Journal of Theoretical and Applied Information Technology
15th November 2025. Vol.103. No.21

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

A ROBUST EXPLAINABLE RECURRENT DEEP Q LEARNING FOR DETECTING MULTICLASS INTRUSIONS IN IOT

PADMASRI TURAKA[1] , SAROJ KUMAR PANIGRAHY[2]

[1]Research Scholar, VIT-AP University, School of Computer Science and Engineering, Amaravati, Andhra Pradesh, India,522241

[2]Professor, VIT-AP University, School of Computer Science and Engineering, Amaravati, Andhra Pradesh, India,522241

E-mail: [1]padmasri.turaka@gmail.com, [2]skp.nitrkl@gmail.com

ABSTRACT

The deployment of explainable Artificial Intelligence (XAI) in cybersecurity applications is gaining much attention these days, but more research is still needed to better understand how effective it is at identifying attack exteriors and trajectories. The growing applications of machine learning (ML) and deep learning (DL) models in cyber defense, particularly anomaly-based Intrusion Detection Systems (IDS), necessitate the interpretation and the justification of their predictions to investigate potential cyberattacks. Hence, in this framework, an effective explainable DL technique is introduced to classify multiple intrusion attacks in the IoT-WSN system. Initially, the input data are taken from the UNSW-NB15 dataset, and then Weighted Correlated Adaptive Min-Max normalization (WCAdapt-MMN) is utilized to remove unwanted null values. Finally, the pre-processed data are fed as input into a Recurrent Deep Q learning (RDQN) model for intrusion classification as Fuzzers, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Normal. Furthermore, four XAI models are investigated for enriched visualizations over the IDS: Shapley additive explanations (SHAP), Permutation Feature Importance, and Partial Dependence Plot (PDP). A Python simulation tool is utilized for the simulation process, and a freely accessible UNSW-NB15 dataset is considered for the training process. In the resultant section, the accuracy of 98.48%, Matthew's correlation coefficient (MCC) of 0.98, precision of 98.62%, recall of 98.6%, False positive rate (FPR) of 0.319, and computation time (CT) of 49s for enhancing the robustness against IoT attacks.

Keywords: *Intrusion Attack; Multi-Attack Detection; Internet of Things; Explainable Artificial Intelligence; Recurrent Deep Q Learning; Shapley Additive Explanations; Permutation Feature Importance; Partial Dependence Plot*

1. INTRODUCTION

Large-scale intelligent ecosystems, such as smart cities, have been enabled by emerging technologies like the Internet of Things (IoT). The proliferation of IoT devices is profoundly transforming modern society by providing continuous monitoring, adaptive control, and automated optimization of interconnected systems [1,2]. However, securing these rapidly evolving technologies remains a significant challenge. Misconfigurations, increasing system complexity, and incompatibilities between legacy and modern infrastructures often undermine the availability and stability of IoT networks. IoT ecosystems typically comprise numerous low-cost devices generating vast volumes of heterogeneous data across various protocols and use cases [3,4].

This high-dimensional, multimodal data requires advanced analytics to ensure efficient processing and timely decision-making. Artificial Intelligence (AI), and specifically Machine Learning (ML) and Deep Learning (DL), has emerged as a powerful solution due to its ability to detect complex patterns and deliver high performance in large-scale data environments [5,6]. While traditional cybersecurity measures—such as firewalls, identity verification, and malware protection—offer some protection [7,8], they often fall short against sophisticated and rapidly evolving attack strategies. The unique characteristics of IoT environments demand intrusion detection systems (IDS) tailored to these

settings [9,10]. IDS solutions enhanced by AI have shown strong results in identifying both known and unknown threats, including zero-day attacks [11,12]. Nevertheless, such models often suffer from high false positive rates and a "black-box" nature that hinders user trust and interpretability.

The ability to explain and justify AI-driven intrusion detection decisions is critical for security analysts, system operators, and end users. Although ML-based IDS models are adaptive and effective, they typically require large datasets and significant computational resources, which can limit their scalability in real-time IoT scenarios [13]. Furthermore, the opacity of many AI/DL models creates uncertainty, as cybersecurity professionals may not fully understand how specific threat classifications are made [14–16]. Explainable Artificial Intelligence (XAI) addresses this challenge by making AI models' predictions transparent and interpretable [17]. In the context of IDS, XAI can reveal the reasoning behind detection outcomes, highlight influential features, and support trust-building among stakeholders. Current DL-based IDS frameworks, however, still face limitations: they often fail to generalize to new attack types, struggle with heterogeneous IoT data, and provide incomplete or overly complex explanations [18,19]. Additionally, the heterogeneity of data that deviates from consistent standards includes an assortment of textual, acoustic, and visual data, often in a format ideal for the manufacturer [20, 21]. Therefore, the developed study introduces an effective explainable DL technique to classify multiple intrusion attacks in the IoT-WSN system.

1.1 Motivation

The IoT has become an integral part of our lives, with smart devices and sensors deployed in different areas such as healthcare, conveyance, and intelligent homes. Nonetheless, with the cumulative number of IoT devices connected to the internet, the risk of cyberattacks and intrusions has also escalated. Intruders can feat vulnerabilities in IoT devices to advance unauthorized access, steal sensitive data, or disrupt amenities. Detecting and preventing intrusions in IoT networks is critical to ensuring the security and integrity of the data transmitted by these devices. Explainable AI, a subset of artificial intelligence that focuses on providing translucent and interpretable results, can play a crucial role in effectively detecting multiclass intrusions in IoT networks. By offering enlightenment to the resolutions made by AI models, explainable AI can help security analysts understand why a particular intrusion is flagged, enabling them to take appropriate action. Furthermore, explainable AI can help in identifying the root cause of intrusions, improving the complete security deployment of IoT networks. Inspired by this, the developed study introduces an effective explainable DL technique to classify multiple intrusion attacks in the IoT-WSN system. Unlike prior works, this study is motivated by the dual necessity of high detection accuracy and interpretability while ensuring low computational cost for real-time IoT deployments. Previous methods often optimized one or two of these aspects but not all simultaneously. Additionally, few studies have explored preprocessing tailored to feature–label correlations, and most employed a single XAI approach, limiting the depth of interpretability. This work introduces a novel Weighted Correlated Adaptive Min-Max Normalization (WCAdapt-MMN) to preserve important feature relationships and employs a combination of KernelSHAP, PFI, and PDP for a richer, multi-perspective explanation of model predictions.

The key contributions of the developed framework are depicted as follows:

- Introduce an advanced Recurrent Deep Q-learning Network with Explainability (RDQN-XAI) specifically designed for the detection of diverse intrusion attacks in IoT networks, enhancing security protocols.
- Propose the Weighted Correlated Adaptive Min-Max Normalization (WCAdapt-MMN) method, which effectively normalizes datasets by removing unwanted null values and preserving essential data relationships, thereby improving the quality of inputs for intrusion detection.
- Implement the RDQN technique to accurately identify and classify various types of attacks including Fuzzers, DoS, Exploits, Generic attacks, Reconnaissance, Shellcode and Normal traffic thereby ensuring robust protection of user data in IoT environments.
- Augment the RDQN model with state-of-the-art interpretability techniques, including kernel Shapley Additive Explanations (kernelSHAP), Permutation Feature Importance, and Partial Dependence Plots (PDP), to provide users with clear insights into the decision-making process behind intrusion detection.
- Conduct a thorough validation of the developed model's effectiveness through various performance metrics, including accuracy,

Matthews Correlation Coefficient (MCC), precision, recall, False Positive Rate (FPR), computation time (CT), and compare results with existing conventional studies to demonstrate superiority and reliability.

- Foster increased user trust and system usability in IoT security solutions by addressing the critical need for transparency and explainability in AI-driven intrusion detection systems, ultimately enhancing the overall effectiveness of IoT security frameworks.

The remaining sections are organized as follows: Section 2 delineates the section about literature review. Section 3 deliberates on the suggested method. Section 4 presents the results and discussion, and Section 5 concludes the proposed framework.

2. LITERATURE SURVEY

Keshk et al. [22] defined the XAI-DL technique for identifying various intrusion attacks on IoTs. An XAI-IDS methodology for IoT networks was utilized in this study Then, the IDS uses an LSTM model to detect cyberattacks and provide context for the model's judgments. The LSTM model was trained and evaluated using a novel set of input features that were extracted using a SHAP, PDP, ICE, and PFI framework. Utilizing the NSL-KDD, UNSW-NB15, and TON-IoT datasets, the framework was verified. In analyzing the experimental part, accuracy, detection time, and interpretability feature plots were analyzed and distinguished from other studies. However, the interpretability of the model could be limited due to the complexity of LSTM networks, making it difficult to understand and trust the explanations provided by the XAI component.

Abou El Houda et al. [23] introduced the XAI-DL technique for detecting intrusions in IoT networks. Here, an XAI-powered framework was introduced that can identify intrusions/attacks in IoT networks while also interpreting crucial decisions made by DL-based IDSs. As a result, the DL-based IDS with a deep neural network (DNN) was emphasized to identify and anticipate IoT assaults in real-time. Then, DNN architecture, and the multiple XAI models (RuleFit and SHAP) were deliberated to increase confidence, transparency, and explanation of DL-based IDS's judgments to cyber security specialists. The detailed experiment results with well-known IoT threats demonstrate the effectiveness and explainability of the suggested architecture. Nevertheless, the lack of transparency and interpretability can limit the adoption of DNN-XAI for IoT-IDS in critical applications where trust and accountability remain questionable.

Sauka et al. [24] puts forth the XAI network IDS system using the DL technique for minimizing the adversarial attacks. Anomaly-based NIDS has three essential components: traffic procurement, feature engineering, and classification. The feature extractor receives a stream of packets from the monitoring network, extracts features, and sends them to the anomaly detector. This technique also includes an initial stage of data preprocessing to ensure that the data was utilized for DL-based NIDS to generate accurate predictions. After processing the data, the anomaly detector assigns a score based on the input received and the threshold and it distinguishes between harmful and benign inputs. In analyzing the experimental part, accuracy, ROC curve, and interpretability feature plots were analyzed and distinguished from other studies. Nevertheless, the need for continuous retraining and updating of the model to adapt to evolving IoT threats can be challenging and may require significant resources.

Prabhudev and Haripriya. [25] Contemplated the optimized DL technique for ID with XAI models. This study suggests a hybrid method for hyperparameter optimization that makes use of hyperband. Taking into account all of the attack types in the CIC-IDS2018 dataset, an accuracy of 98.58% was obtained than other frameworks. The suggested hybrid framework improves Network IDS performance by selecting an optimal set of parameters and utilizing XAI techniques like LIME and SHAP to comprehend model predictions. In analyzing the experimental part, accuracy, ROC curve, and interpretability feature plots were analyzed and distinguished from other studies. Nevertheless, the high computational requirements and adding more explainability techniques to these models can further increase the computational burden.

Patil et al. [26] established the ML-XAI for detecting various intrusions in IoT platforms. Initially, cleaning, normalizing, and SMOTE processes were performed in the preprocessing stage. Then, the various features were extracted from the correlated heat mapping process. Finally, DT, RF, SVM, and voting classifiers were utilized to classify multiple attacks effectively. Moreover, the LIME-based XAI model was utilized to explain the utilized model for the attack detection process. In analyzing the experimental part, accuracy, F-measure, recall, and interpretability feature plots were analyzed and distinguished from other

studies. Nevertheless, the computational complexity was high while processing larger data with black-box models.

Zegarra Rodr´ıguez et al. [27] studied a transformer-based DL model for identifying multiple attacks with XAI feature selection (FS). Initially, the missing values were imputed in the preprocessing stage. Finally, the attentive TabNet model was utilized to classify multiple attacks effectively. For tuning the model, the Optuna h scheme was introduced that minimizes the network complexities. Different databases like CIC-IDS2017, CIC-IDS2018, and CIC-DDoS2019 were utilized for the simulation process. In analyzing the experimental part, accuracy, F-measure, recall, and interpretability feature plots were analyzed and distinguished from other studies. Nevertheless, the efficacy of the model reduces when any new type of attack arises on the IoT network.

Bakhsh et al. [28] determined the DL model for identifying multiple attacks in the IoT-WSN system. Initially, cleaning, normalizing, and transformation processes were performed in the preprocessing stage. Then, the various features were extracted and balanced using the data balancing method. Finally, LSTM, feed-forward neural network FFNN, and random NN techniques were utilized to classify multiple attacks effectively. The experimentation was performed using the CIC-IoT2022 database and various measures like accuracy, F-measure, and recall were analyzed and distinguished from other studies. However, the long-term dependency problem was high while introducing new attack features for the execution process.

Gaspar et al. [29] outlined the XAI-IDS scheme for IoT systems on the MLP framework. Two XAI methods, LIME and SHAP, were selected to retrieve explanations for the results of a black-box model as part of an IDS solution performing intrusion detection on IoT devices, enhancing its interpretability. To validate these explanations, a perturbation analysis was conducted, attempting to obtain different classifications based on the features highlighted in the explanations. This approach allowed conclusions to be drawn about the negative impact of specific features on the model's results when present in the input data, making it easier for cybersecurity experts to analyze model outputs and support continuous model improvement. The perturbations additionally provided a performance comparison between LIME and SHAP. To assess the interpretability, increase, and explanations offered by each XAI method, a survey analysis was performed to enable direct comparison between the XAI techniques. However, if the datasets lack diversity or do not capture relevant attack vectors, the accuracy and interpretability of the model explanations may be compromised.

Gummadi et al. [30] presented the XAI framework for determining anomaly detection in IoT platforms. The framework comprises two primary components. First, an AI-based anomaly detection approach was proposed, adapting two classes of AI methods (single AI methods and ensemble methods) to identify anomalies in smart IoT environments. This anomaly detection focused on identifying unusual data patterns from deployed sensors or network traffic among IoT devices. Second, a feature importance analysis was conducted to pinpoint the key features enabling AI models to detect anomalies within IoT systems. For this analysis, seven different XAI methods were applied to extract significant features across various AI methods and attack types. For the IoT manufacturing dataset, the framework detected defect levels in sensor data, while for the IoT botnet attack dataset, it identified distinct attack classes stemming from various botnet attacks on the IoT network. Conducting feature importance analysis with multiple XAI methods can lead to redundant or conflicting feature importance scores.

2.1. Research Gaps

Recent research on intrusion detection in IoT systems has largely focused on applying deep learning and machine learning models to improve classification accuracy. In [22], employed an LSTM-based IDS with SHAP, PDP, ICE, and PFI explanations, achieving reasonable accuracy but facing interpretability challenges due to LSTM complexity. In [23] used a DNN with RuleFit and SHAP to enhance transparency, though limited real-time applicability was noted. In [24] developed a DL-based NIDS aimed at minimizing adversarial attacks, requiring frequent retraining to handle evolving threats. Other approaches, such as [25] introduced hyperband-optimized DNN with LIME and SHAP, achieved improved accuracy but at the cost of high computation. In [26] utilized ensemble ML classifiers with LIME-based explanations but struggled with scalability for large datasets. Transformer-based models like TabNet-IDS [27] provided strong accuracy but exhibited reduced efficacy for new attack types. Most of these studies relied on standard normalization (min–max, Z-score) and single explainability techniques, and many reported

higher false positive rates and longer computational times, making them less suitable for resource-constrained IoT environments.

2.2. Problem statement

The rapid proliferation of Internet of Things (IoT) devices has transformed numerous sectors, enhancing efficiency and connectivity. However, this expansion has also introduced significant security vulnerabilities, making the safeguarding of these devices and their interconnected networks a critical concern. Intrusion Detection Systems (IDS) are pivotal in identifying and mitigating potential threats within IoT environments. Yet, traditional IDS solutions frequently suffer from a lack of transparency and interpretability, which hampers user ability to understand the rationale behind detected intrusions. This gap in understanding can lead to mistrust in the system's efficacy and complicate decision-making processes in response to threats. To address these challenges, the integration of Explainable Artificial Intelligence (XAI) models in IDS for IoT environments is essential. XAI aims to provide clear and comprehensible explanations of AI-driven decision-making processes, thereby enhancing user trust and engagement. In the context of multiclass intrusion detection, XAI models can illuminate the features and patterns responsible for identifying various types of intrusions, empowering users to make informed decisions regarding threat responses. Despite ongoing research into AI-based IDS for IoT systems, a substantial number of these models still fall short in terms of transparency and interpretability. By incorporating explainability into intrusion detection methodologies, researchers can significantly improve the usability and reliability of IDS in IoT environments. Therefore, this study proposes the development of an effective explainable deep learning technique specifically designed to classify multiple intrusion attacks within IoT-WSN systems, aiming to bridge the gap between advanced detection capabilities and user understanding.

2.3. Research Questions

Based on the identified problem and the literature gaps, the study addresses the following research questions:

- How can a Recurrent Deep Q-learning Network (RDQN) be effectively utilized for multiclass intrusion detection in IoT-WSN systems?
- Can the integration of explainability techniques (e.g., KernelSHAP, PFI, PDP) improve the transparency and trustworthiness of intrusion detection decisions without compromising model performance?
- What is the impact of the proposed Weighted Correlated Adaptive Min-Max Normalization (WCAdapt-MMN) on data quality and model robustness compared to traditional normalization methods?
- How does the proposed RDQN-XAI framework perform in comparison to existing deep learning models (e.g., LSTM, CNN, GRU, Deep Q) in terms of accuracy, precision, recall, false positive rate, and computational time?
- To what extent can the proposed framework mitigate false positives and enhance real-time applicability for resource-constrained IoT environments?

3. PROPOSED METHODOLOGY

Currently, the wide variety of IoT programs has expanded melodramatically. IoT is made of a wide variety of different physical endpoints or sensors. These are connected primarily to the internet, and they can accumulate statistics from the encompassing environments and proportion them with one another. Security in the IoT discipline is becoming more challenging alongside the IoT industry development. It's far because of the heterogeneity of IoT structure on the accessed devices and a couple of conversation processes, in addition to the large extent of data being transmitted via the community. In the present scenario, the need for rationalization predictions via DL-based IDS arises as exclusive forms of better user service quality from the intrusion detection process. The need for XAI has been brought to deal with the rising prerequisites to interpret DL models in cyber defense solicitations. Figure 1 depicts the Workflow of the proposed method.

Journal of Theoretical and Applied Information Technology
15th November 2025. Vol.103. No.21

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

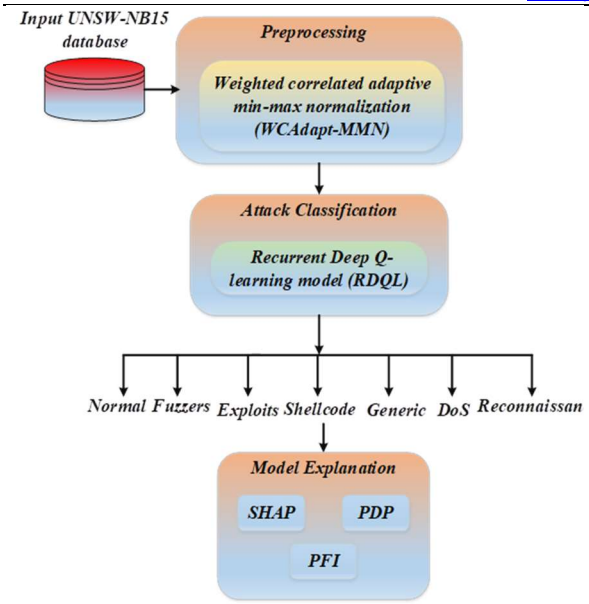


Figure 1: Workflow of the proposed method

Hence, in this framework, an effective explainable DL technique is introduced to classify multiple intrusion attacks in the IoT-WSN system. Initially, the input data are taken from the UNSW-NB15 dataset and then Weighted Correlated Adaptive Min-Max normalization (WCAdapt-MMN) is utilized to remove unwanted null values. Finally, the pre-processed data are fed as input into a Recurrent Deep Q learning (RDQN) model for intrusion classification as Fuzzers, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Normal. Furthermore, four explainable algorithms are investigated for improved visualizations over the IDS: SHAP, PFI, and PDP.

3.1 Preprocessing

The Weighted Correlated Adaptive Min-Max Normalization (WCAdapt-MMN) technique is designed in the preprocessing stage to address the limitations of standard min-max normalization [31], particularly in scenarios involving complex, non-linear data such as intrusion detection. Traditional min-max normalization scales data linearly within a specified range, but it lacks adaptability for non-linear data patterns and fails to emphasize the significance of features correlated with intrusion patterns. Here's a rigorous mathematical justification and derivation for WCAdapt-MMN, illustrating how it overcomes these limitations. The min-max normalization of a feature x in a dataset is using Equation (1).

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \quad (1)$$

Here, x_{min} and x_{max} signify the minimum and maximum values of x across the dataset. x' deliberates the normalized value of x, constrained to the *[0, 1]* range. This linear scaling method is straightforward but assumes a linear relationship among features and does not account for non-linear dependencies or correlations, often present in intrusion detection data.

The WCAdapt-MMN approach introduces weighted correlation to adaptively adjust the min-max normalization, taking into account the interdependencies among features. Let $X = [x_1, x_2, \ldots\ldots, x_m]$ be the set of all features, and let x_i be the feature we want to normalize. For each feature x_i, we calculate the weighted correlation coefficient ρ_i for intrusion detection relevance. This correlation coefficient is used to give higher importance to features that are strongly correlated with the target. Let y denote the label for intrusion detection and the correlation coefficient $\rho_{x,y}$ between x_i and y_i is computed using Equation (2)

$$\rho_{x,y} = \frac{\mathrm{cov}(x_i, y)}{\sigma_{x_i} . \sigma_y} \quad (2)$$

To ensure positivity, we denote the absolute value of ρx,y as $|\rho_{xi,y}|$. Features with higher correlation are given more weight in the normalization process. The WCAdaptMMN method adjusts the range of normalization by incorporating a weighted adaptive scaling factor α_i derived from the correlation coefficient using Equation (3)

$$\alpha_i = 1 + \lambda \left| \rho_{x_i, y} \right| \quad (3)$$

Here, λ Determines the tuning parameter that controls the influence of the correlation on the normalization. Higher values of λ increase the non-linear adaptation. The

adaptive scaling modifies the normalized value of x_i As follows:

$$x_i' = \frac{x_i - x_{\min}}{x_{i,\max} - x_{i,\min}} . \alpha_i \quad (4)$$

This scaling adjusts the influence of each feature based on its relevance, so highly correlated features have a broader normalization range, while weakly correlated features have a standard or contracted range. Combining the elements above, the Weighted Correlated Adaptive Min-Max Normalization (WCAdapt-MMN) for a feature x_i Can be expressed as:

$$x_i' = \frac{(x_i - x_{\min}).\left(1 + \lambda \left| \rho_{x_i,y} \right|\right)}{x_{i,\max} - x_{i,\min}} \quad (5)$$

This formula ensures that the range is adaptively scaled based on the correlation with the target label, addressing non-linear patterns. Features highly correlated with intrusion labels are emphasized, improving the sensitivity of the model to relevant patterns.

3.2. Intrusion Attack classification using the RDQN technique

The preprocessed data is then fed into the proposed recurrent deep Q-learning network (RDQN) technique for detecting various attacks accurately. The traditional Q-learning (QL) technique [32] is not effective when extensive attack instances are processed for the training process. The proposed RDQN technique minimizes error and time complexity with an adaptive feature learning mechanism involved in the execution process. A brief analysis of the developed RDQN technique is depicted below:

The QL is the general reinforcement learning scheme that creates a policy for the state. X-action A integration to Q-values $Q(X, A)$. The Q-values are generated from recursions passed via the search space and the deliberated predicted value by computing the action in the obtained states. At the final stage, the Q-values are updated for all state-action pairs, and this is encompassed via the agent using the mathematical formulation depicted as Equation (6).

$$Q(X, A) = Q(X, A) + \beta \left(z + \alpha \max_{\widehat{A}} Q\left(\widehat{X}, \widehat{A}\right) - Q(X, A) \right) \quad (6)$$

The DQN technique uses the LSTM model to assign Q-values to perform actions, presenting the prevailing state and combined weights and biases. The problem in DQN is different from CNN, in which the interactions are complicated and continue over varying time slots. Finally, the DQN model has dual structural changes that are not found in CNN, such as involvement of replay and parameter setting of the second model $\widehat{Q}$. The involvement replay indicates the utilization of replay memory. VThat saves the involvement tuples in the form $y_t = (X_t, A_t, z_t, X_{t+1})$. The involvement tuples are the instances that are obtained after accomplishing that joint state action z_t and the state correspondingly transitioned to X_{t+1}. In our problem, the state is deliberated using an ID with accompanying attack depth information, along with the stimuli distributed so far. This will work as follows.

3.2.1 Defining experience in replay buffer

Let the replay buffer be defined as y, which stores past experiences as sequences: using Equation (7)

$$\mathrm{y} = \{(\mathrm{X_t}, \mathrm{A_t}, \mathrm{z_t}, \mathrm{X_{t+1}})_{\mathrm{t=1}}^{\mathrm{T}}\} \quad (7)$$

Here, T Signifies the length of the episode sequence stored in the buffer. **Sampling sequences:** To train a DRQN model, we sample batches of experience sequences from the replay buffer. Let:

The batch of sequence B is randomly sampled from y using Equation (8)

$$\mathrm{B} = \left\{(X_t, A_t, z_t, X_{t+1})_{t=k}^{k+L-1}\right\}_{i=1}^{N} \quad (8)$$

Here, k Indicates the starting point of each sequence in the episode. Each sampled sequence in B has a length. L, and N Such sequences are included in the batch. Figure 2 encompasses the Architecture of the RDQN model.

Journal of Theoretical and Applied Information Technology
15th November 2025. Vol.103. No.21

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

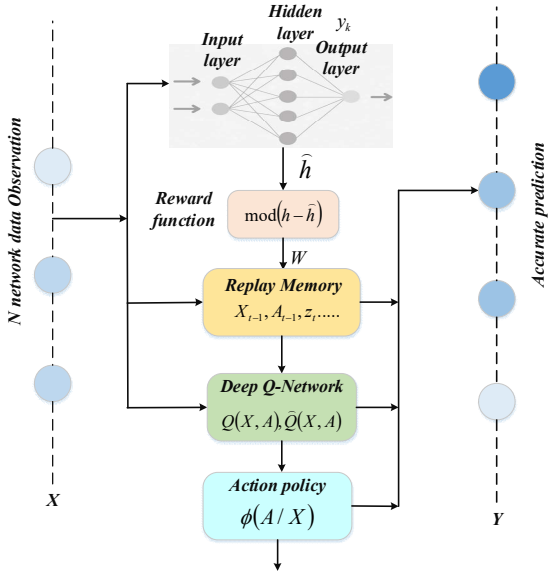


Figure 2: Architecture of RDQN model

3.2.2 LSTM State Initialization

The LSTM consists of two essential components: gating unit (GU) and memory unit (MU). The MU helps to store and transfer data whereas, the GU maintains the data flow regulation. The input gates (IG) deliberate the concern over transmitting new input into the MU. Simultaneously, the *tanh* function is computed to obtain vectors thereby upgrading the parameters within the MU. By utilizing the sigmoid function and previous memory states, this gate produces a value between the range 0 and 1. The outcome gate is assigned to discriminating the memory cell segments to send the corresponding hidden state. The mathematical expressions for the LSTM training process are deliberated in Equations (9), (10), (11), (12), and (13).

$$x_k = \sigma(W_{ix} i_k + W_{ih} h_{k-1} + W_{id} d_{k-1} + Bias_x), \quad (9)$$

$$f_k = \sigma\left(W_{if} i_k + W_{hf} h_{k-1} + W_{fd} d_{k-1} + Bias_f\right), \quad (10)$$

$$d_k = f_k d_{k-1} + x_k \tanh\left(W_{id} i_k + W_{hd} h_{k-1} + Bias_d\right), \quad (11)$$

$$y_k = \sigma\left(W_{yx} i_k + W_{yh} h_{k-1} + W_{yd} d_k + Bias_y\right), \quad (12)$$

$$h_k = y_k \tanh\left(d_k\right) \quad (13)$$

Here, σ indicates the sigmoid function, and x, f, d, and y represent the IG, forget gate, activated cell vector, and output gate respectively. Moreover, h and W represent the hidden trajectory, and the weights between two units respectively. However, there arises some complexity in accessing previous state information using the LSTM model. To tackle this issue, the Bidirectional long short-term memory (BiLSTM) model utilizes forward and backward LSTMs to process the incoming features in opposite directions. This duo-directed technique optimally controls the signals based on the current time step, which is shown in Figure 3.

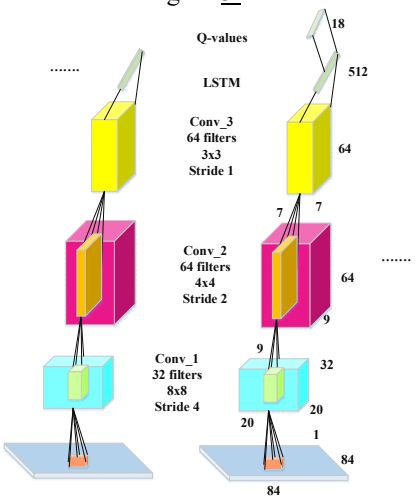


Figure 3: Detailed layer-wise Architecture of RDQN model

3.2.3 Q-Value calculation using LSTM hidden state

For each state X_t^i in the sequence, the Q -value for action A_t^i is calculated by applying a fully connected layer to the LSTM hidden state h_t^i :

$$Q_t^i = W_Q h_t^i + \mathrm{Bias}_Q \quad (14)$$

Here, Q_t^i deliberates the vector of Q -values for each action given the current state represented by the LSTM hidden state h_t^i. So, for each action $a \in A$, the Q-value is:

$$Q(X_t, A_t) = \left[Q_t^i\right]_a = W_{Q,a} h_t^i + \mathrm{Bias}_{Q,a} \quad (15)$$

The DQN is executed arbitrarily by choosing the tuples from V using X_t as the input and one-hot encoding of A_t multiplied by z_t For classes. The operationally matching model $\hat{Q}$ Is utilized to create the expected reward for the successive state.

Finally, the loss function can be mathematically formulated as,

$$l_k(\phi_k) = E(X_t, A_t, z_t, X_{t+1})\left[\left(m_k - Q(X_t, A_t; \phi_k)\right)^2\right] \quad (16)$$

The above Equation (16) helps to update k^{th} iteration value of ϕ and X_t indicates the non-lethal state and the parameter m_k can be formulated as Equation (17)

$$m_k = z + \alpha \max_{\hat{A}} \hat{Q}\left(X_{t+1}, A; \tilde{\phi}\right) \quad (17)$$

3.2.4 Mini-Batch preparation for training

After sampling, the mini-batch of experiences is prepared for training:

The replay buffer, therefore, facilitates efficient, stabilized training by reusing sequences of experiences and reducing the correlation between them, allowing the DRQN to learn from the temporal structure of sequential data. Figure 3 depicts the Detailed layer-wise Architecture of the RDQN model.

3.2.5 Computation complexity analysis

For an RDQN with an LSTM recurrent layer, the forward pass complexity is approximately. $O(T.h.(d+h)+h.m)$ whereas, T, d, h, and m Indicate the input sequence of length, input dimensionality, hidden layer size, and output size respectively. The training complexity per update step with BPTT and experience replay will be proportional to $O(T.h.N.(d+h)+N.h.m)$ where N Indicates the batch size. In general, the recurrent component $O(T.h.(d+h))$ dominates the complexity, especially in long sequences or large hidden states.

3.3. Explainable Artificial Intelligence Models

In this section, various XAI techniques like kernel Shapley Additive Explanations (kernelSHAP), Permutation Feature Importance (PFI), and Partial Dependence Plot (PDP) are implemented for visualizing the distinct predictions made by the developed model. These models are considered the global XAI schemes that assist in finding a customized set of particular input features that explain the various functionalities of the developed model effectively. A brief analysis of various XAI models is deliberated below.

3.3.1 KernelSHAP explanation

The Shapley value is a technique that incorporates the cooperative game theory method, and it has an efficient hypothetical foundation to evaluate the importance of each attack in a collaborative manner. It also determines the number of features that have contributed to that particular attack. Generally, every feature value deliberates the Shapley values, indicating its mean contribution to the model outcome. This value is calculated and added to the overall joint feature values for the set of features, and calibrated prediction can be mathematically formulated as Equation (18),

$$\theta_i(value) = \sum_{s \subseteq N} \frac{|s|\Gamma(n-|s|-1)}{\Gamma m}[value(s+i) - value(s)], i = 1, \ldots . n \quad (18)$$

here, s indicates the feature subsets utilized, $|s|$ deliberates the feature value, N indicates the total feature subsets, $n$encompasses the feature size, and $i$indicates the value of i$^{\text{th}}$ feature vector to be explained. Moreover, $value(s)$ and $value(s+i)$ signifies the function assigned to the subset $s$and subset s with i$^{\text{th}}$ feature respectively.

The Shapley values consist of four characteristics namely efficiency, symmetry, dummy, and additivity. The efficiency defines the contributed features that sum up the prediction for a single instance which is equal to the difference between the original output and average prediction. It can be formulated as Equation (19),

$$\sum_{j=1}^{n} \theta_j(value) = value(N) \quad (19)$$

In symmetry two features j^{th} and l^{th} subset $s$of feature are considered and it can be formulated as Equation (20),

$$value(s+j) = value(s+l), \quad s \subseteq N \setminus [j, l] \quad (20)$$

The above equation signifies that the two features j^{th} and l$^{\text{th}}$ subset $s$of feature contributes equally and the function θ is balanced which is represented as Equation (21),

$$\theta_j(value) = \theta_l(value) \quad (21)$$

In the dummy, the contributed features remain unchanged and its Shapley value must be similar to zero and it is deliberated as Equation (22), and (23),

$$value(s+j) = value(s) \quad (22)$$

$$\theta_j(value) = 0 \quad (23)$$

In additivity, the accuracy of integrating two functions $value$ and $\widehat{value}$, which is equal to the sum of distinct gains from every function for every j^{th} feature, can be expressed as Equation (24),

$$\theta_j\left(value + \widehat{value}\right) = \theta_j(value) + \theta_j\left(\widehat{value}\right) \quad (24)$$

The SHAP value can be measured by distinct sample instances and the explanation for a sample $y$can be calibrated using the expression given as Equation (25):

$$f\left(\widehat{x}\right) = \theta_0 + \sum_{j=1}^{n} \theta_j \widehat{x}_j, \qquad \widehat{x} \in [0,1]^n \quad (25)$$

Here, $n$encompasses the feature size, f indicates the explanation model, $\widehat{x}$signifies the complex features of the sample x, and θ_j $indicates the Shapley value for j^{th} feature of the sample y. The samples in SHAP are weighted to enhance the effectiveness of estimated Shapley values. The mathematical expression for kernel SHAP is depicted as Equation (26),

$$\pi_y\left(\hat{x}\right) = \frac{(n-1)}{\left(\binom{n}{|\hat{x}|}\right)|\hat{x}|\left(n-|\hat{x}|\right)} \quad (26)$$

Here, n encompasses the size of possible combinations, and $\widehat{x}$ signifies the total current features.

3.3.2 PFI explanation

The PFI is one of the model-agnostic schemes that computes the alteration in prediction error of the proposed model when permuting input feature values. This technique deliberates the feature relationship between all features and the target outcome. Hence, the model's error increases when the model contains the necessary features. Here, three major phases are involved by considering the trained model G, matrix features

$Z$and the desired outcome $q$and the error estimation $\delta(q, G)$. The PFI value for the j^{th} feature can be depicted as Equation (27),

$$s_{\geq} = \frac{e^{(permutation)}}{e^{(actual)}} \quad (27)$$

Finally, the features are arranged in descending order based on the obtained value of $s_{\geq}$.

3.3.3 PDP explanation

The PDP is another widely utilized visualization technique that helps to provide the relationship between the outcome of the black box model and the feature values by plotting visualizations. The PDP is the global scheme that stores all the information and defines the relationship between prediction and a particular feature. It can be mathematically formulated as Equation (28),

$$PD(z_s) = E(z_c)[G(z_s, z_c)] = \int f(z_s, z_c) dP(z_c) \quad (28)$$

Here, Gindicates the proposed model, $z_s \in s$deliberates the plotted input features, $z_c \in s$ signifies the input features f, and $dP(z_c)$signifies the marginal distribution of $z\_\{c\}$.

ICE Analysis opposed to PDP, which estimates how a model behaves but puts this estimation in the light of individual observations rather than averages across the whole data, the ICE method gives one distinct line for every observation reflecting in what way a prediction depends on a feature, using variations of the feature: Essentially, although PDP is an average of all these individual lines, ICE shines more light on the interaction that may be unique and exist between a feature and the prediction for each of the observations. PDP is only good when interactions between features are relatively weak because it suppresses the heterogeneous patterns caused by the interactions among features. This is the case with ICE because it can expose these heterogeneous patterns. For a set of observations $\left\{\left(x_s^{(i)}, x_c^{(i)}\right)\right\}_{i=1}^{k}$ ICE plots $g_s^{(i)}$ against $x_s^{(i)}$ for each of the k observations, showing each observation's response to the feature. Averaging all ICE curves then gives the PDP.

4. RESULTS AND DISCUSSION

The developed framework is processed and experimented with via the Python platform. The UNSW-NB15 dataset [33] is collected from the openly accessible Kaggle platform. It consists of combined practical synthetic and normal contemporaries of attack activities. A total of nine different types of attacks: Exploits, Generic,

Reconnaissance, Fuzzers, Shellcode, Analysis, Backdoors, DDoS, and worms are presented. Moreover, 49 different features are extracted using Argus and Bro-IDS tools based on twelve algorithms. A total of 2,54,0044 samples are illustrated along with the training and testing sets of 17,53,41 and 823,32 samples respectively. In the developed frameworks, a total of seven attack types Exploits, Generic, Reconnaissance, DoS, normal, shellcode, and Fuzzers are processed. The learning rate (LR) of 0.001, min-batch size of 32, the dropout rate of 0.2, and a total of 100 epochs are considered for the training process. The proposed method is processed under Intel(R) Core (TM) i5-4300M CPU with 4GB installed RAM using a 64-bit operating system. The number of permutations for each feature is about 10 and the input sequence length is about 62. Here, a multi-pass permutation strategy is performed.

4.1. Performance metrics

Performance indicators like Accuracy, Mathew's correlation coefficient (MCC), False Positive Rate (FPR), Precision, and Recall were scrutinized to better understand the developed method.

- **Accuracy:** It determines the model's overall detection accuracy, accounting for both TP and TN. It is calculated as Equation (29).

$$Accuracy = \frac{w+x}{w+x+y+z} \quad (29)$$

- **MCC:** A metric used to assess the validity of multi-attack detection, especially when there is an imbalance between the classes, and it is assessed using Equation (30),

$$MCC = \left(\frac{x \times w - z \times y}{\sqrt{(x+z)(x+y)(w+z)(w+y)}}\right) \quad (30)$$

- **•FPR:** False Positive Rate measures the proportion of negative instances that are incorrectly classified as positive by the model. It indicates the likelihood of false alarms in a classification system, and it is assessed using Equation (31).

$$FPR(\%) = \frac{z}{z+w} \quad (31)$$

- **Precision:** It is the measure of finding the reliability of the developed framework in correctly identifying normal and abnormal classes similar to ground truth samples and it is formulated as Equation (32),

$$Precision(\%) = \frac{x}{x+z} \quad (32)$$

- **Recall:** Recall measures the proportion of actual positive cases that are correctly identified by the model. It indicates the model's ability to detect positive instances and is particularly important in scenarios where missing positive cases is costly. It is calculated as Equation (33),

$$Recall(\%) = \frac{x}{x+y} \quad (33)$$

Where, *w, x, y,* and *z* indicate the true negative (TN), true positive (TP), false negative (FN), and false positive (FP) respectively.

4.2. Confusion Matrix of the developed Method

In this section, the Confusion matrix analysis for the developed framework is analyzed under normal and attacked classes. The confusion matrix is one of the most essential and effective analyses that assist in a better understanding of the proposed approach over misclassified outcomes. For the testing process, a total of 28 normal instances and 144 attack instances are considered.

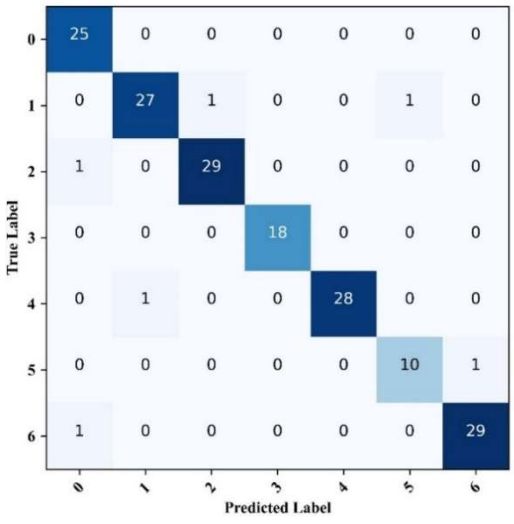
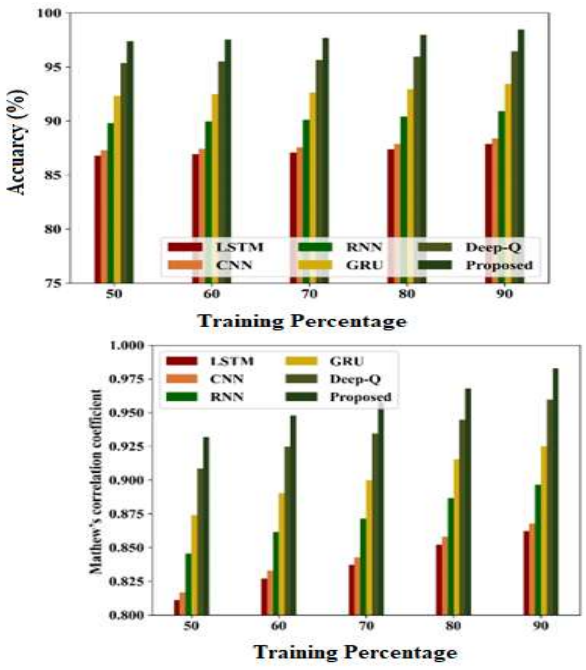


Figure 4: Confusion matrix analysis of the developed Framework

Figure 4 depicts the Confusion matrix analysis of the developed Framework. Here, the classes 0, 1, 2, 3, 4, 5, and 6 represent various attacks like DoS, exploit, fuzzers, generic, normal, reconnaissance, and shellcode, respectively. Here, false negatives (FN) represent instances where an intrusion has occurred but was mistakenly classified as a non-intrusive event or a different class, potentially leaving the system vulnerable to undetected threats. Examining this matrix, we see a few misclassifications across classes, but the FNs are particularly worth noting. For example, class 3 (with 18 correct predictions) has a relatively lower true positive count compared to other classes, and some instances in this class were misclassified as class 0 or 1. This pattern suggests that class 3 could share some overlapping feature characteristics with these other classes, leading the model to incorrectly predict the label. Similarly, for class 5, only 10 samples were correctly classified, while one sample each was misclassified as class 3 and class 6. This indicates that either the features for class 5 are less distinct or the model has difficulty distinguishing between these classes, which could affect detection accuracy for certain types of intrusions represented by class 5. The implications of these FNs in an IDS setting are significant. Every classification, especially an FN, could mean an actual intrusion is accurately detected, thus enhancing to trigger an alert or response.

4.3. Performance analysis of the developed method over conventional schemes

In this section, the performance obtained by the proposed method is analyzed via a graphical illustration. Various performance measures like Accuracy, MCC, Precision, Recall, FPR, and Computational time (CT) are scrutinized and compared with existing techniques. The detailed analysis of the obtained outcomes is encompassed as given below.



(a) Accuracy (b)MCC

Figure 5: Overall Accuracy and MCC Analysis by varying Training %

Figures 5(a) and 5(b) indicate the Accuracy and MCC Analysis for varying training % under different Techniques. The graphical manifestation shows that the developed RDQN technique proved a higher accuracy performance than other studies. For 50% of the training data, the existing LSTM, CNN, RNN, GRU, Deep-Q, and developed RDQN technique achieved an accuracy of 86.77%,

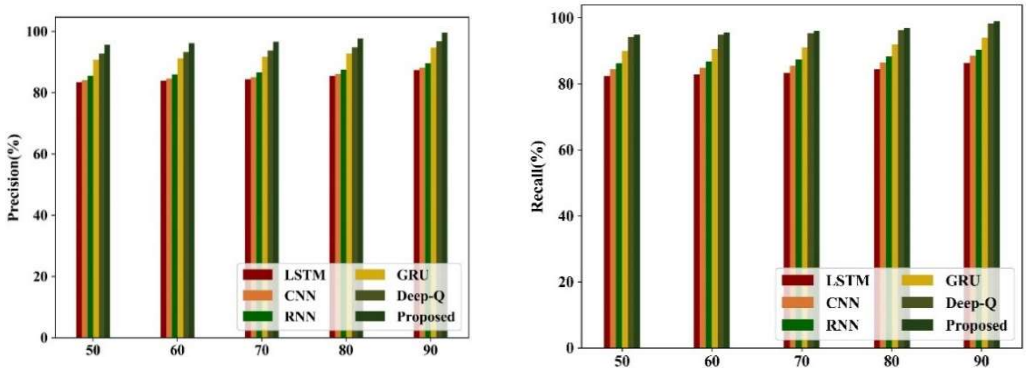
87.28%, 89.3%, 94.35%, 96.37%, and 97.38%, respectively. Table 1 encompasses the Obtained Accuracy computation over different studies under varying training %. For 50% of the training data, the existing LSTM, CNN, RNN, GRU, Deep-Q, and developed RDQN technique achieved an MCC of 0.81, 0.810, 0.83, 0.89, 0.92, and 0.93, respectively. Table 2 encompasses the obtained MCC computation over different studies under varying training %.

Table 1: Obtained Accuracy computation over different studies under varying training %

Accuracy (%)						
Training %	LSTM	CNN	RNN	GRU	Deep-Q	Proposed (RDQN)
60	86.92	87.43	89.45	94.50	96.52	97.53
70	87.07	87.58	89.6	94.65	96.67	97.68
80	87.37	87.88	89.90	94.95	96.97	97.98
90	87.87	88.38	90.40	95.45	97.47	98.48

Table 2: Obtained MCC computation over different studies under varying training %

MCC						
Training %	LSTM	CNN	RNN	GRU	Deep-Q	Proposed (RDQN)
60	0.82	0.83	0.85	0.91	0.93	0.94
70	0.83	0.84	0.86	0.92	0.94	0.95
80	0.85	0.85	0.88	0.93	0.95	0.96
90	0.86	0.86	0.89	0.94	0.97	0.98



(a)Precision *(b)Recall*

Figure 6: Overall Precision and Recall Analysis by varying Training %

Figures 6(a) and 6(b) indicate the Precision and Recall Analysis for varying training % under different Techniques. From the graphical manifestation, it is noted that the developed RDQN technique proved a higher performance than other studies. For 50% of training data, the existing

Journal of Theoretical and Applied Information Technology
15th November 2025. Vol.103. No.21

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

LSTM, CNN, RNN, GRU, Deep-Q, and developed RDQN technique achieved a precision of 83.39%, 84.07%, 85.5%, 90.6%, 92.7%, and 95.6% respectively. Table 3 encompasses the Obtained Precision computation over different studies under varying training %. For 50% of training data, the existing LSTM, CNN, RNN, GRU, Deep-Q, and developed RDQN technique achieved a recall of 82.24%, 84.64%, 85.32%, 90.16%, 92.74%, and 95.6% respectively. The Table 6(b) encompasses the Obtained recall computation over different studies under varying training %.

Table 3: Obtained Precision computation over different studies under varying training %

Precision (%)						
Training %	LSTM	CNN	RNN	GRU	Deep-Q	Proposed (RDQN)
60	83.89	84.57	86.01	91.19	93.24	96.12
70	84.39	85.07	86.51	91.69	93.74	96.62
80	85.39	86.07	87.51	92.69	94.74	97.62
90	87.39	88.07	89.51	94.69	96.74	98.62

Table 4: Obtained Recall computation over different studies under varying training %

Recall (%)						
Training %	LSTM	CNN	RNN	GRU	Deep-Q	Proposed (RDQN)
60	82.74	85.14	85.82	90.66	93.24	96.10
70	83.24	85.64	86.32	91.16	93.74	96.6
80	84.24	86.64	87.32	92.16	94.74	97.6
90	86.24	88.64	89.32	94.16	96.74	98.6

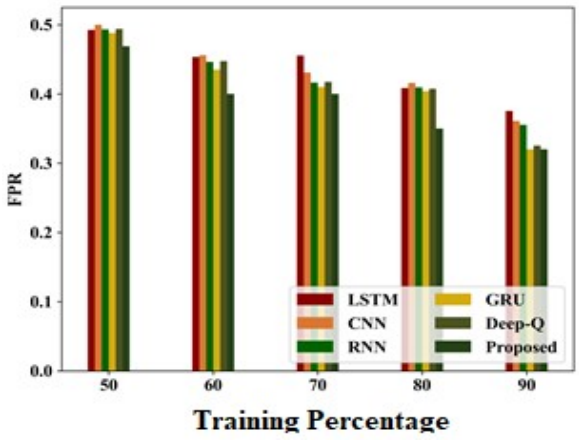


Figure 7: Overall FPR Analysis by varying Training %

Figures 7 indicate FPR Analysis for varying learning rates under different Techniques. The graphical manifestation showed that the developed RDQN lessor FPR over other studies. For 50% of training data, the existing LSTM, CNN, RNN, GRU, Deep-Q, and developed RDQN technique achieved an FPR of 0.492, 0.50, 0.498, 0.488, and 0.469 respectively. Tables 5 encompass the Obtained FPR computations over different studies

under varying training %. Table 5 encompasses the Obtained FPR computations over different studies under varying training %.

Table 5: Obtained FPR computation over different studies under varying training %

FPR						
Training %	LSTM	CNN	RNN	GRU	Deep-Q	Proposed (RDQN)
60	0.453	0.461	0.451	0.435	0.442	0.4
70	0.455	0.436	0.421	0.410	0.412	0.4
80	0.408	0.421	0.415	0.404	0.402	0.35
90	0.37	0.366	0.360	0.32	0.320	0.319

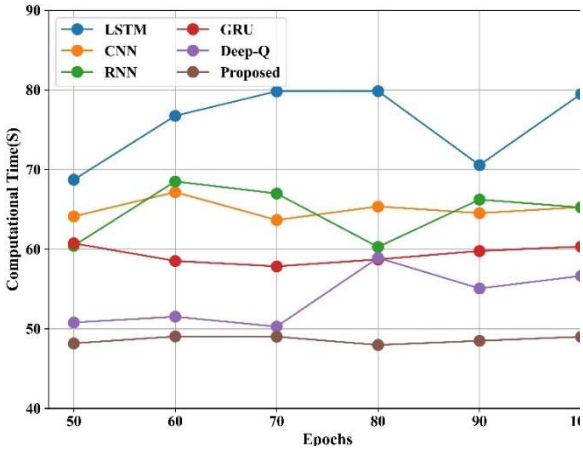


Figure 8: CT Analysis by varying Epochs

Figure 8 depicts the time complexity analysis for varying epochs using different techniques. The graphical interpretation shows that the developed method attains lesser complexities compared to traditional studies. The traditional techniques consume increased time while training with complex attack patterns that look similar to normal class. The developed method overcomes all the issues faced by the existing studies thereby providing accurate results with minimal training time. For 100 epochs, the existing LSTM, CNN, RNN, GRU, Deep-Q, and developed RDQN-XAI technique consumed a CT of about 79.47s, 65.37s, 65.23s, 60.32s, 56.64s, and 49s respectively. For the minimum of 50 epochs, the existing LSTM, CNN, RNN, GRU, Deep-Q, and developed RDQN technique consumed an accuracy of about 68.72s, 64.11s, 60.43s, 60.73s, 50.79s, and 48.17s respectively.

Table 6 illustrates Outcome analysis under various conventional studies with different datasets. The tabulation shows the efficacy of the developed framework and proves that it highly assists in upcoming real-time IoT applications and provides better security to users.

Table 6: Outcome analysis under various conventional studies with different datasets

Methods	Dataset used	XAI used	Accuracy (%)	Recall (%)	Precision (%)	Time consumption
Proposed (RDQN)	UNSW-NB15	kernelSHAP, PDP, PFI	98.48	98.6	98.62	49s
LSTM [22]	NSL-KDD	SHAP, ICE	86.6	98.2	81.11	121.9s
Hyperband optimization with DNN [24]	CSE CIC IDS 2018	LIME and SHAP	96.67	-	-	-
Ensemble ML models [25]	CICIDS-2017	LIME	96.25	80	89	-

LSTM+ FFNN+ RandNN [27]	CIC-IoT22	-	96.42	97.6	98.5	509.04s
TabNet-IDS [26]	CIC-IDS2017	-	97.03	97.02	97.10	-
FFNN [28]	CIC-IoT22	-	97.97	-	-	10 708.47
MLP [29]	ADFA-LD	LIME, SHAP	93.71	89.46	98.06	
Ensemble AI models [30]	MEMS, N-BaIoT	LIME, ALE, SHAP, PFI, CEM, LOCO, and Profweight	82	73	77	-

Table 7 tabulates the Inference Time Analysis on various Processes. The experimental outcome proves that the model consumes less time in processing each stage which supports for upcoming IoT communication.

Table 7: Inference Time Analysis on various Process

Process	Inference Time
Preprocessing	12.5s
RDQN	15.3s
XAI	24.32s

Table 8 indicates the Accuracy Analysis on various Normalization techniques. The experimental outcome prove that the introduced normalization technique effectively minimizes the error and enhances the detection performance.

Table 8 : Accuracy Analysis on various Normalization techniques

Technique used	Accuracy (%)
WCAdapt-MMN-RDQN (Proposed)	98.15
Z-score-RDQN	95.32
Robust scaling-RDQN	90.21
Min-max-RDQN	88.15

4.4 XAI Performance Analysis

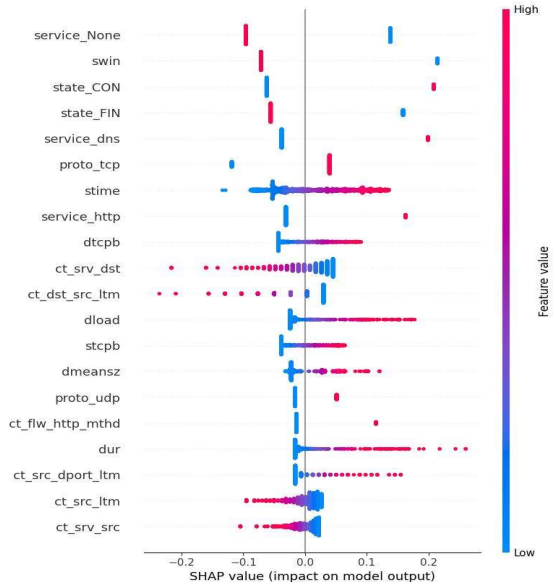
This section shows the explanations obtained for the proposed RDQN model using SHAP, PDP, and PFI-based model-agnostic plots. A brief analysis of the obtained XAI plots is discussed below:

4.4.1 KernelSHAP XAI analysis

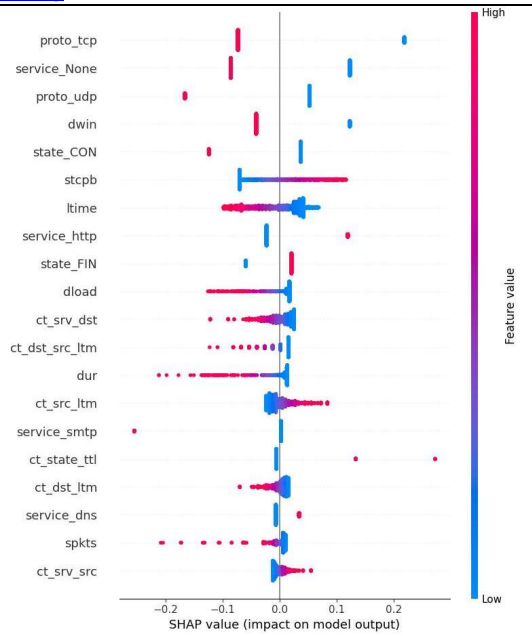
The SHAP conquers the list of most essential features by plotting its location on the graphical manifestation as illustrated in Figure 9. Here, the analysis is made by each dot to comprehend the plots that are represented in the data instances present in the database. The vertical dot‘s position signifies the feature of a particular attack. On the other hand, the horizontal dot‘s position signifies the impact value on the model‘s forecast. The colors present in each dot indicate the feature values of a specific attack, encompassing high, intermediate, or small (red, purple, or blue). For example, the maximum value for the feature ‘stime‘ in Figure 9(a) enhances the probability of the attack activity by 20% to 50%. The SHAP provides the feature importance and depicts how each feature staple.

Journal of Theoretical and Applied Information Technology
15th November 2025. Vol.103. No.21

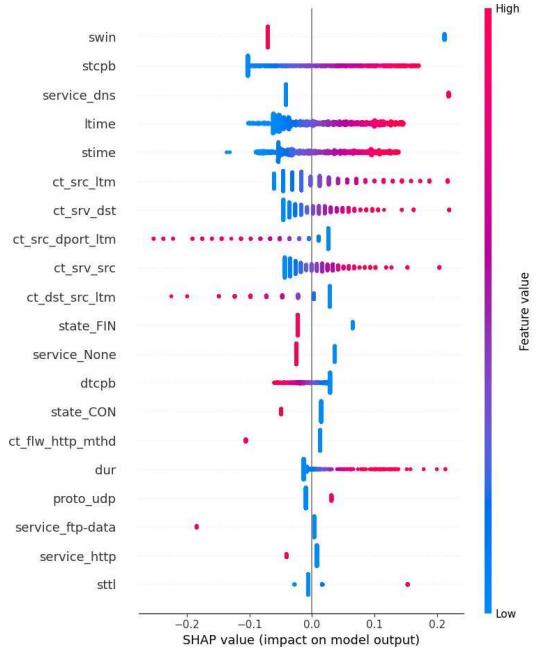
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195



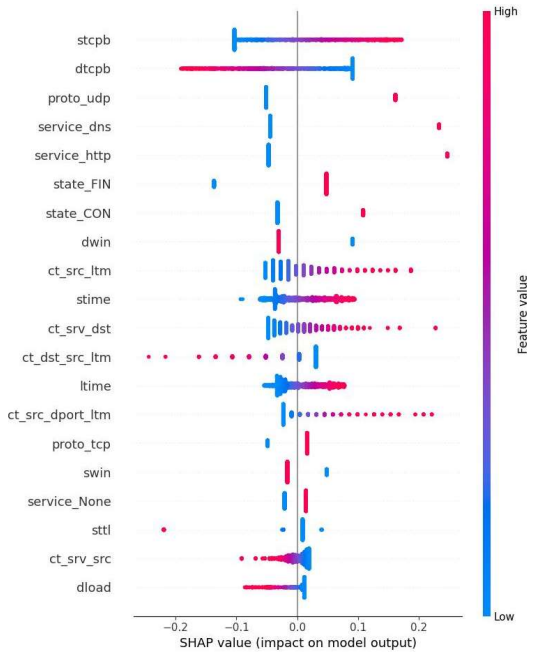
(a) KernelSHAP Plot for DoS Attack



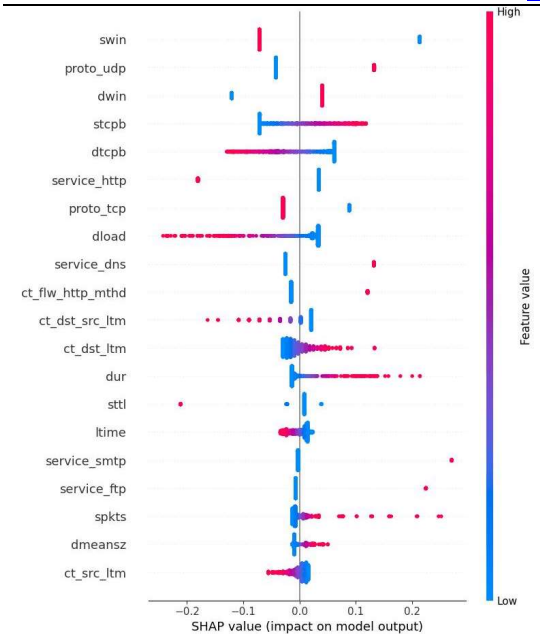
(b) KernelSHAP Plot for Exploits Attack



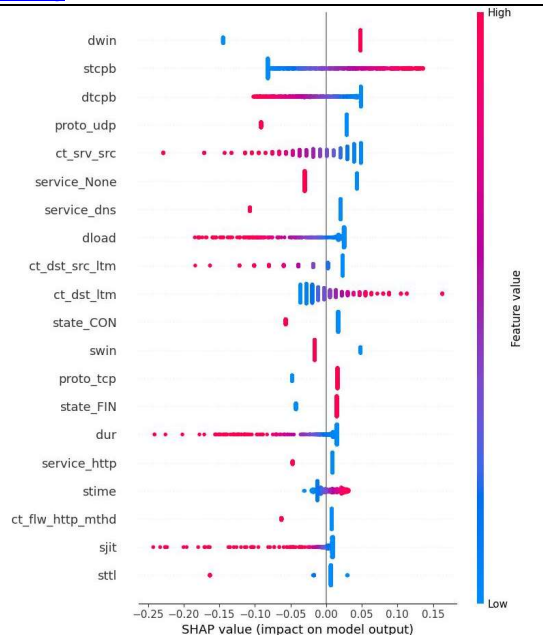
(c) KernelSHAP Plot for Fuzzers Attack



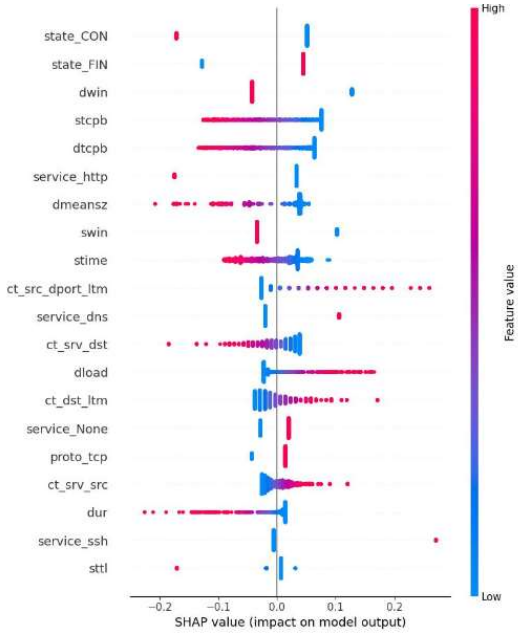
(d) KernelSHAP Plot for Generic Attack



(e) KernelSHAP Plot for Normal Class



(f) KernelSHAP Plot for reconnaissance Attack



(g) KernelSHAP Plot for Shell Code Attack

Figure 9: SHAP explanation of various Attack classes

4.4.2 PFI XAI Analysis

The PFI conquers the list of most essential features by placing its location on the graphical manifestation as illustrated in Figure 10. Here, the values of the features are sorted in descending order from top to bottom manner. For example, the maximum value for the feature 'dttl' is placed in the topmost position with its value ranging between 0.98 and 0.99. Finally, the probability of the attack activity is estimated to be more than 60% to 99% which is represented in green color. The white part of the features represents the features that are less important in predicting the specific activity in the IoT network.

Weight	Feature
0.9858 ± 0.0230	dttl
0.1195 ± 0.0138	sbytes
0.0743 ± 0.0503	sload
0.0110 ± 0.0032	proto_ospf
0.0106 ± 0.0060	dload
0.0091 ± 0.0009	dur
0.0079 ± 0.0023	dmeansz
0.0077 ± 0.0024	smeansz
0.0048 ± 0.0020	dintpkt
0.0015 ± 0.0005	ct_state_ttl
0.0015 ± 0.0020	sjit
0.0014 ± 0.0023	service_None
0.0005 ± 0.0000	sintpkt
0 ± 0.0000	ltime
0 ± 0.0000	ackdat
0 ± 0.0000	tcprtt
0 ± 0.0000	djit
0 ± 0.0000	stime
0 ± 0.0000	is_sm_ips_ports
0 ± 0.0000	dbytes

(a) PFI Plot for DoS Attack

Weight	Feature
0.9172 ± 0.2350	dttl
0.1892 ± 0.2519	sbytes
0.0125 ± 0.0060	sload
0.0095 ± 0.0000	dur
0.0091 ± 0.0149	dload
0.0006 ± 0.0000	ct_state_ttl
0 ± 0.0000	sloss
0 ± 0.0000	is_sm_ips_ports
0 ± 0.0000	ackdat
0 ± 0.0000	synack
0 ± 0.0000	smeansz
0 ± 0.0000	sintpkt
0 ± 0.0000	ltime
0 ± 0.0000	stime
0 ± 0.0000	djit
0 ± 0.0000	sjit
0 ± 0.0000	dmeansz
0 ± 0.0000	sttl
0 ± 0.0000	tcprtt
0 ± 0.0000	dintpkt

(b) PFI Plot for Exploit Attack

Weight	Feature
0.8870 ± 0.1074	dttl
0.1370 ± 0.1022	sbytes
0.0577 ± 0.0783	sload
0.0456 ± 0.0028	service_snmp
0.0223 ± 0.0173	dload
0.0183 ± 0.0025	synack
0.0153 ± 0.0127	dur
0.0130 ± 0.0025	proto_ospf
0.0068 ± 0.0013	smeansz
0.0062 ± 0.0077	sintpkt
0.0055 ± 0.0030	dmeansz
0.0042 ± 0.0001	ct_state_ttl
0.0030 ± 0.0071	stime
0.0020 ± 0.0036	dintpkt
0.0014 ± 0.0067	sjit
0.0008 ± 0.0008	service_None
0.0007 ± 0.0000	res_bdy_len
0 ± 0.0000	ltime
0 ± 0.0000	is_sm_ips_ports
0 ± 0.0000	dbytes

(c) PFI Plot for Fuzzers Attack

Weight	Feature
1.0193 ± 0.0791	dttl
0.1552 ± 0.0528	sbytes
0.1262 ± 0.0823	sload
0.0114 ± 0.0007	smeansz
0.0096 ± 0.0045	dmeansz
0.0089 ± 0.0002	dur
0.0077 ± 0.0035	dintpkt
0.0022 ± 0.0022	service_None
0.0020 ± 0.0066	dload
0.0011 ± 0.0000	stcpb
0.0010 ± 0.0000	res_bdy_len
0.0009 ± 0.0023	synack
0.0006 ± 0.0000	sintpkt
0.0003 ± 0.0021	sjit
0 ± 0.0000	stime
0 ± 0.0000	dloss
0 ± 0.0000	tcprtt
0 ± 0.0000	ltime
0 ± 0.0000	ackdat
0 ± 0.0000	djit

(d) PFI Plot for Generic Attack

Weight	Feature
0.9793 ± 0.0604	dttl
0.1763 ± 0.0205	sbytes
0.0865 ± 0.0200	sload
0.0253 ± 0.0013	proto_ospf
0.0204 ± 0.0023	smeansz
0.0144 ± 0.0025	dload
0.0103 ± 0.0024	synack
0.0085 ± 0.0033	dmeansz
0.0053 ± 0.0001	ct_state_ttl
0.0047 ± 0.0022	dintpkt
0.0043 ± 0.0001	dur
0.0018 ± 0.0018	sjit
0.0013 ± 0.0011	service_None
0.0011 ± 0.0000	stcpb
0.0006 ± 0.0000	sintpkt
0 ± 0.0000	dbytes
0 ± 0.0000	spkts
0 ± 0.0000	tcprtt
0 ± 0.0000	ltime
0 ± 0.0000	is_sm_ips_ports

(e) PFI Plot for Normal Class

Weight	Feature
0.9812 ± 0.0763	dttl
0.1721 ± 0.0311	sbytes
0.0736 ± 0.0503	sload
0.0261 ± 0.0007	proto_ospf
0.0260 ± 0.0050	smeansz
0.0142 ± 0.0043	synack
0.0112 ± 0.0042	dload
0.0084 ± 0.0021	dmeansz
0.0055 ± 0.0021	dintpkt
0.0050 ± 0.0014	dur
0.0037 ± 0.0001	ct_state_ttl
0.0030 ± 0.0046	sjit
0.0015 ± 0.0000	stcpb
0.0011 ± 0.0013	service_None
0.0008 ± 0.0000	sintpkt
0.0007 ± 0.0000	res_bdy_len
0 ± 0.0000	sttl
0 ± 0.0000	ltime
0 ± 0.0000	dbytes
0 ± 0.0000	swin

(f) PFI Plot for reconnaissance Attack

Weight	Feature
0.0014 ± 0.0004	smeansz
0.0008 ± 0.0008	sbytes
0.0005 ± 0.0000	sttl
0.0004 ± 0.0004	ct_srv_src
0.0003 ± 0.0005	sload
0.0003 ± 0.0005	stcpb
0.0001 ± 0.0004	ct_srv_dst
0.0001 ± 0.0004	sintpkt
0.0001 ± 0.0004	dtcpb
0 ± 0.0000	ltime
0 ± 0.0000	is_sm_ips_ports
0 ± 0.0000	dintpkt
0 ± 0.0000	tcprtt
0 ± 0.0000	synack
0 ± 0.0000	stime
0 ± 0.0000	ackdat
0 ± 0.0000	djit
0 ± 0.0000	sjit
0 ± 0.0000	state_URH
0 ± 0.0000	ct_state_ttl
... 42 more ...	

(g) PFI Plot for Shell Code Attack

Figure 10: PFI explanation of various Attack classes

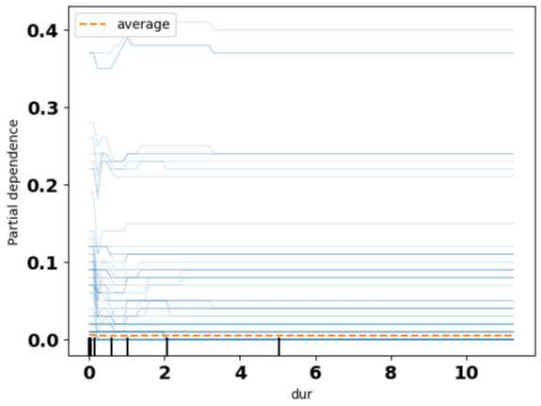
4.4.3 PDP XAI analysis

The PDP analyses the effects of each feature based on the overall decision as shown in Figure 11(a-g). The individual conditional expectation (ICE) is deliberated in blue color and the average of all blue lines is indicated in orange color which is PDP. These methods establish the changes in the model's forecasting when there are any changes in the features. Figure 11(a) depicts at low values of "dur" (close to 0), the ICE curves show significant variance. This indicates that the impact of small values of "dur" on the model's output varies widely across individual instances, with partial dependence values ranging up to about 0.4. As "dur" increases, especially beyond 2 units, most ICE curves converge towards 0, and the variation between instances decreases. This pattern suggests that higher "dur" values have a minimal or negligible impact on the model's predictions, and the dependence of the target on "dur" diminishes. The orange dashed line represents the average partial dependence across all instances. This line remains close to zero, indicating that "dur" has little overall impact on the model's prediction

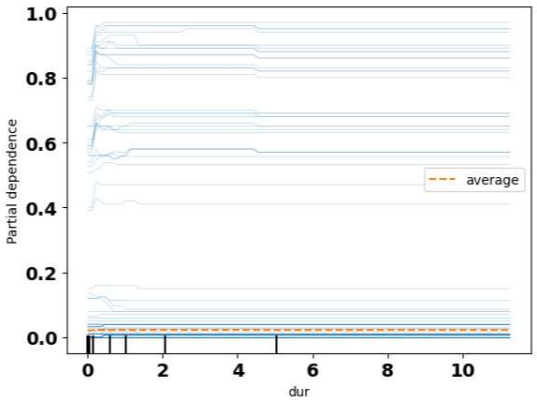
Journal of Theoretical and Applied Information Technology
15th November 2025. Vol.103. No.21

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

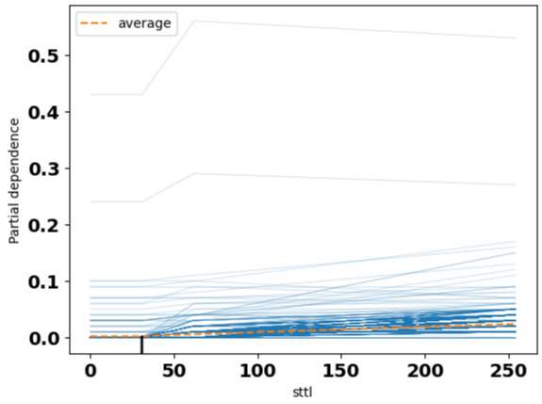
when averaged across all data points. However, the slight initial rise suggests that smaller values of "dur" might have a more noticeable but varied effect across instances.



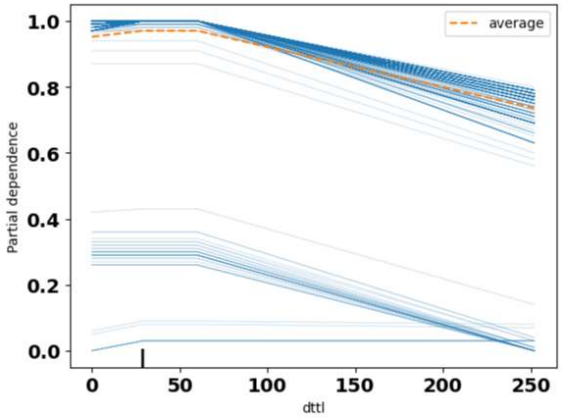
(a) DoS attack feature importance analysis using PDP plot



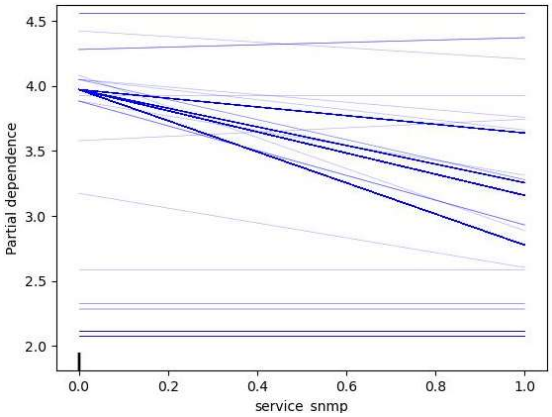
***(b)** Exploits attack feature importance analysis using PDP plot*



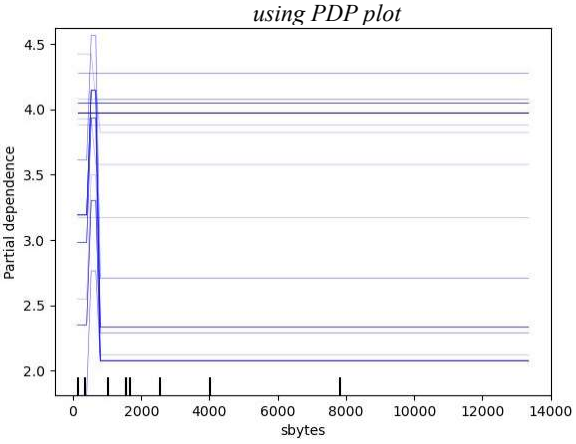
***(c)** Fuzzers attack feature importance analysis using PDP plot*



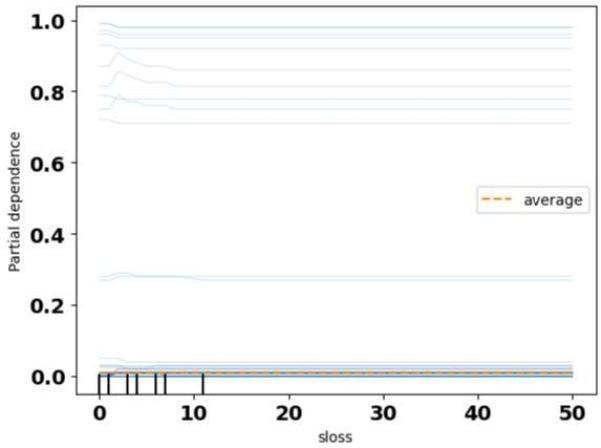
***(d)** Generic attack feature importance analysis using PDP plot*



***(e)** Reconnaissance attack feature importance analysis using PDP plot*



***(f)** Shellcode attack feature importance analysis using PDP plot*



(g) *Normal Class feature importance analysis using PDP plot*

Figure 11: PDP explanation of various Attack classes

4.5 Discussion

The experimental evaluation of the proposed RDQN-XAI framework demonstrates its effectiveness across multiple dimensions, particularly in the context of multiclass intrusion detection for IoT networks. This section discusses the outcomes by directly aligning them with the research objectives set forth in this study. The proposed RDQN-XAI model successfully classified multiple intrusion types, including Fuzzers, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Normal traffic, using the UNSW-NB15 dataset. As evidenced in Figure 5 and Table 1, the model achieved an accuracy of 98.48%, surpassing conventional models such as LSTM (86.6%), CNN (87.28%), GRU (94.35%), and Deep-Q (96.37%). The confusion matrix (Figure 4) further confirms the RDQN's effectiveness, with most attack classes accurately predicted, despite minor misclassifications for classes with overlapping characteristics. These results validate the model's capability in handling complex, sequential attack patterns, thereby fulfilling the primary goal of building a robust multiclass IDS for IoT. Interpretability was a central focus of this study. The integration of KernelSHAP, Permutation Feature Importance (PFI), and Partial Dependence Plot (PDP) enabled transparent visualization of how individual features contribute to each classification. For instance, SHAP plots (Figure 9) highlighted influential features like 'stime' and 'dttl', while PFI plots (Figure 10) ranked features by their predictive power. The PDP plots (Figure 11) further demonstrated how changes in feature values impact prediction probabilities. These insights increase user trust in the model's decisions, thereby achieving the objective of explainable AI-based decision-making**.** The novel Weighted Correlated Adaptive Min-Max Normalization (WCAdapt-MMN) outperformed traditional normalization techniques by emphasizing features highly correlated with attack patterns. As shown in Table 8, this technique achieved a detection accuracy of 98.15%, compared to Z-score (95.32%) and Min-max normalization (88.15%). This confirms the effectiveness of WCAdapt-MMN in enhancing model sensitivity to relevant intrusion features, thus addressing the goal of optimized data preprocessing**.** Low false positives are critical for practical IDS deployment in IoT, as excessive alerts can lead to resource exhaustion and reduced trust. The proposed model achieved a false positive rate of 0.319, the lowest among all compared models (Figure 7, Table 5). Moreover, it maintained low computational complexity, requiring only 49 seconds of training time at 100 epochs (Figure 8), which is significantly better than existing models such as FFNN (509s) and LSTM (79.47s). These results confirm that the framework is not only accurate but also resource-efficient and scalable for real-time IoT applications. Overall, the RDQN-XAI framework successfully addresses the limitations of conventional IDS by offering a powerful, transparent, and real-time solution for intrusion detection in IoT environments. The alignment of results with research objectives validates the framework's utility in both academic research and practical deployment. It not only enhances security through accurate attack classification but also fosters user trust through explainability, an

essential criterion in high-stakes cybersecurity applications.

4.6. Problems and Open Research Issues

While the proposed RDQN-XAI framework demonstrates strong capabilities in intrusion detection for IoT systems, several unresolved challenges and limitations emerged throughout this study. Addressing these issues is essential for advancing research and ensuring the practical deployment of such systems in real-world scenarios. The following problems and open research questions have been identified:

- **Limited Dataset Diversity**: The current study is based solely on the UNSW-NB15 dataset. The model's ability to generalize across diverse and heterogeneous IoT data sources remains untested.
- **Lack of Online Adaptation**: The RDQN-XAI model is trained in a static offline setting. Its inability to adapt to zero-day attacks or dynamically changing threat patterns is a notable limitation.
- **Interpretability Constraints**: Although global XAI methods were employed, the system lacks local explanation capabilities that provide justification for specific predictions in real time.
- **Scalability for Edge Deployment**: The RDQN-XAI model has not yet been validated on resource-constrained edge devices, which are common in IoT deployments.
- **Vulnerability to Adversarial Attacks**: The resilience of the model to adversarial inputs or spoofed network traffic remains unexplored.
- **Privacy-Preserving Learning**: The framework does not currently support federated or privacy-preserving learning methods necessary for decentralized IoT environments.
- **Multimodal Threat Detection**: The study is restricted to structured tabular network data. Integrating multimodal data (e.g., audio, image, sensor) remains an open challenge.
- **Human-in-the-Loop Feedback**: The system lacks interactive mechanisms for security analysts to validate or adjust predictions based on domain knowledge.

5 CONCLUSIONS

This research proposed a novel and explainable deep reinforcement learning framework (RDQN-XAI) for multiclass intrusion detection in IoT-WSN systems. The developed method successfully combined the Recurrent Deep Q-Network (RDQN) with global model-agnostic explainability techniques such as KernelSHAP, Permutation Feature Importance (PFI), and Partial Dependence Plots (PDP), addressing the dual challenge of accurate detection and transparent decision-making in black-box IDS models. The proposed Weighted Correlated Adaptive Min-Max Normalization (WCAdapt-MMN) technique improved input data quality and preserved feature relevance, leading to enhanced model performance. Extensive evaluations on the UNSW-NB15 dataset confirmed the robustness of the RDQN-XAI model, achieving superior accuracy (98.48%), precision (98.62%), recall (98.6%), and a notably low false positive rate (0.319). The computational efficiency of the model (49s training time) and its consistent performance across various training conditions validated its suitability for real-time deployment in resource-constrained IoT environments. By explicitly aligning the results with the research objectives, this study demonstrated a comprehensive and scalable solution to secure IoT networks while ensuring interpretability—thereby bridging the gap between high-performance intrusion detection and user trust in AI systems. Although the RDQN-XAI framework demonstrated significant improvements in intrusion detection accuracy, interpretability, and computational efficiency, several avenues remain open for future exploration. One of the foremost areas for advancement is enhancing dataset diversity. The current model was trained and validated solely on the UNSW-NB15 dataset, which limits its ability to generalize across other real-world IoT environments. Future research should involve cross-dataset validation or domain adaptation techniques to improve robustness. Additionally, integrating continual learning or online reinforcement learning methods could enable the system to dynamically adapt to zero-day attacks and evolving intrusion patterns without the need for complete retraining. Another important direction is the development of lightweight or compressed versions of the RDQN-XAI framework that are suitable for edge computing environments, where computational resources are limited. Furthermore, although global

explainability was achieved through model-agnostic tools like SHAP, PDP, and PFI, incorporating local explanation methods such as LIME or Integrated Gradients could help security analysts understand individual predictions in real time. Ensuring adversarial robustness is also an urgent need, as the system must be resilient against spoofed or manipulated inputs in practical settings. Privacy-preserving learning approaches, such as federated learning, should be explored to ensure secure and decentralized model training, especially when dealing with sensitive IoT data across devices. Moreover, expanding the detection capabilities to multimodal data inputs—such as image, audio, or sensor streams—can enhance the generality of the system in smart cities and industrial IoT scenarios. Lastly, incorporating human-in-the-loop explainability systems, where domain experts can validate, guide, or refine the system's predictions, could greatly increase the operational trust and usability of AI-powered intrusion detection solutions.

REFERENCES

[1]. Sharma, Bhawana, et al., "Explainable Artificial Intelligence for Intrusion Detection in IoT Networks: A Deep Learning Based Approach." *Expert Systems with Applications*, vol. 238, Mar. 2024, p. 121751. *DOI.org (Crossref)*, https://doi.org/10.1016/j.eswa.2023.121751

[2]. Moustafa, Nour, et al., "Explainable Intrusion Detection for Cyber Defences in the Internet of Things: Opportunities and Solutions." *IEEE Communications Surveys & Tutorials*, vol. 25, no. 3, 2023, pp. 1775–807. *DOI.org (Crossref)*, https://doi.org/10.1109/COMST.2023.3280465

[3]. Tcydenova, Erzhena, et al., "Detection of Adversarial Attacks in AI-Based Intrusion Detection Systems Using Explainable AI." *Human-Centric Computing and Information Sciences*, vol. 11, no. 0, Sept. 2021, pp. 1–1. *DOI.org (CSL JSON)*, https://doi.org/10.22967/HCIS.2021.11.035

[4]. Shaheen, and Hina. *WITHDRAWN: Machine Learning Based Intrustion Detection. System for IoT Applications Using Explainable AI*. 13 July 2023. *In Review*, https://doi.org/10.21203/rs.3.rs-3157399/v1

[5]. Arisdakessian, Sarhad, et al., "A Survey on IoT Intrusion Detection: Federated Learning, Game Theory, Social Psychology, and Explainable AI as Future Directions." *IEEE Internet of Things Journal*, vol. 10, no. 5, Mar. 2023, pp. 4059–92. *DOI.org (Crossref)*, https://doi.org/10.1109/JIOT.2022.3203249.

[6]. Mahbooba, Basim, et al.,"Explainable Artificial Intelligence (XAI) to Enhance Trust Management in Intrusion Detection Systems Using Decision Tree Model." *Complexity*, edited by Ahmed Mostafa Khalil, vol. 2021, no. 1, Jan. 2021, p. 6634811. *DOI.org (Crossref)*, https://doi.org/10.1155/2021/6634811.

[7]. Roy, Souradip, et al., "An Explainable Deep Neural Framework for Trustworthy Network Intrusion Detection." *2022 10th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud)*, IEEE, 2022, pp. 25–30. *DOI.org (Crossref)*, https://doi.org/10.1109/MobileCloud55333.2022.00011.

[8]. Pande, Sagar, and Aditya Khamparia. "EXPLAINABLE DEEP NEURAL NETWORK BASED ANALYSIS ON INTRUSION DETECTION SYSTEMS." *Computer Science*, vol. 24, no. 1, Mar. 2023. *DOI.org (Crossref)*, https://doi.org/10.7494/csci.2023.24.1.4551

[9]. Wang, Yiwen, et al., "Network Intrusion Detection Based on Explainable Artificial Intelligence." *Wireless Personal Communications*, vol. 131, no. 2, July 2023, pp. 1115–30. *DOI.org (Crossref)*, https://doi.org/10.1007/s11277-023-10472-7.

[10]. Sivamohan, S., and S. S. Sridhar. "An Optimized Model for Network Intrusion Detection Systems in Industry 4.0 Using XAI Based Bi-LSTM Framework." *Neural Computing and Applications*, vol. 35, no. 15, May 2023, pp. 11459–75. *DOI.org (Crossref)*, https://doi.org/10.1007/s00521-023-08319-0.

[11]. Alani, Mohammed M., et al., "ARP-PROBE: An ARP Spoofing Detector for Internet of Things Networks Using Explainable Deep Learning." *Internet of Things*, vol. 23, Oct.

2023, p. 100861. *DOI.org (Crossref)*, https://doi.org/10.1016/j.iot.2023.100861

[12]. Neupane, Subash, et al. "Explainable Intrusion Detection Systems (X-IDS): A Survey of Current Methods, Challenges, and Opportunities." *IEEE Access*, vol. 10, 2022, pp. 112392–415. *DOI.org (Crossref)*, https://doi.org/10.1109/ACCESS.2022.3216617

[13]. Das, Suprabha, et al. "Integration of Machine Learning with Cybersecurity: Applications and Challenges." *Artificial Intelligence in Cyber Security: Theories and Applications*, edited by Tushar Bhardwaj et al.,., vol. 240, Springer International Publishing, 2023, pp. 67–81. *DOI.org (Crossref)*, https://doi.org/10.1007/978-3-031-28581-3_7.

[14]. Mohamed, and Nachaat. "Current Trends in AI and ML for Cybersecurity: A State-of-the-Art Survey." *Cogent Engineering*, vol. 10, no. 2, Dec. 2023, p. 2272358. *DOI.org (Crossref)*, https://doi.org/10.1080/23311916.2023.2272358

[15]. Vaddadi, Srinivas A., et al. "Utilizing AI and Machine Learning in Cybersecurity for Sustainable Development through Enhanced Threat Detection and Mitigation." *International Journal of Sustainable Development Through AI, ML and IoT*, vol. 2, no. 2, Dec. 2023, pp. 1–8. *ijsdai.com*, https://ijsdai.com/index.php/IJSDAI/article/view/25

[16]. Salem, Aya H., et al. "Advancing Cybersecurity: A Comprehensive Review of AI-Driven Detection Techniques." *Journal of Big Data*, vol. 11, no. 1, Aug. 2024, p. 105. *DOI.org (Crossref)*, https://doi.org/10.1186/s40537-024-00957-y

[17]. Dang, and Quang-Vinh. "Improving the Performance of the Intrusion Detection Systems by the Machine Learning Explainability." *International Journal of Web Information Systems*, vol. 17, no. 5, Sept. 2021, pp. 537–55. *DOI.org (Crossref)*, https://doi.org/10.1108/IJWIS-03-2021-0022

[18]. Xu, Yizhou, and Parteek Kumar "Interpreting Machine Learning-Based Intrusion Detection in IoT Using Explainable AI: A Case Study with Explainable Boosting Machine." *Journal of Computing Sciences in Colleges*, vol. 39, no. 1, 2023, pp. 56–65.

[19]. Mallampati, Seshu Bhavani, and Hari Seetha. "Enhancing Intrusion Detection with Explainable AI: A Transparent Approach to Network Security." *Cybernetics and Information Technologies*, vol. 24, no. 1, Mar. 2024, pp. 98–117. *DOI.org (Crossref)*, https://doi.org/10.2478/cait-2024-0006.

[20]. Maricar, Saahira Banu Ahamed, et al. "An Improved Explainable Artificial Intelligence for Intrusion Detection System." *International Journal of Intelligent Systems and Applications in Engineering*, vol. 12, no. 14s, Feb. 2024, pp. 108–15. *ijisae.org*, https://ijisae.org/index.php/IJISAE/article/view/4642.

[21]. Masud, Mohammed T., et al. "An Explainable Intrusion Discovery Framework for Assessing Cyber Resilience in the Internet of Things Networks." *Proceedings of the Future Technologies Conference*, Springer, 2023, pp. 199–215.

[22]. Keshk, Marwa, et al. "An Explainable Deep Learning-Enabled Intrusion Detection Framework in IoT Networks." *Information Sciences*, vol. 639, Aug. 2023, p. 119000. *DOI.org (Crossref)*, https://doi.org/10.1016/j.ins.2023.119000

[23]. El Houda, Zakaria Abou, et al. "A Novel IoT-Based Explainable Deep Learning Framework for Intrusion Detection Systems." *IEEE Internet of Things Magazine*, vol. 5, no. 2, June 2022, pp. 20–23. *DOI.org (Crossref)*, https://doi.org/10.1109/IOTM.005.2200028

[24]. Sauka, Kudzai, et al. "Adversarial Robust and Explainable Network Intrusion Detection Systems Based on Deep Learning." *Applied Sciences*, vol. 12, no. 13, June 2022, p. 6451. *DOI.org (Crossref)*, https://doi.org/10.3390/app12136451.

[25]. C, Haripriya, and Prabhudev Jagadeesh M. P. "An Explainable and Optimized Network Intrusion Detection Model Using Deep Learning." *International Journal of*

Advanced Computer Science and Applications, vol. 15, no. 1, 2024. *DOI.org (Crossref)*, https://doi.org/10.14569/IJACSA.2024.0150145

[26]. Patil, Shruti, et al. "Explainable Artificial Intelligence for Intrusion Detection System." *Electronics*, vol. 11, no. 19, Sept. 2022, p. 3079. *DOI.org (Crossref)*, https://doi.org/10.3390/electronics11193079

[27]. Zegarra Rodríguez, Demóstenes, et al. "Attentive Transformer Deep Learning Algorithm for Intrusion Detection on IoT Systems Using Automatic Xplainable Feature Selection." *PLOS ONE*, edited by Kathiravan Srinivasan, vol. 18, no. 10, Oct. 2023, p. e0286652. *DOI.org (Crossref)*, https://doi.org/10.1371/journal.pone.0286652

[28]. Bakhsh, Shahid Allah, et al. "Enhancing IoT Network Security through Deep Learning-Powered Intrusion Detection System." *Internet of Things*, vol. 24, Dec. 2023, p. 100936. *DOI.org (Crossref)*, https://doi.org/10.1016/j.iot.2023.100936.

[29]. Gaspar, Diogo, et al. "Explainable AI for Intrusion Detection Systems: LIME and SHAP Applicability on Multi-Layer Perceptron." *IEEE Access*, vol. 12, 2024, pp. 30164–75. *DOI.org (Crossref)*, https://doi.org/10.1109/ACCESS.2024.3368377.

[30]. Namrita Gummadi, Anna, et al. "XAI-IoT: An Explainable AI Framework for Enhancing Anomaly Detection in IoT Systems." *IEEE Access*, vol. 12, 2024, pp. 71024–54. *DOI.org (Crossref)*, https://doi.org/10.1109/ACCESS.2024.3402446.

[31]. Kim, Hyun-Jin, et al. "Associative Knowledge Graph Using Fuzzy Clustering and Min-Max Normalization in Video Contents." *IEEE Access*, vol. 9, 2021, pp. 74802–16. *DOI.org (Crossref)*, https://doi.org/10.1109/ACCESS.2021.3080180.

[32]. Camci, Efe, et al. "QLP: Deep Q-Learning for Pruning Deep Neural Networks." *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 10, Oct. 2022, pp. 6488–501. *DOI.org (Crossref)*, https://doi.org/10.1109/TCSVT.2022.3167951

[33]. *UNSW-NB15*. https://www.kaggle.com/datasets/alextamboli/unsw-nb15. Accessed 4 Jan. 2025.