Prediction-Aware and Reinforcement Learning-Based Altruistic Cooperative Driving

Rodolfo Valiente®, Mahdi Razzaghpour®, Member, IEEE, Behrad Toghi, Ghayoor Shah®, Graduate Student Member, IEEE, and Yaser P. Fallah

Abstract—Autonomous vehicle (AV) navigation in the presence of Human-driven vehicles (HVs) is challenging, as HVs continuously update their policies in response to AVs. In order to navigate safely in the presence of complex AV-HV social interactions, the AVs must learn to predict these changes. Humans are capable of navigating such challenging social interaction settings because of their intrinsic knowledge about other agents’ behaviors and use that to forecast what might happen in the future. Inspired by humans, we provide our AVs the capability of anticipating future states and leveraging prediction in a cooperative reinforcement learning (RL) decision-making framework, to improve safety and robustness. In this paper, we propose an integration of two essential and earlier-presented components of AVs: social navigation and prediction. We formulate the AV’s decision-making process as a RL problem and seek to obtain optimal policies that produce socially beneficial results utilizing a prediction-aware planning and social-aware optimization RL framework. We also propose a Hybrid Predictive Network (HPN) that anticipates future observations. The HPN is used in a multi-step prediction chain to compute a window of predicted future observations to be used by the value function network (VFN). Finally, a safe VFN is trained to optimize a social utility using a sequence of previous and predicted observations, and a safety prioritizer is used to leverage the interpretable kinematic predictions to mask the unsafe actions, constraining the RL policy. We compare our prediction-aware AV to state-of-the-art solutions and demonstrate performance improvements in terms of efficiency and safety in multiple simulated scenarios.

Index Terms—Altruistic cooperative driving, prediction-aware, reinforcement learning.

I. INTRODUCTION

The adoption of connected and autonomous vehicles (CAVs) is expected to improve safety and efficiency, decrease traffic accidents, and increase mobility [1]. A necessary step toward the widespread integration of autonomous vehicles (AV) in society is allowing the coexistence of safe AVs and human-driven vehicles (HVs). Nevertheless, coordination and cooperation with HVs are still challenging for AVs, particularly in complex social interactions [1].

Manuscript received 7 October 2022; revised 15 June 2023 and 17 September 2023; accepted 21 September 2023. Date of publication 23 October 2023; date of current version 18 April 2024. This work was supported in part by the National Science Foundation under Grant CNS-1932037. The Associate Editor for this article was J. Hu.

Corresponding author: Rodolfo Valiente.

The authors are with the Connected and Autonomous Vehicle Research Laboratory (CAVREL), University of Central Florida, Orlando, FL 32816 USA (e-mail: ro434186@ucf.edu).

Digital Object Identifier 10.1109/TITS.2023.3323440

In order to experience those benefits and allow real adoption of AVs on the road, AVs should not only perceive and understand the current environment state but also proactively predict their future states and learn to coordinate and influence other agents. The innate capability to anticipate agents’ behaviors and use this knowledge to forecast potential future outcomes allows humans to navigate through complex scenarios; therefore, prediction capabilities are a crucial component in creating secure AVs that can be integrated into society. [4], [5]. CAVs use Vehicle to Vehicle (V2V) communication to acquire precise situational awareness [6], [7], [8], [9], [10], [11]. We highlight the advantage of CAVs to improve AVs’ robustness and safety in two main directions, first, to overcome the limitations of local sensors, and second, to allow coordination among AVs. An effective and reliable means of communication among agents can facilitate AV-HV coordination. AVs and HVs equipped with such reliable vehicular communication can coordinate, improving safety and efficiency [6]. However, even in the presence of perfect communication, HV-AV interactions are still challenging as the behavior and intentions of HVs are still unknown, despite the vehicles’ ability to perceive or share information in the communication network. Anticipating agents’ behaviors and actions is an important part of real AVs, particularly in mixed autonomy environments. Due to the significance, prediction and HV behavior modeling is an active area of research [4], [5], [12], [13], [14].
Fig. 2. An overview of our prediction-aware and social-aware cooperative driving approach for Multi-agent Cooperative Reinforcement Learning to improve the safety of CAVs. The proposed decentralized RL architecture employs an HPN, which gives AVs the ability to anticipate future states, which is used by the VFN to optimize for social utility and by the safety prioritizer to avoid high collision potential actions.

However, while extensive research has been done in forecasting vehicles trajectories for classical AV stack [4], [5], [12], [13]; using prediction in the decision-making process has received less attention. Particularly in the domain of cooperative driving, social interactions, and multi-agent reinforcement learning (MARL), it is important to provide AVs with such capabilities, as presented in Figure 1, in which AV decision-making may benefit from anticipating the future states of other vehicles. Existing literature proposes probabilistic HV modeling [15] learned from human driving data or rule-based or hand-engineering methods to guide the AVs [16]. These systems frequently have difficulty communicating or negotiating with other vehicles in complex scenarios. Other approaches use RL [17], [18], [19] and consider social interactions [18], [19] of AVs-HVs, proposing AVs that can learn from experience and influence HVs, while optimizing a social utility function that benefits all vehicles on the road. However, these approaches do not consider the evolution of the environment into the future and lack the capability of anticipating future states that can be used for decision-making.

Towards this end, we study how CAVs can leverage prediction and social awareness in RL decision-making, to improve safety and efficiency. Therefore, we propose the integration of those important components for AVs, i.e., prediction and social navigation. Figure 2 presents an overview of our approach. To achieve this, first, we propose a Hybrid Predictive Network (HPN) that intends to provide AVs the ability to predict other agents’ potential future, as illustrated in Figure 2-I. Second, we use the HPN in a multi-step prediction chain that delivers a window of predicted observations to the Value Function Network (VFN), as illustrated in Figure 2-II. Finally, the safe VFN relies on a decentralized cooperative RL architecture that optimizes for a social utility and uses expressive velocity map predictions as part of the input states and interpretable kinematic predictions for a safety prioritizer. The safety prioritizer uses the kinematic predictions from the multi-step HPN to constrain the RL policy to ensure the safety of decision-making by masking the Q-states that produce unsafe results, as shown in Figure 2-III. We evaluate our prediction-aware and social-aware AV with related approaches under a variety of settings and show that, given the ability to forecast future observations, the AVs can use our proposed approach to improve safety, effectiveness, and overall traffic flow. We propose a novel architecture integrating the proposed HPN and VFN within a decentralized cooperative reinforcement learning framework, making our system both prediction-aware and social-aware. This integration goes beyond a mere combination of existing techniques. On the contrary, it proposes a new way of processing information within a decentralized cooperative RL framework.

Our contributions are as follows:

- We address the challenge of effectively integrating prediction and decision-making in autonomous vehicles within a cooperative reinforcement learning framework. Our unique approach makes our autonomous vehicles both prediction-aware and social-aware, going beyond a standard combination of existing techniques.
- We propose a solution to the problem of leveraging a prediction chain within a cooperative reinforcement learning framework and present a HPN that provides AVs the ability to anticipate future observations and use it in a multi-step prediction chain that delivers multiple future observations to the value function network.
- We tackle the problem of safety in autonomous driving that uses reinforcement learning policies. To achieve this, we propose a robust safety prioritizer that uses interpretable kinematic predictions from the HPN to minimize future high-risk actions, constraining the RL policy to ensure safe decision-making, enhancing awareness of the imminent hazards, reducing collisions, and accelerating the learning process.

II. RELATED WORK

A. Reinforcement Learning, Social Navigation, and Cooperative Driving

Current research in social navigation has demonstrated the importance of AVs as social actors and the benefits of AV-HV coordination [20]. A method for modeling and forecasting human behavior in situations involving multi-human interactions in highly multi-modal situations is
proposed in [21]. HV models are learned from demonstration using inverse RL in [22] and [23]. Similarly, a centralized stochastic game model approach is presented in [24]. The authors in [25] and [26] proposed a shared reward function to enable cooperative trajectory planning for robots and humans. Sadigh et al. presents a strategy based on imitation learning, allowing AVs to influence HVs [27].

At the traffic level, the importance of coordination and the benefit of using AVs to guide traffic have also been investigated. Wu et al. [28] examines AVs’ ability to stabilize a system of HVs and presents conditions under which enforcing safety constraints on the AVs while stabilizing the traffic improves the overall traffic performance. Similar works have highlighted the potential of influencing HVs and how AVs can guide the traffic flow [28], [29].

Cooperative driving and social interaction are explored by Zimmermann et al. with a game-theoretic approach, where both rewards and sanctions are used to encourage cooperative lane-change maneuvers [30]. A study by Hu et al. takes advantage of cooperative driving and proposed controller aimed to optimize travel duration and ensure fuel efficiency, especially when approaching intersections [31]. Utilizing real-time vehicle data and a nonlinear bicycle model, demonstrates improved fuel efficiency. Long et al. introduce an optimal controller for a roundabout with cooperative optimization [32]. This controller not only boosts roundabout capacity and mobility but is also designed with practicality in mind. Similarly, Wang et al. proposed a Cooperative Adaptive Cruise Control Lane Change (CACCLC) controller designed to facilitate CACC platoon lane changes in dense traffic [33]. This controller demonstrated remarkable improvements in lane-change efficiency, further supporting its potential for real-time application. Deep reinforcement learning and cooperative driving are used by Li et al. for pedestrian collision avoidance [34]. Furthermore, Jiang et al. introduced a cooperative longitudinal control based on Soft Actor-Critic (SAC) RL aimed at reducing traffic oscillations and enhancing platoon stability [35].

Recent works focus on optimizing traffic networks in mixed autonomy to reduce traffic congestion and improve safety. A model of vehicle flow is presented in [36] in which the planner optimizes for a social goal while improving traffic efficiency. The vehicle routing problem is studied in [37], which proposes an innovative learning-augmented local search system using a transformer architecture to mitigate the problem. In contrast to previous works, we do not rely on human cooperation, and secondly, our AVs incorporate prediction and planning to improve decision-making in cooperative driving.

1) Social Value Orientation: Schwarting et al. [2] were the pioneers in proposing the application of Social Value Orientation (SVO) in the form of an angular representation. This concept was utilized to analyze and govern the social behavior of autonomous vehicles in a game-theoretic setup, which differs from our reinforcement learning framework. Crosato et al. [38] devised a solution for the freezing robot problem, which is the result of agents excessively estimating risks posed by human drivers. Our research diverges from this by focusing on a more comprehensive issue of training altruistic agents. These agents can influence and regulate the road behavior of vehicles in their vicinity. Van Vugt et al. [39] also leveraged the concept of SVO to probe into some ethical and societal issues related to AVs. Geary et al. [40] introduced “Information Sufficiency” and utilized it to define a reward function that encourages AVs to make altruistic decisions. Our research also formulates the problem as a Partially Observable Stochastic Game (POSG). However, we distinguish our work by presenting a suitable state representation, a unique architecture, and a prediction-aware planning algorithm for our POSG problem. We adopt the notion of SVO from psychology to measure the agent’s degree of selfishness or altruism.

B. Safety in Autonomous Vehicles
Safety is a priority for real-world adoption of AVs [41], [42]. As a result of the complexity of the driving task, safety concerns have been raised, such as: How can we prioritize AV safety in the face of uncertainty? How can we train RL agents that prioritize AV safety?

Although priority should be given to safety, it often comes at the expense of effectiveness. Consider the following scenario: you decide to pass a vehicle that is moving slowly in front of you. Overtaking a slower vehicle poses a risk because the driver may abruptly change lanes and cause an accident. The only way to ensure safety is to avoid overtaking. Following this and similar arguments, it becomes clear that the only condition that completely guarantees safety is avoiding driving. As a result, the goals of efficiency and safety are frequently conflicting.

AVs based on RL can raise safety concerns as they can select unsafe actions due to function approximation [43]. To improve safety, authors in [44] propose a rule-based system for evaluating controller decisions and masking collision-causing actions. Cameron et al. investigate how humans can supervise agents to achieve an acceptable level of safety [45]. Others use a pure reward-shaping strategy, however, in this case, safety is not prioritized and the AVs are susceptible to selecting dangerous actions [43], [46]. To overcome this problem, authors in [43] present the concept of safe RL, which aims to increase safety in unobserved environment conditions. A short-horizon safety supervisor is proposed in [46] to replace unsafe actions with safer ones. A Q-masking strategy is presented in [47] to prevent collisions by deleting actions that might lead to a crash. Authors in [48] propose a safety supervisor that considerably decreases crashes [48].

To improve safety, we employ a safety prioritizer that uses kinematic predictions from the multi-step prediction network to look ahead and avoid imminent collisions by masking high-risk actions in the short horizon. Different from other studies, we integrate our robust safety prioritizer within a cooperative RL framework. This unique combination is specifically tailored to boost the safety of AVs in cooperative driving scenarios, involving interaction between numerous autonomous and human-operated vehicles.
C. Behavior Modeling and Prediction

Human behavior is difficult to predict, and human decision-making is governed by inherently unobservable cognitive processes. The current works on driver behavior and social navigation approach agents’ coordination by either modeling driver behavior [15], [21], [49], [50] or simplifying and making assumptions about the nature of agent interactions [51], [52]. Other works on driver behavior modeling consider data mining [53], driver attributes [54], graph theory [55], or game theory [2], [30].

In vehicular safety applications, the use of kinematic equations by CAVs for the prediction of neighboring vehicles’ positions and trajectories in short time horizons is a common approach. Dynamic and kinematics-based solutions are used in [56]. These methods usually consider the vehicles to be rigid point masses and assume that the longitudinal velocity, acceleration, or other motion moments are constant, which are frequently accepted assumptions for prediction by conventional vehicle manufacturers. Amongst these kinematic models, the constant speed (CS) or acceleration (CA) model has more popularity for the prediction of road participants’ position and speed in the cooperative vehicle safety application domain [57], [58], [59], [60], [61].

Numerous methods have defined trajectory prediction as a regression problem, and potent methods like Inverse Reinforcement Learning [62], Recurrent Neural Networks [63], [64] and Gaussian Process Regression and Gaussian Mixture Models [65], [66], [67], [68], have been successfully applied in different applications. Authors in [69] present a LSTM model to forecast vehicle’s trajectories. Other works leverage non-parametric Bayesian approaches to predict fundamental patterns of observable time series. Particularly, within the non-parametric Bayesian inference approaches, Gaussian Process (GP) has demonstrated a significant performance [61]. Because of recent advances in deep generative models, generative approaches have been widely used [70], [71]. The majority of works in this domain use Autoencoders, such as Conditional Variational Autoencoder, Recurrent Variational Autoencoder, or Generative Adversarial Network [70], [71].

Differently, we propose a combined approach to predict velocity map images directly using a predictive autoencoder architecture and interpretable kinematics predictions using a GP solution. We use these predictions to improve decision-making, thereby integrating social navigation with prediction, which are crucial elements for AV navigation.

To summarize, our solution proposes an HPN and a VFN, along with a safety prioritizer, to enhance decision-making and safety. The VFN estimates state-action value functions, combining HPN’s predictive capabilities with decision-making, improving navigation. A multi-step prediction chain uses the HPN to generate future hypotheses based on observation history; the VFN strategically leverages the multi-step prediction in an end-to-end system. Integrated within the VFN, the safety prioritizer penalizes high-risk actions, masking them when selected, increasing safety. The proposed method is more than a combination of existing methods, as a whole the solution is a new architecture and algorithm that applies deep reinforcement learning in a prediction-aware and social-aware decentralized cooperative RL framework.

III. PRELIMINARIES AND PROBLEM FORMULATION

A. Autoencoder

An Autoencoder (AE) is a neural network that is trained in an unsupervised approach to minimize the reconstruction error. The AE learns important features that allow reconstructing the original input and its architecture is generally divided into two main components: an encoder and a decoder. Following a similar approach, an AE can be trained for a prediction task assuming that we have the state at time \( t \) and the corresponding state at time \( t + 1 \). Formally, the encoder maps the input \( x \) to a latent feature representation \( z \) denoted by \( z = f_w(x) \). The decoder uses the latent representation \( z \) to obtain a reconstruction \( y \) of the input \( x \), denoted by \( y = f_{wd}(z) \), where \( f_w(x) \) and \( f_{wd}(z) \) represents the encoder and decoder respectively, and \( w_e \) and \( w_d \) are neural network parameters. The reconstruction error, i.e., the difference between \( x \) and the reconstruction \( y \) is used as the objective function.

B. Gaussian Process

Gaussian processes (GP) are frequently used to predict future trajectory time series by regressing the observed time-series realizations from history, capturing the distinct patterns as they emerge in the data, making GP a useful tool for detecting patterns in time series [15], [72]. When using GP to forecast future trajectories, the set of \( m \) observed values is represented by an \( m \)-dimensional multivariate Gaussian random vector, described by an \( m \times m \) covariance matrix and a \( m \) mean vector. This covariance matrix, often known as the GP kernel, is the foundation upon which GP detects and anticipates the underlying behavior of time series based on their recorded history. The fundamental GP components can be expressed mathematically as follows:

\[
\begin{align*}
    f(t) &\sim gp(m(t), k(t, t')) \\
    \{X_i\}_{i=1,2,\ldots,m} &\sim \mathcal{N}(\mu, \Sigma), \\
    \mu &\equiv \left[ m(t_1), \ldots, m(t_m) \right]^T, \\
    \Sigma_{ij} &\equiv \kappa(t_i, t_j) \quad \forall i, j \in \{1, 2, \ldots, m\}
\end{align*}
\]

where \( X_i \), \( f(t) \), \( m(\cdot) \), and \( \kappa(\cdot, \cdot) \) are the samples of the vehicles’ state, observed or to be predicted at the time \( t_i \), the unknown underlying function that the vehicles’ states are sampled from, the mean and the covariance functions, respectively. \( \mathcal{N}(\mu, \Sigma) \) is the multivariate Gaussian distribution with mean vector \( \mu \) and covariance matrix \( \Sigma \).

In this work, we leverage GP to improve kinematics prediction, and instead of working directly with the position time series, our GP inference algorithm treats the vehicles’ heading and longitudinal speed as two independent time series that are regressed using GPs, and then using the predicted heading and longitudinal speed, the vehicles’ positions are calculated. A model built using a non-parametric Bayesian inference framework dynamically adapts its complexity to the observed data, preventing overly complicated models yet catching unexpected patterns in the data as they emerge.
C. Partially Observable Stochastic Games (POSG)

In this section we present the notation for our RL-based altruistic cooperative driving problem, defined as a POSG denoted by \( (\mathcal{I}, \mathcal{S}, \mathcal{P}, \{\mathcal{A}_i\}_{i \in \{1, \ldots, N\}}, \{\mathcal{O}_i\}_{i \in \{1, \ldots, N\}}, \{R_i\}_{i \in \{1, \ldots, N\}}, \gamma, \varepsilon) \), where

- \( \mathcal{I} \): is the set of agents.
- \( \mathcal{S} \): is the state space, a set of possible states.
- \( \mathcal{P} \): is the state transition probability function from state \( s \in \mathcal{S} \) to state \( s' \in \mathcal{S} \), \( \mathcal{P}(s = s'|S = s, A = a) \).
- \( \mathcal{A}_i \): is the action space, set of possible actions for the \( i \)th agent.
- \( \mathcal{O}_i \): is the observation space, set of observations for the \( i \)th agent.
- \( R_i \): is the reward function for the \( i \)th agent.
- \( \gamma \): is the discount factor, \( \gamma \in [0, 1] \).

At a given time \( t \) each agent \( i \in \mathcal{I} \) perceives the environment and receives a partial observation \( o_i : \mathcal{S} \rightarrow \mathcal{O}_i \), considering the observation \( o_i \) and its policy \( \pi_i : \mathcal{O}_i \times \mathcal{A}_i \rightarrow [0, 1] \), the agent takes an action \( a_i \in \mathcal{A}_i \) and transits to the state \( s' \in \mathcal{S} \) based on the transition probability \( \mathcal{P}(s'|s, A) \) and receives a local reward \( r_i \in \mathcal{R} \). Each agent \( i \) seeks to find an optimal policy \( \pi^* : \mathcal{S} \rightarrow \mathcal{A} \), that maximizes the sum of future rewards \( r_i \in \mathcal{R} \), i.e., \( \pi^*(s) = \arg \max_a Q^*(s, a) \), where, \( Q^*(s, a) := \mathbb{E}_\pi[\sum_{t'=0}^{\infty} \gamma^t R_t(s, a)|s_0 = s, a_0 = a] \), in which, \( \gamma \in [0, 1] \) is the discount factor. The optimal action-value function can then be obtained using the Bellman optimality equation, \( Q^*(s, a) = \mathbb{E}_\pi[R(s, a) + \gamma \max_{a'} Q^*(s', a')|s_0 = s, a_0 = a] \).

D. Deep Q-Network

Deep Q-network (DQN) and Double Deep Q-Network (DDQN) have been widely used in RL problems [73]. DDQN regularly samples data from a buffer in order to calculate an estimate of the Bellman error, which is denoted by the following formula:

\[
\mathcal{L}(w) = \mathbb{E}_{s,a,r,s' \sim R,M}[\{(Target - \hat{Q}(s, a; w))^2\}] \tag{2}
\]

\[
Target = R(s, a) + \gamma \hat{Q}(s', \arg \max_a \hat{Q}(s', a'; w); \tilde{w}) \tag{3}
\]

Following this, the DDQN algorithm learns an approximate action-value function (\( \hat{Q}(\cdot) \)) by performing gradient descent steps as \( w_{t+1} = w_t - \alpha_t \nabla_w \mathcal{L}(w) \), on the loss \( \mathcal{L} \). Here, \( w \) represents the online network weights, and \( \tilde{w} \) represents the target network weights (updated at frequency \( Target_{update} \)). The experience replay buffer (RM) is used to generate training samples \( (s, a, r, s') \), which are randomly drawn to protect from correlated observations and non-stationary data distribution.

E. Driving Scenarios and Behaviors

We study the performance of our framework on multiple HV behaviors and scenarios. We design a set of scenarios, \( \mathcal{F} \) such as straight highway, highway exiting, highway merging, intersection, and roundabout scenarios, defined as \( f_h, f_e, f_n, f_l, f_r \in \mathcal{F} \) correspondingly. Using these scenarios, we train AVs that are social-aware by using an altruistic reward that incorporates Social Value Orientation (SVO) in the AVs. Properly, we describe social preferences (altruism or egoism) by the AV’s SVO angular phase \( \phi \) [18], [74]. To simulate diverse behaviors we compute the appropriate parameter values that simulate the desired behaviors. We compute the HV driver parameters (\( \mathcal{P} \)) and based on the parameters (\( \mathcal{P} \)) generate a set of behaviors \( \mathcal{B} \), i.e., conservative, moderate and aggressive, \( b_c, b_m, b_a \in \mathcal{B} \) used within the simulator [55], [75]. A mixed behavior scenario is obtained by sampling from the behaviors in \( \mathcal{B} \).

F. Problem Formulation

In this work, we focus on prediction-aware planning for altruistic cooperative driving. We assume our scenarios contain a set of AVs \( i_j \in \mathcal{I} \) and HVs \( h_k \in \mathcal{H} \) with diverse SVO. We assume that AVs are connected and perceive a partial observation of the environment \( \tilde{o}_i \in \mathcal{O}_i \), perceiving a subset of vehicles \( \mathcal{C} = \mathcal{H} \cup \tilde{\mathcal{I}} \), i.e., a subset of HVs \( \tilde{\mathcal{H}} \subset \mathcal{H} \) and AVs \( \tilde{\mathcal{I}} \subset \mathcal{I} \). We study the following problem: How AVs can leverage prediction in decision-making to learn optimal cooperative policies \( \pi^*(c) \) in a mixed-autonomy environment under different HVs behaviors \( b \in \mathcal{B} \) and scenarios \( f \in \mathcal{F} \).

The RL-based altruistic cooperative driving problem is formalized as a POSG as described previously, attempting to obtain optimal policies that produce socially advantageous outcomes. To formalize our prediction problem, let us represent our state at time \( t \) for the vehicle (car) \( c \), \( c \in \mathcal{C} \), as \( s_t^c \) and let \( s_t = s_t^1, \ldots, s_t^C \) represent the state for all the vehicles within the perception range. We assume that our state \( s_t \) consists of a stack of \( N \) past observations and \( M \) future hypotheses, accounting for temporal and prediction information, i.e., \( s_t = [\tilde{o}_{t-N-1}, \tilde{o}_{t+1:t+M}] \) for all the vehicles within the local observation. The prediction system takes as input the previous observations \( \tilde{o}_{t-N} \) and aims to produce \( \tilde{o}_{t+1:t+M} \). We note that this is the general notation, and in our framework, \( s_t \) is not just the vehicle trajectory, but a combination of vehicle kinematic trajectory and a velocity map, the details of which are presented in the following sections.

The previous \( (\tilde{o}_{t-N}) \) and anticipated observations \( (\tilde{o}_{t+1:t+M}) \) are used to learn an optimal policy at a given state \( s_t \), \( \pi^* : \mathcal{S} \rightarrow \mathcal{A} \). The goal of this work is to train prediction-aware and social-aware AVs that can drive safely in a mixed-autonomy scenario.

IV. PREDICTION-AWARE ALTRUISTIC COOPERATIVE DRIVING

The POSG becomes significantly more complex in the presence of HVs since their behavior is difficult to predict and change over time. Therefore, predicting HV behavior is crucial for AVs’ in a mixed-autonomy environment. On the basis of this insight, we develop a framework that combines prediction and planning. We propose a predictive network that provides predictions to the planner, and the planner learns to use those predictions for decision-making. The prediction networks give the AV the capability to anticipate the future, and the VFN embeds the predictions and learns the inter-agent relations while optimizing for a social utility.

Our approach uses the HPN that provides possible future observations. Then the HPN is used in a multi-step prediction
chain that produces multiple possible future observations to the VFN. Finally, the VFN is trained to optimize a social utility within the RL framework. The VFN outputs Q-values, that are masked by a safety prioritizer, constraining the RL policy to a safe action space. The outline of our framework is presented in Figure 2.

The two main sub-systems are the HPN and the VFN, where HPN is a predictive autoencoder and VFN is a 3D convolutional neural network (CNN). We hypothesize that the combination of prediction (HPN) and decision-making (VFN) improves the AVs’ ability to learn to navigate complex scenarios. The input of the system is the hybrid spatio-temporal state representation, i.e., VelocityMaps and kinematic state and the output are the action-values, and after the unsafe actions are masked, the action with the highest Q-values is selected \( a = \max_{a'} \tilde{A}_{\text{safe}} Q(s, a'; w) \) at the given state \( s \in \mathcal{S} \). To encourage the required safe social behavior in the AVs, we designed a suitable reward function.

A. Action and State Space

1) Action Space: This study aims to investigate interactions between agents and between AVs and HVs. As a result, we decide to choose the action-space as a collection of discrete meta-actions \( a_i \in \mathcal{A}_i \) at an abstract level, and the abstract actions are transformed into control signals. We specifically choose a set of actions \( a_i \) as follows:

\[
\mathcal{A}_i = \begin{bmatrix}
\text{Lane Left} & \text{Idle} & \text{Lane Right} & \text{Accelerate} & \text{Decelerate}
\end{bmatrix}
\]

2) State Space: The AVs at every time step \( t \) receive a local observation of the environment \( \tilde{O}_t \). As temporal information is crucial for the driving task we incorporate \( N \) consecutive observations. We use VelocityMaps \( (\tilde{V}M) \) and Kinematic \( (K) \) information, at time step \( t \), each combination of \( \tilde{V}M \) and \( K \) is an observation from the environment as,

\[
\tilde{O}_t \in \tilde{\mathcal{O}}_t = \begin{bmatrix}
\tilde{V}M_t & K_t = X_t, Y_t, V_t, \psi_t
\end{bmatrix}
\]

The kinematic information is included to explicitly incorporate the movement data, which helps the training process, and also serves to obtain accurate Kinematic prediction for the safety prioritizer. Additionally, as anticipating future states is also important for decision-making in complex scenarios, the prediction chain generates a sequence of \( M \) hypotheses from the observations that provide information on how the environment could probably evolve into the future. We combine \( N \) consecutive past observations and \( M \) hypotheses from the prediction network to construct a more useful state. Therefore, our state consists of a stack of \( N \) past observations and \( M \) futures hypothesis, accounting for temporal and prediction information, i.e., \( s_t = [\tilde{O}_{t-N:t}, \tilde{O}_{t+1:t+M}] \).

The \( \tilde{V}M \) information incorporates the relative vehicle’s speed in pixel values [74]. The \( K_t \) is a matrix in which the rows are the number of vehicles included in the observation and the columns contain the kinematics information for each vehicle. The kinematics information for each vehicle \( c \), i.e., \( k_c^t \), is a 4-dimensional vector encoding the vehicle’s position, velocity, and heading, and it is formed by the kinematics of the vehicle \( x, y, v, \psi \), where \( (x, y) \) represents the vehicle position, \( v \) is the longitudinal speed, \( \psi \) is the heading, and \( (\hat{x}, \hat{y}) \) are computed as \( \hat{x} = v \cos \phi, \hat{y} = v \sin \phi \). The \( K_t \) embeds the kinematics of surrounding vehicles, and it includes the kinematics information \( k_c^t \) for all the \( c \in \mathcal{C} \) vehicles, in addition to the ego vehicle, i.e., \( K_t = [\tilde{k}_c^t, k_1^t, k_2^t, \ldots, k_C^t] \). Each row \( r_c \) of \( K_t \) matrix contains the kinematics information for the vehicle \( c, k_c^t = [x_c^t, y_c^t, v_c^t, \psi_c^t] \).

B. Reward Function

Inspired by the work in sympathy and cooperation for autonomous driving [18], [74], we design a reward function that takes into account the traffic metrics, social and individual rewards of AVs and HVs. Therefore, we separate the reward function for each agent \( I_i \in \mathcal{I} \) in two terms: an egoistic reward \( R^{\text{ego}} \), and a social reward \( R^{\text{social}} \), as

\[
R_i(s, a) = R^{\text{ego}} + R^{\text{social}},
\]

\[
R^{\text{ego}} = \cos \phi r_i(s, a),
\]

\[
R^{\text{social}} = \sin \phi \sum_j r_{AV}^{AV}(s, a) + \sin \phi \sum_k r_{HV}^{HV}(s, a)
\]

in which \( i \in \mathcal{I}, j \in (\mathcal{I} \setminus \{I_i\}) \), \( k \in \mathcal{H}_i \). The ego vehicle’s reward is defined by \( r_i \) and the angle \( \phi \) allows the adjustment of the level of the egoistic and social components.

The \( R^{\text{social}} \) term considers the social utility of the \( k \) HVs and \( j \) AVs for the agent \( i \), i.e., \( r_{AV}^{AV} \) and \( r_{HV}^{HV} \) defined as \( r_{AV}^{AV} = \frac{1}{d_{ik}} \sum_m \omega_m x_m \) and \( r_{HV}^{HV} = \frac{1}{d_{ij}} \sum_m \omega_m x_m \), respectively, where \( m \) is the set of traffic metrics that have been taken into account in the utility of the vehicle (crashes, speed, and distance traveled), \( x_m \) represents the \( m \) metric, and \( \omega_m \) represents the weights (metrics importance). The terms \( d_{ik}/d_{ij} \) represent the gap between the ego-AV and the associated HV/AV and \( \lambda \) is a hyperparameter that sets the importance of neighboring vehicles.

C. Architecture

The proposed prediction-aware planning architecture is presented in Figure 2 and consists of the HPN (Figure 2-I), the multi-step prediction chain (Figure 2-II), the VFN (Figure 2-III), and the safety prioritizer. The HPN (as shown in Figure 3) serves as a predictive autoencoder network. It takes as input the history of observations at time \( t \), i.e., \( \tilde{O}_{t-N:t} \) and produces a predicted observation at time \( t + 1 \), i.e., \( \tilde{O}_{t+1} \). The multi-step prediction chain uses the HPN in a chain to produce a set of \( M \) hypotheses. It takes a history of observations at time \( t \), i.e., \( \tilde{O}_{t-N:t} \) and produces a set of \( M \) predicted observations, i.e., \( \tilde{O}_{t+1:M} \). For the VFN. Prediction-aware planning is made possible by combining prediction (HPN) and decision-making (VFN), which improve driving performance in challenging situations. The details of the architecture are presented in the following sections.
1) Hybrid Predictive Network (HPN): The HPN (Figure 3) is a prediction autoencoder network, it uses the sequence of \( N \) observations at time \( t \), i.e., \( \tilde{\mathbf{o}}_{t-N:t} \) and outputs a predicted observation at time \( t + 1 \), i.e., \( \tilde{\mathbf{o}}_{t+1} \). The HPN consists of a symmetric encoder-decoder architecture. The encoder consists of 3 convolutional layers with 3 convolutional layers and 2-FC layers. The convolutional layers produce a prediction for the next \( K \) layers and the \( V_M \) features and \( K \) features representations) are combined using element-wise addition operation. The decoder consists of a symmetric version of the encoder, i.e., a deconvolutional network with 3-convolutional layers and 2-FC layers. The convolutional layers produce a prediction for the next \( K' \) layers and the \( V_M' \) features and outputs a predicted \( \tilde{\mathbf{o}}_{t+1} \) and the target \( \mathbf{o}_{t+1} \).

Although the AE provides kinematic predictions, we found that an indirect hybrid GP prediction approach to correct the kinematic predictions provides better results. Our findings are based on previous works that show how a GP-based prediction system is powerful for accurate kinematic predictions and often performs better than other models like AE and LSTM models [15], [42], [66].

Therefore, while we use the predictive AE to predict the next \( V_M'_{t+1} \) image and \( K'_{t+1} \) state, we correct the kinematic state \( K'_{t+1} \) using a GP approach to improve kinematics prediction. We particularly find that accurate kinematics predictions are important for the safety prioritizer that uses the predictions to constrain the RL policies to a safer space. In this work, instead of directly using the position time series \( x_{t-N:t}, y_{t-N:t} \) for each vehicle, our GP inference algorithm treats the vehicles’ heading \( \psi_{t-N:t} \) and speed \( v_{t-N:t} \) as two independent time series that are regressed using GPs, and then calculates the vehicles’ position \( x_{t+1,t+M}, y_{t+1,t+M} \) using the predicted heading and speed. We call this approach GP-indirect prediction and present more details in the following sections.

For each vehicle, the GP prediction algorithm takes as input the history of the 4-dimensional kinematic vector \( (x_{t-N:t}, y_{t-N:t}, v_{t-N:t}, \psi_{t-N:t}) \), uses the heading \( \psi_{t-N:t} \) and speed \( v_{t-N:t} \) time series to predict their future values \( (x'_{t+1,t+M}, y'_{t+1,t+M}) \). Then after modeling speed and heading, the future position of the vehicle, i.e., \( x'_{t+1,t+M}, y'_{t+1,t+M} \) is computed as follows,

\[
\{V_i\}_{i=1,2,...,m} = \{f_{\text{speed}}(t_i)\}_{i=1,2,...,m} \sim N(\mu_v, \Sigma_v),
\]

\[
\mu_v = m_v(t_i); \quad \Sigma_{v_{i,j}} = k_v(t_i, t_j)
\]

\[
f_{\text{heading}}(t) \sim \text{gp}(m_{\psi}(t), k_{\psi}(t, t'))
\]

From the output of the GP model, the 4-dimensional kinematic vector \( (k_{t+1}^{GP} = x_{t+1}, y_{t+1}, v_{t+1}, \psi_{t+1}) \) for each vehicle is used to correct the AE kinematic prediction \( (k_{t+1}^{AE}) \), and the GP prediction is performed for each vehicle in the \( K \) matrix (rows of the matrix) and a new matrix is formed with all the predictions at time \( t + 1 \), i.e., \( K'_{t+1} = [k_{t+1}^{GP_{k_1}}, k_{t+1}^{GP_{k_2}}, ..., k_{t+1}^{GP_{k_{|K|}}}]. \) The final predicted observation is a combination of the predicted velocity map \( (V_M'_{t+1}) \) and the corrected kinematic prediction \( (K'_{t+1}) \) as shown in Figure 3, i.e.,

\[
\tilde{\mathbf{o}}_{t+1} = \begin{bmatrix} K'_{t+1} \quad V_M'_{t+1} \quad V'_{t+1} \quad \Psi'_{t+1} \end{bmatrix}
\]

2) Multi-Step Prediction Chain: The prediction chain, as presented in Figure 4 is a multi-step prediction process that uses the HPN (Figure 3) in a chain to produce a set of \( M \) future hypotheses. It takes a history of observation at time \( t \), i.e., \( \tilde{\mathbf{o}}_{t-N:t} \) and produces a set of \( M \) predicted observations, i.e., \( \tilde{\mathbf{o}}_{t+1:t+M} \), as described in Algorithm 1, to compute the input state for the VFN, i.e., \( s_t = [\mathbf{o}_{t-N:t}, \tilde{\mathbf{o}}_{t+1:t+M}] \).

3) Safety Prioritizer: In order to improve safety, we propose a safety prioritizer within our VFN. The safety prioritizer penalizes high-risk actions, thereby reducing imminent crashes. If the AVs come into an unexpected situation and based on the output of the VFN, decide to perform a high-risk action, the safety prioritizer will mask the action. The safety prioritizer is comprised of two algorithms, i.e., Algorithm 2 that checks for safe actions and Algorithm 3 that performs action selection.
Fig. 4. Multi-step prediction chain and Safe Value Function Network (VFN). The prediction chain is a multi-step prediction chain that uses the HPN in a chain to produce a set of \( M \) future hypotheses. The VFN is a 3D CNN that acts as a function approximator, it uses temporal information and prediction to improve decision-making.

Algorithm 1: Multi-Step Prediction Chain

\[
\text{Input } \hat{o}_{t-N:t} \text{. The sequence of previous observations.}
\]

\[
\text{for } t = t \text{ to } t + M \text{ do}
\]

\[
\text{Predict } \hat{o}'_{t+1} = \text{HPN} \left( \hat{o}_{t-N:t} \right)
\]

\[
\text{Save prediction } \hat{o}'_{t+1} \text{ and use it for the next step}
\]

\[
\text{end for}
\]

\[
\text{Output } \hat{o}'_{t+1:t+M}. \text{ The sequence of predicted observations.}
\]

Algorithm 2 verifies if the selected action \( a_t \) is safe based on a safety score for \( M_{steps} \) of prediction. Algorithm 2 simulates \( I_t \) taking the action \( a_t \) and uses the kinematic predictions from HPN, i.e., \( \hat{K}_{t+1:t+M} = \text{HPN} \left( \hat{K}_{t-N:t} \right) \), for all vehicles in the road \( c \in C \) to compute the time-to-collision (\( ttc \)) at time \( t \), i.e., \( ttc \) between \( I_t \) and all \( c \in ( \tilde{I} \cup \tilde{H} ) \setminus \{I_t\} \) using \( x, y, v, \psi \), at each prediction step is calculated and the minimum \( ttc \) is saved, and using the predicted \( ttc \) for all the \( M_{steps} \) of prediction (\( ttc_{t+1:t+M} \)), the \( \text{safety score} \) is computed. The \( \text{safety score} \) is a weighted average of the \( ttc_{t+1:t+M} \), with exponential decay to give more importance to the short-term predictions. Finally, if \( \text{safety score} < \text{safety threshold} \) or any of the predicted \( ttc \) is less than the critical threshold, i.e., \( \text{any}(ttc_{t+1:t+M}) < \text{critical} \), the action is considered unsafe. The \( \text{safety threshold} \) is the safe \( ttc \) threshold for a possible crash, and \( \text{critical} \) is a critical \( ttc \) threshold for an imminent crash. If the current action is considered unsafe, Algorithm 3 will select another action.

The thresholds are based on the concept of Time-To-Collision (TTC). The TTC is defined as the time that would elapse before a collision if both vehicles continue at their present speed and direction. TTC is computed as:

\[
TTC = \frac{d}{\Delta v}
\]

where \( d \) is the distance between the vehicles, and \( \Delta v \) is the relative speed between the vehicles.

This relative speed depends on the speed of the AV vehicle \( (v_{av}) \) and the speed of the other vehicle \( (v_{other}) \):

\[
\Delta v = |v_{av} - v_{other}|
\]

The \( \text{critical} \) is effectively dynamically adjusted based on the speed of our vehicle. As the vehicle’s speed increases, the threshold also increases to account for the longer stopping distances and reaction times. \( \text{critical} \) is computed as follows:

\[
\text{critical} = T_{reaction} + \frac{v}{a_{max}}
\]

where \( T_{reaction} \) is constant and equal to 1.5s, and \( a_{max} \) is the maximum deceleration (braking) capability of the vehicle which is a constant based on the vehicle model. If \( \text{any}(ttc_{t+1:t+M}) < \text{critical} \), the algorithm identifies this as a potential imminent crash scenario.

Algorithm 3 presents the selection of the action. It iteratively verifies the actions using Algorithm 2 and selects a safe action that follows the learned policy. The restricted actions will prevent the agent from engaging in risky behavior during training, resulting in a more balanced learning and efficient sampling.
Algorithm 2 Action Evaluation
Simulate $I_t$ taking the action $a_t$
Get Kinematic predictions from HPN, i.e., $\hat{K}_{t+1:t+M} = \text{HPN}$ ($\hat{K}_{t:t-1}$) for all vehicles in the road $c \in \mathcal{C}$ \ $\{I_t\}$ for $t = t + 1$ to $t + M$ (Compute safety score for $M_{\text{steps}}$ predictions)
\[
\begin{align*}
\text{Compute } & ttc_i \text{ between } I_t \text{ and all } c \in \mathcal{C} \backslash \{I_t\} \\
\text{Compute } & \text{min}(ttc_i) \\
\text{Get the next prediction at } & t = t + 1
end for
\end{align*}
\]
Compute $safety_{score}$ using the predicted $ttc_{t+1:t+M}$
\[
safety_{score} = \frac{\sum_{i=1}^{N_{\text{vehicles}}} w_i ttc_i}{\sum_{i=1}^{N_{\text{vehicles}}} w_i}
\]
if $safety_{score} < safety_{th}$ or any($ttc_{t+1:t+M} < critical_{th}$
\[
\begin{align*}
\text{Return unsafe } & \\
\text{else if } & \\
\text{Return safe }
end if
\end{align*}
\]
Algorithm 3 Action Selection
Initialize $\tilde{A}_{\text{safe}} = A$
while $\tilde{A}_{\text{safe}}$ is not empty do
\[
\begin{align*}
\text{if } & \text{during training then} \\
& \text{Select } a_t \text{ following the exploration policy on set } \tilde{A}_{\text{safe}} \\
\text{else if } & \text{during test then} \\
& \text{Select } a_t \text{ as } \max_{a' \in \tilde{A}_{\text{safe}}} \hat{Q}(s_t, a'; w)
end if
\end{align*}
\]
if $a_t$ is safe (Algorithm 2) then
\[
\begin{align*}
\text{Return } & a_t \\
\text{else } & \\
\text{Remove } a_t \text{ from } \tilde{A}_{\text{safe}}
end if
\end{align*}
\]
end while
Compute the $safety_{y_{\text{score}}}$ as in Algorithm 2
Return $a_t$ with highest $safety_{y_{\text{score}}}$ in $A$

4) Safe Value Function Network (VFN): The VFN estimates the state-action value function. The combination of prediction (HPN) and decision-making (VFN) allows prediction-aware planning and improves the AVs’ ability to learn to navigate complex scenarios, and the safety prioritizer further increases safety. The proposed approach utilizes deep reinforcement learning (DRL) to achieve a high-level policy for safe tactical decision-making. As presented, the input consists of a stack of $N$ past observations and $M$ future hypotheses, i.e., $s_t = [\hat{o}_{t-N:t}, \hat{I}_{t:t+M}]$, and the 3D CNN operates as a feature extractor. The VFN is trained to learn the optimal Q-values that maximize our social reward function, optimizing social utility. During training, agents are trained in a semi-sequential manner, as in [18].

The VFN outputs the Q-values that are masked by a safety prioritizer, constraining the RL policy to a safe action space. Therefore, in our framework, when the agent policy chooses an unsafe action, the safety prioritizer masks the action and selects a safer action, saving the unsafe action ($a_t$) and the associated state in the $RM$ with a negative reward ($r_{\text{unsafe}}$). By reducing episode restarts due to potential collisions, the safety prioritizer increases sample efficiency and safety.

The proposed prediction-aware planning and the social-aware optimization algorithm is described in Algorithm 4. We first run a batch of sample simulations to pre-fill our replay buffer before starting the learning phase. To account for the unbalance in training data, the experience replay buffer is re-balanced [74].

V. EXPERIMENTAL RESULTS
This section begins with a description of the simulation environment and the HV model in mixed-autonomy traffic. Before presenting our findings, practical aspects of training and validation are explored. Finally, we present our results showing the importance of prediction-aware planning for cooperative driving.

A. Implementation Details
1) Training and Simulation Environment: In our system, we train two different models, i.e., the HPN which provides the predictions, and the VFN which learns the RL policy. To train the HPN, we create a custom dataset using simulated data by running the simulator and creating a matching dataset of history, current state, and next state, across all scenarios.
consisting of 100k training samples. Using the dataset, the HPN is trained to predict the next $V M'_{t+1}$ image and $K'_{t+1}$ state. The HPN has two branches: 1) AE that predicts the next position error ($P E$) and the Kinematic state ($K$). The HPN is trained to predict the next state. The HPN has two branches: 1) AE that predicts the next position error ($P E$) and the Kinematic state ($K$). The HPN is trained to predict the next state.

To train the VFN we use a RL simulation environment. For that, we customize the OpenAI Gym Highway environment in [76]. We design five scenarios for our experiments, i.e., a straight highway, highway exiting, highway merging, intersection, and roundabout scenarios ($f_h, f_r, f_m, f_i \in \mathcal{F}$). The AVs are trained surrounded by HVs with various behaviors, i.e., conservative, moderate, and aggressive, ($b_c, b_m, b_a \in B$). A scenario with mixed behavior is obtained by sampling from the behaviors in $B$ for each HV. The VFN is trained for $N_{episodes} = 15,000$ episodes and multiple iterations of the training procedure are carried out to guarantee the convergence of the policies. Table I lists our training and simulation parameters.

2) Computation Time: In terms of time consumption, training the VFN for $N_{episodes} = 15,000$ took around 11 hours. The training process was repeated several times to ensure the convergence of the experiments to a similar policy. For the HPN, the predictive AE was trained across 100 epochs of the shuffled dataset (100k training samples), which contained observations from all the driving scenarios, with a training time of 65h. We utilized 1,900 GPU hours for all our simulation experiments and fine-tuning (Table I).

After training, a forward pass of the VFN takes around 11ms on an NVIDIA Tesla V100 GPU. We also tested on a different GPU to compare the time consumption of the forward pass of the deep Q-network after training, and as a reference, a forward pass in a GeForce RTX 3080 took around 17ms. We present some results for different platforms in Table II.

3) Driver Modeling: We model the HV’s lateral behavior using the MOBIL model [77] and the longitudinal behavior of HVs is based on the Intelligent Driver Model (IDM) [78]. MOBIL is based on safety and incentive criteria. For safety, it verifies $a_n > -b_{safe}$, where $a_n$ is the deceleration of the following vehicle in the new lane, and $-b_{safe}$ is a safe threshold. Then MOBIL verifies the incentive to change lane measured by $a'_{ego} - a_{ego} + p((a'_n - a_n) + (a'_o - a_o)) > \Delta a_{th}$, where $p$ is the politeness term and $a_{ego}$, $a_n$ and $a_o$ are the accelerations of the ego-HV, the new following vehicle, and the old following vehicle, respectively. Finally, based on the MOBIL model, if both criteria are verified, then the HV performs a lane change.

The longitudinal behavior of HV $k$ is modeled by using the IDM model which computes the acceleration $\dot{v}_k$ as $\dot{v}_k = a_{max}\left[1 - (\frac{v}{v_{safety}})^{\delta} - \left(\frac{d^* (v, \Delta v_k)}{d_{k}}\right)^2\right]$; in which $\delta$ is the exponent of a polynomial, $d_k$ is the gap, $v_k^* is the preferred speed, and $v_k^*$ is the current speed. Following, the preferred minimum gap is computed as $d^*(v_k, \Delta v_k) = d_k^0 + v_k T_k^0 + \frac{v_k^2}{a_{des} (d_{max} - d_{des})}$, where $d_k^0$ is the minimum distance, $T_k^0$ is the safe time gap, $a_{max}$ is the acceleration limit, and $a_{des}$ is the deceleration limit.

We use the centrality metrics as in [55] and [75] to obtain the parameters $\mathcal{P}$ that simulate the driver behaviors for the MOBIL and IDM models. In our scenarios, the computed parameters $\mathcal{P}$ that represent aggressive, conservative, and moderate behavior are shown in Table III.

### Table I

| Parameter | Value | Parameter | Value |
|-----------|-------|-----------|-------|
| K prediction | OP, RBF kernel | $N_{episodes}$ | 15,000 |
| Prediction Horizon | 4s | $\epsilon$-decay | Linear |
| History window | 2s | RM buffer size | 8,000 |
| Latent Dimension | 512 | Initial exploration $t_0$ | 1.0 |
| Batch size | 64 | Final exploration | 0.05 |
| Learning rate $\alpha_0$ | 0.0005 | Optimizer | ADAM |
| $Target_{update}$ | 300 | Discount factor $\gamma$ | 0.95 |

### Table II

| Computing platform | forward pass of the VFN |
|--------------------|-------------------------|
| NVIDIA Tesla V100 GPU | 11ms |
| GeForce RTX 3080 | 17ms |
| NVIDIA Jetson AGX Xavier | 87ms |
| NVIDIA Jetson TX2 | 139ms |

### Table III

| Model | Parameter | Aggressive | Moderate | Conservative |
|-------|-----------|------------|----------|-------------|
| MOBIL | $p$ | 0 | 0.3 | 1 |
| $\Delta a_{th}$ | 0 m/s$^2$ | 0.1 m/s$^2$ | 0.4 m/s$^2$ |
| $b_n$ | 12.0 m/s$^2$ | 6.0 m/s$^2$ | 2.0 m/s$^2$ |
| IDM | $T_k^0$ | 0.5s | 1s | 3s |
| $d_k^0$ | 1 m | 2 m | 6.0 m |
| $acc_{max}$ | 7.0 m/s$^2$ | 3.0 m/s$^2$ | 1.0 m/s$^2$ |
| $acc_{des}$ | 12.0 m/s$^2$ | 7.0 m/s$^2$ | 2.0 m/s$^2$ |

B. Evaluation Metrics and Hypotheses

The system’s performance is evaluated based on safety, effectiveness, and prediction error. We select two indicators that, notwithstanding their correlation, offer distinct perspectives on the effectiveness of our approach. We calculate the proportion of episodes with at least one crash ($C(\%)$) in order to measure safety. The vehicles’ average traveled distance ($DT(m)$) is utilized to measure efficiency. Finally, the prediction error is measured in terms of the prediction reconstruction error ($PRE$) of the $V M$ predictions and in terms of position error ($PE$) for the Kinematic prediction $K$. In this paper, efficiency refers to how effectively AVs can navigate in traffic and reach their intended destinations. This is critical for achieving optimal traffic flow and minimizing congestion. The average traveled distance ($DT(m)$) in our paper is a proxy for this aspect of efficiency. If each AV is able to traverse a greater distance within a given time period, this indicates that it is able to maintain a higher average speed, implying fewer instances of slowdowns or stops due to traffic conditions. Essentially, a higher average traveled distance is indicative of smoother, less interrupted driving, which we interpret as higher efficiency.

Based on these evaluation metrics we investigate the following hypotheses:
the prediction performance. We compare different kinematic prediction baselines and measure their performance using the position error \( PE \) in meters. We compare five prediction approaches, i.e., the GP prediction approach, an LSTM network, Constant Speed (CS), Constant Acceleration (CA) based prediction, and the predictive AE kinematic prediction. GP, LSTM, and AE can be used to predict any time series, and leveraging that, we consider two approaches for each method: direct and indirect prediction. Therefore, we compare eight baselines the GP direct (GP D), the GP indirect (GP I), LSTM Direct (LSTM D), LSTM Indirect (LSTM I), CS, CA, AE direct (AE D), and AE indirect (AE I).

In a direct prediction approach, \((x, y)\) are regressed by two distinct models learned from the history \((x_{t-N:t}, y_{t-N:t})\), producing direct predictions of futures \((x, y)\). Differently, in an indirect prediction approach, the vehicle’s heading \((\phi_{t-N:t})\) and speed \((v_{t-N:t})\) histories are considered and the models for heading and speed \((\phi, v)\) are learned using the predictive approach. Using the learned models, the predictions for future heading and speed \((\phi, v)\) are computed and utilized to calculate the future position \((x_{t+1:t+M}, y_{t+1:t+M})\). In our experiments, a compound kernel of linear and RBF is used following our previous work [15].

As illustrated in Figure 5 the indirect GP (GP I, in orange) approach outperforms the other baselines in terms of \( PE \), showing how this non-parametric Bayesian scheme allows the incorporation of complex model structures and is a suitable option for our kinematic prediction method, verifying our \textbf{H1}.

\textbf{b) Observation history:} Together with the Kinematic prediction, the HPN outputs the velocity map image prediction \((VM)\). A prediction reconstruction error \((P RE)\) loss is utilized to calculate the error between the predicted observation \(o^{pred}\) and the corresponding \(o\), i.e., \(L_2(o_{i+1:t+1}^{pred}, o_{i+1}^{true}) = \sum_o |o_i - o_i^{true}|^2\). We evaluate the HPN’s performance using only the current observation as input and the history of \(N\) observations, demonstrating the importance of temporal information for accurate prediction measured by the \(P RE\). Figure 6 depicts the training loss results, where the left image is for the HPN that uses the history of \(N\) observation and the right image uses just the current observation. As shown in the figure, when using the temporal information the \(P RE\) loss is approximately 20% smaller when using history (left) than without history (right). Similarly, Figure 8 shows the qualitative output of the HPN prediction chain using the current observation or a history of observations as input. The results with history (top) show clearer and more accurate visual predictions and confirm why the \(P RE\) is lower when using the history.

Figure 7 presents some qualitative results of the internal representations at different layers of the HPN for a merging scenario. In Figure 7 (bottom), a zoomed-in version of some internal representations are illustrated for visualization. As observed, the HPN learns to extract and highlight important representations at different layers of the HPN for a merging scenario. In Figure 7 (bottom), a zoomed-in version of some internal representations are illustrated for visualization. As observed, the HPN learns to extract and highlight important representations at different layers of the HPN for a merging scenario.
Fig. 7. Internal representations of the features at different layers for a merging scenario (Top), selected internal representations that have been zoomed in (Bottom). (a,b,c,d) denote specific features learned from the input observations. For instance, (a) the internal representations here capture the structure of the lanes in the driving environment. The network has learned to identify and differentiate lanes, despite not being explicitly trained on a lane-detection task. This ability to understand lane structure is crucial for navigating traffic scenarios, as it allows the system to position itself correctly within the traffic flow and make safe lane changes; (b) The learned features here represent other vehicles or ‘agents’ on the road. The network is capable of distinguishing these entities from the background, effectively performing a form of object detection. This allows the system to keep track of other road users, anticipate their actions, and plan their movements accordingly.

2) Safer VFN Leveraging Prediction: Using the HPN and prediction chain, the VFN is trained to optimize for a social utility. We evaluate the performance of the VFN when using just the history as input, i.e., $s_t = [o_{t−N:t}]$ and when additionally utilizing the prediction output of the prediction chain, i.e., $s_t = [o_{t−N:t}, \hat{o}_{t+1:t+M}]$. Figure 9 shows performance improvement by using prediction quantified by crash percentage (Top, $C(\%)$) and average distance traveled (Bottom, $DT(m)$). The results present the AV’s performance in the highway merging scenario $f_m$, in the presence of HVs with conservative, aggressive, or mixed behavior. We observe that when training and testing are performed in a conservative environment, or in other words, when HV yields and takes safer actions, the gains from prediction capabilities are not as noticeable, whereas, in an aggressive and mixed environment in which the behavior changes, the performance increases are significant. We believe that anticipating the future is especially useful in those scenarios, which is why performance has improved.

We evaluate the effectiveness of our architecture in diverse scenarios, as well as the performance enhancement of leveraging prediction. We argue that by using prediction, we provide the VFN a prior on how the world will evolve which is helpful for decision-making. Table IV presents the results in different traffic scenarios, i.e., Exiting ($f_e$), Merging ($f_m$), Roundabout ($f_r$), intersection ($f_i$), and Highway ($f_h$) under mixed HV behaviors ($b \in B$). We compare our architecture when using prediction (VFN+P), and without prediction (VFN) with other related architectures [18], [19], [73]. Our architectures as shown in Table IV outperform the alternative methods, and the improvements are particularly pronounced in the more complex scenarios.

The combination of prediction (HPN) and decision-making (VFN) allows for prediction-aware planning and improves the AVs’ ability to learn to navigate complex scenarios, and the safety prioritizer is further improved by leveraging the
TABLE IV

| Approach     | \( f_c \) | \( f_m \) | \( f_r \) | \( f_i \) | \( f_h \) |
|--------------|-----------|-----------|-----------|-----------|-----------|
| Conv2D+DQN [73] | 24.62     | 29.12     | 49.03     | 54.78     | 17.21     |
| Conv3D+2C2 [18] | 9.24      | 14.99     | 21.17     | 36.62     | 7.43      |
| Conv3D+DQN [19] | 3.91      | 2.59      | 14.62     | 24.30     | 1.31      |
| Safe DQN [75] | 2.51      | 1.95      | 9.04      | 18.67     | 0.44      |
| VFN          | 2.47      | 1.96      | 8.94      | 17.90     | 0.39      |
| VFN + P      | 1.91      | 1.04      | 7.01      | 11.10     | 0.31      |

Fig. 10. Qualitative results of the prediction chain output for multiple scenarios.

D. Ablation Studies

In this subsection, we present our ablation experiments, which aim to highlight the contribution of each individual part (HPN, VFN, and safety prioritizer) to the overall performance. These experiments, which involve selectively omitting or modifying components, afford us a clear understanding of how each component feeds into the overall model’s success.

1) Assessing Different Kinematic Predictors: Our first ablation examines the influence of using different kinematic predictors in the HPN on the final architecture’s performance. We examine their individual performance in different HV behaviors within a merging scenario (Table V) and an intersection scenario (Table VI). The best performance is achieved with the GP I approach, particularly in the more complex interception scenario. This prompts our hypothesis that inaccurate predictions may lead the VFN to learn incorrect decisions, thus leading to more crashes. This ablation shows the individual impact of different kinematic predictors on the VFN, measured by crash percentage (C%).

2) Impact of Different Input State Representations: In the following experiment, we use GP I as our Kinematic predictor model and then ablate the individual impact of using only Velocity Map prediction vs. Kinematic prediction vs. the combination of both. We compare the results without Prediction (No Prediction), using prediction of \( VM_{t+1} \) image and \( K_{t+1}' \) state (Pred AE + K), using prediction of only \( VM_{t+1} \) image (Pred AE), and using prediction of only \( K_{t+1}' \) state (Pred K). As presented in Table VII, Pred AE + K yields a higher performance (combination of vehicle kinematic trajectory and a velocity map).

3) Role of Safety Prioritizer: Next, we ablate the importance of the safety prioritizer, gauging the performance

TABLE V

| Scenario     | Method      | Conservative | Crashed [%] | Mixed |
|--------------|-------------|--------------|-------------|-------|
| \( f_m \)   | No Prediction | 0.32         | 0.77        | 1.96  |
|              | Pred AE + K | 0.24         | 0.53        | 1.04  |
|              | Pred AE     | 0.25         | 0.61        | 1.28  |
|              | Pred K      | 0.27         | 0.69        | 1.33  |
| \( f_i \)   | No Prediction | 8.74         | 14.19       | 17.90 |
|              | Pred AE + K | 4.13         | 9.20        | 11.10 |
|              | Pred AE     | 4.31         | 9.75        | 12.33 |
|              | Pred K      | 4.60         | 10.46       | 13.68 |

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
of our system when using a safety prioritizer (SP) versus a no safety prioritizer (No SP). As observed in Table VIII, the safety prioritizer plays a significant role in reducing collisions showing a reduced number of crashed scenarios (C(\%)) when the SP is employed. The gains are especially noteworthy in more complex environments like intersections and mixed scenarios where behavior is variable.

4) Training Kinematic Prediction With Field Data: In the case of kinematic prediction, we presented a particular case when using a GP predictor trained on field data. For this test, we train the GP predictor using the SPMD [79] dataset. The SPMD dataset comprises an extensive selection of trips recorded on a variety of cities and roads. SPMD is a robust compilation of data showcasing a broad spectrum of driver/vehicle behaviors and movements. It offers various in-car data, tracked through CAN, including longitudinal velocity and acceleration, yaw rate, steering angle, turning signal status, etc [15], [80].

The results of using SPMD to train the GP model versus using the simulated data are presented in Table IX. As we can observe, the kinematic models are still valid and show reasonable results, however, as expected, the models learned on the simulation show higher performance as they were trained with data of a more similar distribution.

VI. CONCLUSION

We propose the integration of two crucial components for AVs, social navigation and prediction. The safety and reliability of AVs depend on their predictive capabilities, social awareness, and ability to engage in complex social interactions. For that reason, we propose prediction-aware planning and social-aware optimization in a cooperative RL framework, to allow safe and socially-desirable outcomes.

We provide AVs the ability to anticipate the future, allowing them to make informed decisions and proactive actions in AV-HV social interaction scenarios. The safety prioritizer leverages interpretable kinematic predictions from the HPN to restrict the RL policy to assure safe decision-making, reducing future high-risk actions, increasing awareness of the immediate risks, and consequently decreasing crashes. We compare our prediction-aware AV to other solutions and demonstrate how our approach consistently improves safety and efficiency on the road in multiple scenarios.

A. Limitations and Future Work

Further research needs to be performed using real human driver data, and more complex traffic scenarios. We plan to extend this work in this direction and show the robust generalization capabilities of our agents. We intend to explore learning meaningful and interpretable representations and predictions to help build intuition on the AV’s decision-making process.

In this work, we have used individual meta-actions for modeling the decision-making process, prioritizing simplicity and tractability. This approach eases computational demand, thus allowing for real-time operations. However, our future research will explore the use of combined actions, which could lead to a more nuanced model. We intend to further enhance the realism of our simulator by including a wider range of behaviors and more sophisticated models. This includes validation against real-world driving data and iterative refinement based on observed discrepancies.

REFERENCES

[1] A. Cosgun et al., “Towards full automated drive in urban environments: A demonstration in GoMentum station, California,” in Proc. IEEE Intell. Vehicles Symp. (IV), Jun. 2017, pp. 1811–1818.
[2] W. Schwanting, A. Pierson, J. Alonso-Mora, S. Karaman, and D. Rus, “Social behavior for autonomous vehicles,” Proc. Nat. Acad. Sci. USA, vol. 116, no. 50, pp. 24972–24978, 2019.
[3] F. Sagberg, G. F. B. Piccinini, and J. Engström, “A review of research on driving styles and road safety,” Hum. Factors, J. Hum. Factors Ergonom. Soc., vol. 57, no. 7, pp. 1248–1275, Nov. 2015.
[4] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, “Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data,” in Proc. Eur. Conf. Comput. Vis. Cham, Switzerland: Springer, 2020, pp. 683–700.
[5] S. Mozaffari, O. Y. Al-Jarrah, M. Dianati, P. Jennings, and A. Mouzakitis, “Deep learning-based vehicle behavior prediction for autonomous driving applications: A review,” IEEE Trans. Intell. Transp. Syst., vol. 23, no. 1, pp. 33–47, Jan. 2022.
[6] S. Aoki, T. Higuchi, and O. Altintas, “Cooperative perception with deep reinforcement learning for connected vehicles,” in Proc. IEEE Intell. Vehicles Symp. (IV), Oct. 2020, pp. 328–334.
[7] B. Toghi, M. Saifuddin, M. O. Mughal, and Y. P. Fallah, “Spatiotemporal dynamics of cellular V2X communication in dense vehicular networks,” in Proc. IEEE 2nd Connected Automated Vehicles Symp. (CAVS), Sep. 2019, pp. 1–5.
[8] G. Shah et al., “Real-time hardware-in-the-loop emulation framework for DSRC-based connected vehicle applications,” in Proc. IEEE 2nd Connected Automated Vehicles Symp. (CAVS), Sep. 2019, pp. 1–6.
[9] G. Shah, M. Saifuddin, Y. P. Fallah, and S. D. Gupta, “RVE-CV2X: A scalable emulation framework for real-time evaluation of CV2X-based connected vehicle applications,” in Proc. IEEE Veh. Netw. Conf. (VNC), Dec. 2020, pp. 1–8.
[10] G. Shah, S. Shahrman, Y. Fallah, D. Tian, and E. Moradi-Pari, “Enabling a cooperative driver messenger system for lane change assistance application,” in Proc. IEEE 25th Int. Conf. Intell. Transp. Syst. (ITSC), Oct. 2022, pp. 4320–4327.
A. Xie, D. Losey, R. Tolsma, C. Finn, and D. Sadigh, “Learning latent representations to influence multi-agent interaction,” in Proc. 4th Conf. Robot Learn. (CoRL), Nov. 2020, pp. 1–12.

B. Ivanovic and M. Pavone, “The trajectron: Probabilistic multi-agent trajectory modeling with dynamic spatiotemporal graphs,” in Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), Oct. 2019, pp. 2375–2384.

B. Ivanovic and M. Pavone, “Injecting planning-awareness into prediction and detection evaluation,” in Proc. IEEE Intell. Vehicles Symp. (IV), Jun. 2022, pp. 821–828.

A. Xie, D. Losey, R. Tolsma, C. Finn, and D. Sadigh, “Learning latent representations to influence multi-agent interaction,” in Proc. 4th Conf. Robot Learn. (CoRL), Nov. 2020, pp. 1–12.

H. N. Mahjoub, A. Raftari, R. Valiente, Y. P. Fallah, and S. K. Mahmud, “Representing realistic human driver behaviors using a finite size Gaussian process kernel bank,” in Proc. IEEE Veh. Netw. Conf. (VNC), Dec. 2019, pp. 1–8.

J. Rios-Torres and A. A. Malikopoulos, “A survey on the coordination of connected and automated vehicles at intersections and merging at highway on-ramps,” IEEE Trans. Intell. Transp. Syst., vol. 18, no. 5, pp. 1066–1077, May 2017.

Y. Lin, J. McPhee, and N. L. Azad, “Anti-jerk on-ramp merging using deep reinforcement learning,” in Proc. IEEE Intell. Vehicles Symp. (IV), Oct. 2020, pp. 7–14.

B. Toghi, R. Valiente, D. Sadigh, R. Pedarsani, and Y. P. Fallah, “Altruistic maneuver planning for cooperative autonomous vehicles using multi-agent advantage actor-critic,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR) Workshops, Jul. 2021, pp. 1–12.

B. Toghi, R. Valiente, D. Sadigh, R. Pedarsani, and Y. P. Fallah, “Cooperative autonomous vehicles that sympathize with human drivers,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Sep. 2021, pp. 4517–4524.

A. Pokle et al., “Deep local trajectory replanning and control for robot navigation,” in Proc. Int. Conf. Robot. Autom. (ICRA), May 2019, pp. 5815–5822.

B. Ivanovic, E. Schmerling, K. Leung, and M. Pavone, “Generative modeling of multimodal multi-human behavior,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Oct. 2018, pp. 3088–3095.

M. Kuderer, S. Gulati, and W. Burgard, “Learning driving styles for autonomous vehicles from demonstration,” in Proc. IEEE Int. Conf. Robot. Autom. (ICRA), May 2015, pp. 2641–2646.

D. Sadigh, N. Landoft, S. S. Sastry, S. A. Seshia, and A. D. Dragan, “Planning for cars that coordinate with people: Leveraging effects on human actions for planning and active information gathering over human internal state,” Auto. Robots, vol. 42, no. 7, pp. 1405–1426, Oct. 2018.

D. Hadfield-Menell, S. J. Russell, P. Abbeel, and A. Dragan, “Cooperative inverse reinforcement learning,” in Proc. Adv. Neural Inf. Process. Syst., vol. 29, 2016, pp. 3909–3917.

L. Crosato, C. Wei, E. S. L. Ho, and H. P. H. Shum, “Human-centric autonomous driving in an AV-pedestrian interactive environment using SVO,” in Proc. IEEE Int. Conf. Intell. Mach. Syst. (ICIMCS), Sep. 2021, pp. 1–6.

J. Geary, H. Gouk, and S. Ramamoorthy, “Active altruism learning and information sufficiency for autonomous driving,” 2021, arXiv:2110.04580.

K. Srivivasan, B. Eysenbach, S. Ha, J. Tan, and C. Finn, “Learning to be safe: Deep RL with a safety critic,” 2020, arXiv:2010.14603.

M. Razzaghpour, S. Mohsharafian, A. Raftari, J. M. Velni, and Y. P. Fallah, “Impact of information flow topology on safety of tightly-coupled connected and automated vehicle platoons utilizing stochastic control,” in Proc. Eur. Control Conf. (ECC), Jul. 2022, pp. 27–33.

Z. Li, U. Kalabic, and T. Chu, “Safe reinforcement learning: Learning with supervision using a constraint-admissible set,” in Proc. Ann. Amer. Control Conf. (ACC), Jun. 2018, pp. 6390–6395.

J. Wang, Q. Zhang, D. Zhao, and Y. Chen, “Lane change decision-making through deep reinforcement learning with rule-based constraints,” in Proc. Int. Joint Conf. Neural Netw. (IJCNN), Jul. 2019, pp. 1–6.

C. Hickert, S. Li, and C. Wu, “Cooperation for scalable supervision of autonomy in mixed traffic,” 2021, arXiv:2112.07569.

S. Nageshrao, H. E. Tseng, and D. Filev, “Autonomous highway driving using reinforcement learning,” in Proc. IEEE Int. Conf. Intell. Syst., Man Cybern. (SMC), Oct. 2019, pp. 2326–2331.

A. Mohammadhasani, H. Mehrivash, A. Lynch, and Z. Shu, “Reinforcement learning based safe decision making for highway autonomous driving,” 2021, arXiv:2105.06517.

D. Chen et al., “Deep multi-agent reinforcement learning for highway on-ramp merging in mixed traffic,” 2021, arXiv:2105.05701.

K. Brown, K. Driggs-Campbell, and M. J. Kochenderfer, “A taxonomy and review of algorithms for modeling and predicting human driver behavior,” 2020, arXiv:2006.08812.

A. Jami, M. Razzaghpour, H. Alinevari, and Y. P. Fallah, “Augmented driver behavior models for high-fidelity simulation study of crash detection algorithms,” 2022, arXiv:2208.05540.

M. Lauer and M. Riedmiller, “An algorithm for distributed reinforcement learning in cooperative multi-agent systems,” in Proc. 17th Int. Conf. Mach. Learn., 2000, pp. 535–542. [Online]. Available: https://dl.acm.org/doi/10.5555/645529.658113

S. Omidshafiei, J. Pazis, C. Amato, J. P. How, and J. Vian, “Deep decentralized multi-task multi-agent reinforcement learning under partial observability,” in Proc. Int. Conf. Mach. Learn., 2017, pp. 2681–2690.

Z. Constantinescu, C. Marinouin, and M. Vladoiu, “Driving style analysis using data mining techniques,” Int. J. Comput. Commun. Control, vol. 5, no. 5, pp. 654–663, Dec. 2010.

K. H. Beck, B. Ali, and S. B. Daughters, “Distress tolerance as a predictor of risky and aggressive driving,” Traffic Injury Prevention, vol. 13, no. 4, pp. 349–354, May 2014.

R. Chandra, U. Bhattacharya, T. Mittal, A. Bera, and D. Manocha, “C-Meter: A driving behavior measure using centrality functions,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS), Oct. 2020, pp. 2035–2042.
