Context-Aware Timewise VAEs for Real-Time Vehicle Trajectory Prediction

Pei Xu , Jean-Bernard Hayet , *Member, IEEE*, and Ioannis Karamouzas

Abstract—Real-time, accurate prediction of human steering behaviors has wide applications, from developing intelligent traffic systems to deploying autonomous driving systems in both real and simulated worlds. In this letter, we present ContextVAE, a context-aware approach for multi-modal vehicle trajectory prediction. Built upon the backbone architecture of a timewise variational autoencoder, ContextVAE observation encoding employs a dual attention mechanism that accounts for the environmental context and the dynamic agents' states, in a unified way. By utilizing features extracted from semantic maps during agent state encoding, our approach takes into account both the social features exhibited by agents on the scene and the physical environment constraints to generate map-compliant and socially-aware trajectories. We perform extensive testing on the nuScenes prediction challenge (Caesar et al., 2023), Lyft Level 5 dataset (Houston et al., 2023) and Waymo Open Motion Dataset (Ettinger et al., 2023) to show the effectiveness of our approach and its state-of-the-art performance. In all tested datasets, ContextVAE models are fast to train and provide high-quality multi-modal predictions in real-time.

Index Terms—Vehicle trajectory prediction, multimodal prediction, timewise variational autoencoder.

I. INTRODUCTION

REAL-TIME trajectory prediction for traffic agents is fundamental for the development of self-driving systems and intelligent traffic control solutions [4]. In this letter, we particularly focus on the problem of predicting the multi-modal behavior of vehicles in challenging scenes populated with heterogeneous neighbors, including pedestrians, cyclists and other vehicles. Accurate prediction for traffic agents requires the predictive model to fully take into account the external (observable) and internal (intentions) characteristics of the agents, in addition to the contextual influence from the agent's neighbors and the scene environment [5]. While numerous approaches have been proposed over the past few years for context-aware vehicle trajectory prediction [6], most existing solutions focus on the model's prediction accuracy ignoring the requirements for real-time inference performance [7]. To address this issue, we introduce ContextVAE as a real-time approach for high-fidelity vehicle trajectory prediction.

Our work follows the line of recent variational autoencoder (VAE) methods that have been shown to achieve state-of-the-art performance in agent trajectory prediction tasks involving homogeneous, human-human interaction settings [8], [9], [10]. Despite their success, VAE-based methods tend to underperform when tasked to predict vehicle trajectories in complex, heterogeneous traffic scenes [1]. To that end, we explore recent trends in *timewise* VAE architectures [11] to better capture the highly dynamic and multi-modal nature of agent interactions. In addition, we propose a novel, *dual-attention* mechanism for observation encoding that simultaneously accounts for the environmental context (roads, lanes, etc.) extracted from the local map features and the social context extracted from the perceived neighbors' states. To enable real-time inference, our approach uses an RNN-based encoding of the perceived neighbors [9] instead of plotting their trajectories on rasterized maps [7]. Meanwhile, the extracted environmental features directly participate in the agent's state encoding process, leading to a fully, context-aware scheme for observation encoding and fast and accurate inference of the agent's navigation strategy.

This letter proposes ContextVAE, a real-time approach for vehicle trajectory prediction conditioned on short-term observations. Overall, we make the following contributions:

- We introduce a novel, dual-attention architecture for RNN-based observation encoding that accounts for environmental (map) and social (neighbors) features in a *unified* manner. We show that such a unified scheme is paramount to achieve high-quality predictions compared to the decoupled schemes that current VAE-based solutions employ, where environmental and social features are encoded separately [9], [10], [12], [13].
- We combine our observation encoding scheme with a VAE architecture where latent variables are sampled timewisely to better model the uncertainty in agent decision making [11]. The resulting ContextVAE approach is generic as it does use any post-processing techniques or prior assumptions about the agents' motion, leading to state-of-the-art performance compared to existing solutions with similar attributes. In addition, ContextVAE models are orders of magnitude faster to train due to their low complexity and small memory footprint, and exhibit real-time prediction performance (< 30 ms).
- We demonstrate the effectiveness and computational efficiency of ContextVAE on three heterogeneous vehicle datasets: the nuScenes prediction challenge [1], Lyft Level
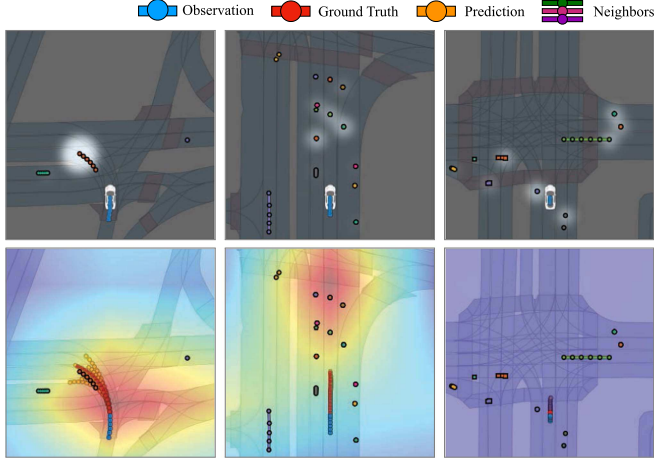
Fig. 1. Trajectory predictions on the Lyft dataset using our proposed approach. ContextVAE employs a timewise VAE architecture and a context-aware observation encoding scheme that accounts for environmental (map) and social (neighbor) features in a unified manner. Top: Corresponding attention that the target vehicle is paying to its neighbors within a radius of 30 m. Bottom: Predicted trajectories and map regions with high activation in the attention model. Red regions are considered more visually salient than blue regions.

5 dataset [2] and Waymo Open Motion Dataset [3]. ContextVAE provides high-fidelity predictions while allowing for real-time inference and can serve as a strong baseline for future work of vehicle trajectory prediction.

II. RELATED WORK

There has been a lot of prior work on agent trajectory prediction focusing on human-human interactions [8], [14], [15], [16], [17] and autonomous driving applications [18], [19]. We refer to recent excellent surveys for an overview [6], [20]. Below, we briefly focus on methods exploiting both environmental and social contexts for *vehicle* trajectory prediction.

Leveraging Environmental Context: To produce map-compliant predictions, early works directly plot the observed trajectories (agent of interest and neighbors) on the rasterized scene maps, performing prediction mainly by using convolutional neural networks (CNNs) to extract map features without fully exploiting the agents' states [2], [5], [7], [21], [22], [23]. More recent works exploit both the environmental context and vectorized neighbors' states as inputs to the prediction model to better synthesize the agent's motion states and context [9], [10], [24], [25], [26], [27], [28]. Beyond rasterized maps, some approaches leverage the graph representation of high-definition maps to avoid information loss during rasterization [29], [30], [31], [32]. Systems such as VectorNet [29] and AutoBots [25] have shown that CNNs applied on rasterized maps can be replaced by graph neural networks (GNNs) using graph-represented maps. Instead of modeling the overall environment, QCNet [33] employs a polygon-based representation of the environment and denotes the scene via embedding each agent-polygon pair.

Generating Sequential Predictions: CoverNet [21] and Multi-Path [22] generate trajectories by picking them from a predefined set and cast the generative task as a classification problem. WIMP [34], Trajectron++[9] and DiversityGAN [27] use recurrent neural networks (RNNs) to perform prediction sequentially.

MTP [23], AutoBots [25] and LaPred [35] output fixed-length predicted trajectories at once. WIMP [34], CXX [36], TNT [37], GoalNet [26] first predict goal positions and then generate trajectories between the last observed position and the predicted goals. IMAP [12] and M2I [38] do not predict trajectories only for target agents but perform prediction sequentially for all the agents. P2T [24] and PGP [39] use reinforcement learning, with a reward-based model upon which a policy is trained to perform prediction.

Handling Multi-Modality: To model the uncertainty during human driving and account for multi-modal decision making, Trajectron++[9] and BiTrap [10] follow DESIRE [8] and exploit a conditional variational autoencoder (CVAE) architecture where the prior on the latent variable is modeled as a mixture of Gaussians. MTP [23], AutoBots [25], and MultiPath++ [40] use Gaussian mixtures as well but with an encoder-decoder architecture.DiversityGAN [27] takes the architecture of generative adversarial networks to produce stochastic results. LaPred [35] and LanceGCN [30], employs K predictor networks with random initialization to produce K predictions parallelly. Our approach uses a CVAE [11] as its backbone architecture but the latent variables are introduced timewisely [11] to better capture the uncertainty in agent decision making. The final distribution is modeled as a sequentially conditioned Gaussian distribution instead of a mixture model.

Introducing Prior Knowledge: Recent works introduce prior knowledge [41] into the process of vehicle trajectory prediction by assuming that vehicles' possible movement patterns. For example, CXX [36], WIMP [34], PGP [39], LaPred [35], LaneRCNN [42], HOME [43] and GOHOME [32] select interesting paths traversing the road graph or specific lanes in the graph as the basis to generate numeric coordinates. TNT [37] and GoalNet [26] select goal positions based on the road graph and then perform goal-directed prediction. Though achieving impressive results, such approaches heavily rely on the assumption that vehicles will move along the road graph, which is not always true in real-life scenarios. Thus, we focus on comparisons with works that do not rely on this assumption.

III. APPROACH

A. Problem Formulation

Given a local, T-frame observation $\mathcal{O}_i^{1:T}$ gathered from a target agent i, we seek to estimate the agent's trajectory predictive distribution over the future H frames, i.e. $p(\mathbf{x}_i^{T+1:T+H}|\mathcal{O}_i^{1:T})$, where $\mathbf{x}_i^{T+1:T+H}$ are the agent's future coordinates. To perform context-aware prediction, we use observations that include both the N_i observed neighboring agent positions and the local environmental context information $\mathbf{M}_i^t$, i.e. $\mathcal{O}_i^t := \{\mathbf{M}_i^t, \{\mathbf{x}_j^t\}_j\}$, where $j = 1, \ldots, N_i$ and $||\mathbf{x}_j^t - \mathbf{x}_i^t|| \leq r_i$ given r_i as the observation radius of agent i. We model $\mathbf{M}_i^t$ as a vector of features extracted from rasterized semantic maps via a CNN module, as shown in Fig. 2. To train our models in a scene-invariant way, we do not predict directly the global coordinates $\mathbf{x}_i^{T+1:T+H}$ but rather the *displacement* distribution $p(\mathbf{d}_i^{T+1:T+H}|\mathcal{O}_i^{1:T})$, where $\mathbf{d}_i^t := \mathbf{x}_i^t - \mathbf{x}_i^{t-1}$ is the displacement between two frames. Future trajectories are reconstructed as $\mathbf{x}_i^{T+\tau} = \mathbf{x}_i^T + \sum_{t=T+1}^{T+\tau} \mathbf{d}_i^\tau$ by sampling $\mathbf{d}_i^\tau$ from the predicted distributions.
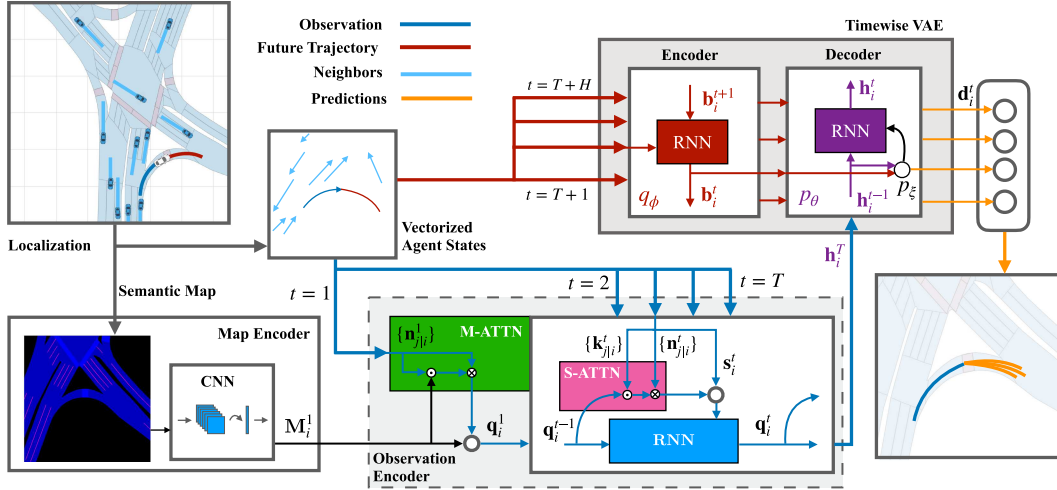
Fig. 2. ContextVAE backbone architecture is a timewise VAE shown in the gray block at the upper right. The proposed map encoding module is linked to the backbone timewise VAE architecture through the context-aware observation encoder (dashed box), which processes the vectorized agent states in a unified way with the map features. M-ATTN and S-ATTN represent the map and social attention respectively, where $\odot$ is the dot product operator, $\otimes$ is the element-wise multiplication operator, and $\bigcirc$ is the concatenation operator. Red parts for posterior estimation are used only during training.

B. Model Architecture and Training Objective

The backbone architecture of ContextVAE is a timewise VAE with a conditional prior p_θ and a posterior q_ϕ estimated bidirectionally (see Fig. 2). The training objective is to maximize a timewise evidence lower bound:

$$
\mathbb{E}_i \left[\frac{1}{H} \sum_{t=T+1}^{T+H} \mathbb{E}_{\mathbf{z}_i^t \sim q_\phi(\cdot|\mathbf{b}_i^t, \mathbf{h}_i^{t-1})} [\log p_\xi(\mathbf{d}_i^t | \mathbf{z}_i^t, \mathbf{h}_i^{t-1}) \right.
$$
$$
\left. - D_{KL} \left[q_\phi(\mathbf{z}_i^t | \mathbf{b}_i^t, \mathbf{h}_i^{t-1}) || p_\theta(\mathbf{z}_i^t | \mathbf{h}_i^{t-1})]] \right], \tag{1}
$$

where $\mathbf{z}_i^t$ is the latent variable sampled timewisely, and $\mathbf{b}_i^t$ and $\mathbf{h}_i^{t-1}$ are the RNN hidden states, which are updated backward in the encoder and forward in the decoder, respectively. We parameterize the output distribution p_ξ, the posterior q_ϕ and the prior p_θ as Gaussians, using neural networks.

The timewise VAE decoder hidden state $\mathbf{h}_i^t$ for $t > T$ is updated recurrently based on the sampled $\mathbf{z}_i^t$ and the displacement $\mathbf{d}_i^t$, i.e. $\mathbf{h}_i^t = \overrightarrow{g}(\psi_{\mathbf{zd}}(\mathbf{z}_i^t, \mathbf{d}_i^t), \mathbf{h}_i^{t-1})$. While feeding $\mathbf{z}_i^t$ allows to produce the necessary variations to match the predictive distribution, the displacement $\mathbf{d}_i^t$ allows to enforce some continuity in the produced displacements across time. The initial state $\mathbf{h}_i^T = \psi_{\mathbf{h}}(\mathcal{O}_i^{1:T})$ encodes the observation, where $\psi_{\mathbf{zd}}$ and $\psi_{\mathbf{h}}$ are embedding networks. This leads to a prior conditioned by the context observation $\mathcal{O}_i^{1:T}$ (see Section III-C). The timewise VAE encoder hidden state $\mathbf{b}_i^t$ is updated backwards, i.e. $\mathbf{b}_i^t = \overleftarrow{g}(\mathcal{O}_i^t, \mathbf{b}_i^{t+1})$, given the ground-truth observation $\mathcal{O}_i^t$ at the future frame $t > T$ and a zero initial state $\mathbf{b}_i^{T+H+1}$.

At training, the posterior q_ϕ over the latent variable $\mathbf{z}_i^t$ at timestep t uses the information collected from the *future* part of the ground-truth trajectory from $T + H$ down to t, through the backward encoding $\mathbf{b}_i^t$. This leverages the whole trajectory availability at training time for better extracting the agent's strategy. It also uses the forward encoding $\mathbf{h}_i^{t-1}$ evaluated from 1 to $t-1$, thus considering the *whole* trajectory $\mathcal{O}_i^{1:T+H}$. The

displacement distribution $p_\xi(\mathbf{d})$ relies on this latent variable $\mathbf{z}_t$, drawn from $q_\phi(\cdot|\mathbf{b}_i^t, \mathbf{h}_i^{t-1})$ at training, and on $\mathbf{h}_i^{t-1}$.

At inference, predictions are generated through latents drawn from the prior: $\mathbf{d}_i^t \sim p_\xi(\cdot|\mathbf{z}_i^t, \mathbf{h}_i^{t-1})$ where $\mathbf{z}_i^t \sim p_\theta(\cdot|\mathbf{h}_i^{t-1})$, relying on the historical observation $\mathcal{O}_i^{1:T}$ only.

C. Context-Aware Observation Encoding

We introduce a *unified* scheme for observation encoding, where the environmental context extracted from the semantic map and the social context extracted from the observed agent states are exploited simultaneously. Our scheme leverages a dual attention mechanism that combines *map attention* with *social attention* as shown at the bottom of Fig. 2, and is based on the intuition that humans account for environmental features dynamically and instantaneously during driving, rather than encoding these features one after the other.

Following prior literature [9], [10], [11], [17], our encoder employs an RNN structure g (blue box in Fig. 2) that encodes the agent state sequentially. The RNN is updated as $\mathbf{q}_i^{t+1} = g(\mathcal{O}_i^t, \mathbf{q}_i^t)$, where, from the local observation $\mathcal{O}_i^t$, we use the agent's states $\mathbf{s}_i^t$, including its velocity and acceleration, and the observed neighbors' state $\{\mathbf{n}_{j|i}^t\}$, including the relative position and velocity of each neighbor j to the agent i. The final $\mathbf{q}_i^T$ is used to initialize the decoder's hidden states in the timewise VAE backbone, i.e., $\psi_{\mathbf{h}}(\mathcal{O}_i^{1:T}) \equiv \mathbf{q}_i^T$ (Section III-B).

To account for how the environmental context influences the agent's steering actions, we initialize the hidden state $\mathbf{q}_i^1$ using features extracted from the local semantic map, rather than encoding the map and the agent's state independently. However, instead of directly plugging the map features into the observation encoder, we introduce a *map attention* (M-ATTN) mechanism and initialize the encoder via

$$
\mathbf{q}_i^1 = \text{CONCAT}\left(\mathbf{M}_i^1, \text{M-ATTN}(\{\mathbf{n}_{j|i}^1\})\right), \tag{2}
$$

where

$$\text{M-ATTN}(\{\mathbf{n}_{j|i}^1\}) \triangleq \text{ATTN}(\mathbf{M}_i^1, \{\mathbf{n}_{j|i}^1\}, \{\mathbf{n}_{j|i}^1\})$$
$$= \sum_j w_{j|i} f_{\mathbf{n}_{\text{val}}^1}(\mathbf{n}_{j|i}^1). \tag{3}$$

Here, $\mathbf{M}_i^1$ denotes vectorized features extracted from agent i's local map at the *first* observation frame, and $f.$ are embedding networks. $\mathbf{M}_i^1$, $\{\mathbf{n}_{j|i}^1\}$ and $\{\mathbf{n}_{j|i}^1\}$ are the query, key, and value vectors, respectively, of the attention mechanism. The attention weights $w_{j|i}$ are obtained through dot-product operations between $\mathbf{M}_i^1$ and $\mathbf{n}_{j|i}^1$ followed by a softmax operation, i.e. $w_{j|i} = \text{SOFTMAX}(f_{\mathbf{M}^1}(\mathbf{M}_i^1) \odot f_{\mathbf{n}_{\text{key}}^1}(\mathbf{n}_{j|i}^1))$. This M-ATTN mechanism can help identify neighbors close to the target agent but with less influence in the given context, e.g., vehicles spatially close to the agent but on a lane that is inaccessible to it. We refer to Section IV-F and Fig. 1 for qualitative results and to Section IV-D for quantitative analysis.

During agent state encoding, we synthesize the social context represented by the neighbors' states $\{\mathbf{n}_{j|i}^t\}$ via a *social attention* (S-ATTN) mechanism as:

$$\text{S-ATTN}(\{\mathbf{n}_{j|i}^t\}) \triangleq \text{ATTN}(\mathbf{q}_i^t, \{\mathbf{k}_{j|i}^t\}, \{\mathbf{n}_{j|i}^t\})$$
$$= \sum_j \omega_{j|i} f_{\mathbf{n}_{\text{val}}}(\mathbf{n}_{j|i}^t), \tag{4}$$

where, $\mathbf{q}_i^t$, $\{\mathbf{k}_{j|i}^t\}$, and $\{\mathbf{n}_{j|i}^t\}$ correspond to the query, key and value vectors of the attention, and each weight $\omega_{j|i}$ of the attention graph is computed as the dot product between $\mathbf{q}_i^t$ and $\mathbf{k}_{j|i}^t$ after obtaining their fixed-length embeddings. Here, $\mathbf{k}_{j|i}^t$ denotes the social features exhibited by the neighbor j and observed by the agent i. As introduced in [11], we include the distance, bearing angle and minimal predicted distance from agent i to j as the social features $\mathbf{k}_{j|i}^t$.

By choosing a proper size of the local map, we can have $\mathbf{M}_i^1$ covering the whole trajectory of the agent during the observation and prediction horizon from $t = 1$ to $t = T + H$. This allows us to sidestep introducing $\mathbf{M}_i^t$ timewisely, and update the hidden state in the RNN g by representing $\mathcal{O}_i^t$ as $\mathcal{O}_i^t \equiv \text{CONCAT}(\mathbf{s}_i^t, \text{S-ATTN}(\{\mathbf{n}_{j|i}^t\}))$. This way, we perform fast observation encoding directly on the environmental and social contexts after one pass of the map encoding module, and avoid the computationally expensive processing a sequence of rasterized maps. We refer to Section IV-D for our model's real-time prediction performance.

As shown in Fig. 2, we exploit rasterized semantic maps to capture contextual information and extract the related features $\mathbf{M}_i^1$ using a CNN module. Note that there is no constraint about the form of $\mathbf{M}_i^1$: It can also be represented using graphs processed by a GNN [29], [30], [32], [37], [39], or using a lane attention block [35], [36]. Since rasterization is the common way to generate semantic maps for contextual information and since most open-source vehicle datasets do not provide an API to directly generate road graphs, in our experiments we rely on CNNs to extract contextual features from rasterized maps.
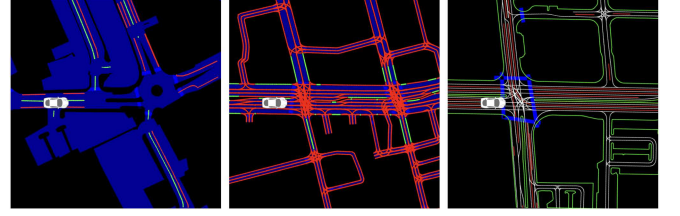

Fig. 3. Examples of rasterized semantic maps for (from left to right): *nuScenes*, *Lyft*, and *Waymo*. Blue identifies drivable areas and crosswalks; green identifies road edges; red identifies lane dividers. The drawn cars identify the first observed position of the agent under prediction and give the map local origin and x-axis. The cars are not part of the rasterized maps.

IV. EXPERIMENTS

We evaluate our approach on vehicle trajectory prediction tasks using the nuScenes prediction challenge (*nuScenes*) [1], Lyft Level 5 Prediction Dataset (*Lyft*) [2] and Waymo Open Dataset (*Waymo*) [3]. These datasets handle heterogeneous types of agents, including various vehicles, cyclists, and pedestrians. We focus our prediction on vehicles' trajectories and all other types of agents are considered as neighbors.

A. Setup

1) Implementation Details: We use a rasterized, 224×224 local semantic map to represent the environmental context. The map is translated and rotated such that the vehicle under prediction heads towards the positive x-axis and stands at the 122nd row and 51st column of the map in the 1st frame of observation. To be consistent, the coordinates of all agents are expressed using the same local system. Fig. 3 shows examples of the extracted maps for the three datasets. Since each dataset has its own semantic map definition, the generated rasterized maps are slightly different. We refer to the supplementary material for details on semantic map rasterization.

2) Evaluation Metrics and Baselines: We use the minimum average displacement error over k predictions (minADE$_k$) and the minimum final displacement error over k predictions (minFDE$_k$) to assess a model's performance. We consider both the most likely prediction ($k = 1$) and the top-5 predictions ($k = 5$) as key metrics (in meters). We provide comparisons to three deterministic baselines: *Constant Velocity* assumes target vehicles always move at the velocity defined between the last two observed frames; *Constant Lane* assumes target vehicles keep their last observed speed but always along the lane where they were at the last observed frame; *Kalman Filter* applies an extended Kalman filter while assuming constant longitudinal speed and heading direction. We also consider state-of-the-art, data-driven baselines for each dataset. As ContextVAE does not use prior knowledge to constrain the movement of vehicles, we limit our comparisons to approaches that do not restrict the accessibility of vehicles via maps or refine the predicted trajectories by post-processing.

B. Quantitative Results

1) nuScenes: We use the official training and validation sets of *nuScenes* prediction challenge to perform model training and performance evaluation. An observation window of up to 2 s is used and the prediction horizon is 6 s. As the data was recorded at 2 FPS, this leads to a 12-frame prediction and a varying

TABLE I
PERFORMANCE ON NUSCENES PREDICTION CHALLENGE

nuScenes (Validation Set)	minADE$_k$ (m)		minFDE$_k$ (m)	
	$k = 1$	$k = 5$	$k = 1$	$k = 5$
Constant Velocity	4.61	-	11.21	-
Constant Lane	5.45	-	12.73	-
Kalman Filter	4.17	-	10.99	-
Trajectron++	4.08	2.41	9.67	5.63
P2T	3.82	1.86	8.95	4.08
AutoBots-Ego	3.86	1.70	8.89	3.40
ContextVAE	**3.54**	**1.59**	**8.24**	**3.28**

The best performance is highlighted in bold.

TABLE II
PERFORMANCE ON LYFT LEVEL 5 PREDICTION DATASET

Lyft	minADE$_k$ (m)		minFDE$_k$ (m)	
	$k = 1$	$k = 5$	$k = 1$	$k = 5$
Constant Velocity	0.70	-	1.64	-
Constant Lane	0.78	-	2.04	-
Kalman Filter	0.60	-	1.30	-
SAMPP [12]	0.39	-	0.83	-
IMAP [12]	0.28	-	0.65	-
Trajectron++	0.38	0.26	0.89	0.57
ContextVAE	**0.24**	**0.16**	**0.54**	**0.32**

The best performance is highlighted in bold.

TABLE III
PERFORMANCE ON WAYMO OPEN DATASET

Waymo (Validation Set)	minADE$_k$ (m)		minFDE$_k$ (m)	
	$k = 1$	$k = 5$	$k = 1$	$k = 5$
Constant Velocity	2.04	-	5.25	-
Constant Lane	2.54	-	5.85	-
Kalman Filter	1.99	-	4.07	-
SAMPP [12]	1.26	-	2.80	-
IMAP [12]	0.97	-	2.03	-
Trajectron++	0.88	0.56	2.37	1.41
MotionCNN	0.83	0.40	1.99	0.81
M2I	0.67	0.42	1.60	0.85
ContextVAE	**0.59**	**0.30**	**1.49**	**0.68**

The best performance is highlighted in bold.
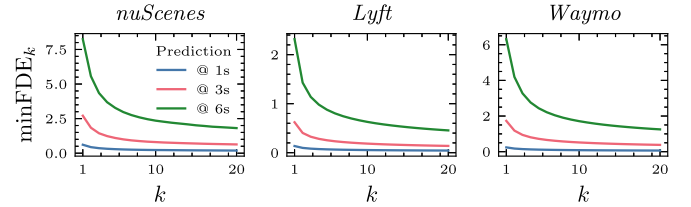

Fig. 4. Minimal final displacement error (in meters) as a function of the number k of predicted trajectories for different prediction horizons.

observation window between 2 and 5 frames. Additionally to the heuristic baseline models, we introduce three other baselines for comparison. The Trajectron++ [9] model is trained from scratch with two known bugs fixed as indicated in the official repository. P2T [24] is tested using their provided pre-trained model. Since no pre-trained models are provided, we train the AutoBots-Ego model using the official implementation. These three baselines exploit CNN modules for map encoding. We report the results in Table I. Among the considered baselines, *Constant Velocity*, *Constant Lane* and *Kalman Filter* perform the worst. While *Constant Lane* ensures that the predicted position is always along a lane, it exhibits worse performance than *Constant Velocity* as it ignores the possibility that the target vehicle would change lanes. As recently published works, P2T and AutoBots-Ego work significantly better than Trajectron++, with AutoBots-Ego outperforming P2T. Our approach not only achieves the best performance on all four metrics, but is also very fast to train (see Section IV-C).

2) Lyft: This dataset is much larger than *nuScenes* and has more than 1,000 hours of data with 170,000 scenes. We use only the first 16,265 training scenes, each scene being about 24 s long. For evaluation, we use the full validation dataset (16,220 scenes). We downsample the data from 10 FPS to 5 FPS and train models with an observation window of 1 s and prediction horizon of 3 s. In Table II, we report the performance obtained by our approach and two new baselines: SAMPP and IMAP [12]. These are deterministic models that rely on lane graphs rather than rasterized semantic maps. Trajectron++ is introduced as a multimodal baseline, trained using the same data as ours. As can be seen, IMAP achieves better deterministic results (minADE$_1$/minFDE$_1$) compared to Trajectron++, though these results are slightly worse compared to Trajectron++'s multimodal performance. Our approach outperforms IMAP and Trajectron++ on both the deterministic and multimodal results,

with an improvement around 15% on minADE$_1$/minFDE$_1$ and around 40% on minADE$_5$/minFDE$_5$.

3) Waymo: This dataset has two types of training data formats, consisting of 20 s and 9 s sequential data, respectively. We use the 9 s data for training, which contains 487,002 scenes, while the validation set has 44,097 scenes. In addition to performing downsampling from 10 FPS to 5 FPS, we apply the data filter from the *Lyft* dataset to filter out invalid data. Similar to our *Lyft* implementation we use a 1 s window for observation and a 3 s horizon for prediction. Table III shows the performance of our approach along with two new baselines: MotionCNN [5] and M2I [38]. We run the pre-trained models from the two baselines and keep the first 3 s prediction for evaluation. As shown in Table III, MotionCNN and M2I outperform Trajectron++ significantly on minADE$_5$/minFDE$_5$. M2I also brings a large improvement on the most likely prediction. Compared with these strong baselines, ContextVAE achieves more than 15% improvement overall, and a 29% and 20% improvement on minADE$_5$ and minFDE$_5$, respectively.

The default Waymo motion prediction challenge setup uses an 8 s prediction horizon with $k = 6$. The performance of our model trained for 8 s on the *interactive* validation set for vehicle-only target agents is: minADE$_6 = 1.59$ and minFDE$_6 = 3.67$. As a comparison, the Waymo LSTM baseline achieves minFDE$_6 = 6.07$ for 8 s prediction [44], and M2I [38] reports minFDE$_6 = 5.49$.

C. Computational Performance

We evaluated the prediction performance of ContextVAE with several popular CNN modules for map encoding. In Table IV, we report the corresponding results along with the number of parameters and inference time taken by each CNN module. The inference time refers to the time needed to perform $k = 5$ predictions for one batch of 32 input samples. We pick a batch size of 32 based on the observed statistics in the three tested

TABLE IV
PREDICTION ERRORS WITH DIFFERENT CNN MODULES FOR MAP FEATURE EXTRACTION

CNN Module	Infer. Time	# of Params.	nuScenes				Lyft				Waymo			
			minADE$_k$ (m)		minFDE$_k$ (m)		minADE$_k$ (m)		minFDE$_k$ (m)		minADE$_k$ (m)		minFDE$_k$ (m)	
			$k=1$	$k=5$	$k=1$	$k=5$	$k=1$	$k=5$	$k=1$	$k=5$	$k=1$	$k=5$	$k=1$	$k=5$
ResNet18	0.029s	11.7M	**3.543**	**1.586**	**8.244**	**3.277**	0.247	0.165	0.548	0.324	0.600	0.306	1.532	0.709
ResNet152	0.102s	60.2M	3.728	1.728	8.773	3.657	**0.244**	**0.164**	**0.544**	**0.321**	0.598	0.306	1.530	0.708
EfficientNet-B0	0.029s	5.3M	3.715	1.740	8.733	3.676	0.246	0.165	0.548	0.325	0.585	0.299	1.491	0.686
MobileNet-V2	0.025s	3.5M	3.780	1.738	8.888	3.671	0.251	0.165	0.560	0.321	**0.585**	**0.298**	**1.492**	**0.684**

The best performance is highlighted in bold.

TABLE V
MODEL ABLATION ON WAYMO

S-ATTN	MAP	M-ATTN	minADE$_k$ (m)		minFDE$_k$ (m)	
			$k=1$	$k=5$	$k=1$	$k=5$
-	-	-	0.68	0.45	1.78	1.11
✓	-	-	0.65	0.43	1.71	1.06
✓	Indie	-	0.63	0.37	1.68	0.85
✓	Integrated	-	0.62	0.33	1.65	0.70
✓	Integrated	✓	**0.59**	**0.30**	**1.49**	**0.68**

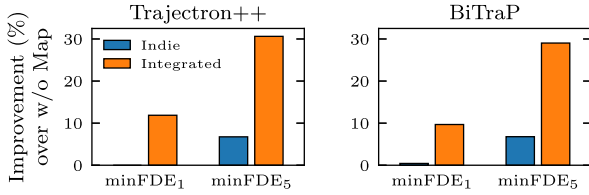The best performance is highlighted in bold.

Fig. 5. Performance improvement when using map context on *Waymo* dataset.

datasets, where an ego-vehicle has on average 10 ± 7 agents within an observation radius of 30 m. The reported time is measured on a machine equipped with a V100 GPU and averaged over 10,000 test trials. As can be seen in the table, there are no large differences regarding minADE $_k$/minFDE$_k$ among the tested CNN modules on *Lyft* and *Waymo*. We can therefore safely choose a small module, like EfficientNet-B0 or MobileNet-V2, for the sake of faster inference. In *nuScenes*, the best performance is obtained with ResNet18, but the differences among the CNN modules are larger. A possible reason is the small size of the *nuScenes* dataset, which makes the model more sensitive to the choice of CNN modules. For a complete list of all the tested CNN modules, we refer to the supplementary material. Overall, using ResNet18 on *nuScenes* and *Lyft* and MobileNet-V2 on *Waymo* can meet the real-time inference requirements, while guaranteeing high prediction accuracy. While ContextVAE only provides marginalized predictions for each agent, the overall time that it needs to predict the trajectories for all agents on a scene is less than 30 ms. This is orders of magnitude faster than existing state-of-the-art approaches on the leaderboards of motion prediction challenges (e.g., PGP [39] needs 407 ms on *nuScenes*) and of the same order with approaches that focus on computational efficiency such as PredictionNet [7] at the cost though of prediction quality.

Besides its real-time inference performance, ContextVAE is also very fast to train due to its lightweight architecture that relies on RNNs to perform encoding/decoding sequentially. It takes only about 2 hours to train our *nuScenes* model on a single V100 GPU for 80 epochs. In contrast, Trajectron++ needs

around nine and a half hours for the same number of epochs. The more complex P2T and Autobots-Ego models require more than 2 and 3 days respectively to finish training. We note that other state-of-the-art solutions are equally slow to train due to the large number of parameters and high complexity of their models, involving transformer-based architectures [31], [33], multi-stage training phases [39], [45], ensemble training [32], [40], etc. While such solutions may slightly outperform ContextVAE, they are typically finetuned to a specific dataset by leveraging a number of post-processing techniques and hard-coded assumptions In contrast, our approach can achieve high-fidelity predictions in real-time for a range of datasets.

D. Sensitivity Analysis

1) Prediction Horizon and Choice of k: Intuitively, increasing the prediction horizon leads to more uncertainty, making prediction harder. This is consistent with the results shown in Fig. 4, where larger horizons lead to higher minFDE values. For a given horizon, an important question is how many prediction samples k are enough to lead to a sufficiently low error? For a 1 s prediction horizon (blue line), the error converges around $k = 10$. For a 3 s horizon (red line), $k = 20$ roughly leads to convergence. However, for a 6 s horizon (green line), there is still a decreasing trend at $k = 20$. We refer to the supplementary material for more numeric results of model performance with different prediction horizons.

2) Model Ablation: In Table V, we perform ablation studies on *Waymo*, related to our proposed unified observation encoding scheme and its underlying dual attention mechanism (M-ATTN in (3) and S-ATTN in (4)). When neither map nor social context is exploited, the neighbors' states are simply added together for observation encoding. When only S-ATTN is used, the observation encoder $\mathbf{q}_i^1$ is initialized without map features $\mathbf{M}_i^1$ but social attention is introduced (4). When the map is introduced independently ("Indie"), $\mathbf{M}_i^1$ is concatenated with the output $\mathbf{q}_i^T$ of the encoder instead of being used to initialize $\mathbf{q}_i^1$. In our proposed observation encoding scheme, the map features are integrated into the encoder ("Integrated") where $\mathbf{q}_i^1$ is initialized with $\mathbf{M}_i^1$ by concatenation (without M-ATTN) or by the attention mechanism in (2) (with M-ATTN). As can be seen from Table V, S-ATTN brings about a 4% improvement over the basic timewise VAE backbone. Handling the semantic map independently from the social context brings an extra 3% improvement on the deterministic predictions and around 15% on the multimodal predictions. Integrating the map into the encoder without M-ATTN results in even better performance, highlighting its potential over Indie maps. M-ATTN further boosts the performance with an improvement around 35% on minADE$_5$/minFDE$_5$, and around 15% on minADE$_1$/minFDE$_1$
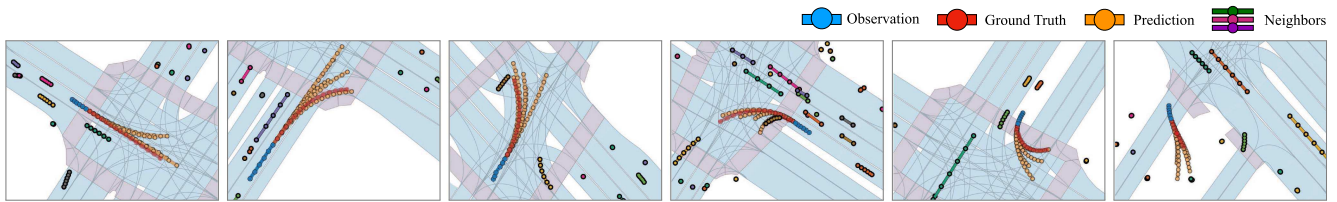
Fig. 6. Qualitative examples from *Lyft*. Observed trajectories are shown in blue. The ground-truth future trajectories are shown in red. Five predictions are shown in orange. Other colored dots and lines denote stationary and moving neighbors respectively.

compared to the basic backbone architecture. Similar conclusions can be drawn from *nuScenes* and *Lyft*, as described in the supplementary material.

E. Integrated Versus Independent Maps for Observation Encoding

To further highlight the effectiveness of using Integrated over Indie maps for RNN-based observation encoding, we consider experiments with additional VAE-based approaches that by default decouple environmental from social context. Fig. 5 shows the corresponding results for Trajectron++ and BiTraP [10]. As can be seen, models with Integrated maps significantly improve the performance of Trajectron++ and Bitrap on both deterministic and multimodal predictions. In contrast, Indie maps do not provide any noticeable performance gain on deterministic predictions and only bring a small improvement on multimodal ones. We refer to the supplementary material for complete results on all the tested datasets.
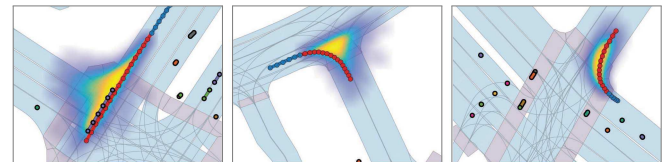
F. Case Studies

Fig. 6 shows predictions obtained with ContextVAE on the *Lyft* dataset. As can be seen, using maps makes the model aware of the lanes and results in covering the ground-truth trajectories closely, while also accounting for other possible driving maneuvers. In the first three scenes, ContextVAE accurately assigns samples to different lanes at the multi-lane intersections. In the fourth, it predicts the target vehicle's turning-left behavior along with a potential U-turn behavior. The last two cases exemplify non map-compliant behaviors, as the vehicle drives from and to the off-map region, respectively. While graph-based methods typically struggle with off-road cases, ContextVAE accurately predicts the vehicles' motion.

In Fig. 7, we highlight additional examples comparing the prediction results of ContextVAE when the map and its related attention mechanism are disabled from the observation encoder. Ignoring maps can lead to myopic as well as wrong predictions. In the first two cases, the model predicts that the target vehicle will keep driving straight and does not account for the multi-modal decisions at intersections. In the third scene, the model predicts multi-modal trajectories but are invalid as the vehicle can only take a right turn. In contrast, accounting for map attention leads to map-compliant paths and helps to predict potential turning behaviors at intersections.

To highlight how our unified observation encoding scheme (Section III-C) helps the model make accurate predictions, we consider three urban traffic examples. In Fig. 1, we visualize the corresponding activation maps of the map encoder obtained using the GradCAM technique [46] along with the social attention weights of the perceived neighbors. In the first example, the model focuses on the challenging intersection area around the

Prediction with S-ATTN + M-ATTN

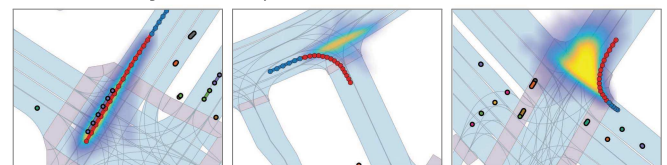Prediction w/o Map (S-ATTN only)

Fig. 7. Case studies with (top) and without (bottom) considering rasterized maps in the observation encoder. The trajectory distributions heatmaps are generated using 2,000 predictions with a Gaussian KDE method.

target vehicle and pays high attention to the leading car along its lane. In the second example, the most salient map area is the crosswalk (light red polygons) in front of the target vehicle, where a number of agents are waiting for the traffic lights to turn green. Hence, the vehicle's attention focuses on these neighbors while ignoring irrelevant agents such as the three stopped cars on the outer right lane. In the third example, the target vehicle is static and the model relies only on the observed neighbors' states to make predictions with little attention paid to the map. These examples demonstrate that our approach can flexibly utilize the environmental map information and the agents' social states to make predictions.

V. CONCLUSION

We introduce ContextVAE, a real-time for context-aware vehicle trajectory prediction. ContextVAE relies on a timewise VAE architecture and employs a map encoding module that performs observation encoding in a unified and socially-aware way. We show that our approach provides high-fidelity, map-compliant predictions on a variety of heterogeneous datasets, capturing the multi-modal nature of vehicle motions and their interactions with neighbor agents. We note that the model performance can be further improved by leveraging post-processing techniques like clustering approaches [11], [24], ensemble models [32], [40], and other recently proposed schemes [7], [43], at the cost of higher running times. Interesting avenues for future work are to account for dynamic environmental information like traffic lights in the observation encoding scheme by introducing local maps timewisely, and sampling trajectories from the *joint* distribution over a full set of agents through interactive trajectory prediction [12], [38]. It is also worth exploring how map representation influences context encoding and model performance, including trying alternative ways of map rasterization and using

vectorized road graphs. As it stands, our proposed dual-attention mechanism for RNN-based observation encoding enables high-fidelity predictions with low latency, adding value to existing approaches for map and agent state fusion. Combined with a timewise VAE architecture, the resulting ContextVAE approach can serve as a strong baseline for real-time vehicle trajectory prediction and inspire future work on VAE-based models.

REFERENCES

[1] H. Caesar et al., "nuScenes: A multimodal dataset for autonomous driving," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11621–11631.

[2] J. Houston et al., "One thousand and one hours: Self-driving motion prediction dataset," in *Proc. Conf. Robot. Learn.*, 2021, pp. 409–418.

[3] S. Ettinger et al., "Large scale interactive motion forecasting for autonomous driving: The Waymo open motion dataset," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021, pp. 9710–9719.

[4] S. Kolekar, S. Gite, B. Pradhan, and K. Kotecha, "Behavior prediction of traffic actors for intelligent vehicle using artificial intelligence techniques: A review," *IEEE Access*, vol. 9, pp. 135034–135058, 2021.

[5] S. Konev, K. Brodt, and A. Sanakoyeu, "Motion CNN: A strong baseline for motion prediction in autonomous driving," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2021.

[6] Y. Huang, J. Du, Z. Yang, Z. Zhou, L. Zhang, and H. Chen, "A survey on trajectory-prediction methods for autonomous driving," *IEEE Trans. Intell. Veh.*, vol. 7, no. 3, pp. 652–674, Sep. 2022.

[7] A. Kamenev et al., "PredictionNet: Real-time joint probabilistic traffic prediction for planning, control, and simulation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2022, pp. 8936–8942.

[8] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. S. Torr, and M. Chandraker, "Desire: Distant future prediction in dynamic scenes with interacting agents," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2165–2174.

[9] T. Salzmann, B. Ivanovic, P. Chakravarty, and M. Pavone, "Trajectron : Dynamically-feasible trajectory forecasting with heterogeneous data," in *Proc. 16th Eur. Conf. Comput. Vis.*, 2020, pp. 683–700.

[10] Y. Yao, E. Atkins, M. Johnson-Roberson, R. Vasudevan, and X. Du, "BiTraP: Bi-directional pedestrian trajectory prediction with multi-modal goal estimation," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 1463–1470, Apr. 2021.

[11] P. Xu, J.-B. Hayet, and I. Karamouzas, "SocialVAE: Human trajectory prediction using timewise latents," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 511–528.

[12] J. L. V. Espinoza, A. Liniger, W. Schwarting, D. Rus, and L. Van Gool, "Deep interactive motion prediction and planning: Playing games with motion prediction models," in *Proc. 4th Anuu. Learn. Dyn. Control Conf.*, 2022, pp. 1006–1019.

[13] 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 *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 1349–1358.

[14] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social LSTM: Human trajectory prediction in crowded spaces," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 961–971.

[15] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social GAN: Socially acceptable trajectories with generative adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2255–2264.

[16] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social GAN: Socially acceptable trajectories with generative adversarial networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2255–2264.

[17] J. Amirian, J.-B. Hayet, and J. Pettré, "Social ways: Learning multi-modal distributions of pedestrian trajectories with GANs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2019.

[18] N. Deo and M. M. Trivedi, "Convolutional social pooling for vehicle trajectory prediction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2018, pp. 1468–1476.

[19] N. Rhinehart, R. McAllister, K. Kitani, and S. Levine, "PRECOG: Prediction conditioned on goals in visual multi-agent settings," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2019, pp. 2821–2830.

[20] A. Rudenko, L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, and K. O. Arras, "Human motion trajectory prediction: A survey," *Int. J. Rob. Res.*, vol. 39, no. 8, pp. 895–935, 2020.

[21] T. Phan-Minh, E. C. Grigore, F. A. Boulton, O. Beijbom, and E. M. Wolff, "CoverNet: Multimodal behavior prediction using trajectory sets," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 14074–14083.

[22] Y. Chai, B. Sapp, M. Bansal, and D. Anguelov, "MultiPath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction," in *Proc. Conf. Robot Learn.*, 2019, pp. 86–99.

[23] H. Cui et al., "Multimodal trajectory predictions for autonomous driving using deep convolutional networks," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2019, pp. 2090–2096.

[24] N. Deo and M. M. Trivedi, "Trajectory forecasts in unknown environments conditioned on grid-based plans," 2020, *arXiv:2001.00735*.

[25] R. Girgis et al., "Latent variable sequential set transformers for joint multi-agent motion prediction," in *Proc. Int. Conf. Learn. Represents*, 2021.

[26] L. Zhang, P.-H. Su, J. Hoang, G. C. Haynes, and M. Marchetti-Bowick, "Map-adaptive goal-based trajectory prediction," in *Proc. Conf. Robot Learn.*, 2021, pp. 1371–1383.

[27] X. Huang et al., "DiversityGAN: Diversity-aware vehicle motion prediction via latent semantic sampling," *IEEE Robot. Automat. Lett.*, vol. 5, no. 4, pp. 5089–5096, Oct. 2020.

[28] C. Wang, Y. Wang, M. Xu, and D. J. Crandall, "Stepwise goal-driven networks for trajectory prediction," *IEEE Robot. Automat. Lett.*, vol. 7, no. 2, pp. 2716–2723, Apr. 2022.

[29] J. Gao et al., "VectorNet: Encoding HD maps and agent dynamics from vectorized representation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 11525–11533.

[30] M. Liang et al., "Learning lane graph representations for motion forecasting," in *Proc. Eur. Conf. Comput. Vis.*, 2020, pp. 541–556.

[31] S. Shi, L. Jiang, D. Dai, and B. Schiele, "Motion transformer with global intention localization and local movement refinement," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, pp. 6531–6543.

[32] T. Gilles, S. Sabatini, D. Tsishkou, B. Stanciulescu, and F. Moutarde, "GO-HOME: Graph-oriented heatmap output for future motion estimation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2022, pp. 9107–9114.

[33] Z. Zhou, J. Wang, Y.-H. Li, and Y.-K. Huang, "Query-centric trajectory prediction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 17863–17873.

[34] S. Khandelwal, W. Qi, J. Singh, A. Hartnett, and D. Ramanan, "What-if motion prediction for autonomous driving," 2020, *arXiv:2008.10587*.

[35] B. Kim et al., "LaPred: Lane-aware prediction of multi-modal future trajectories of dynamic agents," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 14636–14645.

[36] C. Luo, L. Sun, D. Dabiri, and A. Yuille, "Probabilistic multi-modal trajectory prediction with lane attention for autonomous vehicles," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 2370–2376.

[37] H. Zhao et al., "TNT: Target-driven trajectory prediction," in *Proc. Int. Conf. Robot Learn.*, 2021, pp. 895–904.

[38] Q. Sun, X. Huang, J. Gu, B. C. Williams, and H. Zhao, "M2I: From factored marginal trajectory prediction to interactive prediction," in *Proc. IEEE Conf. Comp. Vis. Pattern Recognit.*, 2022, pp. 6543–6552.

[39] N. Deo, E. Wolff, and O. Beijbom, "Multimodal trajectory prediction conditioned on lane-graph traversals," in *Proc. 5th Conf. Robot Learn.*, 2021, pp. 203–212.

[40] B. Varadarajan et al., "Multipath : Efficient information fusion and trajectory aggregation for behavior prediction," in *Proc. IEEE Int. Conf. Robot. Automat*, 2022, pp. 7814–7821.

[41] S. Casas, C. Gulino, S. Suo, and R. Urtasun, "The importance of prior knowledge in precise multimodal prediction," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 2295–2302.

[42] W. Zeng, M. Liang, R. Liao, and R. Urtasun, "LaneRCNN: Distributed representations for graph-centric motion forecasting," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 532–539.

[43] T. Gilles, S. Sabatini, D. Tsishkou, B. Stanciulescu, and F. Moutarde, "HOME: Heatmap output for future motion estimation," in *Proc. IEEE Int. Intell. Transp. Syst. Conf.*, 2021, pp. 500–507.

[44] J. Ngiam et al., "Scene transformer: A unified architecture for predicting future trajectories of multiple agents," in *Proc. Int. Conf. Learn. Representations*, 2021.

[45] M. Liu et al., "LAformer: Trajectory prediction for autonomous driving with lane-aware scene constraints," 2023, *arXiv:2302.13933*.

[46] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in *Proc. IEEE Int. Conf. Comput. Vis*, 2017, pp. 618–626.