Analyzing the Impact of Hyperparameters in AWS DeepRacer

Claudio A. Cespedes
Texas State University
United States
cac570@txstate.edu

Ethan Fernandez
Austin College
United States
efernandez22@austincollege.edu

Kecheng Yang
Texas State University
United States
yangk@txstate.edu

ABSTRACT

Optimizing hyperparameters is crucial for the performance of reinforcement learning (RL) models. This study examines the effects of batch size, learning rate, discount factor (gamma), and epsilon on RL models using the AWS DeepRacer platform. Extensive experiments revealed that smaller batch sizes and lower learning rates generally improved completion rates and cleaner laps. Higher gamma values favored long-term rewards, enhancing overall performance, while lower epsilon values led to better exploitation. These findings provide valuable insights into hyperparameter tuning, offering practical guidelines for optimizing RL models in autonomous racing and similar applications, and underscore the importance of systematic hyperparameter tuning in RL.

1 INTRODUCTION

Autonomous driving has emerged as a significant application of artificial intelligence, aiming to revolutionize transportation by enhancing safety and efficiency. Reinforcement learning (RL), a subfield of machine learning, has proven to be particularly effective in developing models for autonomous vehicles [3]. One notable platform for experimenting with RL in autonomous driving is the AWS DeepRacer, a 1/18th scale race car designed to test RL models on a physical track or in a simulated environment.

AWS DeepRacer leverages RL to train models that can navigate a track autonomously by learning from interactions with the environment [7]. While the platform offers an accessible means for researchers and enthusiasts to delve into RL, optimizing the performance of DeepRacer models remains a complex task. The choice of hyperparameters plays a crucial role in determining the effectiveness of the trained models.

This study aims to investigate the impact of various hyperparameters on the performance of AWS DeepRacer models. Specifically, we examine the effects of batch size, learning rate, discount factor (gamma), and e-greedy-value (epsilon) on model performance. By conducting a series of experiments, we aim to provide insights into how these hyperparameters influence the learning process and overall performance of the models.

2 BACKGROUND

AWS DeepRacer is an autonomous 1/18th scale race car designed to test reinforcement learning (RL) models. It provides a hands-on experience for developers to experiment with RL algorithms in a physical environment[2]. This platform includes a 3D racing simulator and a robust community-driven learning environment, making it an ideal tool for studying RL techniques and their practical applications. The model can be trained in this virtual environment and be translated into the real time physical car. In this environment the hyperparameters and reward function can be tuned for the user's purpose. The AWS DeepRacer environment has been used extensively and extended further driven by its community.

2.1 Reinforcement Learning Overview

Reinforcement learning is a type of machine learning where an agent learns to make decisions by performing actions in an environment to maximize cumulative reward [4]. The agent interacts with the environment in discrete time steps, observes the state, takes actions, and receives rewards. The goal is to learn a policy that maximizes the expected cumulative reward over time.

Key concepts in RL include:

- **Agent**: The learner or decision-maker.
- **Environment**: Everything the agent interacts with.
- **State**: A representation of the current situation of the agent.
- **Action**: All possible moves the agent can take.
- **Reward**: The feedback from the environment based on the action taken.
- **Policy**: The strategy used by the agent to determine the next action based on the current state.
- **Value Function**: The expected cumulative reward from each state, used to evaluate the long-term benefit of states.
- **Q-value (Action-Value Function)**: The expected cumulative reward of taking an action in a given state, used to evaluate the long-term benefit of actions.

2.2 Hyperparameters in Reinforcement Learning

Hyperparameters are the variables that can change how the model is trained and learns, but is not conducive to a physical parameter, like the speed or steering angle. Hyperparameters can significantly influence the performance of reinforcement learning models [1]. Some critical hyperparameters that were tested include:

- **Batch Size**: This parameter determines the number of training samples used in one iteration[5]. It affects the convergence speed and stability of the learning process.
- **Learning Rate**: This controls how much the model's parameters are adjusted in response to the estimated error each time the model weights are updated. A smaller learning rate can lead to better convergence but may require more training time.
- **Discount Factor (Gamma)**: This parameter determines the importance of future rewards. A higher gamma value prioritizes long-term rewards, while a lower gamma value focuses on short-term rewards.
- **E-Greedy-Value (Epsilon)**: In epsilon-greedy strategies, this parameter controls the balance between exploration (trying new actions) and exploitation (using known actions that yield high rewards).

- **Entropy**: This parameter controls how confident the model is in choosing an action for a given state. A low entropy rate is very confident in its decision, whereas a high entropy rate is uncertain in its decision.
- **Number of Episodes**: This parameter controls the amount of episodes per iteration trained. It encapsulates the amount of experience the model has. This can be changed based off usage for training.

2.3 AWS DeepRacer and Hyperparameter Tuning

The AWS DeepRacer platform offers a practical environment to study the effects of hyperparameter tuning in RL[6]. By varying hyperparameters such as batch size, learning rate, gamma, and epsilon, we can observe their impact on the performance metrics like completion rate, fastest lap time, percentage of clean laps, and average reward per episode. These experiments provide valuable insights into optimizing RL models for autonomous racing and similar applications.

3 EXPERIMENTS AND TRAINING

The AWS DeepRacer model was trained on the base track, 'reInvent 2019', with default settings. Three workers were used for each experiment in order to stay consistent and due to the limitation of our hardware. We used this as a control and then changed the specific hyperparameter value in a monotonic system. The default settings are described in Table 1.

Hyperparameter	Value
Batch Size	64
Beta Entropy	0.01
Discount Factor	0.99
E-Greedy Value	0.05
Epsilon Steps	10000
Exploration Type	Categorical
Loss Type	Huber
Learning Rate (lr)	0.0003
Num Episodes Between Training	20
Num Epochs	5
Stack Size	1
Termination Condition Avg Score	350.0
Termination Condition Max Episodes	1000
SAC Alpha	0.2

Table 1: Default Hyperparameters

Each of the experiments were run in 5, 10, and 20 iterations in order to show the total range of the model's training. This way, we can see how a model performs given a limited amount of time. This correlates to a runtime of about 30 minutes per 5 iterations, but is dependent on the specifics of the model being trained.

3.1 Evaluation of Training

The experiments were measured in 4 measures of improvement.

- **Completion Rate**: This is the rate at which the model is able to complete a lap. It takes the total completed laps divided by the amount of attempted laps.
- **Percentage of Clean laps**: The proportion of laps completed without going off-track.
- **Fastest Lap Time**: Shows the fastest time the model completed a lap out of all the iterations trained.
- **Average Reward per Episode**: The mean reward obtained per episode.

4 CONCLUSION

Based on our extensive experiments and data analysis, we conclude the following for each of the four hyperparameters studied:

4.1 Batch Size

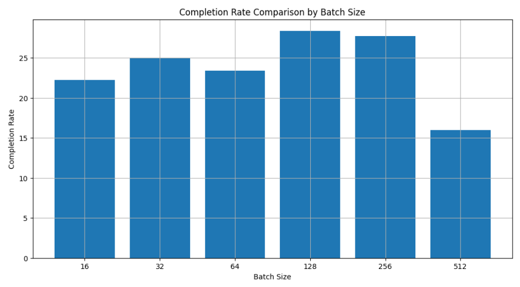

Figure 1: Completion Rate Comparison by Batch Size

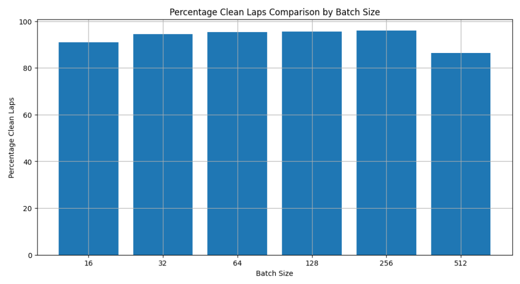

Figure 2: Percentage Clean Laps Comparison by Batch Size

Observations:

- Batch sizes of 128 and 256 showed the highest completion rates.
- Clean lap percentages were high across all batch sizes, with 32 and 256 performing slightly better.
- Fastest lap times showed no significant variation.
- Batch size 128 achieved the highest mean reward per episode.

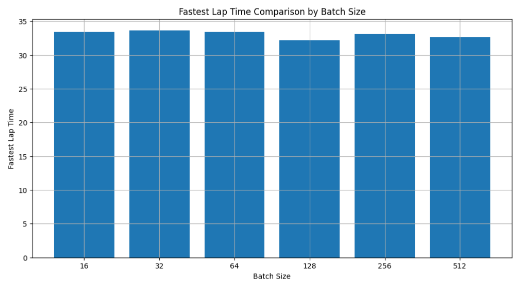

Figure 3: Fastest Lap Time Comparison by Batch Size

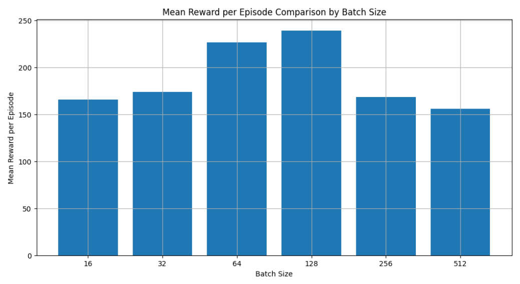

Figure 4: Mean Reward per Episode Comparison by Batch Size

Analysis:

- Medium batch sizes (128, 256) provide a balance between computational efficiency and learning stability.
- Smaller batch sizes might not provide enough data to stabilize learning.
- Larger batch sizes might cause the model to generalize too much, affecting performance.

Trends:

- Optimal batch size for AWS DeepRacer models lies between 128 and 256.
- Consistent high clean lap percentages indicate the reliability of models trained with these batch sizes.

4.2 Learning Rate

Observations:

- Learning rates of 0.0001 and 0.0005 showed the highest completion rates.
- These learning rates also maintained high percentages of clean laps.
- No significant variation in fastest lap times was observed.
- Learning rate 0.0001 achieved the highest mean reward per episode.

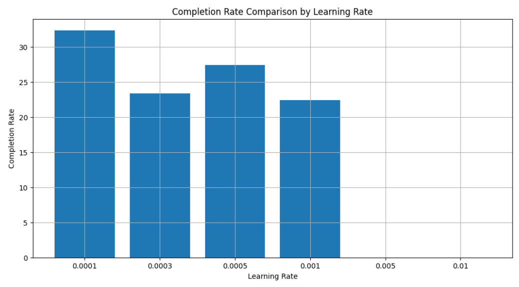

Figure 5: Completion Rate Comparison by Learning Rate

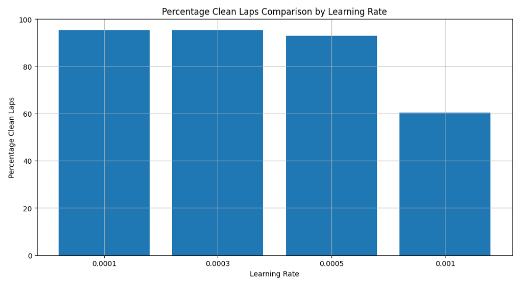

Figure 6: Percentage Clean Laps Comparison by Learning Rate

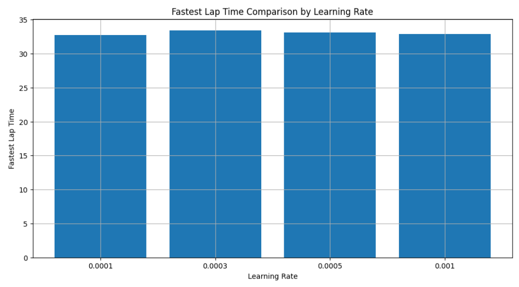

Figure 7: Fastest Lap Time Comparison by Learning Rate

Analysis:

- Lower learning rates provide more gradual and stable updates to the model, aiding better convergence.
- High learning rates cause the model to overshoot optimal solutions, leading to poor performance or complete failure.

Trends:

- Learning rates around 0.0001 to 0.0005 are optimal for AWS DeepRacer models.
- Moderate learning rates help maintain high completion rates and rewards.

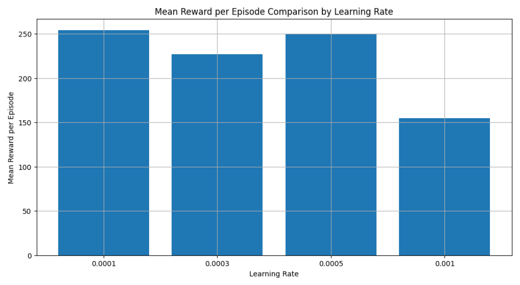

Figure 8: Mean Reward per Episode Comparison by Learning Rate

4.3 Discount Factor (Gamma)

Observations:

- Gamma values of 0.9 and 0.95 showed the highest completion rates.
- High percentages of clean laps were maintained across all gamma values, with 0.9 and 0.95 performing slightly better.
- Fastest lap times showed no significant variation.
- Gamma 0.95 achieved the highest mean reward per episode.

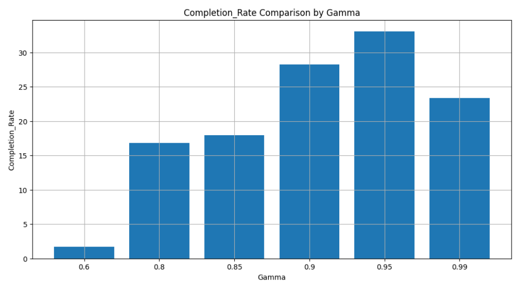

Figure 9: Completion Rate Comparison by Gamma

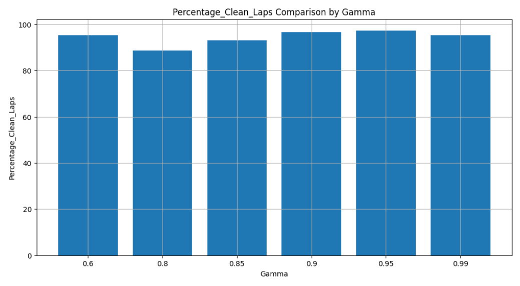

Figure 10: Percentage Clean Laps Comparison by Gamma

Analysis:

- The discount factor is crucial for the model to value future rewards appropriately.
- Higher gamma values encourage the model to consider long-term rewards, beneficial for complex tracks.
- Lower gamma values make the model short-sighted, focusing on immediate rewards and hence performing poorly on long-term tasks.

Trends:

- A gamma value close to 0.95 is optimal, ensuring a balance between immediate and future rewards.
- Models with high gamma values tend to perform better in terms of long-term planning and stability.

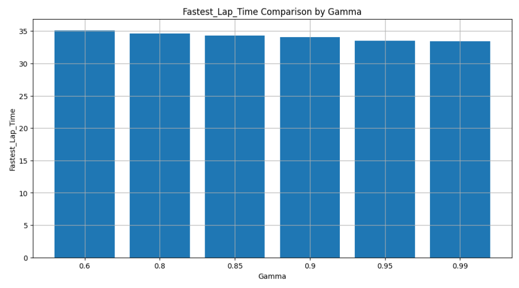

Figure 11: Fastest Lap Time Comparison by Gamma

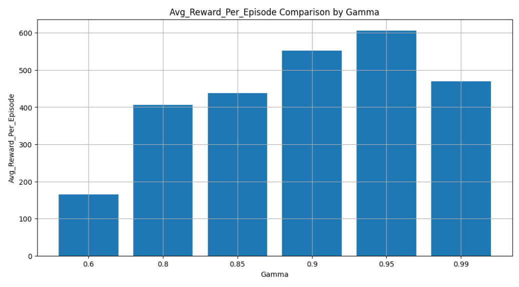

Figure 12: Mean Reward per Episode Comparison by Gamma

4.4 E-greedy-value (Epsilon)

Observations:

- Epsilon 0.1 showed the highest completion rate.
- All epsilon values maintained high percentages of clean laps.
- Fastest lap times showed no significant variation.
- Epsilon 0.1 achieved the highest mean reward per episode.

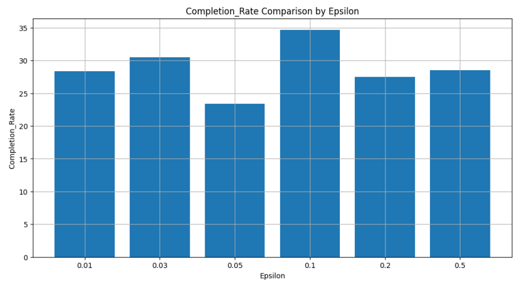

Figure 13: Completion Rate Comparison by Epsilon

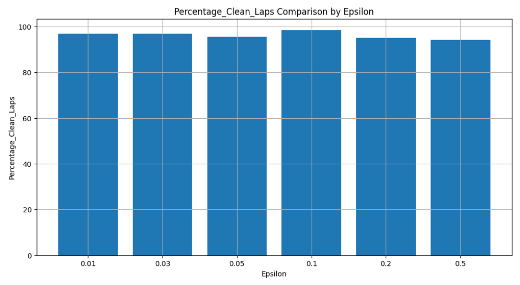

Figure 14: Percentage Clean Laps Comparison by Epsilon

Analysis:

- Epsilon controls the exploration-exploitation trade-off.
- A lower epsilon value means less exploration, while a higher value means more exploration.
- The default epsilon value ensures a good balance, allowing the model to explore new strategies while exploiting known good strategies.

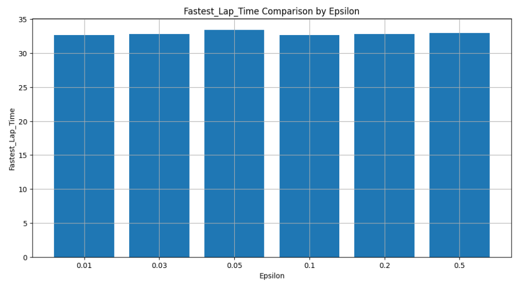

Figure 15: Fastest Lap Time Comparison by Epsilon

Trends:

- Epsilon values around 0.05 to 0.1 are optimal for AWS DeepRacer models.
- Ensuring balanced exploration and exploitation leads to better overall performance.

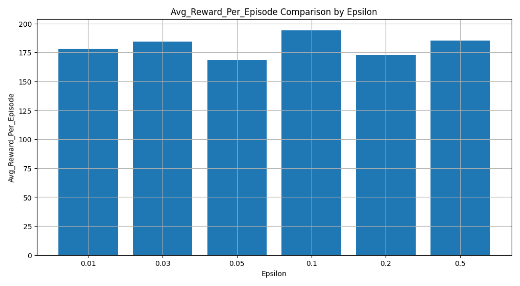

Figure 16: Mean Reward per Episode Comparison by Epsilon

4.5 Overall Trends and Analysis

- **Completion Rate**: Smaller batch sizes (32, 128), lower learning rates (0.0001, 0.0005), medium gamma values (0.9, 0.95), and medium epsilon values (0.1) tend to yield higher completion rates.
- **Percentage Clean Laps**: Most hyperparameters maintain high percentages of clean laps, indicating consistent training performance.
- **Fastest Lap Time**: Fastest lap times remain consistent across different hyperparameters, suggesting that lap time is less sensitive to these changes.
- **Mean Reward per Episode**: A balance in learning rate (0.0001), gamma (0.95), and epsilon (0.1) generally results in higher mean rewards per episode, indicating better learning outcomes.

4.6 Final Conclusion

Hyperparameter	Optimal Value
Batch Size	128
Learning Rate	0.0001
Gamma	0.95
Epsilon	0.1

Table 2: Optimal Hyperparameters for AWS DeepRacer

5 IMPLICATIONS AND FUTURE WORK

These findings provide valuable insights into hyperparameter tuning for reinforcement learning models using the AWS DeepRacer platform. They underscore the importance of selecting appropriate hyperparameters to optimize model performance in autonomous racing and similar applications.

For future work, we plan to modify the reward function while running the same experiments. Specifically, we will use a new non-default AWS DeepRacer center of line reward function. This will help us understand the impact of different reward functions on the performance of the RL models, thereby extending our current findings and providing a more comprehensive analysis of hyperparameter tuning in the context of autonomous racing.

ACKNOWLEDGMENTS

We would like to thank Texas State University and National Science Foundation. This material is based upon work supported by the National Science Foundation under Grant No. CNS-2149950. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] Maimoonah Ahmed, Abdelkader Ouda, and Mohamed Abusharkh. 2022. An Analysis of the Effects of Hyperparameters on the Performance of Simulated Autonomous Vehicles. In *2022 International Telecommunications Conference (ITC-Egypt)*. 1–5. https://doi.org/10.1109/ITC-Egypt55520.2022.9855682
[2] Jamir Leal Cota, José A. Tavares Rodríguez, Brandon García Alonso, and Carlos Vázquez Hurtado. 2022. Roadmap for development of skills in Artificial Intelligence by means of a Reinforcement Learning model using a DeepRacer autonomous vehicle. In *2022 IEEE Global Engineering Education Conference (EDUCON)*. 1355–1364. https://doi.org/10.1109/EDUCON52537.2022.9766659
[3] Lydia A. Garza-Coello, Marco Moreno Zubler, Axayacatl Nava Montiel, and Carlos Vazquez-Hurtado. 2023. AWS DeepRacer: A Way to Understand and Apply the

Reinforcement Learning Methods. In *2023 IEEE Global Engineering Education Conference (EDUCON)*. 1–3. https://doi.org/10.1109/EDUCON54358.2023.10125166

[4] Mukesh Ghimire. 2021. A Study of Deep Reinforcement Learning in Autonomous Racing Using DeepRacer Car. (2021).

[5] Sinan Koparan and Bahman Javadi. 2024. DeepRacer on Physical Track: Parameters Exploration and Performance Evaluation. arXiv:2406.03769 [cs.LG] https://arxiv.org/abs/2406.03769

[6] Junyao Li, Mohamed Abusharkh, and Yong Xu. 2022. DeepRacer Model Training for Autonomous Vehicles on AWS EC2. In *2022 International Telecommunications Conference (ITC-Egypt)*. 1–5. https://doi.org/10.1109/ITC-Egypt55520.2022.9855675

[7] Bohdan Petryshyn, Serhii Postupaiev, Soufiane Ben Bari, and Armantas Ostreika. 2024. Deep Reinforcement Learning for Autonomous Driving in Amazon Web Services DeepRacer. *Information* 15, 2 (2024), 113.