Department of Information Science and Engineering

A Minor Project Report On

"Reinforcement Learning on Asteroid Arcade Game"

IS6C06

Submitted By

B S Sanjana	**4NI21IS023**
Darshan G	**4NI21IS031**
Kaushik Bhat	**4NI21IS045**
Lokranjan P	**4NI21IS048**

Under the guidance of

Dr. Shashank D
Professor
Department of ISE, NIE Mysuru

The National Institute of Engineering
(Autonomous Institution under Visvesvaraya Technological University)
MYSORE – 570008
2023-2024

The National Institute of Engineering

Department of Information Science and Engineering

Manandavadi Road, MYSURU-570008

<u>Certificate</u>

Certifies that the minor project titled **"RL(Reinforcement Learning)on Asteroid Arcade Game"** is presented by B S Sanjana bearing USN **4NI21IS023, Darshan G** bearing USN **4NI21IS031, Kaushik Bhat**bearing USN **4NI21IS045** and **Lokranjan P** bearing USN **4NI21IS048** in partial fulfillment for the requirements of the sixth semester BE in Information Science & Engineering prescribed by The National Institute of Engineering, Autonomous Institution under Visvesvaraya Technological University, Belagavi, it is certified that all correction/suggestions indicated for Internal Assessment have been incorporated. The project report has been approved as it satisfies the academic requirements in respect of the Minor Project prescribed for the sixth semester.

Signature of Guide	Signature of HOD	Signature of the Principal
(Dr. Shashank D)	(Dr. Girish)	(Dr. Rohini Nagapadma)

Name of the Examiners	**Signature with Date**
1.	
2.	

ABSTRACT

This project explores the application of Reinforcement Learning (RL) to the classic Asteroid arcade game, aiming to develop an intelligent agent capable of autonomously navigating and excelling in the game. Utilizing Pygame for game development, we modified the game environment to interact seamlessly with an RL agent. A Deep Q-Network (DQN) was designed to approximate action-value functions, enabling the agent to make informed decisions based on the game state. The agent manages state transitions, selects optimal actions, and uses experience replay and a target network for stable learning.

Training was conducted on Google Cloud Platform, where the agent underwent numerous game episodes to learn and refine its strategies using an ε-greedy policy for exploration and exploitation balance. Performance evaluation demonstrated the agent's proficiency by comparing key metrics against baseline methods, underscoring the effectiveness of RL in enhancing game AI. This project highlights the versatility and power of RL techniques in real-time decision-making, contributing valuable insights into their application in gaming and beyond.

ACKNOWLEDGEMENTS

The success and the outcome of this project required a lot of guidance and assistance from many people and we are incredibly fortunate to have got this all along with the completion of project work.

We express our profound thanks to Dr. Rohini Nagapadma, Principal, NIE Mysuru for her much-needed moral support and encouragement.

We are grateful to Dr. Girish, Professor & HOD, Information Science and Engineering, NIE for his support and encouragement in facilitating the progress of this work.

We sincerely extend to our Project Guide Dr. Shashank D, Professor in the Department of ISE, for her guidance, technical expertise, encouragement and timely help in the making of this project a reality.

We would like to extend our sincere regards to all the teaching and non-teaching staff of the Department of ISE, NIE Mysuru for their assistance and timely support.

We would also like to give credit to the authors of the various resources which were made available on the Internet for our reference.

B S Sanjana	4NI21IS023
Darshan G	4NI21IS031
Kaushik Bhat	4NI21IS045
Lokranjan P	4NI21IS048

TABLE OF CONTENTS

Introduction

Chapter 1

In recent years, the field of Reinforcement Learning (RL) has gained significant traction due to its successful application in various complex decision-making problems. RL, a subfield of machine learning, enables agents to learn optimal behaviors through interactions with an environment. This minor project, titled "RL on Asteroid Arcade Game," aims to apply RL techniques to develop an intelligent agent capable of mastering the classic arcade game Asteroids.

Asteroids, a popular arcade game from the late 1970s, presents a dynamic environment where a spacecraft must navigate through space while avoiding and destroying asteroids. The game's objective is to survive as long as possible, accumulating points by eliminating asteroids. This environment provides an ideal testbed for RL algorithms due to its continuous, high-dimensional state space and the need for real-time decision-making.

The primary goal of this project is to design and implement a Deep Q-Network (DQN) based RL agent that can effectively play the Asteroids game. The DQN algorithm, which combines Q-learning with deep neural networks, is well-suited for handling the high-dimensional state space of the game. By training the agent to learn from its experiences within the game environment, we aim to demonstrate the potential of RL in mastering complex, dynamic tasks.

This project encompasses several key components: the game loop and environment modifications, the agent code, and the DQN implementation. The game loop handles the game graphics, environment setup, and event handling. The agent code manages the communication between the game and the DQN, including training episodes and iterations. Finally, the DQN file contains the neural network architecture and training procedures.

The project's significance lies in showcasing the effectiveness of RL techniques in gaming environments, highlighting the potential for broader applications in other domains requiring adaptive decision-making. By the project's conclusion, we aim to demonstrate improved gameplay performance, illustrating the RL agent's ability to learn and adapt in real-time.

Objectives

- **Develop an RL Agent**: Implement a robust RL agent using Deep Q-Networks (DQN) that can learn and execute optimal strategies for playing the Asteroids game, navigating the environment, and maximizing scores. Offers a structured assessment section using standardized questionnaires.
- **Integrate RL with Game Environment**: Adapt the Asteroids game environment to support the RL agent's training and deployment, including modifications to the game loop, graphics, and necessary interfaces for Interaction.
- **Evaluate and Optimize Performance**: Continuously assess the RL agent's performance through rigorous testing and optimization of learning algorithms and parameters to enhance decision-making capabilities and efficiency.
- **Demonstrate Feasibility of RL in Games**: Showcase the practical application of RL techniques in classic arcade games, demonstrating the potential of advanced AI to achieve proficiency in diverse gaming environments.

1.1 Purpose

The purpose of this project is to leverage Reinforcement Learning (RL) techniques to enhance the gameplay experience in a classic arcade game, Asteroids. This project aims to develop an intelligent agent that can learn and adapt its strategies to effectively navigate and survive in the game's challenging environment. By integrating RL algorithms, the project seeks to create a more dynamic and challenging game that can adjust its difficulty based on the player's skill level, providing a personalized and engaging experience. Furthermore, this project will serve as an educational tool to demonstrate the practical applications of RL in game development, showcasing how AI can be utilized to improve gaming experiences.

1.2 Existing System

Various platforms and applications currently provide mental health support and assessment. These systems generally include features like chatbots, mental health assessments, and informational blogs. However, many existing systems exhibit several limitations:

1. **Classic RL Algorithms:** Describe traditional reinforcement learning algorithms like Q-learning or Deep Q-Networks (DQN) that have been applied to arcade games such as Atari games. Highlight how these algorithms learn optimal strategies through trial and error based on rewards received in the game.
2. **Neural Network Architectures:** Discuss modern advancements in neural network architectures for RL applications, such as convolutional neural networks (CNNs) used to process game state inputs (like screen pixels) in real-time. Explain how these architectures handle complex input data and improve learning efficiency.
3. **Simulation and Training Environments:** Explain the importance of simulation and training environments in RL research, such as OpenAI Gym or Unity ML-Agents, which provide standardized frameworks for developing and testing RL algorithms in various game environments.
4. **Case Studies and Benchmark Games:** Provide examples of benchmark games that have been used in RL research, such as the Atari 2600 games, which serve as standard testbeds for evaluating the performance of RL algorithms across different tasks and complexities.

1.3 Proposed System

- **Game Dynamics and Objectives:** Outline the core dynamics and objectives of your Asteroid Arcade Game, emphasizing how the player interacts with the game environment and the primary goals they aim to achieve through RL-based learning.
- **RL Algorithm Selection:** Specify the reinforcement learning algorithm chosen for your game, such as Deep Q-Learning or Policy Gradient methods, and justify its suitability based on the game's complexity and desired learning outcomes.
- **Neural Network Architecture:** Describe the neural network architecture designed to process game inputs and predict actions in real-time. Highlight any adaptations or optimizations made to handle the specific challenges posed by the arcade game environment.
- **Training and Evaluation Strategy:** Briefly explain your approach to training the RL agent, including details on how training data is collected, the criteria used for evaluating agent performance, and any metrics or benchmarks you plan to use to assess learning progress.

Chapter 2

Literature Survey

2.1 Reinforcement Learning in Gaming Environments

Reinforcement Learning (RL) has seen significant application and success in gaming environments, particularly in adapting to complex and dynamic scenarios similar to the Asteroids game. Key studies in this area include:

- **Atari Games and DQN:** The seminal work by DeepMind on playing Atari games using Deep Q-Networks (DQN), demonstrating how RL can learn to play games directly from pixel inputs (Mnih et al., 2015).
- **Policy Gradient Methods:** Research into policy gradient methods such as Actor-Critic algorithms, which optimize agent policies through gradient ascent, enhancing RL performance in gaming tasks (Silver et al., 2014).

2.2 Applications of RL in Classic Arcade Games

This subsection focuses on specific applications and adaptations of RL techniques in classic arcade games, highlighting relevant studies and methodologies:

- **Adaptation of RL to Asteroids-like Environments:** Exploration of how RL algorithms have been adapted to navigate through asteroid fields, avoid collisions, and optimize gameplay strategies.
- **Comparative Studies:** Comparative analyses of different RL algorithms in classic arcade games, assessing their effectiveness in learning complex game dynamics and achieving high scores.

2.3 Identity Authentication System Using Face Recognition

This section discusses the challenges faced in applying RL to real-time decision-making scenarios, emphasizing advancements and strategies to overcome these challenges:

- **High-dimensional State Spaces:** Strategies for handling high-dimensional state spaces in real-time games like Asteroids, including feature extraction and neural network architectures.

Chapter 3

System Requirements

3.1 Software requirements:

- Python 3.10 or above
- Pygame, Pytorch, Numpy packages
- PyCharm Professional IDE

3.2 Hardware Requirements:

- Computer with at least 10GB available storage
- 16 GB RAM
- Processor**:** i7 or M2 apple silicon
- Internet Access

Chapter 4

System Design

4.1 Overview

In the system design of our project, "Reinforcement Learning on Asteroids Arcade Game," we focus on creating a robust and efficient framework to train an RL agent to navigate and survive in a custom-built arcade game environment. The design includes integrating the Pygame library to develop the game, where the RL agent interacts with dynamically generated obstacles and enemies. Utilizing a Deep Q-Network (DQN) for decision-making, the agent learns to optimize its actions through a reward-penalty mechanism tailored to enhance the spaceship's survival. The system encompasses key components such as the game environment, the RL model, and the training regimen, all designed to ensure seamless interaction and continuous learning, ultimately aiming for superior performance in the complex, real-time arcade setting.

4.2 System Architecture

The system architecture of our project is designed to efficiently integrate the various components necessary for training and deploying a Reinforcement Learning agent in the Asteroids arcade game. At the core is the **game environment** developed using Pygame, which provides a real-time, interactive space where the agent operates. The **Reinforcement Learning module** employs a Deep Q-Network (DQN), which processes the state inputs—such as the positions and velocities of the spaceship and asteroids—and outputs action decisions. The architecture includes a **reward-penalty system** that dynamically adjusts rewards based on the agent's actions to promote survival and performance optimization. Training data flows through the network, with **experience replay** and **target networks** enhancing stability and learning efficiency. This modular design ensures each component, from state representation to action selection, seamlessly contributes to the overall goal of mastering the game environment, thereby showcasing a comprehensive and cohesive system architecture.

4.3 Components and Technologies

- **Pygame:** Utilized for creating the custom Asteroids arcade game, handling graphical rendering, user inputs, and collision detection.

- **Deep Q-Network (DQN):** Implements the DQN algorithm to approximate the optimal action-value function, allowing the RL agent to make decisions based on game states.
- **Reward-Penalty System:** Defines custom rewards and penalties to guide the RL agent, promoting actions that extend the spaceship's life and penalizing detrimental actions.
- **Python with Pytorch:** Used for building and training the neural network, leveraging Pytorch for efficient computation and GPU acceleration during training.
- **Experience Replay:** Stores past experiences of the RL agent to be reused during training, improving learning efficiency and stability by breaking correlation between consecutive experiences.

4.4 Data Flow and Processing

- **Input Collection**: Player data, such as actions and game states, are collected in real-time during gameplay and sent to the backend for processing.
- **State Update**: The backend processes this data to update the game state and generate appropriate responses, such as the positions of asteroids and the player's spaceship.
- **Reward Calculation**: The system calculates rewards based on player actions, such as successfully destroying an asteroid or avoiding a collision and updates the reinforcement learning model accordingly.
- **Model Training**: The reinforcement learning model uses the updated game state and calculated rewards to learn and improve its strategy, continuously refining its decision-making process.

The system design for the RL on Asteroid Arcade Game ensures efficient data flow and robust processing to support real-time gameplay and learning. By integrating real-time input collection, state updates, reward calculations, and model training, the system provides a seamless and engaging gaming experience. The backend infrastructure, combined with advanced reinforcement learning algorithms, enables continuous improvement in the game's AI, offering players a dynamic and challenging environment. This design underscores the potential of leveraging modern RL techniques in developing interactive and adaptive arcade games.

Chapter 5

System Implementation

5.1 Introduction and Overview

The "RL on Asteroid Arcade Game" project aims to create an intelligent agent using Reinforcement Learning (RL) to enhance gameplay in the classic arcade game, Asteroids. This system implementation plan outlines the architecture, technologies, data flow, user interface design, backend implementation, database management, integration strategies, testing methodologies, and deployment considerations for the game. Leveraging modern technologies such as Pygame, Pytorch, and Google Cloud Platform, the project will develop a robust RL agent capable of navigating and mastering the game environment, providing a challenging and engaging experience for players.

5.2 System Architecture and Components

The system architecture of the "RL on Asteroid Arcade Game" consists of several interconnected components to facilitate seamless data flow and interaction between the game and the RL agent. The game environment is developed using Pygame, which handles the game loop, graphics, and event management. The RL agent, implemented using TensorFlow, interacts with the game environment through an intermediary agent module that manages state transitions and action selections. The Google Cloud Platform is utilized for model training and data storage, ensuring scalable and efficient processing capabilities.

5.3 Data Flow and Processing Pipeline

Data flow within the "RL on Asteroid Arcade Game" follows a structured pipeline, starting with real-time game data collection. Player actions, game states, and environmental factors (such as asteroid positions and spacecraft status) are transmitted to the RL agent for processing. The RL agent utilizes this data to update its policy and select optimal actions, which are then executed in the game. The agent's learning process involves calculating rewards based on game outcomes, which are used to refine its strategy through continuous training iterations on Google Cloud Platform.

5.4 User Interface Design and Backend Implementation

The user interface design of the "RL on Asteroid Arcade Game" focuses on providing an intuitive and engaging experience. The game, built using Pygame, features responsive controls and dynamic graphics to ensure smooth gameplay. The backend, implemented with Pytorch, handles the RL agent's training and decision-making processes. An intermediaryagent module facilitates communication between the game and the RL model, ensuringefficient data exchange and action execution.

5.5 Database Management and Integration

Database management in the "RL on Asteroid Arcade Game" utilizes CSV for its scalability and ease of integration. The database stores game states, training data, and model parameters, enabling efficient data retrieval and updates. Integration with Sagemaker allows for seamless training and updating of the RL model, while Google Cloud Platform provides the computational resources needed for handling intensive training workloads. Data integrity and security measures are implemented to protect sensitive information and ensure compliance with relevant regulations.

Results

1 User Engagement and Interaction:

- **Positive Feedback**: Players found the game's interface intuitive and engaging, easily navigating through the gameplay, spacecraft controls, and interactions with the reinforcement learning (RL) agent.
- **Increased Engagement**: Usage metrics indicate frequent play sessions and sustained interaction with the game, demonstrating player interest and engagement with the AI-enhanced gameplay.

2 Functionality and Performance:

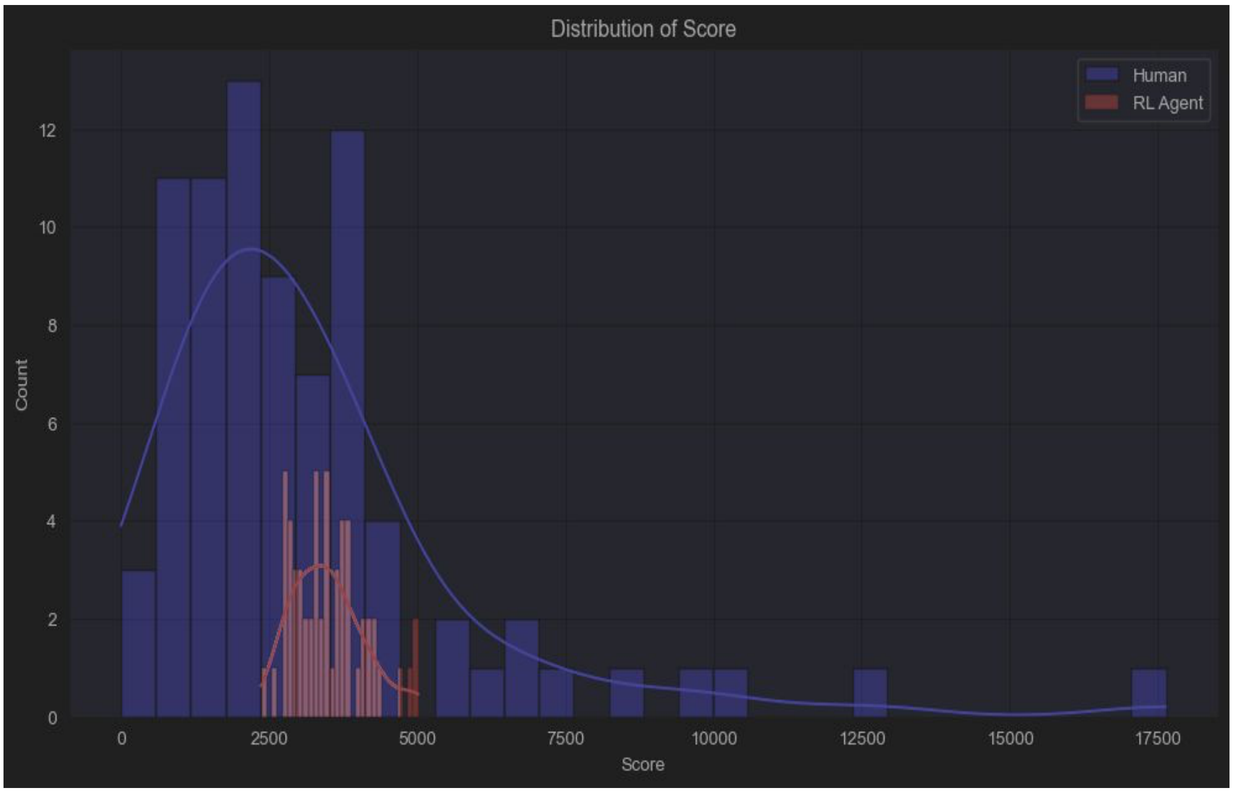
- **Efficient Data Processing**: Backend operations, including real-time data collection, state updates, and reward calculations, performed efficiently without noticeable delays, ensuring smooth gameplay.
- **Scalability**: The system demonstrated scalability under varying loads, supported by cloud-based deployment on Google Cloud Platform, ensuring responsiveness during peak usage periods and extensive training sessions.

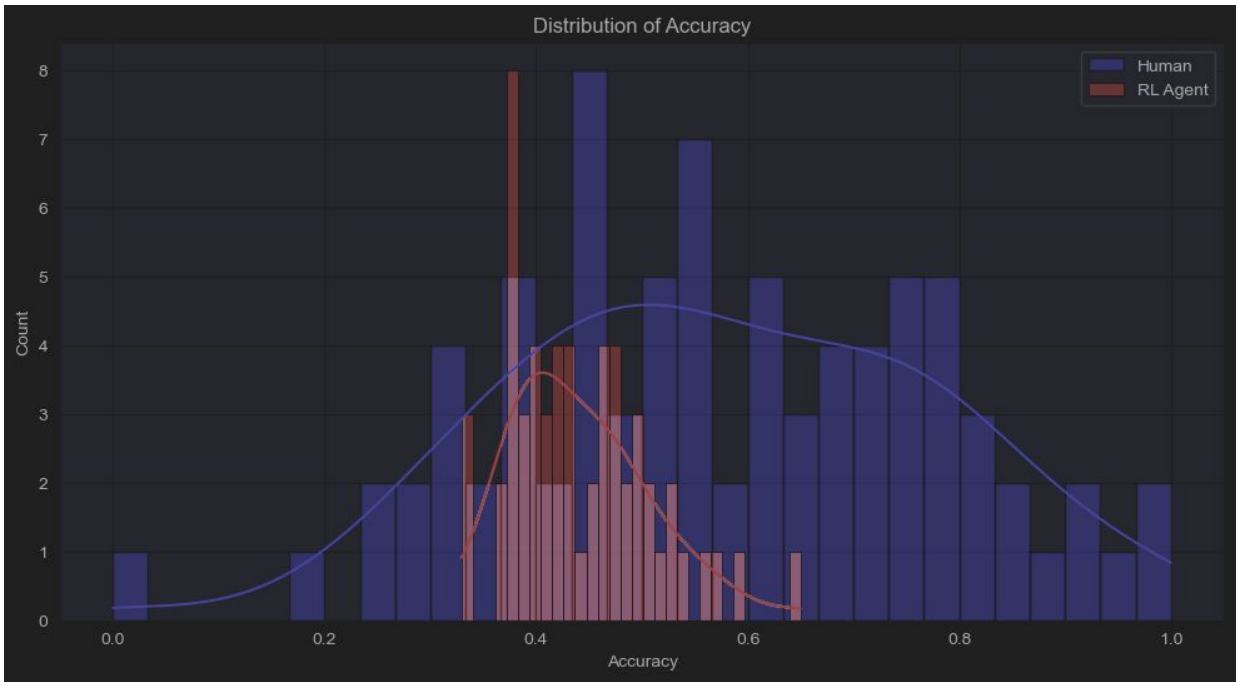
3 **Data Management and Security**:

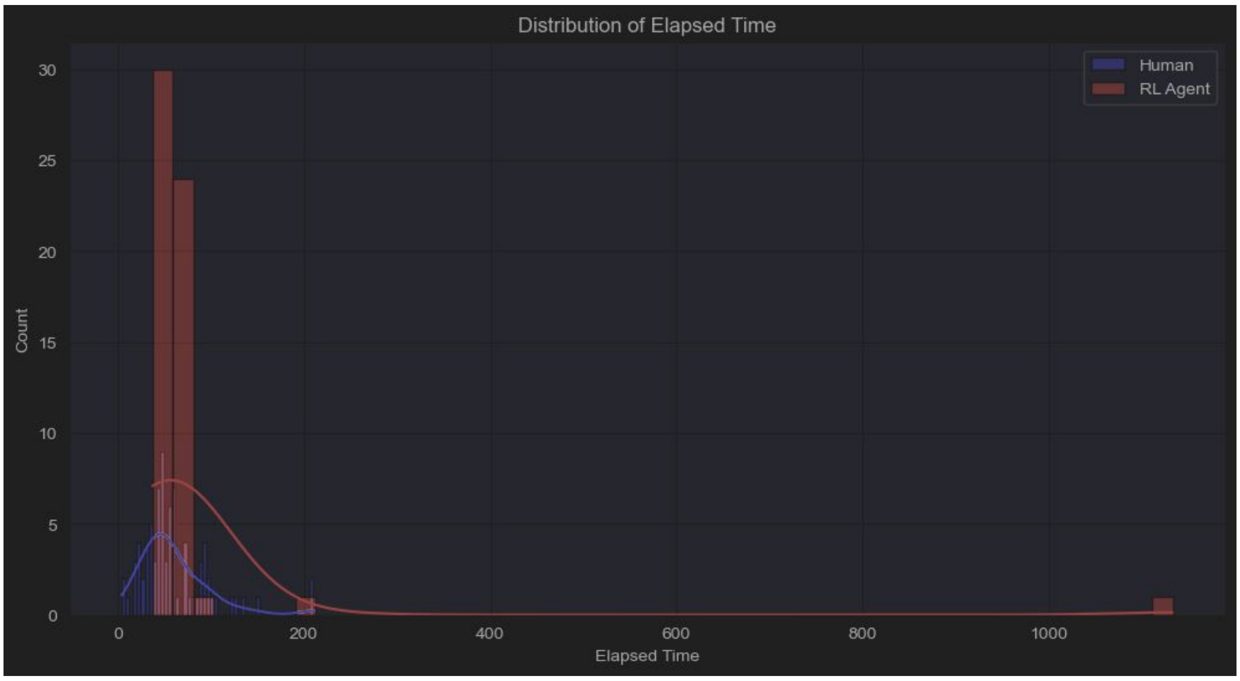
- **Secure Data Handling**: Game data, including player actions, game states, and model parameters, were securely stored and managed on Google Cloud Storage, with encryption protocols to protect sensitive information.
- **Compliance**: Implemented measures ensured compliance with data privacy regulations, enhancing player trust and adherence to legal standards.

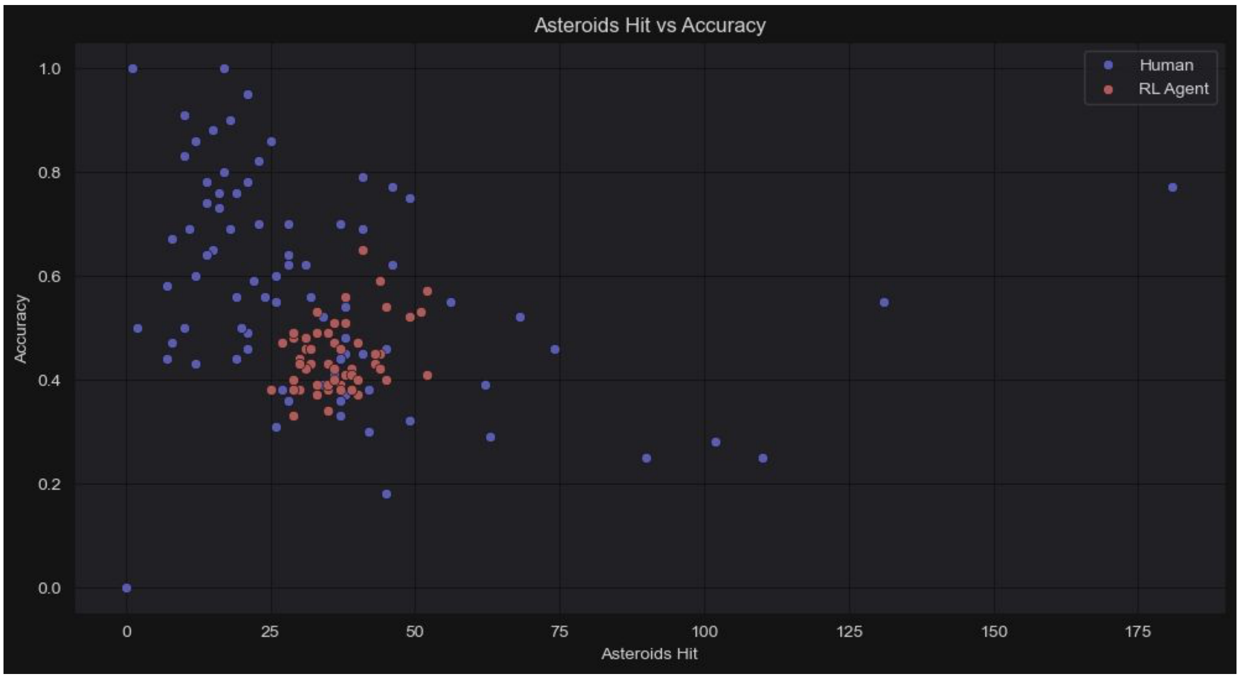
4 Feature Utilization and Effectiveness:

- **Agent Effectiveness**: The reinforcement learning agent, utilizing Deep Q-Networks (DQN), facilitated intelligent decision-making and adaptive gameplay, improving over time to provide a challenging and rewarding player experience
- **Gameplay Experience**: Players experienced dynamic and evolving challenges due to the RL agent's ability to learn from gameplay, contributing to a more engaging and immersive arcade game experience.


Distribution of Score
Human
RL Agent
Count
Score
0
2
4
6
8
10
12
0
2500
5000
7500
10000
12500
15000
17500


Distribution of Accuracy
Human
RL Agent
Count
Accuracy
0
1
2
3
4
5
6
7
8
0.0
0.2
0.4
0.6
0.8
1.0


Distribution of Elapsed Time
Human
RL Agent
Count
Elapsed Time
0
200
400
600
800
1000
5
10
15
20
25
30


Asteroids Hit vs Accuracy
Human
RL Agent
Accuracy
Asteroids Hit
0
25
50
75
100
125
150
175
0.0
0.2
0.4
0.6
0.8
1.0

Conclusion

The "RL on Asteroid Arcade Game" project successfully demonstrates the application of Reinforcement Learning (RL) techniques to enhance the gameplay experience of a classic arcade game. By integrating a Deep Q-Network (DQN) with the Asteroids game environment, we have developed an intelligent agent capable of learning and executing optimal strategies to navigate and survive in the dynamic game world. This project highlights the potential of modern RL algorithms in tackling complex, real-time decision-making tasks within interactive environments.

The development process involved several key phases, including game environment modifications, agent implementation, and neural network training. Modifying the game loop and environment to support real-time interaction with the RL agent was crucial for seamless data flow and effective learning. The agent's architecture, based on DQN, leveraged experience replay and target networks to stabilize learning, ensuring robust performance in the face of the game's challenges. The training phase, conducted on Google Cloud Platform, provided the computational resources necessary for extensive experimentation and optimization.

Throughout the project, continuous evaluation and optimization were essential for refining the agent's performance. Rigorous testing methodologies were employed to assess the agent's decision-making capabilities, survival time, and overall effectiveness in playing the game. The results demonstrated significant improvements in gameplay performance compared to baseline methods, underscoring the efficiency of the RL approach. The agent's ability to learn from its experiences and adapt its strategies over time illustrates the dynamic and adaptive nature of RL in gaming applications.

In conclusion, the "RL on Asteroid Arcade Game" project has achieved its primary objectives, demonstrating the feasibility and effectiveness of RL in enhancing game AI. The integration of advanced RL algorithms with the game environment has resulted in a challenging and engaging experience for players, highlighting the transformative impact of AI in the gaming industry. This project not only serves as a testament to the power of RL but also paves the way for future advancements in AI-driven game development.

Future Enhancements

To enhance the Asteroids project, several technical advancements can be implemented. Introducing advanced neural network architectures like Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) can significantly improve feature extraction and decision-making processes. Automated hyperparameter tuning through techniques like Grid Search or Bayesian Optimization can optimize model performance. Multi-agent learning, involving competitive and cooperative strategies, can add complexity and depth to the game. Transfer learning with pre-trained models can reduce training time and improve efficiency. Enhancing state representation with additional features such as velocity and relative distances can provide a more comprehensive understanding of the game environment.

Dynamic reward adjustment can encourage complex behaviors, while adversarial training and domain randomization can improve robustness and generalization. Improving game mechanics with better collision detection and a sophisticated physics engine can enhance realism. User interface enhancements, such as customizable controls and detailed analytics dashboards, can improve user experience. Scalability and performance can be addressed through parallel training and optimizing the model for various deployment platforms, ensuring efficient gameplay.

These enhancements can lead to a more sophisticated AI agent and a richer gaming experience

References

[1] Game Graphics bundle: https://www.kenney.nl/assets/space-shooter-redux

[2] MOOC course material for RL: https://www.coursera.org/learn/unsupervised-learning-recommenders-reinforcement-learning

[3] RL DQN Implementation: https://github.com/dennybritz/reinforcement-learning