2026 Volume: 02 Issue: 04 www.ijrpa.com ISSN 2456-9995 Research Article

International Journal Research Publication Analysis

Page: 01-05

GAME PLAYING AGENT USING DEEP-Q-NETWORK(DQN)

Dr Ramya B. N.*[1], Vishal S Hakki[2], Vaishnavi A. N.[3]

[1]Associate Professor, Department of Computer Science and Engineering, Jyothy Institute of Technology, Bengaluru, India.

[2],[3]Department of Computer Science and Engineering, Jyothy Institute of Technology, Bengaluru, India.

Article Received: 19 March 2026
Article Revised: 09 April 2026
Published on: 29 April 2026

***Corresponding Author: Dr Ramya B. N.**
Associate Professor, Department of Computer Science and Engineering, Jyothy Institute of Technology, Bengaluru, India.

DOI: https://doi-doi.org/101555/ijrpa.9872

ABSTRACT

Reinforcement Learning (RL) has emerged as a powerful paradigm for training intelligent agents capable of decision-making in dynamic environments. This paper presents a study on the application of **Deep Q- Networks (DQN)** for developing a game-playing agent. DQN combines Q-learning with deep neural networks to approximate value functions, enabling agents to learn optimal policies in high-dimensional state spaces. The agent is trained on classic environments such as *CartPole* and *Atari games*, with performance evaluated in terms of convergence speed, reward maximization, and policy stability. Results demonstrate that DQN effectively balances exploration and exploitation, achieving human-level performance in certain tasks. The study highlights the importance of replay memory, target networks, and reward shaping in stabilizing training and improving agent performance.

KEYWORDS: Reinforcement Learning, Deep Q-Network, Game Playing Agent, Policy Learning, Exploration- Exploitation, Neural Networks.

I. INTRODUCTION

Game-playing agents have long been a benchmark for artificial intelligence research, from early rule-based systems to modern deep reinforcement learning approaches. Traditional Q-learning struggles with large state spaces due to its reliance on tabular methods. Deep Q-Networks (DQN), introduced by DeepMind, address this limitation by leveraging deep neural networks to approximate Q-values, enabling agents to learn directly from raw sensory input such as pixels.

This paper explores the design and training of a DQN-based agent for game environments. The focus is on understanding how DQN stabilizes learning through mechanisms like **experience replay** and **target networks**, and how these contribute to robust policy learning. By analysing performance metrics and agent behaviour, we aim to provide insights into the strengths and limitations of DQN in game-playing contexts.

II. METHODOLOGY

Environment and Preprocessing

Experiments are conducted on:

- **CartPole-v1 (OpenAI Gym):** Balancing a pole on a moving cart.
- **Atari Breakout:** Pixel-based environment requiring strategic paddle control.

Preprocessing

- Frame normalization and grayscale conversion for Atari games.
- State representation as stacked frames to capture temporal dynamics.

Model Architecture

- Input: State representation (pixels or environment variables).
- Hidden Layers: Two fully connected layers with ReLU activation.
- Output Layer: Q-values for each possible action.

Stabilization Techniques

- **Experience Replay:** Random sampling of past transitions to break correlation.
- **Target Network:** Separate network updated periodically to stabilize Q-value targets.

Training Configuration

The models are trained using the following configuration:

- Optimizer: Adam
- Learning Rate: 0.00025
- Replay Memory: 50,000 transitions
- Batch Size: 32
- Discount Factor (γ): 0.99
- Exploration Strategy: ε-greedy with decay

III. SYSTEM ARCHITECTURE AND DATA FLOW

The proposed system follows a structured pipeline for training and evaluating a game-playing agent using the Deep-Q-Network (DQN) model. The architecture is designed to analyze how reinforcement learning impacts both performance and internal policy behavior.

Input Phase (Data Preparation)

- The agent receives the current state from the game environment (e.g., CartPole or Atari).
- States are preprocessed (frame normalization, stacking for temporal context).
- The state is represented as a vector or image input for the neural network.

Neural Network Architecture

The model consists of a fully connected feedforward structure with multiple layers:

- **Input Layer:** Encodes the environment state.
- **Hidden Layers:** Two fully connected layers with ReLU activation for feature extraction.
- **Output Layer:** Produces Q-values for each possible action.

This architecture enables hierarchical representation learning, where deeper layers capture complex game dynamics.

Training and Learning Process

The system is trained using supervised learning with labeled input data. The training process includes:

- **Forward Propagation:** The agent predicts Q-values for all possible actions.
- **Policy Selection:** Action chosen via ε-greedy strategy (balance between exploration and exploitation).
- **Environment Interaction:** Action executed, reward and next state observed.
- **Replay Memory Update:** Transition stored in replay buffer for future sampling.
- **Learning Phase:** Mini-batches sampled, Q-values updated using the Bellman equation.
- **Target Network Update:** Periodic synchronization with target network to stabilize learning.

Reinforcement Flow Integration

Reinforcement techniques are incorporated during the training phase as follows:

- **Experience Replay:** Breaks correlation between consecutive samples.
- **Target Network:** Provides stable Q-value targets.

- **Reward Shaping:** Guides agent toward optimal strategies

Each mechanism modifies the learning dynamics, allowing comparative analysis of their effects.

Evaluation and Output

After training, the agent is evaluated using test episodes:

- **Performance Metrics:** Average reward, episode length, convergence speed.
- **Visualization Tools:** Reward curves, Q-value distributions, and action frequency plots.
- **Behavioral Analysis:** Examines learned strategies, robustness, and stability.

The system outputs both quantitative metrics (reward, accuracy) and qualitative insights (policy visualization, Q-value heatmaps), enabling a comprehensive understanding of agent behavior under DQN.

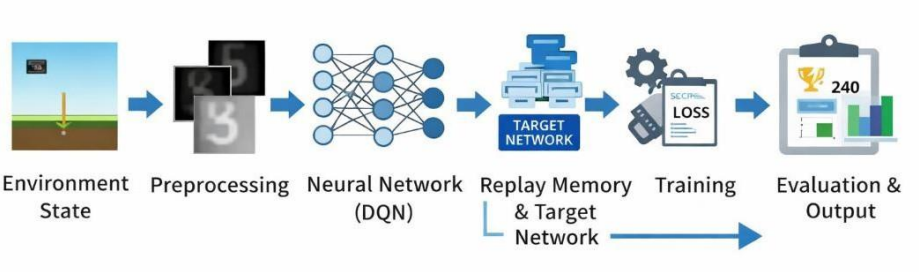

Fig 1 System Architecture Flow.

IV. RESULTS AND DISCUSSION

Performance Comparison

The classification accuracy of different models is summarized below:

Table1 Accuracy of Models

Model	CartPole (Avg Episodes Balanced)	Breakout(Avg score)
Random play	23	5.6
DQN	280+	95.4

Key Observations:

- Replay memory prevents catastrophic forgetting.
- Target networks reduce oscillations in Q-value updates.
- Exploration decay ensures transition from random actions to optimal policy.

Limitations:

- Training is computationally expensive.
- DQN struggles with sparse reward environments.
- Hyperparameter tuning is critical for stability.

These results indicate that moderate exploration and balanced learning parameters enhance policy generalization, while excessive exploration or unstable updates may slightly hinder convergence and overall performance.

V. CONCLUSION

This study demonstrates the effectiveness of Deep Q-Networks in training game-playing agents. DQN successfully learns optimal policies in environments with large state spaces, achieving competitive performance. The integration of replay memory and target networks is essential for stabilizing training. Future work may explore extensions such as **Double DQN**, **Dueling DQN**, and **Prioritized Experience Replay** to further enhance agent performance and efficiency.

VI. ACKNOWLEDGEMENT

The authors would like to express their sincere gratitude to Dr. Ramya B.N. for her valuable guidance and unwavering support throughout the course of this work. Her insightful feedback and expert advice played a crucial role in shaping the direction of this research. The encouragement she provided at every stage of this work was truly invaluable. The authors are deeply thankful for her time, dedication, and commitment to their academic growth.

VII. REFERENCES

1. Mnih, V. et al. (2015). *Human-level control through deep reinforcement learning*. Nature.
2. Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. MIT Press.
3. OpenAI Gym Documentation.
4. Hessel, M. et al. (2018). *Rainbow: Combining Improvements in Deep Reinforcement Learning*.