DrillBit

The Report is Generated by DrillBit Plagiarism Detection Software

Submission Information

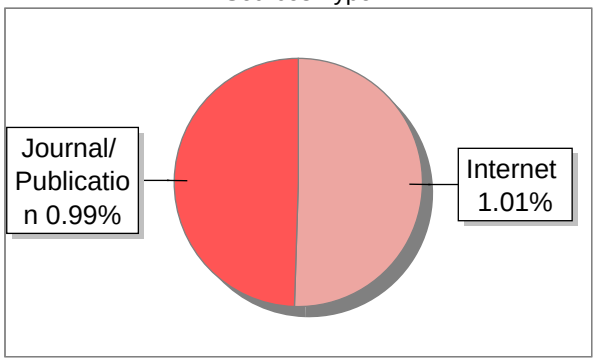
Author Name	Hamsa K P
Title	“STUDENT PERFORMANCE PREDICTOR BASED ON STUDY HABITS: ANALYZE FACTORS AFFECTING GRADES USING REGRESSION/CLASSIFICATION”
Paper/Submission ID	3588010
Submitted by	premu.kumarv@gmail.com
Submission Date	2025-05-07 13:02:35
Total Pages, Total Words	8, 2171
Document type	Research Paper

Result Information

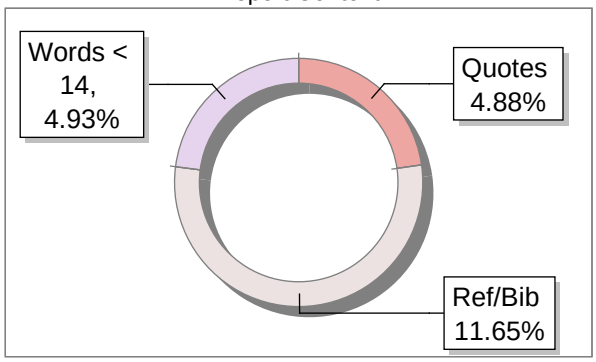
Similarity **2 %**

1 10 20 30 40 50 60 70 80 90

Sources Type



Report Content



Exclude Information

Quotes	Excluded
References/Bibliography	Excluded
Source: Excluded < 14 Words	Excluded
Excluded Source	**0 %**
Excluded Phrases	Not Excluded

Database Selection

Language	English
Student Papers	Yes
Journals & publishers	Yes
Internet or Web	Yes
Institution Repository	Yes

A Unique QR Code use to View/Download/Share Pdf File

DrillBit

DrillBit Similarity Report

2	3	A
SIMILARITY %	MATCHED SOURCES	GRADE

A-Satisfactory (0-10%)
B-Upgrade (11-40%)
C-Poor (41-60%)
D-Unacceptable (61-100%)

LOCATION	MATCHED DOMAIN	%	SOURCE TYPE
1	ijircce.com	<1	Publication
2	www.freepatentsonline.com	1	Internet Data
4	ijarcet.org	1	Publication

“STUDENT PERFORMANCE PREDICTOR BASED ON STUDY HABITS: ANALYZE FACTORS AFFECTING GRADES USING REGRESSION/CLASSIFICATION”

1st Hamsa K P

Department of Information Science,

The Oxford College of Engineering,

Bangalore, India

hamsakpise@gmail.com

2nd Likitha B S

Department of Information Science,

The Oxford College of Engineering,

Bangalore, India

likhithabsise20222@gmail.com

ABSTRACT-The security of networked systems is at serious risk due to the growing complexity and frequency of cyberattacks, necessitating more sophisticated and flexible defenses. Conventional intrusion detection systems (IDS) frequently use static rule-based methodologies, which restricts their capacity to identify new and developing threats. In order to dynamically modify detection strategies in response to real-time network behavior, this paper suggests an Adaptive IDS that makes use of Reinforcement Learning (RL), specifically Q-Learning and Deep Q-Network (DQN) models. Realistic traffic patterns and threat vectors are modeled using the CICIDS2017 dataset and a specially designed simulation environment. Through interactions with the environment, the RL agents are trained to maximize cumulative rewards, which allows them to gradually learn the best detection policies. Metrics like the true positive rate and average reward are used to assess performance.

KEYWORDS-*Reinforcement Learning (RL), Q-Learning, Deep Q-Network (DQN), CICIDS2017 Dataset, Adaptive Security, Cyber Threat Detection, Policy Latency, Average Reward, True Positive Rate (TPR), Network Simulation, and Real-Time Intrusion Detection System (IDS).*

I. INTRODUCTION

With their static nature, traditional intrusion detection systems (IDS) find it difficult to remain effective in the face of increasingly complex cyberattacks and dynamic network environments. These systems frequently fall short in identifying new threats and adjusting to shifting network traffic patterns. In order to tackle this issue, we suggest an Adaptive IDS that makes use of Reinforcement Learning (RL), particularly Q-Learning and Deep Q-Networks (DQN). This method enhances the IDS's capacity to recognize novel and intricate threats by allowing it to learn and modify detection policies in real time. The CICIDS2017 dataset and a custom simulation are used to train the system, enabling realistic and varied learning environments. The model's ability to detect intrusions accurately and adaptively is demonstrated by evaluation metrics like Average Reward, True Positive Rate (TPR), and Policy Latency.

Objectives of the Project

To efficiently detect intrusions, create an adaptive intrusion detection system (IDS) utilizing reinforcement learning techniques (Q-Learning and DQN).

Use a custom simulation setup to train and test the IDS in dynamic network environments.

Incorporate the CICIDS2017 dataset for training and validation to guarantee coverage of various attack types and real-world relevance.

Maximize reward signals based on accurate classification and prompt response by optimizing detection policies.

Use the True Positive Rate (TPR), Average Reward, and Policy Latency as key metrics to assess system performance.

Examine and contrast Q-Learning and DQN's efficacy in terms of detection

accuracy and learning efficiency.
Show how the IDS can be adjusted in real

time to changing network threats and traffic patterns.

II. LITERATURE SURVEY

Traditional Intrusion Detection Systems Conventional IDS techniques fall into two main categories: signature-based and anomaly-based methods. While signature based systems, such as Snort and Suricata, use predefined patterns to detect known attacks, they have trouble identifying new or obfuscated threats, and anomaly-based systems detect deviations from normal behaviour but frequently produce high false positives. These limitations highlight the need for adaptive systems that can respond to emerging threats.

Machine Learning in IDS A variety of machine learning models, including Support Vector Machines (SVM), Decision Trees, and Neural Networks, have been applied to IDS to increase detection accuracy. Although these models provide better generalization than static rule sets, they require frequent retraining as network behaviour changes, which makes them less effective in real-time or dynamic environments.

Methods of Reinforcement Learning One solution to adaptability issues is Reinforcement Learning (RL). Specifically, the model-free RL algorithm Q-Learning learns the best detection policies through trial-and-error interaction with the environment. Because it must maintain large Q-tables, it performs poorly in high-dimensional spaces, although it has demonstrated promise in adapting to shifting threatlandscapes.

DQN, or Deep Q-Networks By utilizing deep neural networks to approximate Q-values, Deep Q-Networks (DQN) get around the scalability problems with Q-Learning. This makes DQNs more suitable for real-time IDS applications with complex and large input spaces. Studies have demonstrated that DQN-based IDS models can improve detection accuracy and maintain adaptive behaviour in changing environment.

III. METHODOLOGY

In order to detect network intrusions intelligently and dynamically, this project develops an Adaptive Intrusion Detection System (IDS) using Reinforcement Learning (RL), specifically Q-Learning and Deep Q-Networks (DQN). According to the diagram, the process is broken down into two main stages: the offline phase, which includes training and validation, and the online phase, which includes testing and deployment. Phase Offline (Training + Validation) The learning takes place here. By repeatedly interacting with a simulated environment and actual traffic data, the IDS agent gains the ability to identify intrusions. Data for Training Sources of data: CICIDS2017 Dataset: Actual traffic marked with different kinds of attacks (brute force, denial-of-service, etc.).

Personalized Network Simulation: creates fake traffic on the fly to introduce the agent to changing trends. The agent is guaranteed to learn from both static real-world data and dynamic network conditions thanks to this hybrid approach.

Q-Learning

Stores the expected reward of performing an action in a specific state using a Q-table. By categorizing traffic (action), monitoring outcomes (reward), and updating the table, the agent engages with the environment. Deep Q-Network, or DQN, approximates the Q-values using a neural network. more effective and scalable when processing high-dimensional input (such as flow statistics and packet features). By using rewards—+1 for accurate intrusion detection, -1 for false positives, and 0 for uncertain results—the RL agent iteratively enhances its detection policy.

Model Results

A trained model is saved, which can be either Q-table or DQN weights. The learned policy—how to categorize incoming traffic according to its features—is encoded by this model.

Evaluation (during training)

After predetermined intervals, the model is validated using: Accuracy in identifying actual intrusions is known as the True Positive Rate (TPR).

- Average Reward: Indicates how successfully the agent is maximizing detection over time.
- Policy Latency: An important metric for real-time security, it gauges how fast the model decides what to detect. Training is restarted with modified hyperparameters or more varied data if the model exhibits poor performance on any of these metrics.
- Online Phase (Test/Deployment): To evaluate the model's real-time detection abilities, it is deployed in a real or simulated network following training.

Data Input

The system receives a stream of live network traffic, or real-time simulated traffic. Algorithm for Machine Learning (Inference Mode) Every flow or packet is classified using the trained model (Q-table or DQN). For every observation: The current state is determined by the system, determines the optimal course of action (e.g., block, alert, or allow) and carries it out with the least amount of delay. Forecasting and Warning The outputs of the system: A forecast, whether benign or malevolent, An automated action, such as blocking or flagging the source IP, is optional. Real-time decision-making guarantees prompt reaction to new threats.

Library	Purpose
NumPy	Efficient numerical operations and array manipulation.
Pandas	Data loading, cleaning, and preprocessing of the CICIDS2017 dataset.
Scikit-learn	Feature scaling, label encoding, model evaluation, and splitting datasets.
TensorFlow	Building and training Deep Q-Network (DQN) models.

Matplotlib	Plotting graphs for rewards, TPR, and performance trends.
Seaborn	Enhanced visualizations like heatmaps and correlation matrices.

IV. IMPLEMENTATION

CICIDS2017 Dataset Preprocessing

- Load CSV Files: Using a loop, you

load multiple CSV files from the CICIDS2017 dataset. You use pandas.read_csv() to read the file and make sure that it exists.

	Destination Port	Flow Duration	Total Fwd Packets	Total Backward Packets	al Lengt
1	49188	4	2	0	
2	49188	1	2	0	
3	49188	1	2	0	
4	49188	1	2	0	
5	49486	3	2	0	
6	49486	1	2	0	
7	49486	1	2	0	
8	49486	1	2	0	
9	88	609	7	4	
10	88	879	9	4	

Fig1: Data set.

- Features that are clean and Selected Whitespace is removed from column names after all CSVs have been combined. Only the pertinent features—"Flow Duration," "Total Fwd Packets," "Total Backward Packets," and "Packet Length Mean"—as well as the "Label"—are filtered.
- Label Encoding In binary classification, BENIGN is encoded as 0 and all other values as 1.
- Deal with Infinite/Missing Values You substitute NaN for infinite values and use column means to fill in all of the NaN values.
- Use SMOTE By oversampling the minority class, SMOTE can be used to balance the distribution of classes.

Implementation of Q-Learning Agent

- Initialization: A multi-dimensional NumPy array called a Q-table is based on the number of actions and discretized observation bins. Continuous state spaces can be handled with the aid of discretization.
- Action Selection Employs the greedy policy: Select a random action with probability ε.

If not, select the Q-table's most well-known action'.Update on Q-Value applies the update rule for Q-learning:

$Q\ (s,a) \leftarrow Q\ (s,a) + \alpha\ [r + \gamma \max\ Q\ (s\ ',\ a\ ') - Q\ (s,\ a)]$

$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max Q(s\ ', a\ ') - Q(s,a)]$.

- Loop of Training Repeat until the end of each episode: Select and carry out an action. See the next state and reward. Revise the Q-table. Monitor latency (time per episode) and average reward. After every episode, ε decays.

DQN Agent Implementation

Construct a two-layer neural network: Dense (16) → Output layer (linear activation, one value per action) → Input layer. stores experiences in replay memory, or deque. Choosing an Action applies ε greedy to neural network predictions. Replay the Experience To update the

network, randomly select batches from memory. Estimate the Q-values for the present and upcoming states. Fit the model and determine the desired Q-values. Loop of Training For every episode: Run through the surroundings After every step, replay the memory and store the experience. After every episode, decay ε.

Assessment and Visualization

- A heatmap that illustrates the relationship between average latency and average reward.
- Bar Chart: Shows the average reward and latency for both agent's side by side.

Fig : bar graph for models.

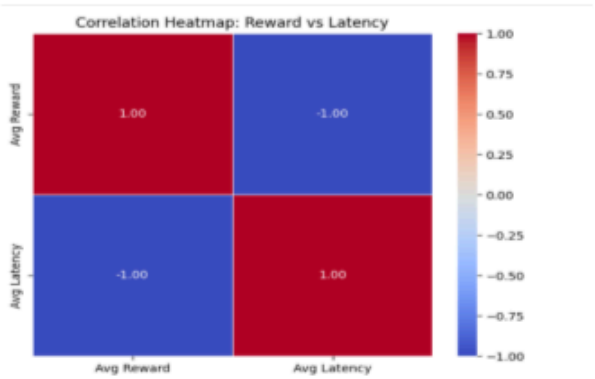


Fig : correlation map for reward and latency.

Agent Testing

- To assess the trained models, you invoke test_qlearning_agent () and test_dqn_agent () (although your snippet does not display these functions; normally, they would require the environment to be run in inference mode without training updates).

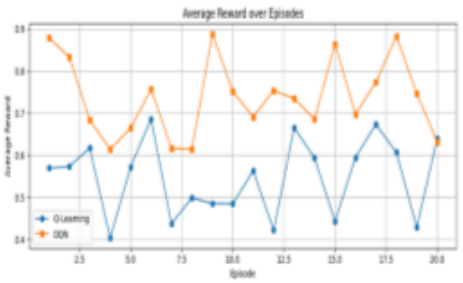


Fig:Line graph for comparing average reward between Q-learning and DQN.

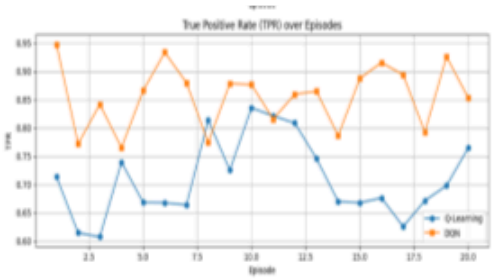


Fig: Line graph for comparing TPR between Q-learning and DQN..

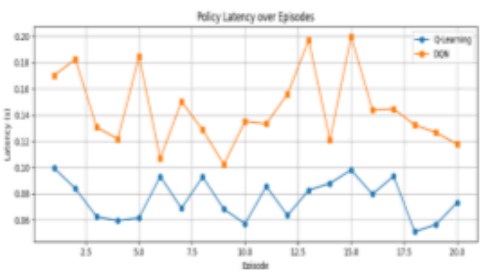


Fig: Line graph for comparing Latency between Q-learning and DQN.

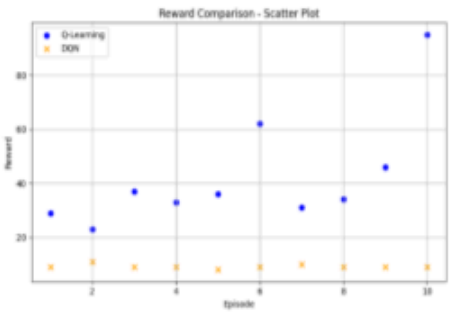


Fig:Scatter plot for reward comparision between Q-Learning and DQN.

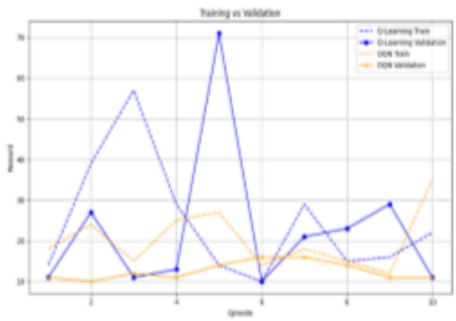


Fig: Training vs Validation for Q-Learning and DQN.

VI. CONCLUSION AND FUTURE WORK

CONCLUSION

Using Q-Learning and Deep Q Network (DQN) reinforcement learning algorithms, an adaptive intrusion detection system was created. The RL agent was trained in dynamic and realistic scenarios using both the CICIDS2017 dataset and a custom network simulation. Using important evaluation metrics, successful detection was attained: Average Reward: a measure of learning effectiveness. The True Positive Rate (TPR) indicates the accuracy of detection. Policy Latency: gauges responsiveness in real time. Q-Learning fared well in more isolated, smaller settings. DQN performed better when dealing with high-dimensional state spaces. The system outperformed conventional static IDS models in terms of adaptability.

Future Works

- Improved Information Gathering To increase the precision and generalizability of the results across various conditions and populations, future research should concentrate on gathering bigger and more varied datasets.
- Utilizing Cutting-Edge Methods Using deep learning or machine learning techniques can improve prediction and reveal intricate patterns that conventional methods might overlook.
- Application in the Real World The effectiveness of the model or system can be confirmed and any limitations that are not apparent in controlled settings can be found by testing it in real-time, realistic settings.
- Collaboration Across Disciplines Involving specialists from different domains could yield fresh viewpoints, resulting in more thorough solutions and inventive approaches to the research issue.

VII. REFERENCES

1. Hussain, Saqib, et al. "An adaptive intrusion detection system for WSN using reinforcement learning and deep classification." Arabian Journal for

Science and Engineering (2024): 1-15.

2. Hussain, S., He, J., Zhu, N., Mughal, F. R., Hussain, M. I., Algarni, A. D., ... & Ateya, A. A. (2024). “An adaptive intrusion detection system for WSN using reinforcement learning and deep classification” . Arabian Journal for Science and Engineering, 1-15.

3. Hussain, Saqib, Jingsha He, Nafei Zhu, Fahad Razaque Mughal, Muhammad Iftikhar Hussain, Abeer D. Algarni, Sadique Ahmad, Mira M. Zarie, and Abdelhamied A. Ateya. "An adaptive intrusion detection system for WSN using reinforcement learning and deep classification." Arabian Journal for Science and Engineering (2024): 1-15.

4. Hussain, S., He, J., Zhu, N., Mughal, F.R., Hussain, M.I., Algarni, A.D., Ahmad, S., Zarie, M.M. and Ateya, A.A., 2024. “An adaptive intrusion detection system for WSN using reinforcement learning and deep classification” . Arabian Journal for Science and Engineering, pp.1-15..

5. Hussain S, He J, Zhu N, Mughal FR, Hussain MI, Algarni AD, Ahmad S, Zarie MM, Ateya AA. “An adaptive intrusion detection system for WSN using reinforcement learning and deep classification” . Arabian Journal for Science and Engineering. 2024 Dec 2:1-5.

6. Alavizadeh, Hooman, Hootan Alavizadeh, and Julian Jang-Jaccard. "Deep Q-learning based reinforcement learning approach for network intrusion detection." Computers 11, no. 3 (2022): 41.

9. Alavizadeh, Hooman, Hootan Alavizadeh, and Julian Jang-Jaccard. "Deep Q-learning based reinforcement learning approach for network intrusion detection." Computers 11.3 (2022): 41.