Exploring Dynamic Context for Multi-path Trajectory Prediction

Hao Cheng[1], Wentong Liao[2], Xuejiao Tang[2], Michael Ying Yang[3], Monika Sester[1], and Bodo Rosenhahn[2]

***Abstract*— To accurately predict future positions of different agents in traffic scenarios is crucial for safely deploying intelligent autonomous systems in the real-world environment. However, it remains a challenge due to the behavior of a target agent being affected by other agents dynamically, and there being more than one socially possible paths the agent could take. In this paper, we propose a novel framework, named Dynamic Context Encoder Network (DCENet). In our framework, first, the spatial context between agents is explored by using self-attention architectures. Then, two LSTM encoders are trained to learn temporal context between steps by taking the observed trajectories and the extracted dynamic spatial context as input, respectively. The spatial-temporal context is encoded into a latent space using a Conditional Variational Auto-Encoder (CVAE) module. Finally, a set of future trajectories for each agent is predicted conditioned on the learned spatial-temporal context by sampling from the latent space, repeatedly. DCENet is evaluated on the largest and most challenging trajectory forecasting benchmark *Trajnet* and reports a new state-of-the-art performance. It also demonstrates superior performance evaluated on the benchmark *InD* for mixed traffic at intersections. A series of ablation studies are conducted to validate the effectiveness of each proposed module. Our code is available at `git@github.com:tanjatang/DCENet.git`.**

I. Introduction

Intelligent autonomous systems, such as robots and autonomous vehicles, have a high demand for the ability of accurately perceiving, understanding and predicting the future behavior of humans for effective and safe deployments in our real-world environment. For example, an autonomous agent will adjust its moving path according to the possible locations of other agents to prevent obstructions or collisions. However, it is challenging to predict the future location of an agent because it is not deterministic: (1) an agent may change its mind during the movement, (2) other agents' behaviors will affect its next step (*e.g.,* to avoid collisions), and (3) the influence from other agents is dynamic. Therefore, it is more beneficial to predict a set of most potential trajectories adaptive to the dynamic interactions between agents than to predict a deterministic one. In this work, we seek to explore the dynamic context between agents in traffic scenarios to predict a set of possible trajectories for each agent in the short future (12 steps) by observing their trajectories (8 steps), as showcased in Fig 1.

Specifically, the main contributions of this work are as follows: (1) It provides a novel end-to-end framework to predict

[1]Institute of Cartography and Geoinformatics, Leibniz University Hannover, Germany, {cheng, sester}@ikg.uni-hannover.de

[2]Institute of Information Processing, Leibniz University Hannover, Germany, {lastname}@tnt.uni-hannover.de

[3]Scene Understanding Group, University of Twente, The Netherlands, michael.yang@utwente.nl

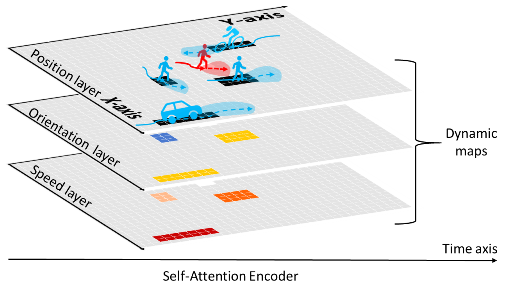


Fig. 1: Predicting multiple future trajectories (the most-likely one indicated by dash line over multiple ones indicated by blue shadow area) of a target agent (in red) conditioned on its observed movement (solid line) with the consideration of its interactions between neighboring agents (in blue) in *mixed traffic*. Interaction is learned through the dynamic maps with each layer dedicated to capturing position, orientation and speed information (indicated by color-coded rectangles) using the self-attention structure.

trajectories of homogeneous agents (pedestrians, bicycles, vehicles, *etc.*) rather than only for pedestrians that most of the prior works focus on [1]. (2) Self-attention modules are integrated into our framework to explore the dynamic context among agents. (3) A set of possible trajectories for each agent is predicted conditioned on its observed trajectory and the learned dynamic context using a CVAE [2] module. DCENet is evaluated on the Trajnet Challenge [3] which is one of the largest and most challenging benchmarks for trajectory forecasting. We further evaluate DCENet on the large-scale benchmark InD [4] to justify its efficacy and generalization ability. To judge the effectiveness of each proposed module, we conduct additional ablation studies. An overview of our framework is depicted in Fig. 2.

II. Related Work

Trajectory Prediction. Forecasting human trajectory has been researched for decades. In the early stages, many classic approaches are widely applied such as linear regression and Kalman filter [5], Gaussian processes [6, 7] and Markov decision processing [8, 9]. These traditional methods heavily rely on the quality of manually designed features, which cannot work reliably in a real-world environment and are poor at scaling up for dealing with the so-called Big Data. In recent years, many artificial intelligent (AI) technologies have been boosted by the thriving deep learning technologies [10], including human trajectory prediction [1, 11]–[16]. The deep learning models, especially the Long Short-Term

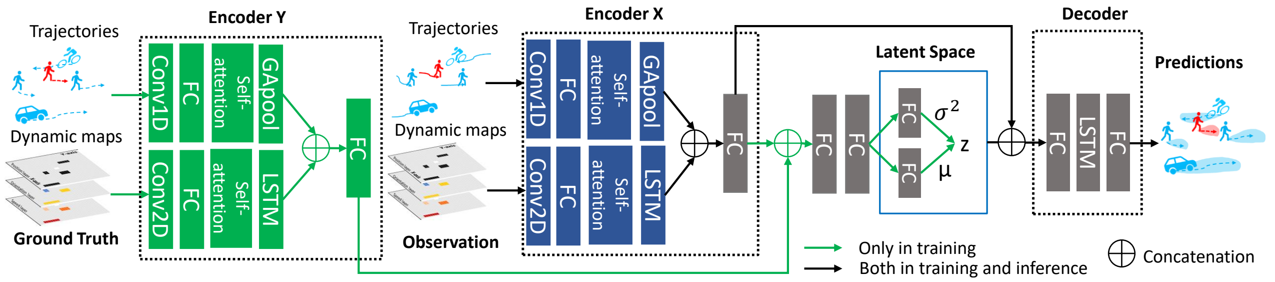


Fig. 2: The pipeline for the proposed method. The Encoder Y and Encoder X are identical in structure.

Memories (LSTMs) and Recurrent Neural Network (RNN), show great power in modeling complex social interactions between agents for collision avoidance and exploiting the time dependency for predicting futures [17]. The Social LSTM network [11] explores the interaction between pedestrians by connecting neighboring LSTMs in the social pooling layer and predicts trajectories for multiple pedestrians. Zhang *et al.* [1] proposes the States Refinement LSTM (SR-LSTM) model that aligns all the agents together and refines the state of each agent through a message-passing framework. The attention module [18] is incorporated in LSTMs to learn the spatial-temporal context of trajectories between pedestrians in [13, 19, 20]. However, many works have figured out the limited capability of LSTM in modeling human-human interaction [21, 22]. Recently, Transformer structure [23] has shown its power in context learning and sequential prediction [24]–[26]. In this paper, we will adopt the self-attention module to encode the dynamic interaction between traffic agents. Recent work of [27] seeks to utilize the Transformer structure to predict trajectory instead of LSTM. Our work is different from theirs essentially: (1) we use the generic self-attention module while they use the Deep Bidirectional Transformers (BERT) [25], which is a heavy stacked Transformer structure and is pre-trained on large-scale datasets, and (2) our framework is a generative model.

Multi-path Trajectory Prediction. Many approaches have been proposed to predict a socially compliant set of possible trajectories for an agent [12, 28]–[33]. Generative Adversarial Nets (GAN) [34] and CVAE [2, 35] are the most popular generative models used for this task. In [12], a trajectory sampler named "Social GAN" is proposed that considers the social effects of all agents. The generator is trained to predict a set of trajectories for each agent against a recurrent discriminator. Social and Physical attention mechanism are implemented in the GAN sampler so as to predict paths for each agent [13]. In [28], multiple plausible prediction samples are generated by a CVAE-based RNN encoder-decoder conditioned on observations. Katyal *et al.* [33] proposes to predict the intent of the target agent using a Bayesian approach as a condition of their CVAE-based LSTM encoder-decoder to help generate multiple paths. Meanwhile, they introduce an LSTM discriminator to train the framework in an adversarial way. In [36], scene context and the interactions between individual and group agents are accounted as a condition in their CVAE-based framework to sample multiple trajectories. Some other works treat the multi-path trajectory prediction problem as the estimation of a multimodal distribution. Cui *et al.* [32] proposes to model the multimodality of vehicle movement prediction with Deep Convolutional Networks. In [30], first, the multimodal distributions are predicted with an evolving strategy by combining the Winner-Takes-ALL loss [37]. Then, the samples from the first stage fit a distribution for trajectory prediction.

Our proposed framework is CVAE-based and integrates self-attention architectures to extract dynamic interactions among agents. We adopt the two-stream architecture [38] where one respective stream is dedicated for learning the spatial and temporal context explicitly, and they are later fused. In this way, exploiting the complex dynamic spatial-temporal context is decomposed into learning spatial context at a step among agents by using the self-attention modules and learning the temporal dependencies between steps by using a following LSTM encoder.

III. Method

A. Problem Formulation

Trajectory prediction is defined as to sequentially predict the future positions $\hat{\mathbf{Y}}_i = \{\hat{\mathbf{y}}_i^{T+1}, \cdots, \hat{\mathbf{y}}_i^{T'}\}$ of target agent i by observing its trajectory $\mathbf{X}_i = \{\mathbf{x}_i^1, \cdots, \mathbf{x}_i^T\}$, where $\mathbf{x}_i^t = (x_i^t, y_i^t)$ is the coordinates at the t-th step and so as $\hat{\mathbf{y}}_i^t$. T is the length of observed trajectory while T' is the total length of being observed and predicted trajectory. $\hat{\mathbf{Y}}_i$ should be as close to the corresponding ground truth $\mathbf{Y}_i$ as possible. The problem of multi-path trajectory prediction can be formulated as predicting a set of trajectories $\hat{\mathbf{Y}}_i = \{\hat{\mathbf{Y}}_{i,1}, \cdots, \hat{\mathbf{Y}}_{i,N}\}$ by observing $\mathbf{X}_i$ for agent i, where N is the total number of predicted trajectories.

B. Dynamic Maps

To model the interactions among agents, we first create dynamic maps for each agent that consists of the orientation, speed and position layers of its intermediate environment. Centralized on the target agent, a map is defined as a rectangular area of size $W \times H$ and divided into grid cells. First, referring to the target agent i, the neighboring agents $\mathrm{N}(i)$ are mapped into the closest grid cells$_{w \times h}^t$ according to their relative position. They are also mapped into the cells reached by their anticipated relative offset (speed) in the x

and y directions:

$$\begin{aligned}\text{cells}_w^t &= x_j^t - x_i^t + (\Delta x_j^t - \Delta x_i^t),\\ \text{cells}_h^t &= y_j^t - y_i^t + (\Delta y_j^t - \Delta y_i^t),\end{aligned} \tag{1}$$

where $w \leq W,\ h \leq H,\ j \in \mathsf{N}(i)$ and $j \neq i$. The *orientation layer* O stores the heading direction that is defined as the angle ϑ_j in the Euclidean plane and calculated in the given radians by $\vartheta_j = \arctan 2(\Delta y_j^t, \Delta x_j^t)$. $(\Delta y_j^t, \Delta x_j^t)$ is the offset of the position from t-th step to the next one for neighboring agent j. The angle is shifted into degree $[0, 360)$. Similarly, the *speed layer* S stores the travel speed and the *position* layer P stores the positions using a binary flag in the cells mapped above. Last, layer-wise, a Min-Max normalization scheme is applied for normalization, see Fig. 1. The map should cover a large vicinity area. Empirically we found $32 \times 32 m^2$ a proper setting considering both the coverage and the computational cost. The cell size is set to $1 \times 1 m^2$ as a balance to avoid the overlap of multiple agents in one cell based on the distribution of the experimental data, which is also supported by the preservation of personal space [39].

C. Encoder Network

Our encoder has two branches and each one mainly consists of stacked self-attention layers followed by an LSTM module, as illustrated in Fig. 2. One branch is trained to learn motion information from the observed trajectories, and the other one is trained to explore dynamic interactions among agents from the dynamic maps as discussed in Sec. III-B. The input of the former is the locations vector of the observed trajectory of the target agent $\mathbf{X}_i = \{\mathbf{x}_i^1, \cdots, \mathbf{x}_i^T\} \in \mathbb{R}^{T \times 2}$, while the latter one is the dynamic maps of the target agent noted as $DM = \{O, S, P\} \in \mathbb{R}^{T \times H \times W \times 3}$. For simplicity, we take the former one as an example. To get a sparse high-dimension representation, $\mathbf{X}_i$ is first passed to a convolution layer (Conv) and a fully connected (FC) layer. Each of them is followed by a ReLU non-linear activation. We denote this operation as $\pi(\mathbf{X}_i)$. A self-attention layer takes as input the Query (Q), Key (K) and Value (V) and outputs a weighted sum of the value vectors, as depicted in Fig. 3. The weight assigned to each value is calculated as the dot-product of the query with the corresponding key:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^{\mathbf{T}}}{\sqrt{d_k}})V, \tag{2}$$

where $\sqrt{d_k}$ is the scaling factor, d_k is the dimension of the vector K and $\mathbf{T}$ is the transpose operation. This operation is also called *scaled dot-product attention* [23]. The Q, K and V are obtained by three linear transformations with the same input separately:

$$Q = \pi(\mathbf{X})W_Q, \quad K = \pi(\mathbf{X})W_K, \quad V = \pi(\mathbf{X})W_V, \tag{3}$$

where $W_Q, W_K, W_V \in \mathbb{R}^{d_\pi \times d_k}$ are the trainable parameters and d_π is the dimension of $\pi(\mathbf{X})$.

Unlike LSTM whose inputs are in the order of sequence and the temporal information is retained explicitly, the self-attention module takes all inputs at the same time. To make use of the order of the sequence, position encodings are added to the Q, K and V at the bottom of each self-attention layer. The sine and cosine functions of different frequencies (varying in time here) are the most widely used:

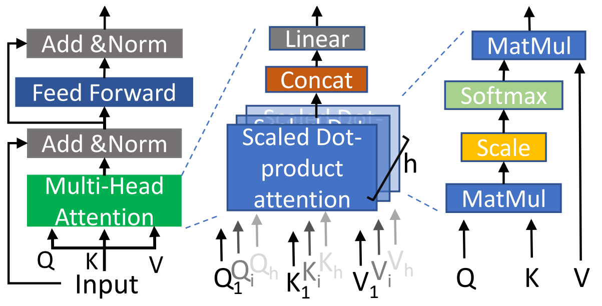


Fig. 3: Self-attention encoder network.

$$\mathbf{p}^t = \{p_{t,d}\}_{d=1}^{D}, \ p_{t,d} = \begin{cases} sin(\frac{t}{10000^{d/D}}), & \text{for } d \text{ even;} \\ cos(\frac{t}{10000^{d/D}}), & \text{for } d \text{ odd,} \end{cases} \tag{4}$$

where $D = d_k$ ensures position encodings to have the same dimension as the vectors of Q, K and V.

To attend to different information from different representation subspaces jointly, the *multi-head attention* [23] strategy is applied as a conventional operation, where a head is an independent scaled dot-product attention module:

$$\begin{aligned}\text{MultiHead}(Q, K, V) &= \text{ConCat}(\text{head}_1, ..., \text{head}_h)W_O,\\ \text{head}_i &= \text{Attention}(QW_{Qi}, KW_{Ki}, VW_{Vi}),\end{aligned} \tag{5}$$

where W_{Qi}, W_{Ki}, $W_{Vi} \in \mathbb{R}^{D \times d_{ki}}$ are the linear transformation parameters same as in Eq. (3) and W_O are the linear transformation parameters for aggregating the extracted information from different heads. Note that, $d_{ki} = \frac{d_k}{h}$ and d_{ki} must be an aliquot part of d_k. h is the total number of the attention heads and we use 2 heads in the implementation.

Then an LSTM is trained to exploit the temporal dependencies between steps by taking as input the output of the self-attention module and output an encoded representation. The other branch that exploits the dynamic interactions among agents works in the same way. Finally, the output of these two branches is connected and passed to a FC layer for fusion as the encoded information that includes dynamic spatial-temporal context.

D. Multiple Trajectories Prediction

Our method is CVAE-based and predicts multiple trajectories by repeatedly sampling from a learned latent space conditioned on the encoded information. The CVAE is an extension of the VAE [40] by introducing a condition to control the output [2]. Given a set of samples $(\mathbf{X}, \mathbf{Y}) = ((\mathbf{X}_1, \mathbf{Y}_1), \cdots, (\mathbf{X}_N, \mathbf{Y}_N))$, it jointly learns a recognition model $q_\phi(\mathbf{z}|\mathbf{Y}, \mathbf{X})$ of a variational approximation of the true posterior $p_\theta(\mathbf{z}|\mathbf{Y}, \mathbf{X})$ and a generation model $p_\theta(\mathbf{Y}|\mathbf{X}, \mathbf{z})$ for predicting the output $\mathbf{Y}$ conditioned on the input $\mathbf{X}$. $\mathbf{z}$ are the stochastic latent variables, ϕ and θ are the respective recognition and generative parameters. The goal is to maximize the *Conditional Log-Likelihood*: $\log p_\theta(\mathbf{Y}|\mathbf{X}) =$

$\log \sum_{\mathbf{z}} p_\theta(\mathbf{Y},\mathbf{z}|\mathbf{X}) = \log(\sum_{\mathbf{z}} q_\phi(\mathbf{z}|\mathbf{X},\mathbf{Y}) \frac{p_\theta(\mathbf{Y}|\mathbf{X},\mathbf{z}) p_\theta(\mathbf{z}|\mathbf{X})}{q_\phi(\mathbf{z}|\mathbf{X},\mathbf{Y})})$. According to Jensen's inequality [41], the evidence lower bound can be obtained:

$$\log p_\theta(\mathbf{Y}|\mathbf{X}) \geq -D_{KL}(q_\phi(\mathbf{z}|\mathbf{X},\mathbf{Y})||p_\theta(\mathbf{z})) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{X},\mathbf{Y})}[\log p_\theta(\mathbf{Y}|\mathbf{X},\mathbf{z})]. \tag{6}$$

Here both the approximated posterior $q_\phi(\mathbf{z}|\mathbf{X},\mathbf{Y})$ and the prior $p_\theta(\mathbf{z})$ are assumed to be Gaussian distribution for an analytical solution [40]. During training, the Kullback-Leibler divergence $D_{KL}(\cdot)$ pushes the approximated posterior to the prior distribution $p_\theta(\mathbf{z})$. The generation error $\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{X},\mathbf{Y})}(\cdot)$ measures the distance between the generated output and the ground truth. During inference, for a given observation $\mathbf{X}_i$, one latent variable $\mathbf{z}_i$ is drawn from the prior distribution $p_\theta(\mathbf{z})$, and one of the possible output $\hat{\mathbf{Y}}_i$ is generated from the distribution $p_\theta(\mathbf{Y}_i|\mathbf{X}_i,\mathbf{z}_i)$. The latent variables $\mathbf{z}$ allow for the one-to-many mapping from the condition to the output via multiple sampling. In this work, we model a conditional distribution $p_\theta(\mathbf{Y}_n|\mathbf{X})$, where $\mathbf{X}$ is the observed trajectory information and $\mathbf{Y}_n$ is one of its possible future trajectories.

Training: As shown in Fig. 2, during the training, both the observed trajectory $\mathbf{X}_i$ and its future trajectory $\mathbf{Y}_i$ are encoded by our encoder (see Sec. III-C), respectively. Then, their encodings are concatenated and passed through two FC layers (each is followed by a ReLU activation) for fusion. Then, two side-by-side FC layers are used to estimate the mean μ_{z_i} and the standard deviation σ_{z_i} of the latent variables $\mathbf{z}_i$. A trajectory $\hat{\mathbf{Y}}_i$ is reconstructed by an LSTM decoder step by step by taking $\mathbf{z}_i$ and the encodings of observation as input. Because the random sampling process of $\mathbf{z}_i$ can not be back propagated during training, the standard reparameterization trick [40] is adopted to make it differentiable. To minimize the error between the predicted trajectory $\hat{\mathbf{Y}}_i$ and the ground truth $\mathbf{Y}_i$, the reconstruction loss is defined as the $L2$ loss (Euclidean distance). Thus, the whole network is trained by minimizing the loss function using the stochastic gradient descent method.

$$L = ||\hat{\mathbf{Y}} - \mathbf{Y}||^2 + D_{KL}(q_\phi(\mathbf{z}|\mathbf{X},\mathbf{Y})||\mathcal{N}(0,I)). \tag{7}$$

Test: In the test phase, the ground truth of future trajectory is no more available and its pathway is removed (color coded in green in Fig. 2). A latent variable $\mathbf{z}$ is sampled from the prior distribution $\mathcal{N}(0,I)$ and concatenated with the observation encodings that serve as the condition for the following trained decoder, so that the decoder can predict a trajectory. To predict multiple trajectories, this process (sampling and decoding) is repeated multiple times.

E. Trajectory Ranking

We propose a ranking strategy to select the *most-likely* predicted trajectory out of the multiple predictions in order to adjust the Trajnet challenge setting. We apply bivariate Gaussian distribution to rank the predicted trajectories $(\hat{\mathbf{Y}}_{i,1}, \cdots, \hat{\mathbf{Y}}_{i,N})$ for each agent. At step t', all the predicted positions for agent i are stored in $|\hat{\mathsf{X}}_i, \hat{\mathsf{Y}}_i|^{t'}$. We follow [42] to fit the positions into the probability density function:

$$f(\hat{x}_i, \hat{y}_i)^{t'} = \frac{1}{2\pi \mu_{\hat{\mathsf{X}}_i} \mu_{\hat{\mathsf{Y}}_i} \sqrt{1-\rho^2}} \exp \frac{-Z}{2(1-\rho^2)},$$
$$Z = \frac{(\hat{x}_i - \mu_{\hat{\mathsf{X}}_i})^2}{{\sigma_{\hat{\mathsf{X}}_i}}^2} + \frac{(\hat{y}_i - \mu_{\hat{\mathsf{Y}}_i})^2}{{\sigma_{\hat{\mathsf{Y}}_i}}^2} - \frac{2\rho(\hat{x}_i - \mu_{\hat{\mathsf{X}}_i})(\hat{y}_i - \mu_{\hat{\mathsf{Y}}_i})}{\sigma_{\hat{\mathsf{X}}_i}\sigma_{\hat{\mathsf{Y}}_i}}. \tag{8}$$

where μ denotes the mean and σ the standard deviation, and ρ is the correlation between $\hat{\mathsf{X}}_i$ and $\hat{\mathsf{Y}}_i$. A predicted trajectory is scored as the sum of the relative likelihood of all its steps: $S(\hat{\mathbf{Y}}_{i,n}) = \sum_{t'=1}^{T'} f(\hat{x}_i, \hat{y}_i)^{t'}$. All predicted trajectories are ranked by this score and the one with the highest score stands out for the single-path prediction.

IV. Experiments

To evaluate the performance of our proposed method, we compare DCENet with the most influential and recent nine state-of-the-art models that on the Trajnet [3] challenge leader-board for fair comparison: (1) *Linear (off)*: a simple temporal linear regressor; (2) *Social Force* [43]: the very high impact rule-based model that implements social force to avoid collisions; (3) *S-LSTM* [11]: the highly cited LSTM-based model that introduces social pooling layer for modeling interactions; (4) *S-GAN* [12]: a GAN-based trajectory predictor; (5) *MX-LSTM* [44]: an LSTM trajectory predictor that utilizes the head direction of agent; (6) *SR-LSTM* [1]: an LSTM-based model that refines the hidden states by message passing; (7) *RED* [21]: an RNN encoder-decoder model that predicts trajectory directly only using observations; (8) *Ind-TF* [27]: a Transformer-based trajectory predictor; (9) *AMENet* [45]: the most recent state-of-the-art on the Trajnet leader-board. We further design a series of ablation studies to analyze the impact of each proposed module, *i.e.,* dynamic maps, transformer and LSTM encoder/decoder: (1) *Baseline:* an LSTM encoder-decoder only using the observed trajectory as input; (2) *DCENet w/o DMs:* the branch of encoding dynamic maps is removed from our final model; (3) *Trans. En&De:* the LSTM encoder-decoder is substituted by the Transformer encoder/decoder [23] in our framework.

A. Datasets

The Trajnet Challenge [3] is the largest multi-scenario forecasting benchmark. In the challenge, 8 consecutive ground-truth locations (3.2 seconds) of each trajectory are for observation and the following 12 steps (4.8 seconds) are required to forecast. Trajnet is a superset of diverse popular benchmark datasets: ETH [46], UCY [47], Stanford Drone Dataset [48], BIWI Hotel [46], and MOT PETS [49]. There is a total of 11448 trajectories from these four subsets covering 38 scenes for training. The test data is from the diverse partitions of them (besides MOT PETS) of the other 20 scenes without ground truth. The Trajnet Challenge provides a specific server for online evaluation. It is worth noting that many existing works are evaluated on a subset of Trajnet using their own train/test splits. In the contrast, a comparison

TABLE I: Results of different methods on the Trajnet Challenge [3]. Models are categorized into deterministic (determ.) and stochastic (stoch.) depending on if they incorporate a generative module.

Model	Category	Avg. [m]↓	FDE [m]↓	ADE [m]↓
S-LSTM [11]	determ.	1.3865	3.098	0.675
S-GAN [12]	stoch.	1.334	2.107	0.561
MX-LSTM [44]	determ.	0.8865	1.374	0.399
Linear (off)	stoch.	0.8185	1.266	0.371
Social Force [43]	determ.	0.8185	1.266	0.371
SR-LSTM [1]	determ.	0.8155	1.261	0.370
RED [21]	determ.	0.7800	1.201	0.359
Ind-TF [27]	determ.	0.7765	1.197	0.356
AMENet [45]	stoch.	0.7695	1.183	0.356
Baseline	stoch.	0.8045	1.239	0.370
DCENet w/o DMs	stoch.	0.7760	1.195	0.357
Trans. En&De	stoch.	0.7780	1.196	0.360
DCENet	stoch.	**0.7660**	**1.179**	**0.353**

of performance between different methods on the server is more fair and solid. For this sake, we only compare DCENet to the works which have shown their performance on the Trajnet Challenge leader-board.

InD was acquired by Bock *et al.* [4] using drones at four busy intersections in Germany in 2019. The traffic is dominated by vehicles and they interact with pedestrians heavily. The speed difference and confrontation makes the trajectory prediction challenging. The data was processed to obtain the same format as Trajnet: 8 steps for observation and the following 12 steps for prediction.

B. Evaluation Metrics

We adopt the most popular evaluation metrics: the mean average displacement error (ADE) and the final displacement error (FDE) to measure the trajectory prediction performance. ADE measures the aligned Euclidean distance from the prediction to its corresponding ground truth trajectory averaged overall steps. The mean value across all the trajectories is reported. FDE measures the Euclidean distance between the last position from the prediction to the corresponding ground truth position. In addition, the most-likely prediction is decided by the ranking method as described in Sec III-E. Compared with the ground truth (only if it is available), $@top10$ is the one out of ten predicted trajectories that has the smallest ADE and FDE.

The implementation details of training and testing our methods can be found in our code repository.

C. Results

The experimental results from different methods including our ablative models reported on the Trajnet leader-board are listed in Table I. We can see that DCENet reports new state-of-the-art performance and the ablative models also have comparable performances compared to the previous works.

First, by comparing to the Baseline, both DCENet w/o DMs and Ind-TF have much better results, and DCENet w/o DMs is slightly better (0.7760m vs. 0.7765m) in the average score and FDE but a little inferior in ADE than Ind-TF.

TABLE II: Quantitative results of our model and the comparative models on the InD benchmark measured by ADE/FDE.

Model	S-LSTM	S-GAN	AMENet	DCENet
InD		*@top 10*		
Intersection-(A)	2.04/4.61	2.84/4.91	0.95/1.94	**0.72/1.50**
Intersection-(B)	1.21/2.99	1.47/3.04	0.59/1.29	**0.50/1.07**
Intersection-(C)	1.66/3.89	2.05/4.04	0.74/1.64	**0.66/1.40**
Intersection-(D)	2.04/4.80	2.52/5.15	0.28/0.60	**0.20/0.45**
Avg.	1.74/4.07	2.22/4.29	0.64/1.37	**0.52/1.23**
InD		*Most-likely*		
Intersection-(A)	2.29/5.33	3.02/5.30	1.07/2.22	**0.96/2.12**
Intersection-(B)	1.28/3.19	1.55/3.23	0.65/1.46	**0.64/1.41**
Intersection-(C)	1.78/4.24	2.22/4.45	**0.83/1.87**	0.86/1.93
Intersection-(D)	2.17/5.11	2.71/5.64	0.37/0.80	**0.28/0.62**
Avg.	1.88/4.47	2.38/4.66	0.73/1.59	**0.69/1.52**

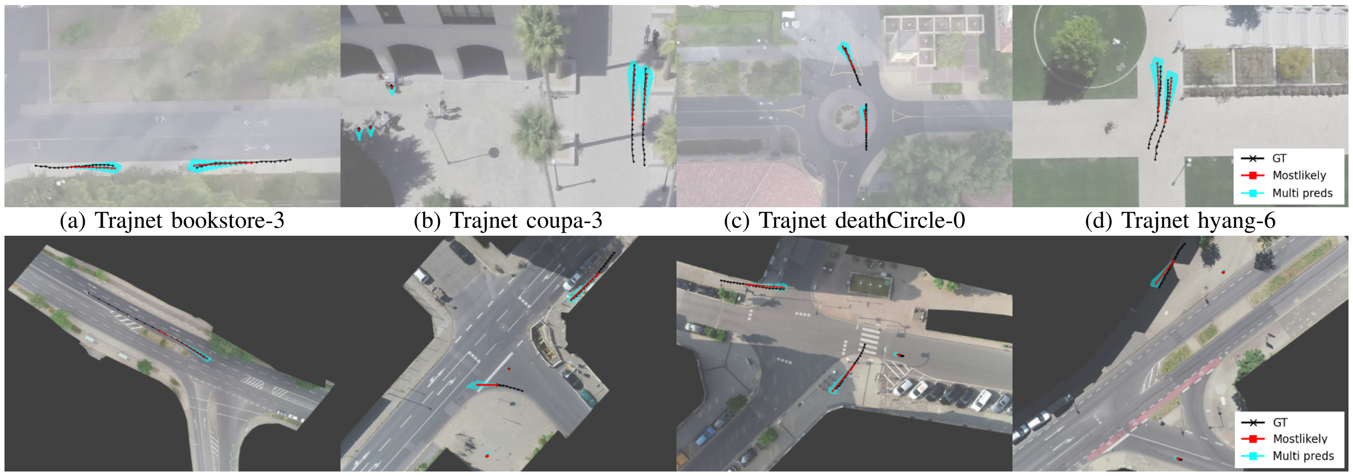
Considering both models only use observed trajectories as input, it indicates that our method (self-attention + LSTM encoder/decoder) explores a better spatial-temporal context than Transformer. Furthermore, Ind-TF utilizes BERT, a heavily stacked Transformer structure and must be pre-trained on an external large-scale dataset, while DCENet does not require it. The results of DCENet w/o DMs proves that its superior performance is not because we use more information (dynamic maps).

Second, by comparison between the Baseline and S-LSTM we can see that our Baseline model is significantly better. The only difference between them is that our Baseline is CVAE-based and generates multiple trajectories. It indicates that the future motion of humans is of high uncertainty, and predicting a set of possible trajectories is better than only predicting a single one. It also proves the effectiveness of the trajectory ranking methods (see Sec. III-E), which is used to select the most-likely trajectory from the multiple predictions. Our Baseline also outperforms S-GAN significantly, which is also a generative model for multiple trajectories prediction.

Third, interestingly, Trans. En&De that adopts the Transformer encoder and decoder in our framework did not achieve improved performance compared to DCENet. This phenomenon proves that our self-attention + LSTM encoder/decoder structure explores better dynamic context between agents than Transformer encoder/decoder in terms of trajectory prediction. The superior performance of Trans. w/o DMs against Ind-TF also confirms that.

Lastly, DCENet outperforms DCENet w/o DM. It indicates that the dynamic maps help model the interactions between agents and are useful for trajectory prediction.

Discussion According to the comparison above, the results indicate: (1) DCENet is effective for predicting accurate trajectories for homogeneous agents in various real-world traffic scenes, even without modeling interactions explicitly (the Baseline model). (2) The ranking method correctly estimates the multiple predictions and recommends a reliable candidate for the single-path trajectory prediction task. (3) Compared to the Baseline model, DCENet learns interaction via the dynamic maps with the self-attention structure effectively and achieved improved performance. (4) Both LSTM



(a) Trajnet bookstore-3 (b) Trajnet coupa-3 (c) Trajnet deathCircle-0 (d) Trajnet hyang-6

(e) InD Intersection-(A) (f) InD Intersection-(B) (g) InD Intersection-(C) (h) InD Intersection-(D)

Fig. 4: Multi-path trajectory predictions in shared spaces in Trajnet (1st row) and at different intersections in InD (2nd row).

and Transformer networks are capable of learning complex sequential patterns but their combination further enhances the performance in terms of trajectory prediction.

Furthermore, we tested DCENet on InD [4] to justify its performance and generalization ability. We compare our model with the three most relevant models: S-LSTM for comparing with its occupancy grid mapping for agent-to-agent interaction, S-GAN for its generative module, and AMENet for its CVAE module and LSTM sequential modeling. To guarantee a fair comparison, all the models were trained and tested using the same data. Table II lists the quantitative results measured by ADE/FDE. Our model achieved the best performance for the $@top10$ prediction across all the intersections and reduced the errors by a big margin. Our model also outperformed the other models for the most-likely prediction at three out of four intersections. It only slightly fell behind the AMENet model on the intersection-(C). We anticipate that the most-likely prediction fell behind the $@top10$ prediction. However, the ranking method is still effective in recommending a reliable candidate in comparison to the other models. The results indicate: (1) Our model is able to generalize on different datasets and maintain superior performance. (2) Predicting multiple paths is more beneficial than predicting a single one for an agent. On the one hand, multiple predictions increase the chances to narrow down the errors. On the other hand, a single prediction may lead to a wrong conclusion especially if the initial steps predicted are deviating from the ground truth and the errors will accumulate significantly with time. The multiple predictions form into an *area* indicating the potential intent of an agent and the area size reflects the uncertainty of an agent's intent.

The qualitative results are shown in Fig. 4. The first row showcases the scenarios in the Trajnet dataset. Note that the qualitative analysis on Trajnet was carried out on the validation set (an independent subset of the training set) for comparing with the ground truth. Our model accurately predicted two pedestrians walking towards each other at bookstore-3. The shadow areas indicate multiple possible trajectories. It also correctly predicted the static pedestrians in coupa-3, as well as the pedestrians walking in parallel. In deathCircle-0, our model predicted different possible turning angles for the cyclist in the roundabout. In hyang-6, two pedestrians walking closely to each other were predicted correctly. The second row showcases the scenarios in the InD dataset. Our model predicted a fast driving vehicle with a slightly different predicted speed at the Intersection-(A). It predicted that a left-turning vehicle may turn at the intersection-(B) with varying tuning angle and speed. The model also correctly predicted the interaction at the zebra crossing at the intersection-(C), where the vehicle stops to yield the way to the pedestrian. Similar predictions can be seen for the walking and static pedestrians, as well as the vehicle waiting at the entrance of the intersection-(D). Overall, we can also see that the recommended single path is very close to the corresponding ground truth for each agent.

V. Conclusion

In this paper, we proposed a novel framework DCENet for multi-path trajectory prediction for homogeneous agents in various real-world traffic scenarios. We decompose the learning of dynamic spatial-temporal context into exploiting the dynamic spatial context between agents using self-attention architectures and learning temporal context between steps with the following LSTM encoder. The spatial-temporal context is encoded into a latent space using a CVAE module. Finally, a set of future trajectories for each agent is predicted conditioned on the spatial-temporal context using the trained CVAE module. DCENet was evaluated on the Trajnet Challenge benchmark and achieved the new state-of-the-art on the leader-board. Its superior performance on the InD benchmark further validated its efficacy and generalization ability. The ablation studies justified the impact of each module in DCENet. In the future, we are interested in extending the method for learning the impact from environment/static context, *e.g.,* space layout and scene deployment, to further enhance the performance of trajectory prediction.

References

[1] P. Zhang, W. Ouyang, P. Zhang, J. Xue, and N. Zheng, "Sr-lstm: State refinement for lstm towards pedestrian trajectory prediction," in *CVPR*, 2019, pp. 12 085–12 094.

[2] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling, "Semi-supervised learning with deep generative models," in *NIPS*, 2014, pp. 3581–3589.

[3] A. Sadeghian, V. Kosaraju, A. Gupta, S. Savarese, and A. Alahi, "Trajnet: Towards a benchmark for human trajectory prediction," *arXiv preprint*, 2018.

[4] J. Bock, R. Krajewski, T. Moers, S. Runde, L. Vater, and L. Eckstein, "The ind dataset: A drone dataset of naturalistic road user trajectories at german intersections," *arXiv preprint arXiv:1911.07602*, 2019.

[5] A. C. Harvey, *Forecasting, structural time series models and the Kalman filter*. Cambridge university press, 1990.

[6] M. K. C. Tay and C. Laugier, "Modelling smooth paths using gaussian processes," in *Field and Service Robotics*, 2008, pp. 381–390.

[7] D. Ellis, E. Sommerlade, and I. Reid, "Modelling pedestrian trajectory patterns with gaussian processes," in *ICCV Workshops*, 2009, pp. 1229–1234.

[8] D. Makris and T. Ellis, "Spatial and probabilistic modelling of pedestrian behaviour." in *BMVC*, 2002, pp. 1–10.

[9] K. M. Kitani, B. D. Ziebart, J. A. Bagnell, and M. Hebert, "Activity forecasting," in *ECCV*, 2012, pp. 201–214.

[10] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, p. 436, 2015.

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

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

[13] A. Sadeghian, V. Kosaraju, A. Sadeghian, N. Hirose, and S. Savarese, "Sophie: An attentive gan for predicting paths compliant to social and physical constraints," in *CVPR*, 2019, pp. 1349–1358.

[14] N. Mohajerin and M. Rohani, "Multi-step prediction of occupancy grid maps with recurrent neural networks," in *CVPR*, 2019, pp. 10 600–10 608.

[15] R. Chandra, U. Bhattacharya, A. Bera, and D. Manocha, "Traphic: Trajectory prediction dense and heterogeneous traffic using weighted interactions," in *CVPR*, 2019, pp. 8483–8492.

[16] C. Tang and R. R. Salakhutdinov, "Multiple futures prediction," in *NIPS*, 2019, pp. 15 398–15 408.

[17] P. Kothari, S. Kreiss, and A. Alahi, "Human trajectory forecasting crowds: A deep learning perspective," *arXiv preprint arXiv:2007.03639*, 2020.

[18] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *ICLR*, 2015.

[19] A. Al-Molegi, M. Jabreel, and A. Martinez-Balleste, "Move, attend and predict: An attention-based neural model for people's movement prediction," *Pattern Recognition Letters*, vol. 112, pp. 34–40, 2018.

[20] A. Vemula, K. Muelling, and J. Oh, "Social attention: Modeling attention human crowds," in *ICRA*, 2018, pp. 1–7.

[21] S. Becker, R. Hug, W. Hübner, and M. Arens, "An evaluation of trajectory prediction approaches and notes on the trajnet benchmark," *arXiv preprint arXiv:1805.07663*, 2018.

[22] S. Becker, R. Hug, W. Hubner, and M. Arens, "Red: A simple but effective baseline predictor for the trajnet benchmark," in *ECCV*, 2018, pp. 138–153.

[23] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *NIPS*, 2017, pp. 5998–6008.

[24] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang, "Bottom-up and top-down attention for image captioning and visual question answering," in *CVPR*, 2018, pp. 6077–6086.

[25] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," in *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019, pp. 4171–4186.

[26] S. He, W. Liao, H. R. Tavakoli, M. Yang, B. Rosenhahn, and N. Pugeault, "Image captioning through image transformer," *arXiv preprint arXiv:2004.14231*, 2020.

[27] F. Giuliari, I. Hasan, M. Cristani, and F. Galasso, "Transformer networks for trajectory forecasting," in *ICPR*, 2020.

[28] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. Torr, and M. Chandraker, "Desire: Distant future prediction dynamic scenes with interacting agents," in *CVPR*, 2017, pp. 336–345.

[29] J. Amirian, J.-B. Hayet, and J. Pettré, "Social ways: Learning multimodal distributions of pedestrian trajectories with gans," in *CVPR Workshops*, 2019, pp. 0–0.

[30] O. Makansi, E. Ilg, O. Cicek, and T. Brox, "Overcoming limitations of mixture density networks: A sampling and fitting framework for multimodal future prediction," in *CVPR*, 2019, pp. 7144–7153.

[31] A. Poibrenski, M. Klusch, I. Vozniak, and C. Müller, "M2p3: multimodal multi-pedestrian path prediction by self-driving cars with egocentric vision," in *Annual ACM Symposium on Applied Computing*, 2020, pp. 190–197.

[32] 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*, 2019, pp. 2090–2096.

[33] K. D. Katyal, G. D. Hager, and C.-M. Huang, "Intent-aware pedestrian prediction for adaptive crowd navigation," in *ICRA*, 2020, pp. 3277–3283.

[34] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *NIPS*, 2014, pp. 2672–2680.

[35] K. Sohn, H. Lee, and X. Yan, "Learning structured output representation using deep conditional generative models," in *NIPS*, 2015, pp. 3483–3491.

[36] H. Cheng, W. Liao, M. Y. Yang, M. Sester, and B. Rosenhahn, "Mcenet: Multi-context encoder network for homogeneous agent trajectory prediction mixed traffic," in *ITSC*, 2020.

[37] A. Guzman-Rivera, D. Batra, and P. Kohli, "Multiple choice learning: Learning to produce multiple structured outputs," in *NIPS*, 2012, pp. 1799–1807.

[38] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Advances in neural information processing systems*, 2014, pp. 568–576.

[39] C. L. Gérin-Lajoie, Martand Richards and B. J. McFadyen, "The negotiation of stationary and moving obstructions during walking: anticipatory locomotor adaptations and preservation of personal space," *Motor control*, vol. 9, no. 3, pp. 242–269, 2005.

[40] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in *ICLR*, 2014.

[41] J. L. W. V. Jensen *et al.*, "Sur les fonctions convexes et les inégalités entre les valeurs moyennes," *Acta mathematica*, vol. 30, pp. 175–193, 1906.

[42] A. Graves, "Generating sequences with recurrent neural networks," *arXiv preprint arXiv:1308.0850*, 2013.

[43] D. Helbing and P. Molnar, "Social force model for pedestrian dynamics," *Physical review E*, vol. 51, no. 5, p. 4282, 1995.

[44] I. Hasan, F. Setti, T. Tsesmelis, A. Del Bue, F. Galasso, and M. Cristani, "Mx-lstm: mixing tracklets and vislets to jointly forecast trajectories and head poses," in *CVPR*, 2018, pp. 6067–6076.

[45] H. Cheng, W. Liao, M. Y. Yang, B. Rosenhahn, and M. Sester, "Amenet: Attentive maps encoder network for trajectory prediction," *arXiv preprint arXiv:2006.08264*, 2020.

[46] S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool, "You'll never walk alone: Modeling social behavior for multi-target tracking," in *ICCV*, 2009, pp. 261–268.

[47] A. Lerner, Y. Chrysanthou, and D. Lischinski, "Crowds by example," in *Computer Graphics Forum*, vol. 26, no. 3. Wiley Online Library, 2007, pp. 655–664.

[48] A. Robicquet, A. Sadeghian, A. Alahi, and S. Savarese, "Learning social etiquette: Human trajectory understanding crowded scenes," in *ECCV*, 2016, pp. 549–565.

[49] J. Ferryman and A. Shahrokni, "Pets2009: Dataset and challenge," in *International workshop on performance evaluation of tracking and surveillance*, 2009, pp. 1–6.