# DiversityGAN: Diversity-Aware Vehicle Motion Prediction via Latent Semantic Sampling

Xin Huang[1,2], Stephen G. McGill[1], Jonathan A. DeCastro[1],
Luke Fletcher[1], John J. Leonard[1,2], Brian C. Williams[2], Guy Rosman[1]

***Abstract*—Vehicle trajectory prediction is crucial for autonomous driving and advanced driver assistant systems. While existing approaches may sample from a predicted distribution of vehicle trajectories, they lack the ability to explore it – a key ability for evaluating safety from a planning and verification perspective. In this work, we devise a novel approach for generating realistic and diverse vehicle trajectories. We first extend the generative adversarial network (GAN) framework with a low-dimensional *approximate semantic* space, and shape that space to capture semantics such as merging and turning. We then sample from this space in a way that mimics the predicted distribution, but allows us to control coverage of semantically distinct outcomes. We validate our approach on a publicly available dataset and show results that achieve state-of-the-art prediction performance, while providing improved coverage of the space of predicted trajectory semantics.**

***Index Terms*—Intelligent Transportation Systems, Representation Learning, Computer Vision for Transportation**

## I. Introduction

VEHICLE trajectory prediction is crucial for autonomous driving and advanced driver assistant systems. For planning and verification, an ideal predictor must both accurately mimic the distribution of future trajectories and efficiently cover possible outcomes with little computational cost. While many recent efforts cater to accuracy [1]–[6], and some work addresses diversity and coverage [7]–[10], sampling efficiency still remains a challenge. This is important for vehicle trajectory prediction, since reasoning about the implications of predictions (e.g. with planning and collision checking) is expensive [11], [12] and becomes a limiting factor when seeking edge cases or extending the planning horizon.

Manuscript received: February 24, 2020; Revised May 12, 2020; Accepted June 6, 2020.

This paper was recommended for publication by Editor Youngjin Choi upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by Toyota Research Institute (TRI). This article solely reflects the opinions and conclusions of its authors and not TRI or any other Toyota entity.

[1]Toyota Research Institute, Cambridge, MA 02139, USA

[2]Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA 01239, USA xhuang@csail.mit.edu

Digital Object Identifier (DOI): see top of this page.

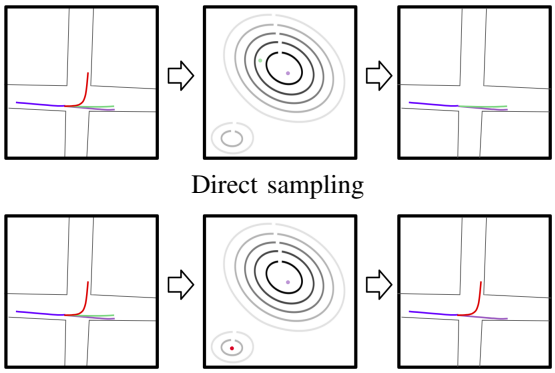



Fig. 1: When performing direct sampling, samples are taken uniformly from the distribution (middle), which fails to cover diverse behaviors. In latent semantic sampling, representative samples are taken, with weights associated with the distribution. In this way, a few samples can capture relevant semantic aspects, while ensuring consistency with the true prediction distribution.

This work is motivated by such a need to produce both accurate and diverse predictions using fewer samples.

Hybrid and discrete semantics have been crucial when reasoning about agent behavior. They manifest both in representations for planners [13] and in the design of tests for driving systems [14]. Semantic-level reasoning is crucial for attaining sample coverage across a diverse set of behaviors, as well as improving explainability. It is, however, not fully utilized in prediction and verification, in part because a discrete representation may limit the expressive power of the predictor, and defining a complete taxonomy for driver trajectories is difficult, especially when reasoning about multiple agents.

In this paper, we propose a model that is both accurate and diverse by incorporating a latent semantic layer into the trajectory generation. This layer represents high-level vehicle behaviors, matching discrete semantic outcomes where they are defined. Our construction of this layer does not require a complete taxonomy of maneuvers, and can extend easily to multiple definitions of semantics such as interaction types. We expect driving semantics to have a low effective dimensionality at every instance,

since a driver can perform only a few distinct maneuvers at any given moment. We illustrate our approach in Figure 1, where the goal is to produce diverse trajectory predictions and cover distinct outcomes. The top row shows traditional sampling, which fails to sample diverse behaviors efficiently. The bottom row demonstrates our latent semantic sampling technique, which captures both maneuvers that can be performed in the intersection.

We avoid the need for a taxonomy by shaping the intermediate layer via metric learning [15]. We train the latent semantic layer activations to match annotations of high-level labels when they exist. Distances in the semantic layer between two trajectories should be large if they represent different semantic labels, and should be small otherwise.

Finally, our proposed latent state affords some interpretation of the network, which is crucial in safety-critical tasks, such as autonomous driving. By tuning the high-level latent vector, our samples better cover the human intuition about diverse outcomes.

Our work has three main contributions: i) We extend a generative adversarial network to produce diverse and realistic future vehicle trajectories. We do so using a latent layer, which is shaped via metric learning to capture semantic context, as well as trajectory geometry. ii) We describe an efficient sampling method to cover the possible future actions and their likelihoods, which is important for safe motion planning and realistic behavior modeling in simulation. iii) We validate our approach on a publicly available dataset with vehicle trajectories collected in urban driving. Quantitative and qualitative results show our method successfully learns a semantically meaningful latent space that allows for generating diversified trajectories efficiently, while achieving state-of-the-art accuracies.

### A. Related Work

Our work relates to several topics in probabilistic trajectory prediction. Unlike deterministic alternatives [1], it allows us to reason about the uncertainty of driver's behavior. There are several representations that underlie reasoning about trajectories. [2], [4], [16]–[19] predict future vehicle trajectories as Gaussian mixture models, whereas [20] utilizes a grid-based map. Different from [17] that consolidates a large number of GMM outputs into succinct representations, we generate trajectory samples directly from an approximated distribution space, while improving the coverage of rare events.

For longer-term prediction horizons, additional context cues are needed from the driving environment. Spatial context, such as mapped lanes and scene information, not only indicates the possible options a vehicle may take (especially at intersections), but also improves the prediction accuracy, as vehicles usually follow lane centers closely [5], [21] or follow common movements in similar scene layouts [22]. Another important cue is social context, which allows for reasoning about interaction among agents [3], [6], [16]. Our method takes advantage of these two cues by feeding map data and nearby agent positions into our model, improving the accuracy of predictions over a few seconds.

Recently proposed generative adversarial networks (GANs) can sample trajectories by utilizing a vehicle trajectory generator and a discriminator that distinguishes real trajectories and generated trajectories [3], [23]. Despite their success, efficiently producing unlikely events, such as lane changes and turns, remains a challenge. These events are important to consider, as they can pose a significant risk and affect driving decisions.

Hybrid models [5], [24] are effective at producing distinct vehicle behaviors. They classify modes (or maneuvers) first before predicting the future positions. As such, they are restricted to cases where a fixed taxonomy of modes is well defined. In many cases, especially with multiple agents, a complete taxonomy of driving scenarios is hard to obtain. In [10], a set of trajectory modes are proposed to cover all possible motions, given vehicle dynamics. This provides sufficient coverage but suffers from the large number of samples predicted, as further reasoning about their implications (e.g., with collision checking) is expensive. In contrast, our method allows more sufficient coverage with fewer samples and handles more general cases involving with undefined semantics, including multi-vehicle interactions.

Many recent models use an intermediate representation [25], to improve performance and sample efficiency [26]. [18] utilizes a set of discrete latent variables to represent different driver intentions and behaviors. [27] explores the semantic features in the latent space of GANs, while InfoGAN in [28] successfully distills important features in the latent space based on an information bottleneck approach. In [23], a network based on InfoGAN is proposed to produce predictions that preserve multi-modality. In addition to GANs, [29] learns a disentangled latent representation in a variational autoencoder (VAE) framework to ground spatial relations between objects, while [30], [31] propose a conditional VAE latent variable framework to handle multi-modality in trajectory predictions. Unlike [23], [28], we use metric learning [15] to explicitly capture high-level notions, such as maneuvers and interactions, by training the latent space to match human similarity measures. It allows us to sample from the learned geometry to obtain distinct vehicle behaviors efficiently, without assuming the dissimilarity in trajectory space captures the semantic meaning, as in [17], [30]. In other domains, [32] diversifies samples via a potential field motivation for image generation GANs; [33] regularizes the generator by introducing multiple noise sources, to encourage multi-

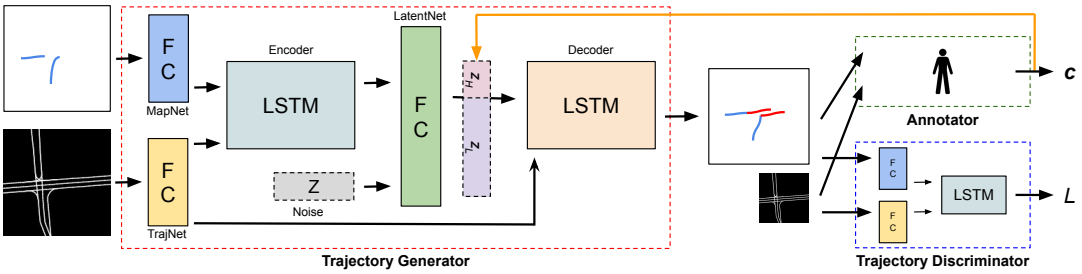



Fig. 2: Architecture diagram of prediction model. We shape the space of the intermediate vector $z_H$ to resemble a human's concept of semantic distances and then use it to select the samples that are fed to the decoder. Inputs include past vehicle trajectories and map information represented as arclength-parameterized curves (cf. Sec. II-C2).

modal behaviors in computer vision tasks; [9] explores diversity in policy through conjugated policies with a few samples in reinforcement learning.

Finally, our work has applications to sampling and estimation of rare events in support of verification tasks, which is its own active field, see [11], [34]–[36] and references therein. Efficient sampling is crucial for safety reasoning, as verification of safety properties for a given driving strategy requires numerous simulations over a large number of scene conditions and agent behaviors. The closest work to ours is [36], [37], which also propose sample-based estimation of probabilities. Our work focuses explicitly on sampling from diverse modes of behaviors, and effectively improves both search efficiency and representational power, allowing sufficient coverage with fewer simulations.

## II. Model

Here, we present the problem formulation and describe the model underlying our work, including loss functions and our proposed sampling procedure.

### A. Problem Formulation

The input to the trajectory prediction problem includes a sequence of observed vehicle trajectories $\mathbf{X} = X_1, X_2, \ldots, X_{t_{obs}}$, as well as the surrounding lanes denoted as $M$. Our goal is to predict a set of possible future trajectories $\hat{\mathbf{Y}} = \hat{Y}_{t_{obs}+1}, \hat{Y}_{t_{obs}+2}, \ldots, \hat{Y}_{t_{obs}+t_{pred}}$, where the observed future trajectories are denoted as $\mathbf{Y} = Y_{t_{obs}+1}, Y_{t_{obs}+2}, \ldots, Y_{t_{obs}+t_{pred}}$.

In the probabilistic setting, since many future trajectories are possible, the goal is to estimate the predicted probability distribution $P(\mathbf{Y}|\mathbf{X}, M)$. As $P(\mathbf{Y}|\mathbf{X}, M)$ often lacks a closed-form expression, many approaches use some form of sample generation techniques, including traditional ones such as MCMC and particle filters [38], planning-based approaches such as RRTs [39], and probabilistic generative networks.

### B. Model Overview

We now describe the network structure and sampling approach, as illustrated in Figure 2. The trajectory generator takes the past trajectory of target vehicles, a map of lane centerlines, and a noise sample, to produce samples of future trajectories. The discriminator identifies whether the generated trajectory is realistic.

**Semantic training cues** - In addition to the generator and discriminator networks, our model assumes a supervisory source for trajectory semantics. Both similarity votes [40] and discrete label vectors [41] have been used for metric learning of semantically meaningful spaces, from human annotations or from computational surrogates, such as classifiers. In driving, labels can include maneuvers, such as merging, turning, or interaction patterns such as giving right of way or turning at a four-way stop junction. For the purpose of our experiments, we hand-coded detectors for specific maneuvers, yielding three-logic (True/False/"undefined") values. These are coded as vectors $\mathbf{c}$ with elements $c_l \in \{-1, 1, \phi\}$. We chose three-valued logic since in some instances no Boolean choice makes sense – e.g., "lane keep" defined in [24] is ambiguous if the vehicle is approaching a Y-junction. This motivates a representation that avoids a single taxonomy of all road situations with definite semantic values. The values are used to compute the embedding loss in training time, indicated by the orange line in Fig. 2, with details discussed in Sec. II-E4.

### C. Trajectory Generator

The trajectory generator predicts realistic future vehicle trajectories, given inputs of the past trajectories and the map information. It embeds the inputs before sending them into a long short-term memory (LSTM) network encoder. The encoder output is combined with a noise vector generated from a standard normal distribution, and fed into a latent network that separates the combined information into a high-level vector and a low-level vector. The decoder, taking these two vectors, produces the trajectory samples.

*1) Trajectory Network:* A series of fully connected layers that embed spatial coordinates into a trajectory embedding vector [6].

*2) Map Network:* We represent nearby lanes via their polynomial coefficients, taken from the road point that is closest to the vehicle. We use arclength parameterization

[42] to resample the road and project it onto a second-order polynomial. The map network consists of a series fully connected layers that take the nearby lane coefficients as inputs and output a map embedding vector. In our experiments, this improves training efficiency while maintaining accuracy, compared to rasterizing the road network (e.g., in [5], [43]).

*3) Encoder:* An LSTM network that encodes the spatial and map embedding vectors from time steps 1 to $t_{obs}$ into an encoder hidden state vector.

*4) Latent Network:* Through a nonlinear fully connected network, the encoder hidden state is transformed, along with a standard Gaussian noise sample, into a latent state vector, $(z_H, z_L)$: $z_H \in \mathbb{R}^{d_H}$ represents high-level information, such as maneuvers, whereas $z_L \in \mathbb{R}^{d_L}$ represents fine trajectory information. We set $d_H \ll d_L$ so that $z_H$ can be sampled efficiently. As described in Section II-E, we encourage $z_H, z_L$ to be uncorrelated, and $z_H$ to separate semantically different trajectories. This representation disentangles semantic concepts from low-level trajectory information, similar to information bottlenecks [28], but is shaped by human notions of semantic similarity, as learned from the labels.

*5) RNN-based decoder:* An LSTM network takes $z_H$, $z_L$, and a map embedding vector, and generates a sequence of future vehicle positions.

### D. Trajectory Discriminator

An encoder converts the past trajectory, future predictions, and map information into a label $L$ = {fake, real}, where fake means a trajectory is generated by our predictor, while real means the trajectory is from data. The structure of the discriminator mirrors that of the trajectory encoder, except in its output dimensionality.

### E. Losses

Similar to [3], we assess the performance of our model using the average displacement error (ADE) and the final displacement error (FDE):

$$\mathcal{L}_{ADE}(\hat{Y}) = \frac{1}{t_{pred}} \sum_{t=t_{obs}+1}^{t_{obs}+t_{pred}} ||Y_t - \hat{Y}_t||_2 \tag{1}$$

$$\mathcal{L}_{FDE}(\hat{Y}) = ||Y_{t_{obs}+t_{pred}} - \hat{Y}_{t_{obs}+t_{pred}}||_2 \tag{2}$$

*1) Best prediction displacement loss:* Also as in [3], we compute the Minimum over N (MoN) losses to encourage the model to cover groundtruth options, while maintaining diversity in its predictions:

$$\mathcal{L}_{MoN} = \min_n \left( \mathcal{L}_{ADE} \left( \hat{Y}^{(n)} \right) \right), \tag{3}$$

where $\hat{Y}^{(1)}, \ldots, \hat{Y}^{(N)}$ are samples generated by our model. The loss, over $N$ samples from the generator, is computed as the average distance between the best predicted trajectories and observed future trajectories. Although minimizing MoN loss leads to a diluted probability density function compared to the groundtruth [44], we use it to show that our method can estimate an approximate distribution efficiently. We defer a different, more accurate, supervisory cue to future work.

*2) Adversarial loss:* We use standard binary cross entropy losses, $\mathcal{L}_{GAN,G}, \mathcal{L}_{GAN,D}$, to compute the loss between outputs from the discriminator and the labels. These losses are used to encourage diversity in predictions.

*3) Latent space regularization loss:* We encourage the two latent space components $\mathbf{z}_L, \mathbf{z}_H$ to be independent and normally distributed with a unit variance for each vector element. We do so by adding the two regularization terms,

$$\mathcal{L}_{\textbf{ind}} = \left( \sum_{i=1}^{d_H} \sum_{j=1}^{d_L} z_H^i z_L^j \right)^2, \; \mathcal{L}_{\textbf{lat}} = \begin{matrix} \|\Sigma_{z_H} - I\|_F^2 + \|\mu_{z_H}\|_F^2 + \\ \|\Sigma_{z_L} - I\|_F^2 + \|\mu_{z_L}\|_F^2 \end{matrix}, \tag{4}$$

where $\| \cdot \|_F^2$ denotes the Frobenius norm, $I$ denotes the identity operator with appropriate dimensions, and the means and variances are computed as empirical estimates at each batch.

*4) Embedding loss:* After enforcing $\mathbf{z}_H$ and $\mathbf{z}_L$ are independent vectors, we introduce an embedding loss to enforce the correlation between high-level latent vector $\mathbf{z}_H$ and prediction coding $\mathbf{c}$. Similar to [45], if two data samples have the same answer element for label $l$, we expect the differences in their high-level latent vectors to be small. On the other hand, if two predictions have different codings, we want to encourage the difference to be large. Note that it would be ideal to take human votes on the similarity between trajectories, and in this work, we use $\mathbf{c}$ as a surrogate to approximate their votes in order to mimic human notions of semantic similarity. The loss can be written as

$$\mathcal{L}_{\textbf{emb}} = \sum_{m=1,n=1}^{B} \sum_{l=1}^{s} \text{sign}\left( c_l^{(m)} c_l^{(n)} \right) ||\mathbf{z}_H^{(m)} - \mathbf{z}_H^{(n)}||_2, \tag{5}$$

where $B$ is batch size, $s$ is the number of defined labels, $c_l^{(m)}, c_l^{(n)}$ denote the label $l$ answers on examples $m, n$ respectively, and sign(·) = 0 if either label is $\phi$.

*5) Total loss:* In total, we combine the losses listed above together with appropriate coefficients.

$$\mathcal{L}, \mathcal{D} = \mathcal{L}_{GAN,D} \tag{6}$$

$$\mathcal{L}, \mathcal{G} = \lambda_1 \mathcal{L}_{MoN} + \lambda_2 \mathcal{L}_{GAN,G} + \lambda_3 \mathcal{L}_{\textbf{ind}} + \lambda_4 \mathcal{L}_{\textbf{lat}} + \lambda_5 \mathcal{L}_{\textbf{emb}} \tag{7}$$

### F. Sampling Approach

We now describe how we sample from the space of $z_H$ in Alg. 1. We generate an oversampled set of $N_{all}$ latent samples, and reduce them to a subset of $N$ representatives using the Farthest Point Sampling (FPS) algorithm [46], [47]. We store the nearest representative identity as we compute the distances, to augment the

FPS representatives with a weight proportional to their Voronoi cell weight, computed as the ratio of the number of cell samples to total sample size. This gives us a *weighted* set of samples that converges to the original distribution, but favors samples from distinct regions of space. FPS allows us to emphasize samples that represent distinct high-level maneuvers encoded in $z_H$.

**Algorithm 1** Semantic Sampling

**for all** $i = 1 \dots N_{all}$ **do**
  Sample from $z_{(i)} \sim Z$.
  Generate latent sample $(z_{H,(i)}, z_{L,(i)})$.
**end for**
Perform Farthest Point Sampling on $\{\mathbf{z}_{H,(i)}\}$ to obtain $N$ representative samples, $\{(\mathbf{z}_{H,(j)}, \mathbf{z}_{L,(j)})\}_{j=1}^{N}$
Compute Voronoi weights $w_j$ for each FPS sample.
Decode from $(\mathbf{z}_{H,(j)}, \mathbf{z}_{L,(j)})$ a full prediction $Y_{(j)}$.
Return $\{(Y_{(j)}, w_j)\}_{j=1}^{N}$

The samples cover (in the $\epsilon$-covering sense) the space of possible high-level choices. The high-level latent space is shaped according to human notions of semantic similarity. With this similarity metric shaping, FPS can leverage its 2-optimal distance coverage property [46], [47] in order to capture the majority of semantically different prediction roll-outs in just a few samples.[1]

## III. Results

In this section, we describe the details of our model and dataset, followed by a set of quantitative results against state-of-the-art baselines and qualitative results on accurate and diverse predictions.

### A. Model Details

The *Trajectory Network* utilizes two stacked linear layers with (32, 32) neurons. The *Map Network* uses four stacked linear layers with (64, 32, 16, 32) neurons. An LSTM with one layer and a hidden dimension of 64 forms both the *Encoder* and *Decoder* in the *Trajectory Generator*. The *Latent Network* fuses the Encoder output and a 10-dimensional noise vector. This network is composed of two individual linear layers with output dimensions of 2 and 72 for the high-level and low-level layers, respectively. The *Discriminator* is an LSTM with the same structure as the Generator's Encoder, followed by a series of stacked linear layers with dimensions of (64, 16, 1), activated by a sigmoid layer at the end. All linear layers in the Generator are followed by a batch norm, ReLU, and dropout layers. The linear layers in the Discriminator use a leakyReLU activation instead. The number of overall latent samples $N_{all}$ is 200, and the number of samples $N$ we use for the MoN loss is 5. The loss coefficients in Eq. (7) are selected to be 4, 1, 100, 2, 50, respectively.

The model is implemented in Pytorch and trained on a single NVIDIA Tesla V100 GPU. It takes approximately 45 milli-seconds (ms) to generate 200 latent samples, and approximately 0.1 ms and 1 ms to perform FPS and produce 5 prediction samples, respectively, which makes our model practical in real-time systems[2]. We use the Argoverse forecasting dataset [21] for training and validation, and select the trained model with the smallest MoN ADE loss on the validation set.

### B. Semantic Annotations

In order to test our embedding over a large scale dataset, we devised a set of classifiers for the data as surrogates to human annotations. They check for specific high-level trajectory features, and each of them outputs a ternary bit representing whether the feature exists, does not exist, or is unknown. The list of feature filters used in this paper includes: accelerate, decelerate, turn left, turn right, lane follow, and lane change.

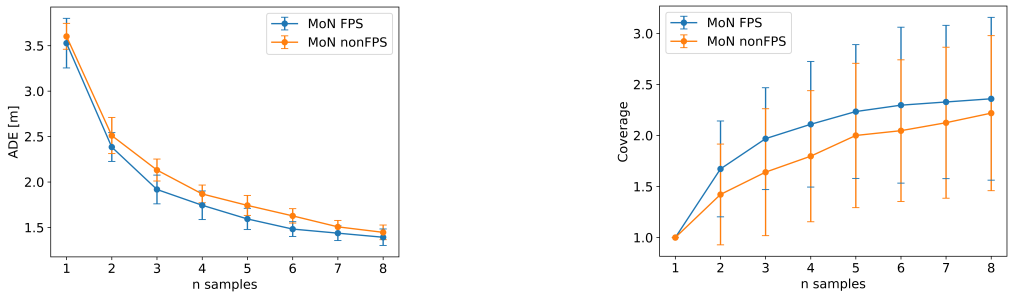
### C. Prediction Accuracy

Over 1 and 3 second prediction horizons, with $N = 5$ samples, we compute the MoN ADE (1) and FDE (2) losses, respectively. Compared to weighted losses, MoN is more suitable to downstream planning tasks as it identifies the worst case scenario when there exists risk. In addition, we introduce a few baseline models to demonstrate the prediction accuracy of our method. The first two baselines include a linear Kalman filter with a constant velocity (CV) model and with a constant acceleration (CA) model, respectively. We sample multiple trajectories given smoothing uncertainties. The third baseline is an LSTM-based encoder-decoder model [21], which produces deterministic predictions. The fourth baseline is SocialGAN [3], based on its open-source model. In order to show the efficacy of our approach in separating and shaping the latent space, we also introduce a vanilla GAN-based model that is similar to our network, but does not regularize the latent space using the embedding loss. The vanilla model has a few variants that take different input features, where social contains the positions of nearby agents and map contains the nearby lane information as described in II-C2.

The results are summarized in Table I. The first two rows indicate that physics-based models can produce predictions with reasonable accuracy. Using five samples, constant velocity (CV) Kalman Filter outperforms a deterministic deep model with results shown on the third row. While [49] shows that a physics-based model can outperform SocialGAN in pedestrian prediction, Table I

[1]We note that a modified FPS [48] can trade off mode-seeking with coverage-seeking when generating samples.

[2]The computational costs can be further optimized through parallelization and better memory transfers.




(a) (b)

Fig. 3: Accuracy and coverage comparisons between FPS sampling (blue) and direct sampling (orange) over 3 seconds with $N$ from 1 to 8. (a) Accuracy comparison using MoN as the metrics, where the gap between two curves indicates the improvement using FPS, especially when $N$ is from 2 to 6. (b) Coverage comparison, where $y$-axis measures the number of distinct discrete label codings extracted from the predicted trajectories. The gap indicates that FPS achieves better coverage of prediction options with smaller numbers of samples.

| | 1 Second | | 3 Seconds | |
|---|---|---|---|---|
| Model Name | ADE | FDE | ADE | FDE |
| Kalman Filter, CV | 0.51 | 0.79 | 1.63 | 3.62 |
| Kalman Filter, CA | 0.69 | 1.22 | 2.87 | 7.08 |
| LSTM Encoder-Decoder | 0.57 | 0.94 | 1.81 | 4.13 |
| SocialGAN | 0.47 | 0.69 | 1.72 | 3.49 |
| vanillaGAN | 0.42 | **0.62** | 1.55 | 3.09 |
| vanillaGAN+social | 0.44 | 0.66 | 1.68 | 3.04 |
| vanillaGAN+social+map | 0.44 | 0.63 | 1.34 | 2.75 |
| vanillaGAN(FPS)+social+map | 0.44 | 0.64 | 1.35 | 2.75 |
| DiversityGAN+social+map | **0.41** | 0.65 | 1.35 | 2.74 |
| DiversityGAN(FPS)+social+map | 0.44 | **0.62** | **1.33** | **2.72** |

TABLE I: MoN average displacement errors (ADE) and final displacement errors (FDE) of our method and baseline models with $N = 5$ samples.

demonstrates the performance gain of GANs in vehicle prediction, due to effectively longer and more complex maneuvers over the prediction horizon. From the ablation study, it is observed that the map features contribute more to long-horizon predictions than the social features. Additionally, our method is competitive compared to standard ones (e.g., vanillaGANs) and outperforms an off-the-shelf SocialGAN model trained on Argoverse, after regularizing the latent space using loss functions defined in Section II-E, while adding sample diversification, as shown in the rest of the section. Our results on the Argoverse test set ranked 5th in the Argoverse competition and earned Honorable Mention in the Machine Learning for Autonomous Driving (ML4AD) workshop at Neurips 2019.

### *D. Latent Space Learning*

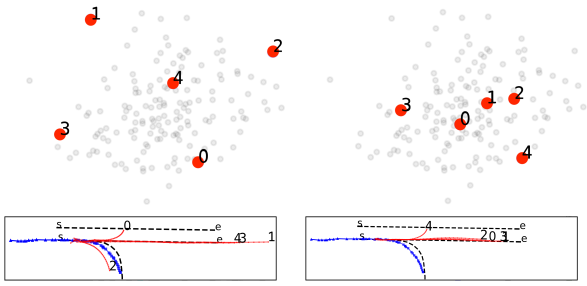
To demonstrate the latent space is learned to be semantically meaningful, we measure the k-nearest neighbor (kNN) entropy [50] of the label distribution in the latent space, using both our method and a vanilla GAN model without an embedding loss.

The kNN entropy is computed as follows: First, we generate a large number of $S$ latent space samples and obtain $S'$ local neighborhoods by sub-sampling $S'$ points, and $k$-nearest-neighbors sets. We then generate the trajectories and their labels for each neighborhood, and compute the entropy of the labels, which measures how well we decouple distinct labels (low entropy indicates decoupled labels). We select $S = 1000, S' = 30, k = 40$ to provide ample coverage in the low-dimensional latent space.

In the validation dataset, our learned latent space has an average entropy of 0.55, with a standard deviation of 0.33. In contrast, the latent space from a vanilla GAN has an average entropy of 1.45, with a standard deviation of 0.29. The percentage of the majority vote label within each neighborhood is 71.53% using our method and 50.94% using a vanilla GAN. This suggests that our method successfully shapes the latent space, so as to locally disentangle semantic space. In the remainder of this section, we complete our claim that the learned space is semantically meaningful by showing that far away samples are generating trajectories with distinct labels through FPS.

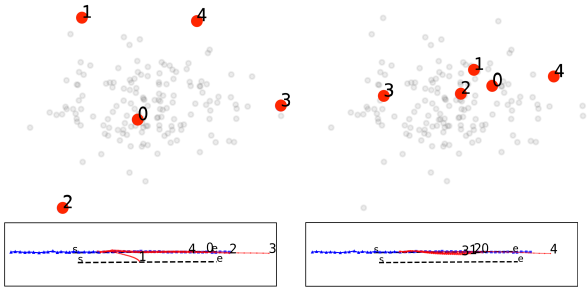
### *E. Latent Space Sampling*

To show the effectiveness of our latent sampling approach, we measure the MoN loss with and without the FPS method. We test using a challenging subset of the Argoverse validation dataset that filters out straight driving with constant velocity scenarios, resulting in a trajectory distribution that emphasizes rare events in the data. As indicated in Figure 3(a), when the number of samples increases, the prediction loss using FPS drops faster compared to direct sampling. We note the improvement is larger in the regime of 2-6 samples, where reasoning about a full roll-out of multiple hypotheses is still practical in real-time systems, and we obtain an improvement of 8%. Beyond the improved accuracy, the proposed method is able to sample the additional modes of the distribution of trajectories, which is validated in Figure 3(b), where we compare the coverage of distinct maneuver codes by sampled predictions and show that FPS has better coverage with smaller numbers of sam-



FPS Direct Sampling

(a) FPS provides accurate coverage of observed future trajectory by generating rare turning samples.



FPS Direct Sampling

(b) FPS covers a low-likely lane change event that matters for decision making for the ego car.

Fig. 4: Latent space samples (numbered dots) are shown above their predicted trajectory representations. Blue: observed past and future trajectories. Red: predicted trajectory samples. Black: lane centers. Left column: samples selected by FPS and their associated predictions (numbers indicate sampling order). Right column: samples using direct sampling, which only cover high likelihood events.

ples. We further demonstrate this advantage with a small number of samples in Section III-F.

### *F. Qualitative Examples*

We show how FPS can be used to improve both prediction accuracy and diversity coverage, by illustrating two examples in Figure 4.

In the first example, as illustrated in Figure 4(a), our method, as described in Algorithm 1, first generates $N_{all} = 200$ samples in grey, and selects $N = 5$ samples using FPS (highlighted on the left column). By selecting samples that are farther away, FPS is able to produce rare events, such as a right turn, as labeled in 2, which matches with the observed future trajectory and thus improves the prediction accuracy. On the other hand, direct sampling (highlighted on the right column) tends to sample points from more dense regions, which lead to high likelihood events. We show two additional challenging examples in Figure 5(a), where FPS is able to reduce the prediction error, by covering turning events when the vehicle is approaching an off-ramp and a full intersection, respectively.

In the second example in Figure 4(b), although our method predicts rare events that do not improve displacement losses compared to direct sampling, they are still important for decision making and risk estimation. Although the target vehicle is most likely to go forward, it is useful for our predictor to cover lane change behavior, as labeled in 1, even with a low likelihood, since such prediction could help avoid a possible collision if our ego car is driving in the right lane. Similarly, in other two examples, as shown in Figure 5(b), our method produces events, such as merging and turning, which are unlikely to happen, but are important to consider for robust and safe decision making by the ego car.

## IV. Conclusion

We propose a vehicle motion prediction method that caters to both prediction accuracy and diversity. We divide a latent variable into a learned semantic-level part, which embeds discrete options, and a low-level part, which encodes fine trajectory information. The learned geometry in the semantic part allows efficient sampling of diverse trajectories. The method is demonstrated to achieve state-of-the-art prediction accuracy, while efficiently obtaining trajectory coverage. Future work includes adding more complicated semantic labels such as interactions, and exploring other sampling methods.

## References


[1] A. Houenou, P. Bonnifait, V. Cherfaoui, and W. Yao, "Vehicle trajectory prediction based on motion model and maneuver recognition," in *IROS*. IEEE, 2013, pp. 4363–4369.

[2] J. Wiest, M. Höffken, U. Kreßel, and K. Dietmayer, "Probabilistic trajectory prediction with Gaussian mixture models," in *IVS*. IEEE, 2012, pp. 141–146.

[3] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social GAN: Socially acceptable trajectories with generative adversarial networks," in *CVPR*, 2018, pp. 2255–2264.

[4] X. Huang, S. McGill, B. C. Williams, L. Fletcher, and G. Rosman, "Uncertainty-aware driver trajectory prediction at urban intersections," in *ICRA*, 2019, pp. 9718–9724.

[5] 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 *ICRA*. IEEE, 2019, pp. 2090–2096.

[6] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human trajectory prediction in crowded spaces," in *CVPR*, 2016, pp. 961–971.

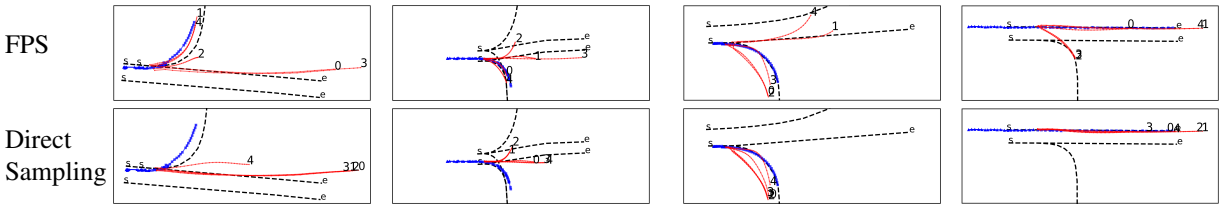
[7] Z.-W. Hong, T.-Y. Shann, S.-Y. Su, Y.-H. Chang, T.-J. Fu, and C.-Y. Lee, "Diversity-driven exploration strategy for deep reinforcement learning," in *NeurIPS*, 2018, pp. 10 489–10 500.

[8] T. Gangwani, Q. Liu, and J. Peng, "Learning self-imitating diverse policies," in *ICLR*, 2019.

[9] A. Cohen, X. Qiao, L. Yu, E. Way, and X. Tong, "Diverse exploration via conjugate policies for policy gradient methods," in *AAAI*, vol. 33, 2019, pp. 3404–3411.

[10] T. Phan-Minh, E. C. Grigore, F. A. Boulton, O. Beijbom, and E. M. Wolff, "CoverNet: Multimodal behavior prediction using trajectory sets," *arXiv:1911.10298*, 2019.

[11] E. Schmerling and M. Pavone, "Evaluating trajectory collision probability through adaptive importance sampling for safe motion planning," in *RSS*, Cambridge, Massachusetts, 2017.




(a) Predicting diversified events helps reduce prediction error in challenging scenarios.

(b) Predicting merging and turning events enables robust and safe decision making for the ego car.

Fig. 5: Predictions of rare events in complicated driving scenarios help improve both accuracy (a) and diversity (b). Top to bottom: FPS and direct sampling with $N = 5$ trajectory samples.


[12] A. Wang, X. Huang, A. Jasour, and B. Williams, "Fast risk assessment for autonomous vehicles using learned models of agent futures," in *RSS*, 2020.

[13] H. X. Li and B. C. Williams, "Generative planning for hybrid systems based on flow tubes." in *ICAPS*, 2008, pp. 206–213.

[14] K. Esterle, V. Aravantinos, and A. Knoll, "From Specifications to Behavior: Maneuver Verification in a Semantic State Space," in *IVS*, Jun. 2019, pp. 2140–2147.

[15] K. Q. Weinberger and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," *JMLR*, vol. 10, no. Feb, pp. 207–244, 2009.

[16] B. Ivanovic and M. Pavone, "The Trajectron: Probabilistic multi-agent trajectory modeling with dynamic spatiotemporal graphs," in *ICCV*, 2019.

[17] A. Zyner, S. Worrall, and E. Nebot, "Naturalistic driver intention and path prediction using recurrent neural networks," *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[18] Y. C. Tang and R. Salakhutdinov, "Multiple futures prediction," in *NeurIPS*, 2019.

[19] Y. Chai, B. Sapp, M. Bansal, and D. Anguelov, "Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction," in *CoRL*, 2019.

[20] B. Kim, C. M. Kang, J. Kim, S. H. Lee, C. C. Chung, and J. W. Choi, "Probabilistic vehicle trajectory prediction over occupancy grid map via recurrent neural network," in *ITSC*. IEEE, 2017, pp. 399–404.

[21] M.-F. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, D. Wang, P. Carr, S. Lucey, D. Ramanan *et al.*, "Argoverse: 3D tracking and forecasting with rich maps," in *CVPR*, 2019, pp. 8748–8757.

[22] M. Huynh and G. Alaghband, "Trajectory prediction by coupling scene-LSTM with human movement LSTM," in *International Symposium on Visual Computing*. Springer, 2019, pp. 244–259.

[23] J. Amirian, J.-B. Hayet, and J. Pettré, "Social ways: Learning multi-modal distributions of pedestrian trajectories with GANs," in *CVPR Workshops*, 2019.

[24] N. Deo and M. M. Trivedi, "Multi-modal trajectory prediction of surrounding vehicles with maneuver based LSTMs," in *IVS*, 2018, pp. 1179–1184.

[25] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE-TPAMI*, vol. 35, no. 8, pp. 1798–1828, 2013.

[26] Y. Bengio, G. Mesnil, Y. Dauphin, and S. Rifai, "Better mixing via deep representations," in *ICML*, 2013, pp. 552–560.

[27] Y. Shen, J. Gu, X. Tang, and B. Zhou, "Interpreting the latent space of GANs for semantic face editing," in *CVPR*, 2020.

[28] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets," in *NIPS*, 2016, pp. 2172–2180.

[29] Y. Hristov, D. Angelov, M. Burke, A. Lascarides, and S. Ramamoorthy, "Disentangled relational representations for explaining and learning from demonstration," in *CoRL*, 2019.

[30] Y. Yuan and K. Kitani, "Diverse trajectory forecasting with determinantal point processes," *ICLR*, 2020.

[31] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron++: Multi-agent generative trajectory forecasting with heterogeneous data for control," *arXiv:2001.03093*, 2020.

[32] T. Unterthiner, B. Nessler, C. Seward, G. Klambauer, M. Heusel, H. Ramsauer, and S. Hochreiter, "Coulomb GANs: provably optimal nash equilibria via potential fields," in *ICLR*, 2018.

[33] D. Yang, S. Hong, Y. Jang, T. Zhao, and H. Lee, "Diversity-sensitive conditional generative adversarial networks," in *ICLR*, 2019.

[34] A. Bhatia and E. Frazzoli, "Incremental search methods for reachability analysis of continuous and hybrid systems," in *In Hybrid Systems: Computation and Control*, 2004, pp. 142–156.

[35] J. Bucklew, *Introduction to rare event simulation*. Springer Science & Business Media, 2013.

[36] M. O'Kelly, A. Sinha, H. Namkoong, J. Duchi, and R. Tedrake, "A scalable risk-based framework for rigorous autonomous vehicle evaluation," 2019.

[37] M. Koren and M. Kochenderfer, "Efficient autonomy validation in simulation with adaptive stress testing," in *ITSC*, 2019, pp. 4178–4183.

[38] V. Karasev, A. Ayvaci, B. Heisele, and S. Soatto, "Intent-aware long-term prediction of pedestrian motion," in *ICRA*, 2016, pp. 2543–2549.

[39] G. Aoude, J. Joseph, N. Roy, and J. How, "Mobile agent trajectory prediction using bayesian nonparametric reachability trees," in *Infotech@ Aerospace 2011*, 2011, p. 1512.

[40] I. Borg and P. Groenen, *Modern Multidimensional Scaling: Theory and Applications*. Springer, 2005.

[41] C. Strecha, A. Bronstein, M. Bronstein, and P. Fua, "Ldahash: Improved matching with smaller descriptors," vol. 34, no. 1, pp. 66–78, 2011.

[42] M. P. Do Carmo, *Differential geometry of curves and surfaces: revised second edition*. Courier Dover Publications, 2016.

[43] A. Amini, G. Rosman, S. Karaman, and D. Rus, "Variational end-to-end navigation and localization," in *ICRA*, 2019, pp. 8958–8964.

[44] L. A. Thiede and P. P. Brahma, "Analyzing the variety loss in the context of probabilistic trajectory prediction," in *ICCV*, 2019, pp. 9954–9963.

[45] G. Rosman, L. Paull, and D. Rus, "Hybrid control and learning with coresets for autonomous vehicles," in *IROS*, 2017, pp. 6894–6901.

[46] T. F. Gonzalez, "Clustering to minimize the maximum intercluster distance," *Theor. Comp. Sci.*, vol. 38, pp. 293 – 306, 1985.

[47] D. S. Hochbaum and D. B. Shmoys, "A best possible heuristic for the k-center problem," *Math. Oper. Res.*, vol. 10, no. 2, pp. 180–184, May 1985.

[48] M. Volkov, G. Rosman, D. Feldman, J. W. Fisher, and D. Rus, "Coresets for visual summarization with applications to loop closure," in *ICRA*, 2015, pp. 3638–3645.

[49] C. Schöller, V. Aravantinos, F. Lay, and A. Knoll, "What the constant velocity model can teach us about pedestrian motion prediction," *RA-L*, vol. 5, no. 2, pp. 1696–1703, 2020.

[50] D. Lombardi and S. Pant, "Nonparametric k-nearest-neighbor entropy estimator," *Physical Review E*, vol. 93, no. 1, p. 013310, 2016.