Volume: 13 Issue: V Month of publication: May 2025
DOI: https://doi.org/10.22214/ijraset.2025.70151

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 13 Issue V May 2025- Available at www.ijraset.com

5G Network Resource Allocation Using Reinforcement Learning

Md. A. Khudhus[1], Dr. M. Humera Khanam[2], Dr. D. Vivekanandha Reddy[3], A. Sreenadh[4], K. Vishnu[5], V. Satwik[6]
[1, 2, 3, 4, 5, 6] *Department of Computer Science and Engineering, Sri Venkateswara University College of Engineering, Tirupati, A.P*

Abstract: The rapid evolution of 5G and LTE networks, coupled with the increasing diversity of user requirements, has necessitated intelligent and dynamic resource allocation techniques. This paper explores the application of reinforcement learning (RL) algorithms—specifically Deep Q-Network (DQN), Proximal Policy Optimization (PPO), and Mean Field Q-learning (MFQ)—to optimize network resource allocation based on critical Quality of Service (QoS) parameters such as packet loss rate and packet delay. Utilizing a real-world-inspired dataset that includes multiple dimensions like time of day, user equipment categories, and usage scenarios across various sectors (healthcare, public safety, smart cities, etc.), an OpenAI Gym-compatible environment is developed to simulate real-time allocation challenges. Each RL agent learns to allocate network resources efficiently under changing conditions, aiming to minimize packet loss and delay while maximizing system throughput. The paper further includes the deployment of these trained models using Django on a local server, with a user-friendly React-based frontend that accepts input features and returns the allocation results from all three models. Comparative analysis of cumulative rewards, convergence rates, and action selections helps determine the effectiveness of each algorithm. The system demonstrates a scalable, intelligent approach to managing the growing demands of next-generation wireless networks, emphasizing practical feasibility and deployment readiness.
Keywords: Reinforcement Learning, Proximal Policy Optimization, Deep Q Networks, Mean Field Q Learning, Resource Allocation.

I. INTRODUCTION

The explosive growth of wireless communication technologies, particularly 5G and LTE networks, has led to a significant increase in user demands and diverse application scenarios. As the number of connected devices rises and real-time services become more critical, efficient and intelligent resource allocation in these networks has emerged as a major challenge. Traditional static or rule-based approaches are often inadequate in adapting to rapidly changing network conditions and varying Quality of Service (QoS) requirements. In this context, Reinforcement Learning (RL) offers a promising solution by enabling systems to learn optimal policies through interaction with dynamic environments. This paper focuses on leveraging three state-of-the-art RL algorithms—Deep Q-Network (DQN), Proximal Policy Optimization (PPO), and Mean Field Q-learning (MFQ)—to manage network resources intelligently. The goal is to minimize key QoS issues such as packet delay and packet loss rate while maximizing system throughput. A realistic, multi-dimensional dataset encompassing various times of day, user equipment types, and domain-specific usage patterns (e.g., healthcare, public safety, and smart cities) forms the basis for a custom OpenAI Gym-compatible environment. This environment is designed to simulate real-time network scenarios and evaluate the performance of each RL agent under dynamic conditions. Furthermore, the trained models are deployed using Django on a local server, with a user-friendly frontend built in React. This interface allows users to input real-world parameters and receive allocation decisions from all three models. A comparative evaluation based on metrics such as cumulative rewards, convergence speed, and decision-making trends provides deep insights into the strengths and limitations of each approach. This paper not only highlights the effectiveness of RL in wireless network management but also showcases the practical deployment of AI-driven solutions in the context of next-generation communication systems.

II. RELATED WORK

The rapid evolution of 5G network slicing has catalyzed the development of various resource allocation strategies aimed at efficiently managing shared network infrastructure for diverse service requirements. Existing systems in this domain can broadly be categorized into two groups: traditional optimization-based approaches and machine learning-based approaches, particularly those leveraging reinforcement learning (RL).

Traditional resource allocation methods have predominantly relied on optimization techniques such as binary and integer programming to solve the slice admission control and cross-slice resource allocation problems. Although these approaches have provided a solid theoretical foundation, they are often limited by their computational complexity and inability to adapt to the dynamic nature of network traffic and varying quality-of-service (QoS) requirements. As 5G networks support heterogeneous services such as enhanced Mobile Broadband (eMBB), ultra-reliable low latency communications (URLLC), and massive machine-type communications (mMTC), these conventional methods struggle to efficiently balance the trade-offs between throughput, latency, and reliability. In recent years, a growing body of research has explored the application of reinforcement learning to address these challenges. Early studies applied classical RL algorithms like Q-learning to approximate optimal slice admission policies and resource allocation strategies. For instance, surveys such as Han et al. demonstrated that Q-learning could effectively manage network slices by mapping network states to resource allocation decisions. However, these methods are often constrained by their limited ability to handle high-dimensional state-action spaces and non-linear reward functions inherent in complex network environments. Subsequent advancements integrated deep neural networks with RL, giving rise to deep Q-networks (DQN), which leverage deep learning to approximate value functions and better capture the non-linear dynamics of the environment. Despite their improved performance, DQN-based systems sometimes suffer from instability during training and a slow convergence rate, particularly when deployed in rapidly changing network conditions.

TABLE I
REFERENCES OF RELATED WORK

Author(s)	Title	Year	Outcome	Source
Chien-Nguyen Nhu, Minho Park	Dynamic Network Slice Scaling Assisted by Attention-Based Prediction in 5G Core Network	2022	Proposed an attention-based encoder-decoder model for proactive resource forecasting, enhancing network slice scaling efficiency.	IEEE Access
Amr Abo-Eleneen, Alaa Awad Abdellatif, Aiman Erbad, Amr Mohamed	DRL-APNS: A Deep Reinforcement Learning-Powered Framework for Accurate Predictive Network Slicing Allocation	2024	Introduced a DRL-based framework for predictive network slicing, achieving a minimum cost reduction of 15% compared to baselines	Hamad Bin Khalifa University Research Portal
Qiang Liu, Nakjung Choi, Tao Han	Deep Reinforcement Learning for End-to-End Network Slicing: Challenges and Solutions	2022	Analyzed challenges in deploying DRL for network slicing and explored techniques like safety DRL and imitation learning for automation.	arXiv
Jaehoon Koo, Veena B. Mendiratta, Muntasir Raihan Rahman, Anwar Walid	Deep Reinforcement Learning for Network Slicing with Heterogeneous Resource Requirements and Time Varying Traffic Dynamics	2019	Demonstrated that DRL improves resource utilization and latency performance in dynamic network slicing scenarios.	arXiv

Author(s)	Title	Year	Outcome	Source
Tu Nguyen, et al.	Efficient Embedding VNFs in 5G Network Slicing: A Deep Reinforcement Learning Approach	2022	Presented a DRL scheme achieving over 80% successful routing of slices in resource-limited networks.	arXiv
Yi Shi, Yalin E. Sagduyu, Tugba Erpek	Reinforcement Learning for Dynamic Resource Optimization in 5G Radio Access Network Slicing	2020	Showed that RL significantly enhances network utility over traditional methods in dynamic resource allocation	arXiv

III. PROPOSED SYSTEM

1) *Motivation :* With the rapid growth of real-time data generated by both humans and machines, the demand for high-speed, low-latency communication is more critical than ever. 5G technology addresses this by supporting massive data traffic and enabling future technologies like IoT. According to Shannon‘s theorem, higher channel capacity leads to higher data rates, but different frequency bands affect coverage and loss differently. Modern applications demand diverse network capabilities. Enhanced Mobile Broadband (eMBB) supports high-bandwidth services like video streaming, VR/AR, and smart city surveillance. Massive Machine-Type Communications (mMTC) focus on energy-efficient, large-scale IoT devices with minimal latency and bandwidth needs. Ultra-Reliable Low-Latency Communications (URLLC) enable mission-critical applications such as self-driving cars, industrial automation, and remote surgery. These diverse needs drive the development of intelligent resource allocation systems, forming the core motivation for this paper.
2) *Problem Statement:* This paper, titled *"5g network resource allocation using reinforcement learning,"* focuses on optimizing resource allocation in 5G architecture using reinforcement learning (RL). It highlights the need for intelligent allocation methods beyond traditional techniques due to the complexity and dynamic nature of modern networks. Using a dataset that captures key network parameters like packet delay and packet loss rate, a simulated environment is created where RL agents—specifically Deep Q-Network (DQN), Proximal Policy Optimization (PPO), and Mean Field Q-learning (MFQ)—learn to make effective resource allocation decisions. Each network state reflects real-time conditions, and available actions correspond to different bandwidth levels.The agents receive rewards based on how well their actions minimize delay and packet loss, encouraging behaviors that enhance overall network performance. The paper ultimately compares the efficiency and learning outcomes of the three RL algorithms in managing 5G network resources.
3) *Objective:* The primary objectives of this paper are to understand the 5G network architecture and the need for network slicing, and to explore various existing machine learning-based solutions for resource allocation. It involves implementing and evaluating three reinforcement learning algorithms—Deep Q Learning, Proximal Policy Optimization, and Mean Field Q Learning—on a relevant dataset. Key performance metrics are visualized to enhance understanding, and a comparative analysis is conducted to highlight the strengths and weaknesses of each algorithm.
4) *Deep Q Networks:* Deep Q Networks is an extension of Q-learning and overcomes some of the shortcomings of normal Q-learning. It is a model-free algorithm as it does not learn any model of the environment in advance. It is kind of value-based as it computes Q-values for a given state and action.

Traditional Q-learning suffers from the curse of dimensionality, which refers to the exponential increase of state-action space as state variables increase. Deep Q Networks remedies this to an extent by making use of neural networks to find Q-value approximation, as they can handle complex functions easily. This algorithm maintains a main Q-network and a target Q-network.

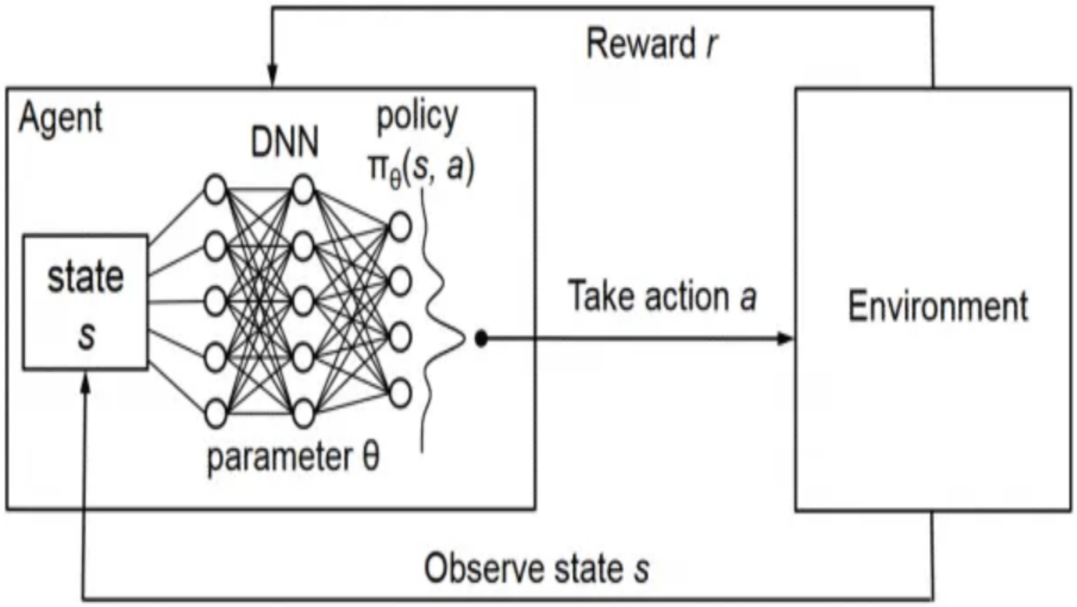


Fig. 1 Deep Q Networks (source. Towards AI)

5) *Proximal policy optimization:* Unlike DQN, PPO is a policy-based algorithm where it improves policy that provides probability of choosing the next action, given a state and action performed. Just like DQN, this algorithm trains two networks namely Policy network and value network. Policy networks decide the state based on the values provided by the value networks. Both these networks are updated simultaneously.

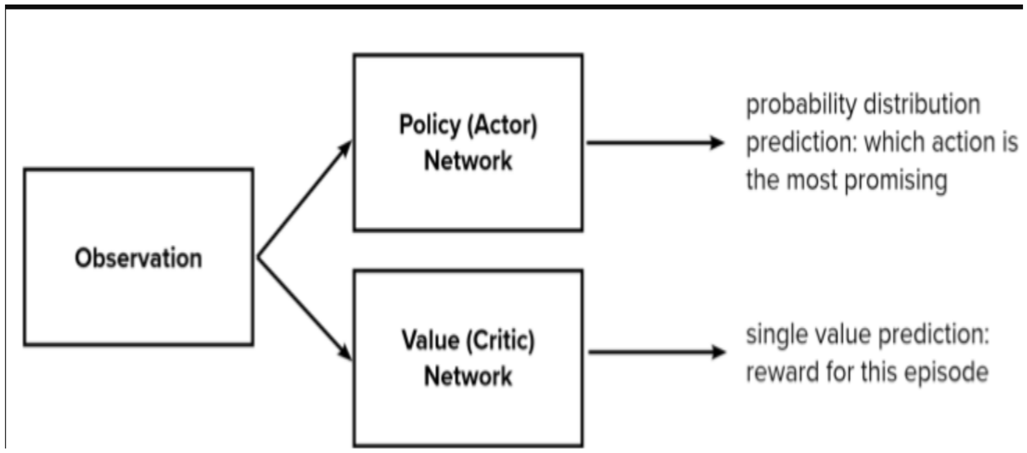

Observation
Policy (Actor) Network
Value (Critic) Network
probability distribution prediction: which action is the most promising
single value prediction: reward for this episode

Fig. 2 proximal policy optimization (source Applied AI course)

6) *Mean Field Q learning:* Mean Field Q learning is another extension of Q learning which improves upon in multi-agent settings. This algorithms accounts of average action of all other agents in the environment and collectively updates the individual agent's learning based on average actions taken by all agents. This algorithm assumes that all other agents are similar to the agent in question. The underlying principle is from Mean Field Theory of Mathematics. The mean actions performed is calculated by $1NN-1*\Sigma aajj\neq ii$ for jth action. The update rule is as follows Qi(s,ai,a¯−i) = (1−α)Qi(s,ai,a¯−i)+α(ri+γ*maxQi(s′,ai′,a¯−i′)). Due to the assumption of homogeneity the algorithm may suffer a little. As well as the fact that each agent depends on other agents mean action value. The MFQ does not make use of neural networks in its original form. That version is called Deep MFQ. Nonetheless it is a scalable solution in multi-agent settings and efficient because it does need to track all other agents. It reduces the problem of explosive state-action space using Mean Field Theory.

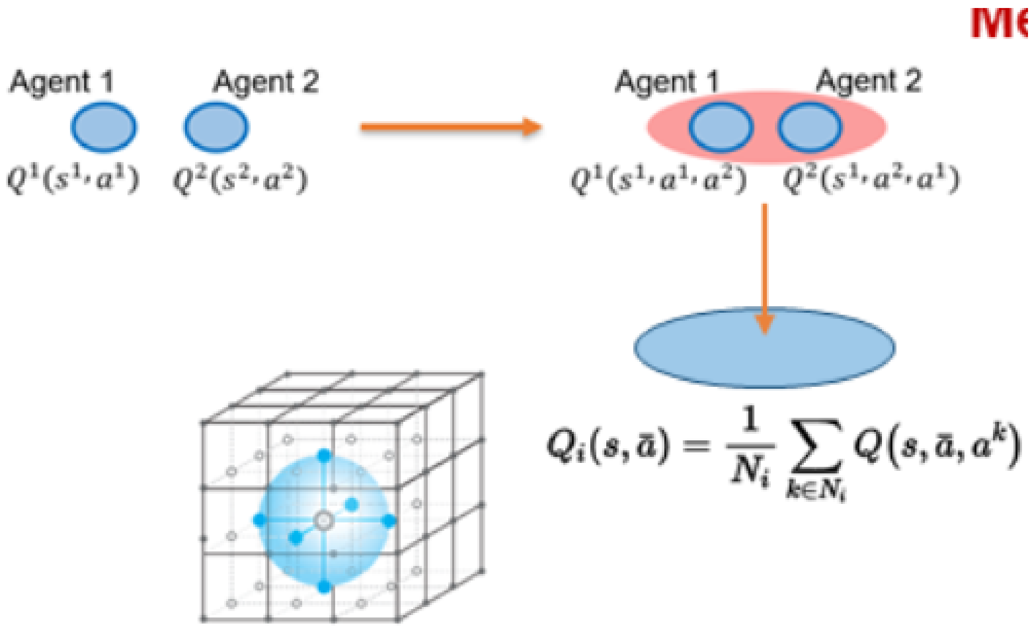


Fig. 3 Mean Field Q learning

The proposed system is a Django-based web application that predicts Quality of Service (QoS) metrics—latency, jitter, and throughput—using a Random Forest machine learning model. Key Features:

7) *User Input Form:* A React-based interface allows users to input features like time, device type, and sector-specific usage scenarios. Inputs are sent to the backend via REST API for processing.
8) *Machine Learning Backend*: Django REST Framework handles incoming requests and feeds inputs into trained DQN, PPO, and MFQ models hosted in a Gym-compatible environment for prediction.
9) *Prediction Output:* The backend returns resource allocation results from all three RL models, including predicted actions and QoS metrics like expected delay and packet loss.
10) *Deployment-Ready* **:** The system runs locally with Docker support for scalability, offering real-time allocation insights through an integrated frontend-backend architecture.

The flow diagram of the proposed system is shown in the below diagram.

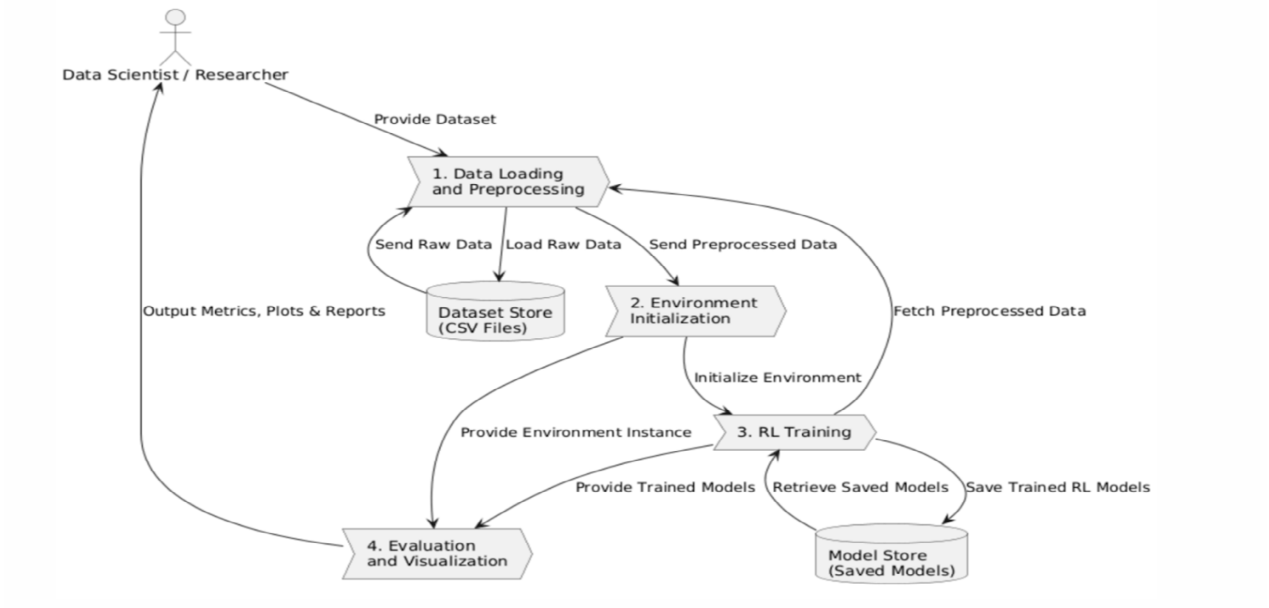

Data Scientist / Researcher
Provide Dataset
1. Data Loading and Preprocessing
Send Raw Data
Load Raw Data
Send Preprocessed Data
Dataset Store (CSV Files)
2. Environment Initialization
Output Metrics, Plots & Reports
Fetch Preprocessed Data
Initialize Environment
Provide Environment Instance
3. RL Training
Provide Trained Models
Retrieve Saved Models
Save Trained RL Models
4. Evaluation and Visualization
Model Store (Saved Models)

Fig. 4 Flow Diagram

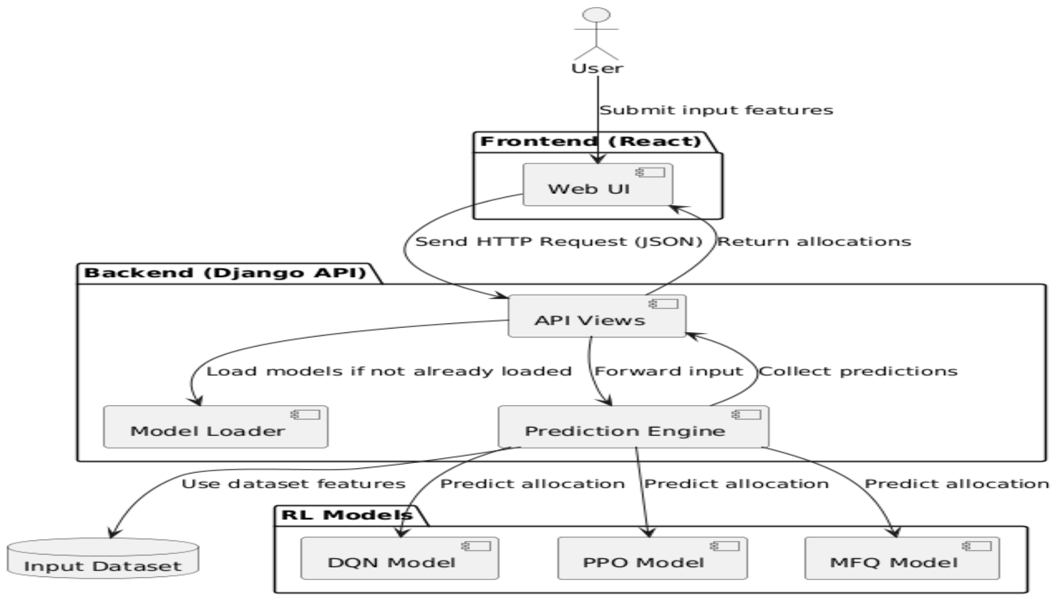


Fig. 5 Architecture

The Fig 4 The diagram illustrates a *Level 1 Data Flow Diagram (DFD)* for an RL Model Training and Evaluation System. The process begins with a Data Scientist or Researcher providing a dataset, which is handled by the Data Loading and Preprocessing module. Raw data is loaded from the Dataset Store (CSV Files) and preprocessed for training. The cleaned data is then sent to the Environment Initialization module, where a simulation environment is set up for reinforcement learning. This environment instance is passed to the RL Training module, which trains the agents and either saves new models or retrieves existing ones from the Model Store (Saved Models.

The Fig 5 The provided diagram illustrates the architecture of a Reinforcement Learning-based Network Allocation System. The system starts with a user submitting input features such as signal strength, bandwidth, and mobility via a React-based Web UI. These inputs are sent as JSON via HTTP requests to the Django-based backend API. The API Views component receives the request, invokes the Model Loader to ensure the necessary models (DQN, PPO, MFQ) are loaded, and then forwards the input to the Prediction Engine. This engine interacts with all three RL models using the provided dataset features to predict optimal network resource allocations.

IV. IMPLEMENTATION

The implementation involves using Python-based tools and frameworks to simulate and solve a network resource allocation problem through reinforcement learning. The main tools used are Pandas, NumPy, Matplotlib, and PyTorch. Pandas helps load, manipulate, and analyze structured data efficiently using tools like read_csv, loc, and time-series functions. NumPy is used for numerical operations with multi-dimensional arrays, providing functions for array creation, reshaping, statistical analysis, and linear algebra. Matplotlib is used for data visualization through line plots, scatter plots, histograms, and more. PyTorch handles the deep learning component, offering tensor operations, neural network building, and data loading utilities.

The paper features a custom reinforcement learning environment called NetworkAllocationEnv, which models how network resources (like bandwidth) are allocated. The environment is built around a dataset, where each row represents the current network state based on 16 features. The action space consists of 5 possible allocation levels, and the reward is calculated by comparing how close the allocated value is to the ideal requirement, based on performance metrics like packet loss rate and packet delay.

Key methods in the environment:

1) __init__: Sets up the dataset, action/observation spaces, and reward logic.
2) reset: Starts a new episode by returning the initial state.
3) _get_obs: Fetches the current state from the dataset.
4) compute_reward: Calculates a score based on how accurate the allocation is.
5) step: Advances the environment to the next time step, returns new state, reward, and status flags.

6) render: Prints the current step for basic visualization.

This setup allows for training reinforcement learning models that learn to make efficient resource allocation decisions based on network conditions.

Python libraries like Pandas, NumPy, Matplotlib, and PyTorch has been used to build and train a reinforcement learning model for network resource allocation.

- Pandas is used to read and manage the dataset.
- NumPy supports numerical operations and array handling.
- Matplotlib is used for plotting results.
- PyTorch handles tensors, models, and training.

A custom environment (NetworkAllocationEnv) is created where each state is a row of network features. The agent chooses one of five allocation levels (actions). The environment gives a reward based on how well the action matches the ideal allocation calculated using packet loss and delay.

Key functions:

- reset(): Starts a new episode.
- step(): Applies an action, returns new state and reward.
- compute_reward(): Calculates how good the action was.
- render(): Prints the current step.

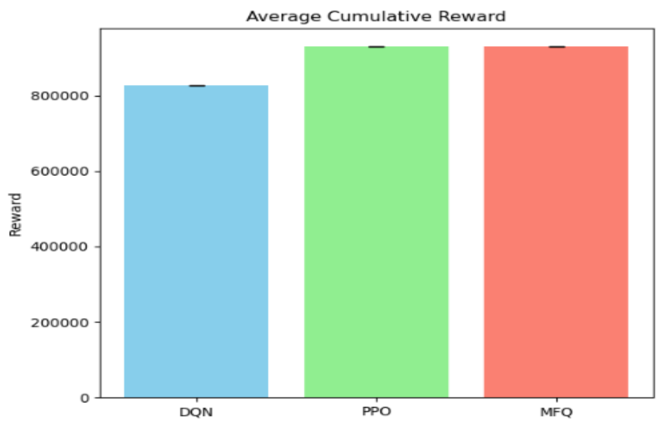
In short, the system trains a model to smartly allocate network resources based on changing network conditions.

This setup allows reinforcement learning agents to learn from real network data and gradually improve their allocation strategies through trial and error. By simulating real-world conditions, it provides a controlled environment for testing various algorithms like DQN, PPO, or MFQ. The modular design also makes it easy to tweak parameters, try different reward strategies, and integrate more features or constraints in future work. Overall, this implementation forms a strong foundation for intelligent, data-driven network management.

Ref.	Technique	Key Focus	Context	Remarks
[1]	Deep Q-Network (DQN)	Value-based RL method for resource allocation	5G slicing, QoS optimization	Struggles with convergence in complex environments
[2]	Proximal Policy Optimization (PPO)	Policy-based method with improved stability	Dynamic network scenarios	Better convergence and adaptability
[3]	Mean Field Q-Learning (MFQ)	Multi-agent RL for scalable decision making	Large-scale 5G networks	Handles agent interactions efficiently
[4]	Django + React Deployment	Web-based simulation and user input integration	Local server testing & visualization	Enables real-time input-output interaction

V. RESULTS

A. Comparision between ML models



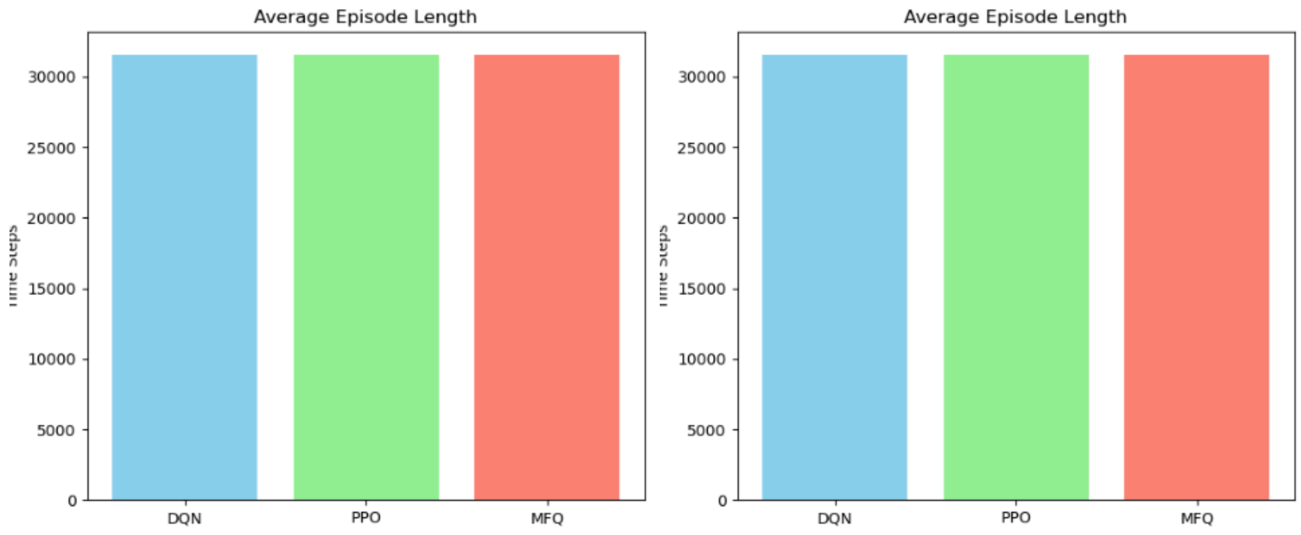


Fig. 5 Comparision between ML models

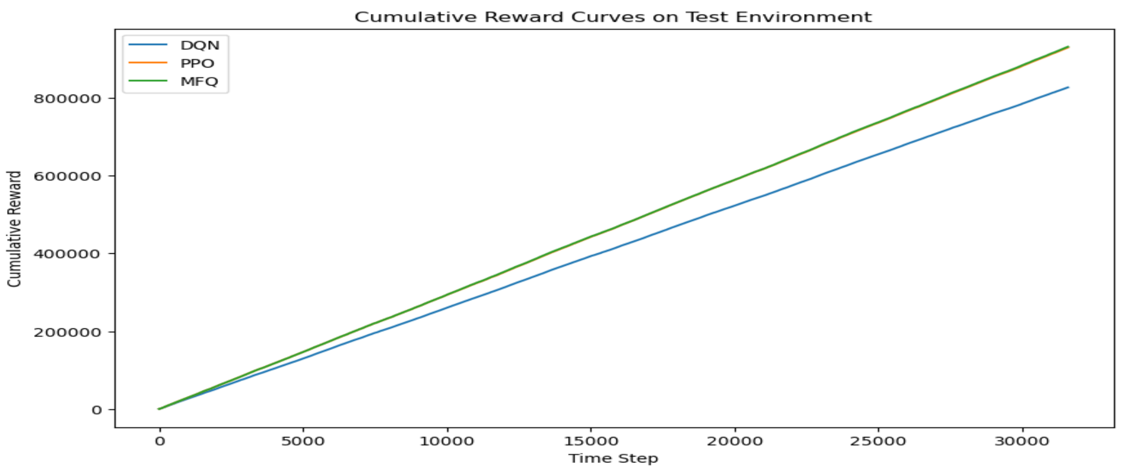
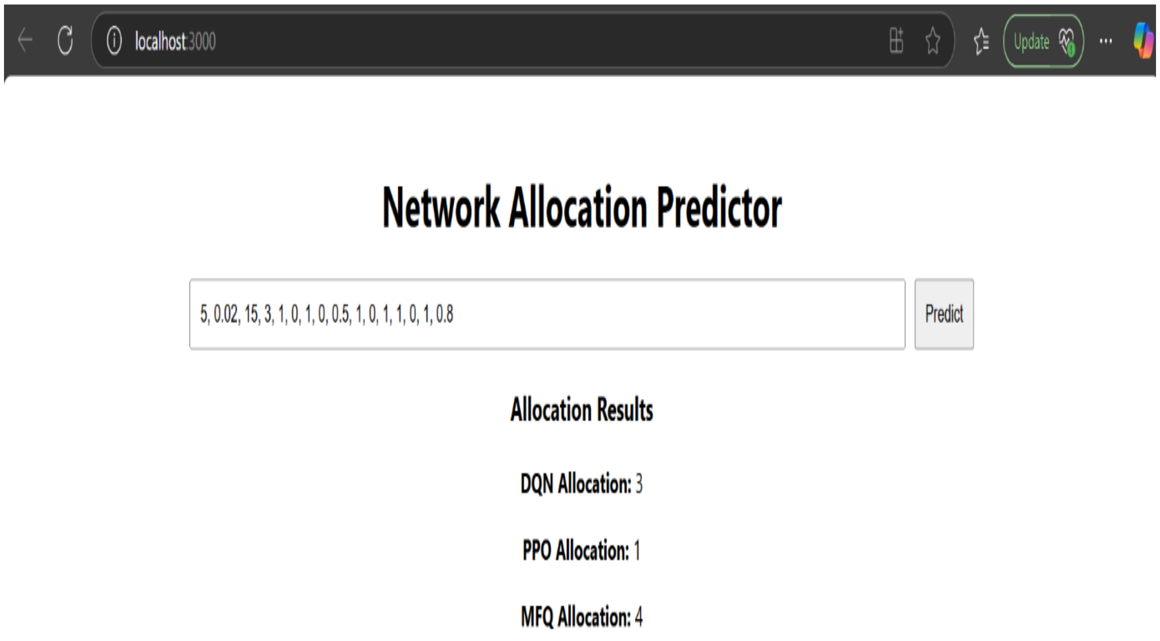


Fig 6. Cumulative rewards on Test Environment

In the network allocation environment, *Mean Field Q-learning (MFQ)* and *Proximal Policy Optimization (PPO)* demonstrated significantly better cumulative rewards compared to *Deep Q-Network (DQN).* MFQ achieved the highest performance due to its ability to model average agent behavior, resulting in more stable learning and effective decision-making under variable network conditions. PPO closely followed, leveraging its policy-gradient approach to adapt smoothly to changing states like delay and loss. DQN lagged behind, likely due to its discrete action-value estimation and limitations in handling continuous or noisy reward signals, leading to less optimal performance overall.

B. *Prediction Results*


Network Allocation Predictor

5, 0.02, 15, 3, 1, 0, 1, 0, 0.5, 1, 0, 1, 1, 0, 1, 0.8

Allocation Results

DQN Allocation: 3

PPO Allocation: 1

MFQ Allocation: 4

Fig. 6 Result

VI. CONCLUSIONS

In the network allocation environment, Mean Field Q-learning (MFQ) and Proximal Policy Optimization (PPO) demonstrated significantly better cumulative rewards compared to Deep Q-Network (DQN). MFQ achieved the highest performance due to its ability to model average agent behavior, resulting in more stable learning and effective decision-making under variable network conditions. PPO closely followed, leveraging its policy-gradient approach to adapt smoothly to changing states like delay and loss. DQN lagged behind, likely due to its discrete action-value estimation and limitations in handling continuous or noisy reward signals, leading to less optimal performance overall.

REFERENCES

[1] Faheem, M. T., Saidahmed, Ibrahim, M. Z., & Elshennawy, N. M. (n.d.). Efficient resource allocation of latency-aware slices for 5G networks.
[2] Han Bin, & Hans D. Schotten. (n.d.). Machine learning for network slicing resource management: A comprehensive survey.
[3] Moreira, R., Rodrigues, L. F., Moreira, & Carvalho, T. C. (n.d.). Resource allocation influence on application performance in sliced test.
[4] Chien-Nguyen Nhu, & Minho Park. Dynamic network slice scaling assisted by attention-based prediction in 5G core network.
[5] Amr Abo-Elenen, Alaa Awad Abdellatif, Aiman Erbad, & Amr Mohamed. A deep reinforcement learning-powered framework for accurate predictive network slicing allocation.
[6] Qiang Liu, Nakjung Choi, & Tao Han. Deep reinforcement learning for end-to-end network slicing: Challenges and solutions.
[7] Jaehoon Koo, Veena B. Mendiratta, Muntasir Raihan Rahman, & Anwar Walid. Deep reinforcement learning for network slicing with heterogeneous resource requirements and time-varying traffic dynamics.
[8] Tu Nguyen, et al. Efficient embedding VNFs in 5G network slicing: A deep reinforcement learning approach.
[9] [9] Yi Shi, Yalin E. Sagduyu, & Tugba Erpek. Reinforcement learning for dynamic resource optimization in 5G radio access network slicing.