AMENet: Attentive Maps Encoder Network for Trajectory Prediction

Hao Cheng\textsuperscript{a}, Wentong Liao\textsuperscript{b}, Michael Ying Yang\textsuperscript{c}, Bodo Rosenhahn\textsuperscript{b}, Monika Sester\textsuperscript{a}

\textsuperscript{a}Institute of Cartography and Geoinformatics, Leibniz University Hannover, Germany
\textsuperscript{b}Institute of Information Processing, Leibniz University Hannover, Germany
\textsuperscript{c}Scene Understanding Group, ITC Faculty, University of Twente, The Netherlands

Abstract

Trajectory prediction is a crucial task in different communities, such as intelligent transportation systems, photogrammetry, computer vision, and mobile robot applications. However, there are many challenges to predict the trajectories of heterogeneous road agents (e.g., pedestrians, cyclists and vehicles) at a microscopical level. For example, an agent might be able to choose multiple plausible paths in complex interactions with other agents in varying environments, and the behavior of each agent is affected by the various behaviors of its neighboring agents. To this end, we propose an end-to-end generative model named \textit{Attentive Maps Encoder Network (AMENet)} for accurate and realistic multi-path trajectory prediction. Our method leverages the target road user’s motion information (\textit{i.e.}, movement in \textit{xy}-axis in a Cartesian space) and the interaction information with the neighboring road users at each time step, which is encoded as dynamic maps that are centralized on the target road user. A conditional variational auto-encoder module is trained to learn the latent space of possible future paths based on the dynamic maps and then used to predict multiple plausible future trajectories conditioned on the observed past trajectories. Our method reports the new state-of-the-art performance (final/mean average displacement (FDE/MDE) errors 1.183/0.356 meters) on benchmark datasets and wins the first place in the open challenge of Trajnet.

\textit{Keywords:} Trajectory prediction, Generative model, Encoder

Preprint submitted to ISPRS Journal of Photogrammetry and Remote Sensing June 16, 2020
1. Introduction

Accurate trajectory prediction of road users is a crucial task in different communities, such as intelligent transportation systems (ITS) [1, 2, 3], photogrammetry [4, 5, 6], computer vision [7, 8], mobile robot applications [9]. This task enables an intelligent system to foresee the behaviors of road users and make a reasonable and safe decision for its next operation, especially in urban mixed-traffic zones (a.k.a. shared spaces [10]). Trajectory prediction is generally defined as to predict the plausible (e.g., collision free and energy efficient) and socially-acceptable (e.g., considering social relations, social rules and norms between agents) positions in 2D or 3D of non-erratic target agents at each time step within a predefined future time interval relying on observed partial trajectories over a certain period of time [11, 7]. The target agent is defined as the dynamic object for which the actual prediction is made, mainly pedestrian, cyclist, vehicle and other road users [12]. A typical prediction process of mixed traffic is exemplified in Fig. 1.

How to effectively and accurately predict trajectories of mixed agents is still an unsolved problem. The challenges are mainly from three aspects: 1) the complex behavior and uncertain moving intention of each agent, 2) the presence and interactions between the target agent and its neighboring agents and 3) the multi-modality of paths: there are usually more than one socially-acceptable paths that an agent could use in the future.

There exists a large body of literature that focuses on addressing parts or all of the aforementioned challenges in order to make accurate trajectory prediction. The traditional methods model the interactions based on hand-crafted
features, such as force-based rules [11], Game Theory [13], or a constant velocity assumption [14]. Their performance is crucially affected by the quality of manually designed features and they lack generalizability [15]. Recently, boosted by the development of deep learning technologies [16], data-driven methods keep reporting new state-of-the-art performance on benchmarks [7, 17, 18, 19, 3]. For instance, Recurrent Neural Networks (RNNs) based models are used to model the interactions between agents and predict the future positions in sequence [7, 20]. However, these works design a discriminative model and produce a deterministic outcome for each agent. The models tend to predict the “average” trajectories because the commonly used objective function minimizes the Euclidean distance between the ground truth and the predicted outputs.

To predict multiple socially-acceptable trajectories for each target agent, different generative models are proposed, such as Generative Adversarial Networks (GANs) [21] based framework Social GAN [19] and Conditional Variational Auto-Encoder (CVAE) [22, 23, 24] based framework DESIRE [17].

In spite of the great success in this domain, most of these methods are designed for a specific agent type: pedestrians. In reality, pedestrians, cyclists and vehicles are the three major types of agents and their behaviors affect each other. To make precise trajectory prediction, their interactions should be considered jointly. Besides, the interactions between the target agent and the others are equally treated. But different agents may not equally affect the target agent on how to move in the near future. For instance, the closer agents should affect the target agent stronger than the more distant ones, and the target vehicle is affected more by the pedestrians who tend to cross the road than the vehicles which are behind it. Last but not least, the robustness of the models are not fully tested in real world outdoor mixed traffic environments (e.g., roundabouts, intersections) with various unseen traffic situations. So an important research question is: Can a model trained in some spaces to predict accurate trajectories in other unseen spaces?

To address the aforementioned limitations, we propose this work named Attentive Maps Encoder Network (AMENet) leveraging the ability of generative
models that generate diverse patterns of future trajectories and modeling the interactions between target agents with the others as attentive dynamic maps. The dynamic map manipulates the information extracted from the neighboring agents' orientation, speed and position in relation to the target agent at each time step for interaction modeling and the attention mechanism enables the model to automatically focus on the salient features extracted over different time steps. An overview of our proposed framework is depicted in Fig. 2. It has an encoding-decoding structure.

**Encoding.** Two encoders are designed for learning representations of the observed trajectories (X-Encoder) and the future trajectories (Y-Encoder) respectively and they have a similar structure. Taking the X-Encoder as an example (see Fig. 3), the encoder first extracts the motion information from the target agent (coordinate offset in sequential time steps) and the interaction information with the other agents respectively. Particularly, to explore the dynamic interactions, the motion information of each agent is characterised by its orientation, speed and position at each time step. Then a self-attention mechanism is utilized over all agents to extract the dynamic interaction maps. This is where the name **Attentive Maps Encoder** comes from. The motion and interaction information along the observed time interval are collected by two independent Long Short-Term Memories (LSTMs) and then fused together. The output of the Y-Encoder is supplement to a variational auto-encoder to learn the latent space of future trajectories distribution, which is assumed as a Gaussian distribution.

**Decoding.** The output of the variational auto-encoder module (it is achieved by re-parameterization of encoded features during training phase and resampling from the learned latent space during the inference phase) is fed forward to the following decoder associated with the output of the X-Encoder as condition to forecast the future trajectory, which works in the same way as a conditional variational auto-encoder (CVAE) [22, 23, 24].

The main contributions are summarized as follows:

1. We propose a generative framework Attentive Maps Encoder Network
(AMENet) for multi-path trajectory prediction. AMENet inserts a generative module that is trained to learn a latent space for encoding the motion and interaction information in both observation and future, and predicts multiple feasible future trajectories conditioned on the observed information.

2 We design a novel module, attentive maps encoder that learns spatio-temporal interconnections among agents based on dynamic maps using a self-attention mechanism.

3 Our model is able to predict heterogeneous road users, i.e., pedestrians, cyclists and vehicles rather than only focusing on pedestrians, in various unseen real-world environments, which makes our work different from most of the previous ones that only predict pedestrian trajectories.

The efficacy of the proposed method has been validated on the recent benchmark Trajnet [25] that contains various datasets in various environments for trajectory prediction. Our method reports the new state-of-the-art performance and wins the first place on the leader board. Each component of the proposed model is validated via a series of ablative studies.

2. Related Work

Our work focuses on predicting trajectories of heterogeneous road agents. In this section we discuss the recent related works mainly in the following aspects: modeling this task as a sequence prediction, modeling the interactions between agents for precise path prediction, modeling with attention mechanisms, and utilizing generative models to predict multiple plausible trajectories. Our work focuses on modeling the dynamic interactions between agents and training a generative model to predict multiple plausible trajectories for each target agent.

2.1. Sequence Modeling

Modeling the trajectory prediction as a sequence prediction task is the most popular approach. The 2D/3D position of a target agent is predicted step by
step along the time axis. The widely applied models include linear regression and Kalman filter [26], Gaussian processes [27] and Markov decision processing [28]. However, these traditional methods largely rely on the quality of manually designed features and are unable to tackle large-scale data. Recently, data-driven deep learning technologies, especially RNN-based models and the variants, e.g., Long Short-Term Memories (LSTMs) [29], have demonstrated the powerful ability in extracting representations from massive data automatically and are used to learn the complex patterns of trajectories. In recent years, RNN-based models keep pushing the edge of accuracy of predicting pedestrian trajectory [7, 19, 30, 31], as well as other types of road users [8, 32, 33]. In this work, we also utilize LSTMs to encode the temporal sequential information and decode the learned features to predict trajectories in sequence.

2.2. Interaction Modeling

The behavior of an agent is not only decided by its own will but also crucially affected by the interactions between it and the other agents. Therefore, effectively modeling the social interactions among agents is important for accurate trajectory prediction. One of the most influential approaches for modeling interactions is the Social Force Model [11], which uses the repulsive force for collision avoidance and the attractive force for social connections. Game Theory is utilized to simulate the negotiation between different road users [13]. Such rule-based interaction modelings have been incorporated into deep learning models. Social LSTM proposes an occupancy grid to locate the positions of close neighboring agents and uses a social pooling layer to encode the interaction information for trajectory prediction [7]. Many works design their specific “occupancy” grid for interaction modeling [17, 20, 34, 2, 6, 13]. Cheng et al. [3] consider the interaction between individual agent and group agents with social connections and report better performance. Meanwhile, different pooling mechanisms are proposed for interaction modeling. For example, Social GAN [19] embeds relative positions between the target and all the other agents with each agent’s motion hidden state and uses an element-wise pooling to extract the in-
teraction between all the pairs of agents, not only the close neighboring agents; Similarly, all the agents are considered in SR-LSTM [31]. It proposes a states refinement module for aligning all the agents together and adaptively refines the state of each agent through a message passing framework. The motion gate and agent-wise attention are used to select the most important information from neighboring agents. Most of the aforementioned models extract interaction information based on the relative position of the neighboring agents in relation to the target agent. The dynamics of interactions are not fully captured both in spatial and temporal domains.

2.3. Modeling with Attention

Recently, different attention mechanisms [35, 36, 37] are incorporated in neural networks for learning complex spatio-temporal interconnections. Particularly, their effectiveness have been proven in learning powerful representations from sequence information in, e.g., neural machine translation [35, 37] and image caption generation [36, 38, 39]. Some of the recent state-of-the-art methods also have adapted attention mechanisms for sequence modeling and interaction modeling to predict trajectories. For example, a soft attention mechanism [36] is incorporated in LSTMs to learn the spatio-temporal patterns from the position coordinates [40]. Similarly, SoPhie [30] applies two separate soft attention modules, one called physical attention for learning the salient features between agent and scene and the other called social attention for modeling agent to agent interactions. In the MAP model [41], an attentive network is implemented to learn the relationships between the location and time information. The most recent work Ind-TF [42] replaces RNN with Transformer [40] for modeling trajectory sequences. In this work, we model the dynamic interactions among all road users by utilizing the self-attention mechanism [40] along the time axis. The self-attention mechanism is defined as mapping a query and a set of key-value pairs to an output. First, the similarity between the query and each key is computed to obtain a weight. The weights associated with all the keys are then normalized via, e.g., a softmax function and are applied to weigh the correspond-
ing values for obtaining the final attention. Unlikely RNN based structures that propagate the information along the symbol positions of the input and output sequence, which leads to increasing difficulties for information propagation in long sequences, the self-attention mechanism relates different positions of a single sequence in order to compute a representation of the entire sequence [40]. The dependency between the input and output is not restricted to their distance of positions.

2.4. Generative Models

Up to date, VAE [22] and GAN [21] and their variants (e.g., Conditional VAE [23, 24]) are the most popular generative models in the era of deep learning. They are both able to generate diverse outputs by sampling from noise. The essential difference is that, GAN trains a generator to generate a sample from noise and a discriminator to decide whether the generated sample is real enough. The generator and discriminator enhance mutually during training. In contrast, VAE is trained by maximizing the lower bound of training data likelihood for learning a latent space that approximates the distribution of the training data. Generative models have shown promising performance in different tasks, e.g., super resolution, image to image translation, image generation, as well as trajectory prediction [17, 19, 3]. Predicting one single trajectory may not sufficient due to the uncertainties of road users’ behavior. Gupta et al. [19] train a generator to generate future trajectories from noise and a discriminator to judge whether the generated ones are fake or not. The performance of the two modules are enhanced mutually and the generator is able to generate trajectories that are as precise as the real ones. Similarly, Amirian et al. [43] propose a GAN-based model for generating multiple plausible trajectories for pedestrians. Lee et al. [17] propose a CVAE model to predict multiple plausible trajectories. Cheng et al. [3] propose a CVAE like model named MCENet to predict multiple plausible trajectories conditioned on the scene context and previous information of trajectories. In this work, we incorporate a CVAE module to learn a latent space of possible future paths for predicting multiple plausi-
Figure 2: An overview of the proposed framework. It consists of four modules: the X-Encoder and Y-Encoder are used for encoding the observed and the future trajectories, respectively. They have a similar structure. The Sample Generator produces diverse samples of future generations. The Decoder module is used to decode the features from the produced samples in the last step and predicts the future trajectory sequentially. The specific structure of the X-Encoder/Y-Encoder is given by Fig. 3.

Our work essentially distinguishes from the above generative models in the following two points: (1) We insert not only ground truth trajectory, but also the dynamic maps associated with the ground truth trajectory into the CVAE module during training, which is different from the conventional CVAE that follows a consistent input and output structure (e.g., the input and output are both trajectories in the same structure [17].) (2) Our method does not explore information from images, i.e., visual information is not used and future trajectories are predicted only based on the map data (i.e., position coordinate). The visual information, such as vegetation, curbside and buildings, are very different from one space to another. Our model is trained on some available spaces but is validated on other unseen spaces. The over-trained visual features, on the other hand, may jeopardise the model’s robustness and lead to a bad performance in an unseen space of totally different environment [3].

3. Methodology

In this section, we introduce the proposed model AMENet in details (Fig. 2) in the following structure: a brief review on CVAE (Sec. 3.1), Problem Definition
3.1. Diverse Sample Generation with CVAE

In tasks like trajectory prediction, we are interested in modeling a conditional distribution \( P(Y_n | X) \), where \( X \) is the previous trajectory information and \( Y_n \) is one of the possible future trajectories. In order to realize this goal that generates controllable diverse samples of future trajectories based on past trajectories, a deep generative model, a conditional variational auto-encoder (CVAE), is adopted inside our framework. CVAE is an extension of the generative model VAE [22] by introducing a condition to control the output [23]. Concretely, it is able to learn the stochastic latent variable \( z_i \) that characterizes the distribution \( P(Y_i | X_i) \) of \( Y_i \) conditioned on the input \( X_i \), where \( i \) is the index of sample. The objective function of training CVAE is formally defined as:

\[
\log P(Y_i | X_i) \geq -D_{KL}(Q(z_i | Y_i, X_i) || P(z_i)) + \mathbb{E}_{Q(z_i | Y_i, X_i)} \left[ \log P(Y_i | z_i, X_i) \right],
\]

where \( Y_i \) and \( X_i \) stand for the future and past trajectories in our task, respectively. \( z_i \) is the learned latent variable of \( Y_i \). The objective is to maximize the conditional probability \( \log P(Y_i | X_i) \), which is equivalent to minimize \( \ell(\hat{Y}_i, Y_i) \) and minimize the Kullback-Leibler divergence \( D_{KL}(\cdot) \) in parallel. In order to enable the back propagation for stochastic gradient descent in \( \mathbb{E}_{Q(z_i | Y_i, X_i)} \left[ \log P(Y_i | z_i, X_i) \right] \), a reparameterization trick [44] is applied: \( z_i = \mu_i + \sigma_i \circ \epsilon_i \). Here, \( z_i \) is assumed to have a Gaussian distribution \( z_i \sim Q(z_i | Y_i, X_i) = \mathcal{N}(\mu_i, \sigma_i) \). \( \epsilon_i \) is sampled from noise that follows a normal Gaussian distribution, and the mean \( \mu_i \) and the standard deviation \( \sigma_i \) over \( z_i \) are produced by two side-by-side fc layers, respectively (as shown in Fig. 2). In this way, the derivation problem of the sampling process \( Q(z_i | Y_i, X_i) \) is turned into deriving the sample results \( z_i \) w.r.t. \( \mu_i \) and \( \sigma_i \). Then, the back propagation for stochastic gradient descent can be utilized to optimize the networks, which produce \( \mu_i \) and \( \sigma_i \).
3.2. Problem Definition

The multi-path trajectory prediction problem is defined as follows: agent \( i \), receives as input its observed trajectories \( X_i = \{X^1_i, \ldots, X^T_i\} \) and predict its \( n \)-th plausible future trajectory \( \hat{Y}_{i,n} = \{\hat{Y}^1_{i,n}, \ldots, \hat{Y}^T_{i,n}\} \). \( T \) and \( T' \) denote the sequence length of the past and being predicted trajectory, respectively.

The trajectory position of \( i \) at time step \( t \) is characterized by the coordinate as \( X^t_i = (x^t_i, y^t_i) \) (3D coordinates are also possible, but in this work only 2D coordinates are considered) and so as \( \hat{Y}^t_{i,n} = (\hat{x}^t_{i,n}, \hat{y}^t_{i,n}) \). The objective is to predict its multiple plausible future trajectories \( \hat{Y}_i = \hat{Y}_{i,1}, \ldots, \hat{Y}_{i,N} \) that are as accurate as possible to the ground truth \( Y_i \). This task is formally defined as \( \hat{Y}_{i,n} = f(X_i, \text{Map}_i), \ n \in N \). Here, the total number of predicted trajectories is denoted as \( N \) and Map\(_i\) denotes the dynamic maps centralized on the target agent for mapping the interactions with its neighboring agents over the time steps. More details of the dynamic maps will be given in Sec. 3.4.

3.3. Motion Input

The motion information for each agent is captured by the position coordinates at each time step. Specifically, we use the offset \( (\Delta x^t_i, \Delta y^t_i) \) of the trajectory positions between two consecutive time steps as the motion information instead of the coordinates in a Cartesian space, which has been widely applied in this domain [19, 45, 31, 3]. Compared to coordinates, the offset is independent from the given space and less sensitive in the regard of overfitting a model to a particular space or travel directions. The offset can be interpreted as speed over time steps that are defined with a constant duration. As long as the original position is known, the absolute coordinates at each position can be calculated by cumulatively summing the sequence offsets. As the augmentation technique we randomly rotate the trajectories to prevent the system from only learning certain directions. In order to maintain the relative positions and angles between agents, the trajectories of all the agents coexisting in a given period are rotated by the same angle.
Figure 3: Structure of the X-Encoder. The encoder has two branches: the upper one is used to extract motion information of target agents (i.e., movement in $x$- and $y$-axis in a Cartesian space), and the lower one is used to learn the interaction information among the neighboring road users from dynamic maps over time. Each dynamic map consists of three layers that represents orientation, travel speed and relative position, which are centralized on the target road user respectively. The motion information and the interaction information are encoded by their own LSTM sequentially. The last outputs of the two LSTMs are concatenated and forwarded to a $fc$ layer to get the final output of the X-Encoder. The Y-Encoder has the same structure as the X-Encoder but it is used for extracting features from the future trajectories and only used in the training phase.

3.4. Dynamic Maps

Different from the recent works of parsing the interactions between the target and neighboring agents using an occupancy grid [7, 17, 20, 34, 2, 6, 13], we propose a novel and straightforward method—attentive dynamic maps—to learn interaction information among agents. As demonstrated in Fig. 3, a dynamic map at a given time step consists of three layers that interpret the information of orientation, speed and position, respectively, which is derived from the trajectories of the involved agents. Each layer is centralized on the target agent’s position and divided into uniform grid cells. The layers are divided into grids because: (1) comparing with representing information in pixel level, is more computationally effective in grid level; (2) the size and moving
speed of an agent is not fixed and it occupies a local region of pixels in arbitrary
form, the spatio-temporal information of each pixel is different from each other
even though they belong to the same agent. Therefore, we represent the spatio-
temporal information as an average value within a grid. We calculate the value
of each grid in different layers as follows: the neighboring agents are located into
the corresponding grids by their relative position to the target agent and they
are also located into the grids by the anticipated relative offset (speed) to the
target agent at each time step in the x- and y-axis direction. Eq. (2) denotes
the mapping mechanism for target user $i$ considering the orientation $O$, speed
$S$ and position $P$ of all the neighboring agents $j \in \mathcal{N}(i)$ that coexist with the
target agent $i$ at each time step.

$$
\text{Map}_i^t = \sum_{j \in \mathcal{N}(i)} (O, S, P) |(x_j^t - x_i^t, y_j^t - y_i^t, \Delta x_j^t - \Delta x_i^t, \Delta y_j^t - \Delta y_i^t)|. \tag{2}
$$

The orientation layer $O$ represents the heading direction of neighboring agents.
The value of the orientation is in degree $[0, 360]$ and then is mapped to $[0, 1]$. The
value of each grid is the mean of the orientations of all the agents existing within
the grid. The speed layer $S$ represents all the neighboring agents’ travel speed.
Locally, the speed in each grid is the mean speed of all the agents within a grid.
Globally, across all the grids, the value of speed is normalized by the Min-Max
normalization scheme into $[0, 1]$. The position layer $P$ stores the positions of
all the neighboring agents in the grids calculated by Eq. (2). The value of the
corresponding grid is the number of individual neighboring agents existing in the
grid normalized by the total number of all of the neighboring agents at that time
step, which can be interpreted as the grid’s occupancy density. Each time step
has a dynamic map and therefore the spatio-temporal interaction information
among agents are interpreted dynamically over time.

To more intuitively show the dynamic maps information, we gather all the
agents over all the time steps and visualize them in Fig. 4 as an example show-
cased by the dataset nexus-0 (see more information on the benchmark datasets
in Sec 4.1). Each rectangular grid cell is of 1 meter for both width and height
and up to 16 meters in each direction centralized on the target agent is within
the region of interest, in order to include not only close but also distant neighboring agents. The visualization demonstrates certain motion patterns of the agents, including the distribution of orientation, speed and position over the grids of the maps. For example, all the agents move in a certain direction with similar speed on a particular area of the maps, and some areas are much denser than the others.

3.4.1. Attentive Maps Encoder

As discussed above, each time step has a dynamic map which summaries the orientation, speed and position information of all the neighboring agents. To capture the spatio-temporal interconnections from the dynamic maps for the following modules, we propose the *Attentive Maps Encoder* module.

The X-Encoder is used to encode the past information. It has two branches in parallel to process the motion information (upper branch) and dynamic maps information for interaction (lower branch). The upper branch takes as motion information input the offsets \( \sum_t^T (\Delta x_t^i, \Delta y_t^i) \) for each target agent over the observed time steps. The motion information firstly is passed to an 1D convolutional layer (Conv1D) with one-step stride along the time axis to learn motion features one time step after another. Then it is passed to a fully connected (fc) layer. The output of the fc layer is passed to an LSTM module for encoding the temporal features along the trajectory sequence of the target agent into a
hidden state, which contains all the motion information.

The lower branch takes the dynamic maps \( \sum_i^T \text{Map}_i \) as input. The interaction information at each time step is passed through a 2D convolutional layer (Conv2D) with ReLU activation and Max Pooling layer (MaxPool) to learning the spatial features among all the agents. The output of MaxPool at each time step is flattened and concatenated along the time axis to form a timely distributed feature vector. Then, the feature vector is fed forward to a self-attention module to learn the interaction information with an attention mechanism. Here, we adopt the multi-head attention method from Transformer, which is a linear projecting of multiple self-attention in parallel and concatenating them together [37].

The attention function is described as mapping a query and a set of key-value pairs to an output. The query \( (Q) \), keys \( (K) \) and values \( (V) \) are transformed from the spatial features, which are encoded in the above step, by linear transformations:

\[
Q = \pi(\text{Map})W_Q, \ W_Q \in \mathbb{R}^{D \times D_q},
\]
\[
K = \pi(\text{Map})W_K, \ W_K \in \mathbb{R}^{D \times D_k},
\]
\[
V = \pi(\text{Map})W_V, \ W_V \in \mathbb{R}^{D \times D_v},
\]

where \( W_Q, W_K \) and \( W_V \) are the trainable parameters and \( \pi(\cdot) \) indicates the encoding function of the dynamic maps. \( D_q, D_k \) and \( D_v \) are the dimension of the vector of query, key, and value (they are the same in the implementation). Then the self-attended features are calculated as:

\[
\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V
\]

This operation is also called scaled dot-product attention [37]. To improve the performance of the attention layer, multi-head attention is applied:

\[
\text{MultiHead}(Q, K, V) = \text{ConCat}(\text{head}_1, ..., \text{head}_h)W_O,
\]
\[
\text{head}_i = \text{Attention}(QW_Q, KW_K, VW_V), \ (4)
\]

\[15\]
where $W_{Qi} \in \mathbb{R}^{D \times D}$ indicates the linear transformation parameters for query in the $i$-th self-attention head and $D_{qi} = \frac{D_q}{\# head}$. It is the same for $W_{Ki}$ and $W_{Vi}$. Note that, $\# head$ is the total number of attention heads and it must be an aliquot part of $D_q$. The outputs of each head are concatenated and passed a linear transformation with parameter $W_O$.

The output of the multi-head attention is passed to an LSTM which is used to encode the dynamic interconnection in time sequence. Both the hidden states (the last output) from the motion LSTM and the interaction LSTM are concatenated and passed to a $fc$ layer for feature fusion, as the complete output of the X-Encoder, which is denoted as $\Phi_X$.

The Y-Encoder has the same structure as X-Encoder, which is used to encode both target agent’s motion and interaction information from the ground truth during the training time. The output of the Y-Encoder is denoted as $\Phi_Y$. The dynamic maps are also leveraged in the Y-Encoder, however, it is not reconstructed from the Decoder (only the future trajectories is reconstructed). This extended structure distinguishes our model from the conventional CVAE structure [22, 23, 24] and the work from [17] that the input and output maintain in the same form.

3.5. Diverse Sample Generation

In the training phase, $\Phi_X$ and $\Phi_Y$ are concatenated and forwarded to two successive $fc$ layers followed by the ReLU activation and then passed two parallel $fc$ layers to produce the mean and standard deviation of the distribution, which are used to re-parameterize $z$ as discussed in Sec. 3.1. Then, $\Phi_X$ is concatenated with $z$ as condition and fed to the following decoder (based on LSTM) to reconstruct $Y$ sequentially. It is worth noting that $\Phi_X$ is used as condition to help reconstruct $Y$ here. The MSE loss $\ell_2(\hat{Y}, Y)$ (reconstruction loss) and the $KL(Q(z|Y,X) || P(z))$ loss are used to train our model. The MSE loss forces the reconstructed results as close as possible to the ground truth and the KL-divergence loss will force the set of latent variables $z$ to be a Gaussian distribution.
During inference at test time, the Y-Encoder is removed and the X-Encoder works in the same way as in the training phase to extract information from observed trajectories. To generate a future prediction sample, a latent variable $z$ is sampled from $\mathcal{N}(0, I)$ and concatenated with $\Phi_X$ (as condition) as the input of the decoder. To generate diverse samples, this step is repeated $N$ times to generate $N$ samples of future prediction conditioned on $\Phi_X$.

To summarize, the overall pipeline of Attentive Maps Encoder Network (AMENet) consists of four modules, namely, X-Encoder, Y-Encoder, Z-Space and Decoder. Each of the modules uses different types of neural networks to process the motion information and dynamic maps information for multi-path trajectory prediction. Fig 2 depicts the pipeline of the framework.

3.6. Trajectories Ranking

A bivariate Gaussian distribution is used to rank the multiple predicted trajectories $\hat{Y}^1, \cdots, \hat{Y}^N$ for each agent. At each time step, the predicted positions $(\hat{x}_{i,n}^{t'}, \hat{y}_{i,n}^{t'})$, where $n \in N$ at time step $t' \in T'$ for agent $i$, are used to fit a bivariate Gaussian distribution $\mathcal{N}(\mu_{xy}, \sigma_{xy}^2, \rho)^{t'}$. The predicted trajectories are sorted by the joint probability density functions $p(.)$ over the time axis using Eq. (5)(6). $\hat{Y}^*$ denotes the most-likely prediction out of $N$ predictions.

$$P(\hat{x}_{i,n}^{t'}, \hat{y}_{i,n}^{t'}) \approx p((\hat{x}_{i,n}^{t'}, \hat{y}_{i,n}^{t'})|N(\mu_{xy}, \sigma_{xy}^2, \rho)^{t'})$$ (5)

$$\hat{Y}^* = \arg \max_{n=1}^N \sum_{t'=1}^{T'} \log P(\hat{x}_{i,n}^{t'}, \hat{y}_{i,n}^{t'})$$ (6)

4. Experiments

In this section, we will introduce the benchmark which is used to evaluate our method, the evaluation metrics and the comparison of results from our method with the ones from the recent state-of-the-art methods. To further justify how each proposed module in our framework impacts the overall performance, we design a series of ablation studies and discuss the results in detail.
4.1. Trajnet Benchmark Challenge Datasets

We verify the performance of the proposed method on the most challenging benchmark Trajnet datasets [25]. It is the most popular large-scale trajectory-based activity benchmark in this domain and provide a uniform evaluation system for fair comparison among different submitted methods.

Trajnet covers a wide range of datasets and includes various types of road users (pedestrians, bikers, skateboarders, cars, buses, and golf cars) that navigate in a real-world outdoor mixed traffic environment. The data were collected from 38 scenes with ground truth for training and the ones from the other 20 scenes without ground truth for test (i.e., open challenge competition). The most popular pedestrian scenes ETH [46] and UCY [47] are also included in the benchmark. Each scene presents various traffic density in different space layout, which makes the prediction task challenging. It requires a model to generalize, in order to adapt to the various complex scenes. Trajectories are recorded as the \( xy \) coordinates in meters or pixels projected on a Cartesian space. Each trajectory provides 8 steps for observation and the following 12 steps for prediction. The duration between two successive steps is 0.4 seconds. However, the pixel coordinates are not in the same scale across the whole benchmark. Without uniforming the pixels into the same scale, it is extremely difficult to train a general model for the whole datasets. Hence, we follow all the previous works [11, 7, 31, 45, 34, 19, 42] that use the coordinates in meters.

In order to train and evaluate the proposed method, as well as the ablative studies, 6 different scenes are selected as offline test set from the 38 scenes in the training set. Namely, they are \textit{bookstore3, coupa3, deathCircle0, gates1, hyang6}, and \textit{nexus0}. The best trained model is based on the evaluation performance on the offline test set and then is used for the online evaluation. Fig. 5 shows the visualization of the trajectories in each scene.

4.2. Evaluation Metrics

The mean average displacement error (MAD) and the final displacement error (FDE) are the two most commonly applied metrics to measure the per-
Figure 5: Visualization of each scene of the offline test set. Sub-figures are not aligned in the same size, in order to demonstrate the very different space size and layout.
formance in terms of trajectory prediction [7, 19, 30].

- MAD is the aligned L2 distance from $Y$ (ground truth) to its prediction $\hat{Y}$ averaged over all time steps. We report the mean value for all the trajectories.

- FDE is the L2 distance of the last position from $Y$ to the corresponding $\hat{Y}$. It measures a model’s ability for predicting the destination and is more challenging as errors accumulate in time.

We evaluate both the most-likely prediction and the best prediction @top10 for the multi-path trajectory prediction. Most-likely prediction is selected by the trajectories ranking mechanism, see Sec 3.6. @top10 prediction means the 10 predicted trajectories with the highest confidence, the one which has the smallest ADE and FDE compared with the ground truth. When the ground truth is not available (for online test), only the most-likely prediction is selected. Then it comes to the single trajectory prediction problem, as most of the previous works did [11, 7, 31, 45, 34, 42].

4.3. Quantitative Results and Comparison

We compare the performance of our method with the most influential previous works and the recent state-of-the-art works published on the Trajnet challenge (up to 14/06/2020) for trajectory prediction to ensure the fair comparison. The compared works include the following models.

- **Social Force** [11] is a rule-based model with the repulsive force for collision avoidance and the attractive force for social connections;

- **Social LSTM** [7] proposes a social pooling with a rectangular occupancy grid for close neighboring agents which is widely adopted thereafter in this domain [17, 20, 34, 2, 6, 13];

- **SR-LSTM** [31] uses a states refinement module for extract social effects between the target agent and its neighboring agents;
• *RED* [45] uses RNN-based Encoder with Multilayer Perceptron (MLP) for trajectory prediction;

• *MX-LSTM* [34] exploits the head pose information of agents to help analyze its moving intention;

• *Social GAN* [19] proposes to utilize the generative models GAN for multi-path trajectory prediction, which is the one of the closest works to our work; (the other one is *DESIRE* [17], however neither the online test nor code was reported. Hence, we do not compare with *DESIRE* for a fairness purpose);

• *Ind-TF* [42] only utilizes the Transformer network [37] for sequence modeling with no consideration for social interactions between agents.

Table 1 lists the performances from above methods and ours on the Trajnet test set measured by MAD, FDE and overall average (MAD + FDE)/2. The data are originally reported on the Trajnet challenge leader board\(^1\). We can see that, our method (AMENet) outperforms the other methods significantly and wins the first place on all metrics. Even thought compared with the most recent model *Ind-TF* [42](under reviewed), our method performs better. Particularly, our method improves the FDE performance with large margin (reducing the error from 1.197 to 1.183 meters). Note that, our model predicts multiple trajectories conditioned on the observed trajectories with the stochastic variable sampled from a Gaussian distribution repeatedly (see Sec. 3.5). We select the most-likely prediction using the proposed ranking method as discussed in Sec. 3.6. The outstanding performances from our method also demonstrate that our ranking method is effective.

\(^1\)http://trajnet.stanford.edu/result.php?cid=1

\(^2\)Our method is named as *ikg_tnt* on the leadboard.
Table 1: Comparison between our method and the state-of-the-art works. The smaller number is better. Best values are highlighted in boldface.

| Model             | Avg. [m] | FDE [m] | MAD [m] |
|-------------------|----------|---------|---------|
| Social LSTM [7]   | 1.3865   | 3.098   | 0.675   |
| Social GAN [19]   | 1.334    | 2.107   | 0.561   |
| MX-LSTM [34]      | 0.8865   | 1.374   | 0.399   |
| Social Force [11] | 0.8185   | 1.266   | 0.371   |
| SR-LSTM [31]      | 0.8155   | 1.261   | 0.370   |
| RED [45]          | 0.78     | 1.201   | 0.359   |
| Ind-TF [42]       | 0.7765   | 1.197   | 0.356   |
| Ours (AMENet)     | **0.7695** | **1.183** | **0.356** |

4.4. Results for Multi-Path Prediction

Multi-path trajectories prediction is one of the main contributions of this work and distinguishes our work from most of the existing works essentially. Here, we discuss its performance w.r.t. multi-path prediction with the latent space. Instead of generating a single prediction, AMENet generates multiple feasible trajectories by sampling the latent variable $z$ multiple times (see Sec 3.1). During training, the motion information and interaction information from observation and ground truth are encoded into the so-called Z-Space (see Fig. 2). The KL-divergence loss forces $z$ to be a normal Gaussian distribution. Fig. 6 shows the visualization of $z$ in two dimensions, with $\mu$ visualized on the left and $\log \sigma$ visualized on the right. From the figure we can see that, the training phase successfully re-parameterizes the variable $z$ into a Gaussian distribution that captures the stochastic properties of agents’ behaviors. When the Y-Encoder is not available in the inference time, the well-trained Z-Space, in turn, enables us to randomly sample a latent variable $z$ from the Gaussian distribution multiple times for generating more than one feasible future trajectories conditioned on the observation.

Table 2 shows the quantitative results for multi-path trajectory prediction.
Predicted trajectories are ranked by top@10 with the prior knowledge of the corresponding ground truth and most-likely ranking if the ground truth is not available. Compared to the most-likely prediction, top@10 prediction yields similar but better performance. It indicates that: 1) the generated multiple trajectories increase the chance to narrow down the errors from the prediction to the ground truth, and 2) the predicted trajectories are feasible (if not, the bad predictions will deteriorate the overall performance and leads to worse results than the most-likely prediction).

Fig. 7 showcases some qualitative examples of multi-path trajectory prediction from our model. We can see that in roundabouts, the interactions between different agents are full of uncertainties and each agent has more possibilities to choose their future paths. We also notice that the predicted trajectories diverge more widely in further time steps. It is reasonable because the further the future is the uncertainty of agents’ intention is higher. It also proves that the ability of predicting multiple plausible trajectories is important for analyzing the movements of road users because of the increasing uncertainty of the future movements. Single prediction provides limited information for analyzing in this case and is likely to lead to false conclusion if the prediction is not
Table 2: Evaluation of multi-path trajectory prediction using AMENet on the offline test set of Trajnet. Predicted trajectories are ranked by top@10 (former) and most-likely and are measured by MAD/FDE.

| Dataset      | Top@10  | Most-likely |
|--------------|---------|-------------|
| bookstore3   | 0.477/0.961 | 0.486/0.979 |
| coupa3       | 0.221/0.432  | 0.226/0.442  |
| deathCircle0 | 0.650/1.280  | 0.659/1.297  |
| gates1       | 0.784/1.663  | 0.797/1.692  |
| hyang6       | 0.534/1.076  | 0.542/1.094  |
| nexus6       | 0.642/1.073  | 0.559/1.109  |
| Average      | 0.535/1.081  | 0.545/1.102  |

Figure 7: Multi-path predictions from AMENet correct/precise in the early steps.
4.5. Ablation Studies

In order to analyze the impact of each proposed module in our framework, i.e., dynamic maps, self-attention, and the extended structure of CVAE, three ablative models are carried out.

- **AMENet**, the full model of our framework.
- **AOENet**, substitutes dynamic maps with occupancy grid [7, 17, 20, 34, 2, 6, 13] in both the X-Encoder and Y-Encoder. This setting is used to validate the contribution from the dynamic maps.
- **MENet**, removes self-attention module in the dynamic maps branch. This setting is used to validate the effectiveness of the self-attention module that learns the spatial interactions among agents alone the time axis.
- **ACVAE**, only uses dynamic maps in X-Encoder. It is equivalent to CVAE [22, 23, 24] with self-attention. This setting is used to validate the contribution of the extended structure for processing the dynamic maps information in the Y-Encoder.

Table 3 shows the quantitative results from the above ablative models. Errors are measured by MAD/FDE on the most-likely prediction. By comparison between AOENet and AMENet we can see that when we replace the dynamic maps with the occupancy grid, both MAD and FDE increase by a remarkable margin across all the datasets. It demonstrates that our proposed dynamic maps are more helpful for exploring the interaction information among agents than the occupancy grid.

We can also see that if the self-attention module is removed (MENet) the performance decreases by a remarkable margin across all the datasets. This phenomena indicates that the self-attention module is effective in learning the interaction among agents from the dynamic maps.

The comparison between ACVAE and AMENet shows that when we remove the extended structure in the Y-Encoder for dynamic maps, the performances, especially FDE, decrease significantly across all the datasets. The extended
Table 3: Evaluation of dynamic maps, self-attention, and the extended structure of CVAE via AOENet, MENet and ACVAE, respectively, in comparison with the proposed model AMENet. Errors are measured by MAD/FDE on the most-likely prediction. Best values are highlighted in bold face.

| Dataset     | AMENet   | AOENet   | MENet    | ACVAE    |
|-------------|----------|----------|----------|----------|
| bookstore3  | **0.486**/**0.979** | 0.574/1.144 | 0.576/1.139 | 0.509/1.030 |
| coupa3      | **0.226**/**0.442** | 0.260/0.509 | 0.294/0.572 | 0.237/0.464 |
| deathCircle0| **0.659**/**1.297** | 0.726/1.437 | 0.725/1.419 | 0.698/1.378 |
| gates1      | **0.797**/**1.692** | 0.878/1.819 | 0.941/1.928 | 0.861/1.823 |
| hyang6      | **0.542**/**1.094** | 0.619/1.244 | 0.657/1.292 | 0.566/1.140 |
| nexus6      | **0.559**/**1.109** | 0.752/1.489 | 0.705/1.140 | 0.595/1.181 |
| Average     | **0.545**/**1.102** | 0.635/1.274 | 0.650/1.283 | 0.578/1.169 |

Structure provides the ability of the model to process the interaction information even in prediction. It improves the model’s performance, especially for predicting more accurate destinations. This improvement has been also confirmed by the benchmark challenge (see Table 1). One interesting observation of the comparison between ACVAE and AOENet/MENet is that, ACVAE performs much better than AOENet and MENet measured by MAD and FDE. This observation further proves that, even without the extended structure in the Y-Encoder, the dynamic maps with self-attention are very beneficial for interpreting the interactions between a target agent and its neighboring agents. Its robustness has been demonstrated by the ablative models across various datasets.

Fig. 8 showcases some examples of the qualitative results of the full AMENet in comparison to the ablative models in different scenes. In general, all the models are able to predict trajectories in different scenes, e.g., intersections and roundabouts, of various traffic density and motion patterns, e.g., standing still or moving fast. Given a short observation of trajectories, i.e., 8 time steps, all the models are able to capture the general speed and heading direction.
for agents located in different areas in the space. AMENet predicts the most accurate trajectories which are very close or even completely overlap with the corresponding ground truth trajectories. Compared with the ablative models, AMENet predicts more accurate destinations (the last position of the predicted trajectories), which is in line with the quantitative results shown in Table 1. One very clear example in hyang6 (Fig. 8e) shows that, when the fast-moving agent changes its motion, AOENet and MENet have limited performance to predict its travel speed and ACVAE has limited performance to predict its destination. One the other hand, the prediction from AMENet is very close to the ground truth.

Nevertheless, our models have limited performance in predicting abnormal trajectories, like suddenly turning around or changing speed drastically. Such scenarios can be found in the lower right corner in gate1 (Fig. 8d). The sudden maneuvers of agents are very difficult to forecast even for human observers.

5. Conclusions

In this paper, we present a generative model called Attentive Maps Encoder Networks (AMENet) that uses motion information and interaction information for multi-path trajectory prediction of mixed traffic in various real-world environments. The latent space learnt by the X-Encoder and Y-Encoder for both sources of information enables the model to capture the stochastic properties of motion behaviors for predicting multiple plausible trajectories after a short observation time. We propose a novel way—dynamic maps—to extract the social effects between agents during interactions. The dynamic maps capture accurate interaction information by encoding the neighboring agents’ orientation, travel speed and relative position in relation to the target agent, and the self-attention mechanism enables the model to learn the global dependency of interaction over different time steps. The efficacy of the model has been validated on the most challenging benchmark Trajnet that contains various datasets in various real-world environments. Our model not only achieves the state-of-the-
Figure 8: Trajectories predicted by AMEnet (AME), AOENet (AOE), ME (MENet), CVAE (ACVAE) and the corresponding ground truth (GT) in different scenes. Sub-figures are not aligned in the same size, in order to demonstrate the very different space size and layout.

art performance, but also wins the first place on the leader board for predicting 12 time-step positions of 4.8 seconds. Each component of AMENet is validated
via a series of ablative studies.

In the future work, we will extend our prediction model for safety prediction, for example, using the predicted trajectories to calculate time-to-collision [48] and detecting abnormal trajectories by comparing the anticipated/predicted trajectories with the actual ones.

Acknowledgement

This work is supported by the German Research Foundation (DFG) through the Research Training Group SocialCars (GRK 1931).

References

[1] B. T. Morris, M. M. Trivedi, A survey of vision-based trajectory learning and analysis for surveillance, Transactions on circuits and systems for video technology 18 (8) (2008) 1114–1127.

[2] H. Cheng, M. Sester, Modeling mixed traffic in shared space using lstm with probability density mapping, in: In Proceedings of ITSC, IEEE, 2018, pp. 3898–3904.

[3] H. Cheng, W. Liao, M. Y. Yang, M. Sester, B. Rosenhahn, Mcenet: Multi-context encoder network for homogeneous agent trajectory prediction in mixed traffic (2020).

[4] K. Schindler, A. Ess, B. Leibe, L. Van Gool, Automatic detection and tracking of pedestrians from a moving stereo rig, ISPRS Journal of Photogrammetry and Remote Sensing 65 (6) (2010) 523–537.

[5] T. Klinger, F. Rottensteiner, C. Heipke, Probabilistic multi-person localisation and tracking in image sequences, ISPRS Journal of Photogrammetry and Remote Sensing 127 (2017) 73–88.
[6] H. Cheng, M. Sester, Mixed traffic trajectory prediction using lstm–based models in shared space, in: The Annual International Conference on Geographic Information Science, Springer, 2018, pp. 309–325.

[7] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, S. Savarese, Social lstm: Human trajectory prediction in crowded spaces, in: In Proceedings of CVPR, 2016, pp. 961–971.

[8] N. Mohajerin, M. Rohani, Multi-step prediction of occupancy grid maps with recurrent neural networks, in: In Proceedings of CVPR, 2019, pp. 10600–10608.

[9] M. Mohanan, A. Salgoankar, A survey of robotic motion planning in dynamic environments, Robotics and Autonomous Systems 100 (2018) 171–185.

[10] S. Reid, DfT Shared Space Project Stage 1: Appraisal of Shared Space, MVA Consultancy, 2009.

[11] D. Helbing, P. Molnar, Social force model for pedestrian dynamics, Physical review E 51 (5) (1995) 4282.

[12] A. Rudenko, L. Palmieri, M. Herman, K. M. Kitani, D. M. Gavrila, K. O. Arras, Human motion trajectory prediction: A survey, arXiv preprint arXiv:1905.06113 (2019).

[13] F. T. Johora, H. Cheng, J. P. Müller, M. Sester, An agent-based model for trajectory modelling in shared spaces: A combination of expert-based and deep learning approaches, in: Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems, 2020, pp. 1878–1880.

[14] R. A. Best, J. Norton, A new model and efficient tracker for a target with curvilinear motion, Transactions on Aerospace and Electronic Systems 33 (3) (1997) 1030–1037.
[15] H. Cheng, F. T. Johora, M. Sester, J. P. Müller, Trajectory modelling in shared spaces: Expert-based vs. deep learning approach?, in: International Workshop on Multi-Agent Systems and Agent-Based Simulation, 2020.

[16] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436.

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

[18] A. Vemula, K. Muelling, J. Oh, Social attention: Modeling attention in human crowds, in: In Proceedings of ICRA, IEEE, 2018, pp. 1–7.

[19] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, A. Alahi, Social gan: Socially acceptable trajectories with generative adversarial networks, in: In Proceedings of CVPR, 2018, pp. 2255–2264.

[20] H. Xue, D. Q. Huynh, M. Reynolds, Ss-lstm: A hierarchical lstm model for pedestrian trajectory prediction, in: In Proceedings of WACV, 2018, pp. 1186–1194.

[21] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in: In Proceedings of NIPS, 2014, pp. 2672–2680.

[22] D. P. Kingma, M. Welling, Auto-encoding variational bayes, arXiv preprint arXiv:1312.6114 (2013).

[23] D. P. Kingma, S. Mohamed, D. J. Rezende, M. Welling, Semi-supervised learning with deep generative models, in: In Proceedings of NIPS, 2014, pp. 3581–3589.

[24] K. Sohn, H. Lee, X. Yan, Learning structured output representation using deep conditional generative models, in: In Proceedings of NIPS, 2015, pp. 3483–3491.
[25] A. Sadeghian, V. Kosaraju, A. Gupta, S. Savarese, A. Alahi, Trajnet: Towards a benchmark for human trajectory prediction, arXiv preprint (2018).

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

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

[28] K. M. Kitani, B. D. Ziebart, J. A. Bagnell, M. Hebert, Activity forecasting, in: In Proceedings of ECCV, Springer, 2012, pp. 201–214.

[29] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (8) (1997) 1735–1780.

[30] A. Sadeghian, V. Kosaraju, A. Sadeghian, N. Hirose, S. Savarese, Sophie: An attentive gan for predicting paths compliant to social and physical constraints, arXiv preprint arXiv:1806.01482 (2018).

[31] P. Zhang, W. Ouyang, P. Zhang, J. Xue, N. Zheng, Sr-lstm: State refinement for lstm towards pedestrian trajectory prediction, in: In Proceedings of CVPR, 2019, pp. 12085–12094.

[32] R. Chandra, U. Bhattacharya, A. Bera, D. Manocha, Traphic: Trajectory prediction in dense and heterogeneous traffic using weighted interactions, in: In Proceedings of the CVPR, 2019, pp. 8483–8492.

[33] C. Tang, R. R. Salakhutdinov, Multiple futures prediction, in: In Proceedings of NIPS, 2019, pp. 15398–15408.

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

[35] D. Bahdanau, K. Cho, Y. Bengio, Neural machine translation by jointly learning to align and translate, arXiv preprint arXiv:1409.0473 (2014).
[36] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, Y. Bengio, Show, attend and tell: Neural image caption generation with visual attention, in: In Proceedings of ICML, 2015, pp. 2048–2057.

[37] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, in: In Proceedings of NIPS, 2017, pp. 5998–6008.

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

[39] S. He, W. Liao, H. R. Tavakoli, M. Yang, B. Rosenhahn, N. Pugeault, Image captioning through image transformer, arXiv preprint arXiv:2004.14231 (2020).

[40] D. Varshneya, G. Srinivasaraghavan, Human trajectory prediction using spatially aware deep attention models, arXiv preprint arXiv:1705.09436 (2017).

[41] A. Al-Molegi, M. Jabreel, A. Martinez-Balleste, Move, attend and predict: An attention-based neural model for peoples movement prediction, Pattern Recognition Letters 112 (2018) 34–40.

[42] F. Giuliari, I. Hasan, M. Cristani, F. Galasso, Transformer networks for trajectory forecasting, arXiv preprint arXiv:2003.08111 (2020).

[43] J. Amirian, J.-B. Hayet, J. Pettré, Social ways: Learning multi-modal distributions of pedestrian trajectories with gans, in: In Proceedings of CVPR Workshops, 2019, pp. 0–0.

[44] D. J. Rezende, S. Mohamed, D. Wierstra, Stochastic backpropagation and approximate inference in deep generative models, arXiv preprint arXiv:1401.4082 (2014).
[45] S. Becker, R. Hug, W. Hübner, M. Arens, An evaluation of trajectory prediction approaches and notes on the trajnet benchmark, arXiv preprint arXiv:1805.07663 (2018).

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

[47] A. Lerner, Y. Chrysanthou, D. Lischinski, Crowds by example, in: In Proceedings of Computer Graphics Forum, Vol. 26, Wiley Online Library, 2007, pp. 655–664.

[48] S. R. Perkins, J. L. Harris, Traffic conflict characteristics-accident potential at intersections, Highway Research Record (225) (1968).