Chapter 7 - Case Study: Driver Performance Difference and Model Difference Evaluation on Different Car-following Scenarios under Mixed Traffic Flow

Yang LIU
https://orcid.org/0009-0000-1416-0265
Intelligent Transportation Thrust
Hong Kong University of Science and Technology (Guangzhou)
Guangzhou, Guangdong, China

Dengbo HE
https://orcid.org/0000-0003-4359-4083
Intelligent Transportation Thrust
Robotics and Autonomous Systems Thrust
Hong Kong University of Science and Technology (Guangzhou)
Guangzhou, Guangdong, China

Xiaoyu ZHOU
https://orcid.org/0009-0000-2214-5647
Robotics and Autonomous Systems Thrust
Hong Kong University of Science and Technology (Guangzhou)
Guangzhou, Guangdong, China

Jiahao ZHANG
https://orcid.org/0009-0002-3833-3911
Robotics and Autonomous Systems Thrust
Hong Kong University of Science and Technology (Guangzhou)
Guangzhou, Guangdong, China

Abstract

In this chapter, we analyzed the difference in human driver behavior when following autonomous vehicles versus that when following human-driven vehicles. We also adopted commonly used car-following models to recover the car-following behaviors of human drivers and evaluated the performance of different car-following models. The car-following model code in this case study is open to download at GitHub: https://github.com/lovezoyaxu/car-following-model-in-mixtraffic.

7.1 Introduction

The high-level, SAE Level 4 or higher (SAE OnRoad Automated Vehicle Standards Committee, 2018), autonomous vehicles (AVs) running on the road has become a reality in many cities around the world. However, before the AVs saturate the market or designated lanes are built for AVs, the AVs will have to share the road with human-driven vehicles (HDVs),

leading to mixed traffic. At the same time, it has been widely acknowledged that human operators may adapt to the technologies and adopt different strategies to work with them (Janssen et al., 2019; Kaber et al., 2009). Hence, to better model and manage the mixed traffic, we will need to model the human drivers' behaviors appropriately. Given that the rear-end crashes dominate the AV-involved crashes (Liu et al., 2021), this case study will mainly focus on the car-following behaviors of human drivers in mixed traffic.

So far, most of the studies on human drivers' behavior changes in mixed traffic were conducted via traffic/numerical simulations (Zheng et al., 2020) or field experiments (Mahdinia et al., 2021). A few research has been conducted to model human drivers' behavior changes based on on-road data, due to the lack of real-world empirical data as a result of the low AV market penetration rate at this stage. This case study will demonstrate the application of real-world public AV driving datasets to model human drivers' car-following behaviors in mixed traffic and analyze the differences in human drivers' car-following behavior when the leading vehicle is AV versus HDVs.

The driving models used in this case include:

- Traditional car-following models, including the Intelligent Driver Model (IDM), Gazis–Herman–Rothery (GHR) car-following model (Al-Jameel, 2009), Gipps car-following model (Rakha et al., 2007), Wiedemann car-following model (Abbas & Medina, n.d.), General Force (GF) model, and the Full Velocity Difference (FVD) model (Gong et al., 2008);
- Deep neural network models, including Deep Neural Network (DNN) and Long Short-Term Memory (LSTM);
- Car-following models based on deep reinforcement learning (DRL), including Proximal Policy Optimization (PPO) and Generative Adversarial Imitation Learning (GAIL).
  All models were trained and calibrated based on the Waymo open dataset.

7.2 Data Description

A. Waymo open dataset

The data used in this case is retrieved from the Waymo Open Dataset released by Waymo LLC. Waymo has been testing their AV fleet on public roads across various U.S. cities, accumulating over 32 million kilometres (Ettinger et al., 2021). The Waymo AVs are equipped with an array of sensors, including radars, lidars, and cameras, which can collect high-quality trajectory data of the vehicles and surrounding traffic at a frequency of 10 Hz. All the vehicles used in these tests can be categorized as SAE Level-4 or higher and have a uniform appearance (Di & Shi, 2021).

The Waymo Open Dataset is constituted of two parts: the perception and motion parts. In August 2019, Waymo released the perception part, including 1,000 20-second segments of well-synchronized and calibrated high-resolution lidar and camera data in urban and suburban environments (Sun et al., 2020). The motion portion of the dataset was released in March 2021, including 103,354 20-sec long segments, totaling 574 hours of driving data. This data covers

1,750 kilometers of roadways across the U.S. Both perception and motion parts of the Waymo datasets were used in this case study. Interesting readers can find more information about the Waymo Open Dataset from the link: https://waymo.com/open/.

B. Car-Following Periods Extraction

In this case study, considering the sample size, the timespan of each segment, and the sensor detection range, the following criteria were adopted to extract qualified car-following events:

- The leading and following vehicles should be in the same lane on a straight segment;
- The lane changes and turns by either the leading or following vehicle were excluded;
- The space between the leading and following vehicles should be less than 85m;
- The following vehicle's velocity should be greater than 1m/s;
- The duration of the car-following event should be at least 10 seconds.

In this case, 171 HDV-following-AV events and 523 HDV-following-HDV events on highwa ys were extracted from the Waymo Open Dataset. In the following analysis, 80% of the event s were randomly selected as the training set of the model, and the remaining 20% were used f or testing. This dataset is publicly available on GitHub: https://github.com/lovezoyaxu/car-fol lowing-model-in-mixtraffic.

7.3. Comparison between HDV-following-HDV and HDV-following-AV events

This section will reveal the difference in driver performance between HDV-following-HDV and HDV-following-AV events. In this case study, we mainly focus on two metrics that can quantify traffic stability and safety, i.e., driving volatility, which reflects the comfort and fuel economy of the vehicle through the acceleration and deceleration fluctuations of the vehicle (Chai et al., 2022; Rios-Torres et al., 2019) and time headway, which can quantify the safety in car following events (Naito & Nagatani, 2012).

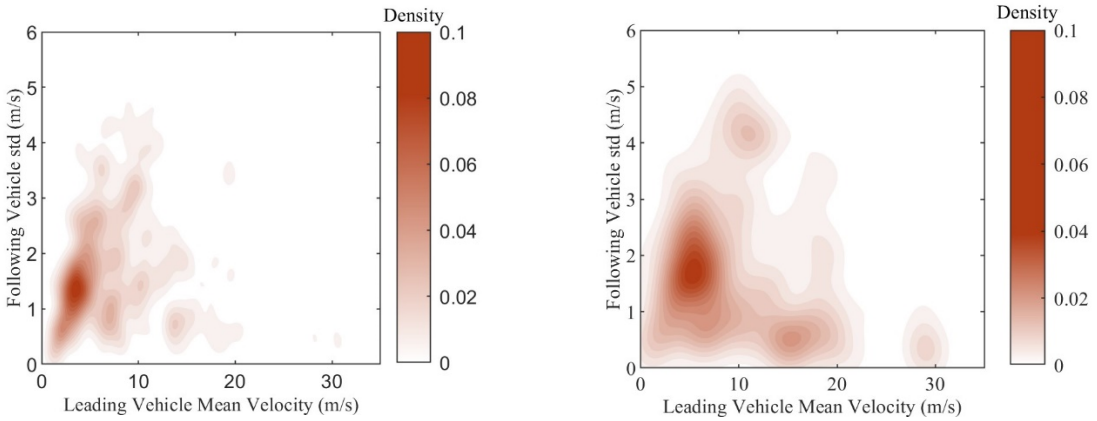
A. Driving volatility

Driving aggression can be represented by driving volatility, which can be calculated from vehicle speed, acceleration, and deceleration data. Previous research has shown that higher driving volatility is associated with higher driver instability, leading to higher crash risk, more energy consumption, and more greenhouse gas emissions (Kamrani et al., 2018) Driving volatility can be quantified by the standard deviation of the vehicle speed, as shown in equation 7.3.1.

$$Std = \sqrt{\frac{\sum_{i=1}^{n}(v_i-\bar{v})^2}{n-1}} \tag{7.3.1}$$

where v_i is the velocity value of observation i, $\bar{v}$ is the mean of observations and n is the sample size.

Fig. 7.3.1 shows the density of the driving volatility when the HDV followed HDV versus AVs. Here, the density is calculated as the number of events that a volatility value fell into a specific range out of all car-following events in that category (i.e., either HDV-following-HDV or HDV-following-AV). In general, when a human driver follows HDVs, the distribution of driving volatility mostly concentrated in the range of 1.2~1.6 m/s; while when following AVs, the distribution of driving volatility mostly concentrated in the range of 1.5~2 m/s. It is likely that drivers have different perceptions of AVs and thus exhibit different behaviors when following AVs (Mahdinia et al., 2021) .



(a) HDV-following-HDV (b) HDV-following-AV

Figure. 7.3.1 Velocity standard deviation distributions of following vehicles

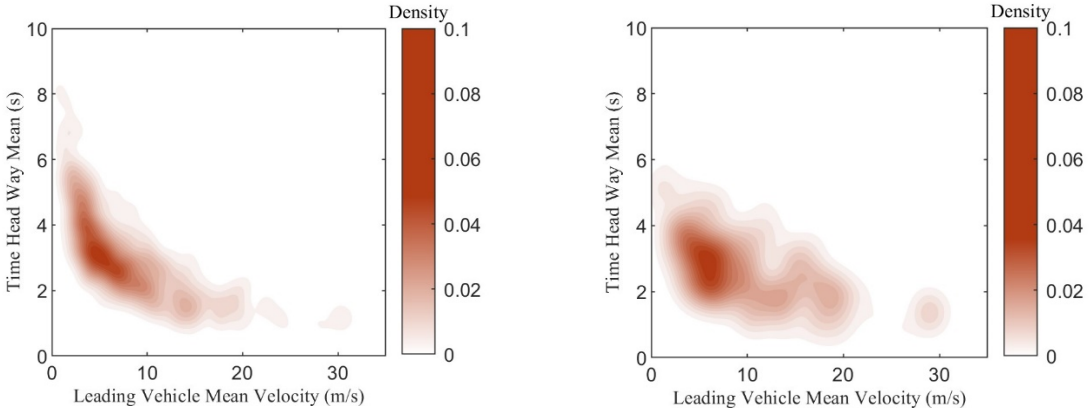
B. Driving Safety

Time headway is defined as the time difference between two consecutive vehicles passing through the same point. Time headway is calculated as the space headway between the leading and following vehicles divided by the following vehicle's velocity, as shown in equation 7.3.2.

$$h_w(t) = \frac{X_L(t) - X_F(t)}{V_F(t)} \tag{7.3.2}$$

where $X_L(t)$ and $X_F(t)$ are the positions of the leading vehicle and following vehicle at time t, and $V_F(t)$ is the velocity of the following vehicle at time t.

The downward tendency can be observed in both subplots of Fig. 7.3.2, suggesting that time headways decrease with the increments of average leading vehicle velocity. Again, it can be observed that when following AVs, the behaviors of HDVs are more diverse compared to that when following HDVs.



(a) HDV-following-HDV (b) HDV-following-AV

Figure. 7.3.2 Time headway distributions

7.4. Car-following Model Introduction

In the car-following event, the state of the vehicle is defined as 1) the speed of the self-vehicle $V_n(t)$, 2) the relative speed between the self-vehicle and the lead vehicle $\Delta V_{n-1,n}(t)$, and 3) the distance between the self-vehicle and the lead vehicle $S_{n-1,n}(t)$. According to Newton's theory, the relationship between them is shown in the following three equations. The action that the following vehicle needs to take at time step t is the longitudinal acceleration, denoted as $a_n(t)$.

$$V_n(t+1) = V_n(t) + \Delta T \times a_n(t) \tag{7.4.1}$$

$$\Delta V_{n-1,n}(t+1) = V_n(t+1) - V_{n-1}(t+1) \tag{7.4.2}$$

$$S_{n-1,n}(t+1) = S_{n-1,n}(t) - \Delta T \times \frac{\Delta V_{n-1,n}(t) + \Delta V_{n-1,n}(t+1)}{2} \tag{7.4.3}$$

Where ΔT is the simulation time interval, which is the data collection interval and was set to be $0.1s$ in this study; and $V_{n-1}(t+1)$ is the speed of the leading vehicle, which is set to be known over time.

The training process of one car-following event is set as an episode in this study. When a traffic crash happens (i.e., $S_{n-1,n}(t+1) \leq 0$) or the simulation reaches the maximum time step (i.e., the length of an event), all states will be re-initialized using the next car-following event data. Next, we will introduce several commonly used car-following models and illustrate how they can be applied to mixed traffic research.

A. Intelligent Driver Model

Microscopic car-following models have long been used to simulate individual driver behaviour in the presence of preceding vehicles, of which, the Intelligent Driver Model (IDM) is one of the most commonly used (Derbel et al., 2013):

$$a_n(t) = a_{max}^{(n)} \left(1 - \left(\frac{V_n(t)}{\widetilde{V_n}(t)}\right)^{\beta} - \left(\frac{\widetilde{S_n}(t)}{S_n(t)}\right)^2\right) \tag{7.4.4}$$

$$\widetilde{S_n}(t) = S_{jam}^{(n)} + max\left(0, V_n(t)\widetilde{T_n}(t) + \frac{V_n(t)\Delta V_n(t)}{2\sqrt{a_{max}^{(n)} a_{comfort}^{(n)}}}\right) \quad (7.4.5)$$

In the above equations, $a_{max}^{(n)}$ represent the maximum acceleration/deceleration of the following vehicle, $\widetilde{V_n}(t)$ is the desired speed, $S_{jam}^{(n)}$ is the minimum spacing at standstill, $\widetilde{T_n}(t)$ is the desired time headway, $a_{comfort}^{(n)}$ is the comfortable deceleration, and β is the parameter that controls the degree of nonlinearity of acceleration behavior when the vehicle speed approaches the desired speed. We set it as 4 in this research. By inserting the known state of the car-following trajectory into the above equations, we can obtain the desired action $a_n(t)$.

B. Deep Neural Network

Deep Neural Network (DNN) [4] is a machine learning model composed of multiple neural network layers. It typically consists of an input layer, multiple hidden layers, and an output layer, with each hidden layer containing multiple neurons. Each neuron is connected to all neurons in the previous layer, transmitting information through connections with associated weights. The structure of DNN is shown in Fig. 7.4.1.

The core idea of DNN is to perform tasks by learning patterns and features from data. During training, DNN automatically adjusts connection weights to minimize the error between predicted results and actual labels. This is usually achieved through optimization algorithms like gradient descent, where weights are adjusted based on the error gradient, gradually converging the model towards the optimal solution. By inserting the known state of the car-following trajectory into the DNN, we can get the desired action $a_n(t)$. In this case study, we used 3 layers in the neural networks.

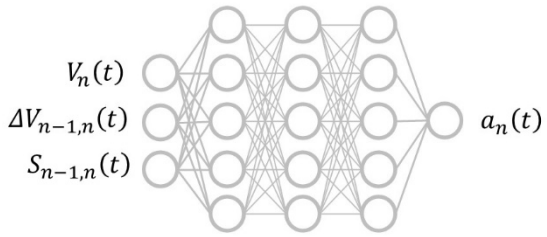


Figure. 7.4.1 the structure of DNN

C. Long Short-Term Memory

Long Short-Term Memory (LSTM) [5] is utilized to model the dynamic relationships between vehicles and traffic flow characteristics, enabling intelligent car-following strategies. By learning and predicting time-series data such as relative speeds, distances, and accelerations between vehicles, LSTM provides a deeper understanding, aiding the autonomous driving system in accurately determining when to accelerate, decelerate, or maintain distance for safe and efficient car-following. The objective function of the LSTM model is described as follows:

$$C(W,B) = \frac{(S_{n-1,n}(t) - S^{obs}_{n-1,n}(t))^2}{(S^{obs}_{n-1,n}(t))^2} \quad (8.4.6)$$

Where $S^{obs}_{n-1,n}(t)$ is the observed spacing at time step t, and W and B represent the weights and biases in the LSTM model. The LSTM model gradually minimizes the objective function by backpropagating a small update through time in the direction of optimizing the weights and biases. In this case study, we used a one-layer LSTM that considers 10 historical steps in the data.

D. Proximal Policy Optimization

To explain the Proximal Policy Optimization (PPO) (Wei et al., 2019), we first introduce the Actor-Critic algorithm (ATC) (Xu et al., 2021), given that the PPO algorithm is evolved based on the ATC algorithm. First, we assume that the target strategy π_θ is a random strategy and is differentiable everywhere, where θ is the corresponding parameter. We can use a linear model or a neural network model to model such a strategy function, input a state and output a probability distribution of an action. Our goal is to find an optimal strategy and maximize the expected return of that strategy in the environment. We define the objective function of strategy learning as

$$J(\theta) = E_{s_0}[V^{\pi_\theta}(S_0)] \quad (7.4.7)$$

Where S_0 represents the initial state. Now that we have the objective function, we take the derivative of the objective function with respect to the strategy, and once we get the derivative, we can use the gradient ascent method to maximize this objective function and get the optimal strategy.

$$\nabla_\theta J(\theta) = E_{\pi_\theta}\left[\sum_{t=0}^{T} \psi_t \nabla_\theta \, log_{\pi_\theta}(a_t \mid s_t)\right] \quad (7.4.8)$$

ATC algorithm is simple and intuitive, but it can be unstable during training in practical applications. This is because when the strategy network gradient updates the parameters, it is likely that the strategy will suddenly deteriorate significantly due to the long step size, thus affecting the training effect. To solve this problem, researchers added trust regions to ensure that each policy gradient update resulted in a performance improvement. Mathematically, we try to use the current θ to find a better parameter θ', so that $J(\theta') \geq J(\theta)$.

$$J(\theta') - J(\theta) = \frac{1}{1-\gamma} E_{s\sim v^{\pi_{\theta'}}} E_{a\sim\pi_{\theta'}(\cdot|s)}[A^{\pi_\theta}(s,a)] \quad (7.4.9)$$

The above equation is complicated to solve. Therefore, the state access distribution is processed accordingly. In particular, we ignore the change in the state access distribution between the two policies and directly adopt the state distribution of the old policy π_θ. The improved equation is as follows:

$$L_\theta(\theta') = J(\theta) + \frac{1}{1-\gamma} E_{s\sim v^{\pi_\theta}} E_{a\sim\pi_\theta(\cdot|s)}\left[\frac{\pi_{\theta'}(a|s)}{\pi_\theta(a|s)} A^{\pi_\theta}(s,a)\right] \quad (7.4.10)$$

The objective function of the above equation is restricted to ensure that the gap between the new and old parameters is not too large, and the resulting equation is the PPO update strategy.

$$argmax_\theta E_{s\sim v^{\pi_\theta}} E_{a\sim\pi_\theta(\cdot|s)}\left[min\left(\frac{\pi_\theta(a|s)}{\pi_k(a|s)}A^{\pi_\theta}(s,a), clip\left(\frac{\pi_\theta(a|s)}{\pi_k(a|s)}, 1-\varepsilon, 1+\varepsilon\right)A^{\pi_{\theta_k}}(s,a)\right)\right] \quad (7.4.11)$$

In this case, we used the reward function to facilitate human-like driving by minimizing the disparity between simulated and observed behavior values at each timestep. The reward equation is shown below.

$$r(t) = -\log\left(\left|\frac{S_{sim}(t)-S_{obs}(t)}{S_{obs}(t)}\right|\right) \quad (7.4.12)$$

Where $S_{sim}(t)$ is the spacing at time t given by the algorithm, and $S_{obs}(t)$ is the real spacing. When using PPO, put the state of the car-following trajectory into policy network π_θ and use equation 4.11 and equation 7.4.12 to training the policy network. The optimum policy network will give the desired $a_n(t)$ and its corresponding probability at each state.

E. Generative Adversarial Imitation Learning

The Generative Adversarial Imitation Learning (GAIL) (Ho & Ermon, 2016) is based on the Generative Adversarial Network (GAN) (Creswell et al., 2018). GAN consists of two main components: a generator and a discriminator. These two networks compete with each other through adversarial learning to improve the performance of the generator model.

The goal of a generator is to generate realistic data samples from random noise, such as images, audio, or text. It takes a random vector or noise as input and attempts to map it into a space similar to the real data sample. In contrast, the discriminator is used to distinguish between fake samples generated by the generator and real data samples. It takes samples from the generator as well as samples from the real data set and tries to tell them apart.

The main difference between the GAN and the GAIL is the generator. The generator of GAIL consists of an agent that can generate actions based on state. A typical structure of GAIL is shown in Fig. 7.4.2. In our case study, the generator uses the PPO algorithm improved by the policy gradient.

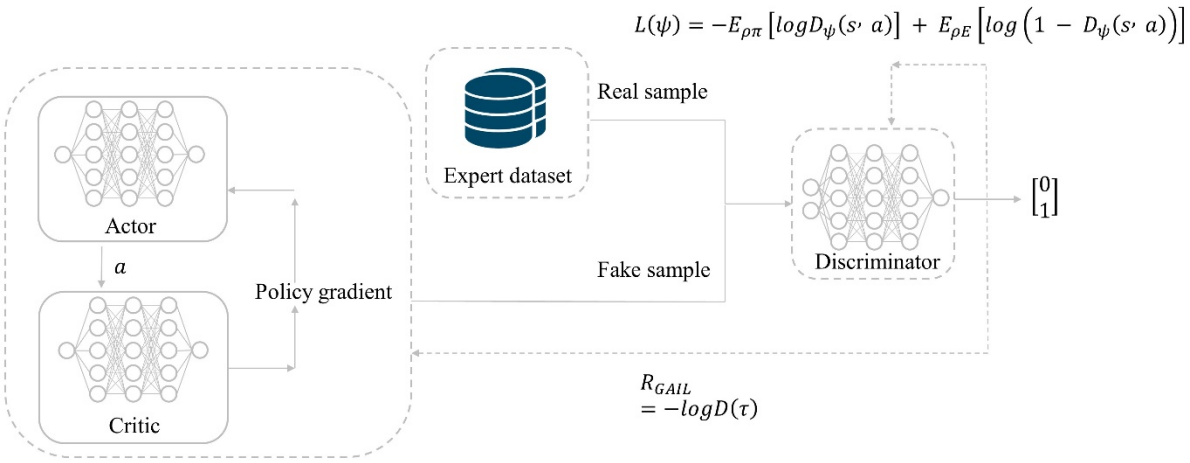


Figure. 7.4.2. The structure of GAIL

When using GAIL, firstly, an expert trajectory, which is the recorded real-world trajectory from the Waymo dataset, was randomly selected, and the state of the trajectory is initialized. Then, the initial state and leading vehicle data are input into the generator, and the PPO algorithm generates a trajectory based on this data. The discriminator optimizes itself by discriminating between the generated trajectory and the expert trajectory. The pseudo-reward generated by the discriminator will help optimize the PPO algorithm. The loss function of the discriminator is as follows:

$$L(\psi) = -E_{\rho\pi}\left[logD_{\psi}(s,a)\right] + E_{\rho E}\left[log\left(1 - D_{\psi}(s,a)\right)\right] \tag{7.4.13}$$

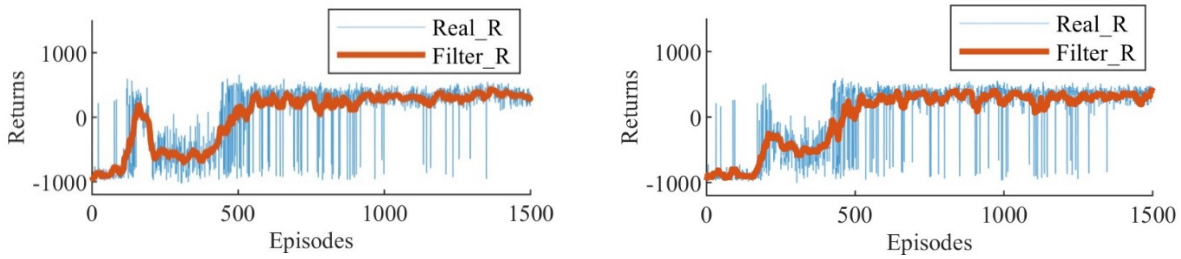
In equation 4.13, D_{ψ} represent discriminator network; $E_{\rho E}$ and $E_{\rho\pi}$ represent are the expectations for state-action pairs of expert policy and generator policy, respectively. The discriminator tries to push the output of the expert data close to 0 and the output of the mimic policy close to 1. Thus, when minimizing the loss function, the discriminator attempts to distinguish between data generated by expert strategies and data generated by generator strategies. At the same time, the generator tries to generate data that "cheats" the discriminator, thus approximating the expert strategy. The equation that the discriminator provides for the generator of pseudo-rewards is:

$$R_{GAIL} = -logD(s,a) \tag{7.4.14}$$

When the discriminator believes that the trajectory generated by PPO is close to the expert trajectory, the pseudo-reward will become large, encouraging the PPO generation to generate the trajectory in this way. Otherwise, the pseudo-reward will become small, inhibiting the PPO to generate the trajectory in such a way.

The performance of the GAIL can be expressed by equation 7.4.12. It should be noted that this reward is different from the pseudo-reward generated by the discriminator. In the GAIL algorithm, PPO are only updated by the pseudo-reward of equation 7.4.14. equation 7.4.12 only reflects the quality of GAIL training results. However, in the PPO algorithm, equation 7.4.12 reward is used to update the PPO algorithm. You may compare the open-source code provided in this case study to distinguish the difference between these two algorithms.

Fig. 7.4.3 shows how the performance of GAIL model changes with training time for different car following types (i.e., HDV-following-HDV versus HDV-following-AV). In general, the model becomes stable after 800 episodes.



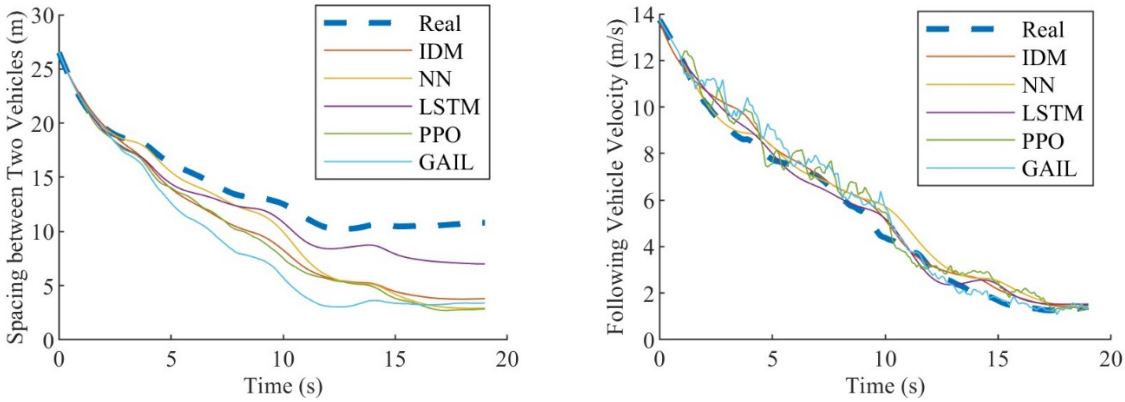
(a) HDV-following-HDV (b) HDV-following-AV

Figure. 7.4.3 GAIL model training return change trend

7.5. Car-following Model Performance

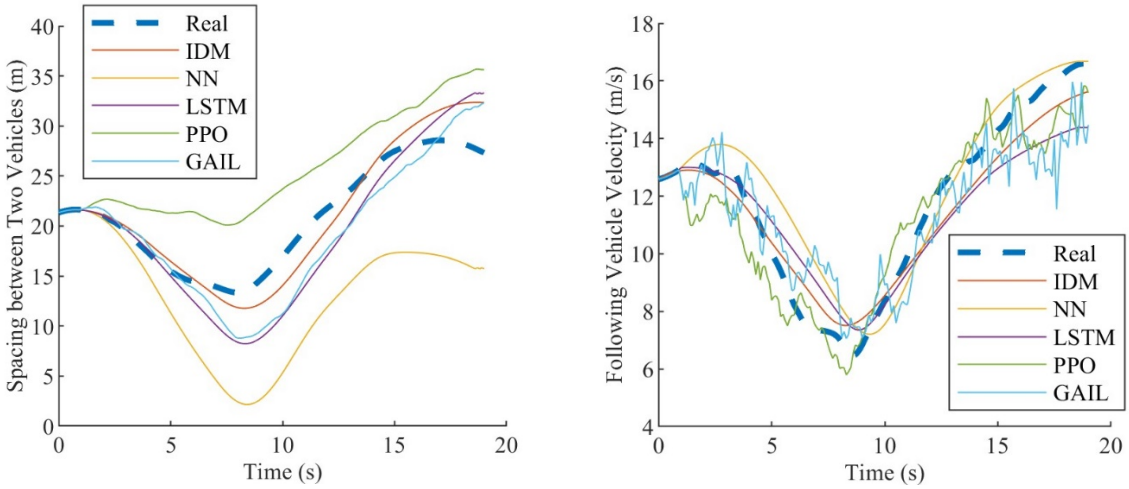
To give readers an intuitive sense of the differences among different models, this section compares the spacing between following vehicle and leading vehicle and the speed of the following vehicle velocity generated by different algorithms. Fig. 7.5.1 shows the HDV-following-HDV result and Fig. 7.5.2 shows the HDV-following-AV results.

It can be observed that compared with DRL algorithms (i.e., PPO and GAIL), IDM and deep neural network car following algorithms (i.e., DNN and LSTM) are smoother. DRLs led to frequent fluctuations in speed. In addition, the error between real trajectory and LSTM algorithm is the smallest among all algorithms in terms of spacing and velocity.



(a) Spacing (b) Velocity

Figure. 7.5.1 Car-following model performance in HDV-following-HDV scenarios



(a) Spacing (b) Velocity

Figure. 7.5.2 Car-following model performance in HDV-following-AV scenarios

Tables 7.5.1 to 7.5.2 provide quantitative results regarding the performance of different car following models on the test dataset. In the tables, MSE means Squared Error. MSE (spacing) and MSE (velocity) are the average of the square of the spacing and velocity errors between each generated trajectory and the expert trajectory, respectively. The jerk metric evaluates the smoothness of vehicle acceleration or deceleration. It measures the rate of change of acceleration and reflects the comfort level experienced by the passengers of the ego vehicle. A lower jerk value indicates a smoother and more comfortable driving experience. TTC is Time-To-Collision. TTC is a commonly used surrogate safety measure, which was initially introduced in (Vogel, 2003) and defined as "*the time that remains until a crash between two vehicles would have occurred if the crash course and speed difference are maintained*." as shown below:

$$TTC(t) = \frac{X_L(t) - X_F(t) - L}{V_F(t) - V_L(t)} \tag{7.4.12}$$

where $X_L(t)$ and $X_F(t)$ are the positions of centres of mass of the leading and following vehicles at time t; L is the length of the vehicles; and $V_F(t)$ and $V_L(t)$ are the velocity of the leading and following vehicles at time t.

Table 7.5.1 Model performance comparison in HDV-following-HDV scenarios

Model / Test standard	IDM	DNN	LSTM	PPO	GAIL
Test of MSE (spacing)	37.31	54.66	**26.76**	48.49	46.81
Test of MSE (velocity)	1.25	1.48	**0.99**	1.17	3.78
Jerk	0.20	0.28	**0.27**	8.91	19.81
TTC	**34.43**	14.35	5.26	23.39	17.12

Table 7.5.2 Model performance comparison in HDV-following-AV scenarios

Model / Test standard	IDM	DNN	LSTM	PPO	GAIL
Test of MSE (spacing)	32.06	21.41	**20.47**	23.85	41.43
Test of MSE (velocity)	45.52	41.48	**38.75**	45.50	49.04
Jerk	12.31	**6.48**	9.71	7.73	20.76
TTC	114.94	116.56	119.66	124.52	**123.37**

The descriptive statistical results in Tables 7.5.1 to 7.5.2 are the average values without outlier. As can be seen from the two tables, the LSTM has the best performance when recovering the car-following trajectory. DRL algorithms, especially the GAIL algorithm, led to the largest jerk values, which can impair driving comfort and fuel economy. Compared with following HDVs, TTC is significantly improved when following AVs. This is because, in the Waymo dataset, the spacing between vehicles is larger when following AVs versus that when following HDVs.

7.6. Conclusion

This case study analyses the performance differences of human drivers when following different types of vehicles by extracting car following events from Waymo dataset. In addition, we demonstrated the application of five different algorithms in recovering car-following behaviours in mixed traffic. The results show that LSTM algorithm has the best performance in terms of approaching the real trajectory and DRL algorithms tend to lead to large jerk. The readers can refer to the sample codes published in GitHub for more details regarding the five algorithms. Readers with more powerful computer equipment can adjust the parameters in the code to get better results.

Reference List

Al-Jameel, H. a. E. (2009). Examining and improving the limitations of Gazis–Herman –Rothery car following model. https://salford-repository.worktribe.com/output/1459924

Chai, C., Shi, X., Zhou, Z., Zeng, X., Yin, W., & Islam, Md. M. (2022). Driving Style Recognition Based on Naturalistic Driving: Volatilities, Decision-Making, and Safety Performances. In A. Riener, M. Jeon, & I. Alvarez (Eds.), User Experience Design in the Era of Automated Driving (pp. 359–394). Springer International Publishing. https://doi.org/10.1007/978-3-030-77726-5_14

Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B., & Bharath, A. A. (2018). Generative Adversarial Networks: An Overview. IEEE Signal Processing Magazine, 35(1), 53–65. https://doi.org/10.1109/MSP.2017.2765202

Derbel, O., Peter, T., Zebiri, H., Mourllion, B., & Basset, M. (2013). Modified Intelligent Driver Model for driver safety and traffic stability improvement. IFAC Proceedings Volumes, 46(21), 744–749. https://doi.org/10.3182/20130904-4-JP-2042.00132

Di, X., & Shi, R. (2021). A survey on autonomous vehicle control in the era of mixed-autonomy: From physics-based to AI-guided driving policy learning. Transportation Researc

h Part C: Emerging Technologies, 125, 103008. https://doi.org/10.1016/j.trc.2021.103008

Ettinger, S., Cheng, S., Caine, B., Liu, C., Zhao, H., Pradhan, S., Chai, Y., Sapp, B., Qi, C. R., Zhou, Y., Yang, Z., Chouard, A., Sun, P., Ngiam, J., Vasudevan, V., McCauley, A., Shlens, J., & Anguelov, D. (2021). Large Scale Interactive Motion Forecasting for Autonomous Driving: The Waymo Open Motion Dataset. 9710–9719. https://openaccess.thecvf.com/content/ICCV2021/html/Ettinger_Large_Scale_Interactive_Motion_Forecasting_for_Autonomous_Driving_The_Waymo_ICCV_2021_paper.html

Gong, H., Liu, H., & Wang, B.-H. (2008). An asymmetric full velocity difference car-following model. Physica A: Statistical Mechanics and Its Applications, 387(11), 2595–2602. https://doi.org/10.1016/j.physa.2008.01.038

Ho, J., & Ermon, S. (2016). Generative Adversarial Imitation Learning. Advances in Neural Information Processing Systems, 29. https://proceedings.neurips.cc/paper_files/paper/2016/hash/cc7e2b878868cbae992d1fb743995d8f-Abstract.html

Janssen, C. P., Donker, S. F., Brumby, D. P., & Kun, A. L. (2019). History and future of human-automation interaction. International Journal of Human-Computer Studies, 131, 99–107. https://doi.org/10.1016/j.ijhcs.2019.05.006

Kaber, D. B., Stoll, N., Thurow, K., Green, R. S., Kim, S.-H., & Mosaly, P. (2009). Human–automation interaction strategies and models for life science applications. Human Factors and Ergonomics in Manufacturing & Service Industries, 19(6), 601–621. https://doi.org/10.1002/hfm.20156

Kamrani, M., Arvin, R., & Khattak, A. J. (2018). Extracting useful information from connected vehicle data: An empirical study of driving volatility measures and crash frequency at intersections. Transportation Research Record, 2672(38), 290–301.

Liu, Q., Wang, X., Wu, X., Glaser, Y., & He, L. (2021). Crash comparison of autonomous and conventional vehicles using pre-crash scenario typology. Accident Analysis & Prevention, 159, 106281. https://doi.org/10.1016/j.aap.2021.106281

Mahdinia, I., Mohammadnazar, A., Arvin, R., & Khattak, A. J. (2021). Integration of automated vehicles in mixed traffic: Evaluating changes in performance of following human-driven vehicles. Accident Analysis & Prevention, 152, 106006. https://doi.org/10.1016/j.aap.2021.106006

Naito, Y., & Nagatani, T. (2012). Effect of headway and velocity on safety–collision transition induced by lane changing in traffic flow. Physica A: Statistical Mechanics and Its Applications, 391(4), 1626–1635. https://doi.org/10.1016/j.physa.2011.09.020

Rakha, H., Pecker, C. C., & Cybis, H. B. B. (2007). Calibration Procedure for Gipps Car-Following Model. Transportation Research Record, 1999(1), 115–127. https://doi.org/10.3141/1999-13

Rios-Torres, J., Liu, J., & Khattak, A. (2019). Fuel consumption for various driving styles in conventional and hybrid electric vehicles: Integrating driving cycle predictions with fuel consumption optimization*. International Journal of Sustainable Transportation, 13(2), 123–137. https://doi.org/10.1080/15568318.2018.1445321

SAE OnRoad Automated Vehicle Standards Committee. (2018). SAE Levels of Driving AutomationTM Refined for Clarity and International Audience. https://www.sae.org/site/blog/sae-j3016-update

Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., Vasudevan, V., Han, W., Ngiam, J., Zhao, H., Timofeev, A., Ettinger, S., Krivokon, M., Gao, A., Joshi, A., … Anguelov, D. (2020). Scalability in Perception for Autonomous Driving: Waymo Open Dataset. 2446–2454. https://openacces

s.thecvf.com/content_CVPR_2020/html/Sun_Scalability_in_Perception_for_Autonomous_Driving_Waymo_Open_Dataset_CVPR_2020_paper.html

Vogel, K. (2003). A comparison of headway and time to collision as safety indicators. Accident Analysis & Prevention, 35(3), 427–433. https://doi.org/10.1016/S0001-4575(02)00022-2

Wei, H., Liu, X., Mashayekhy, L., & Decker, K. (2019). Mixed-Autonomy Traffic Control with Proximal Policy Optimization. 2019 IEEE Vehicular Networking Conference (VNC), 1–8. https://doi.org/10.1109/VNC48660.2019.9062809

Xu, Z., Yang, D., Tang, J., Tang, Y., Yuan, T., Wang, Y., & Xue, G. (2021). An Actor-Critic-Based Transfer Learning Framework for Experience-Driven Networking. IEEE/ACM Transactions on Networking, 29(1), 360–371. https://doi.org/10.1109/TNET.2020.3037231

Zheng, F., Liu, C., Liu, X., Jabari, S. E., & Lu, L. (2020). Analyzing the impact of automated vehicles on uncertainty and stability of the mixed traffic flow. Transportation Research Part C: Emerging Technologies, 112, 203–219. https://doi.org/10.1016/j.trc.2020.01.017