IET Networks

WILEY

The Institution of Engineering and Technology

ORIGINAL RESEARCH OPEN ACCESS

An Effective Technique of Zero-Day Attack Detection in the Internet of Things Network Based on the Conventional Spike Neural Network Learning Method

Nadia Adnan Shiltagh Al-Jamali[1] | Ahmed R. Zarzoor[2] | H. S. Al-Raweshidy[3]

[1]Department of Computer Engineering, College of Engineering, University of Baghdad, Baghdad, Iraq | [2]Directorate of Inspection, Ministry of Health, Baghdad, Iraq | [3]Brunel University of London, London, UK

Correspondence: H. S. Al-Raweshidy (Hamed.Al-Raweshidy@brunel.ac.uk)

Received: 15 July 2025 | **Revised:** 16 September 2025 | **Accepted:** 3 October 2025

Handling Editor: Panagiotis Sarigiannidis

Funding: This research was supported by the Brunel University London.

Keywords: cryptography | internet of things | next generation networks

ABSTRACT

The fast evolution of cyberattacks in the Internet of Things (IoT) area, presents new security challenges concerning Zero Day (ZD) attacks, due to the growth of both numbers and the diversity of new cyberattacks. Furthermore, Intrusion Detection System (IDSs) relying on a dataset of historical or signature-based datasets often perform poorly in ZD detection. A new technique for detecting zero-day (ZD) attacks in IoT-based Conventional Spiking Neural Networks (CSNN), termed ZD-CSNN, is proposed. The model comprises three key levels: (1) Data Pre-processing, in this level a thorough cleaning process is applied to the CIC IoT Dataset 2023, which contains both malicious and the most recent attack patterns in network traffic, ensuring data quality for analysis, (2) CSNN-based Detection, where outlier identification is conducted by comparing two dataset groups (the normal set and the attack set) within the same time period to enhance anomaly detection and (3) In the evaluation level, the detection performance of the proposed model is assessed by comparing it with two benchmark models: ZD-Deep Learning (ZD-DL) and ZD- Convolutional Neural Network (ZD-CNN). The implementation results demonstrate that ZD- CSNN achieves superior accuracy in detecting zero-day attacks compared to both ZD-DL and ZD-CNN.

1 | Introduction

The Internet of Things (IoT) has been a new technology in the last 2 decades, which includes the idea of data exchanged between Machine to Machines (M2M) without human intervention [1, 2]. The core idea is to convey the smart ability to ordinary human life utilising https://www.keaipublishing.com/en/journals/space-solar-power-and-wireless-transmission/ ubiquitous 'things' or devices within sensing, communication serviceability and computational, frequently denoted as IoT devices [3]. Besides the growing consequences of IoT devices in different domains such as smart cities, homes, streets, etc. (i.e., the real world is going into the uprising of being converted hooked on the smart world) [4]. Nevertheless, the implications of IoT devices face issues predominantly due to these devices having limited battery power and small calculation capability [5]. Therefore, IoT devices are disposed to cybersecurity hazards. Where, cybersecurity threat pretences are performed by persons with damaging intent, whose aim is to snip data, and cause destruction or loss to computing systems [6]. Furthermore, some threats exploit vulnerabilities that have not yet been discovered or disclosed—these are known as Zero-Day (ZD) attacks. Such attacks are rare and difficult to detect because they target unknown weaknesses that have not been publicly reported or patched. [7, 8].

Moreover, zero-day (ZD) attacks represent a novel class of threats that have not been previously identified, often designed to infiltrate or disrupt IoT network communications by exploiting unknown vulnerabilities. Their concealed nature makes them exceptionally difficult to detect using conventional intrusion detection methods, thereby granting attackers a significant advantage in executing full-scale compromises or gaining unauthorised control over the network [9, 10]. The ZD attack delectation methods can be classified into three techniques: Anomaly-based, Graph-based, and Machine Learning (ML) -based. In the anomaly-based technique [11], the detection of the attack is performed according to the comparison between two behaviours (normal and abnormal) during the time period to detect ZD attacks in the network. The graph-based method [12], the graph model is used to represent and compare the behaviour of the IoT devices' communication in two cases: ordinary and malicious (according ratio of IoT network attack) graphically to identify ZD attacks in the network. While, in the ML-based technique [13], the ML approaches are used to identify IoT vulnerability to improve the detection of ZD attacks. Where vulnerability identification and classification of the IoT network are reliant on determining characteristics of varied IoT devices which require to be painstakingly examined.

One of the major challenges in detecting Zero-Day (ZD) attacks is achieving reliable performance in terms of false positive (FP) and false negative (FN) rates. Traditional machine learning and deep learning models often struggle to generalise to novel or evolving threats. This leads to *high FN rates*, where subtle or previously unseen attacks go undetected, and *high FP rates*, where benign traffic is mistakenly flagged as malicious. In IoT environments, these issues are particularly problematic: undetected intrusions can compromise sensitive devices, while excessive false alarms can drain limited computational resources and disrupt normal operations.

These limitations stem from the static nature of conventional models, which often rely on fixed patterns and lack temporal sensitivity. They are not well-suited to capturing the dynamic and asynchronous behaviour of IoT traffic, especially under Zero-Day conditions.

To address these challenges, we propose ZD-CSNN, a novel detection model based on Convolutional Spiking Neural Networks. By leveraging neuromorphic computing principles, ZD-CSNN enhances temporal pattern recognition and anomaly sensitivity. This allows for more accurate differentiation between benign and malicious traffic—even when the attack patterns are previously unseen. While the model is generally applicable to network traffic analysis, it is specifically motivated by the constraints and demands of IoT environments. These include limited processing power, real-time responsiveness, and the need for lightweight, adaptive security solutions. Although full deployment details are discussed in the outlook section, the model's architecture is designed with edge applicability in mind. The use of the CICIoT2023 dataset further supports the relevance of our approach to current IoT security challenges.

This paper introduces a novel detection model, ZD-CSNN, designed to identify Zero- Day (ZD) attacks in IoT networks through outlier detection using a Convolutional Spiking Neural Network. The proposed approach is structured into three key levels: (1) Data Preprocessing — the CIC IoT Dataset 2023, which includes recent and malicious network traffic, is cleaned and prepared for analysis; (2) Outlier Detection via CSNN — anomalies are identified by comparing the full dataset with a subset containing known malicious traffic over a defined time window; and (3) Evaluation — the performance of ZD-CSNN is benchmarked against two existing models, ZD-DL and ZD-CNN, using metrics such as accuracy, F1-score, false positive rate (FP), and false negative rate (FN). The remainder of this paper is structured as follows: Section 2 reviews existing research on zero-day (ZD) threat detection techniques. Section 3 details the levels of the proposed ZD-CSNN method. Section 4 analyses and discusses the experimental results. Finally, Section 5 includes study conclusion.

2 | Related Works

This section provides an overview of state-of-the-art methodologies utilised to enhance the detection of zero-day attacks. Several studies have proposed machine learning-based techniques for ZD attack detection. The authors introduced a Federated Deep Learning (FDL)- based framework aimed at detecting Zero-Day (ZD) botnet attacks in IoT environments. Their approach initiates with the deployment of Deep Neural Networks (DNNs) on individual IoT edge devices to perform local traffic classification [14, 15]. To preserve data privacy and enhance scalability, the Federated Averaging algorithm is then applied to consolidate model updates from all devices. Through multiple rounds of communication between the central server and the edge nodes, a robust global DNN model is constructed, effectively enabling the detection of ZD botnet threats across the network. In ref. [16], the authors developed an Intrusion Detection System (IDS) aimed at reducing the high false negative (FN) rate in detecting Zero-Day (ZD) attacks by leveraging an autoencoder-based approach. The system operates in two levels: initially, a one-class Support Vector Machine (SVM) is used to identify potential ZD threats. In the subsequent level, outlier detection is performed using a combination of autoencoder and autodecoder techniques to enhance the accuracy and robustness of the IDS. [17] employed semi-supervised machine learning techniques to detect zero-day (ZD) threats, the central concept of their approach was the application of Benford's Law (BL) for feature selection. Benford's Law identifies naturally occurring numerical patterns—such as packet sizes—whose distribution tends to be nonuniform over time in legitimate network behaviour. By leveraging this principle, the study selected numeric features from IoT network traffic that deviated from expected patterns, thereby highlighting anomalies indicative of malicious activity and distinguishing between normal and abnormal (ZD) traffic.

In ref. [18] the researchers developed a new framework called 'Zero Shoot Leaning' (ZSL) to assess the implementation of ML-based network intrusion detection system (NIDS) in diagnosing ZD threats. The ZSL gauges how well ML-based NIDS can detect ZD threats utilising a group of semantic features from unseen threats. The ZSL consists of two stages: The learning stage, in which unique features of seen threats are extracted,

while in the second stage, a model links the connection between the seen and unseen threats formed to evaluate the classification process to identify the locating of malicious. Reference [19] presents a new ZD attack detection model based on a conventional neural network (CNN) and two regularisation methods (L1 and L2). In their model, a CNN is utilised as a classifier for traffic network threats, while L1 and L2 methods are used to mitigate the CNN models' overfitting problem. The study in ref. [20] introduces a novel model for detecting zero-day (ZD) traffic threats in network environments by leveraging the Heavy-Hitter (HH) concept in combination with a graph-based approach. The HH technique focuses on identifying dominant patterns within fixed-length data streams by filtering out low-frequency elements. The proposed model operates in two stages: the first stage involves signature generation, where HH analysis is applied to both high-level and low-level attack patterns to create representative signatures; the second stage is dedicated to performance evaluation, assessing the model's effectiveness in identifying and classifying ZD threats. In ref. [21], the authors improved a machine learning-based image approach for detecting zero-day (ZD) malware attacks by transforming IoT network traffic into visual representations. To optimise feature selection, they applied the Ant Colony Optimisation (ACO) algorithm, reducing the dimensionality of the data while enhancing the performance of Support Vector Machine (SVM) classifiers for malware detection. Reference [10] proposes a new technique-based ML to detect ZD threats on social networks (Twitter). They used ML techniques to specify additional similar text (either words or phrases) within potential threats via testing the content of tweets, besides the ability, to specify user identity and position. However, in this study, the CSNN model is utilised as a classifier for ZD attacks in the IoT network traffic. In this study, Convolutional Spiking Neural Networks (CSNNs) are used as classifiers to detect outliers in abnormal network traffic across two datasets the complete dataset and a malicious detection subset over a defined time period. Where, the outlier detection process is the comparison of the recent input of the data value in contradiction to the variation from the average of the previous value. To summarise all ZD detection-based ML approaches in comparison with this study, see Table 1.

3 | ZD-CSNN Method

In this study, a new ZD attack detection model based on CSNN to detect ZD threats in IoT network traffic. The model consists of three levels: data preprocessing, classification (i.e., CSNN), and performance evaluation level, see Figure 1.

3.1 | Data Preprocessing Level

The proposed model uses the CICIoT2023 dataset from the Canadian Institute for Cybersecurity [22]. This dataset is designed to support research on intrusion detection in IoT environments. It contains network traffic data from 105 IoT devices exposed to 32 different cyber-attack scenarios, see Table 2. Provided in CSV format, the dataset includes a wide range of features extracted from both benign and malicious traffic.

For this study, we selected seven common attack types: Distributed Denial of Service (DDoS), Denial of Service (DoS), brute-force, reconnaissance, spoofing, web-based attacks, and the Mirai botnet. These attacks were chosen because they are prevalent in real-world IoT threats and exhibit diverse behaviours that challenge traditional detection models.

We chose the CICIoT2023 dataset for its comprehensive coverage, variety of attack types, and relevance to current IoT security issues. Its realistic traffic patterns and inclusion of recent attack methods make it a strong benchmark for evaluating the ZD-CSNN model.

Before training, redundant data was removed to improve efficiency. We then applied Linear Discriminant Analysis (LDA) to reduce the dataset's size and extract the most relevant features. These features were used as input for the second level of our model, the CSNN.

The core idea behind the Linear Discriminant Analysis (LDA) technique is to project the training data onto a line (or lower-dimensional space) in such a way that samples from the same class are positioned as close together as possible, while samples from different classes are placed as far apart as possible. Thus, when a new data sample is introduced, it is projected onto the same line. The classification of the sample is then determined based on the position of its projection relative to the predefined class boundaries. Therefore, data points will be more differentiated after the dimension is reduced [23]. A Fisher-criteria is used with LDA to compute the projection vector (PV) by using Equation (1).

$$\acute{\varphi} = \underset{\varphi}{\operatorname{argmax}} \frac{\varphi^T S_b \varphi}{\varphi^T S_\varphi \varphi} \tag{1}$$

$$\mathrm{S_b} = \sum_{i=1}^{n} \left(\mathrm{a_i} - \mu_{\mathrm{b_i}}\right)\left(\left(\mathrm{a_i} - \mu_{\mathrm{b_i}}\right)^T\right) \tag{2}$$

$$S_\varphi = \sum_{c=1}^{m} n_c(\mu_c - \mu)\left((\mu_c - \mu)^T\right) \tag{3}$$

Where $\acute{\phi}$ represent the best projection vector (with coefficients ϕ) which maximises the rate of the S_b between-class- scattering (calculated by Equation 2) to S_ϕ the within-class-scattering (calculated by Equation 3). In Equation (2), a_i represent samples from a1, ..., an and b_i represent a_i class label from b1, ..., bn. μ_{bi} represent the mean of the class labels b_i. In Equation (3), μ_c is the sample mean of the cth class, m is the number of classes, μ is the mean of the entire sample, and n_c is the number of samples in c (i.e., data samples in the cth class). Thus, a computed value $\acute{\phi}$ Equation (1), gives a good PV when the eigenvector has a minimum eigenvalue ($S_\phi = S_b$). While, the big majority of the time, makes an individual PV is inadequate for recognising among several groups.

TABLE 1 | Overview of machine learning approaches for zero-day threat detection.

Study	Year	Detection technique	Summary
[15]	2022	Federated deep learning (FDL) and deep neural network (DNN)	— DNN is used to classify network traffic locally in each IoT edge device. — Federated average technique to aggregate the update results of each DNN in each IoT edge device. — A global DNN model is created when several communication sets occur between the central server and IoT edge devices
[16]	2022	DL (autoencoder) method and one class SVM	— A one-class support vector machine (SVM) is employed to detect ZD attacks — Auto-encoder and auto-decoder technique is used for the outliers' detection
[17]	2022	Semi-supervised ML methods and Benfor's Law (BL)	— The BL is used to select numeric data patterns subtype of IoT network traffic, which reveal a malicious behaviour between normal and abnormal (ZD network traffic).
[18]	2022	Zero shoot Leaning' (ZSL) to assess the implementation of ML-based network intrusion detection system (NIDS)	— The ZSL consists of two stages: The learning stage, in which unique features of seen threats are extracted — In the second stage, a model links the connection between the seen threats and unseen threats is formed to evaluate their classification and locating as malicious.
[19]	2022	Conventional neural network (CNN) and two regularisation methods (L1 and L2)	— A CNN is utilised as a classifier for traffic network threats — L1 and L2 methods are used to mitigate the CNN models' overfitting problem
[20]	2021	ML-based image model	The model is composed of two levels: Signature producing that generates signature via using heavy- hitter (HH) consists two models (high- and low-level attacks) — The second level is used to evaluate the model performance.
[21]	2023	ML image-based method	— The ant colony optimiser (ACO) technique is used to minimise the number of selected features that will be utilised in SVM classifiers' results (i.e., malware detection).
[10]	2023	TensorFlow ML	— They used ML techniques to specify additional similar text (either words or phrases) within potential threats via testing the content of tweets, besides the ability to specify user identity and position
Proposed study		ZD-CSNN	— CSNNs were as classifiers for outliers of abnormal traffic networks on two datasets (whole dataset and malicious detection dataset) during the time period.

3.2 | CSNN Classifier Level

In this level, a CSNN model is used as a time-based classifier to detect ZD threats. The Spiking Neural Network (SNN) is composed of multiple neuromorphic nodes, each acting like a biological neuron. Every node receives an input signal wave and generates an output signal wave independently of other nodes. These nodes have internal dynamics that evolve over time.

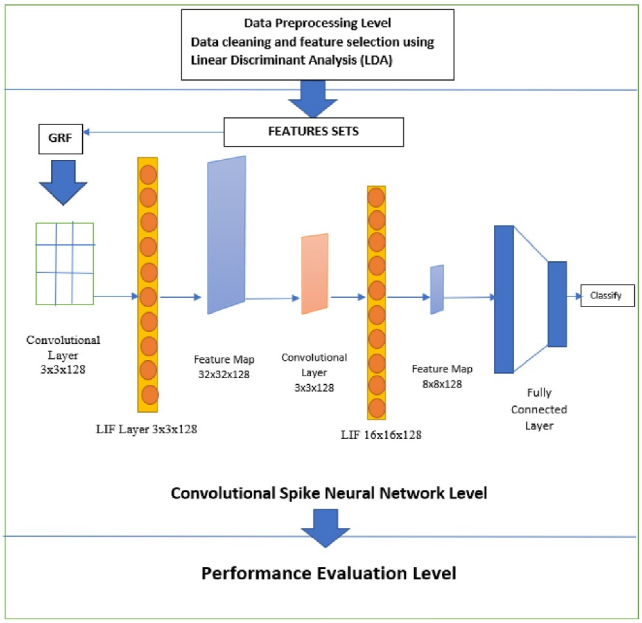

FIGURE 1 | Heading the zd-csnn method's three levels.

When a node exceeds its time threshold, it resets and lowers its membrane tolerance.

As a result, isolated input spikes may no longer trigger a response [24, 25]. Each neuromorphic node is connected to others through synaptic links, which carry weights.

These weights are adjusted during learning, using either supervised or unsupervised techniques.

The input signals are transformed into sequences of spikes, known as 'spiketrains,' through a process called encoding. In this study, the CSNN architecture includes two levels. Each level contains fully interconnected 'Leaky Integrate-and-Fire' (LIF) neurons and two conventional layers as illustrated in Figure 1.

The LIF neuron is arithmetically calculated by using Equation (4), where, ($\tau\_m = 10$), Vmem(t), and V_reset represent the membrane voltage and reset voltage respectively, while I(t) is the pre-neuron input current at (t) time that computes using Equation (5). Where, M number of pre-neuron, w_xy is the pre-synaptic weight between (neuron y pre-synaptic layer and x pos-synaptic layer), z_y (t) is the neuron response of pre-synaptic y. V [t] is the instant membrane voltage at time t, calculated by utilising Equations (6–8) [24, 25]. Also, V [t] represents the V(t) value before the neuromorphic node spikes free and after it's exciting, E [t] is the spike released from the neuron and V_th is the member threshold voltage.

$$\tau_m \frac{dV(t)}{dt} = -(Vmem(t) - V_{\text{reset}}) + I(t) \tag{4}$$

$$I(t) = \sum_{y=1}^{m} W_{xy} Z_y(t) \tag{5}$$

TABLE 2 | Description of attack types and dataset feature overview.

No	Attacks	Data description	Features
1	DDoS	ACK fragmentation	1. Flow_duration
		UDP flood	2. Header_length
		SlowLoris	3. Protocol type
		ICMP flood	4. Duration
		RSTFIN flood	5. Rate
		PSHACK flood	6. Srate
		HTTP flood	7. Drate
		UDP fragmentation	8. Fin_flag_number
		TCP flood	9. Syn_-flag_number
		SYN flood	10. Rst_flag_number
		SynonymousIP flood	11. Psh_flag_-number
2	Dos	TCP flood	12. Ack_flag_-number
		HTTP flood	13. Ece_-flag_number
		SYN flood	14. Cwr_flag_-number
		UDP flood	15. Ack_count
3	Brute-force	Dictionary brute force	16. Syn_count
4	Reconnaissance	Ping sweep	17. Fin_count
		OS scan	18. Urg_count
		Vulnerability scan	19. Rst_count
		Port scan	20. HTTP
		Host discovery	21. HTTPS
5	Spoofing	Arp spoofing	22. DNS
		DNS spoofing	23. Telnet
6	Web-based	Sql injection	24. SMTP
		Command injection	25. SSH
		Backdoor malware	26. IRC
		Uploading attack XSS	27. TCP
		Browser hijacking	28. UDP
7	Mirai botnet	GREIP flood	29. DHCP
		Greeth flood	30. ARP
			31. ICMP
			32. IPv
			33. LLC
		UDPPlain	

$$V[t] = f(V[t-1], I[t]) = V[t-1] + \frac{1}{\tau_m}\,(-V[t-1] - v_{\text{reset}}) + I[t] \tag{6}$$

$$[t] = f(V[t-1], i[t]) \tag{7}$$

$$E\,[t] = s\,(V\,[t] - V_{th}) \tag{8}$$

The first conventional layer has a size of 3 × 3 × 128. It takes the extracted features from level one as input and passes them to the first LIF (Leaky Integrate-and-Fire) layer, which has a size of 32 × 32 × 128. The output from this LIF layer is collected over time and processed using average pooling with a size of 2 × 2. This pooling operation performs minimal sampling on the feature map of size 32 × 32 × 128.

Next, the pooled data is fed into a second LIF layer with a size of 16 × 16 × 128. Its output is again collected and passed through another 2 × 2 average pooling layer, resulting in a feature map of size 8 × 8 × 128.

Figure 1 illustrates the conversion of emitted signals over time. In this process, LIF neuromorphic nodes handle spikes synchronously across time steps. The input wave tensor, with a channel size of 4 × 4, is divided into four time steps. A 3 × 3 kernel is applied concurrently across all time steps, producing a tensor that contains potential values for each step.

Spikes are stored cumulatively, so the corresponding potentials are also aggregated. The final feature map of size 8 × 8 × 128 is then passed to the next level. This level consists of two sets, each containing 128 LIF neuromorphic nodes. These nodes are responsible for categorising the wave signals.

To train the LIF nodes, a surrogate gradient method [26] is used. A sigmoid function is applied during backpropagation, as defined in Equation (9). For forward propagation, a step function is used to separate binary outputs (0 and 1s) from the LIF nodes. This step function is calculated using Equations (10) and (11), where ∅ represents the Heaviside step function and φ denotes the Dirac delta function.

$$\text{a(x)} = \text{sigmid}\,[\text{ax}] = \frac{1}{1 + e^{-\alpha x}} \tag{9}$$

$$S[t] = \varnothing(\text{V[t]} - V_{\text{th}}) \tag{10}$$

$$\frac{\partial \text{S}}{\partial \text{v}} = \varphi(V - V_{th}) \in \{0, 1\} \tag{11}$$

To train the value of the selected feature in the first level, to be used in the input layer a 'Gaussian Receptive Fields' GRF algorithm is utilised to encode information into firing times for the input layer by utilising Equation (12). Where the input value between value Vmin and Vmax, with σ is centred by using Equation (13). The spike timing is arranged between (0 to T), and the T value is computed in Equation (14). The σ is allocated by the passing points of the V with corresponding Gaussian summits: the *i*th input receives a spike at T-ai(V). So, when ai (V) > 0.01 and no spikes, then the nearest value of v to the σ will be obtained. Also, the Backpropagation method is used in the hidden layer to update weight to alleviate error, see Equation (15). Where, Wi, represents a new weight and (b) represents the learning rate (the minimum value) for the error function.

$$a_i(V) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{(x-\mu)^2}{\sigma^2}\right) \tag{12}$$

$$\mu_i = V_{\min} + (V_{\max} - V_{\min}) \cdot \frac{i}{n-1} \quad for\ i = 0\ to\ n-1 \tag{13}$$

$$T = \max(a_i(V)) \tag{14}$$

$$W_i = W_i - b\left(\frac{\partial \text{Error}}{\partial Wi}\right) \tag{15}$$

The outlier detection process in the proposed CSNN model is designed to identify anomalous patterns in IoT network traffic that may indicate Zero-Day (ZD) attacks. This is achieved by comparing the current input data values against the statistical variation of previously observed values. Specifically, the method calculates the deviation between the predicted value and the actual value, and then compares this deviation to a predefined threshold. If the deviation exceeds this threshold, the data point is classified as an outlier [27]. However, time-series data often contain irregular fluctuations referred to as 'infrequent hops' which can lead to classification errors due to slight misalignments between input and prediction timestamps. To mitigate this issue, the model incorporates a Q-function to measure the classification error using the *Squared Reconstruction Error (SRE)*, as defined in Equation (16) [28]. The mean, $\tilde{\mu}_t$ and standard deviation O_t of the SRE within a specified time window (W) are computed using the Welford online algorithm [29], which allows for efficient and accurate real-time updates. For each time step (t), the outlier score O_t is calculated independently for each feature dimension. These scores are then aggregated by selecting the maximum value across all dimensions at time t, resulting in a single outlier score. In this study, a threshold value of 0.8 adopted from ref. [28] is used. Any data point with an outlier score exceeding 0.8 is classified as an outlier; otherwise, it is considered normal. This approach enhances the model's ability to detect Zero-Day attacks by isolating subtle deviations that traditional methods may overlook.

3.3 | Evaluation Level

In this level, an evaluation process is implemented for three models: ZD-DL, ZD-CNN, and ZD-CSNN for performance measuring according to the accuracy, f1 score, FP rate, and FN rate metrics. The ZD-DL model, where detection of ZD attacks is implemented in two steps: a class Support Vector Machine (SVM) is employed to detect attacks set in general, in the second step, the outliers' detection is performed according to auto-encoder and auto-decoder technique by using two sets (attacks set from the first step and entire dataset) as input to the autoencoder. So, if there is a difference between the output of the auto-encoder, (which is used as input to the auto-decoder) and the output of the auto-decoder then outlier detection [16]. In ZD-CNN model, the CNN consists of three layers: convolutional (conv), pooling, and fully connected. The conv layer

handles the major portion of the network calculation load, it implements a dot product among two matrices (kernel and the restricted part of the received field). The pooling layer substitutes the network output at a particular position via extraction of a summary statistic of close-by output (i.e., it helps reduce computation and weight The fully connected layer maps the representation between the input and output [30]. In the ZD-CNN [19] the CNN is used as a classifier for ZD attacks. However, to measure the three model's performance the six parameters are used: accuracy, FP rate (FPR), FN rate (FNR), recall, precision, and f1 score are computed using Equations (16–23) [30, 31], where, TPR is the rate for the entire number of marked malicious devices divided by all number of malicious devices. FPR is the rate of the summation of devices incorrectly identified as malicious divided by the whole number of normal devices. TNR is the ratio of devices being correctly marked as anomalous devices. FNR is the ratio of the rate of anomalous devices to the entire ordinary device being wrongly marked as a normal device.

$$FPR = \left(\frac{FPR}{FPR + TN}\right) \times 100 \tag{16}$$

$$TNR = \left(\frac{TNR}{TNR + FPR}\right) \times 100 \tag{17}$$

$$FNR = \left(\frac{TPR + TNR}{FPR + TNR + TPR + FNR}\right) \times 100 \tag{18}$$

$$TPR = \left(\frac{TPR}{TPR + FNR}\right) \times 100 \tag{19}$$

$$Accuracy = \frac{TPR + TNR}{TPR + TNR + FNR + FPR} \tag{20}$$

$$Presision = \frac{TP}{TP + FD} \tag{21}$$

$$Recall = \frac{TP}{TP + FD} \tag{22}$$

$$F1\ Score = 2 \times \frac{Presision \times Recall}{Presision + Recall} \tag{23}$$

4 | Results and Discussions

The ZD-CSNN model is implemented on a laptop type Lenovo (CPU speed 2.8 GHz Intel Core i7, RAM 8 GB, and operation system MS Windows 10. Three scenarios have been conducted to, evaluate the performance of the three models: ZD-CSNN, ZD-CNN, and ZD- DL for the dataset CICIoT2023. The dataset captures network traffic generated during cyber-attacks targeting IoT devices. It includes feature-rich CSV files representing traffic data from 105 IoT devices subjected to 32 distinct attack types. Three experimental scenarios were implemented using Python (version 3.8). The ZD-CSNN model was developed using the snnTorch library, while TensorFlow, Keras and PyTorch were employed to implement the ZD-CNN and ZD-DL models, respectively. Also, the datasets are divided into 70% for training process and 30% for testing process, for three models and each model is independently performed on the same datasets. For the ZD-CNN, the number of input neurons is 120, hidden neurons 40, and activation function (Relu) see Table 3. To illustrate TPR against FPR, a 'receiver operating characteristic curve' (ROC) [32–34], has been used to plot the TPR against the FPR, and the area under the curve (AUC) falls between 0.0 and 1.0. Thus, AUC is employed to evaluate the performance of three classification models. So, a maximum AUC value means better performance, in the classification of the ZD attacks:

However, the implementation results for the ROC curve of burteforce attack AUC value for the three models: ZD-DL AUC = 0.8038, ZD-CNN AUC = 0.9628, and ZD-CSNN AUC = 0.9655) see Figure 2. While AUC values for: DDoS attack (ZD-DL AUC = 0.785, ZD- CNN AUC = 0.9655 and ZD-CSNN AUC = 0.99719) see Figure 3, DoS attack (ZD-DL AUC = 0.7902, ZD-CNN AUC = 0.9723 and ZD-CSNN AUC = 0.9885) see Figure 4.

TABLE 3 | Parameter specifications for the ZD-CNN model.

Parameter	Value
Input neuron	120
Hidden neuron	40
Activation function	Relu
Epochs	120/20
Batch size	64
Optimiser	Adam
Dropout rate	0.9

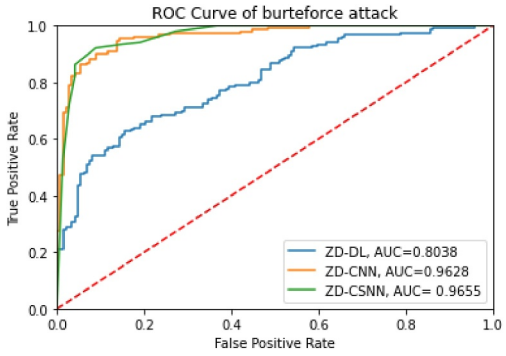

FIGURE 2 | AUC performance comparison for brute-force attack detection across three models.

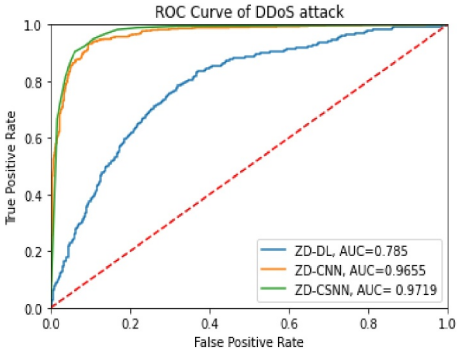

FIGURE 3 | AUC performance comparison for DDoS attack detection across three models.

Mirai-Botnet attack (ZD-DL AUC = 0.8131, ZD-CNN AUC = 0.9498, and ZD-CSNN AUC = 0.9521) see Figure 5, Web-Based attack (ZD-DL AUC = 0.779, ZD-CNN AUC = 0.9704, and ZD-CSNN AUC = 0.9663) see Figure 6, spoofing attack (ZD-DL AUC = 0.8377, ZD-CNN AUC = 0.9678, and ZD-CSNN AUC = 0.979) see Figure 7 and reconnaissance attack (ZD-DL AUC = 0.7982, ZD-CNN AUC = 0.9668, and ZD-CSNN AUC = 0.9682) see Figure 8. So, the ZD-CSNN gives better performance in the detection of attacks bureforce, DDoS, DoS, Mirai -Botnet, spoofing and reconnaissance in comparison with ZD-CNN and ZD-DL model. While, ZD-CNN performance in detection Web-Based attack in contrast with ZD-CSNN and ZD-DL model. The accuracy detection for three model: Burte-force (*ZD-DL* = 88.70, *ZD-CNN* = 96.34, *ZD-CSNN* = 97.12), DDoS (*ZD-DL* = 89.89, *ZD-CNN* = 98.88, *ZD-CSNN* = 99.69), DoS (*ZD-DL* = 89.89, *ZD-CNN* = 97.70, *ZD- CSNN* = 98.88), Mirai-Botnet (*ZD-DL* = 80.89, *ZD-CNN* = 95.89, *ZD-CSNN* = 96.99), Web- Based (*ZD-DL* = 78.96, *ZD-CNN* = 97.88, *ZD-CSNN* = 95.90), Spoofing ((*ZD-DL* = 77.66, *ZD-CNN* = 94.97, *ZD-CSNN* = 96.89), reconnaissance (*ZD-DL* = 79.80, *ZD-CNN* = 95.77, *ZD-CSNN* = 96.69). The model ZD-CSNN gives high accuracy of detection in comparison with ZD-CNN and ZD-DL, see Table 4.

For precision, recall and f1-score metrics for two groups: (normal and attack): The normal group includes all the dataset without any classification (detection) during a time period, while the attack group contains the dataset that has been

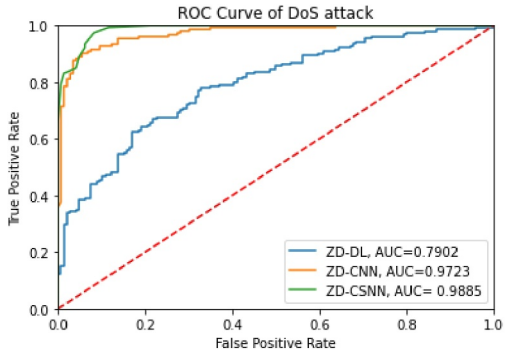

FIGURE 4 | AUC performance comparison for DoS attacks across three models.

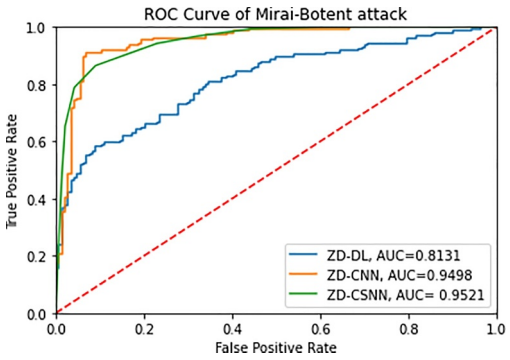

FIGURE 5 | AUC performance comparison for Mirai-botnet attacks across three models.

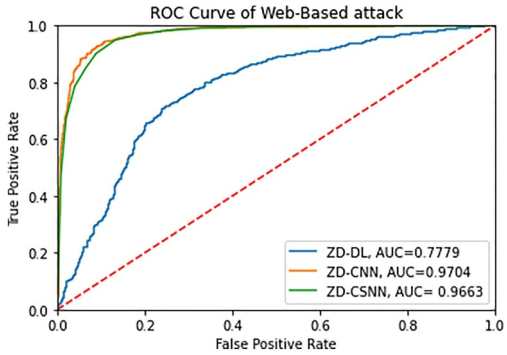

FIGURE 6 | AUC Performance comparison for web-based attack across three models.

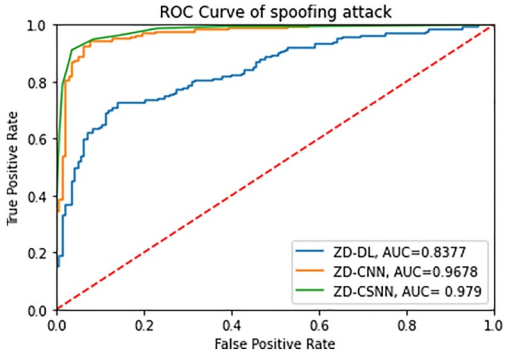

FIGURE 7 | AUC Performance comparison for spoofing attack across three models.

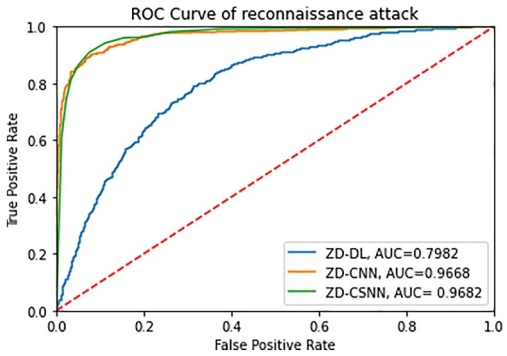

FIGURE 8 | AUC Performance comparison for reconnaissance attack across three models.

TABLE 4 | Comparative summary of detection accuracy across three models.

	Accuracy		
Attacks detection	**ZD-DL**	**ZD-CNN**	**ZD-CSNN**
Burteforce	88.70	96.34	97.12
DDoS	89.89	98.88	99.69
DoS	89.90	97.70	98.88
Mirai-botnet	80.89	95.89	96.99
Web-based	78.96	97.88	95.90
Spoofing	77.66	94.97	96.89
Reconnaissance	79.80	95.77	96.69

classified by using three models during the same time period of normal group. The percentage results of precision, recall and f1-score parameters have shown that the ZD-CSNN model gives high precision, recall, and f1-score values for Burteforce attack: precision (98.01, 98.77), recall (98.55,98.96) and f1-score (98.28, 98.86) in comparison with ZD-CNN (precision (96.66, 97.86), recall (98.31, 97.30) and f1-score (97.48, 97.58) and ZD-DL (precision (80.7, 79.31), recall (81.82, 78.82) and f1-score (81.26, 79.06) see Table 5. For DDoS attack, the ZD-CSNN model gives high precision, recall, and f1-score values: precision (96.01, 97.28), recall (97.15, 98.12) and f1-score (96.58, 97.70) in comparison with ZD-CNN (precision (94.61, 95.26), recall (97.66, 96.88) and f1-score (96.11, 96.06) and ZD-DL (precision (77.34, 78.22), recall (79.22, 79.12) and f1-score (78.27, 78.67) see Table 6. For DoS attack, the ZD-CSNN model gives high precision, recall, and f1-score values: precision (95.89, 96.88), recall (97.45, 96.55), and f1-score (96.66, 96.71) in comparison with ZD-CNN (precision (96.60, 96.26), recall (97.66, 96.88) and f1-score (95.22, 96.18) and ZD-DL (precision (70.34, 76.22), recall (78.34, 80.32) and f1-score (74.12, 78.22) see Table 7. For the Mirai-Botnet attack, the ZD-CSNN model gives high precision, recall, and f1-score values: precision (95.89, 96.12), recall (96.4, 97.58), and f1-score (96.14, 96.84) in comparison with ZD-CNN (precision (95.12, 95.55), recall (96.78, 96.18) and f1-score (95.94, 95.86) and ZD-DL (precision (73.13, 76.55), recall (77.13, 79.12) and f1-score (75.08, 77.81) see Table 8.

While, in the Web-Based attack, the ZD-CNN model gives high precision, recall, and f1-score values: precision (96.01, 94.22), recall (95.47, 96.14), and f1-score (95.74, 95.17) in comparison with ZD-CSNN (precision (95.12, 94.98), recall (96.09, 95.24), f1-score (95.60, 95.11), and ZD-DL (precision (69.23, 66.47), recall (70.13, 68.92) and f1-score (69.68, 67.67) see Table 9.

For spoofing attacks, the ZD-CSNN model gives high precision, recall and f1-score values: precision (96.44, 95.78), recall (94.39, 96.84) and f1-score (95.40, 96.31) in comparison with ZD-CNN (precision (95.91, 95.29), recall (93.97, 95.14) and f1-score (94.93, 95.21) and ZD-DL (precision (68.29, 70.16), recall (73.53, 78.12) and f1-score (70.81, 73.93) see Table 10. For reconnaissance attack, the ZD-CSNN model gives high precision, recall, and f1-score values: precision (94.89, 95.97), recall (96.98,96.78) and f1-score (95.92, 96.37) in comparison

TABLE 5 | Percentage-based detection results for brute-force attack across three models.

	Precision		Recall		F1-score	
Model	**Normal**	**Attack**	**Normal**	**Attack**	**Normal**	**Attack**
ZD-DL	80.7	79.31	81.82	78.82	81.26	79.06
ZD-CNN	96.66	97.86	98.31	97.30	97.48	97.58
ZD-CSNN	98.01	98.77	98.55	98.96	98.28	98.86

TABLE 6 | Percentage-based detection results for DDoS attack across three models.

	Precision		Recall		F1-score	
Model	**Normal**	**Attack**	**Normal**	**Attack**	**Normal**	**Attack**
ZD-DL	77.34	78.22	79.22	79.12	78.27	78.67
ZD-CNN	94.61	95.26	97.66	96.88	96.11	96.06
ZD-CSNN	96.01	97.28	97.15	98.12	96.58	97.70

TABLE 7 | Percentage-based detection results for DoS attack across three models.

	Precision		Recall		F1-score	
Model	**Normal**	**Attack**	**Normal**	**Attack**	**Normal**	**Attack**
ZD-DL	70.34	76.22	78.34	80.32	74.12	78.22
ZD-CNN	96.60	96.26	95.22	96.18	95.91	96.22
ZD-CSNN	95.89	96.88	97.45	96.55	96.66	96.71

TABLE 8 | Percentage-based detection results for Mirai-Botnet attack across three models.

	Precision		Recall		F1-score	
Model	**Normal**	**Attack**	**Normal**	**Attack**	**Normal**	**Attack**
ZD-DL	73.13	76.55	77.13	79.12	75.08	77.81
ZD-CNN	95.12	95.55	96.78	96.18	95.94	95.86
ZD-CSNN	95.89	96.12	96.4	97.58	96.14	96.84

TABLE 9 | Percentage-based detection results for web-based attack across three models.

	Precision		Recall		F1-score	
Model	**Normal**	**Attack**	**Normal**	**Attack**	**Normal**	**Attack**
ZD-DL	69.23	66.47	70.13	68.92	69.68	67.67
ZD-CNN	96.01	94.22	95.47	96.14	95.74	95.17
ZD-CSNN	95.12	94.98	96.09	95.24	95.60	95.11

TABLE 10 | Percentage-based detection results for spoofing attack across three models.

	Precision		Recall		F1-score	
Model	**Normal**	**Attack**	**Normal**	**Attack**	**Normal**	**Attack**
ZD-DL	68.29	70.16	73.53	78.12	70.81	73.93
ZD-CNN	95.91	95.29	93.97	95.14	94.93	95.21
ZD-CSNN	96.44	95.78	94.39	96.84	95.40	96.31

TABLE 11 | Percentage-based detection results for reconnaissance attack across three models.

	Precision		Recall		F1-score	
Model	**Normal**	**Attack**	**Normal**	**Attack**	**Normal**	**Attack**
ZD-DL	62.97	65.16	66.67	69.72	64.77	67.36
ZD-CNN	92.65	94.78	95.51	95.46	94.06	95.12
ZD-CSNN	94.89	95.97	96.98	96.78	95.92	96.37

with ZD-CNN (precision (92.65, 94.78), recall (95.51, 95.46) and f1-score (94.06, 95.12) and ZD-DL (precision (62.97, 65.16), recall (66.67, 69.72) and f1-score (64.77, 67.36) see Table 11.

5 | Conclusion

In this paper, a novel model is proposed for detecting zero-day (ZD) attacks in IoT network traffic using the CSNN learning method. The model is structured into three key levels: (1) Data Preprocessing, where CIC IoT Dataset 2023 undergoes a thorough cleaning process to remove inconsistencies and ensure high-quality data. This dataset contains both malicious activities and the latest attack patterns in network traffic. (2) CSNN-based Detection, in this level, a outlier detection is conducted by comparing two dataset groups: the normal set, which includes the entire dataset, and the attack set, which comprises classified malicious activities during the same time period. (3) performance evaluation, where, the effectiveness of the proposed model is evaluated against two benchmark models, ZD-DL and ZD-CNN, to assess its detection capability.

In future research, we aim to enhance the model's detection capabilities for web-based attacks, which currently present a challenge due to their less temporally dynamic nature. We also plan to integrate real-time detection mechanisms to support on-the-fly threat identification in live IoT environments. Beyond performance improvements, a key focus will be evaluating the computational efficiency and deployment feasibility of ZD-CSNN on resource-constrained edge devices. This includes analysing model size, energy consumption, and inference latency to better understand the trade-offs between detection accuracy and system overhead. These factors are essential for ensuring scalability in low-power IoT settings, where processing capacity and battery life are limited.

As a practical example, ZD-CSNN could be deployed in smart home environments to monitor network traffic from connected devices such as smart thermostats, cameras, and voice assistants. By identifying anomalous patterns indicative of zero-day attacks, the model could trigger early alerts and isolate compromised devices before damage spreads. This kind of proactive defence is especially valuable in homes with limited technical oversight, where automated security solutions are critical.

Additionally, we intend to expand our evaluation to include emerging IoT datasets and adversarial scenarios, enabling a more comprehensive assessment of the model's robustness and generalisability across diverse threat landscapes.

Author Contributions

Nadia Adnan Shiltagh Al-Jamali: conceptualization, data curation, formal analysis, investigation, methodology, project administration, resources, software, supervision, validation, visualization, writing – original draft, writing – review and editing. **Ahmed R. Zarzoor:** conceptualization, data curation, formal analysis, investigation, methodology, project administration, resources, software, validation, visualization, writing – original draft, writing – review and editing. **H. S. Al-Raweshidy:** conceptualization, data curation, formal analysis, funding acquisition, investigation, methodology, project administration, resources, software, supervision, validation, visualization, writing – original draft, writing – review and editing.

Acknowledgements

The authors would like to appreciate all the excellent suggestions of anonymous reviewers to enhance the quality of this paper.

Conflicts of Interest

The authors declare no conflicts of interest.

Data Availability Statement

The data that support the findings of this study are available from the corresponding author upon reasonable request.

References

1. R. Djehaiche, S. Aidel, A. Sawalmeh, N. Saeed, and A. H. Alenezi, "Adaptive Control of IoT/M2M Devices in Smart Buildings Using Heterogeneous Wireless Networks," *IEEE Sensors Journal* 23, no. 7 (2023): 7836–7849, https://doi.org/10.1109/jsen.2023.3247007.

2. M. Naveed, F. Arif, S. M. Usman, et al., "A Deep Learning-Based Framework for Feature Extraction and Classification of Intrusion Detection in Networks," *Wireless Communications and Mobile Computing* 2022 (2022): 1–11: Article ID 2215852, https://doi.org/10.1155/2022/2215852.

3. M. W. Iqbal, K. Khaliq, N. A. Al-Dmour, M. Aqeel, N. Ali, and K. Hamid, "Internet of Things (IoT) in Smart Cities: A Statistical Survey," in *2023 International Conference on Business Analytics for Technology and Security (ICBATS)*, (2023), 1–6.

4. R. Vilas Kolhe, P. William, P. Yawalkar, D. Paithankar, and A. Pabale, "Smart City Implementation Based on Internet of Things Integrated With Optimization Technology," *Measurement: Sensors* 27 (2023): 1–6: Article ID 100789, https://doi.org/10.1016/j.measen.2023.100789.

5. R. Dallaev, T. Pisarenko, D. Ţălu, D. Sobola, J. Majzner, and N. Papež, "Current Applications and Challenges of the Internet of Things," *New Trends in Computer Sciences* 1, no. 1 (2023): 51–61, https://doi.org/10.3846/ntcs.2023.17891.

6. U. Tariq, I. Ahmed, A. Bashir, and K. Shaukat, "Critical Cybersecurity Analysis and Future Research Directions for the Internet of Things: A Comprehensive Review," *Sensors* 23, no. 4117 (2023): 1–46, https://doi.org/10.3390/s23084117.

7. P. Anand, Y. Singh, and A. Selwal, "Learning-Based Techniques for Assessing Zero-Day Attacks and Vulnerabilities in IoT," in *Recent Innovations in Computing*, eds. P. K. Singh, Y. Singh, M. H. Kolekar, A. K. Kar, and P. J. S. Gonçalves, Vol. 832 (Springer, 2022), 497–504: Lecture Notes in Electrical Engineering, https://doi.org/10.1007/978-981-16-8248-3_41.

8. B. I Mukhtar, M. S. Elsayed, A. D. Jurcut, and M. A. Azer, "IoT Vulnerabilities and Attacks: SILEX Malware Case Study," *Symmetry* 15, no. 11 (2023): 1–26: Article ID 1978, https://doi.org/10.3390/sym15111978.

9. Y. Guo, "A Review of Machine Learning-Based Zero-Day Attack Detection: Challenges and Future Directions," *Computer Communications* 198 (2023): 175–185, https://doi.org/10.1016/j.comcom.2022.11.001.

10. A. E Topcu, Y. I. Alzoubi, E. Elbasi, and E. Camalan, "Social Media Zero-Day Attack Detection Using TensorFlow," *Electronics* 12, no. 17 (2023): 1–20: 3554, https://doi.org/10.3390/electronics12173554.

11. J. Araya and H. Rifà-Pous, "Anomaly-Based Cyberattacks Detection for Smart Homes: A Systematic Literature Review," *Internet of Things* 22 (2022): 1–28: Article ID 100792, https://doi.org/10.1016/j.iot.2023.100792.

12. M. Kalinin, E. Zavadskii, and A. Busygin, "A. "A Graph-Based Technique for Securing the Distributed Cyber-Physical System Infrastructure"," *Sensors* 23 (2023): 1–15: Article ID 8724, https://doi.org/10.3390/s23218724.

13. S. Bin Hulayyil, S. Li, and L. Xu, "Machine-Learning-Based Vulnerability Detection and Classification in Internet of Things Device Security," *Electronics* 12, no. 18 (2023): 1–24: Article ID 3927, https://doi.org/10.3390/electronics12183927.

14. L. Wawrowski, A. Białas, A. Kajzer, et al., "Anomaly Detection Module for Network Traffic Monitoring in Public Institutions," *Sensors (Basel)* 23, no. 6 (2023): 1–28: Article ID 2974, https://doi.org/10.3390/s23062974.

15. S. I. Popoola, R. Ande, B. Adebisi, G. Gui, M. Hammoudeh, and O. Jogunola, "Federated Deep Learning for Zero-Day Botnet Attack Detection in IoT-Edge Devices," *IEEE Internet of Things Journal* 9, no. 5 (2022): 3930–3944, https://doi.org/10.1109/jiot.2021.3100755.

16. H. Hindy, R. Atkinson, C. Tachtatzis, J.-N. Colin, E. Bayne, and X. Bellekens, "Utilising Deep Learning Techniques for Effective Zero-Day Attack Detection," *Electronics* 9, no. 10 (2020): 1–16: Article ID1684, https://doi.org/10.3390/electronics9101684.

17. I. Mbona and J. H. P. Eloff, "Detecting Zero-Day Intrusion Attacks Using Semi-Supervised Machine Learning Approaches," *IEEE Access* 10 (2022): 69822–69838, https://doi.org/10.1109/access.2022.3187116.

18. M. Sarhan, S. Layeghy, M. Gallagher, and M. Portmann, "From zero-shot Machine Learning to Zero-Day Attack Detection," *International Journal of Information Security* 22, no. 4 (2022): 947–959, https://doi.org/10.1007/s10207-023-00676-0.

19. B. I. Hairab, M. Said Elsayed, A. D. Jurcut, and M. A. Azer, "Anomaly Detection Based on CNN and Regularization Techniques Against Zero-Day Attacks in IoT Networks," *IEEE Access* 10 (2022): 98427–98440, https://doi.org/10.1109/access.2022.3206367.

20. V. Kumar and D. Sinha, "A Robust Intelligent Zero-Day Cyber-Attack Detection Technique," *Complex & Intelligent Systems* 7, no. 5 (2021): 2211–2234, https://doi.org/10.1007/s40747-021-00396-9.

21. A. El-Ghamry, T. Gaber, K. Mohammed, and A. E. Hassanien, "Optimized and Efficient Image- Based IoT Malware Detection Method," *Electronics* 12, no. 70 (2023): 1–21, https://doi.org/10.3390/electronics12030708.

22. CIC IoT Dataset 2023, (2025), https://www.kaggle.com/datasets/madhavmalhotra/unb-cic-iot-dataset/data.

23. G. Singh, Y. Pal, and A. Dahiya, "Classification of Power Quality Disturbances Using Linear Discriminant Analysis," *Applied Soft Computing* 138 (2023): 110181: Article ID 110181, https://doi.org/10.1016/j.asoc.2023.110181.

24. A. R. Zarzoor, N. A. Al-Jamali, and D. A. Abdul Qader, "Intrusion Detection Method for Internet of Things Based on the Spiking Neural Network and Decision Tree Method," *International Journal of Electrical and Computer Engineering (IJECE)* 13, no. 2 (2023): 2278–2288, https://doi.org/10.11591/ijece.v13i2.pp2278-2288.

25. A. R. Zarzoor, N. A. Al-Jamali, and I. R. K. Al-Saedi, "Traffic Classification of IoT Devices by Utilizing Spike Neural Network Learning Approach," *Mathematical Modelling of Engineering Problems* 10, no. 2 (2023): 639–646, https://doi.org/10.18280/mmep.100234.

26. A. Bittar and P. N. Garner, "A Surrogate Gradient Spiking Baseline for Speech Command Recognition," *Frontiers in Neuroscience* 16, no. 86589 (2022): 1–18, https://doi.org/10.3389/fnins.2022.865897.

27. P. S. Macią g, M. Kryszkiewicz, R. Bembenik, J. L. Lobo, and J. D. Ser, "Unsupervised Anomaly Detection in Stream Data With Online Evolving Spiking Neural Networks," *Neural Networks* 139 (2021): 118–139, https://doi.org/10.1016/j.neunet.2021.02.017.

28. D. Bäßler, T. Kortus, and G. Gühring, "Unsupervised Anomaly Detection in Multivariate Time Series With Online Evolving Spiking Neural Networks," *Machine Learning* 111, no. 4 (2022): 1377–1408, https://doi.org/10.1007/s10994-022-06129-4.

29. A. A. Efanov, S. A. Ivliev, and A. G. Shagraev, "Welford's Algorithm for Weighted Statistics," in *2021 3rd International Youth Conference on Radio Electronics, Electrical and Power Engineering (REEPE)* (2021), 1–5.

30. S. P and R. R, "A Review of Convolutional Neural Networks, Its Variants and Applications," in *2023 International Conference on Intelligent Systems for Communication, IoT and Security (ICISCoIS)*, (2023), 31–36.

31. A. R. Zarzoor and T. M. J. Abbas, "Securing Data Conveyance for Dynamic Source Routing Protocol by Using SDSR-ANNETG Technique," in *Machine Intelligence for Smart Applications*, eds. A. Adadi and S. Motahhir, Vol. 1105 (Springer, 2023), 213–225: Studies in Computational Intelligence, https://doi.org/10.1007/978-3-031-37454-8_11.

32. S. H. F. El-Moussa, N. Komninos, and R. Muttukrishnan, "A Systematic Review of Data- Driven Attack Detection Trends in IoT," *Sensors* 23, no. 7191 (2023): 1–29, https://doi.org/10.3390/s23167191.

33. M. Uppal, D. Gupta, A. Mahmoud, et al., "Fault Prediction Recommender Model for IoT Enabled Sensors Based Workplace," *Sustainability* 15, no. 1060 (2023): 1–21, https://doi.org/10.3390/su15021060.

34. W. Lin and Y. Chen, "Robust Network Traffic Classification Based on Information Bottleneck Neural Network," *IEEE Open Access* 12 (2024): 150169–150179, https://doi.org/10.1109/access.2024.3477466.