2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)
October 8-12, 2022, Macau, China

Prediction Based Decision Making for Autonomous Highway Driving

Mustafa Yildirim[1] Sajjad Mozaffari [2] Luc McCutcheon[1] Mehrdad Dianati [2] Alireza Tamaddoni-Nezhad [3]
Saber Fallah[1]

2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC) | 978-1-6654-6880-0/22/$31.00 ©2022 IEEE | DOI: 10.1109/ITSC55140.2022.9922398

***Abstract*— Autonomous driving decision-making is a challenging task due to the inherent complexity and uncertainty in traffic. For example, adjacent vehicles may change their lane or overtake at any time to pass a slow vehicle or to help traffic flow. Anticipating the intention of surrounding vehicles, estimating their future states and integrating them into the decision-making process of an automated vehicle can enhance the reliability of autonomous driving in complex driving scenarios. This paper proposes a Prediction-based Deep Reinforcement Learning (PDRL) decision-making model that considers the manoeuvre intentions of surrounding vehicles in the decision-making process for highway driving. The model is trained using real traffic data and tested in various traffic conditions through a simulation platform. The results show that the proposed PDRL model improves the decision-making performance compared to a Deep Reinforcement Learning (DRL) model by decreasing collision numbers, resulting in safer driving.**

I. Introduction

Highly Automated Vehicles (HAVs) are a rapidly developing technology with a vital role in society. The aim of technology is to improve driving safety for drivers, passengers and pedestrians, reduce traffic congestion and decrease fuel consumption. One major functionality of HAVs is Driving Decision-Making (DDM) which is responsible for making decisions on when to take action(s), what action(s) to perform and how to perform an action (s). These decisions can be one at a time, such as to reach a specific velocity or sequential such as completing an overtake manoeuvre (accelerate-lane change) [1]. HAVs need to cooperate with other traffic participants to make safe and reliable decisions in continuously varying traffic conditions.

Rule-based methods are frequently used for decision-making, specifically for lane-change manoeuvres [2]. Automated vehicles change lanes based on designed rules such as the gap acceptance model [3] or the minimise-overall-braking induced by lane changes (MOBIL) model [4]. Nevertheless, vehicles using rule-based methods are conservative in making decisions in a variety of traffic conditions, which may adversely impact traffic flow [5]. Furthermore, the rules are inflexible and cannot handle unseen driving situations. Methods of DDM should be adaptable and generalisable to cope with uncertainties and unseen driving situations.

[1]Mustafa Yildirim, Luc McCutcheon and Saber Fallah are with CAV-Lab, Department of Mechanical Engineering Sciences, University of Surrey, {m.yildirim,lm01065, s.fallah}@surrey.ac.uk
[2]Alireza Tamaddoni-Nezhad is with Department of Computer Science, University of Surrey a.tamaddoni-nezhad@surrey.ac.uk
[3]Sajjad Mozaffari and Mehrdad Dianati are with Warwick Manufacturing Group, University of Warwick {sajjad.mozaffari,M.Dianati}@warwick.ac.uk

More advanced decision-making techniques compared to rule-based techniques have been studied for highway driving. Decision trees [6] and random forest [7] methods are hugely depend on data and prone to over-fit. The support vector machine (SVM) [8] is sensitive to noise, and a minute change in data might lead to a different result. The game theory approach [9] and Fuzzy-logic [10] are applicable when traffic levels are low, but the complexity of the problem increases proportionally to the number of vehicles. Monte Carlo Tree Search (MCTS) [11] is one promising technique for the decision-making problem of highway driving [12], but a limited number of search branches is considered in most studies due to the computational cost. Combining MCTS with a neural network, known as deep MCTS, helps to better guide the sampling towards the most relevant sub-trees [13] and improves the computational efficiency. Despite their promising results, most of the aforementioned techniques suffer from lack of generalisation or adaptability in unseen driving conditions.

Recently, Reinforcement Learning (RL) has received significant attention from researchers as a powerful technique to solve complex, uncertain DDM problems due to its strong adaptability and generalisation. At first, value-based techniques such as Q-learning [14] were used by researchers for DDM due to its simplicity [15]. However, the performance of such methods for complex and high dimensional driving situations such as dense traffic is limited because of the curse of dimensionality. The combination of Q-learning with neural networks addresses this problem and paves the path for using Q-learning for complex DDM applications [16]. Early implementations of Deep Q-Networks (DQN) was used for lane-keeping control of a vehicle in a race track [17], [18]. Later, DQN algorithms were used by different research works to make decisions in highway driving scenarios [19]. For instance, using a quantile regression DQN (QR-DQN) [20], Min et al. introduced a sensor fusion structure controller to decide lane-keeping, lane changing, and acceleration control in their study [21]. Mo et al. studied [22] the challenging scenario of oncoming traffic by implementing Double DQN [23] as a DDM to perform the lane-change manoeuvre, but decision-making is supported by rule-based Time to Collision (TTC) [24] rewards which calculates collision time based on constant speed. Shi et al. [25] proposed a hierarchical decision-making model, where a DQN model decides when to perform a lane change manoeuvre, and another Q function approximator decides how to complete the manoeuvre. Mirchevska et al. [26], and Shu et. al [27] implemented a safety check to DQN outputs for the

Authorized licensed use limited to: University of Surrey. Downloaded on December 06,2024 at 17:20:32 UTC from IEEE Xplore. Restrictions apply.

execution of action to prevent the collision. Safety check considers output based on safety distance to prevent collision if in case after lane change, the lead vehicle performs sudden brake and cause ego vehicle to collide. The real-time performance of the aforementioned DDM techniques suffers from the lack of foreseeing the other vehicles' intentions. It is known that driving intentions of surrounding vehicles significantly influence the decisions made by HAVs and have to be included in the DDM process.

In separate works, researchers contributed to predicting the intentions of surrounding vehicles. A study by Gindele et al. deals with simplified traffic models and makes accurate predictions based on the Partially Observable Markov Decision Processes (POMDP) [28] . Alizadeh et al. [29] studied the same problem by integrating noise when measuring other vehicles' positions and compared the result with the rule-based lane change model Mobil [30]. In another study [31], authors predicted surrounding cars velocity and used this information for path planning. Based on this prediction, they decreased the calculation cost by 90%. Jiang et al. [32] considered the estimation of other vehicles' intentions for trajectory planning in their work, but the prediction only considered whether the target lane vehicle is cooperative or aggressive when performing the lane change manoeuvre. Kochenderfer et. al. anticipated driver cooperativeness for merging scenario [33]. Recent studies [34], [35], integrated the intention prediction module to decision-making process to improve the performance. For instance, Gonzalez et al. proposed a human-like decision making [12] by implementing a belief tree to estimate lane change of other drivers using MCTS, however the sampling time of tree search and the lack of tractability was the limitation of MCTS. In another study [36], target vehicle trajectory is predicted based on three actions such as we have used in our model, but the prediction horizon considers only two seconds ahead just for the target vehicle on the defined lane, whereas our study contains five seconds prediction horizon for surrounding six vehicles. The studies in literature either predict the behaviour as a general (such as aggressive or passive) or generate a belief tree based on initial prediction/assumption. In contrast to the literature mentioned above, our study continuously considers lane change prediction and updates the prediction for each observation state for a longer time zone as long as the surrounding vehicles are within the radar range.

This paper focuses on how to integrate intention predictions of other vehicles into the DDM process of a HAV to make safer decisions through DQN methods. The paper proposes a prediction-based DDM method for lane-change highway driving scenarios by integrating the intention of surrounding vehicles as a time to lane change in the decision-making model, which helps decreasing potential collisions. In this paper, different DQN techniques are used to formulate the DDM problem and their performance's are compared with and without a prediction module.

The paper is structured as follows. In section 2, the proposed decision-making methodology is explained. Section 3 clarifies network parameters as well as training and test evaluations. Section 4 discusses and evaluates the results, and section 5 concludes the study.

II. Proposed Method

A. Assumptions

We assume that an Ego vehicle (EV) and target vehicles (TVs) are driving on a three-lane straight highway. The EV is equipped with a 360-degree radar sensor that can detect the position and velocity of surrounding vehicles within $R = 250$ m distance to the EV. Using a bird-eye view (BEV) camera, we can observe the environment for the prediction of up to six adjacent vehicles as shown in Figure 1. The Ego vehicle utilizes all this information to predict surrounding cars' intention (whether they are going to perform a lane change or stay on the current lane).

The manoeuvre intention predictor consists of two parts as shown in Figure 2. The first part is a classifier that identifies the intention of the other cars as P_{LK}, P_{RLC}, P_{LLC} where LK represents a lane-keeping manoeuvre, RLC and LLC represent lane change manoeuvres to the right and to the left, respectively. The second part is a regressor which estimates Time-to-Lane Change (TTLC), t ϵ T as $\{5, 4, 3, 2, 1, 0\}$ where numbers represent time in seconds when classifier actions occur.

By adding this information, the decision-making model has distance, velocity and prediction information of nearby vehicles. Having all this information, the decision-making model controls the EV lane change decision such as left lane change, right lane change or lane-keeping and reacts instantly in vehicle dynamics limit by maintaining driving safety.

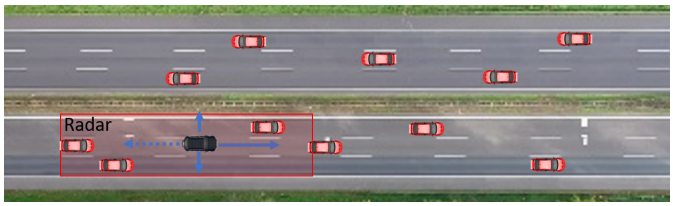

Fig. 1. Ego vehicle driving on highway

B. Decision-Making Model using Reinforcement Learning

A Markov Decision Process (MDP) is the mathematical framework for decision making that is the basis of the RL problem formulation. [38]. An MDP is composed of an action set A, a state set S, a reward function R and a transition model P(s'|s, a). An RL agent learns to maximise expected cumulative reward in an MDP by taking an action a ϵ A, which reaches a new state s' ϵ S and receives a reward r. The agent updates its policy π to maximise future cumulative rewards and the process is repeated until the agent has sufficiently optimised the policy to achieve the highest score in the environment.

Cumulative rewards is the summation of a sequence of rewards received. A discount factor γ ϵ [0,1] is then applied

Authorized licensed use limited to: University of Surrey. Downloaded on December 06,2024 at 17:20:32 UTC from IEEE Xplore. Restrictions apply.

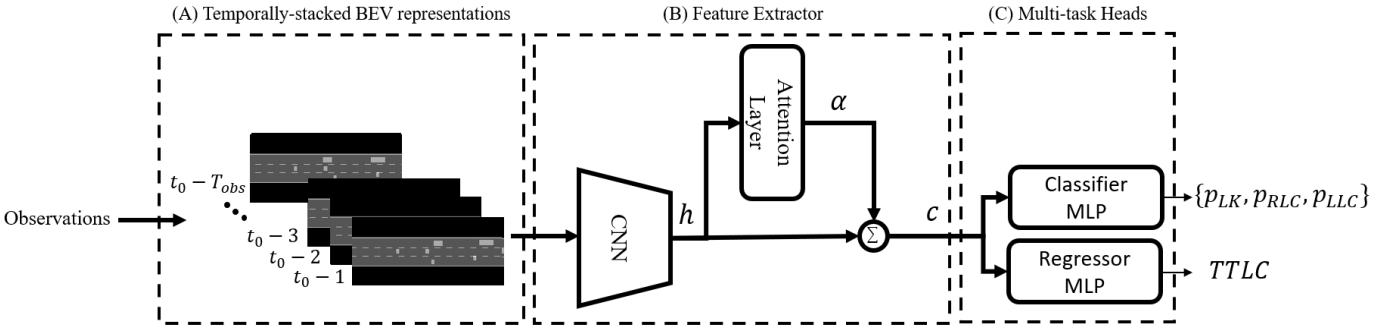

Fig. 2. An overview of the prediction model [37]

at each time-step to trade off the importance of immediate rewards over long term rewards.

The state-space includes the position and velocity of EV, surrounding vehicles' distance to EV, their velocities, and intentions. The EV's action space is defined as a left lane change, right lane change or lane-keeping A={LLC, RLC, LK}. The goal of RL is to find optimal policy π that maximizes total future rewards:

$$R(\pi, r) = \mathbb{E}_{\pi}\left[\sum_{t=0} \gamma^t r\left(s_t, a_t\right)\right] \tag{1}$$

When solving sequential decision problems, estimations are learned for the optimal value of each action, which is defined as the expected value of the rewards in the future when taking that action and following the optimal policy accordingly. The policy is used to predict the value for each action in A for the current state. This action-value function is formulated as:

$$Q_{\pi}(s, a) = \mathbb{E}[R_1 + \gamma R_2 + ... | S_0 = s, A_0 = a, \pi] \tag{2}$$

Actions can then be chosen greedily with respect to this action-value function, or alternatively, exploratory actions can be taken given the current observation s ϵ S.

A neural network is then trained by minimising the loss function below at each iteration i, optimising the network weights θ,

$$\begin{aligned} L_i\left(\theta_i\right) &= \mathbb{E}_{s,a\sim\rho(\cdot)}\left[\left(y_i - Q\left(s, a; \theta_i\right)\right)^2\right] \\ y_i &= \mathbb{E}_{s'\sim\varepsilon}\left[r + \gamma \max_{a'} Q\left(s', a'; \theta_{i-1}\right) \mid s, a\right] \end{aligned} \tag{3}$$

y_i is the target and $\rho(s, a)$ is the probability distribution of states and actions. The differential of the loss function with respect to the weights gives us the following gradient.

$$\begin{aligned} \nabla_{\theta_i} L_i\left(\theta_i\right) = \mathbb{E}_{s,a\sim p(\cdot);s'\sim\mathcal{E}}\Bigg[\left(r + \gamma \max_{a'} Q\left(s', a'; \theta_{i-1}\right)\right) \\ - Q\left(s, a; \theta_i\right) \nabla_{\theta_i} Q\left(s, a; \theta_i\right)\Bigg] \end{aligned} \tag{4}$$

This gradient is followed through Adam (Adaptive Moment Estimation) gradient decent on each iteration.

C. Reward Functions

- Collision Penalty: Collision is an undesirable situation and to prevent this event, the highest penalty is given for collision status.
- End of Track: If the agent completes the track and reaches the goal, it receives a positive reward value.
- Lane Change Penalty: To prevent an unnecessary lane change, the agent receives a small penalty if there is a lane change, since the value is very low comparing to other rewards and penalties, it does not effect the goal of the agent but prevents it from taking unnecessary actions.

Combining all reward functions defines the total reward function as follows:

$$\begin{aligned} Reward\ Function = R_{End\ of\ Track} - R_{Lane\ Change} \\ - R_{Collision} \end{aligned} \tag{5}$$

D. Intention Prediction Model

The proposed decision-making model (Figure 3) is combined with predicted actions of surrounding cars. The information for the lane change manoeuvre fed into the fully connected layer for each state, in additional to velocity and distance information. The intention of other cars to lane change is implemented as Time-To-Lane-Change (TTLC) which is introduced in the study of Mozaffari et al. [37]. Figure 2 shows an overview of the prediction model and a summary of key processing steps of the prediction model is provided below:

- **BEV input representation**: First, the states of the target vehicle and its surrounding vehicles at each time-step are used to generate a simplified top-down view representation of the driving environment. This representation includes the vehicle bounding boxes, the road marking and the drivable area, each encoded with a specified value. A temporal channel-wise stack of BEV representation for the past few time-steps creates the input data at the current time-step.
- **Attention-based Feature Extractor**: A six-layer Convolutional Neural Network (CNN) [39] is used to extract relevant features for the prediction task from the stacked

Authorized licensed use limited to: University of Surrey. Downloaded on December 06,2024 at 17:20:32 UTC from IEEE Xplore. Restrictions apply.

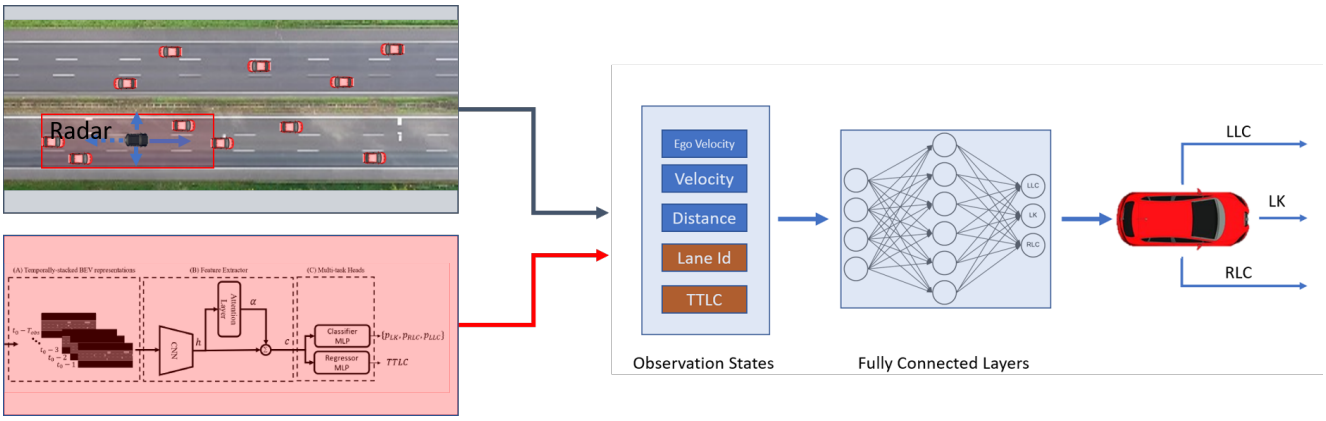

Fig. 3. Proposed TTLC method network structure

BEV representations. To enhance the feature extraction performance, a special attention mechanism is designed to selectively focus on one of the quarter areas around the target vehicle, namely, front-right, front-left, back-right and back-left. The introduction of the attention mechanism helps the prediction model to identify the informative area of the surrounding environment in each prediction query.

- **Multi-task Heads**: A multi-task learning approach has been used in dealing with lane change prediction problem. The lane change prediction problem is defined as estimating the likelihood of lane change manoeuvres (i.e., right lane change, left lane change and lane-keeping) within a prediction horizon and regressing the Time-To-Lane-Change. A separate multilayer perceptron head is considered for the likelihood estimation and regression parts, although both heads use the extracted features by the attention-based CNN model. The overall model is trained using a weighted sum of a mean squared loss for the regression part and cross-entropy loss for the classifier (i.e., likelihood estimation).

E. Vehicle Longitudinal and Lateral Controllers

The EV lateral controls are defined as discrete actions and the decision making model perform actions to change or keep lane. The Intelligent Driver Model (IDM) [40] controls the longitudinal movement.

1) Actions: The agent has three actions: left lane change, right lane change, and stay at the current lane. These actions are discrete, and the decision-making model performs the best action based on observed inputs. The IDM controls the EV's acceleration and velocity.

2) IDM: The IDM controls vehicle acceleration and deceleration based on the distance and velocity between the EV and the vehicle in front of the EV, and adjust vehicle velocity accordingly. The IDM automatically prevents collision by observing distance and decreasing velocity if the EV is too close to the front vehicle. In our model, acceleration is calculated based on the below equations and parameters [41] shown in the Table I for the IDM model.

$$a = a_{\max}\left[1 - \left(\frac{v}{v_{\text{desired}}}\right)^4 - \left(\frac{s^*\left(v, v_{\text{lead}}\right)}{s}\right)^2\right] \tag{6}$$

$$s^*\left(v, v_{\text{lead}}\right) = \max\left(s_0, v\rho + \frac{1}{2}a_{\max}\rho^2 + \right. \\ \left. + \frac{\left(v + \rho a_{\max}\right)^2}{2b_{\text{safe}}} - \frac{\left(v_{\text{lead}}\right)^2}{2b_{\max}}\right) \tag{7}$$

TABLE I
IDM PARAMETERS

Parameter	*Value*
Minimum Distance (s_0)	5 m
Desired Velocity ($V_{desired}$)	130 km/h
Maximum Acceleration (a_{max})	3 m/s
Maximum Deceleration (b_{max})	5 m/s
Safe Deceleration (b_{safe})	4 m/s
Response Time (ρ)	0.25 s

F. Reinforcement learning Methods for DDM

The Deep Q-Networks (DQN) differs from Q-Learning by using deep networks composed of layers and neurons instead of a Q-table. State, action, reward, and next state are the main elements of DQN. Based on observation states, the agent performs random actions using an epsilon greedy policy to explore the environment. Based on these states and actions, the agent reach a new state and obtains a reward. The action-value (Q-value) is calculated based on rewards obtained using the Bellman equation. The targets are the Q-values of each of the actions and the input would be the state that the agent is in and the intention prediction of surrounding cars. This is an iterative process where the agent stores learning information (state, action, reward, next state, terminal state flag) in the replay buffer. The agent then learns to optimise the Q-function to maximise future expected cumulative reward using a random batch of stored transitions from the replay buffer. Repeating this process

Authorized licensed use limited to: University of Surrey. Downloaded on December 06,2024 at 17:20:32 UTC from IEEE Xplore. Restrictions apply.

Fig. 4. Simulation of HighD traffic

helps the agent to maximize its cumulative reward, allowing the agent to perform the best action for a new state. The algorithm for Deep Q-learning with experience replay is shown below. Due to the variety of methods, only the base algorithm [42] is given here.

Algorithm 1: DQN with Experience Replay

Initialize replay memory $\mathcal{D}$
Initialize action-value function Q with random weights
for $episode = 1,\ M$ **do**
  Initialise sequence $s_1 = \{x_1\}$ and $\phi_1 = \phi(s_1)$
  for $t = 1,\ T$ **do**
    Select random action a_t with probability ϵ
    select $a_t = max_a Q * (\phi(s_t), a; \theta)$ based on $1 - \epsilon$
    Execute action a_t and obtain reward r_t and state x_{t+1} Set $s_{t+1} = s_t, a_t, x_{t+1}$ and preprocess $\phi_{t+1} = \phi(s_{t+1})$
    Store transition $(\phi_t, a_t, r_t, \phi_{t+1})$ in $\mathcal{D}$
    Sample random minibatch of transitions $(\phi_j, a_j, r_j, \phi_{j+1})$ from $\mathcal{D}$

$$Sety_j = \begin{cases} r_j : \text{for terminal } \phi_{j+1} \\ r_j + \gamma max'_a Q(\phi_{j+1}, a'; \theta) \\ \quad : \text{for non-terminal } \phi_{j+1} \end{cases}$$

    Perform gradient decent step on $(y_i - Q(\phi_j, a_j; \theta))^2$ according to equation 4
  end for
end for

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \\ + \alpha[r_{t+1} + \gamma \max_a Q(s_{t+1}, a) - Q(s_t, a_t)] \quad (8)$$

The DQN method, by its nature, might have tendency to overestimate Q-values. To overcome this drawback, many improvements have been proposed in the literature. To extend our study and compare the methods proposed in literature, five different DQN variants have been implemented. The variants included in our study are:

- DQN [42]: Have tendency to overestimate Q-values.
- Double DQN [23]: Uses the target network to calculate the Q-value to solve the overestimation problem.
- Averaged DQN [43]:Provides more stable training and reduces approximation errors by taking an average of the last five values.
- Duelling DQN [44]: Separates networks as two layers such as advantage and value.
- Noisy Network [45]: Adds noise to weights to improve exploration efficiency.

III. Performance Evaluation

There are many open-source platforms to simulate traffic environments, such as Carla [46], AirSim [47] and Sumo [48] but generating simulated traffic based on real traffic data was not straightforward to implement to these platforms. Since we aimed to simulate an agent on a real-world application by predicting real drivers intentions, these platforms were also unsuitable. Therefore as shown in Figure 4, we have generated a Pygame [49] based traffic simulation environment.

A. Dataset

Highway driving has been studied in various works based on two main datasets NGSim [50] and HighD [51]. Both datasets contain all necessary information such as lateral and longitudinal position, velocity and acceleration information for each vehicle in traffic. The HighD dataset is collected using a drone capturing a 420 m long part of the German highway. A wide-angle camera observes the position of vehicles and provides accurate data in occluded traffic. In this study, training and testing were performed based on the HighD dataset. Fifteen different track's data were used for training, and a further 15 were used for testing from the 60 track dataset.

Since this traffic is not rule-based or randomly-generated, it reflects real tests results. On the other hand, using datasets has some drawbacks, for example other vehicles can not sense the EV and for that reason, there are inevitable collisions caused by other vehicles.

B. Network & Hyperparameters

Our initial evaluations were performed to determine hyperparameters for optimal and fast convergence. Parameters were chosen based on initial runs (see Table II). In addition to these hyperparameters, analysis was performed to use the dropout feature [52]. Decision making model with the dropout could not be converged to the stable loss value. This feature was also tested as an experiment but it did not contribute to the decision-making model and therefore dropout was excluded.

The network was composed of 3 fully connected layers; the first two layers include 128 nodes, and the final layer is formed of 3 nodes to represent agent action space. The Pytorch library was used for neural networks computations. Networks were optimized using the ADAM algorithm, which is computationally efficient and converges rapidly [53]. The

Authorized licensed use limited to: University of Surrey. Downloaded on December 06,2024 at 17:20:32 UTC from IEEE Xplore. Restrictions apply.

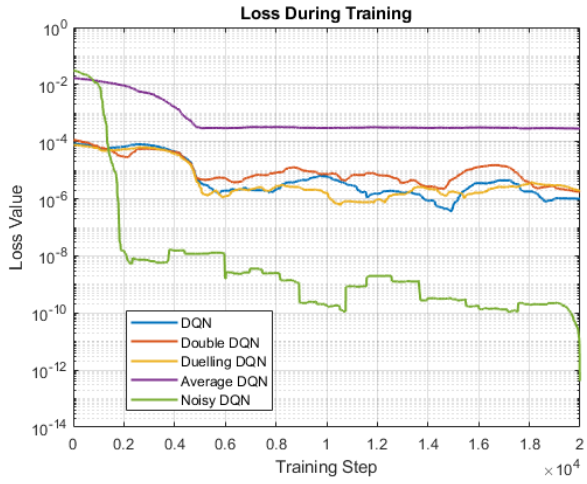

Fig. 5. Loss comparison of methods

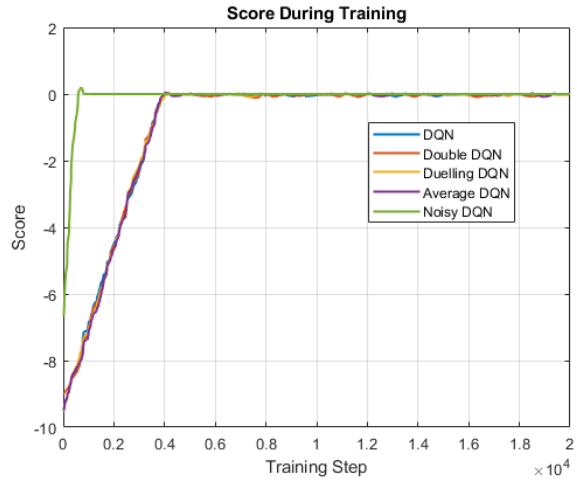

Fig. 6. Score comparison of methods

discount rate was set to $\gamma = 0.95$, and the learning rate to $\alpha = 10e - 5$. The size of the experience replay memory is 10 000. The batch size for stochastic gradient descent is 32. Epsilon-greedy exploration policy was used for all methods except Noisy networks, starting from $\epsilon = 1$ and decreasing to $\epsilon = 0.01$ as a minimum value.

C. Experiments

Initially, 20 000 episodes were chosen since methods were converging between 12 000 and 14 000 episodes, but after hyperparameter optimizations, methods converged around 7 000. Training time differs between methods. Base DQN training finished in 10 h, Noisy DQN and Double DQN were completed in 11 h, Duelling DQN in 16 h and Average DQN in 19 h. TTLC methods added an additional two hours for each method.

The loss graph of methods in Figure 5 shows that the most smooth curve belongs to Average DQN; although it has a higher loss value than other methods, it is the most stable.

TABLE II
MODEL HYPERPARAMETERS

Hyperparameter	*Value*
Learning Rate (α)	10e-5
Discount Factor (γ)	0.95
Epsilon min (ϵ)	0.01
Memory	10e4
Batch Size	32

On the other hand, Noisy DQN has the lowest loss value and has more fluctuations than other methods. In general, DQN, Double DQN and Duelling DQN have similar values in between. DQN is slightly lower than others, and Double DQN is noticeably higher than Duel DQN. Comparison of methods in terms of the score shows that all methods have reached the maximum score and have a stable policy as seen in Figure 6. It is observed that Noisy DQN has reached the maximum score sooner than other DQN variants, and it shows that Noisy DQN is more sample efficient than others. This is because it explores the environment differently than the epsilon-greedy approach.

The tests were performed 45 times, and the average collision number is measured for comparison. Both the base method and proposed method have been tested on the same tracks. As it can be seen from Table III, the proposed TTLC approach improved the performance of decision-making for highway driving comparing to base methods. Comparing all, among the base methods, Noisy DQN showed the best performance as having the lowest collision number on average as 11.55, followed by Averaged DQN as 12.33 and DQN as 12.73. On the other hand, the proposed method contributed most to the Averaged DQN as 29.2% improvement, and collision number is decreased to 9.53 and followed by DQN with 10.84 collisions on average and then Noisy DQN as 10.93 collisions. The proposed TTLC method has contributed a significant improvement to the results. The most noticeable contribution was to the Averaged DQN, followed by Duelling DQN as 21.22% and DQN as 17.43%. On average, for the five methods proposed, there was 15.51% less collisions in general with ground truth data. It can be inferred from Table III that the predicted TTLC shows less contribution for each method comparing to the ground truth. The predicted TTLC decreased the collision rate by 8.19% when averaged over all methods.

IV. DISCUSSION

The results of this study show that the best performance is obtained by Noisy DQN for base approaches and Averaged DQN for the proposed method. The novel improvements and extensions were implemented to DQN, such as Double and Duelling DQN, but these two extensions could not perform better than DQN. A general assumption about these extensions is; Double DQN and Duelling DQN methods have better performance than DQN, as shown in [44] and [54]. However, this is not consistent in all environments, as

Authorized licensed use limited to: University of Surrey. Downloaded on December 06,2024 at 17:20:32 UTC from IEEE Xplore. Restrictions apply.

TABLE III

COLLISION NUMBER COMPARISON FOR ALL 5 DQN EXTENSION FOR BASE AND PROPOSED TTLC METHOD BY USING GROUND TRUTH AND PREDICTION MODEL

	Base	Ground Truth TTLC		Predicted TTLC	
	$CollisionNumber$	$CollisionNumber$	$Improvement\ \%$	$CollisionNumber$	$Improvement\ \%$
DQN	12.73	10.84	17.43	12.44	2.33
Double DQN	13.11	12.6	4.04	12.02	9.06
Averaged DQN	12.33	9.53	29.3	9.93	19.46
Duelling DQN	13.82	11.4	21.22	12.57	9.94
Noisy DQN	11.55	10.93	5.6	11.53	0.17

seen in [44]. Reinforcement learning approaches distinctly depend on the environment and action space of the agent since everything is built on the interaction of these two. According to [44], Duel DQN showed -100% worse performance than DQN on the Freeway environment in which an agent attempts to avoid other traffic participants and pass through the highway. In this environment, similar to our case, the agent has only three actions and avoids collisions. As a result, Duelling DQN is not always better than DQN. In limited action space, it shows reduced performance compared to DQN. The advantage of the duelling network lies in its ability to approximate values efficiently. When the number of actions is high, this advantage over single-stream Q networks increases [55]. It shows superior performance if there is a possibility of the agent having multiple actions per time step, such as the Atlantis environment [56].

The proposed TTLC method improved the result significantly and provided a safer autonomous drive. The contribution of TTLC varied in different DQN extensions and Averaged TTLC showed the best improvement among the TTLC methods. The prediction horizon overlaps with the Average DQN since the Average DQN takes an average of the last five values and the prediction model considers ongoing five seconds for each time step.

The different methods cause variations in the neural network weight updates and give rise to the fluctuations between methods. In addition, a limitation of our model is that the radar can only detect up to 6 cars at a time which causes uncertainty and impedes the decision-making model for the cases where more than six surrounding vehicles are present as they are not visible to the EV. These two factors could be the reason for the variance between the DQN extensions. However, despite differences between the extensions, on average, the proposed TTLC contributed to having 15.51% less collision than the base methods. This improvement clearly shows our proposed TTLC methods' contribution compared to the base methods and therefore concludes that TTLC promises a better approach for real-world applications such as highway driving.

V. CONCLUSION

In this work, we have tested various DQN methods for autonomous highway driving. A decision-making model integrated with a prediction model has been proposed to improve highway driving safety. The intention of surrounding cars is integrated into the decision-making model and compared with base methods. The proposed method shows that predicting surrounding cars' intention to lane change decreases collision possibility and provides safer driving than base approaches. This study has revealed that Averaged DQN TTLC showed the best performance within the methods in this environment. Another outcome of the study is that the action space influences the contribution of DQN variants to the performance.

REFERENCES

[1] Philip E Ross. Robot, you can drive my car. *IEEE Spectrum*, 51(6):60–90, 2014.

[2] Axel Niehaus and Robert F Stengel. Probability-based decision making for automated highway driving. *IEEE Transactions on Vehicular Technology*, 43(3):626–634, 1994.

[3] Peter G Gipps. A model for the structure of lane-changing decisions. *Transportation Research Part B: Methodological*, 20(5):403–414, 1986.

[4] Arne Kesting, Martin Treiber, and Dirk Helbing. General lane-changing model mobil for car-following models. *Transportation Research Record*, 1999(1):86–94, 2007.

[5] Tingting Li, Jianping Wu, and Ching Yao Chan. Evolutionary Learning in Decision Making for Tactical Lane Changing. In *2019 IEEE Intelligent Transportation Systems Conference, ITSC 2019*, 2019.

[6] J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

[7] Tin Kam Ho. Random decision forests. In *Proceedings of 3rd international conference on document analysis and recognition*, volume 1, pages 278–282. IEEE, 1995.

[8] Yonggang Liu, Xiao Wang, Liang Li, Shuo Cheng, and Zheng Chen. A novel lane change decision-making model of autonomous vehicle based on support vector machine. *IEEE Access*, 7:26543–26550, 2019.

[9] Fanlin Meng et al. Dynamic decision making in lane change: Game theory with receding horizon. In *2016 UKACC 11th International Conference on Control (CONTROL)*, pages 1–6. IEEE, 2016.

[10] Esmaeil Balal, Ruey Long Cheu, and Thompson Sarkodie-Gyan. A binary decision model for discretionary lane changing move based on fuzzy inference system. *Transportation Research Part C: Emerging Technologies*, 67:47–61, 2016.

[11] David Silver and Joel Veness. Monte-carlo planning in large pomdps. *Advances in neural information processing systems*, 23, 2010.

[12] David Sierra González, Mario Garzón, Jilles Steeve Dibangoye, and Christian Laugier. Human-like decision-making for automated driving in highways. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 2087–2094. IEEE, 2019.

[13] Carl Johan Hoel, Driggs-Campbell, Krister Katherine Wolff, Leo Laine, and Mykel J. Kochenderfer. Combining Planning and Deep Reinforcement Learning in Tactical Decision Making for Autonomous Driving. *IEEE Transactions on Intelligent Vehicles*, 5(2):294–305, 2020.

[14] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

Authorized licensed use limited to: University of Surrey. Downloaded on December 06,2024 at 17:20:32 UTC from IEEE Xplore. Restrictions apply.

[15] Xin Li, Xin Xu, and Lei Zuo. Reinforcement learning based overtaking decision-making for highway autonomous driving. In *2015 Sixth International Conference on Intelligent Control and Information Processing (ICICIP)*, pages 336–342. IEEE, 2015.

[16] Mustafa Mukadam, Akansel Cosgun, Alireza Nakhaei, and Kikuo Fujimura. Tactical decision making for lane changing with deep reinforcement learning. 2017.

[17] Ahmad EL Sallab, Mohammed Abdou, Etienne Perot, and Senthil Yogamani. Deep reinforcement learning framework for autonomous driving. *Electronic Imaging*, 2017(19):70–76, 2017.

[18] Peter Wolf, Christian Hubschneider, Michael Weber, Andre Bauer, Jonathan Härtl, Fabian Durr, and Johann Marius Zöllner. Learning how to drive in a real world simulation with deep q-networks. *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 244–250, 2017.

[19] Sampo Kuutti, Richard Bowden, Yaochu Jin, Phil Barber, and Saber Fallah. A survey of deep learning applications to autonomous vehicle control. *IEEE Transactions on Intelligent Transportation Systems*, 22(2):712–733, 2021.

[20] Will Dabney, Georg Ostrovski, David Silver, and Remi Munos. Implicit quantile networks for distributional reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 1096–1105. PMLR, 10–15 Jul 2018.

[21] Kyushik Min, Hayoung Kim, and Kunsoo Huh. Deep q learning based high level driving policy determination. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 226–231. IEEE, 2018.

[22] Shuojie Mo, Xiaofei Pei, and Zhenfu Chen. Decision-Making for Oncoming Traffic Overtaking Scenario using Double DQN. *3rd Conference on Vehicle Control and Intelligence, CVCI 2019*, pages 1–4, 2019.

[23] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

[24] C Hydén. Traffic conflicts technique: state-of-the-art. *Traffic safety work with video processing*, 37:3–14, 1996.

[25] Tianyu Shi, Pin Wang, Xuxin Cheng, Ching Yao Chan, and Ding Huang. Driving decision and control for automated lane change behavior based on deep reinforcement learning. *arXiv*, pages 2895–2900, 2019.

[26] Branka Mirchevska, Christian Pek, Moritz Werling, Matthias Althoff, and Joschka Boedecker. High-level Decision Making for Safe and Reasonable Autonomous Lane Changing using Reinforcement Learning. *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, 2018-Novem:2156–2162, 2018.

[27] Arash Mohammadhasani, Hamed Mehrivash, Alan Lynch, and Zhan Shu. Reinforcement learning based safe decision making for highway autonomous driving. *arXiv preprint arXiv:2105.06517*, 2021.

[28] Tobias Gindele, Sebastian Brechtel, and Rudiger Dillmann. Learning driver behavior models from traffic observations for decision making and planning. *IEEE Intelligent Transportation Systems Magazine*, 7(1):69–79, 2015.

[29] Ali Alizadeh, Majid Moghadam, Yunus Bicer, Nazim Kemal Ure, Ugur Yavas, and Can Kurtulus. Automated lane change decision making using deep reinforcement learning in dynamic and uncertain highway environment. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 1399–1404, 2019.

[30] Martin Treiber and Arne Kesting. Car-Following Models Based on Driving Strategies. *Traffic Flow Dynamics*, pages 181–204, 2013.

[31] Junqing Wei, Jarrod M Snider, Tianyu Gu, John M Dolan, and Bakhtiar Litkouhi. A behavioral planning framework for autonomous driving. In *2014 IEEE Intelligent Vehicles Symposium Proceedings*, pages 458–464. IEEE, 2014.

[32] Shenghao Jiang, Jiying Chen, and Macheng Shen. An Interactive Lane Change Decision Making Model With Deep Reinforcement Learning. *2019 IEEE 7th International Conference on Control, Mechatronics and Automation, ICCMA 2019*, pages 370–376, 2019.

[33] Maxime Bouton, Alireza Nakhaei, Kikuo Fujimura, and Mykel J Kochenderfer. Cooperation-aware reinforcement learning for merging in dense traffic. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 3441–3447. IEEE, 2019.

[34] Mohammad Bahram, Anton Wolf, Michael Aeberhard, and Dirk Wollherr. A prediction-based reactive driving strategy for highly automated driving function on freeways. In *2014 IEEE Intelligent Vehicles Symposium Proceedings*, pages 400–406. IEEE, 2014.

[35] Simon Ulbrich and Markus Maurer. Towards tactical lane change behavior planning for automated vehicles. In *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, pages 989–995. IEEE, 2015.

[36] Georg Schildbach and Francesco Borrelli. Scenario model predictive control for lane change assistance on highways. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, pages 611–616. IEEE, 2015.

[37] Sajjad Mozaffari, Eduardo Arnold, Mehrdad Dianati, and Saber Fallah. Early lane change prediction for automated driving systems using multi-task attention-based convolutional neural networks. *IEEE Transactions on Intelligent Vehicles*, pages 1–1, 2022.

[38] Richard Bellman. Markovian decision processes, 1977.

[39] Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*, 2014.

[40] Martin Treiber, Ansgar Hennecke, and Dirk Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical Review E - Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics*, 62(2):1805–1824, 2000.

[41] Mohammad Naghshvar, Ahmed K Sadek, and Auke J Wiggers. Risk-averse behavior planning for autonomous driving under uncertainty. *arXiv preprint arXiv:1812.01254*, 2018.

[42] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

[43] Oron Ansehel, Nir Baram, and Nahum Shimkin. Averaged-DQN: Variance reduction and stabilization for Deep Reinforcement Learning. In *34th International Conference on Machine Learning, ICML 2017*, volume 1, pages 240–253, 2017.

[44] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Van Hasselt, Marc Lanctot, and Nando De Frcitas. Dueling Network Architectures for Deep Reinforcement Learning. *33rd International Conference on Machine Learning, ICML 2016*, 4(9):2939–2947, 2016.

[45] Meire Fortunato, Mohammad Gheshlaghi Azar, Bilal Piot, Jacob Menick, Ian Osband, Alex Graves, Vlad Mnih, Remi Munos, Demis Hassabis, Olivier Pietquin, et al. Noisy networks for exploration. *arXiv preprint arXiv:1706.10295*, 2017.

[46] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. In *Conference on robot learning*, pages 1–16. PMLR, 2017.

[47] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and service robotics*, pages 621–635. Springer, 2018.

[48] Pablo Alvarez Lopez et al. Microscopic traffic simulation using sumo. In *2018 21st international conference on intelligent transportation systems (ITSC)*, pages 2575–2582. IEEE, 2018.

[49] Will McGugan. *Beginning game development with Python and Pygame: from novice to professional*. Apress, 2007.

[50] Hwasoo Yeo, Alexander Skabardonis, John Halkias, James Colyar, and Vassili Alexiadis. Oversaturated freeway flow algorithm for use in next generation simulation. *Transportation Research Record*, 2088(1):68–79, 2008.

[51] Robert Krajewski, Julian Bock, Laurent Kloeker, and Lutz Eckstein. The highd dataset: A drone dataset of naturalistic vehicle trajectories on german highways for validation of highly automated driving systems. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2118–2125. IEEE, 2018.

[52] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[53] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[54] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double Q-Learning. *30th AAAI Conference on Artificial Intelligence, AAAI 2016*, pages 2094–2100, 2016.

[55] Tian Tan and Emma Brunskill. Cs234 notes-lecture 6 cnns and deep q learning, 2018.

[56] Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

Authorized licensed use limited to: University of Surrey. Downloaded on December 06,2024 at 17:20:32 UTC from IEEE Xplore. Restrictions apply.