluating its performance on a publicly available dataset. Experimental results for binary and multi-class detection show high accuracy, low prediction times, outperforming approaches for SRA detection.

In the future, we will intend to study the power consumption of our proposed model and compare it with the state-of-the-art. Future work will also involve the generalisation and stability of detecting different LDoS attacks. Also, a deep learning-based solution for SRA attack mitigation, potentially capable of meeting the requirements of resource-constrained ICPS, could be explored. As resource-constrained applications often have access to limited energy resources, our approach can be optimised to actively secure resource-constrained systems with minimum computational power using tinyML. Moreover, the proposed work can be evaluated on multiple datasets to ensure the broader perspective of its performance and usability.

CRediT authorship contribution statement

Farzana Zahid: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Data curation, Conceptualization. **Matthew M.Y. Kuo:** Writing – review & editing. **Roopak Sinha:** Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The dataset employed in this research is readily accessible via the following link:
https://www.unb.ca/cic/datasets/ids-2018.html. It is important to note that the dataset was created by Sharafaldin et al. (2018).

References

Aamir, Muhammad, Zaidi, Syed Mustafa Ali, 2021. Clustering based semi-supervised machine learning for DDoS attack classification. J. King Saud Univ.-Comput. Inf. Sci. 33 (4), 436–446.

Abbas, Sidra, Bouazzi, Imen, Ojo, Stephen, Al Hejaili, Abdullah, Sampedro, Gabriel Avelino, Almadhor, Ahmad, Gregus, Michal, 2024. Evaluating deep learning variants for cyber-attacks detection and multi-class classification in IoT networks. PeerJ Comput. Sci. 10, e1793.

Ahmad, Zeeshan, Shahid Khan, Adnan, Wai Shiang, Cheah, Abdullah, Johari, Ahmad, Farhan, 2021. Network intrusion detection system: A systematic study of machine learning and deep learning approaches. Trans. Emerg. Telecommun. Technol. 32 (1), e4150.

Al-Haija, Qasem Abu, Altamimi, Shahad, AlWadi, Mazen, 2024. Analysis of extreme learning machines (ELMs) for intelligent intrusion detection systems: A survey. Expert Syst. Appl. 124317.

Alabdulatif, Abdullah, Thilakarathne, Navod Neranjan, Aashiq, Mohamed, 2024. Machine learning enabled novel real-time IoT targeted DoS/DDoS cyber attack detection system. Comput. Mater. Contin. 80 (3).

Almajed, Rasha, Ibrahim, Amer, Abualkishik, Abedallah Zaid, Mourad, Nahia, Almansour, Faris A., 2022. Using machine learning algorithm for detection of cyber-attacks in cyber physical systems. Period. Eng. Nat. Sci. 10 (3), 261–275.

Almaraz-Rivera, Josue Genaro, Perez-Diaz, Jesus Arturo, Cantoral-Ceballos, Jose Antonio, 2022. Transport and application layer DDoS attacks detection to IoT devices by using machine learning and deep learning models. Sensors 22 (9), 3367.

Alslman, Yasmeen, Alkasassbeh, Mouhammd, Almseidin, Mohammad, 2024. A robust SNMP-MIB intrusion detection system against adversarial attacks. Arab. J. Sci. Eng. 49 (3), 4179–4195.

Asad, Muhammad, Asim, Muhammad, Javed, Talha, Beg, Mirza O., Mujtaba, Hasan, Abbas, Sohail, 2020. Deepdetect: detection of distributed denial of service attacks using deep learning. Comput. J. 63 (7), 983–994.

Bocu, Razvan, Iavich, Maksim, 2024. Enhanced detection of low-rate DDoS attack patterns using machine learning models. J. Netw. Comput. Appl. 103903.

Chen, Ming, Chen, Jing, Wei, Xianglin, Chen, Bing, 2021. Is low-rate distributed denial of service a great threat to the Internet? IET Inf. Secur. 15 (5), 351–363.

Cohen, Israel, Huang, Yiteng, Chen, Jingdong, Benesty, Jacob, Benesty, Jacob, Chen, Jingdong, Huang, Yiteng, Cohen, Israel, 2009. Pearson correlation coefficient. Noise Reduct. Speech Process. 1–4.

Deonatan, Dinata, 2023. CNN, KNN, and SVM analysis.

Drinić, Dušan, Čiča, Zoran, 2024. Survey on low-rate DDoS attacks, detection and defense. In: 2024 23rd International Symposium INFOTEH-JAHORINA. INFOTEH, IEEE, pp. 1–6.

ElDahshan, Kamal A., AlHabshy, AbdAllah A., Hameed, Bashar I., 2022. Meta-heuristic optimization algorithm-based hierarchical intrusion detection system. Computers 11 (12), 170.

Famili, A., Shen, Wei-Min, Weber, Richard, Simoudis, Evangelos, 1997. Data preprocessing and intelligent data analysis. Intell. Data Anal. 1 (1), 3–23.

Farhan, Baraa Ismael, Jasim, Ammar D., 2022. Performance analysis of intrusion detection for deep learning model based on CSE-CIC-IDS2018 dataset. Indones. J. Electr. Eng. Comput. Sci. 26 (2), 1165–1172.

Fu, Yu, Duan, Xueyuan, Wang, Kun, Li, Bin, 2022. Low-rate Denial of Service attack detection method based on time-frequency characteristics. J. Cloud Comput. 11 (1), 31.

Gaba, Shivani, Budhiraja, Ishan, Kumar, Vimal, Martha, Sheshikala, Khurmi, Jebreel, Singh, Akansha, Singh, Krishna Kant, Askar, Sameh S, Abouhawwash, Mohamed, 2024. A systematic analysis of enhancing cyber security using deep learning for cyber physical systems. IEEE Access.

Gogoi, Bronjon, Ahmed, Tasiruddin, 2022. HTTP low and slow DoS attack detection using LSTM based deep learning. In: 2022 IEEE 19th India Council International Conference. INDICON, IEEE, pp. 1–6.

Gu, Quanquan, Li, Zhenhui, Han, Jiawei, 2012. Generalized fisher score for feature selection. arXiv preprint arXiv:1202.3725.

Gunjal, Hardik, Patel, Preetkumar, Ebrahimi, Dr. Dariush, 2022. Smart network intrusion detection system for cyber security of industrial IoT.

Guo, Wei, 2019. Robust adaptive online sequential extreme learning machine for predicting nonstationary data streams with outliers. J. Algorithms Comput. Technol. 13, 1748302619895421.

Haque, Nur Imtiazul, Shahriar, Md. Hasan, Dastgir, Md. Golam, Debnath, Anjan, Parvez, Imtiaz, Sarwat, Arif, Rahman, Mohammad Ashiqur, 2021. A survey of machine learning-based cyber-physical attack generation, detection, and mitigation in smart-grid. In: 2020 52nd North American Power Symposium. NAPS, IEEE, pp. 1–6.

Hnamte, Vanlalruata, Hussain, Jamal, 2023. DCNNBiLSTM: An efficient hybrid deep learning-based intrusion detection system. Telemat. Inform. Rep. 10, 100053.

Huang, Guang-Bin, Liang, Nan-Ying, Rong, Hai-Jun, Saratchandran, Paramasivan, Sundararajan, Narasimhan, 2005. On-line sequential extreme learning machine. Comput. Intell. 2005, 232–237.

Huang, Guang-Bin, Zhu, Qin-Yu, Siew, Chee-Kheong, 2006. Extreme learning machine: theory and applications. Neurocomputing 70 (1–3), 489–501.

Hussain, Tariq, Saeed, Muhammad Irfan, Khan, Irfan Ullah, Aslam, Nida, Aljameel, Sumayh S., 2022. Implementation of a clustering-based LDDoS detection method. Electronics 11 (18), 2804.

Jaafar, Ghafar A., Abdullah, Shahidan M., Ismail, Saifuladli, et al., 2019. Review of recent detection methods for HTTP DDoS attack. J. Comput. Netw. Commun. 2019.

Jain, Meenal, Kaur, Gagandeep, Saxena, Vikas, 2022. A K-Means clustering and SVM based hybrid concept drift detection technique for network anomaly detection. Expert Syst. Appl. 193, 116510.

Jamal, Alshaibi Ahmed, Majid, Al-Ani Mustafa, Konev, Anton, Kosachenko, Tatiana, Shelupanov, Alexander, 2021. A review on security analysis of cyber physical systems using Machine learning. Mater. Today: Proc..

Jamal, Alshaibi Ahmed, Majid, Al-Ani Mustafa, Konev, Anton, Kosachenko, Tatiana, Shelupanov, Alexander, 2023. A review on security analysis of cyber physical systems using Machine learning. Mater. Today: Proc. 80, 2302–2306.

Kemp, Clifford, Calvert, Chad, Khoshgoftaar, Taghi M., 2020a. Detection methods of slow read Dos using full packet capture data. In: 2020 IEEE 21st International Conference on Information Reuse and Integration for Data Science. IRI, IEEE, pp. 9–16.

Kemp, Cliff, Calvert, Chad, Khoshgoftaar, Taghi M., 2020b. Netflow feature evaluation for the detection of slow read HTTP attacks. Reuse Intell. Syst. 181–219.

Kemp, Cliff, Calvert, Chad, Khoshgoftaar, Taghi M., Leevy, Joffrey L., 2023. An approach to application-layer DoS detection. J. Big Data 10 (1), 22.

Khraisat, Ansam, Gondal, Iqbal, Vamplew, Peter, Kamruzzaman, Joarder, 2019. Survey of intrusion detection systems: techniques, datasets and challenges. Cybersecurity 2 (1), 1–22.

Leng, Qian, Qi, Honggang, Miao, Jun, Zhu, Wentao, Su, Guiping, et al., 2015. One-class classification with extreme learning machine. Math. Probl. Eng. 2015.

Li, Yuancheng, Qiu, Rixuan, Jing, Sitong, 2018. Intrusion detection system using Online Sequence Extreme Learning Machine (OS-ELM) in Advanced Metering Infrastructure of Smart Grid. PloS One 13 (2), e0192216.

Li, Bin, Wang, Yijie, Xu, Kele, Cheng, Li, Qin, Zhiquan, 2022. DFAID: Density-aware and feature-deviated active intrusion detection over network traffic streams. Comput. Secur. 118, 102719.

Liang, Nan-Ying, Huang, Guang-Bin, Saratchandran, Paramasivan, Sundararajan, Narasimhan, 2006. A fast and accurate online sequential learning algorithm for feedforward networks. IEEE Trans. Neural Netw. 17 (6), 1411–1423.

Lima, Aranildo R., Hsieh, William W., Cannon, Alex J., 2017. Variable complexity online sequential extreme learning machine, with applications to streamflow prediction. J. Hydrol. 555, 983–994.

Lima Filho, Francisco Sales de, Silveira, Frederico AF, de Medeiros Brito Junior, Agostinho, Vargas-Solar, Genoveva, Silveira, Luiz F, 2019. Smart detection: an online approach for DoS/DDoS attack detection using machine learning. Secur. Commun. Netw. 2019, 1–15.

Luo, Yuan, Xiao, Ya, Cheng, Long, Peng, Guojun, Yao, Danfeng, 2021. Deep learning-based anomaly detection in cyber-physical systems: Progress and opportunities. ACM Comput. Surv. 54 (5), 1–36.

Malliga, Subramaniam, Nandhini, P.S., Kogilavani, Shanmuga Vadivel, 2022. A comprehensive review of deep learning techniques for the detection of (distributed) denial of service attacks. Inf. Technol. Control. 51 (1), 180–215.

Mittal, Meenakshi, Kumar, Krishan, Behal, Sunny, 2022. Deep learning approaches for detecting DDoS attacks: A systematic review. Soft Comput. 1–37.

Muraleedharan, N., Janet, B., 2021. A deep learning based HTTP slow DoS classification approach using flow data. ICT Express 7 (2), 210–214.

Offermann, Philipp, Levina, Olga, Schönherr, Marten, Bub, Udo, 2009. Outline of a design science research process. In: Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology. In: 7, Association for Computing Machinery, New York, NY, USA, pp. 1–11.

Oluwatobi, Shadrach Akanji, Abisoye, Opeyemi Aderike, Sulaimon, A. Bashir, Ojerinde, O.A., 2020. A survey on slow DDoS attack detection techniques. In: 3rd International Conference on Information Technology in Education and Development.

Pratomo, Baskoro Adi, Burnap, Pete, Theodorakopoulos, George, 2018. Unsupervised approach for detecting low rate attacks on network traffic with autoencoder. In: 2018 International Conference on Cyber Security and Protection of Digital Services. Cyber Security, IEEE, pp. 1–8.

Qaiwmchi, Nedhal Ahmad Hamdi, Amintoosi, Haleh, Mohajerzadeh, Amirhossein, 2020. Intrusion detection system based on gradient corrected online sequential extreme learning machine. IEEE Access 9, 4983–4999.

Reed, Andy, Dooley, Laurence S., Mostefaoui, Soraya Kouadri, 2021. A reliable real-time slow DoS detection framework for resource-constrained IoT networks. In: 2021 IEEE Global Communications Conference. GLOBECOM, IEEE, pp. 1–6.

Rios, Vinícius De Miranda, Inacio, Pedro R.M., Magoni, Damien, Freire, Mário M., 2022. Detection and mitigation of low-rate denial-of-service attacks: A survey. IEEE Access 10, 76648–76668.

Saghezchi, Firooz B, Mantas, Georgios, Violas, Manuel A, de Oliveira Duarte, A Manuel, Rodriguez, Jonathan, 2022. Machine learning for DDoS attack detection in industry 4.0 CPPSs. Electronics 11 (4), 602.

Sambangi, Swathi, Gondi, Lakshmeeswari, Aljawarneh, Shadi, 2022. A feature similarity machine learning model for DDoS attack detection in modern network environments for industry 4.0. Comput. Electr. Eng. 100, 107955.

Shaik, Tahir Ahmed, Kataoka, Kotaro, 2021. Capsaeul: Slow HTTP DoS attack detection using autoencoders through unsupervised learning. In: Proceedings of the 16th Asian Internet Engineering Conference. pp. 49–55.

Sharafaldin, Iman, Lashkari, Arash Habibi, Ghorbani, Ali A., 2018. Toward generating a new intrusion detection dataset and intrusion traffic characterization. ICISSp 1, 108–116.

Shen, Shigen, Cai, Chenpeng, Li, Zhenwei, Shen, Yizhou, Wu, Guowen, Yu, Shui, 2024a. Deep Q-network-based heuristic intrusion detection against edge-based SIoT zero-day attacks. Appl. Soft Comput. 150, 111080.

Shen, Shigen, Cai, Chenpeng, Shen, Yizhou, Wu, Xiaoping, Ke, Wenlong, Yu, Shui, 2024b. MFGD3QN: Enhancing edge intelligence defense against DDoS with mean-field games and dueling double deep Q-network. IEEE Internet Things J..

Sourbier, Nicolas, Desnos, Karol, Guyet, Thomas, Majorczyk, Frederic, Gesny, Olivier, Pelcat, Maxime, 2022. SECURE-GEGELATI always-on intrusion detection through GEGELATI lightweight tangled program graphs. J. Signal Process. Syst. 94 (7), 753–770.

Tang, Dan, Zhang, Siqi, Chen, Jingwen, Wang, Xiyin, 2021. The detection of low-rate DoS attacks using the SADBSCAN algorithm. Inform. Sci. 565, 229–247.

Tripathi, Nikhil, Hubballi, Neminath, 2021. Application layer denial-of-service attacks and defense mechanisms: A survey. ACM Comput. Surv. 54 (4), 1–33.

Umer, Muhammad, Sadiq, Saima, Karamti, Hanen, Alhebshi, Reemah M., Al-nowaiser, Khaled, Eshmawi, Ala'Abdulmajid, Song, Houbing, Ashraf, Imran, 2022. Deep learning-based intrusion detection methods in cyber-physical systems: Challenges and future trends. Electronics 11 (20), 3326.

Vedula, Vasudha, Lama, Palden, Boppana, Rajendra V., Trejo, Luis A., 2021. On the detection of low-rate denial of service attacks at transport and application layers. Electronics 10 (17), 2105.

Wang, Zhendong, Li, Zeyu, He, Daojing, Chan, Sammy, 2022a. A lightweight approach for network intrusion detection in Industrial cyber-physical systems based on knowledge distillation and deep metric learning. Expert Syst. Appl. 206, 117671.

Wang, Xiaoping, Tu, Shanshan, Zhao, Wei, Shi, Chengjie, 2022b. A novel energy-based online sequential extreme learning machine to detect anomalies over real-time data streams. Neural Comput. Appl. 34 (2), 823–831.

Wang, Weiping, Wang, Zhaorong, Zhou, Zhanfan, Deng, Haixia, Zhao, Weiliang, Wang, Chunyang, Guo, Yongzhen, 2021a. Anomaly detection of industrial control systems based on transfer learning. Tsinghua Sci. Technol. 26 (6), 821–832.

Wang, Zhendong, Zeng, Yong, Liu, Yaodi, Li, Dahai, 2021b. Deep belief network integrating improved kernel-based extreme learning machine for network intrusion detection. IEEE Access 9, 16062–16091.

Xu, Congyuan, Shen, Jizhong, Du, Xin, 2021. Low-rate DoS attack detection method based on hybrid deep neural networks. J. Inf. Secur. Appl. 60, 102879.

Yan, Yudong, Tang, Dan, Zhan, Sijia, Dai, Rui, Chen, Jingwen, Zhu, Ningbo, 2019. Low-rate Dos attack detection based on improved logistic regression. In: 2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems. HPCC/SmartCity/DSS, IEEE, pp. 468–476.

Yu, Shoujian, Wang, Xuan, Shen, Yizhou, Wu, Guowen, Yu, Shui, Shen, Shigen, 2024a. Novel intrusion detection strategies with optimal hyper parameters for industrial internet of things based on stochastic games and double deep Q-networks. IEEE Internet Things J..

Yu, Shoujian, Zhai, Rong, Shen, Yizhou, Wu, Guowen, Zhang, Hong, Yu, Shui, Shen, Shigen, 2024b. Deep Q-network-based open-set intrusion detection solution for industrial internet of things. IEEE Internet Things J. 11.

Zahid, Farzana, Funchal, Gustavo, Melo, Victoria, Kuo, Matthew M.Y., Leitao, Paulo, Sinha, Roopak, 2022. DDoS attacks on smart manufacturing systems: A cross-domain taxonomy and attack vectors. In: 2022 IEEE 20th International Conference on Industrial Informatics. INDIN, IEEE, pp. 214–219.

Zahid, Farzana, Kuo, Matthew M.Y., Sinha, Roopak, Funchal, Gustavo, Pedrosa, Tiago, "Leitão, Paulo, 2024. Actively detecting multiscale flooding attacks & attack volumes in resource-constrained ICPS. IEEE Trans. Ind. Inform. 1–9.

Zhang, Jun, Pan, Lei, Han, Qing-Long, Chen, Chao, Wen, Sheng, Xiang, Yang, 2021. Deep learning based attack detection for cyber-physical system cybersecurity: A survey. IEEE/CAA J. Autom. Sin. 9 (3), 377–391.

Zhijun, Wu, Wenjing, Li, Liang, Liu, Meng, Yue, 2020. Low-rate DoS attacks, detection, defense, and challenges: A survey. IEEE Access 8, 43920–43943.