Transformer Reinforcement Learning

Ambati Hari Charan
Dept. of CSE
IIIT Vadodara
Gandhinagar 382028, India
202151019@iiitvadodara.ac.in

Arghadeep Dey
Dept. of CSE
IIIT Vadodara
Gandhinagar 382028, India
202151027@iiitvadodara.ac.in

Neha Singh
Dept. of CSE
IIIT Vadodara
Gandhinagar 382028, India
202151098@iiitvadodara.ac.in

***Abstract*—This report examines the application of Transformer Reinforcement Learning (TRL) and Q-Learning models in two environments: Lunar Lander and Cartpole. Reinforcement Learning (RL) enables agents to learn optimal behaviors through interactions with their environment. Transformers, initially designed for natural language processing, show potential in enhancing RL by effectively handling long-range dependencies and sequential data.**

We explore the integration of Transformers into RL, creating TRL models tailored for these environments. Experimental results indicate that while Q-Learning excelled in the simpler Cartpole task with higher rewards and stability, the TRL model outperformed in the more complex Lunar Lander environment, demonstrating superior rewards and consistency.

These findings suggest that while Q-Learning is effective for simpler tasks, TRL offers significant benefits in complex scenarios, marking a promising direction for future research in advanced RL techniques. The report concludes with a discussion of results, challenges, and future directions.

***Index Terms*—reinforcement learning, transformers, q learning, replay memory, decision transformers**

I. Introduction

Reinforcement Learning (RL) is a branch of machine learning where an agent learns to make decisions by interacting with an environment to maximize some notion of cumulative reward. Traditional RL methods have shown remarkable success in various domains, yet they often struggle with environments requiring long-term planning and handling sequential dependencies. To address these challenges, recent research has explored integrating advanced deep learning architectures, such as Transformers, into RL frameworks.

Transformers, originally developed for natural language processing tasks [1], excel at capturing long-range dependencies and processing sequential data. Their self-attention mechanism allows for efficient handling of sequential information, making them a suitable candidate for enhancing RL algorithms. Notably, the application of Transformers in RL has shown promising results in various studies [2], [3].

This report investigates the application of Transformer Reinforcement Learning (TRL) in two benchmark environments: Lunar Lander and Cartpole. The Lunar Lander environment, provided by the Gymnasium library, involves controlling a lander to navigate and land on a designated pad. This environment is characterized by continuous state and action spaces, requiring the agent to learn precise control strategies to succeed. On the other hand, the Cartpole environment involves balancing a pole on a moving cart, a classic RL problem with discrete actions and continuous states. Both environments serve as standard testbeds for evaluating the effectiveness of RL algorithms.

In this study, we develop TRL models for both the Lunar Lander and Cartpole environments. The primary contributions of this research are:

1) **Transformer-based RL Model Implementation** : We designed and implemented Transformer-based models tailored to the Lunar Lander and Cartpole environments. These models incorporate multiple transformer blocks to process state embeddings and output action predictions.
2) **Performance Evaluation** : The TRL models were trained and evaluated in both environments. We analyzed performance metrics such as stability, convergence speed, and overall reward compared to traditional RL models.
3) **Theoretical Insights and Practical Challenges** : We provide a comprehensive discussion on the theoretical benefits of integrating Transformers in RL and the practical challenges encountered during implementation and training.

Our findings indicate that TRL can significantly enhance the performance and learning efficiency in the Lunar Lander while there is still scope for improvement in the Cartpole environment. Specifically, the TRL models demonstrated improved stability and faster convergence, suggesting that Transformers can effectively capture the complex dependencies in the state-action space.

This report is organized as follows: Section II discusses related work in RL and Transformers. Section III details some of the preliminary details used in our work. Section IV details the methodology, including the architecture of the TRL models, the training process, as well as the results obtained. Finally, Section V concludes with a summary of findings and potential future research directions.

II. Relevant Work

Recent advancements in machine learning have seen the convergence of Reinforcement Learning (RL) and Transformer architectures, yielding significant improvements in various tasks. This section highlights key contributions that have influenced our approach to applying Transformer Reinforcement Learning (TRL) in benchmark environments.

A. Reinforcement Learning and Deep RL

RL has been extensively studied, with foundational algorithms like Q-learning [5] and policy gradient methods [4] providing the basis for numerous applications. The advent of Deep Reinforcement Learning (DRL) marked a significant milestone, with Mnih et al.'s Deep Q-Network (DQN) demonstrating human-level performance on Atari games [6]. Subsequent improvements, such as Trust Region Policy Optimization (TRPO) [7] and Proximal Policy Optimization (PPO) [8], have enhanced the stability and efficiency of DRL methods.

B. Transformers in Sequential Decision Making

Transformers, introduced by Vaswani et al. [1], have revolutionized natural language processing through their self-attention mechanism, enabling efficient handling of sequential data. Their application has extended beyond NLP to fields like computer vision and RL. Dosovitskiy et al. adapted Transformers for image recognition with notable success [9], paving the way for their use in sequential decision-making tasks.

C. Integrating Transformers with Reinforcement Learning

The integration of Transformers into RL has shown promising results. Parisotto et al. introduced the GTrXL architecture, which incorporates gating mechanisms and layer normalization to stabilize Transformer training in RL, demonstrating enhanced performance in environments with long-term dependencies [2]. Chen et al.'s Decision Transformer frames RL as a sequence modeling problem, leveraging Transformers to predict actions based on past states and rewards, showing improved results in various RL tasks [3].

D. Benchmark Environments

Benchmark environments such as Lunar Lander and Cartpole, provided by the OpenAI Gymnasium suite, offer standardized testbeds for evaluating RL algorithms. Traditional methods like DQN and PPO have been widely tested in these environments, establishing baseline performance metrics. Our study builds on this foundation, applying TRL to these environments to assess the potential improvements in learning efficiency and performance.

In this work, we develop and evaluate TRL models for the Lunar Lander and Cartpole environments, leveraging the self-attention mechanism of Transformers to capture complex dependencies in the state-action space. Our findings suggest that TRL models can achieve improved stability and faster convergence compared to traditional RL methods.

III. Preliminaries

A. Reinforcement Learning Basics

- **Q-Learning** : A value-based RL algorithm where the agent learns a Q-function that estimates the expected cumulative reward for taking an action in a given state and following the optimal policy thereafter. The Q-learning update rule is given by:

Algorithm 1: Q-Learning Update

Result: Q-Learning Update
Initialize $Q(s, a)$ arbitrarily for all state-action pairs;
for *each episode* **do**
  Initialize S;
  for *each step of episode* **do**
    Choose A from S using policy derived from Q (e.g., ϵ-greedy);
    Take action A, observe R, S';
    $Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma \max_{a'} Q(S', a') - Q(S, A)]$;
    $S \leftarrow S'$;
  end
end

- **SARSA** : Similar to Q-learning but updates the Q-value based on the action actually taken in the next state. The SARSA update rule is:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha \delta_t \tag{1}$$

$$\text{where } \delta_t = R_{t+1} + \gamma Q(S_{t+1}, A_{t+1}) - Q(S_t, A_t) \tag{2}$$

- **Policy-Gradient Methods** : Methods that directly parameterize the policy and optimize it using gradient ascent. The policy gradient update rule is:

$$\theta \leftarrow \theta + \alpha \nabla_\theta \log \pi_\theta(a|s) Q(s, a) \tag{3}$$

- **Softmax Policy** : A policy where the probability of selecting an action is given by a softmax function:

$$\pi(a|s) = \frac{e^{Q(s,a)/\tau}}{\sum_b e^{Q(s,b)/\tau}} \tag{4}$$

B. Transformer Architecture

- **Self-Attention Mechanism** : Allows the model to weigh the importance of different elements in the input sequence, capturing long-range dependencies efficiently.
- **Positional Encoding** : Adds information about the positions of the tokens in the sequence, enabling the model to understand the order of the sequence.

IV. Methodology

A. Overview

In this section, we describe the implementation details of our Transformer Reinforcement Learning (TRL) models applied to the Lunar Lander and Cartpole environments. We discuss the architecture, training procedures, and evaluation metrics used to assess the performance of our models.

B. Lunar Lander Environment

*1) **Environment Description**:* The Lunar Lander environment is a well-known benchmark in reinforcement learning, provided by OpenAI Gymnasium. The objective is to control a lander to touch down smoothly on a designated landing pad. The environment presents challenges including managing fuel consumption and avoiding collisions with the surface. The state space includes variables like the lander's position, velocity, and angle, while the action space consists of discrete actions to fire the main engine or side thrusters.

*2) **Model Structure**:*

1) Deep Q-Learning Model : In our initial approach, we implemented a Deep Q-Learning (DQN) model. This model uses a neural network to approximate the Q-function, which estimates the expected cumulative reward for each action in a given state. The neural network comprises an input layer matching the state space dimensions, two hidden layers with ReLU activation functions, and an output layer representing the Q-values for each action. The equation is as discussed in Algorithm 1.
2) Transformer Reinforcement Learning Model : We subsequently applied a Transformer-based architecture to the Lunar Lander environment. The model incorporates self-attention mechanisms to capture long-range dependencies and enhance decision-making. The architecture includes an input layer for state representation, multiple Transformer encoder layers with multi-head self-attention, and an output layer for Q-values.

*3) **Training Procedure**:* Both models were trained using the same procedure to ensure a fair comparison:

- Data Collection : Experience data was collected through interactions with the environment, with an ϵ-greedy strategy for exploration.
- Loss Function: The mean squared error loss between the predicted Q-values and the target Q-values was minimized.
- Optimization Algorithm: The Adam optimizer was used with a learning rate of 0.001.

$$\text{Loss} = \left(r + \gamma \max_{a'} Q(s', a') - Q(s, a)\right)^2 \tag{5}$$

*4) **Evaluation Metrics**:*

- Average Reward : The average reward per episode was tracked to assess the model's performance.
- Episodes to solve : The number of episodes required to reach a solved state (average reward over 200) was recorded.

*5) **Results**:*

- Deep Q-Learning Model: The environment was solved in **596** episodes, achieving an average reward of **208.215**, at a standard deviation of **62.8**.
- Transformer Reinforcement Learning Model: The environment was solved in **663** episodes, achieving an average reward of **225.417** and a total reward of **276.3**. The standard deviation was **44.44**.

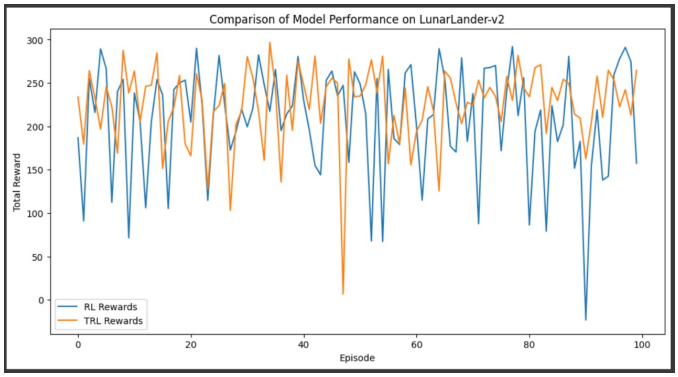

Fig. 1. Comparison of model performance on the Lunar Lander environment

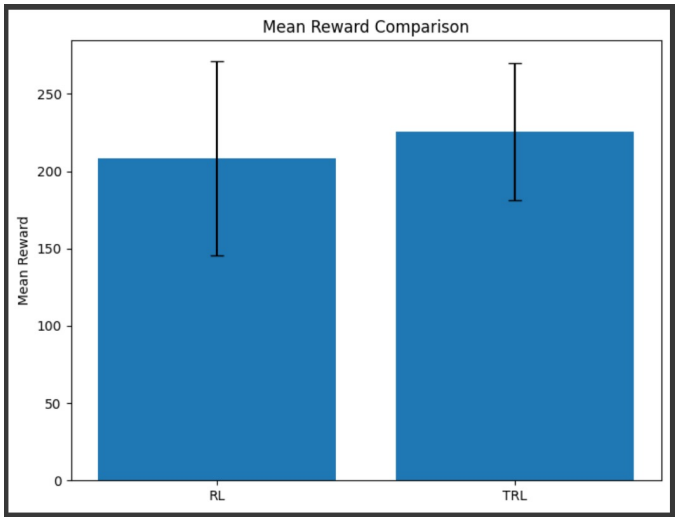

Fig. 2. Comparison of Mean Rewards obtained in RL and TRL models

*6) **Implementation Challenges**:* Implementing the Transformer model presented several challenges, including the increased computational requirements and the need for tuning hyperparameters to ensure stability during training. To address these challenges, we applied techniques such as gradient clipping and learning rate scheduling, which helped stabilize the training process and improve performance.

C. CartPole Environment

*1) **Environment Description**:* The CartPole environment is a classic control problem provided by OpenAI's Gymnasium library. The task involves balancing a pole on a moving cart. The cart can move left or right, and the objective is to keep the pole upright for as long as possible. The state space is continuous, represented by a four-dimensional vector comprising of the cart position, cart velocity, pole angle, pole velocity at the tip.

The action space is discrete with two possible actions, either moving the cart to the left or to the right. The environment terminates if the pole falls over by more than 15 degrees from the vertical or if the cart moves more than 2.4 units from the

center. The reward structure is straightforward: a reward of +1 for each time step the pole remains upright.

*2) **Model Structure**:* Once again, we use a Q-learning and a Transformer Reinforcement Learning model and compare their differences.

1) Q-Learning : The parameters we used were :
   - Learning rate (α) : 0.1 , which controls the extent to which new information overrides the old information. A higher learning rate can lead to faster learning but may also cause instability.
   - Discount Factor (γ) : 0.99, which determines the importance of future rewards. A value close to 1 means that the agent considers future rewards as important as immediate rewards.
   - Exploration Rate (ϵ) : Decaying from 1.0 to 0.01. Initially, the agent explores the environment randomly, and over time it exploits the learned policy more.
2) Transformer Reinforcement Learning Model : The TRL model leverages the transformer architecture, which has shown great success in sequence modeling tasks. The state input is reshaped and fed into a transformer network designed to predict Q-values for each action. The architecture includes:
   - Input layer: Reshapes the state to (1, 1, 4) to fit the transformer input requirements.
   - Transformer layers: Multiple self-attention layers with positional encoding, which allow the model to focus on different parts of the state space.
   - Output layer: Fully connected layer to produce Q-values for the action space, translating the transformed representation into actionable outputs.

*3) **Training Procedure**:* Both the Q-Learning and Transformer Reinforcement Learning (TRL) models were trained over 500 episodes. Each episode began with resetting the environment and using a policy to select actions. The Q-Learning model used an epsilon-greedy policy, while the TRL model predicted Q-values using a transformer network.

For Q-Learning, Q-values were updated directly after each action. In contrast, the TRL model stored experiences in a replay buffer, sampled mini-batches for training, and used a periodically updated target network for stability. The exploration rate decayed over time in both models to balance exploration and exploitation.

The main difference is the use of a replay buffer and target network in the TRL model, which improves learning efficiency and stability.

*4) **Evaluation Metrics**:* The performance of both models was evaluated using the following metrics:

- Average Reward: The mean reward achieved per episode over the evaluation period.
- Maximum Reward: The highest reward achieved in any single episode.
- Minimum Reward: The lowest reward achieved in any single episode.

*5) **Results**:* The Q-Learning model achieved an average reward of **129.37**, with a maximum reward of **138** and a minimum of **121**. This indicates a moderate performance with some consistency in keeping the pole balanced for a reasonable number of time steps.

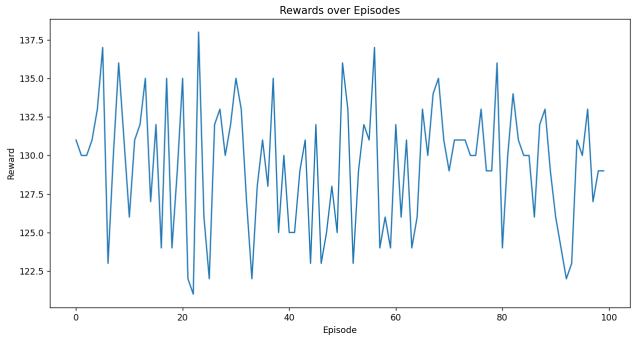

Fig. 3. Graph showing the rewards over episodes for the Q-learning model

In contrast, the TRL model obtained an average reward of **100.65**. This lower performance suggests that the model may not have effectively learned the task or might require further tuning and additional training episodes to realize its potential.

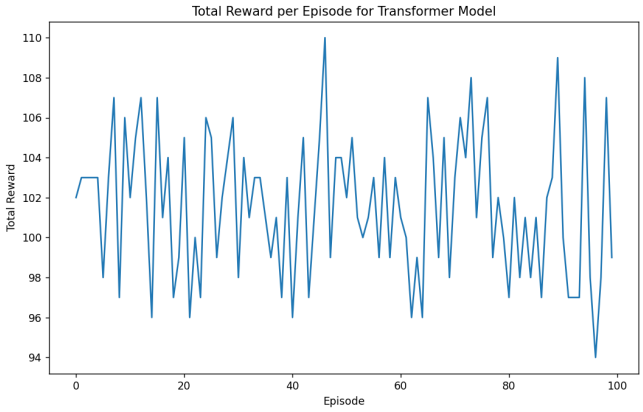

Fig. 4. Graph showing the rewards over episodes for the TRL model

*6) **Implementation Challenges**:* Implementing the TRL model posed several challenges:

1) State Representation: The continuous nature of the state space required careful preprocessing and normalization to fit into the transformer model.
2) Replay Buffer Management: Ensuring efficient storage and retrieval of experiences to train the transformer network, which is crucial for effective learning.
3) Training Stability: Balancing the training between the main network and the target network to avoid divergence. This included managing the update frequency of the target network to maintain stability.
4) Hyperparameter Tuning: Identifying optimal values for learning rates, batch sizes, and transformer-specific parameters. The complexity of the transformer model

introduces additional hyperparameters that need careful tuning to achieve good performance.

5) Computational Resources: Training transformer models typically requires more computational power and memory, which can be a limiting factor compared to simpler models like Q-Learning.

Despite these challenges, the exploration of transformers for reinforcement learning in the CartPole environment provides valuable insights and a foundation for further research and improvement. Further tuning of the model and extended training periods may help in realizing the full potential of the TRL approach.

V. Conclusion

In this study, we explored the application of Q-Learning and Transformer Reinforcement Learning (TRL) models to the CartPole and Lunar Lander environments. The Q-Learning model demonstrated superior performance on the CartPole task, achieving higher average rewards and stability. However, the TRL model excelled in the Lunar Lander environment, achieving higher average rewards and more consistent performance with a lower standard deviation.

The results indicate that while Q-Learning remains effective for simpler environments, TRL models offer significant advantages in more complex scenarios. This suggests a promising direction for future research in leveraging advanced neural network architectures for reinforcement learning tasks.

VI. Future Work

Future research could explore the following areas:

1) Hybrid Approaches: Developing hybrid models that integrate the strengths of Q-Learning and TRL to enhance performance across diverse environments.
2) Hyperparameter Optimization: Conducting comprehensive hyperparameter tuning for TRL models to improve their applicability and performance in simpler tasks.
3) Complexity Analysis: Investigating the impact of model complexity on different types of environments to determine optimal model configurations.
4) Transfer Learning: Implementing transfer learning techniques to apply knowledge from one environment to another, potentially improving learning efficiency and generalization.
5) Scalability Testing: Extending the evaluation of TRL models to more complex and real-world environments to assess their scalability and robustness.

By addressing these areas, future work can further enhance the efficacy and applicability of reinforcement learning models, particularly those utilizing transformer architectures, in a broader range of tasks and environments.

Acknowledgment

We would like to thank Dr. Bhupendra Kumar (Dept. of ECE, IIIT Vadodara) for his continued support in making this report possible. His guidance and insights helped us in getting on the right track and equipped us with the necessary knowledge.

References

[1] A. Vaswani et al., "Attention is All You Need," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

[2] E. Parisotto et al., "Stabilizing Transformers for Reinforcement Learning," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[3] L. Chen et al., "Decision Transformer: Reinforcement Learning via Sequence Modeling," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[4] R. S. Sutton and A. G. Barto, "Reinforcement Learning: An Introduction," MIT Press, 1998.

[5] C. J. C. H. Watkins and P. Dayan, "Q-learning," Machine Learning, vol. 8, no. 3-4, pp. 279–292, 1992.

[6] V. Mnih et al., "Human-level control through deep reinforcement learning," Nature, vol. 518, no. 7540, pp. 529–533, 2015.

[7] J. Schulman et al., "Trust Region Policy Optimization," in *Proceedings of the International Conference on Machine Learning (ICML)*, 2015.

[8] J. Schulman et al., "Proximal Policy Optimization Algorithms," arXiv preprint arXiv:1707.06347, 2017.

[9] A. Dosovitskiy et al., "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," arXiv preprint arXiv:2010.11929, 2020.

[10] A. Hari Charan, A. Dey, and N. Singh, "Transformer Reinforcement Learning," GitHub repository, https://github.com/arghadeep23/trl, 2024.