Improving Movement Predictions of Traffic Actors in Bird's-Eye View Models using GANs and Differentiable Trajectory Rasterization

Eason Wang*, Henggang Cui*, Sai Yalamanchi, Mohana Moorthy, Nemanja Djuric
{junheng,hcui2,syalamanchi,mmoorthy,ndjuric}@uber.com
Uber Advanced Technologies Group

ABSTRACT

One of the most critical pieces of the self-driving puzzle is the task of predicting future movement of surrounding traffic actors, which allows the autonomous vehicle to safely and effectively plan its future route in a complex world. Recently, a number of algorithms have been proposed to address this important problem, spurred by a growing interest of researchers from both industry and academia. Methods based on top-down scene rasterization on one side and Generative Adversarial Networks (GANs) on the other have shown to be particularly successful, obtaining state-of-the-art accuracies on the task of traffic movement prediction. In this paper we build upon these two directions and propose a raster-based conditional GAN architecture, powered by a novel differentiable rasterizer module at the input of the conditional discriminator that maps generated trajectories into the raster space in a differentiable manner. This simplifies the task for the discriminator as trajectories that are not scene-compliant are easier to discern, and allows the gradients to flow back forcing the generator to output better, more realistic trajectories. We evaluated the proposed method on a large-scale, real-world data set, showing that it outperforms state-of-the-art GAN-based baselines.

ACM Reference Format:
Eason Wang*, Henggang Cui*, Sai Yalamanchi, Mohana Moorthy, Nemanja Djuric. 2020. Improving Movement Predictions of Traffic Actors in Bird's-Eye View Models using GANs and Differentiable Trajectory Rasterization. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '20), August 23–27, 2020, Virtual Event, CA, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3394486.3403283

1 INTRODUCTION

Progress in Artificial Intelligence (AI) applications has experienced strides in the last decade, with a number of highly publicized success stories that have captured attention and imagination of professionals and laymen alike. Computers have reached and even surpassed human performance in centuries old games such as go and chess [1, 2], are starting to understand health conditions and suggest medical treatments [3], and can reason about complex relationships conveyed through images [4]. Prior to this, AI was already well-established at the very core of multibillion-dollar industries such as advertising [5] and finances [6], where today algorithmic approaches have all but replaced human experts. Nevertheless, despite these advancements the AI revolution is far from over [7]. The automobile domain is one of the last major industries to be disrupted by the current wave, where AI is yet to make its greatest impact through the development and deployment of self-driving vehicles (SDVs). Coupled with the latest breakthroughs in hardware and the recent advent of electric vehicles at a larger scale, this opens doors to potentially redefine our cities and our very way of life.

Although seemingly slow to adopt new technologies, today's automotive and transportation companies are embracing the AI and making extensive use of advanced algorithms [8]. The scope of application of the AI ranges from optimizing the production pipelines to help reduce costs and improve manufacturing procedures [9], to connecting vehicles into networks [10] and developing wide-scale intelligent transportation systems to improve overall traffic efficiency [11]. Most importantly, a particular focus of the automakers has been on safety of passengers and other traffic actors, defined as other active participants in traffic such as vehicles, pedestrians, and bicyclists. As a result, recent years have seen a surge of advanced driver-assistance systems (ADAS) [12], including advanced cruise control, collision avoidance, or lane departure systems, to name a few. While advancing safety on our roads, these systems are only mitigating the unreliable human factor that is the main cause of a vast majority of traffic accidents [13], and there is a lot to be done to improve road fatalities statistics that are still among the worst in the past decade [14]. A possible solution to this concerning situation is a development of the self-driving technology, which holds promise to completely remove the human factor from the equation, thus improving both safety and efficiency of the road traffic. The efforts on this technology started several decades ago [15, 16], however only recently has it become a center of attention for researchers from both industry and academia [17, 18].

One of the key factors and requirements for the autonomous driving technology to be safely deployed in complex urban environments are efficient and accurate detection, tracking, and motion prediction of surrounding traffic actors [19, 20]. In the current work we focus on the problem of prediction, tasked with capturing and inferring accurate and realistic future behavior and uncertainty of actor movement, in order to ensure more efficient, effective, and

Authors contributed equally. This work was done while Eason Wang was an intern at Uber ATG in Pittsburgh, PA, USA.

KDD '20, August 23–27, 2020, Virtual Event, CA, USA

ACM ISBN 978-1-4503-7998-4/20/08…$15.00
https://doi.org/10.1145/3394486.3403283

safe SDV route planning. This is a critical component of the autonomous system, and significant progress has been made along that direction in recent years. In particular, the current state-of-the-art approaches are bird's-eye view (BEV) rasterization methods [21–23] on one side that generate BEV raster images of an actor's surroundings as an input to deep networks (see Figure 1 for an example image), and Generative Adversarial Networks (GANs) [24–26] on the other. When combined, these two approaches provide all the context required for the task, while matching a real-world distribution through the adversarial framework. However, the output trajectories are still far from optimal, oftentimes not fully obeying the physical and map constraints present in the scene. We address this important issue, and propose a novel GAN architecture conditioned on an input raster image, referred to as Scene-Compliant GAN (SC-GAN). The critical component of the architecture is the differentiable rasterizer, which allows projection of predicted trajectories directly into the raster space in a differentiable manner. This simplifies the discriminator task, leading to higher efficiency of the adversarial training and more realistic output trajectories, as exemplified in the evaluation section.

Main contributions of our work are summarized below:

- we present a raster-based SC-GAN architecture based on a novel differentiable rasterizer module, simplifying the discriminator's job while allowing the gradients to flow back to the generator;
- we evaluate the proposed method on a large-scale, real-world data, showing that SC-GAN outperforms the existing state-of-the-art generative approaches.

2 RELATED WORK

In this section we give an overview of the relevant literature on the topic of traffic motion prediction. We first discuss approaches focused on providing scene-compliant trajectories, followed by a discussion on recently proposed state-of-the-art methods based on adversarial training.

2.1 Predicting scene-compliant trajectories

Trajectory prediction of surrounding actors on the road is a critical component in many autonomous driving systems [27, 28], as it allows the SDV to safely and efficiently plan its path through a dynamic traffic environment. A big part of the solution to this problem is ensuring that the predicted trajectories are compliant with the given scene, obeying the constraints imposed by the other actors as well as the map elements. The method described by Mercedes-Benz [18] makes direct use of the mapped lane information, associating actors to the nearby lanes and predicting trajectories along the lane geometry. The proposed method does not require learning and is based on heuristics, which may be suboptimal for uncommon traffic scenarios (such as in the case of non-compliant actor behavior).

Beyond using map info to predict behavior through hand-tuned rules, recently a number of methods were proposed that use the map data and information on dynamic objects in SDV's vicinity as inputs to learned algorithms. The current state-of-the-art approaches rasterize static high-definition map and the detected objects in BEV raster images, ingested by deep networks trained to predict future trajectories [22, 23, 29, 30]. The raster images are used to convey *scene context information* to the model, while convolutional neural networks (CNNs) are commonly employed to extract *scene context features* from the context image. Then, a trajectory decoder module is used to generate trajectory predictions based on the computed scene context features and the actor's past observations. The scene context info used by the model represents a strong prior for behavior prediction. This especially holds true for vehicle actors, and allows the network to learn to predict *scene-compliant* trajectories that follow lanes and obey traffic rules [21, 23, 31].

Despite the benefits of rasterized input representation, the learned models may still output non-compliant trajectories as the predictions are not constrained in any way. Authors of [32] proposed to combine learned and hand-tuned approaches to constrain the output trajectories. There are several published works on directly improving the scene compliance of learned models through extensions of a loss function. In ChauffeurNet [28], DRF-NET [33], and in the work by Ridel et al. [34], the authors proposed to predict an occupancy probability heatmap for each prediction horizon, where they explicitly penalized the probability mass in off-road regions of the scene to enforce scene compliance. Niedoba et al. [35] proposed a scene-compliance loss applicable to models that directly predict trajectory point coordinates, as opposed to predicting occupancy probability heatmaps. In addition, they also proposed novel scene-compliance metrics that quantify how often the trajectories are predicted to go off-road. In this work, instead of manually designing losses to penalize non-compliant outputs, we leverage the GAN framework and train a model in an adversarial fashion to encourage more realistic, scene-compliant trajectories.

2.2 GAN-based trajectory predictions

Approaches based on GANs have shown outstanding performance in a variety of machine learning tasks [36–38]. Following the success of general GAN architectures, a number of studies applying adversarial models to trajectory prediction have been proposed [24–26, 39]. A common theme in these works is that the generator network outputs trajectory predictions given the inputs, while the discriminator network classifies whether the provided trajectory is coming from the generator or from the ground truth. The gradients from the discriminator help the generator push the distribution of the generated trajectory predictions closer to the ground-truth distribution. However, most of these GAN-based models do not condition on the scene context image in the discriminator (e.g., Social-GAN [24], Sophie [25], MATF [39]), leading to suboptimal performance. In particular, the discriminator encodes the input trajectory with a Long Short-Term Memory (LSTM) encoder and makes classifications based solely on the trajectory embeddings. As a result, the discriminator is not able to distinguish between an actual ground-truth trajectory and a trajectory close to the ground truth that is not scene-compliant.

In a recent work [26] the authors proposed Social-BiGAT that addressed this issue by including the scene context image as an input to the discriminator. Their discriminator uses a CNN to extract features from the scene context image, and simply concatenates the flattened scene features with the trajectory embeddings in order to perform classification. This approach, however, will result in the trajectory embeddings being generated in a separate path from the

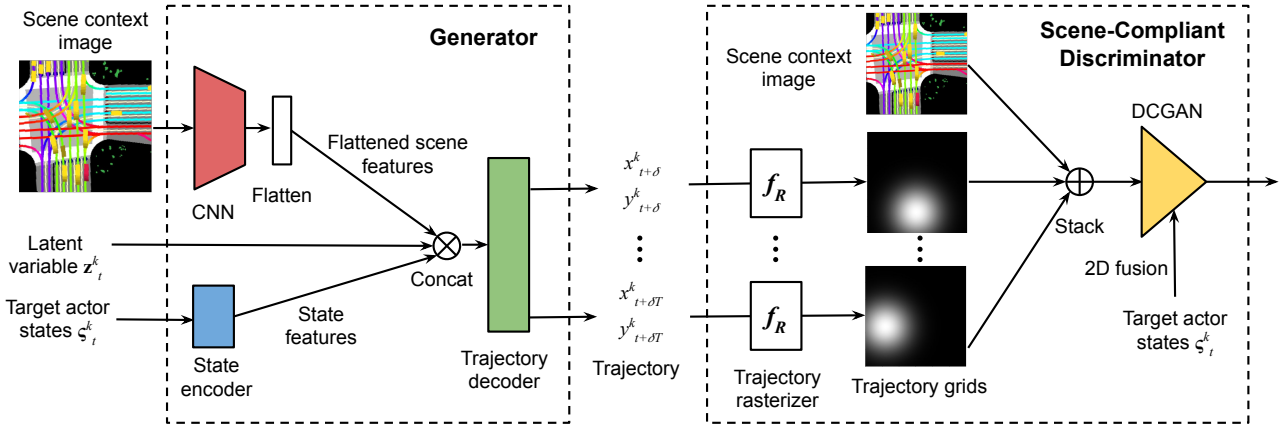

Figure 1: The proposed SC-GAN architecture; modules marked in color are learned during training

scene context embeddings, and a non-compliant prediction (e.g., trajectory going off-road) may not trigger a strong activation in the discriminator network. On the other hand, the proposed SC-GAN has a novel Scene-Compliant Discriminator architecture that transforms the trajectory input into a sequence of 2D occupancy grids in a differentiable way, and stacks the occupancy grids with the scene context image along the channel dimension. By combining the projected trajectory with the scene context image in the same raster space, the discriminator becomes more sensitive to non-compliant trajectories. Evaluation on real-world data collected on public roads shows that the novel model design makes the predicted trajectories considerably more scene-compliant and accurate, compared to Social-BiGAT and other state-of-the-art generative approaches.

3 SCENE-COMPLIANT GAN (SC-GAN)

In this section we propose a GAN-based method aimed at producing trajectories that better follow scene constraints. We first describe the problem setting, followed by the introduction of generator and discriminator submodules, as well as the novel differentiable rasterizer. Lastly, we discuss the loss used during training.

3.1 Problem statement

Let us assume that we have access to a real-time data stream coming from sensors such as lidar, radar, or camera, installed onboard a self-driving vehicle. In addition, we assume that this data is used as an input by an existing detection and tracking system, outputting state estimates for all surrounding actors up to the current time t_c. The tracked state comprises the bounding box, center position $[x, y]$, velocity v, acceleration a, heading θ, and heading change rate $\dot{\theta}$, where the tracker provides its outputs at a fixed sampling frequency of $10Hz$, resulting in the discrete tracking time step of $\delta = 0.1s$. We denote state output of the tracker for the i-th actor at time t as $\mathbf{s}_t^i = [x_t^i, y_t^i, v_t^i, a_t^i, \theta_t^i, \dot{\theta}_t^i]$, and the total number of actors tracked at time t as N_t (note that in general the actor counts vary for different time steps as new actors appear within and existing ones disappear from the sensor range). Moreover, we further assume access to a detailed, high-definition map information of the SDV's operating area denoted by $\mathcal{M}$, including road and crosswalk locations, lane directions, and other relevant map information such as observed traffic lights and signage.

Let $\mathcal{S}_{t_c} = \{\mathbf{S}_{t_c-(L-1)\delta}, \mathbf{S}_{t_c-(L-2)\delta}, \ldots, \mathbf{S}_{t_c}\}$ denote the tracked states of all actors detected at time t_c over the past L timestamps, where $\mathbf{S}_t = \{\mathbf{s}_t^1, \mathbf{s}_t^2, \ldots, \mathbf{s}_t^{N_t}\}$ represents the state of all actors at time t. Then, given the state information $\mathcal{S}_{t_c}$ and the map data $\mathcal{M}$, our goal is to predict future positions of a target actor k over the next T timestamps $\mathbf{o}_{t_c}^k = [x_{t_c+\delta}^k, y_{t_c+\delta}^k, \ldots, x_{t_c+T\delta}^k, y_{t_c+T\delta}^k]$, with k being in the $[1, N_{t_c}]$ range. The output trajectory is in the actor frame of reference, with origin at the center position, x-axis defined by the actor's heading, and y-axis defined by the left-hand side of the actor. Without loss of generality, and similarly to the existing trajectory prediction work [21, 23, 28–31], we only predict future center positions of the actor, from which other state variables can be determined. Alternatively, one could extend the proposed model to predict actor's full future states $\mathbf{s}_{t_c}^k$ by using ideas proposed in [40], however this is beyond the scope of our current work.

3.1.1 Scene rasterization. For each individual actor we separately predict future trajectories, where the input to the network is generated through rasterization approaches similar to those used in earlier work [21, 23, 28–31]. In particular, for the k-th actor the scene context information on the surrounding tracked actors and the map constraints are provided to the network by rasterizing the map $\mathcal{M}$ and all actors' past polygons from $\mathcal{S}_{t_c}$ onto a per-actor RGB raster image $\mathcal{I}_{t_c}^k$ with resolution r. The raster can be represented by a matrix of size $H \times W \times 3$, with the actor of interest k located at cell index $[h_0, w_0]$ and rotated such that actor's heading is pointing up. In our experiments we set $H = W = 300$, $h_0 = 50$, $w_0 = 150$, with $r = 0.2$ meters per pixel, such that the raster image captures $50m$ in front of the actor, $10m$ behind, and $30m$ on both sides of the actor. The rasterized map elements include road polygons, driving paths and their directions, crosswalks, detected traffic signals, and other relevant info. Actors' polygons are rasterized with different shadings representing different past timestamps, and the target actor is colored differently from other actors in its surroundings (see Figure 1 for an example, where we used red and yellow colors to differentiate them, respectively).

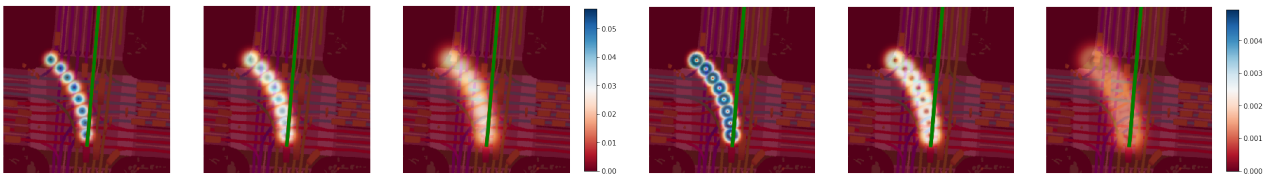

Figure 2: Visualization of the differentiable rasterizer by varying $\sigma \in \{1.4, 2, 3\}$ meters, respectively, with non-scene-compliant predicted trajectory (going into opposite lane and off-map) overlaid onto the scene and the ground-truth trajectory shown in green; left: rasterizer output; right: gradient norms; with larger σ the gradients are more spread out but are also weaker, while for smaller σ values the gradients are stronger but are more concentrated (e.g., a network using such configuration might not learn effectively if the rasterized point is too far away from the ground truth or the on-road map elements)

3.2 Model architecture

In this section we present the overall architecture of the proposed motion prediction model, illustrated in Figure 1. It consists of three main modules: 1) generator network, 2) discriminator network, and 3) differentiable trajectory rasterizer. In the following we describe each module in more detail.

3.2.1 Generator network. The generator network G (parameterized by the parameter set $\boldsymbol{\theta}_G$) generates the trajectory prediction $\hat{\mathbf{o}}^k_{t_c}$ given the concatenated actor's state inputs $\boldsymbol{\varsigma}^k_{t_c} = [\mathbf{s}^k_{t_c-(L-1)\delta}, \ldots, \mathbf{s}^k_{t_c}]$, per-actor raster $\mathcal{I}^k_{t_c}$, and a noise vector $\mathbf{z}^k_{t_c}$ of dimensionality d with each element sampled from a normal distribution $\mathcal{N}(0, 1)$,

$$\hat{\mathbf{o}}_{t_c} = G(\boldsymbol{\varsigma}_{t_c}, \mathcal{I}_{t_c}, \mathbf{z}_{t_c}; \boldsymbol{\theta}_G), \tag{1}$$

where here and in the remainder of the section we drop the actor index superscript k to simplify the notation. As can be seen in Figure 1, the generator first extracts the scene context features from the scene image $\mathcal{I}_{t_c}$ using a convolutional neural network. While any CNN can be used for this purpose [21], in order to allow for fast real-time inference onboard the SDVs we used MobileNet [41]. Past observed actor states $\boldsymbol{\varsigma}_{t_c}$ are also embedded with a shallow encoding layer, and concatenated with the extracted scene context features and the latent noise vector before being passed to a trajectory decoder module that generates the trajectory predictions.

3.2.2 Discriminator network. The discriminator network (parameterized by the parameter set $\boldsymbol{\theta}_D$) classifies whether a given future trajectory $\mathbf{o}_{t_c}$ is coming from a ground truth (i.e., *true*) or the generator (i.e., *fake*), conditioned on the past observed states $\boldsymbol{\varsigma}_{t_c}$ and the scene context image $\mathcal{I}_{t_c}$. In the previous GAN-based work, either the discriminator architectures did not use the scene context information at all (e.g., Social-GAN [24], Sophie [25], and MATF [39]), or the features from the scene context image $\mathcal{I}_{t_c}$ and from the provided trajectory $\mathbf{o}_{t_c}$ were extracted using two separate networks, followed by a simple concatenation of the computed features (e.g., Social-BiGAT [26]). Such discriminator design is not sufficiently sensitive to non-compliant trajectories due to the separation of handling of the two inputs, as confirmed in our experiments. In addition, these discriminators require the use of fully-connected layers, which is advised against by some authors [42]. In this work we propose a scene-compliant architecture that is more sensitive to non-compliant trajectories, comprising only fully convolutional layers. The proposed discriminator relies on a novel module called differentiable rasterizer, described in the following section.

3.2.3 Differentiable rasterizer. The trajectory rasterization module of the scene-compliant discriminator is tasked with rasterizing the future trajectory $\mathbf{o}_{t_c}$ (either predicted or ground-truth) into a sequence of 2D occupancy grids $\Gamma_{t_c} = \{\mathcal{G}_{t_c+\delta}, \ldots, \mathcal{G}_{t_c+T\delta}\}$, where each $\mathcal{G}_t = f_R([x_t, y_t], \sigma)$ encodes a single trajectory point at time t and is an $H \times W$ 2D grid with the same shape and resolution as the raster image $\mathcal{I}_{t_c}$. $f_R(\cdot)$ is a rasterizer function, $[x_t, y_t]$ are coordinates of the trajectory point to be rasterized, and σ is a predefined visualization hyper-parameter to be discussed promptly.

Let us denote distance in the actor frame between a cell $[i, j]$, where $i \in \{1, \ldots, H\}$ and $j \in \{1, \ldots, W\}$, and the trajectory point $[x_t, y_t]$ as follows,

$$\Delta^{ij}_t = \Big[(i - h_0)r, (j - w_0)r\Big] - [x_t, y_t], \tag{2}$$

computed by considering the raster resolution r and the origin cell of the actor of interest $[h_0, w_0]$. Then, the trajectory rasterizer calculates value for cell $[i, j]$ of the image $\mathcal{G}_t$ as a probability density of a 2D Gaussian distribution at Δ^{ij}_t,

$$\{\mathcal{G}_t\}_{ij} = \mathcal{N}(\Delta^{ij}_t | \mathbf{0}, \Sigma). \tag{3}$$

The covariance matrix is defined as $\Sigma = diag([\sigma^2, \sigma^2])$, such that the standard deviation σ modulates the probability density of the rasterized trajectory point. An example of the resulting image is shown in Figure 1. We let the rasterizer $f_R(\cdot)$ rasterize each trajectory point as a Gaussian occupancy density map, as opposed to a one-hot matrix, as this facilitates back-propagation during training through well-defined gradients. In particular, the gradient vector is aligned with the direction of Δ^{ij}_t and computed as follows,

$$\nabla_{[x_t, y_t]}\Big(\{\mathcal{G}_t\}_{ij}\Big) = [\frac{\partial\{\mathcal{G}_t\}_{ij}}{\partial x}, \frac{\partial\{\mathcal{G}_t\}_{ij}}{\partial y}] = -\frac{\{\mathcal{G}_t\}_{ij}}{\sigma^2}\Delta^{ij}_t, \tag{4}$$

justifying the name *differentiable rasterizer* of the module.

For a fixed point $[x_t, y_t]$ the gradient achieves its maximum ℓ_2 norm of $1/(\sqrt{2\pi e}\sigma^2)$ when $\|\Delta^{ij}_t\|_2 = \sigma$. Let us represent a super-level set of the gradient norm as $\mathcal{S}_\alpha = \{[i, j] : \|\nabla_{[x_t, y_t]}(\{\mathcal{G}_t\}_{ij})\|_2 \geq \alpha\}$, where $0 < \alpha \leq \frac{1}{\sqrt{2\pi e}\sigma^2}$. This set consists of points that form an annulus in the spatial extent of the occupancy grid, and it always contains the ring of points that satisfy $\|\Delta^{ij}_t\|_2 = \sigma$. Points in $\mathcal{S}_\alpha$ will map to gradients of significant magnitudes, as per definition of $\mathcal{S}_\alpha$. Increasing σ will increase the width of the annulus (for a

fixed choice of α), but will decrease the value of the maximum of a gradient norm. The choice of σ thus controls the balance between how well spread out gradients spatially are and their norm; see Figure 2 for visualization of this phenomenon. It is interesting to note that when x_t or y_t are outside of the 2D grid range $H \times W$ (e.g., this can happen for a particularly poor initialization of the model weights), only a small part of $\mathcal{S}_\alpha$ needs to be contained within the grid in order for $[x_t, y_t]$ to still receive a meaningful gradient.

Once the trajectory is rasterized in a differentiable manner, all 2D occupancy grids Γ_{t_c} for a trajectory $\mathbf{o}_{t_c}$ are stacked together with the scene context image $\mathcal{I}_{t_c}$ along the channel dimension. This results in a multi-channel image with both the scene context and the trajectory "plotted" on top of it. Using the proposed approach the discriminator has an easier task to decide whether or not the generated trajectory is valid and scene-compliant, as the rasterized scene elements and the trajectory are aligned in the raster space. This is unlike prior work where the raster and trajectories were simply concatenated together [26], which results in a much more difficult task for the discriminator as we confirm in Section 4. We employ the fully-convolutional DCGAN architecture [42] as our discriminator, and also fuse the past observed states $\boldsymbol{\varsigma}_{t_c}$ of the actor of interest into the multi-channel raster using the 2D fusion method proposed in [29].

3.3 Training loss

Unlike previous GAN-based prediction works [24–26] which used the vanilla cross-entropy loss as their GAN loss, we used the Wasserstein GAN loss with gradient penalty [43, 44], shown to outperform other approaches. For brevity, let us denote the discriminator network $D(\boldsymbol{\varsigma}_{t_c}, \mathcal{I}_{t_c}, \Gamma_{t_c}; \boldsymbol{\theta}_D)$ by $D(\Gamma_{t_c})$. Let $\mathbb{P}_g$ be the generated data distribution, $\mathbb{P}_r$ the true data distribution, and $\mathbb{P}_{\tilde{\mathbf{o}}_{t_c}}$ the distribution implicitly defined by sampling uniformly along straight lines between pairs of points sampled from $\mathbb{P}_g$ and $\mathbb{P}_r$ distributions. The distribution $\mathbb{P}_{\tilde{\mathbf{o}}_{t_c}}$ is used for computing the gradient penalty term (see [44] for more details). Then, the GAN-based loss used to train SC-GAN is given as

$$\mathcal{L} = \mathop{\mathbb{E}}_{\hat{\mathbf{o}}_{t_c} \sim \mathbb{P}_g} \Big(D(\hat{\Gamma}_{t_c}) \Big) - \mathop{\mathbb{E}}_{\mathbf{o}_{t_c} \sim \mathbb{P}_r} \Big(D(\Gamma_{t_c}) \Big) + \lambda \mathop{\mathbb{E}}_{\tilde{\mathbf{o}}_{t_c} \sim \mathbb{P}_{\tilde{\mathbf{o}}_{t_c}}} \Big(\big(\|\nabla_{\tilde{\mathbf{o}}_{t_c}} D(\tilde{\Gamma}_{t_c})\|_2 - 1 \big)^2 \Big), \tag{5}$$

where λ is the gradient penalty weight. Moreover, one could also include an additional variety ℓ_2 loss between the generated and the ground-truth trajectory points, defined as a minimum ℓ_2 error of K randomly generated samples, as commonly done in GAN-based prediction [24–26].

Lastly, the discriminator network is discarded following the training phase, and only the generator is used for model evaluation. We used the efficient MobileNet as a generator, as discussed in Section 3.2.1, which allows for fast model inference that is well-suited for running onboard the SDVs.

4 EXPERIMENTS

In this section we present results of empirical evaluation. We first focus on the quantitative comparison of the proposed approach with the current state-of-the-art, followed by an analysis of case studies and the ablation study of the method.

Baselines: We evaluated the following baseline models, focusing on prediction approaches that include the current state-of-the-art social- and/or GAN-based components:

- Social-LSTM (S-LSTM), motion prediction method based on LSTM considering social interactions [45];
- Social-GAN (S-GAN), extension of Social-LSTM that also incorporates the GAN idea [24];
- Social-Ways (S-Ways), GAN-based approach that employs Info-GAN and a social attention layer [46];
- GAN-LSTM-no-scene, where the discriminator uses only trajectory inputs without scene context inputs, and trajectory points are encoded using an LSTM encoder, similar to Sophie [25] and MATF [39];
- GAN-LSTM-concat-scene, where the discriminator concatenates the scene context features with the trajectory features encoded using an LSTM encoder, similar to Social-BiGAT [26];
- GAN-no-scene and GAN-concat-scene, variants of the above two baselines where the trajectory points are encoded using a fully-connected layer in the discriminator instead of the recurrent architecture;
- SC-GAN: the proposed model trained with just the GAN loss as given in Equation (5);
- SC-GAN-ℓ_2: the proposed model trained with both GAN loss and a variety ℓ_2 loss with a weight of 10.

We used the available open-sourced code for the baseline approaches Social-GAN[1] [24], Social-LSTM[2] [45], and Social-Ways[3] [46]. The proposed SC-GAN model and the remaining baselines were implemented in TensorFlow [47], using the same generator network but varying the discriminator architectures. Note that for Social-BiGAT and the other above-mentioned baselines that do not have open-sourced code we used our own implementations of similar architectures. Unless otherwise mentioned, the implemented baseline models were trained end-to-end from scratch with just the Wasserstein GAN loss, defined in Equation (5). In order to satisfy the Lipschitz constraint of the Wasserstein GAN loss, we used a gradient penalty weight of 10. For the SC-GAN-ℓ_2 model, we computed the variety ℓ_2 loss by drawing multiple samples (we used three in our experiments) from the generator and using the sample with the lowest ℓ_2 loss to update the model parameters in the back-propagation phase. The raster size was set to 300×300 pixels with the resolution of $0.2m$ per pixel. For the proposed differentiable rasterizer we set the σ parameter from Equation (3) to $2m$. The models were trained with a per-GPU batch size of 64 and Adam optimizer [48], the learning rates (for both the generator and the discriminator) were tuned separately for each model using grid search, and we ran three discriminator steps for every generator step to balance the two modules. Note that for the S-Ways baseline we tried a large number of parameter settings and tweaking the original source code, however the results remained very suboptimal and thus we do not report them in the following sections.

[1] github.com/agrimgupta92/sgan, last accessed Jan. 2020
[2] github.com/quancore/social-lstm, last accessed Jan. 2020
[3] github.com/amiryanj/socialways, last accessed Jan. 2020

Table 1: Comparison of SC-GAN and the baselines, with the models trained using only the Wasserstein GAN loss

	mean over 3						min over 3		min over 20	
	ℓ_2 [m]		ORD [m]		ORFP [%]		ℓ_2 [m]		ℓ_2 [m]	
Method	**Avg**	**@4s**	**Avg**	**@4s**	**Avg**	**@4s**	**Avg**	**@4s**	**Avg**	**@4s**
GAN-no-scene	4.13	6.57	0.840	1.203	24.50	30.28	3.74	5.87	3.30	5.13
GAN-concat-scene	**2.35**	**5.62**	0.152	0.435	4.40	12.22	1.37	3.13	0.63	1.30
GAN-LSTM-no-scene	3.44	8.30	0.793	2.703	22.54	61.02	3.22	8.04	2.97	7.76
GAN-LSTM-concat-scene	3.08	6.10	0.258	0.732	6.88	17.40	1.94	4.13	1.11	2.58
SC-GAN	2.44	5.86	**0.085**	**0.204**	**2.11**	**5.66**	**1.29**	**2.95**	**0.58**	**1.20**

Data set We used a large-scale, real-world ATG4D data set discussed in [49]. The data set comprises 240 hours of data obtained by driving in various traffic conditions (e.g., varying times of day, days of the week, in several US cities). Each actor at each discrete tracking time step (every $0.1s$) amounts to a single data point, which consists of the current and past $0.4s$ of observed actor states (bounding box positions, velocities, accelerations, headings, and turning rates), used as a model input along with the surrounding high-definition map information, and the states for the future $4s$ are used as the ground-truth labels. The models were trained to predict trajectory points sampled at $2Hz$ to speed up the training of competing approaches. After removing static actors, our data set consisted of 7.8 million data points in total, and we used a 3/1/1 split to obtain train/validation/test data sets.

Evaluation metrics The models were evaluated using the standard average ℓ_2 displacement error (ADE) and final ℓ_2 displacement error (FDE) metrics, computed over the $4s$ prediction horizon. For the generative models we drew multiple K samples at inference time and evaluated both the mean and minimum errors over the samples. The minimum error metric measures the coverage and diversity of the predicted samples and is a standard approach used in other multimodal prediction work [23–26, 30, 39]. In addition to the standard ℓ_2 metrics, we also evaluated the models using two scene-compliance metrics introduced in [35]. More specifically, for each actor we first identify their likely *drivable regions* by traversing the directed lane graph in the map data starting from the actor's current position, and define a trajectory point as being *off-road* if it is outside of the drivable region computed in such a way. Then, the *off-road distance* (ORD) metric measures the distance from a predicted point to the nearest drivable region (defined to be 0 if inside the region), while the *off-road false-positive* (ORFP) metric measures the percentage of predicted trajectory points that were off-road for cases where the corresponding ground-truth trajectory point was inside the drivable region.

4.1 Quantitative results

We first compared the performance of methods that use only the GAN loss, in order to evaluate impact of discriminator designs on the GAN approaches in isolation from other effects. The results of quantitative evaluation of SC-GAN and the baselines are presented in Table 1. Unsurprisingly, GAN-no-scene shows very poor results on the two scene-compliance metrics since the discriminator does not make any use of the scene context. Compared to GAN-no-scene, the GAN-concat-scene model improves both the ℓ_2 errors and the scene-compliance metrics, showing that including the scene context plays a critical role in the discriminator. We can also see that using an LSTM-based decoder does not lead to improved results, and in fact the opposite conclusion can be made. Lastly, the proposed SC-GAN model had slightly worse mean ℓ_2 errors compared to GAN-concat-scene, however its min-over-K ℓ_2 errors were improved and it also reached significantly better results when considering scene-compliance metrics. In particular, SC-GAN reduced the off-road distance and off-road false positives by over 50% compared to GAN-concat-scene. This can be explained by the fact that SC-GAN explicitly projects the predicted trajectories into the raster space and stacks them with the scene context image, allowing the discriminator to more effectively identify non-compliant outputs. Figure 3 shows several case studies from the validation data illustrating the difference in scene compliance between SC-GAN and GAN-concat-scene predictions, detailed in the next section.

Table 2: Comparison of ℓ_2 ADE and FTE error metrics (in meters) of the proposed SC-GAN and the state-of-the-art trajectory prediction methods, with the models trained using both GAN and ℓ_2 losses; note that S-LSTM predicts only a single trajectory so its min-over-K is not reported

	mean@3		min@3		min@20	
Method	**Avg**	**@4s**	**Avg**	**@4s**	**Avg**	**@4s**
S-GAN [24]	3.01	7.66	2.36	5.94	1.93	4.77
S-LSTM [45]	2.93	5.17	-	-	-	-
SC-GAN-ℓ_2	**1.75**	**4.17**	**1.03**	**2.26**	**0.54**	**1.01**

In the above analysis we showed that the proposed approach outperformed the existing state-of-the-art GAN architectures for motion prediction, where only GAN losses were used during training. Next, we compared SC-GAN-ℓ_2 to the state-of-the-art trajectory prediction approaches using the full loss with an additional ℓ_2 loss term, shown in Table 2. First, we can see that the introduction of the explicit trajectory loss led to significantly improved performance, as seen when comparing the results of SC-GAN from Table 1 to the SC-GAN-ℓ_2 results. This is consistent with the findings of many other works that using additional task-dependent losses improves the performance of GAN models on supervised learning problems [24, 50]. In addition, we can see that the ℓ_2 errors of SC-GAN-ℓ_2 are much lower than either Social-GAN [24] or Social-LSTM [45]. Both average and final prediction error numbers decreased significantly, showing the benefits of the proposed GAN architecture and the novel differentiable rasterizer.

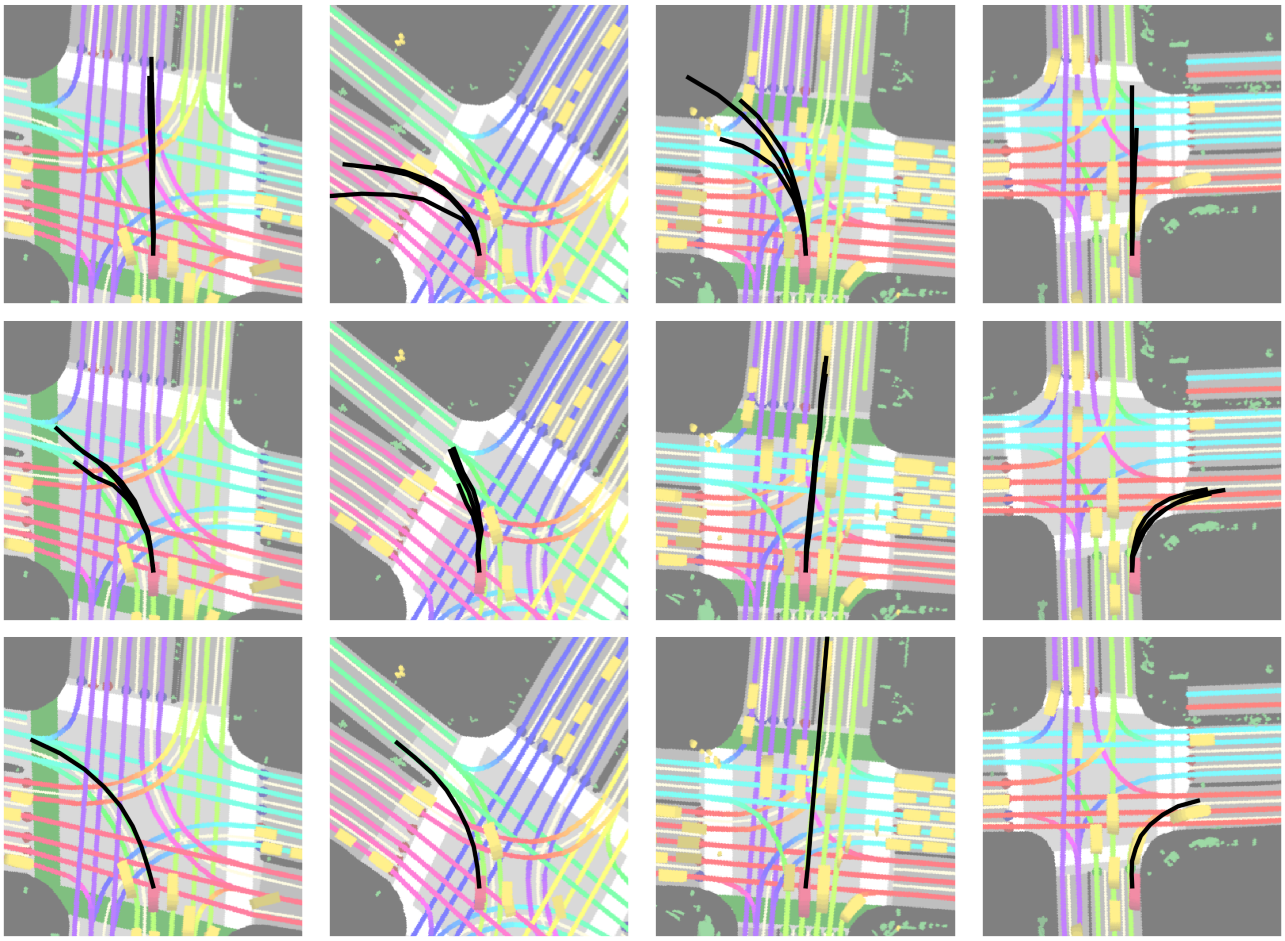

Figure 3: Qualitative results of GAN architectures conditioned on the raster image with and without using the differentiable rasterizer: top: GAN-concat-scene, middle: SC-GAN, bottom: ground-truth trajectories; we show a different traffic scenario in each column, plotting 3 sampled trajectories for each generative model

4.2 Qualitative results

In this section we analyze in detail several traffic scenarios commonly seen on the roads, as well as the outputs of different GAN architectures by sampling 3 trajectories. The scenarios include an actor going straight and taking a turn through an intersection, and an actor approaching an intersection on a straight road. We show the results in Figure 3, comparing SC-GAN and GAN-concat-scene trained using only the GAN loss. Note that these methods only differ in the way the generated trajectories and the rasters are combined at the input of the discriminator (i.e., using differentiable rasterizer or concatenation, respectively).

In case #1 the actor was approaching an intersection in a left-turn-only lane, as indicated by the scene context image. GAN-concat-scene predicted the actor to keep driving straight, which could be a reasonable prediction considering its past trajectory, however it is not scene-compliant for this scenario as the predictions entered the lane going in the opposite direction. On the other hand, the proposed SC-GAN correctly predicted the left turn that is compliant with the scene context. In case #2 the actor was already inside the intersection, performing an unprotected left turn. Trajectories output by GAN-concat-scene turned into the wrong lanes (most likely due to potentially too high heading change rate that was reported by the tracker), while trajectories output by SC-GAN correctly followed the turning lanes. In case #3 the actor was approaching an intersection in a straight-only lane, and GAN-concat-scene predicted the actor to turn left which is not scene-compliant. This happened because the tracked heading slightly tilted to the left, however we can see that SC-GAN still correctly predicted the going-straight trajectories. Lastly, in case #4 the actor was approaching an intersection in a right-turn-only lane. While the GAN-concat-scene model incorrectly predicted the actor to keep going straight, the proposed SC-GAN correctly predicted the compliant right turn. We can see that the proposed model that make use of the differentiable rasterizer outputs trajectories that are much more scene-compliant, leading to better overall prediction accuracy.

Table 3: Ablation study of several variants of the proposed SC-GAN approach

	mean over 3						min over 3		min over 20	
	ℓ_2 [m]		ORD [m]		ORFP [%]		ℓ_2 [m]		ℓ_2 [m]	
Method	**Avg**	**@4s**	**Avg**	**@4s**	**Avg**	**@4s**	**Avg**	**@4s**	**Avg**	**@4s**
SC-GAN-1channel	8.68	26.52	**0.040**	**0.033**	**0.98**	**1.23**	8.30	26.28	8.01	25.94
SC-GAN-MNet	3.82	11.18	0.723	3.068	7.21	22.43	2.62	8.15	1.82	6.08
SC-GAN-no-scene	3.79	7.28	0.58	1.16	18.38	33.62	3.52	6.76	2.27	4.61
SC-GAN	**2.44**	**5.86**	0.085	0.204	2.11	5.66	**1.29**	**2.95**	**0.58**	**1.20**

4.3 Ablation study

In this section we discuss the results of an ablation study of the proposed approach. The results are reported in Table 3, where we compare several variants of the SC-GAN model trained using only the GAN loss:

- SC-GAN-1channel: the rasterized trajectory occupancy grids $[\mathcal{G}^k_{t_c+\delta}, \ldots, \mathcal{G}^k_{t_c+T\delta}]$ are aggregated into a single channel via the `max` operator instead of being stacked in separate channels;
- SC-GAN-MNet: the discriminator uses the same CNN network as used in the generator (i.e., using MobileNet [41]) instead of DCGAN;
- SC-GAN-no-scene: the discriminator uses only the trajectory occupancy grids and state inputs, but does not make use of the scene context image.

Interestingly, we can see that the SC-GAN-1channel model has very low scene-compliance errors, yet high ℓ_2 errors. After visualizing its predictions we found the model often predicted static trajectories or trajectories with the points output in random orders. This was due to the fact that its discriminator is not able to distinguish between two trajectories with the same waypoints but in different orders because of the `max` aggregation, leading to suboptimal results. The SC-GAN-MNet model had worse metrics than the regular SC-GAN model, showing that DCGAN represents a better architecture for the discriminator. This is potentially because DCGAN is a more stable architecture as discussed in [42], leading to more useful gradients during training. Lastly, we observed that the SC-GAN-no-scene model also had much worse scene-compliance metrics than the regular SC-GAN model. This is unsurprising and is similar to the results given in Table 1, where we showed that conditioning the discriminator through scene raster input is a critical component for learning improved prediction models.

5 CONCLUSION

Motion prediction is one of the critical components of the self-driving technology, modeling future behavior and uncertainty of the tracked actors in SDV's vicinity. In this work we presented a novel GAN architecture to address this task, conditioned on the BEV raster images of the surrounding traffic actors and map elements. The main component of the model is the differentiable rasterizer, allowing projection of generated output trajectories into the raster space in a differentiable manner. This simplifies the task of the discriminator, leading to easier separation of observed and generated trajectories and improved performance of the model. We evaluated the proposed approach on a large-scale, real-world data set collected by a fleet of self-driving vehicles. Extensive qualitative and quantitative analysis showed that the method outperforms the current state-of-the-art in GAN-based motion prediction of the surrounding actors, producing more accurate and realistic trajectories.

REFERENCES

[1] D. Silver, A. Huang *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[2] D. Silver, T. Hubert *et al.*, "Mastering chess and shogi by self-play with a general reinforcement learning algorithm," *arXiv preprint arXiv:1712.01815*, 2017.

[3] E. J. Topol, *The patient will see you now: the future of medicine is in your hands*. Tantor Media, 2015.

[4] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: Lessons learned from the 2015 mscoco image captioning challenge," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 4, pp. 652–663, 2017.

[5] A. Z. Broder, P. Ciccolo *et al.*, "Search advertising using web relevance feedback," in *Proceedings of the 17th ACM Conference on Information and Knowledge Management*. ACM, 2008, pp. 1013–1022.

[6] G. Nuti, M. Mirghaemi, P. Treleaven, and C. Yingsaeree, "Algorithmic trading," *Computer*, vol. 44, no. 11, pp. 61–69, 2011.

[7] Y. N. Harari, "Reboot for the ai revolution," *Nature News*, vol. 550, no. 7676, p. 324, 2017.

[8] O. Gusikhin, N. Rychtyckyj, and D. Filev, "Intelligent systems in the automotive industry: applications and trends," *Knowledge and Information Systems*, vol. 12, no. 2, pp. 147–168, 2007.

[9] M. Hofmann, F. Neukart, and T. Bäck, "Artificial intelligence and data science in the automotive industry," *arXiv preprint arXiv:1709.01989*, 2017.

[10] D. Singh and M. Singh, "Internet of vehicles for smart and safe driving," in *2015 international conference on connected vehicles and expo (ICCVE)*. IEEE, 2015, pp. 328–329.

[11] K. N. Qureshi and A. H. Abdullah, "A survey on intelligent transportation systems," *Middle-East Journal of Scientific Research*, vol. 15, no. 5, pp. 629–642, 2013.

[12] A. Shaout, D. Colella, and S. Awad, "Advanced driver assistance systems-past, present and future," in *2011 Seventh International Computer Engineering Conference (ICENCO'2011)*. IEEE, 2011, pp. 72–82.

[13] S. Singh, "Critical reasons for crashes investigated in the national motor vehicle crash causation survey," National Highway Traffic Safety Administration, Tech. Rep. DOT HS 812 506, March 2018.

[14] NHTSA, "2017 fatal motor vehicle crashes: Overview," National Highway Traffic Safety Administration, Tech. Rep. DOT HS 812 603, October 2018.

[15] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," in *Advances in neural information processing systems*, 1989, pp. 305–313.

[16] ——, "Neural network perception for mobile robot guidance," Carnegie-Mellon Univ Pittsburgh PA Dept. of Computer Science, Tech. Rep., 1992.

[17] C. Urmson and W. Whittaker, "Self-driving cars and the urban challenge," *IEEE Intelligent Systems*, vol. 23, no. 2, pp. 66–68, 2008.

[18] J. Ziegler, P. Bender, M. Schreiber *et al.*, "Making bertha drive: An autonomous journey on a historic route," *IEEE Intelligent Transportation Systems Magazine*, vol. 6, pp. 8–20, 10 2015.

[19] C. Urmson, J. A. Bagnell, C. Baker, M. Hebert, A. Kelly, R. Rajkumar, P. E. Rybski, S. Scherer, R. Simmons, S. Singh *et al.*, "Tartan racing: A multi-modal approach to the DARPA urban challenge," 2007.

[20] C. Reinholtz, D. Hong, A. Wicks, A. Bacha, C. Bauman, R. Faruque, M. Fleming, C. Terwelp, T. Alberi, D. Anderson *et al.*, "Odin: Team VictorTango's entry in the DARPA urban challenge," in *The DARPA Urban Challenge*. Springer, 2009, pp. 125–162.

[21] N. Djuric, V. Radosavljevic, H. Cui, T. Nguyen, F.-C. Chou, T.-H. Lin, N. Singh, and J. Schneider, "Uncertainty-aware short-term motion prediction of traffic actors for autonomous driving," *arXiv preprint arXiv:1808.05819*, 2018.

[22] W. Luo, B. Yang, and R. Urtasun, "Fast and furious: Real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net," in *Proceedings of the IEEE CVPR*, 2018, pp. 3569–3577.

[23] H. Cui, V. Radosavljevic, F.-C. Chou, T.-H. Lin, T. Nguyen, T.-K. Huang, J. Schneider, and N. Djuric, "Multimodal trajectory predictions for autonomous driving using deep convolutional networks," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 2090–2096.

[24] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social gan: Socially acceptable trajectories with generative adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2255–2264.

[25] A. Sadeghian, V. Kosaraju, A. Sadeghian, N. Hirose, H. Rezatofighi, and S. Savarese, "Sophie: An attentive gan for predicting paths compliant to social and physical constraints," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1349–1358.

[26] V. Kosaraju, A. Sadeghian, R. Martín-Martín, I. Reid, H. Rezatofighi, and S. Savarese, "Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks," in *Advances in Neural Information Processing Systems*, 2019, pp. 137–146.

[27] A. Cosgun, L. Ma *et al.*, "Towards full automated drive in urban environments: A demonstration in gomentum station, california," in *IEEE Intelligent Vehicles Symposium*, 2017, pp. 1811–1818. [Online]. Available: https://doi.org/10.1109/IVS.2017.7995969

[28] M. Bansal, A. Krizhevsky, and A. Ogale, "Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst," *arXiv preprint arXiv:1812.03079*, 2018.

[29] F.-C. Chou, T.-H. Lin, H. Cui, V. Radosavljevic, T. Nguyen, T.-K. Huang, M. Niedoba, J. Schneider, and N. Djuric, "Predicting motion of vulnerable road users using high-definition maps and efficient convnets," in *Workshop on 'Machine Learning for Intelligent Transportation Systems' at Conference on Neural Information Processing Systems (MLITS)*, 2018.

[30] Y. Chai, B. Sapp, M. Bansal, and D. Anguelov, "Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction," *arXiv preprint arXiv:1910.05449*, 2019.

[31] J. Hong, B. Sapp, and J. Philbin, "Rules of the road: Predicting driving behavior with a convolutional model of semantic interactions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8454–8462.

[32] S. Yalamanchi, T.-K. Huang, G. C. Haynes, and N. Djuric, "Long-term prediction of vehicle behavior using short-term uncertainty-aware trajectories and high-definition maps," in *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2020.

[33] A. Jain, S. Casas, R. Liao, Y. Xiong, S. Feng, S. Segal, and R. Urtasun, "Discrete residual flow for probabilistic pedestrian behavior prediction," *arXiv preprint arXiv:1910.08041*, 2019.

[34] D. Ridel, N. Deo, D. Wolf, and M. Trivedi, "Scene compliant trajectory forecast with agent-centric spatio-temporal grids," *arXiv preprint arXiv:1909.07507*, 2019.

[35] M. Niedoba, H. Cui, K. Luo, D. Hegde, F.-C. Chou, and N. Djuric, "Improving movement prediction of traffic actors using off-road loss and bias mitigation," in *Workshop on 'Machine Learning for Autonomous Driving' at Conference on Neural Information Processing Systems*, 2019.

[36] 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.

[37] B. Dai, S. Fidler, R. Urtasun, and D. Lin, "Towards diverse and natural image descriptions via a conditional gan," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2970–2979.

[38] Y. Zhang, Z. Gan, K. Fan, Z. Chen, R. Henao, D. Shen, and L. Carin, "Adversarial feature matching for text generation," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 4006–4015.

[39] T. Zhao, Y. Xu, M. Monfort, W. Choi, C. Baker, Y. Zhao, Y. Wang, and Y. N. Wu, "Multi-agent tensor fusion for contextual trajectory prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 126–12 134.

[40] H. Cui, T. Nguyen, F.-C. Chou, T.-H. Lin, J. Schneider, D. Bradley, and N. Djuric, "Deep kinematic models for physically realistic prediction of vehicle trajectories," *arXiv preprint arXiv:1908.00219*, 2019.

[41] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation," *arXiv preprint arXiv:1801.04381*, 2018.

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

[43] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *arXiv preprint arXiv:1701.07875*, 2017.

[44] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," *CoRR*, vol. abs/1704.00028, 2017. [Online]. Available: http://arxiv.org/abs/1704.00028

[45] 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.

[46] J. Amirian, J.-B. Hayet, and J. Pettré, "Social ways: Learning multi-modal distributions of pedestrian trajectories with gans," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.

[47] M. Abadi, A. Agarwal, P. Barham, E. Brevdo *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. [Online]. Available: https://www.tensorflow.org/

[48] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[49] G. P. Meyer, A. Laddha, E. Kee, C. Vallespi-Gonzalez, and C. K. Wellington, "Lasernet: An efficient probabilistic 3d object detector for autonomous driving," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 677–12 686.

[50] G. Máttyus and R. Urtasun, "Matching adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8024–8032.