Interaction-aware Multi-agent Tracking and Probabilistic Behavior Prediction via Adversarial Learning

Jiachen Li
University of California, Berkeley
21 PUBLICATIONS **153** CITATIONS

Hengbo Ma
University of California, Berkeley
13 PUBLICATIONS **61** CITATIONS

Masayoshi Tomizuka
University of California, Berkeley
966 PUBLICATIONS **22,282** CITATIONS

Interaction-aware Multi-agent Tracking and Probabilistic Behavior Prediction via Adversarial Learning

Jiachen Li*, Hengbo Ma* and Masayoshi Tomizuka

Abstract— **In order to enable high-quality decision making and motion planning of intelligent systems such as robotics and autonomous vehicles, accurate probabilistic predictions for surrounding interactive objects is a crucial prerequisite. Although many research studies have been devoted to making predictions on a single entity, it remains an open challenge to forecast future behaviors for multiple interactive agents simultaneously. In this work, we take advantage of the Generative Adversarial Network (GAN) due to its capability of distribution learning and propose a generic multi-agent probabilistic prediction and tracking framework which takes the interactions among multiple entities into account, in which all the entities are treated as a whole. However, since GAN is very hard to train, we make an empirical research and present the relationship between training performance and hyperparameter values with a numerical case study. The results imply that the proposed model can capture both the mean, variance and multi-modalities of the groundtruth distribution. Moreover, we apply the proposed approach to a real-world task of vehicle behavior prediction to demonstrate its effectiveness and accuracy. The results illustrate that the proposed model trained by adversarial learning can achieve a better prediction performance than other state-of-the-art models trained by traditional supervised learning which maximizes the data likelihood. The well-trained model can also be utilized as an implicit proposal distribution for particle filtered based Bayesian state estimation.**

I. INTRODUCTION

Accurate real-time state estimation and behavior prediction of surrounding objects play a crucial role in intelligent systems (e.g., industrial robotics and autonomous vehicles) since they serve as a premise for appropriate decision making, trajectory planning and motion control. Traditional approaches based on state space dynamic systems such as extended Kalman filter or based on time-series analysis work well in simple scenarios where the tracked or predicted targets behave independently. However, in highly interactive situations it is desired to take into account the inherent uncertainty of future and interactions among different entities since the future behavior of one agent depends heavily on others' behaviors. Moreover, even with the same environment setting, the agent may make different decisions and behave diversely. In order to capture these characters of prediction, learning based models with large capacity are necessary.

There have been numerous works focusing on tracking and forecasting future behaviors and motions of humans considering mutual reactions. In [1]–[4], the social behaviors among a group of pedestrians in crowded contexts were modeled by deep neural networks. Many recent studies have also been devoted to predicting driver behaviors or future trajectories of autonomous vehicles with various models such as intelligent driver model [5], multi-layer perceptron [6] and recurrent neural network [7], [8] which are deterministic models as well as mixture model [9]–[11], probabilistic graphical models [6], [11], [12] and variational auto-encoder [13] which are probabilistic models. There are also studies combining adversarial learning with behavior cloning or reinforcement learning. In [14], Alex et al. extended Generative Adversarial Imitation Learning (GAIL) framework proposed in [15] to train an action policy network for autonomous vehicles in which only the actions of the ego vehicle are modeled. Instead of modeling each agent individually like above studies, in this work we treat multiple interactive agents as a whole system and model the joint distribution of their future behaviors. Moreover, unlike [14] where the state-action trajectories were modeled as the output of generator, our method models the distribution of actions at current time step given a sequence of historical states.

The contributions of this work are summarized as follows: First, we propose a generic framework of adversarial learning which is utilized for probabilistic prediction of multi-agent behaviors. The framework can be generalized to any time-series prediction tasks. Second, we make an empirical study to find the relationship between the regularization coefficient of consensus optimization [16] and training performance as well as provide a guideline for choosing proper hyperparameters. Third, we propose to employ the proposed model as an implicit proposal distribution in Bayesian state estimation to enhance tracking performance. Finally, we apply the proposed approach to a task of vehicle behavior prediction.

The remainder of the paper is organized as follows. Section II provides a brief overview on related studies about GAN and Bayesian state estimation. Section III presents the details of proposed methodology. In Section IV, a numerical case study is demonstrated. In Section V, the proposed approach is applied to a real-world vehicle behavior prediction problem. Finally, Section VI concludes the paper.

II. RELATED WORK

Generative Adversarial Network

The core idea of training a deep generative model by making it compete with an opponent discriminator model was proposed in [17] where a two-player minimax game is formulated. A straightforward extension of incorporating condition information into both generator and discriminator was put forward in [18] in which the generator can produce

*The authors contributed equally to this work.

J. Li, H. Ma and M. Tomizuka are with the Department of Mechanical Engineering, University of California, Berkeley, CA 94720, USA (e-mail: jiachen_li, hengbo_ma, tomizuka@berkeley.edu)

realistic images of particular categories. There are mainly two aspects that can be modified to enhance the stability of GAN training: model setup [19]–[21] and optimization method [16], [22], [23]. Although there have been a large number of studies on realistic image generation tasks which provided promising results, much fewer research efforts, to our knowledge, have been devoted to continuous time-series data generation with adversarial training [24], [25]. Our task is different from existing works where the generators are trained in unsupervised learning settings without any visualizable groundtruth distribution. This makes it hard to evaluate the learning performance. On the contrary, in this work the groundtruth actions can be utilized to monitor the training quality by calculating a proper error metric as well as provide a heuristic during the training process.

Multi-target Tracking and Prediction

In recent years, widely studied multi-target tracking and prediction approaches can be mainly classified into two groups: (a) end-to-end solutions based on deep learning and computer vision techniques; (b) state estimation based on Bayesian inference. In this paper, we only focus on the latter one with an emphasis on particle filter (aka. sequential Monte Carlo) due to its effectiveness and flexibility. [26] provided a comprehensive summary of particle filter techniques. In the authors' previous work [27], a generic multi-target tracking framework without explicit data association was proposed which can incorporate arbitrary learning-based state evolution models to enhance tracking performance. The framework can be applied to prediction problems as well provided that the measurement update is removed. In this work, we employ the generator network in GAN as an implicit proposal distribution with high flexibility and generalizability which serves as the state evolution model in the tracking and prediction framework.

III. Methodology

In this section, we first make a brief introduction on the formulation and theory of GAN. Then, the proposed prediction model and algorithm are illustrated. Finally, the generic tracking framework is presented.

A. Background and Preliminaries

The Generative Adversarial Network is essentially a minimax two-player game including a generator G and discriminator D. The goal of generator is to learn the data distribution and generate samples as similar to real data as possible while the goal of discriminator is to distinguish whether the given sample is real or produced by the generator. Additional condition information can be incorporated in this competing process. The value function of the minimax game is

$$\begin{aligned}\min_G \max_D V(G,D) = \mathbb{E}_{\mathbf{y}\sim p_{data}(\mathbf{y})}[\log D(\mathbf{y}|\mathbf{x})] + \\ \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}(\mathbf{z})}[\log(1-D(G(\mathbf{z}|\mathbf{x}))],\end{aligned} \tag{1}$$

where $\mathbf{z}$ is the latent noise sampled from normal distribution, $\mathbf{x}$ is condition information and $p_{\text{data}}(\mathbf{y})$ is the data distribution for generator to capture. In practice, it is usually better for the generator to maximize $\log D(G(\mathbf{z}|\mathbf{x}))$ instead of minimizing $\log(1-D(G(\mathbf{z}|\mathbf{x})))$ due to the gradient saturation issue in the early training process.

B. Model Architecture and Algorithms

The proposed model consists of a generator network and a discriminator network like the canonical GAN, which is shown in Fig. 1. The generator and discriminator have a similar architecture which is a combination of one recurrent layer and multiple fully connected (FC) layers. The Long Short-term Memory (LSTM) [28] is adopted due to its superiority on learning long-term dependencies as well as alleviating gradient vanishing problems.

As depicted in Fig. 1, the inputs of the generator network contain a sequence of latent noise $\mathbf{Z} = (\mathbf{z}_{t-T_h}, \mathbf{z}_{t-T_h+1}, ..., \mathbf{z}_t)$, $\mathbf{z} \in \mathbb{R}^d$ where d is the dimension of noise vector and a sequence of condition information which refers to historical state trajectories $\mathbf{X} = (\mathbf{x}_{t-T_h}, \mathbf{x}_{t-T_h+1}, ..., \mathbf{x}_t)$ in this work, although the model itself can be generalized to take in various forms of conditions. The noise vectors are sampled independently from a multivariate normal distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$ or a uniform distribution $\mathbb{U}[0,1]$ at each time step. The outputs of the generator are fake samples which are actions of agents at the current time step in this work. The discriminator network takes as inputs the same condition information as the generator, fake samples and real ones. They are concatenated with outputs of the LSTM layer. The probability of each sample being fake or real is provided by the discriminator output.

Training Phase: Assume that GAN is a two players' zero-sum game, the loss function $V(G,D)$ is denoted as $f(\theta,\phi)$ where θ is the parameter vector of *player 1* (i.e. discriminator) and ϕ is the parameter vector of *player 2* (i.e. generator), then the objective functions are $f(x), -f(x)$ for the two players respectively. The optimum is a Nash equilibrium defined as:

$$\theta^* \in \underset{\theta}{\operatorname{argmax}} f(\theta,\phi^*), \quad \phi^* \in \underset{\phi}{\operatorname{argmax}} -f(\theta^*,\phi). \tag{2}$$

Define $x = (\theta,\phi)^T$, the Nash-equilibrium points possess the following properties:

$$\begin{aligned} v(x^*) &= \begin{bmatrix} \nabla_\theta f(x^*) \\ -\nabla_\phi f(x^*) \end{bmatrix} = 0, \\ H(x^*) &= \begin{bmatrix} \nabla_\theta^2 f(x^*) & \nabla_{\theta,\phi} f(x^*) \\ -\nabla_{\theta,\phi} f(x^*) & -\nabla_\phi^2 f(x^*) \end{bmatrix} \leq 0, \end{aligned} \tag{3}$$

where $H(x)$ is the gradient of vector $v(x)$.

The standard optimization algorithm for GAN is simultaneous gradient ascent (SGA) which gives:

$$x_{k+1} = x_k + \alpha v(x), \tag{4}$$

where α is the learning rate. The optimization is locally stable at Nash equilibrium if $H(x)$ is negative definite and α is small enough. However, in practice this naive update rule usually suffers a low covergence speed or even divergence, which makes the training process slow and unstable. To avoid these issues, we use the consensus optimization proposed by

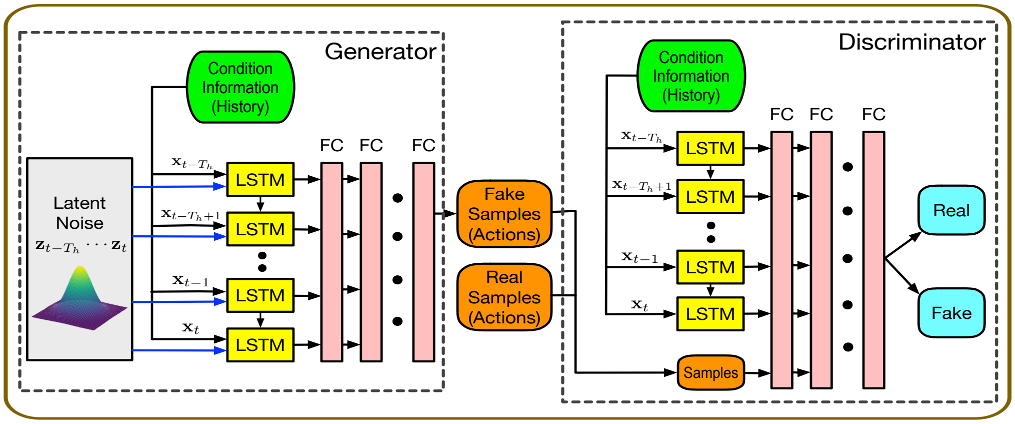

Fig. 1. The general diagram of the proposed model, which consists of a generator network and a discriminator network.

[16]. The objective functions of both players are added as a regularization term $-||v(x)||_2^2$. The update rule becomes:

$$x_{k+1} = x_k + \alpha(I - \gamma H(x)^T)v(x), \tag{5}$$

where γ is the ratio between the objective functions $f(x), -f(x)$ and regularization term $-||v(x)||_2^2$. The optimization will be locally stable at the Nash equilibrium if $H(x)$ is invertible and α and γ have proper values.

This algorithm makes sure that the optimization procedure is locally convergent to the Nash equilibrium of the original game by selecting proper γ and α. Since $H(x)$ may have eigenvalues with zero real parts at Nash equilibrium which leads to limit cycles or divergence, this optimization scheme alleviates this issue by pushing the eigenvalues away from imaginary axis. The generator and discriminator are trained end-to-end.

Testing Phase: When we use the well-trained model to make predictions, only the generator network is employed to generate a group of future trajectory hypotheses given a certain historical state sequence. Since the generated samples only contain actions at current time step, they are processed to obtain the new state which is appended to the historical state sequence for propagation at the next step. Generating predictions of multiple time steps simultaneously is also a promising idea which is left to future research.

C. Generic Mixture Tracking Framework

The recursive Bayesian state estimation typically consists of two steps: prior update and measurement update. A state evolution model and a measurement model are required to obtain the state evolution distribution and measurement likelihood. The standard particle filter is able to provide a satisfactory approximation of the true posterior distribution in single object tracking problems; nevertheless, it usually suffers mode degeneration when the real distribution possess multiple modalities. In order to maintain multi-modality arised from the existence of multiple tracking targets, the posterior state distribution is formulated as a mixture model

$$f(\mathbf{x}_k|\mathbf{z}^k) = \sum_{n=1}^{N} \pi_{n,k} f_n(\mathbf{x}_k|\mathbf{z}^k), \qquad \sum_{n=1}^{N} \pi_{n,k} = 1, \tag{6}$$

where $\mathbf{z}^k = (\mathbf{z}_0, ..., \mathbf{z}_k)$, N and $\pi_{n,k}$ refer to the component number of mixture distribution and corresponding weights, respectively. $f(\cdot)$, $\mathbf{x}_k$ and $\mathbf{z}_k$ refer to probability density function, state and measurement at the kth time step, respectively. The canonical prior update and measurement update are modified into

$$\begin{aligned} f(\mathbf{x}_k|\mathbf{z}^{k-1}) = \sum_{n=1}^{N} \pi_{n,k-1} \iint &[f_n(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{e}_{k-1}, \mathbf{z}^{k-1}) \\ &\times f_n(\mathbf{x}_{k-1}, \mathbf{e}_{k-1}|\mathbf{z}^{k-1})] \mathrm{d}\mathbf{x}_{k-1} \mathrm{d}\mathbf{e}_{k-1}, \end{aligned} \tag{7}$$

$$f(\mathbf{x}_k|\mathbf{z}^k) = \frac{\sum_{n=1}^{N} \pi_{n,k-1} f_n(\mathbf{z}_k|\mathbf{x}_k) f_n(\mathbf{x}_k|\mathbf{z}^{k-1})}{\sum_{m=1}^{N} \pi_{m,k-1} \int f_m(\mathbf{z}_k|\mathbf{x}_k) f_m(\mathbf{x}_k|\mathbf{z}^{k-1}) \mathrm{d}\mathbf{x}_k}, \tag{8}$$

where $\mathbf{e}_{k-1}$ represents exterior information. Please refer to [27] for more details about the tracking framework.

Since it is usually intractable and difficult to sample from the optimal proposal distributions $f_n(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{e}_{k-1}, \mathbf{z}_k)$, approximated ones $f_n(\mathbf{x}_k|\mathbf{x}_{k-1}, \mathbf{e}_{k-1})$ are usually employed to propagate particle hypotheses. In this work, the generator of GAN essentially provides an implicit proposal distribution in the prior update.

IV. Numerical Case Study

In order to demonstrate the performance of distribution learning of the proposed model, we present a numerical case study in which correlation and interaction exist among different random variables.

A. Problem Formulation

The Lotka-Volterra system is a general framework of biological systems where the populations of predators and preys interact with each other which is a simple case suitable for simulating interactions among multiple agents as well as evaluating the distributions learned by generator. The system equations are

$$\begin{aligned} \dot{x} &= ax - bxy, \\ \dot{y} &= cxy - dy, \end{aligned} \tag{9}$$

where a, b, c and d are parameters. The numerical dataset was generated by varying parameters and initial states. Parameters a, b, c and d were generated randomly from $\mathbb{U}[3, 5]$ and the initial states were generated randomly from $\mathbb{U}[1, 3]$.

B. Implementation Details

The generator and discriminator network both consist of one LSTM layer with 128 units and four fully connected layers with 64 hidden units followed by a ReLU activation function except the final output layer. The dimension of latent noise is 16. Both networks were trained with RMSProp [29] optimizer with a learning rate of 0.01. The hyperparameters were tuned manually with cross validation. We trained the model for 30,000 iterations with different regularization coefficients $\gamma = 0.00, 0.33, 1.00$ to investigate the effects of regularization term. The standard loss function suggested in [17] was used in all the experiments.

C. Results and Discussion

The training performance in terms of mean absolute error (MAE) values are shown in Fig. 2(c). It can be seen that higher γ values may push the algorithm to a local optimum instead of the Nash equilibrium, which brings a trade off on the choice of regularization coefficient. In this numerical example, $\gamma = 0.33$ leads to the best performance in terms of MAE during training process.

Since the objective of generator is to learn the data distribution, only employing the MAE metric is not sufficient. Therefore, we propose a generic distribution evaluation algorithm to obtain the groundtruth distribution of some statistic in the training dataset. For each given set of parameters a, b, c and d, the groundtruth trajectories can be obtained by propagating the true model forwardly. Since the proposed model needs a sequence of historical information as input, we propagate the true model from the initial state backwardly to obtain historical trajectories which are employed to produce future trajectories using the proposed model. More details are presented in **Algorithm 1**. We make a violin plot of the predicted distribution and the groundtruth distribution, which is illustrated in Fig. 2(a) and (b). It is implied that not only can the generator capture the mean and variance of the real data distribution, but also it can make a close approximation to the shape of the groundtruth distribution in terms of multi-modality.

Moreover, the well-trained model is used as a probabilistic state evolution model in the mixture tracking framework to track the state of the predator-prey system. The tracking results with 100 particles are shown in Fig. 2(d). The mean of particles are close to the ground truth, and the variance is proper at each time step.

V. Real-world Application: Vehicle Trajectory Prediction

In this section, we apply the proposed approach to solve a trajectory prediction task of interactive on-road vehicles as an illustrative example, although it can be utilized to solve many other tasks such as interactive pedestrian trajectory prediction and human-robot interactions.

Algorithm 1 Distribution Evaluation Algorithm

```
Input:
    $para$: a set of different parameters $a, b, c$ and $d$;
    $m$: number of different sets of parameters $a, b, c$ and $d$;
    $n$: number of trajectories generated by the proposed model;
    $T$: number of time steps to predict.
Output:
    $predicted\_traj$: trajectories generated by the proposed model;
    $real\_traj$: trajectories generated by the true model.
for $i = 1 : m$ do
    $s_{0:T} \leftarrow$ generate_forward_trajectory($para_i$)
    $real\_traj_i \leftarrow s_{0:T}$
    for $j = 1 : n$ do
        $s_{-T_h:-1} \leftarrow$ generate_backward_trajectory($s_0, para_i$)
        $\hat{s}_{-T_h:0} = s_{-k:0}$
        for $t = 0 : T - 1$ do
            $a_t \leftarrow$ policy_model($\hat{s}_{t-T_h:t}$)
            $\hat{s}_{t+1} \leftarrow$ update($\hat{s}_t, a_t$)
        end for
        $predicted\_traj_{ij} \leftarrow \hat{s}_{0:T}$
    end for
end for
return histogram($predicted\_traj$, $real\_traj$)
```

A. Problem Statement

A typical highway scenario shown in Fig. 3 is investigated where the gray car is the ego vehicle which aims at forecasting future motions of its surrounding vehicles (red, green and yellow ones). The observations of environment can be obtained by on-board sensors. The approach can also be adopted in overhead traffic surveillance systems with camera-based monitors. We assume that only the red car can make a lane change to left (LCL) or right (LCR) while all the others maintain the lane keeping (LK) behavior. This is a reasonable assumption since it is rare in the realistic driving dataset that two or more vehicles change lanes simultaneously under the setup in Fig. 3. The behaviors of the red car and green cars are largely dependent on the their front yellow ones. Our objective is to learn a joint distribution of the six cars' future motions given a sequence of historical states to make long-term predictions. We predict both the lateral deviation and longitudinal velocity of the red car at the next time step while only consider the longitudinal motions of surrounding cars since lateral positions within a lane make few effects during interaction as long as they are assumed to perform lane keeping behavior.

B. Dataset and Preprocessing

The NGSIM US-101 highway driving dataset [30] was used to extract training data and test data with a ratio of 4:1. We selected 320 lane change cases and 2,800 lane keeping cases in total with a time horizon of 7 seconds for each case according to the requirements mentioned in Section V-A. Due to the existence of large detection noise on velocity

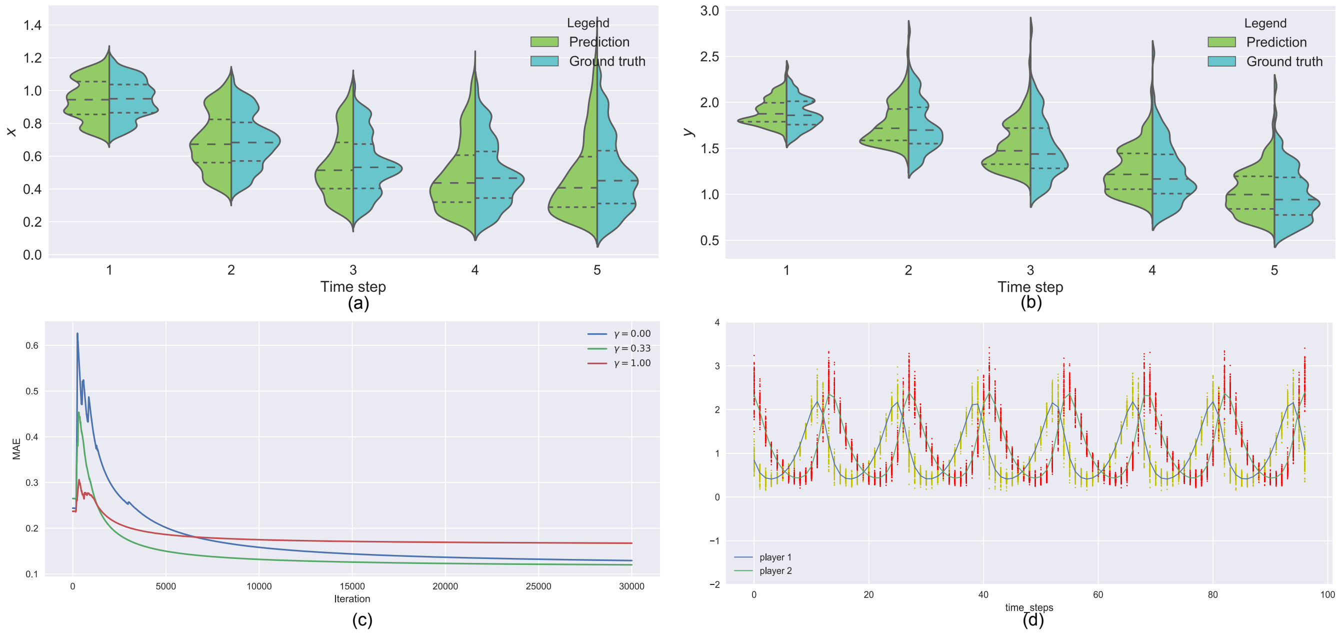

Fig. 2. Figures related to the numerical case study: (a) distribution comparison of x; (b) distribution comparison of y; (c) MAE values during the training process with different γ; (d) mixture tracking results.

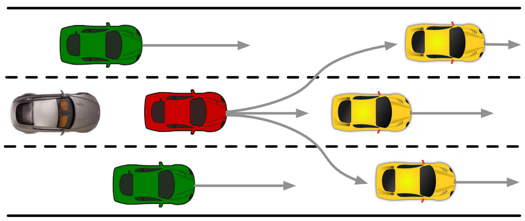

Fig. 3. An illustrative diagram of highway scenario. The gray car is the ego vehicle which predicts future behaviors and motions of the other cars.

and acceleration, an extended Kalman filter was applied to smooth the trajectories before feeding into the network.

C. Experiments and Discussion

The architectures of generator and discriminator network are identical to those in the numerical case study. The dimension of latent noise is 128. Both networks were trained with RMSProp optimizer with a learning rate of 0.0001. We trained the model for 100,000 iterations with a regularization coefficient of 0.1. The state features include relative positions of five surrounding car with respect to the red middle car (set the position of red car as origin) and longitudinal velocities of all the cars.

We compared the prediction performance of the proposed model with the following baseline models using the same input state features and output actions:

1) *Constant Velocity Model (CAM)*: CAM is one of the widely used kinematics models in vehicle tracking and prediction problems which assumes the vehicles go forward with a constant acceleration.

2) *Gaussian Mixture Regression (GMR)*: This model is similar to the behavior model proposed in [27] in which the historical condition information and predicted actions are flattened and concatenated to fit a Gaussian mixture model in the training phase which gives the joint distributions of input and output. In the testing phase, the predicted actions can be sampled from the conditional distribution of actions given a certain conditional input.

3) *Probabilistic Multi-layer Perceptron (P-MLP)*: In order to allow for a fair comparison, we added a Gaussian noise term to the input of MLP to incorporate uncertainty during both training and test process. We trained the P-MLP to minimize the mean square error (MSE) between the output actions and groundtruth actions. The network has five fully connected layers with 128 hidden units in each layer.

4) *Probabilistic LSTM (P-LSTM)*: The P-LSTM is very similar to P-MLP except that the first hidden layer is replaced with an LSTM layer.

All the baseline models were trained with RMSProp optimizer for 100,000 iterations with a learning rate of 0.0001, which is identical to the training setup of the proposed model.

The MAE of prediction results of test cases are shown in Table I, in which the first number is the position error of middle vehicle while the second one is the mean longitudinal position error of surrounding vehicles. The reported values in the table are the average errors of multiple experiments. The bold numbers indicate best prediction performance under corresponding testing conditions. It is shown that learning based models can achieve much better results than the CAM since the latter does not consider the interactions and uncertainty during the prediction period which suggests that vehicle kinematics models are only suitable for traffic scenarios with few interactions. The proposed model can achieve the highest prediction accuracy in most cases except sometimes outperformed by P-MLP in lane keeping cases.

TABLE I
MAE Comparisons of Vehicle Position Prediction (Center vehicle / Surrounding Vehicles)

Cases	Prediction Horizon (s)	Proposed model (m)	GMR (m)	P-MLP (m)	P-LSTM (m)	CAM (m)
Lane Keeping (LK)	1.0	**0.11** / **0.05**	0.14 / 0.11	0.27 / 0.27	0.23 / 0.19	0.63 / 0.37
	2.0	0.83 / **0.31**	1.03 / 0.95	**0.74** / 0.74	0.82 / 0.59	1.93 / 1.28
	3.0	**1.17** / **0.66**	2.34 / 1.68	1.20 / 1.42	1.61 / 1.25	3.23 / 2.93
	4.0	2.31 / **1.29**	3.61 / 2.16	**1.75** / 2.26	2.69 / 1.91	4.73 / 3.88
	5.0	3.23 / **2.53**	4.18 / 3.86	**2.97** / 3.39	4.59 / 2.81	5.91 / 5.12
Lane Change (LC)	1.0	**0.08** / **0.04**	0.21 / 0.13	0.35 / 0.28	0.34 / 0.25	0.51 / 0.44
	2.0	**0.46** / **0.26**	0.93 / 1.06	1.05 / 0.93	0.96 / 0.83	1.56 / 1.47
	3.0	**1.21** / **0.71**	2.13 / 1.74	1.85 / 1.88	1.69 / 1.57	2.68 / 2.11
	4.0	**1.97** / **1.47**	3.26 / 2.48	2.67 / 2.92	2.91 / 2.69	4.72 / 3.64
	5.0	**3.12** / **2.71**	5.21 / 4.06	3.88 / 3.64	4.00 / 3.78	7.33 / 5.27

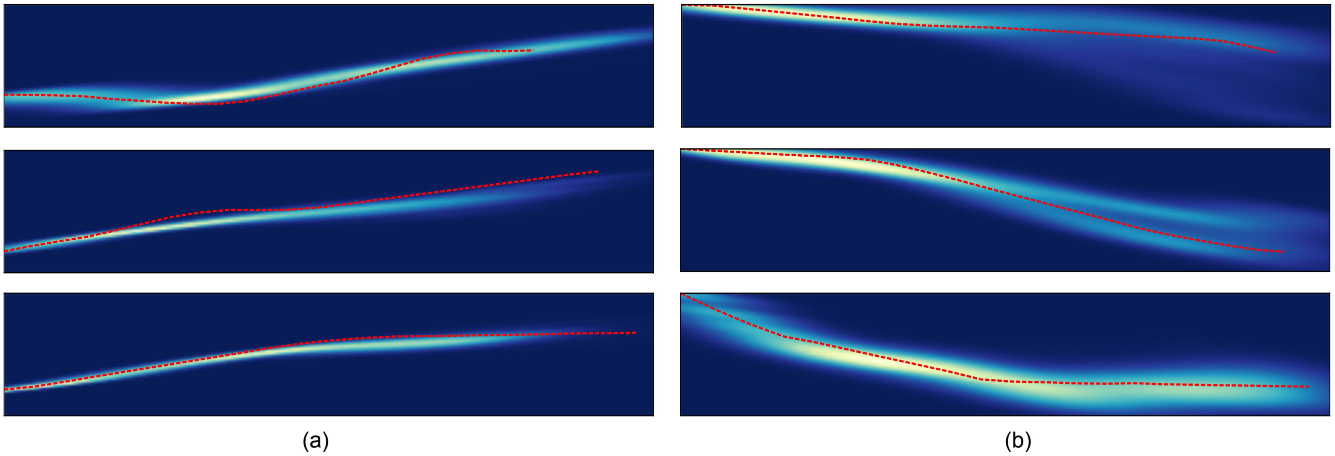

Fig. 4. Visualization of selected cases. (a) lane change left cases; (b) lane change right cases. The red dash lines are groundtruth trajectories.

The reason is that instead of minimizing the divergence of the generated sample distribution and groundtruth distribution, P-MLP tends to overfit training cases by maximizing likelihood which reduces model generalizability. Moreover, minimizing MSE loss may lead to average output of training action labels. Therefore, although P-MLP works well in short-term prediction for lane keeping behaviors due to the low diversity of motion patterns, it fails to maintain a similar prediction accuracy for lane change behaviors where the variance of action distribution is much larger. Also, it seems that the recurrent layer in P-LSTM has little help on improving prediction performance. The GMR model can achieve better performance than P-MLP and P-LSTM in short-term prediction (less than 2 seconds) while it losses superiority as the prediction horizon increases. Another issue is that we cannot incorporate arbitrary length of historical information in GMR since the data points become very sparse in high dimensional space which may lead to singular covariance matrix. This suggests that GMR is not suitable for learning long-term dependencies of time-series data.

Some prediction results of test cases are visualized in Fig. 4. It is illustrated that the groundtruth trajectories are located near the mean of the predicted distribution. Also note that in Fig. 4(b), the proposed model is able to forecast a multi-modal distribution and the groundtruth trajectory belongs to one of the modes.

VI. Conclusions

In this paper, an adversarial learning based generic framework for interaction-aware multi-agent tracking and prediction was proposed, which can be generalized to any time-series prediction tasks. We also proposed to utilize the distribution learned by the generator as an implicit proposal distribution of mixture sequential Monte Carlo methods to improve state estimation accuracy. A numerical case study was demonstrated to provide an empirical study on how to improve training performance of GAN through adding a regularization term to the standard loss function. The results show that increasing the regularization coefficient indeed accelerates convergence but may lead to sub-optimum which implies a trade-off in parameter tuning. It is also shown that our model can capture not only the mean and variance of the groundtruth data distribution but also the inherent multi-modalities. We also applied the proposed approach to a task of multi-vehicle trajectory prediction to illustrate its practicability and effectiveness.

References

[1] Q. Ke, M. Bennamoun, S. An, F. Sohel, and F. Boussaid, “Leveraging structural context models and ranking score fusion for human interaction prediction,” *IEEE Transactions on Multimedia*, vol. 20, no. 7, pp. 1712–1723, 2018.

[2] Y. Xu, Z. Piao, and S. Gao, “Encoding crowd interaction with deep neural network for pedestrian trajectory prediction,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5275–5284.

[3] S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool, “You’ll never walk alone: Modeling social behavior for multi-target tracking,” in *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 2009, pp. 261–268.

[4] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, “Social lstm: Human trajectory prediction in crowded spaces,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 961–971.

[5] M. Zhou, X. Qu, and S. Jin, “On the impact of cooperative autonomous vehicles in improving freeway merging: a modified intelligent driver model-based approach,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 6, pp. 1422–1428, 2017.

[6] A. Sarkar, K. Czarnecki, M. Angus, C. Li, and S. Waslander, “Trajectory prediction of traffic agents at urban intersections through learned interactions,” in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2017, pp. 1–8.

[7] N. Deo and M. M. Trivedi, “Multi-modal trajectory prediction of surrounding vehicles with maneuver based lstms,” *arXiv preprint arXiv:1805.05499*, 2018.

[8] A. Jain, A. Singh, H. S. Koppula, S. Soh, and A. Saxena, “Recurrent neural networks for driver activity anticipation via sensory-fusion architecture,” in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE, 2016, pp. 3118–3125.

[9] N. Deo, A. Rangesh, and M. M. Trivedi, “How would surround vehicles move? a unified framework for maneuver classification and motion prediction,” *IEEE Transactions on Intelligent Vehicles*, vol. 3, no. 2, pp. 129–140, 2018.

[10] H. Kretzschmar, M. Kuderer, and W. Burgard, “Learning to predict trajectories of cooperatively navigating agents,” in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 4015–4020.

[11] J. Li, H. Ma, W. Zhan, and M. Tomizuka, “Generic probabilistic interactive situation recognition and prediction: From virtual to real,” in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 3218–3224.

[12] W. Zhan, L. Sun, Y. Hu, J. Li, and M. Tomizuka, “Towards a fatality-aware benchmark of probabilistic reaction prediction in highly interactive driving scenarios,” in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 3274–3280.

[13] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. Torr, and M. Chandraker, “Desire: Distant future prediction in dynamic scenes with interacting agents,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 336–345.

[14] A. Kuefler, J. Morton, T. Wheeler, and M. Kochenderfer, “Imitating driver behavior with generative adversarial networks,” in *Intelligent Vehicles Symposium (IV), 2017 IEEE*. IEEE, 2017, pp. 204–211.

[15] J. Ho and S. Ermon, “Generative adversarial imitation learning,” in *Advances in Neural Information Processing Systems*, 2016, pp. 4565–4573.

[16] L. Mescheder, S. Nowozin, and A. Geiger, “The numerics of gans,” in *Advances in Neural Information Processing Systems*, 2017, pp. 1825–1835.

[17] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[18] M. Mirza and S. Osindero, “Conditional generative adversarial nets,” *arXiv preprint arXiv:1411.1784*, 2014.

[19] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” *arXiv preprint arXiv:1511.06434*, 2015.

[20] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein generative adversarial networks,” in *International Conference on Machine Learning*, 2017, pp. 214–223.

[21] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, “Infogan: Interpretable representation learning by information maximizing generative adversarial nets,” in *Advances in neural information processing systems*, 2016, pp. 2172–2180.

[22] S. Arora, R. Ge, Y. Liang, T. Ma, and Y. Zhang, “Generalization and equilibrium in generative adversarial nets (gans),” *arXiv preprint arXiv:1703.00573*, 2017.

[23] V. Nagarajan and J. Z. Kolter, “Gradient descent gan optimization is locally stable,” in *Advances in Neural Information Processing Systems*, 2017, pp. 5585–5595.

[24] O. Mogren, “C-rnn-gan: Continuous recurrent neural networks with adversarial training,” *arXiv preprint arXiv:1611.09904*, 2016.

[25] C. Esteban, S. L. Hyland, and G. Rätsch, “Real-valued (medical) time series generation with recurrent conditional gans,” *arXiv preprint arXiv:1706.02633*, 2017.

[26] X. Wang, T. Li, S. Sun, and J. M. Corchado, “A survey of recent advances in particle filters and remaining challenges for multitarget tracking,” *Sensors*, vol. 17, no. 12, p. 2707, 2017.

[27] J. Li, W. Zhan, and M. Tomizuka, “Generic vehicle tracking framework capable of handling occlusions based on modified mixture particle filter,” in *Proceedings of 2018 IEEE Intelligent Vehicles Symposium (IV)*, 2018, pp. 936–942.

[28] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[29] T. Tieleman and G. Hinton, “Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude,” COURSERA: Neural Networks for Machine Learning, 2012.

[30] J. Colyar and J. Halkias, “US highway 101 dataset,” in *Federal Highway Administration (FHWA) Technique Report. FHWA-HRT-07-030*, 2007.