A Game-Theoretic Framework for Joint Forecasting and Planning

Kushal Kedia[1], Prithwish Dan[1], Sanjiban Choudhury[1]

***Abstract*— Planning safe robot motions in the presence of humans requires reliable forecasts of future human motion. However, simply predicting the most likely motion from prior interactions does not guarantee safety. Such forecasts fail to model the long tail of possible events, which are rarely observed in limited datasets. On the other hand, planning for worst-case motions leads to overtly conservative behavior and a "frozen robot". Instead, we aim to learn forecasts that *predict counterfactuals that humans guard against*. We propose a novel game-theoretic framework for joint planning and forecasting with the payoff being the performance of the planner against the demonstrator, and present practical algorithms to train models in an end-to-end fashion. We demonstrate that our proposed algorithm results in safer plans in a crowd navigation simulator and real-world datasets of pedestrian motion. We release our code at `https://github.com/portal-cornell/Game-Theoretic-Forecasting-Planning`.**

I. Introduction

One of the greatest challenges in robotics and AI is reasoning about interaction with other agents in the world. The ability to forecast how other agents behave in response to a robot's decisions is key to enabling safe, interpretable, and responsive behavior. Consider a self-driving car negotiating a left turn at a busy intersection, or a personal robot collaboratively cooking with a human in a kitchen. The robot has to both yield to the human at times, and show intent to go ahead at other times. To do so, it must rely on forecasts that are conservative enough to predict rare but risky events, but not so conservative that the robot stays frozen in place.

Current forecasting approaches are primarily based on Maximum Likelihood Estimate (MLE). For instance, in self-driving, state-of-the-art forecasters [1], [2] are typically trained on the L2 loss between the observed future motions of actors and the predicted motion on data collected off-policy. A planner then uses the forecast to compute a safe, collision-free path. However, a forecaster trained purely on observed data **fails to predict rare but risky events**. The distribution of motions exhibits a long tail, necessitating the modeling of this tail with exorbitant amounts of data to accurately represent the diverse rare events.

Consider the example in Fig. 1 of a self-driving car driving alongside a cyclist. We observe humans leave their lane to give space to a cyclist while actively occupying the lane of an oncoming car. However, an MLE forecaster will likely predict the cyclist staying in their lane. This results in plans that fail to guard against possible rare events such as the cyclist accidentally coming into the lane. An alternate approach is to reason about the worst-case outcome

[1]The authors are affiliated with Cornell University, Ithaca, New York, USA {kk837, pd337, sc2582}@cornell.edu

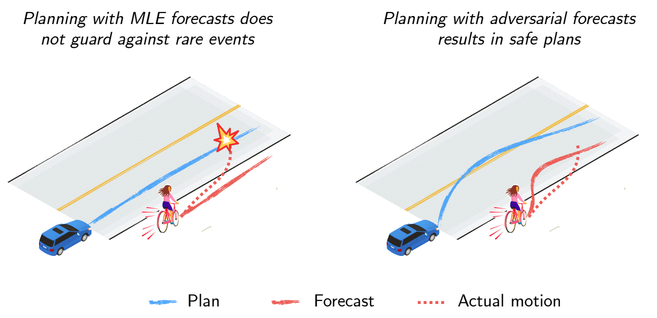

Planning with MLE forecasts does not guard against rare events
Planning with adversarial forecasts results in safe plans
Plan
Forecast
Actual motion

Fig. 1: MLE forecasts fail to predict rare, hazardous events, like a bicyclist suddenly veering into a car's lane. We propose to learn adversarial forecasts that enable a planner to guard against such events.

given the reachability of the cyclist [3], [4]. But this can lead to overtly conservative behavior where the robot stays perpetually behind the cyclist.

We propose a new objective for training forecasters. We argue that MLE loss on observed motions from a finite dataset is fundamentally insufficient due to lack of coverage in the dataset. Instead, we view the problem through the lens of imitation learning. Our key insight is that *humans don't just plan for things that are likely to happen, but plan contingencies for counterfactuals that could possibly happen*. We aim to learn forecasts that enable a planner to guard as well as the demonstrator against any possible rare event. We propose a game-theoretic framework for jointly training planners and forecasts, and use no-regret learning to solve for the approximate equilibrium of the game. Our key contributions are summarized as follows:

1) A novel, game-theoretic framework for joint forecasting and planning that guarantees performance with respect to demonstrations.
2) Practical algorithms and architectures for joint forecasting and planning for multi-agent navigation.
3) Empirical evaluation on a crowd navigation simulator and real-world pedestrian datasets.

II. Related Work

We focus on the problem of planning for robot decisions in the presence of humans in the workspace. The intended motions of the humans are unknown to the robot as it plans a sequence of actions that maximize the reward function for a given task. We look at various clusters of related work.

Multi-agent games Multi-agent games are formulated as both Stackelberg or Nash Equilibrium finding problems. [5] models human action as a function of the robot's action and the system's current state. [6] extends this model to

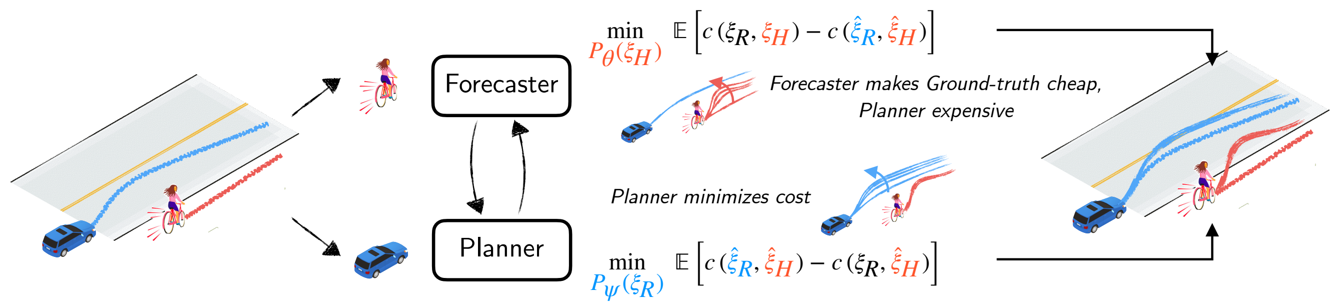

Forecaster
Planner
$\min_{P_\theta(\xi_H)} \mathbb{E}\left[c(\xi_R,\xi_H) - c(\hat{\xi}_R,\hat{\xi}_H)\right]$
Forecaster makes Ground-truth cheap, Planner expensive
Planner minimizes cost
$\min_{P_\psi(\xi_R)} \mathbb{E}\left[c(\hat{\xi}_R,\hat{\xi}_H) - c(\xi_R,\hat{\xi}_H)\right]$

Fig. 2: Overview of our game-theoretic framework for joint forecasting and planning. The forecaster maximizes the performance difference between the generated plans and the observed plans. This results in counterfactual forecasts for the cyclist veering into the vehicle's lane that encourages the planner to guard by nudging away from the cyclist.

distinguish between different types of human actors (attentive and distracted). [7] solves for the Global Nash Equilibrium by using the Hamilton-Jacobi-Bellman equation of optimal control. Trajectory optimization utilized by [8] includes a sensitivity term that allows the vehicle to reason how much the other vehicles will yield to avoid collisions. To capture the problem's constraints effectively, [9] enforces collision-avoidance through augmented Lagrangian constraints instead of imposing a large penalty on collision while constructing the objective functions for the problems. However, unlike our work, in order to solve these games, full information about the cost for each agent is required.

Autonomous Driving In the autonomous driving domain, this problem is broken down into game-theoretic interactions to represent lane-merging, intersection crossing, pedestrian management, etc. Works in this domain have modeled the game by a Stackelberg equilibrium [10], [6], [5] where the behavior of a leader is fixed and best-response strategy is learned for the follower, or by a Nash equilibrium [9], [7], [11], [8], where agents follow strategies such that their objectives cannot be improved upon unilateral deviation. In this paper, we focus on the unstructured domain of pedestrian navigation, where there is large variability in human behavior and the space of possible motions is larger.

Forecasting Human Motion Real-world human movements constitute a broad multimodal distribution containing inherent uncertainty and noise. Prior works have focused on effective model architectures and appropriate representations of interactions between agents. Human-robot and human-human interactions can be effectively modeled with a self-attention mechanism [12] for robot planning. Multimodal deep generative models [13] for trajectory forecasting can effectively represent diverse human behavior. Trajectron++ [14] produces dynamically-feasible predictions by incorporating dynamics constraints into learned multi-agent trajectory forecasting. Representation learning [15] of safer motion representations can be facilitated by contrastive estimation from simulated negative behavior. The problem of transfer-learning forecasting models from different datasets has been tackled by explicitly modeling styles and noise confounders [16]. However, the focus of forecasting literature has been largely restricted to task-agnostic accuracy-based metrics such as average displacement error (ADE) and final displacement error (FDE). For instance, in self-driving, it is important to use task-specific metrics prioritizing the safety of a planner that consumes the forecasts [17]. This has motivated the community to rethink appropriate metrics for planner-centric evaluation of forecasting [18]. In this work, we are ultimately interested in generating forecasts that optimize the final performance of our downstream planner.

III. Problem Formulation

We model the forecasting problem as a multi-agent Contextual Markov Decision Process (CMDP), where one of the agents is the robot, and the rest are humans. Let ϕ denote the context – a history of past states-actions for all the agents and the current scene. We assume contexts are drawn from a distribution $P(\phi)$. Let $\xi_R = (s^R_1, a^R_1, s^R_2, a^R_2, \dots, s^R_T, a^R_T)$ be the T-horizon trajectory of the robot. Let $\xi^H = (s^H_1, a^H_1, s^H_2, a^H_2, \dots, s^H_T, a^H_T)$ be the trajectory of all the other human agents. Let $c(\xi^R, \xi^H)$ denote the cost of a robot-human trajectory pair. This captures terms like safety, burden and deviation from the nominal path.

We assume access to demonstrations of human and robot trajectories drawn from a joint probability $P(\xi_R, \xi_H|\phi)$. We aim to learn a conditional distribution over robot trajectories $P_\psi(\xi_R|\xi_H,\phi)$, where ψ are the learnt parameters, that minimize the average performance difference with respect to the demonstrator.

$$\mathbb{E}_\phi\left[\mathop{\mathbb{E}}_{\substack{\xi_H\sim P(.|\phi)\\ \hat{\xi}_R\sim P_\psi(.|\xi_H,\phi)}} c(\hat{\xi}_R,\xi_H) - \mathop{\mathbb{E}}_{\xi_H,\xi_R\sim P(.|\phi)} c(\xi_R,\xi_H)\right] \tag{1}$$

A. Pitfalls of Planning with MLE forecast

A template for solving Eq.1 is to first train a *forecaster* to approximate the human trajectory distribution $P_\theta(\xi_H|\phi) \approx P(\xi_H|\phi)$, where θ are learnt parameters. A common way is to train a Maximum Likelihood Estimator (MLE).

*Definition 1 (*MLE-Forecaster*):* Given a dataset $\mathscr{D} = \{(\phi, \xi_H)\}$ of context and human trajectories, the goal of the MLE forecaster is to maximize the likelihood of observed trajectories:

$$\max_\theta \mathbb{E}_{\phi,\xi_H} \log P_\theta(\xi_H|\phi) \tag{2}$$

A nominal approach to planning is to minimize cost with respect to this learnt forecast.

*Definition 2 (*NOM-PLANNER*):* Given access to a MLE-FORECASTER $P_\theta(\xi_H|\phi)$, a nominal planner minimizes expected costs w.r.t. the forecasts

$$\min_{\psi} \mathbb{E}_\phi \underset{\substack{\hat{\xi}_H \sim P_\theta(.|\phi) \\ \hat{\xi}_R \sim P_\psi(.|\hat{\xi}_H,\phi)}}{\mathbb{E}} c(\hat{\xi}_R, \hat{\xi}_H) \tag{3}$$

While MLE-FORECASTER+NOM-PLANNER is industry standard, the framework has two fundamental problems: wide, labelwidth=!, labelindent=0pt

1) *Failure to predict rare but risky events:* The MLE loss is dominated by events that occur frequently in the data. It fails to predict events ξ_H that occur rarely in the data. However, these events can be quite risky, i.e., even if $P(\xi_H|\phi)$ is small, the cost $c(\xi_R, \xi_H)$ of the planner can be very large.
2) *Small forecasting errors lead to large planning errors*: The MLE loss optimizes the KL-Divergence $KL(P(\xi_H|\phi)||P_\theta(\xi_H|\phi))$, which is mismatched from the performance difference in (1). Formally, a bounded KL divergence, implies a Total Variation (TV) distance bound of ε (Psinker's inequality). However, a small error in forecasting could result in an approximation error of $C_{max}\varepsilon$ in the downstream planner's performance, where C_{max} is the maximum cost of a trajectory.

IV. APPROACH

We present a novel game-theoretic framework for joint forecasting and planning. We also present a concrete application of the framework in a multi-agent navigation setting.

A. Game-Theoretic Framework for Forecasting / Planning

In Section III-A, we discussed two fundamental problems with the MLE approach: (1) failure to predict rare-events (2) loss mismatched with performance difference (Eq.1). We now propose an approach that addresses both problems. Our key insight is that *humans don't just plan for things that are likely to happen, but plan contingencies for counterfactuals that could possibly happen*. For example, in Fig. 1, the human plans a path ξ_R that guards for the counterfactual that the bicyclist may accidentally veer onto their lane. We aim to learn forecasts that *don't just predict likely motions, but predict counterfactuals that humans guard against*.

We view the problem from the lens of inverse optimal control (IOC) [19]. IOC aims to learn a cost function that explains demonstrated behavior. Here, we aim to learn a forecast (that in turn defines the cost function) that explains demonstrated behavior. IOC in this setting can be best understood as a two-player zero-sum game [20] between a forecaster and a planner. The forecaster generates forecasts $\hat{\xi}^H$ that *increase* the performance difference between the planner and the demonstrator. The planner generates plans $\hat{\xi}_R$ that *decrease* the performance difference. Finally, to ensure that the forecasts are not completely unrealistic, we constrain them to be within an δ-ball of the observed distribution.

```
Algorithm 1: ADV-FORECASTER / SAFE-PLANNER
  Input  : Dataset $\mathscr{D} = \{(\phi, \xi_R, \xi_H)\}$
  Output: ADV-FORECASTER $P_\theta(.|\phi)$,
          SAFE-PLANNER $P_\psi(.|\phi)$
1 Initialize $\theta_1$ with MLE-FORECASTER
2 Initialize $\psi_1$ with NOM-PLANNER
3 for $i = 1 \dots N$ do
4     Invoke current forecaster $P_{\theta_i}$ and planner $P_{\psi_i}$ on
      $\mathscr{D}$ to create a dataset $\{(\phi, \xi_H, \xi_R, \hat{\xi}_H^i, \hat{\xi}_R^i)\}$
5     Update planner $\psi_{i+1} \leftarrow \psi_i - \nabla_\psi \ell^i(P_\psi)$
6     Update forecaster $\theta_{i+1} \leftarrow \theta_i - \nabla_\theta \ell^i(P_\theta)$
7 return $P_{\theta_N}(.|\phi)$, $P_{\psi_N}(.|\phi)$
```

$$\max_\theta \min_\psi \mathbb{E}_\phi \left[\underset{\substack{\hat{\xi}_H \sim P_\theta(.|\phi) \\ \hat{\xi}_R \sim P_\psi(.|\hat{\xi}_H,\phi)}}{\mathbb{E}} c(\hat{\xi}_R, \hat{\xi}_H) - \underset{\substack{\hat{\xi}_H \sim P_\theta(.|\phi) \\ \xi_R \sim P(.|\phi)}}{\mathbb{E}} c(\xi_R, \hat{\xi}_H) \right]$$
$$\text{s.t. } \mathbb{E}_\phi[KL(P_\theta(\hat{\xi}_H|\phi) \;||\; P(\xi_H|\phi))] \leq \delta \tag{4}$$

We aim to compute a (near-optimal) $\varepsilon-$ equilibria of the game above, which would result in a planner P_ψ that bounds the original objective (1) by ε as well. Following the arguments in [20], since the game is bilinear in both P_θ and P_ψ, playing a no-regret strategy for both forecaster and the planner guarantees finding the $\varepsilon-$ equilibria.

We define the forecaster trained in this adversarial fashion as ADV-FORECASTER, and the planner trained to be robust against such an adversarial forecaster as SAFE-PLANNER. We describe the overall approach in Algorithm 1. We setup an online learning game that lasts N rounds. In every round, both the forecaster and the planner receive a loss function and play a no-regret update (we use online gradient descent). We define the loss functions for both players below:

*Definition 3 (*ADV-FORECASTER*):* At round i, the forecaster receives a dataset $\{(\phi, \xi_H, \xi_R, \hat{\xi}_R^i)\}$ of context, human demonstration, robot demonstration, and the planned trajectory, respectively. We define the loss for this round $\ell^i(P_\theta)$ as :

$$\ell^i(P_\theta) = \underset{\phi,\xi_H,\xi_R,\hat{\xi}_R^i}{\mathbb{E}} \left[\underset{\hat{\xi}_H \sim P_\theta(.|\phi)}{\mathbb{E}} \left[c(\xi_R, \hat{\xi}_H) - c(\hat{\xi}_R^i, \hat{\xi}_H) \right] - \lambda \log P_\theta(\xi_H|\phi) \right] \tag{5}$$

where the first term is the difference between the costs of the demonstrated and the planned robot trajectory, and the second term is the log-likelihood of observed human trajectories multiplied by a Lagrange multiplier.

*Definition 4 (*SAFE-PLANNER*):* At round i, the planner receives a dataset $\{(\phi, \xi_R, \hat{\xi}_H^i)\}$ of context, robot demonstration and the adversarial forecast respectively. We define the loss for this round $\ell^i(P_\psi)$ as :

$$\ell^i(P_\psi) = \underset{\phi,\xi_R,\hat{\xi}_H^i}{\mathbb{E}} \left[\underset{\hat{\xi}_R \sim P_\psi(.|\hat{\xi}_H^i,\phi)}{\mathbb{E}} \left[c(\hat{\xi}_R, \hat{\xi}_H^i) - c(\xi_R, \hat{\xi}_H^i) \right] \right] \tag{6}$$

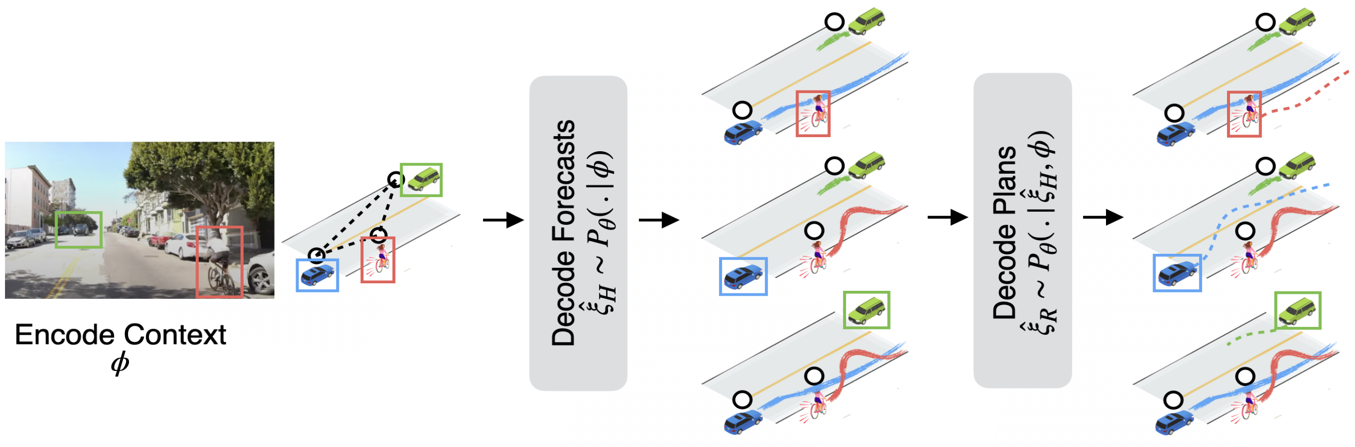

Encode Context
ϕ
Decode Forecasts
$\hat{\xi}_H \sim P_\theta(\,.\,|\,\phi)$
Decode Plans
$\hat{\xi}_R \sim P_\theta(\,.\,|\,\hat{\xi}_H, \phi)$

Fig. 3: **Model architecture for joint forecasting and planning.** Agents in the scene are represented by nodes in the graph. Each agent has two outputs: a forecast and a plan. We apply first self-attention over the encoded contexts for each node which are decoded into adversarial forecasts, which are then used by the planner to generate safe plans for each agent.

where the inner most term is the difference between costs of the planned and the demonstrated robot trajectory against the current forecast.

B. Application for Multi-Agent Navigation

We define a multi-agent navigation scene to contain N agents interacting with each other. We can consider each agent, in turn, to act as the "robot" interacting with the other "humans" in the scene. For every agent, we have to plan a T-horizon trajectory, considering the future motions of the other agents in the scene. We need to encode goal-reaching and collision avoidance to define a cost function for the navigation problem. However, while forecasting motions for agents, we do not always have access to their intended goal locations. From a dataset of prior interactions between agents, given the context of the scene, we can infer the future motions using maximum-likelihood estimation. Additionally, we enforce collision avoidance using the following obstacle cost function [21] between plans and forecasts:

$$c(\xi_R, \xi_H) = \sum_i^T COL(s_i^R, s_i^H)$$

$$COL(s_i^R, s_i^H) = \begin{cases} -dist(s_i^R, s_i^H) + \frac{1}{2}\varepsilon, & dist(s_i^R, s_i^H) < 0 \\ \frac{1}{2\varepsilon}(dist(s_i^R, s_i^H) - \varepsilon)^2 & 0 < dist(s_i^R, s_i^H) < \varepsilon \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

We sum the cost over all robot and human states in the predicted T-horizon planner and forecaster trajectories. When a robot interacts with multiple humans, the human trajectory with the largest cost is considered.

In our model architecture (Fig. 3), each agent in the scene is represented by a node in a graph. Neighboring agents are connected by edges. Each node in the graph takes in its individual context, including state history and other relevant information, such as a local map representation of the scene. To encode interactions with other agents, self-attention is applied across neighboring nodes. Each node has two output heads: a forecaster and a planner. The predicted forecasts can be fed as input to the planning module. While the inputs to the forecaster are restricted to the shared context of the scene, the planner can additionally take in information private to an agent, such as its goal location.

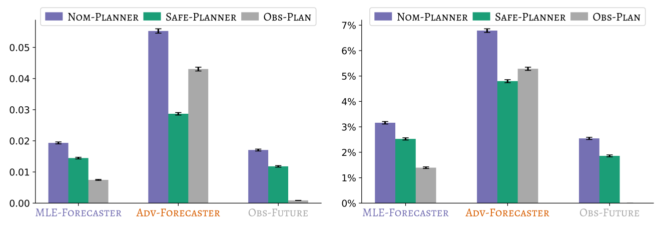
We first train the forecasting and planning model to only maximize the likelihood of the future motion for the agents in the scene (Eq. 2 and Eq. 3). This is equivalent to simply maximizing the likelihood of the future motions in the dataset, and we call it a **MLE-FORECASTER** and **NOM-PLANNER**. Apart from matching the ground truth, we wish to encode collision avoidance for measuring our plan's performance with respect to the forecasts. To incorporate this cost function, we solve the min-max game defined by Eq. 4. For every minibatch of data, we update the forecasting model (**ADV-FORECASTER**) to adversarially maximize the difference of the cost functions between the planner and the ground truth. In response, the planner (**SAFE-PLANNER**) is updated to minimize the costs with the ADV-FORECASTER. We ensure that the predictions do not deviate too much from the ground truth data by continuing to optimize the likelihood of the observed future motion.

V. EVALUATION

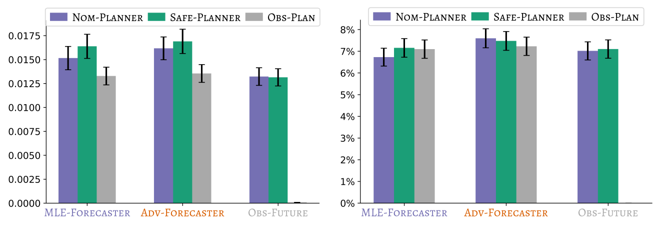
A. Setup

We evaluate our algorithm on a crowd navigation simulator and real-world pedestrian datasets.

1) **CROWDNAV** *[22]:* This is an open-source simulator where a robot has to move forward in the presence of other humans. The humans in the scene move toward their respective goal locations and are simulated using the ORCA [23] algorithm. We are interested in the non-compliant setting of the simulator, in which the five humans in the scene are unresponsive, i.e., they ignore robot motion. To navigate the scene, the robot should be able to plan around the future movements of the humans. To generate expert trajectories, we utilize the reinforcement learning (RL) agent provided by [22], which uses a self-attention (SA) module to encode human-human and human-robot interactions. The



(a) **CrowdNav** - CHOMP Cost (left) and Collision Rates (right)



(b) **ETH-UCY** - CHOMP Cost (left) and Collision Rates (right)

Fig. 4: Evaluation of CHOMP costs (Eq 7) and collision rates for different planner and forecaster combinations.

SARL agent is trained using a reward function that manually encodes collision avoidance and goal-reaching behavior.

We collect a dataset of 5000 episodes of human-robot navigation using this SARL policy. Our models are trained on 50% of the dataset and evaluated on the rest. While the SARL model architecture considers just the current state of the robot and humans to predict the robot's immediate action, we also use an LSTM-module to encode a history of 8 timesteps (2 seconds) for each agent. The predictions produced by the forecaster are given as input to the planning module along with the context and goal location of the robot. We output actions for each agent over a horizon of 8 timesteps.

*2) **The* ETH-UCY *Benchmark**:* There are five different datasets of real-world pedestrian movements in the ETH [24] and UCY [25] benchmark. The scenarios in the dataset showcase a wide range of human-human interactions and are a standard benchmark in the field. The data is captured at a 2.5Hz frequency (0.4s timestep). For the forecasting task, 8 timesteps of the history (3.2s) are considered, and 12 timesteps of the future are to be predicted for each agent. For evaluation, our model is trained on 4 out of the 5 datasets and evaluated on the held-out dataset.

We implement the Trajectron++ [14] model for the base configuration of our planner and forecaster. It is a state-of-the-art multimodal conditional variational autoencoder (CVAE) generative model that can produce dynamically feasible trajectories. While the original model produces a distribution of trajectories, we use deterministic forecasts for each agent for simplicity. To do this, we restrict the outputs of the model to a unimodal distribution for each agent's future motion. To calculate the collision avoidance cost function, we use the mean of this distribution.

B. Results and Analysis

***O1.* Adv-Forecaster predicts more severe hazards than MLE-Forecaster.** In Fig. 5, we show examples of forecasts produced by the Adv-Forecaster that leads to collisions with the plans generated by the Nom-Planner. This is expected as the Adv-Forecaster is trained to increase the cost difference between generated plans and the observed trajectories in the dataset. On the other hand, MLE-Forecaster maximizes likelihood of observed motions and is unable to generate potential hazards that render the generated plans unsafe. Fig 4 shows that the cost (Eq 7) of plans is significantly higher when evaluated with the adversarial forecasts compared with the MLE-Forecasts or

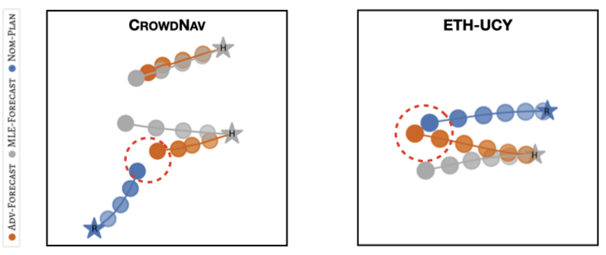


Fig. 5: The MLE-Forecaster predicts the most likely futures for each human. Nom-Planner avoids collisions with the MLE-Forecaster but not with the Adv-Forecaster. Collisions are marked in red.

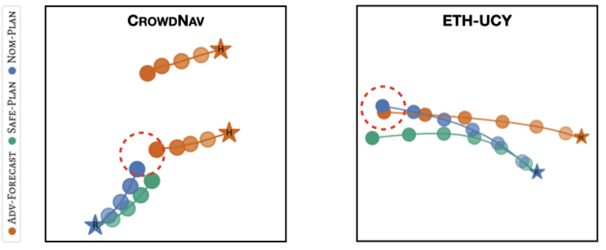


Fig. 6: The Safe-Planner plans around the Adv-Forecasts leading to safe motions, whereas the Nom-Planner collides with the adversarial forecasts. Collisions are marked in red.

the observed futures in the CrowdNav environment and the ETH-UCY benchmark.

***O2.* Safe-Planner guards against rare events better than Nom-Planner.** Fig 6 shows scenarios where the Nom-Planner is in collision with the forecasts produced by the Adv-Forecaster as it does not consider the possibility of adverse events. Since the Safe-Planner is trained to minimize the cost difference with the adversarial forecasts, its plans are safe with respect to the Adv-Forecaster. It naturally encodes behavior that guards against rare events in the dataset. Fig 4 shows that the Safe-Planner has lower costs than the Nom-Planner when evaluated against the observed futures of humans. We also observe lower costs and collision rates for the Safe-Planner compared to the Nom-Planner when tested against MLE-Forecasts and the Adv-Forecaster on the CrowdNav simulator.

***O3.* Safe-Planner and Adv-Forecaster produce plausible trajectories even with higher tracking errors.** Both the Safe-Planner and Adv-Forecaster are trained with the primary objective of optimizing cost difference. But to do so, they have to deviate from ground-truth observations. Table I shows that their average displacement error (ADE) and final displacement error (FDE) is

TABLE I: We compare the Average Displacement Error (ADE) and Final Displacement Error (FDE) of the predictions motions by our different planners/forecasters on the testing splits of the ETH-UCY benchmark and the CROWDNAV simulator.

	MLE-FORECASTER		ADV-FORECASTER		NOM-PLANNER		SAFE-PLANNER	
	ADE	FDE	ADE	FDE	ADE	FDE	ADE	FDE
ETH-UCY	0.387	0.947	0.405	0.950	0.387	0.947	0.391	0.956
CROWDNAV	0.268	0.371	0.274	0.383	0.184	0.268	0.193	0.283

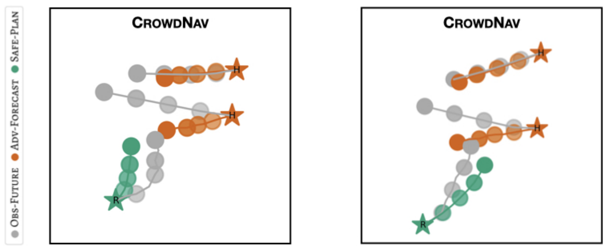


Fig. 7: (**CROWDNAV**) When ADV-FORECASTER and SAFE-PLANNER deviate from the ground truth predictions, they produce *alternate plausible trajectories*. The forecasts produced by the ADV-FORECAST represent risky futures. The SAFE-PLANNER conservatively guards against possible rare events.

slightly higher. However, we observe that the trajectories generated by the models are generally plausible. Fig. 7 shows scenarios in the CROWDNAV simulator where they both deviate from ground truth trajectories but are still quite plausible counterfactuals that the robot should guard against.

VI. DISCUSSION AND LIMITATIONS

This paper introduces a novel game-theoretic framework that addresses joint forecasting and planning. We discuss the **pitfalls of MLE forecasting** that only focus on maximizing the likelihood of observed human motion. Instead, we produce **adversarial counterfactuals** by optimizing the performance difference between generated plans and observed demonstrations, considering the predictions made by our learned forecaster. In response, our framework can **guard against rare but risky events** by generating plans that are safe with respect to the adversary.

There are some limitations to our approach. We observed larger error ranges on the ETH-UCY dataset. This is likely because real-world pedestrian datasets contain significant noise in estimation and a wide variety of behaviors, making it difficult to model human behavior accurately. In future work, we will extend our framework to consider multi-modal distributions of plans and forecasts. On-policy evaluation of our framework in scenarios where humans suddenly change their goals is another promising direction.

VII. ACKNOWLEDGEMENTS

This work was partially funded by NSF RI (#2312956).

REFERENCES

[1] J. Ngiam, B. Caine, V. Vasudevan, Z. Zhang, H.-T. L. Chiang, J. Ling, R. Roelofs, A. Bewley, C. Liu, A. Venugopal *et al.*, "Scene transformer: A unified multi-task model for behavior prediction and planning," *arXiv e-prints*, pp. arXiv–2106, 2021.

[2] L. L. Li, B. Yang, M. Liang, W. Zeng, M. Ren, S. Segal, and R. Urtasun, "End-to-end contextual perception and prediction with interaction transformer," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020.

[3] A. Bajcsy, S. Bansal, E. Ratner, C. J. Tomlin, and A. D. Dragan, "A robust control framework for human motion prediction," *IEEE Robotics and Automation Letters*, vol. 6, no. 1, pp. 24–31, 2020.

[4] K. Leung, A. Bajcsy, E. Schmerling, and M. Pavone, "Towards data-driven synthesis of autonomous vehicle safety concepts," *arXiv preprint arXiv:2107.14412*, 2021.

[5] D. Sadigh, S. S. Sastry, S. A. Seshia, and A. D. Dragan, "Planning for autonomous cars that leverage effects on human actions," in *Robotics: Science and Systems*, 2016.

[6] D. Sadigh, S. S. Sastry, A. Seshia, and A. D. Dragan, "Information gathering actions over human internal state," *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 66–73, 2016.

[7] D. Fridovich-Keil, E. Ratner, L. Peters, A. D. Dragan, and C. J. Tomlin, "Efficient iterative linear-quadratic approximations for nonlinear multi-player general-sum differential games," *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1475–1481, 2020.

[8] M. Wang, Z. Wang, J. Talbot, J. C. Gerdes, and M. Schwager, "Game-theoretic planning for self-driving cars in multivehicle competitive scenarios," *IEEE Transactions on Robotics*, vol. 37, pp. 1313–1325, 2021.

[9] S. L. Cleac'h, M. Schwager, and Z. Manchester, "Algames: a fast augmented lagrangian solver for constrained dynamic games," *Autonomous Robots*, vol. 46, pp. 201–215, 2022.

[10] A. Liniger and J. Lygeros, "A noncooperative game approach to autonomous racing," *IEEE Transactions on Control Systems Technology*, vol. 28, pp. 884–897, 2020.

[11] L. Peters, D. Fridovich-Keil, V. Rubies-Royo, C. J. Tomlin, and C. Stachniss, "Inferring objectives in continuous dynamic games from noise-corrupted partial state observations," *ArXiv*, vol. abs/2106.03611, 2021.

[12] C. Chen, Y. Liu, S. Kreiss, and A. Alahi, "Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning," *2019 International Conference on Robotics and Automation (ICRA)*, pp. 6015–6022, 2018.

[13] B. Ivanovic, K. Leung, E. Schmerling, and M. Pavone, "Multimodal deep generative models for trajectory prediction: A conditional variational autoencoder approach," *IEEE Robotics and Automation Letters*, vol. 6, pp. 295–302, 2020.

[14] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data," in *European Conference on Computer Vision*, 2020.

[15] Y. Liu, Q. Yan, and A. Alahi, "Social nce: Contrastive learning of socially-aware motion representations," *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 15 098–15 109, 2020.

[16] Y. Liu, R. Cadei, J. Schweizer, S. Bahmani, and A. Alahi, "Towards robust and adaptive motion forecasting: A causal representation perspective," *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17 060–17 071, 2021.

[17] J. Philion, A. Kar, and S. Fidler, "Learning to evaluate perception models using planner-centric metrics," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[18] B. Ivanovic and M. Pavone, "Rethinking trajectory forecasting evaluation," *ArXiv*, vol. abs/2107.10297, 2021.

[19] B. D. Ziebart, A. L. Maas, J. A. Bagnell, and A. K. Dey, "Maximum entropy inverse reinforcement learning," in *AAAI*, 2008.

[20] G. Swamy, S. Choudhury, J. A. Bagnell, and S. Wu, "Of moments and matching: A game-theoretic framework for closing the imitation gap," in *International Conference on Machine Learning*. PMLR, 2021, pp. 10 022–10 032.

[21] M. Zucker, N. D. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa, "Chomp: Covariant hamiltonian optimization for motion planning," *The International Journal of Robotics Research*, vol. 32, pp. 1164 – 1193, 2013.

[22] C. Chen, Y. Liu, S. Kreiss, and A. Alahi, "Crowd-robot interaction: Crowd-aware robot navigation with attention-based deep reinforcement learning," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 6015–6022.

[23] J. P. van den Berg, S. J. Guy, M. C. Lin, and D. Manocha, "Reciprocal n-body collision avoidance," in *International Symposium of Robotics Research*, 2011.

[24] S. Pellegrini, A. Ess, K. Schindler, and L. V. Gool, "You'll never walk alone: Modeling social behavior for multi-target tracking," *2009 IEEE 12th International Conference on Computer Vision*, 2009.

[25] A. Lerner, Y. Chrysanthou, and D. Lischinski, “Crowds by example,” *Computer Graphics Forum*, vol. 26, 2007.