Human-in-the-Loop Machine Learning Project

Behaviour Cloning for OpenAI Gym CarRacing - Erencan Tatar

1 Introduction

Behavior cloning, a form of imitation learning, enables the replication of human-like decision-making in autonomous systems. This approach involves training models on demonstrations of human performance, capitalizing on the nuanced understanding humans have of complex tasks. Prior research in this domain has yielded significant insights, particularly in replicating intricate human behaviors in machine learning models. However, behavior cloning also presents challenges, particularly in its susceptibility to domain shift and variations in state spaces during testing. While not always suitable for dynamically changing environments, it offers considerable potential in fine-tuning and enhancing model performance under specific conditions. This study focuses on applying behavior cloning to the "CarRacing-v2" environment from OpenAI Gym (Brockman et al., 2016), a task known for its complexity and requirement for nuanced control, to explore the efficacy and limitations of this approach.

Research questions

> How does behavior cloning perform in the 'CarRacing-v2' environment from OpenAI Gym beyond training data?

2 Behavior cloning

Behavior cloning (BC) is the simplest of IL strategies, essentially reducing the problem to that of traditional supervised learning. BC assumes access to a data set:

$$D = \{(s_n, a_n)\}_{n=1}^{N}, \quad a_t \sim \pi_E(a|s_t) \qquad (1)$$

where π_E is the human expert's policy. These state-action pairs do not have to be sequential in time. BC then fits a policy $\pi_\theta(a|s)$ to this dataset using traditional supervised learning:

$$\hat{\theta} = \arg\min_{\theta \in \Theta} \sum_{n=1}^{N} \ell(a_n, \pi_\theta(\cdot|s_n)) \qquad (2)$$

Behavior cloning (BC) in machine learning faces several distinct limitations. Firstly, training a policy using BC is disconnected from the model's underlying Markov Decision Process (MDP), meaning the policy does not interact with states that it generates itself. Secondly, the success of BC heavily relies on comprehensive coverage of the state space. Lastly, even if the policy demonstrates good performance in supervised learning tasks, this does not automatically translate to effective performance under the MDP.

It has been theorized that even with minimal supervised learning errors, the policy can still exhibit significant errors in terms of reward optimization. Therefore, BC appears to be most effective when used for fine-tuning already high-performing policies.

In light of these considerations, our research aims to investigate the influence of training data volume on the agent's performance and the policy's ability to adapt to different domains. Specifically, we explore questions like how training data quantity affects the agent and the challenges of domain adaptation.

3 Related work

In exploring behavior cloning (BC) for autonomous driving, the research by (Codevilla et al., 2019). and the study on BC using Convolutional Neural Networks (CNNs) provide foundational insights. This work highlights the strengths of BC in complex driving scenarios and its potential scalability, but also underscores inherent challenges like dataset biases and overfitting. Their proposed benchmark for BC emphasizes its capability in executing maneuvers in novel environments.

Complementing this, the study on BCNet (Farag and Saleh, 2018), a seventeen-layer CNN designed for safe driving and smooth steering, demonstrates the effective use of deep learning in behavior cloning. This research shows promising results in

cloning driving behaviors through extensive training and simulations, using data from front-facing cameras and experienced drivers. Both studies collectively contribute to the understanding of BC's effectiveness and limitations, informing our approach in the "CarRacing-v2" environment. These works underscore the significance of comprehensive training data and robust architectures in achieving effective behavior cloning in autonomous systems.

Incorporating the human component in self-driving tasks is crucial for human-in-the-loop machine learning. This integration is essential for several reasons. Firstly, human judgment can provide valuable insights into complex scenarios where automated systems may not be fully reliable. Secondly, human intervention is crucial for safety, ensuring that decisions made by the vehicle are checked and balanced by human understanding and ethics. Lastly, human feedback can significantly improve the learning and adaptation of these systems, particularly in unpredictable and dynamic driving environments. Therefore, the human element not only enhances the reliability and safety of self-driving systems but also plays a vital role in their continuous learning and development.

Starting with the OpenAI Gym, a simpler environment, is beneficial for understanding and mastering the fundamentals required for real-world self-driving tasks. This approach allows for the development and testing of algorithms in a controlled, risk-free environment. Lessons learned here about behavior cloning, model training, and handling dynamic situations can then be applied to more complex, real-world scenarios. This step-by-step progression ensures a solid foundation is built before tackling the intricate challenges of actual self-driving technology.

4 Human data

We collect states and actions from the 2D top-down v2 car racing environment by the gym library [1], see figure 1. This approach allows for track generation based on specific seed values. The primary data for training the behavior cloning models includes 96x96x3 pixel images from the track, paired with corresponding human actions. However, the nature of this game makes it less suitable for learning with multiple annotators, due to the challenging control mechanics and the need for a strong grasp of the game. While it is possible to employ filtering based on a reward-per-step baseline to manage data quality, this process is time-consuming and requires additional resources. Therefore, relying on a single, expert annotator is deemed more practical and efficient for this specific application.

4.1 States

The environment state consits 96x96 pixels with the car starting at rest in the center of the road. The game settings allow for continuous In continuous setting there are 3 actions: steering (-1 is full left, +1 is full right), gas, and breaking. In a discrete setting there are 5 actions: do nothing, steer left, steer right, gas, brake. For all the experiments the default continuous state space was used.

Some indicators are shown at the bottom of the window along with the state RGB buffer. From left to right: true speed, four ABS sensors, steering wheel position, and gyroscope.

4.2 Metric

The evaluation of model performance in the 'CarRacing-v2' environment hinges on a nuanced rewards system. Each frame incurs a small penalty, while successfully navigating track tiles yields substantial points. This scoring framework, therefore, becomes a crucial indicator of the model's efficiency in mimicking human-like driving.

The reward is -0.1 every frame and $+1000/N$ for every track tile visited, where N is the total number of tiles visited in the track[2].

Training data is captured as 96x96x3 pixel representations of the track, paired with corresponding human actions. This approach simplifies the task into a supervised learning paradigm, enabling the use of standard accuracy metrics to compare the model's output with human decisions. Such a setup is instrumental in quantitatively assessing the model's ability to replicate complex driving behaviors under varying track conditions.

The setup simplifies the task into a supervised learning framework, where human actions serve as the ground-truth labels. This allows for direct comparison of the model's predictions against actual human decisions, using standard accuracy metrics to evaluate performance. This method is essential for assessing how closely the model's behavior aligns with human actions under the given conditions, on a single frame level.

[1] https://gymnasium.farama.org/environments/box2d/car_racing/

[2] For example, if you have finished in 732 frames, your reward is 1000 - 0.1*732 = 926.8 points.

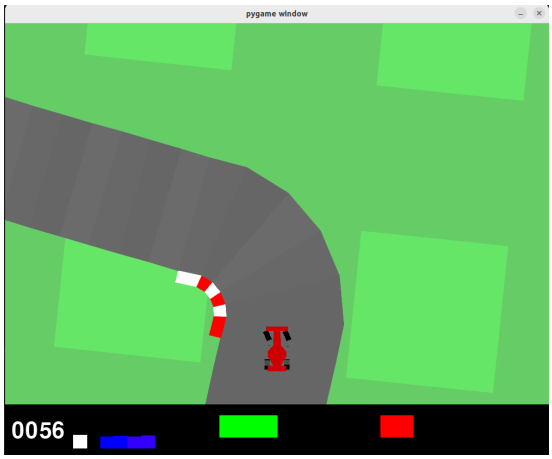

Figure 1: Original track

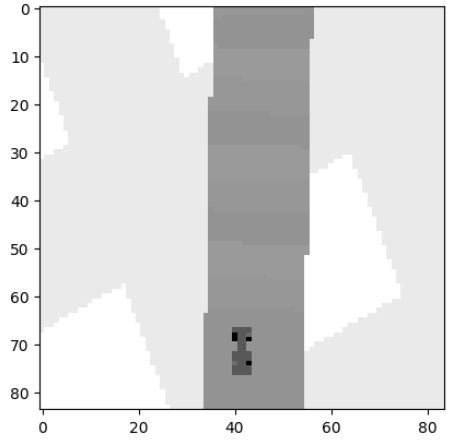

Figure 2: Pre-processed track

5 Method

"The method employed involves an initial step of data transformation. The raw state images from the OpenAI Gym environment are first pre-processed. This pre-processing includes converting the images to grayscale, adding padding, and cropping out irrelevant information such as speed and acceleration indicators. The final step in the data preparation process is normalizing the image pixel values to a range between 0 and 1, ensuring consistency and improved model performance.

5.1 Models

For the model architecture, we use a standard sequential design comprising alternating layers of Conv2d and BatchNorm, interspersed with non-linear activation functions. The architecture culminates in a linear layer that maps to the predefined action set. Detailed specifications and configurations of the model are provided in the Appendix 10.

5.2 Practise

A significant challenge arises from the model's inability to account for dynamic factors such as the speed and rotation of the race car. This limitation complicates the task, particularly in situations requiring nuanced control, such as corner navigation—a task that poses difficulty even for human players. To mitigate these challenges, the model's braking and acceleration parameters are carefully adjusted. This adjustment is crucial to avoid excessive acceleration in straight sections, which can lead to understeering and the consequent failure to execute turns effectively.

6 Results

To evaluate the effectiveness of behavior cloning in the "CarRacing-v2" environment, we conducted a series of experiments using human demonstration data. The primary focus was to assess the performance of the trained agent in replicating human-like driving behaviors and adapting to new track configurations.

6.1 Minimum dataset influence

The study investigated the impact of the volume of training data on the performance of the trained agent. We employed varying proportions of the demonstration data , specifically 10%, 20%, 50%, and 100%, to train the model and then evaluated its performance on a baseline track *(seed4)*.

The results, as summarized in table 1, indicate a clear trend: as the percentage of the training set used increases, there is a significant improvement in the maximum reward achieved by the agent in a single run. Starting from an inability to score any points with just 10% of the data, the model's performance enhances notably with more data, reaching a peak performance when the entire dataset is utilized. This trend underscores the importance of the quantity of training data in behavior cloning, particularly in complex environments like 'CarRacing-v2', where nuanced control and adaptability are crucial for success.

Percentage of training set	10%	20%	50%	100%
Max reward during 1 run	12	110	357	782

Table 1: Impact of variation in data. Used 5000 training instances for 100%

The table with varying percentages of the train-

ing set and corresponding max rewards during a single run clearly demonstrates the positive correlation between the amount of training data and the performance of the behavior cloning model.

Domain adaptation

The model receives a training annotation dataset of 5 laps around the same track (seed of 4) and is trained for 100 epochs, see figure 5 in appendix B. Figure 3 shows the resulting test of racing on tracks that either fall within the training data (track 4) or lie beyond the training data (different tracks and corners). Most test laps on different tracks fail earlier and never make the full track indicating that the model only learns corners and cases specific to the trained track and does not generalize well.

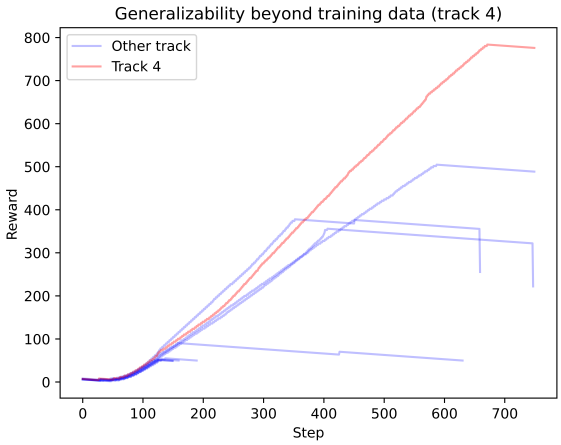

Figure 3: Reward over time. Trained on track 4 only.

To test whether B.C. works in general cases, we trained and tested on different tracks and plotted total reward over time in figure 4. Using the 15000 instances of all different tracks and thus different corners and more cases to learn from, we model performs nearly just as good on tracks it has never seen before.

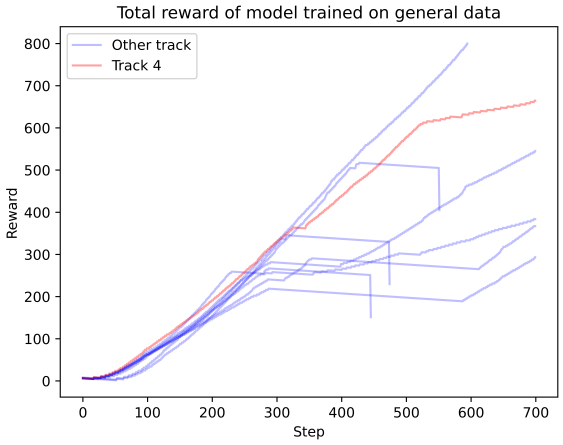

Figure 4: Reward over time. Trained on different tracks.

7 Conclusion

In conclusion, this study on behavior cloning in the "CarRacing-v2" environment from OpenAI Gym illuminates the intricacies of autonomous system training. The results underscore the critical role of human-in-the-loop interaction, enhancing model adaptability and performance. However, the experiments also reveal challenges, such as domain adaptation and managing dynamic scenarios. The varying effectiveness of the model based on training data quantity suggests a need for more robust training methodologies. These insights are invaluable for future research, especially in refining autonomous systems for real-world applications and improving their adaptability to new environments.

8 Discussion

It's evident that using *accuracy* as a metric for evaluating behavior cloning models in dynamic environments like "CarRacing-v2" is inadequate. Training improvements in frame-by-frame accuracy don't necessarily translate to enhanced performance on tracks, partly due to the human-like gaming strategy of relying on memory of previous frames. This can be seen in figure 6 where high accuracy does not directly correlate with mastery in different tracks. Additionally, the models tend to miss critical edge cases, like excessive acceleration on straights without adequate braking before turns, and the inability to distinguish forward from backward movement. The models' struggle to adapt to new tracks not seen during training underscores the necessity for more sophisticated training methods. Employing active learning, particularly focusing

on complex track segments, could significantly improve model performance and generalization.

9 Future work

Exploring the use of advanced multimodal networks like GPT-4V could offer significant improvements in behavior cloning models. These networks, with their enhanced visual and contextual understanding, might provide more nuanced and adaptable responses in complex environments. Additionally, incorporating metrics such as time taken to complete tracks could encourage more balanced strategies, moving beyond the 'slow but steady' approach. Further research could also delve into refining training techniques, exploring more diverse datasets, and enhancing the model's ability to handle unpredictable scenarios.

10 Code and sources

Most of the code is based on the github below, except for the experiments, evaluation and human annotation https://github.com/tayalmanan28/Behaviour_cloning/tree/main. Sources for other approaches: https://github.com/zalkikar/behaviorCloning_CarRacingv0/tree/main and https://notanymike.github.io/Solving-CarRacing/

Additionally, it should be highlighted that this paper has been thoroughly reviewed and refined using Chat-GPT to improve readability and clarity throughout all sections, which is particularly relevant given the focus of this course on human-in-the-loop systems.

References

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. 2016. Openai gym.

Felipe Codevilla, Eder Santana, Antonio M López, and Adrien Gaidon. 2019. Exploring the limitations of behavior cloning for autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9329–9338.

Wael Farag and Zakaria Saleh. 2018. Behavior cloning for autonomous driving using convolutional neural networks. In *2018 International Conference on Innovation and Intelligence for Informatics, Computing, and Technologies (3ICT)*, pages 1–7.

A Model

Layer	Filters	Kernel	Stride
Conv2d	1	32	8, 4
BatchNorm2d	32	-	-
ELU	-	-	-
Dropout2d	-	-	0.5
Conv2d	32	64	4, 2
BatchNorm2d	64	-	-
ELU	-	-	-
Dropout2d	-	-	0.5
Conv2d	64	64	3, 1
ELU	-	-	-
Flatten	-	-	-
BatchNorm1d	64 * 7 * 7	-	-
Dropout	-	-	-
Linear	64 * 7 * 7	120	-
ELU	-	-	-
BatchNorm1d	120	-	-
Dropout	-	-	-
Linear	120	Actionset	-

Table 2: Neural Network Architecture based of code from *tayalmanan28*

B Training logs

This is a section in the appendix.

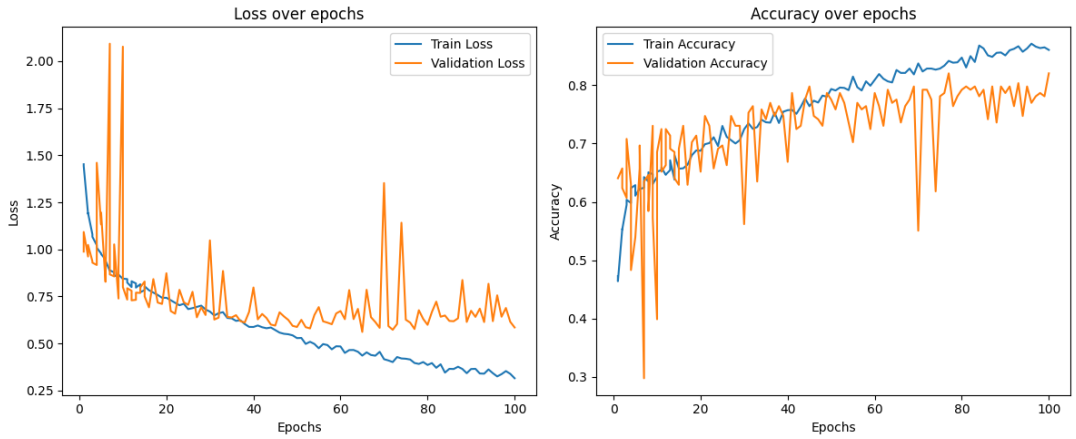

Figure 5: Training only on track 4

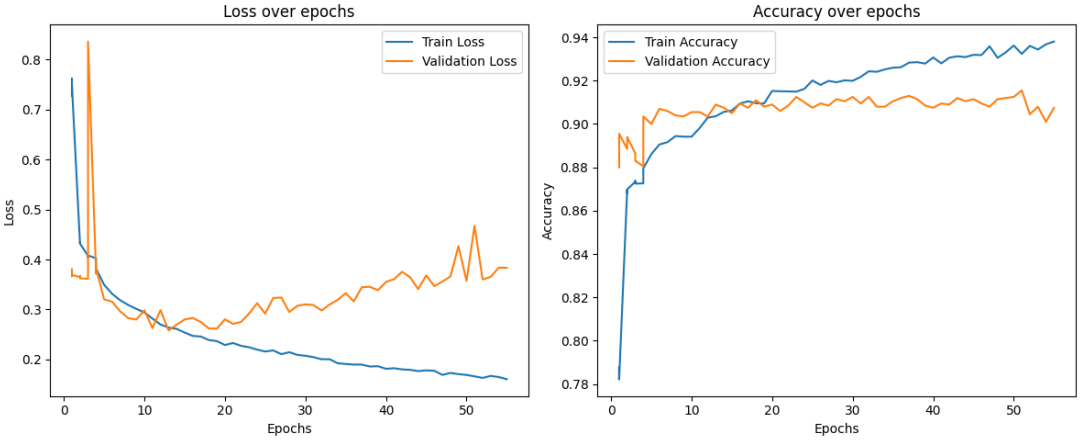

Figure 6: Training on bigger dataset with different tracks