ISPRS Journal of Photogrammetry and Remote Sensing 172 (2021) 253–266

Contents lists available at ScienceDirect

ISPRS Journal of Photogrammetry and Remote Sensing

journal homepage: www.elsevier.com/locate/isprsjprs

AMENet: Attentive Maps Encoder Network for trajectory prediction

Hao Cheng [a,1], Wentong Liao [b,*,1], Michael Ying Yang [c,*], Bodo Rosenhahn [b], Monika Sester [a]

[a] Institute of Cartography and Geoinformatics, Leibniz University Hannover, Germany
[b] Institute of Information Processing, Leibniz University Hannover, Germany
[c] Scene Understanding Group, Faculty of Geo-Information Science and Earth Observation (ITC), University of Twente, the Netherlands

ARTICLE INFO

Keywords:
Trajectory prediction
Generative model
Encoder

ABSTRACT

Trajectory prediction is critical for applications of planning safe future movements and remains challenging even for the next few seconds in urban mixed traffic. How an agent moves is affected by the various behaviors of its neighboring agents in different environments. To predict movements, we propose an end-to-end generative model named *Attentive Maps Encoder Network (AMENet)* that encodes the agent's motion and interaction information for accurate and realistic multi-path trajectory prediction. A conditional variational auto-encoder module is trained to learn the latent space of possible future paths based on attentive dynamic maps for interaction modeling and then is used to predict multiple plausible future trajectories conditioned on the observed past trajectories. The efficacy of AMENet is validated using two public trajectory prediction benchmarks *Trajnet* and *InD*.

1. Introduction

Accurate trajectory prediction is a crucial task in different communities, such as intelligent transportation systems (ITS) for traffic management and autonomous driving (Morris and Trivedi, 2008; Cheng and Sester, 2018; Cheng et al., 2020), photogrammetry mapping and extraction (Schindler et al., 2010; Klinger et al., 2017; Cheng and Sester, 2018; Ma et al., 2019), computer vision (Alahi et al., 2016; Mohajerin and Rohani, 2019) and mobile robot applications (Mohanan and Salgoankar, 2018). It enables an intelligent system to foresee the behaviors of road users and make a reasonable and safe decision for the next operation. It is defined as the prediction of plausible (e. g., collision free and energy efficient) and socially-acceptable (e. g., considering social rules, norms, and relations between agents) positions in 2D/3D of non-erratic target agents (pedestrians, cyclists, vehicles and other types (Rudenko et al., 2020)) at each step within a predefined future time interval relying on observed partial trajectories over a certain period of discrete time steps (Helbing and Molnar, 1995; Alahi et al., 2016). A prediction process in mixed traffic is exemplified in Fig. 1.

How to effectively predict accurate trajectories for heterogeneous agents still remains challenging due to: (1) the complex behavior and uncertain moving intention of each agent, (2) the presence and interactions between agents, and (3) multi-path choices: there is usually more than one socially-acceptable path that an agent could use in the future.

Boosted by Deep Learning (DL) (LeCun et al., 2015) technologies and the availability of large-scale real-world datasets and benchmarks, recent methods utilizing Recurrent Neural Networks (RNNs) and/or Convolutional Neural Networks (CNNs) have made significant progress in modeling the interactions between agents and predicting their future trajectories (Alahi et al., 2016; Lee et al., 2017; Vemula et al., 2018; Gupta et al., 2018; Xue et al., 2018; Cheng et al., 2020). However, it is difficult for those methods to distinguish the effects of heterogeneous neighboring agents in different situations. For example, the target vehicle is affected more by the pedestrians in front of it tending to cross the road than by the following vehicles. Besides, minimizing the Euclidean distance between the ground truth and the prediction is commonly used as the objective function in some discriminative models (Vemula et al., 2018; Xue et al., 2018), which produce a deterministic outcome and is likely to predict the "average" trajectories. In this regard, generative models (Goodfellow et al., 2014; Kingma and Welling, 2014; Kingma et al., 2014; Sohn et al., 2015) are proposed for predicting multiple socially-acceptable trajectories (Lee et al., 2017; Gupta et al., 2018). In spite of the great progress, most of these methods are designed

* Corresponding authors.
E-mail addresses: hao.cheng@ikg.uni-hannover.de, hao.cheng@ikg.uni-hannover.de (H. Cheng), liao@tnt.uni-hannover.de (W. Liao), michael.yang@utwente.nl (M.Y. Yang), rosenhahn@tnt.uni-hannover.de (B. Rosenhahn), monika.sester@ikg.uni-hannover.de (M. Sester).
[1] Joint first author, arranged in alphabetical order.

https://doi.org/10.1016/j.isprsjprs.2020.12.004
Received 15 June 2020; Received in revised form 7 December 2020; Accepted 8 December 2020
Available online 14 January 2021

Fig. 1. Predicting future positions of agents (e. g., target agent in black) at each step within a predefined time interval by observing their past trajectories in mixed traffic situations.

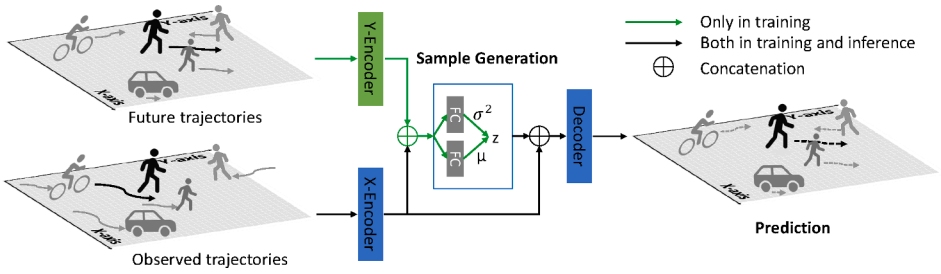


Fig. 2. An overview of the proposed framework. It consists of four modules: the X-Encoder and Y-Encoder are used for encoding the observed and the future trajectories, respectively. They have an identical structure. The Sample Generator produces diverse samples conditioned on the input of the previous encoders. The Decoder is used to decode the features from the produced samples and predicts the future trajectories sequentially. FC stands for fully connected layer. The specific structure of the X-Encoder/Y-Encoder is given by Fig. 3.

for homogeneous agents (e. g., only pedestrians). An important research question remains open: how to predict accurate trajectories in different scenes for all the various types of heterogeneous agents?.

To address this problem, we propose a model named *Attentive Maps Encoder Network* (AMENet). It inherits the ability of deep conditional generative models (Sohn et al., 2015) using Gaussian latent variables for modeling complex future trajectories and learns the interactions between agents by attentive dynamic maps. The interaction module manipulates the information extracted from the neighboring agents' orientation, speed and position in relation to the target agent at each step and the attention mechanism (Vaswani et al., 2017) enables the module to automatically focus on the salient features extracted over different steps. Fig. 2 gives an overview of the model. Two encoders learn the representations of an agent's behavior into a latent space: the X-Encoder learns the information from the observed trajectories, while the Y-Encoder learns the information from the future trajectories of the ground truth and is removed in the inference phase. The Decoder is trained to predict the future trajectories conditioned on the information learned by the X-Encoder and the representations sampled from the latent space.

The main contributions of this work are summarized as follows:

1. The generative framework AMENet encodes uncertainties of an agent's behavior into the latent space and predicts multi-path trajectories.
2. A novel module, *attentive dynamic maps*, learns spatio-temporal interconnections between agents considering their orientation, speed and position.
3. It predicts accurate trajectories for heterogeneous agents in various unseen real-world environments, rather than focusing on homogeneous agents.

The efficacy of AMENet is validated using the recent benchmarks *Trajnet* (Sadeghian et al., 2018) that contains 20 unseen scenes in various environments and InD (Bock et al., 2019) of four different intersections for trajectory prediction. Each module of AMENet is validated via a series of ablation studies. Its detailed implementation information is given in Appendix C and the source code is available at https://github.com/haohao11/AMENet.

2. Related work

Trajectory prediction has been studied for decades and we discuss the most relevant works with respect to sequence prediction, interaction modeling and generative models for multi-path prediction.

2.1. Sequence modeling

Modeling trajectories as sequences is one of the most common approaches. The 2D/3D positions of an agent are predicted step by step. The widely applied models include linear regression and Kalman filter (Harvey, 1990), Gaussian processes (Tay and Laugier, 2008) and Markov decision processing (Kitani et al., 2012). These traditional methods largely rely on the quality of manually designed features and have limited performance in tackling large-scale data. Benefiting from the development of DL technologies (LeCun et al., 2015) in recent years, the RNNs and Long Short-Term Memories (LSTMs) (Hochreiter and Schmidhuber, 1997) are inherently designed for sequence prediction tasks and successfully applied for predicting pedestrian trajectories (Alahi et al., 2016; Gupta et al., 2018; Sadeghian et al., 2019; Zhang et al., 2019) and other types of road users (Mohajerin and Rohani, 2019; Chandra et al., 2019; Tang and Salakhutdinov, 2019). In this work, we use 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 can be crucially affected by others. Therefore, effectively modeling the interactions is important for accurate trajectory prediction. The negotiation between road agents is simulated by Game Theory (Johora et al., 2020) or Social Forces (Helbing and Molnar, 1995), i. e., the repulsive force for collision avoidance and the attractive force for social connections. Such rule-based interaction modelings have been incorporated into DL models. Social LSTM (Alahi et al., 2016) proposes an occupancy grid to map the positions of close neighboring agents and uses a Social pooling layer to encode the interaction information for trajectory prediction. Many works design their specific "occupancy" grid (Lee et al., 2017; Xue et al., 2018; Hasan et al., 2018; Cheng and Sester, 2018; Cheng and Sester, 2018; Johora et al., 2020. Cheng et al. (2020) consider the interactions between individual and group agents with social connections and report better performance. Meanwhile, different pooling mechanisms are proposed. The generative adversarial network (GAN) (Goodfellow et al., 2014) based model Social GAN (Gupta et al., 2018) embeds relative positions between the target and all the other agents and then uses an element-wise pooling to extract the interaction between all the pairs of agents; The SR-LSTM (States Refinement LSTM) model (Zhang et al., 2019) proposes a states refinement module for aligning all the agents

together and adaptively refines the state of each agent through a message passing framework. However, only the position information is leveraged in most of the above DL models. The interaction dynamics are not fully captured both in spatial and temporal domains.

2.3. Modeling with attention

Attention mechanisms (Bahdanau et al., 2015; Xu et al., 2015; Vaswani et al., 2017) have been utilized to extract semantic information for predicting trajectories (Varshneya and Srinivasaraghavan, 2017; Sadeghian et al., 2019; Al-Molegi et al., 2018; Giuliari et al., 2020). A soft attention mechanism (Xu et al., 2015) is incorporated in LSTMs to learn the spatio-temporal patterns from the position coordinates Varshneya and Srinivasaraghavan, 2017. SoPhie (Sadeghian et al., 2019) applies two separate soft attention modules: the physical attention learns salient agent-to-scene features and the social attention models agent-to-agent interactions. In the MAP model (Move, Attend, and Predict) (Al-Molegi et al., 2018), an attentive network is implemented to learn the relationships between the location and time information. The most recent work Ind-TF (Giuliari et al., 2020) utilizes the Transformer network (Vaswani et al., 2017) for modeling trajectory sequences. Transformer is a type of neural network structure for modeling sequences and widely applied in machine translation for sequence prediction. In this work, we model the dynamic interactions among all road users by utilizing the self-attention mechanism (Vaswani et al., 2017) along the time axis.

2.4. Generative models

Nowadays, in the era of DL, GAN (Goodfellow et al., 2014; Kingma and Welling, 2014) and the variants such as CVAE (Kingma et al., 2014; Sohn et al., 2015), are the most popular generative models. Gupta et al. (2018) trained a generator to generate future trajectories from noise and a discriminator to judge whether the generated ones are fake or not. The performances of the two modules are enhanced mutually and the generator is able to generate trajectories that are as precise as the real ones. Amirian et al. (2019) propose a GAN-based model for generating multiple plausible trajectories for pedestrians. The CVAE model is used to predict multi-path trajectories conditioned on the observed trajectories (Lee et al., 2017), as well as scene context (Cheng et al., 2020). Besides the generative models, Makansi et al. (2019) treat the multi-path trajectory prediction problem as multi-model distributions estimation. Their method first predicts multi-model distributions with an evolving strategy by combining Winner-Takes-ALL loss (Guzman-Rivera et al., 2012), and then fits a distribution to the samples from the first stage for trajectory prediction.

In this work, we incorporate a CVAE module to learn a latent space for predicting multiple plausible future trajectories conditioned on the observed trajectories. Our work essentially differs from the above models in the following ways. Interactions are modeled by the dynamic maps considering (1) not only position, but also orientation and speed and (2) are automatically extracted with the self-attention mechanism, and (3) the interactions associated with the ground truth are also encoded into the latent space, which is different from a conventional CVAE model only "auto-encoding" the ground truth trajectories (Lee et al., 2017).

3. Method

In this section, we introduce the proposed model AMENet (Fig. 2) in detail in the following structure: a brief review on the *CVAE* (Section 3.1), the detailed structure of *AMENet* (Section 3.2) and the *Feature Encoding* (Section 3.3) of the motion input and the attentive dynamic maps.

3.1. Diverse sample generation with CVAE

The CVAE model is an extension of the generative model VAE (Kingma and Welling, 2014) that introduces a condition to control the output (Kingma et al., 2014). More details of the theory is provided in Appendix A. The following describes the basics of the CVAE model. Given a set of samples $(\boldsymbol{X},\boldsymbol{Y}) = ((\boldsymbol{X}_1,\boldsymbol{Y}_1),\cdots,(\boldsymbol{X}_N,\boldsymbol{Y}_N))$, it jointly learns a recognition model $q_\phi(\mathbf{z}|\boldsymbol{Y},\boldsymbol{X})$ of a variational approximation of the true posterior $p_\theta(\mathbf{z}|\boldsymbol{Y},\boldsymbol{X})$ and a generation model $p_\theta(\boldsymbol{Y}|\boldsymbol{X},z)$ for predicting the output $\boldsymbol{Y}$ conditioned on the input $\boldsymbol{X}$. 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*:

$$\begin{aligned}\log p_\theta(\boldsymbol{Y}|\boldsymbol{X}) &= \log\sum_z p_\theta\left(\boldsymbol{Y},z|\boldsymbol{X}\right)\\ &= \log\left(\sum_z q_\phi\left(z|\boldsymbol{X},\boldsymbol{Y}\right)\frac{p_\theta(\boldsymbol{Y}|\boldsymbol{X},z)p_\theta(z|\boldsymbol{X})}{q_\phi(z|\boldsymbol{X},\boldsymbol{Y})}\right).\end{aligned} \tag{1}$$

By means of Jensen's inequality, the evidence lower bound can be obtained:

$$\log p_\theta\left(\boldsymbol{Y}|\boldsymbol{X}\right) \geqslant -D_{KL}\left(q_\phi\left(\mathbf{z}|\boldsymbol{X},\boldsymbol{Y}\right)\|p_\theta\left(\mathbf{z}\right)\right) + \mathbb{E}_{q_\phi(\mathbf{z}|\boldsymbol{X},\boldsymbol{Y})}\left[\log p_\theta\left(\boldsymbol{Y}|\boldsymbol{X},\mathbf{z}\right)\right]. \tag{2}$$

Here both the approximate posterior $q_\phi(\mathbf{z}|\boldsymbol{X},\boldsymbol{Y})$ and the prior $p_\theta(\mathbf{z})$ are assumed to be Gaussian distributions for an analytical solution (Kingma and Welling, 2014). During training, the Kullback–Leibler divergence $D_{KL}(\cdot)$ pushes the approximate posterior to the prior distribution $p_\theta(\mathbf{z})$. The generation error $\mathbb{E}_{q_\phi(\mathbf{z}|\boldsymbol{X},\boldsymbol{Y})}(\cdot)$ measures the distance between the generated output and the ground truth. During inference, for a given observation $\boldsymbol{X}_i$, one latent variable z is drawn from the prior distribution $p_\theta(\mathbf{z})$, and one of the possible outputs $\widehat{\boldsymbol{Y}}_i$ is generated from the distribution $p_\theta(\boldsymbol{Y}_i|\boldsymbol{X}_i,z)$. The latent variables $\mathbf{z}$ allow for the one-to-many mapping from the condition to the output via multiple sampling.

3.2. Attentive encoder network for trajectory prediction

In tasks like trajectory prediction, we are interested in modeling a conditional distribution $p_\theta(\boldsymbol{Y}_n|\boldsymbol{X})$, where $\boldsymbol{X}$ is the past trajectory information and $\boldsymbol{Y}_n$ is one of its possible future trajectories. In order to realize this goal that generates controllable samples of future trajectories based on past trajectories, a CVAE module is adopted inside our framework. The multi-path trajectory prediction problem with the consideration of motion and interaction information is defined as follows: agent i, receives as input its observed trajectories $\boldsymbol{X}_i = \{X_i^1,\cdots,X_i^T\}$ for predicting its n-th plausible future trajectory $\widehat{\boldsymbol{Y}}_{i,n} = \{\widehat{Y}_{i,n}^1,\cdots,\widehat{Y}_{i,n}^{T'}\}$. T and T' denote the total number of steps of the past and future trajectories, respectively. The trajectory position of i is characterized by the coordinates as $X_i^t = (x_i{}^t, y_i^t)$ at step t or as $\widehat{Y}_{i,n}^{t'} = (\widehat{x}_{i,n}^{t'}, \widehat{y}_{i,n}^{t'})$ at step t'. 3D coordinates are also possible, but in this work only 2D coordinates are considered. The objective is to predict its multiple plausible future trajectories $(\widehat{\boldsymbol{Y}}_{i,1},\cdots,\widehat{\boldsymbol{Y}}_{i,N})$ that are as accurate as possible to the ground truth $\boldsymbol{Y}_i$. This task is mathematically defined as $\widehat{\boldsymbol{Y}}_{i,n} = f(\boldsymbol{X}_i,\mathrm{AMap}_i)$ and $n \leqslant N$. Here, N denotes the total number of the predicted trajectories and AMap_i denotes the attentive dynamic maps centralized on the target agent for mapping the interactions with its neighboring agents over the steps. More details of the attentive dynamic maps will be given in Section 3.3.2.

We extend the CVAE model as follows to solve this problem:

$$\begin{aligned}f(\boldsymbol{X}_i,\mathrm{AMap}_i) &= \log p_\theta(\boldsymbol{Y}_i|\boldsymbol{X}_i,\mathrm{AMap}_i),\\ &\geqslant -D_{KL}(q_\phi(\mathbf{z}|\boldsymbol{X}_i\boldsymbol{Y}_i,\mathrm{AMap}_i)\|p_\theta(\mathbf{z}))\\ &\quad +\mathbb{E}_{q_\phi(\mathbf{z}|\boldsymbol{X}_i,\boldsymbol{Y}_i,\mathrm{AMap}_i)}\left[\log p_\theta\left(\boldsymbol{Y}_i|\boldsymbol{X}_i,\mathrm{AMap}_i,\mathbf{z}\right)\right].\end{aligned} \tag{3}$$

Note that for simplicity, the notation of steps T and T' is omitted. $q_\phi(\cdot)$ accesses the interactions captured by $(\mathrm{AMap}_i)_{t=1}^{T}$ and $(\mathrm{AMap}_i)_{t'=1}^{T'}$,

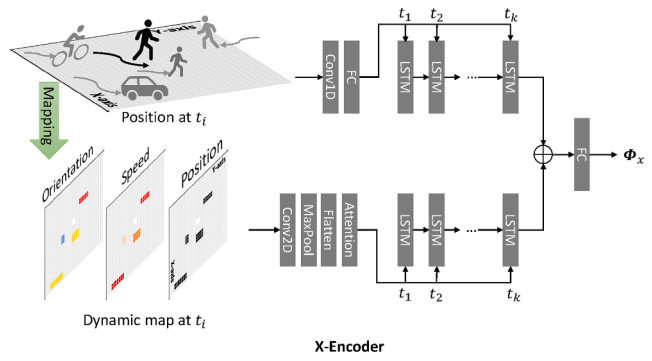


Fig. 3. The structure of the X-Encoder. The upper branch extracts motion information of target agents and the lower one learns the interaction information between neighboring agents from the dynamic maps over time attentively. The motion information and the interaction information are encoded by their respective LSTMs sequentially. The last outputs of the two LSTMs are concatenated and forwarded to a fully connected (FC) layer to get the final output of the X-Encoder. The Y-Encoder has the same structure as the X-Encoder.

respectively, from both the observation and the future time, while $p_\theta(\cdot)$ only accesses the interactions captured by $(\mathrm{AMap}_i)_{t=1}^{T}$ from the observation time.

In the training phase, $q_\phi(\cdot)$ and $p_\theta(\cdot)$ are jointly learned. The recognition model is trained via the X-Encoder and Y-Encoder. The encoded outputs from both encoders are concatenated and then forwarded to two side-by-side fully connected (FC) layers to produce the mean and the standard deviation of the latent variables $\mathbf{z}$. The generation model is trained via the Decoder. It takes the output of the X-Encoder as the condition and the latent variables to generate the future trajectory. We employ an LSTM network in the Decoder for predicting the future trajectory sequentially. The Mean Squared Error (MSE) between the predicted trajectories and the ground-truth ones is used as the reconstruction loss. During inference, the Y-Encoder is removed and the X-Encoder works in the same way as in the training phase. The Decoder generates a prediction conditioned on the output of the X-Encoder and the sampled latent variable z. This step is repeated N times to predict multiple trajectories.

Fig. 3 shows the detailed structure of the X-Encoder/Y-Encoder, which are designed for learning the information from the motion input and the attentive dynamic maps. 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 the offsets $(\Delta x_i^t, \Delta y_i^t)_{t=1}^{T}$ for each target agent over the observed steps. The motion information firstly is passed to a 1D convolutional layer (Conv1D) with a one-step stride along the time axis to learn motion features one step after another. Then the output is sequentially passed to a FC layer and an LSTM module for encoding the temporal features into a hidden state, which contains all the motion information of the target agent. The lower branch takes the dynamic maps $(\mathrm{Map}_i^t)_{t=1}^{T}$ as input. The interaction information at each step is passed through a 2D convolutional layer (Conv2D) with the ReLU activation and a Max Pooling layer (MaxPool) for learning the spatial features among all the agents. The output of MaxPool at each step is flattened and concatenated along the time axis to form a timely distributed feature vector. Then, the feature vector is fed forward to the attention layer for learning the interaction information. The output of the attention layer is passed to an LSTM used to encode the dynamic interconnection in the 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.

The Y-Encoder has the same structure as the X-Encoder, which is used to encode both the target agent's motion and interaction information from the ground truth during the training time. The dynamic maps are also leveraged in the Y-Encoder, although, they are not reconstructed from the Decoder (only the future trajectories are reconstructed). This extended structure distinguishes our model from the conventional CVAE structure (Kingma and Welling, 2014; Kingma et al., 2014; Sohn et al., 2015) and the work from Lee et al. (2017), in which only the ground truth trajectories are inserted for training the recognition model (see Section 3.1).

For tasks of single-path prediction, such as the Trajnet challenge or path planning, a ranking strategy is proposed to select the *most-likely* predicted trajectory out of the multiple predictions. We apply a bivariate Gaussian distribution to rank the predicted trajectories $(\widehat{\boldsymbol{Y}}_{i,1}, \cdots, \widehat{\boldsymbol{Y}}_{i,N})$ for each agent. At step t', all the predicted positions for the agent i are stored in the vector $\left|\widehat{\mathrm{X}}_i, \widehat{\mathrm{Y}}_i\right|^{t'}$. We follow the work (Graves, 2013) to fit the positions into a bivariate Gaussian distribution:

$$f(\hat{x}_i, \hat{y}_i)^{t'} = \frac{1}{2\pi\mu_{\widehat{\mathrm{X}}_i}\mu_{\widehat{\mathrm{Y}}_i}\sqrt{1-\rho^2}}\exp\frac{-Z}{2(1-\rho^2)}, \tag{4}$$

where

$$Z = \frac{\left(\hat{x}_i - \mu_{\widehat{\mathrm{X}}_i}\right)^2}{\sigma^2_{\widehat{\mathrm{X}}_i}} + \frac{\left(\hat{y}_i - \mu_{\widehat{\mathrm{Y}}_i}\right)^2}{\sigma^2_{\widehat{\mathrm{Y}}_i}} - \frac{2\rho\left(\hat{x}_i - \mu_{\widehat{\mathrm{X}}_i}\right)\left(\hat{y}_i - \mu_{\widehat{\mathrm{Y}}_i}\right)}{\sigma_{\widehat{\mathrm{X}}_i}\sigma_{\widehat{\mathrm{Y}}_i}}. \tag{5}$$

μ denotes the mean and σ the standard deviation. ρ is the correlation between $\widehat{\mathrm{X}}_i$ and $\widehat{\mathrm{Y}}_i$. A predicted trajectory is scored as the sum of the relative likelihood of all of its steps:

$$S\left(\widehat{\mathrm{Y}}_{i,n}\right) = \sum_{t'=1}^{T'} f(\hat{x}_i, \hat{y}_i)^{t'}. \tag{6}$$

All the predicted trajectories are ranked according to this score. The one with the highest score is selected for the single-path prediction.

3.3. Feature encoding

In this subsection we discuss how to encode the motion and interaction information in detail.

3.3.1. Motion input

The motion information for each agent is captured by the position coordinates at each step. Specifically, we use the offset of the trajectory positions between two consecutive steps $(\Delta x^t, \Delta y^t) = (x^{t+1} - x^t, y^{t+1} - y^t)$as the motion information, which has been widely applied in this domain (Gupta et al., 2018; Becker et al., 2018; Zhang et al., 2019; Cheng et al., 2020). Compared to coordinates, the offset is independent from the given space and less sensitive with respect to overfitting a model to particular space or travel directions. It is interpreted as speed over steps that are defined with a constant duration. The coordinates at each position are calculated back by cumulatively summing the sequence offsets from the given original position. The class information of agent's type is useful for analyzing its motion (Cheng et al., 2020). However, the Trajnet benchmark does not provide this information for the trajectories and we do not use it here. As augmentation technique we randomly rotate the trajectories to prevent the model 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.

3.3.2. Attentive dynamic maps

We propose a novel and straightforward method: attentive dynamic maps to learn agent-to-agent interaction information. The mapping method is inspired by the recent works of parsing the interactions between agents based on an occupancy grid (Alahi et al., 2016; Lee et al.,

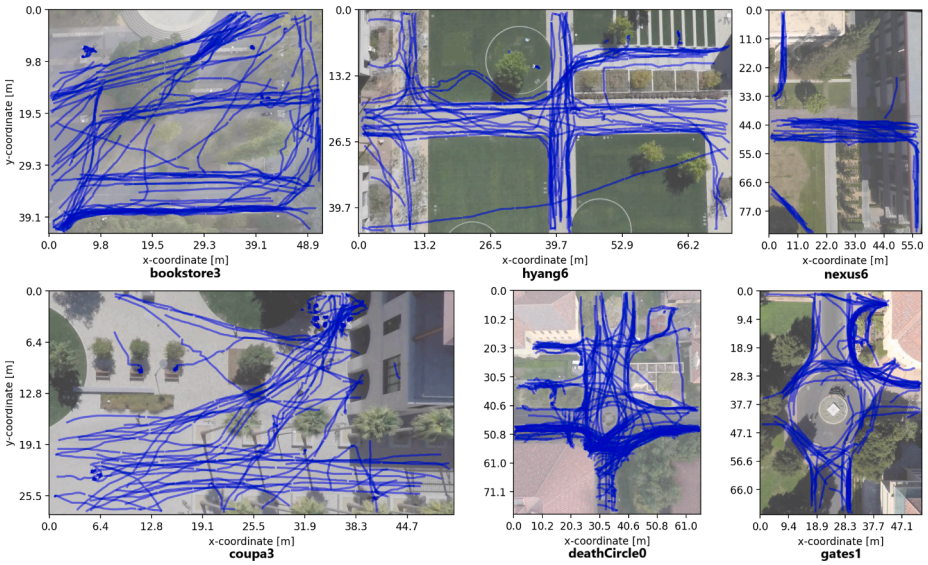


Fig. 4. Visualization of each scene of the offline test set.

2017; Xue et al., 2018; Hasan et al., 2018; Cheng and Sester, 2018; Cheng and Sester, 2018; Johora et al., 2020), which uses a binary tensor to map the relative positions of the neighboring agents of the target agent (Alahi et al., 2016). The so-called dynamic maps extend this method, in which the interactions at each step are modeled via three layers dedicated for orientation, speed and position information. Each map changes from one step to the next and therefore the spatio-temporal interaction information between agents is interpreted dynamically over time.

The map is defined as a rectangular area around the target agent, and is divided into grid cells and centralized on the agent's current location, see Fig. 3. W and H denote the width and height. First, referred to the target agent i, the neighboring agents $\mathsf{N}(i)$ are mapped into the closest grid $\text{cells}^t_{w\times h}$ according to their relative position and they are also mapped onto the cells reached by their anticipated relative offset (speed) in the x and y directions.

$$\begin{aligned} \text{cells}^t_w &= x^t_j - x^t_i + \left(\Delta x^t_j - \Delta x^t_i\right), \\ \text{cells}^t_h &= y^t_j - y^t_i + \left(\Delta y^t_j - \Delta y^t_i\right), \\ \text{where } & w \leqslant W, h \leqslant H, j \in \mathsf{N}(i) \text{and} j \neq i. \end{aligned} \tag{7}$$

Second, the orientation, speed and position information is stored in the mapped cells in the respective layer for each neighboring agent. The *orientation* layer O stores the heading direction. For the neighboring agent j, its orientation from the current to the next position is the angle ϑ_j in the Euclidean plane and calculated in the given radians by $\vartheta_j = \arctan2(\Delta y^t_j, \Delta x^t_j)$. Its value is shifted into the interval $[0°, 360°)$. 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.

The map covers a large vicinity area. Empirically we found $32 \times 32\,\text{m}^2$ a proper setting considering both the coverage and the computational cost. There is a trade-off between a high and a low resolution map. A cell is filled by a maximum of one agent if its size is small. But the high resolution will lead to a very sparse map (most of the cells have no value) and the surrounding areas of the neighboring agent will be treated as having no impact on the target agent. On the other hand, there may exist an overlap of multiple agents in one cell with a very different travel speed or orientation if the cell size is too big. In this work , we resolve this problem by setting the cell size as $1 \times 1\,\text{m}^2$. Based on the distribution of the experimental data, there are only a few cells with overlapped agents, which is also supported by the preservation of personal space (Gérin-Lajoie et al., 2005). However, the information of agents' size is not given by the experimental data, the approximation of the cell size may not be valid for large agents. In future work, the size of the agents will be considered and an extended margin will be applied to avoid the problem of agents falling out of the grid cell bounds.

Interactions among different agents are dynamic in various situations from one step to another in a sequence. Some steps may impact the agents' behaviors more than other steps. To explore such varying information, we employ a self-attention mechanism (Vaswani et al., 2017) to learn the interaction information from the dynamic maps over time and call them *attentive dynamic maps*. The self-attention module takes as input the dynamic maps and attentively learns the interconnections over the steps. The detailed information of this module is given in Appendix B.

4. Experiments

In this section, the evaluation metrics, benchmarks, experimental settings and the recent state-of-the-art methods for comparison are introduced for evaluating the proposed model. The ablation studies that partially remove the modules in the proposed method are conducted to justify each module's contribution. Finally, the experimental results are analyzed and discussed in detail.

4.1. Evaluation metrics

The mean average displacement error (ADE) and final displacement error (FDE) are the most commonly applied metrics to measure the performance of trajectory prediction (Alahi et al., 2016; Gupta et al., 2018; Sadeghian et al., 2019). ADE is the aligned Euclidean distance from Y (ground truth) to its prediction $\widehat{Y}$ averaged over all steps. FDE is the Euclidean distance of the last position from Y to the corresponding $\widehat{Y}$. It measures a model's ability for predicting the destination and is more challenging as errors accumulate with time. We report the mean values for all the trajectories.

We evaluate both the most-likely prediction and the best prediction @*top*10 for the multi-path trajectory prediction. The most-likely prediction is selected by the trajectories ranking as described in Section 3.2. @*top*10 prediction is the best one out of ten predicted trajectories that has the smallest ADE and FDE compared with the ground truth. When

the ground truth is not available (for the online test), only the most-likely prediction is selected. Then it becomes to the single trajectory prediction problem, as most of the previous works did (Helbing and Molnar, 1995; Alahi et al., 2016; Zhang et al., 2019; Becker et al., 2018; Hasan et al., 2018; Giuliari et al., 2020).

4.2. Trajnet benchmark challenge

We first verify the performance of the proposed method on Trajnet (Sadeghian et al., 2018). It is the most popular large-scale trajectory-based activity benchmark in this domain and provides a uniform evaluation system for fair comparison among different submitted methods. A wide range of datasets (e.g., ETH Pellegrini et al., 2009; Lerner et al., 2007 and Stanford Drone Dataset Robicquet et al., 2016) for heterogeneous agents (pedestrians, bikers, skateboarders, cars, buses, and golf cars) that navigate in real-world outdoor mixed traffic environments are included. The data was collected from 38 scenes with ground truth for training and another 20 scenes without ground truth for testing (i.e., open challenge competition). Each scene presents various traffic densities in different space layouts, which makes the prediction task challenging and requires a model to generalize, in order to adapt to the various complex scenes. Trajectories are provided as the *xy* coordinates in meters (or pixels) projected on a Cartesian space, with 8 steps for observation and the following 12 steps for prediction. The duration between two successive steps is 0.4 s. We follow all the previous works (Helbing and Molnar, 1995; Alahi et al., 2016; Zhang et al., 2019; Becker et al., 2018; Hasan et al., 2018; Gupta et al., 2018; Giuliari et al., 2020) that use the coordinates in meters.

In order to train and evaluate the proposed method, as well as the ablation studies, 6 of the total 38 scenes in the training set are selected as the offline test set. Namely, they are *bookstore3*, *coupa3*, *deathCircle0*, *gates1*, *hyang6*, and *nexus0*. The selection of the scenes is based on the space layout, data density and percentage of non-linear trajectories, see Table 2. Fig. 4 visualizes the trajectories in each scene. The trained model that has the best performance on the offline test set is selected as our final model and used for the online testing.

4.3. Quantitative results and comparison

We compare the performance of our model with the most influential previous works and the recent state-of-the-art works published on the Trajnet challenge.

- *Social Force* (Helbing and Molnar, 1995) is a rule-based model with the repulsive force for collision avoidance and the attractive force for social connections;
- *Social LSTM* (Alahi et al., 2016) proposes Social pooling with a rectangular occupancy grid for close neighboring agents, which is widely adopted in this domain (Lee et al., 2017; Xue et al., 2018; Hasan et al., 2018; Cheng and Sester, 2018; Cheng and Sester, 2018; Johora et al., 2020);
- *SR-LSTM* (Zhang et al., 2019) uses a states refinement module for extracting social effects between the target agent and its neighboring agents;
- *RED* (Becker et al., 2018) uses RNN-based Encoder with Multilayer Perceptron (MLP) for trajectory prediction;
- *MX-LSTM* (Hasan et al., 2018) exploits the head pose information of agents to help analyze its moving intention;
- *Social GAN* (Gupta et al., 2018) proposes to utilize GAN for multi-path trajectory prediction, which is the one of the closest works to our work; the other one is DESIRE (Lee et al., 2017). But neither the online test nor code was reported. Hence, we do not compare with DESIRE;
- *Ind-TF* (Giuliari et al., 2020) proposes a novel idea that utilizes the Transformer network (Vaswani et al., 2017) for sequence prediction. No social interactions between agents are considered in this work.

Table 1
Comparison between our method and the state-of-the-art models. Smaller values indicate a better performance and best values are highlighted in boldface.

Model	Avg. [m]↓	FDE [m]↓	ADE [m]↓
Social LSTM (Alahi et al., 2016)	1.3865	3.098	0.675
Social GAN (Gupta et al., 2018)	1.334	2.107	0.561
MX-LSTM (Hasan et al., 2018)	0.8865	1.374	0.399
Social Force (Helbing and Molnar, 1995)	0.8185	1.266	0.371
SR-LSTM (Zhang et al., 2019)	0.8155	1.261	0.370
RED (Becker et al., 2018)	0.78	1.201	0.359
Ind-TF (Giuliari et al., 2020)	0.7765	1.197	**0.356**
This work (AMENet)*	**0.7695**	**1.183**	**0.356**

Table 2
Evaluation measured by ADE/FDE for multi-path trajectory prediction using AMENet on the Trajnet offline test set.

Dataset	Layout	#Trajs	Non-linear traj rate	@top10	Most-likely
bookstore3	Parking	429	0.71	0.477/ 0.961	0.486/ 0.979
coupa3	Corridor	639	0.31	0.221/ 0.432	0.226/ 0.442
deathCircle0	Roundabout	648	0.89	0.650/ 1.280	0.659/ 1.297
gates1	Roundabout	268	0.87	0.784/ 1.663	0.797/ 1.692
hyang6	Intersection	327	0.79	0.534/ 1.076	0.542/ 1.094
nexus6	Corridor	131	0.88	0.542/ 1.073	0.559/ 1.109
Avg.	–	407	0.74	0.535/ 1.081	0.545/ 1.102

The performances of single trajectory prediction from different methods on the Trajnet challenge are given in Table 1. The results were originally reported on the leader board[2] up to the date of 14 June 2020. AMENet outperformed the other models and won the first place measured by the aforementioned metrics. Compared with the most recent model Ind-TF (Giuliari et al., 2020), AMENet achieved comparative performance in ADE and slightly better in FDE (from 1.197 to 1.183 meters). The superior performance given by AMENet here also validates the efficacy of the ranking method to select the most-likely prediction from the multiple predicted trajectories, as introduced in Section 3.2.

4.4. Results for multi-path prediction

The performance for multi-path prediction is investigated using the offline test set. Table 2 shows the quantitative results. Compared to the most-likely prediction, as expected the @top10 prediction yields similar but slightly better performance. It indicates that: (1) the generated multiple trajectories increase the chance to narrow down the errors; (2) the ranking method is effective for ordering the multiple predictions and proposing a good one, which is especially important when the prior knowledge of the ground truth is not available.

Fig. 5 showcases some qualitative examples of the multi-path trajectory prediction by AMENet. As shown in the roundabout deathCircle0 and gates1, each moving agent has more than one possibility (different speeds and orientations) to choose its future path. The predicted trajectories diverge more widely in further steps as the uncertainty about an agent's intention increases with time. Predicting multiple plausible trajectories indicates a larger intended area and raises the chance to cover the path an agent might choose in the future. Also, the "fan" of possible trajectories can be interpreted as reflecting the uncertainty of

[2] http://trajnet.stanford.edu/result.php?cid=1

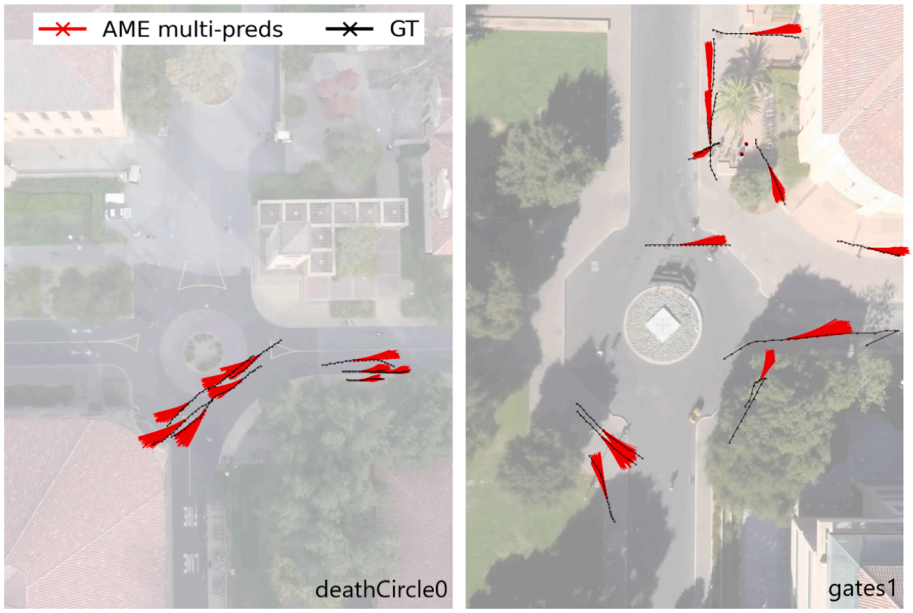


Fig. 5. Multi-path predictions from AMENet.

Table 3
Evaluation results measured by ADE/FDE on the most-likely prediction for the ablative models and the proposed model AMENet. Best values are highlighted in boldface.

Scene	ENet	OENet	AOENet	MENet	ACVAE	AMENet
B	0.532/ 1.080	0.601/ 1.166	0.574/ 1.144	0.576/ 1.139	0.509/ 1.030	**0.486/ 0.979**
C	0.241/ 0.474	0.342/ 0.656	0.260/ 0.509	0.294/ 0.572	0.237/ 0.464	**0.226/ 0.442**
D	0.681/ 1.353	0.741/ 1.429	0.726/ 1.437	0.725/ 1.419	0.698/ 1.378	**0.659/ 1.297**
G	0.876/ 1.848	0.938/ 1.921	0.878/ 1.819	0.941/ 1.928	0.861/ 1.823	**0.797/ 1.692**
H	0.598/ 1.202	0.661/ 1.296	0.619/ 1.244	0.657/ 1.292	0.566/ 1.140	**0.542/ 1.094**
N	0.684/ 1.387	0.695/ 1.314	0.752/ 1.489	0.705/ 1.346	0.595/ 1.181	**0.559/ 1.109**
Avg.	0.602/ 1.224	0.663/ 1.297	0.635/ 1.274	0.650/ 1.283	0.577/ 1.170	**0.545/ 1.102**

the prediction. Conversely, a single prediction provides limited information for inference and is likely to lead to a false conclusion if the prediction is not correct/precise in the early steps. On the other hand, agents that stand still were correctly predicted by AMENet with high certainty, as shown by two agents in gates1 in the upper right area. As designed by the model, only interactions between agents lead to adaptions in the predicted path and deviation from linear paths; the scene context, e.g., road geometry, is not modeled and thus does not affect prediction.

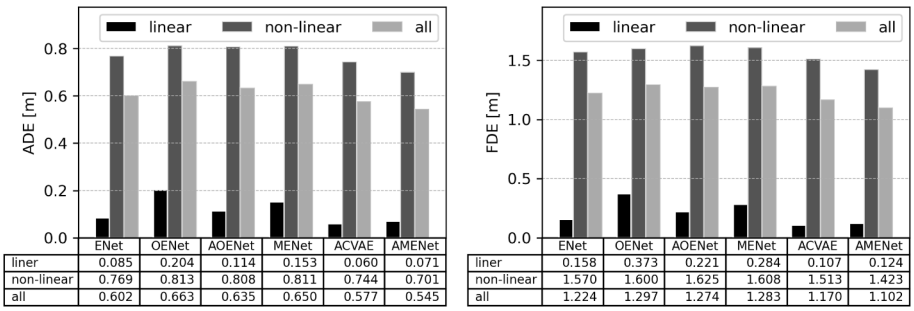
4.5. Ablation study

In order to analyze the impact of each module in the proposed framework, i.e., dynamic maps, self-attention, and the extended structure of the CVAE, several ablative models were investigated.

- ENet: (E) ncoder (Net) work, which is only conditioned on the motion information. The interaction information is not leveraged. This model is treated as the baseline model.
- OENet: (O) ccupancy+ENet, where interactions are modeled by the occupancy grid (Alahi et al., 2016; Lee et al., 2017; Xue et al., 2018; Hasan et al., 2018; Cheng and Sester, 2018; Cheng and Sester, 2018; Johora et al., 2020) in both the X-Encoder and the Y-Encoder.
- AOENet: (A) ttention+OENet, where the self-attention mechanism is added.
- MENet: (M) aps+ENet, where interactions are modeled by the proposed dynamic maps in both the X-Encoder and the Y-Encoder.
- ACVAE: (A) ttention + CVAE, where the dynamic maps are only added in the X-Encoder. It is equivalent to a CVAE model (Kingma and Welling, 2014; Kingma et al., 2014; Sohn et al., 2015) with the self-attention mechanism.
- AMENet: (A) ttention+MENet, where the self-attention mechanism is added. It is the full model of the proposed framework.

Table 3 shows the quantitative results for the ablation studies. Errors are measured by ADE/FDE on the most-likely prediction. The comparison between OENet and the baseline model ENet shows that extracting the interaction information from the occupancy grid did not contribute to a better performance. Even though the self-attention mechanism was added to the occupancy grid (denoted by AOENet), the slightly enhanced performance still fell behind the baseline model. The comparison indicates that interactions were not effectively learned from the occupancy map with or without the self-attention mechanism across the datasets. The comparison between MENet and ENet shows a similar pattern. The performance was slightly less inferior using the dynamic maps than the occupancy grid (MENet vs. OENet) in comparison to the baseline model. However, profound improvements can be seen after employing the self-attention mechanism. First, the comparison between ACVAE and ENet shows that even without the extended structure in the Y-Encoder, the dynamic maps with the self-attention mechanism in the X-Encoder were very beneficial for modeling interactions. On average, the performance was improved by 4.0 % and 4.5 % as measured by ADE and FDE, respectively. Second, the comparison between the proposed model AMENet and ENet shows that after extending the dynamic maps to the Y-Encoder, the errors, especially the absolute values of FDE, further decreased across all the datasets; ADE was reduced by 9.5 % and FDE was reduced by 10.0 %. This improvement was also confirmed by the benchmark challenge (see Table 1).

The evaluation was decomposed for non-linear and linear trajectories across all of the above models. The linearity of a trajectory not only depends on the continuity of the travel direction, but also on the speed. We use the same scheme as (Gupta et al., 2018) to categorize the



ADE	ENet	OENet	AOENet	MENet	ACVAE	AMENet
liner	0.085	0.204	0.114	0.153	0.060	0.071
non-linear	0.769	0.813	0.808	0.811	0.744	0.701
all	0.602	0.663	0.635	0.650	0.577	0.545

FDE	ENet	OENet	AOENet	MENet	ACVAE	AMENet
liner	0.158	0.373	0.221	0.284	0.107	0.124
non-linear	1.570	1.600	1.625	1.608	1.513	1.423
all	1.224	1.297	1.274	1.283	1.170	1.102

(a) ADE (b) FDE

Fig. 6. The prediction errors for linear, non-linear and all trajectories measured by (a) ADE and (b) FDE for all the ablative models, as well as the proposed model AMENet.

linearity of trajectories by a two-degree polynomial fitting. It compares the sum of the squared residuals over the fitting with the least-squares error. A trajectory is categorized as linear if it meets the criteria. Fig. 6 visualizes the values of (a) ADE and (b) FDE averaged over the six scenes in the offline test set. Across the models, the performance for predicting non-linear trajectories demonstrates a similar pattern compared to predicting all the trajectoires (linear + non-linear) and AMENet outperformed the other models measured by both metrics. Obviously, predicting the linear trajectories is easier than the non-linear ones. In this regard, all the models performed very well (ADE $\leqslant$0.2 m and FDE $\leqslant$0.4 m), especially the AMENet and ACVAE models. This observation indicates that if there are other agents interacting with each other, the continuity of their motion is likely to be interrupted, i.e., deviating from the free-flow trajectories (Rinke et al., 2017). The model has to adapt to this deviation to achieve a good performance. On the other hand, if there is no such reason to disrupt the linearity of the motion, then the model does not generate deviated trajectories.

Fig. 7 showcases some qualitative results by the proposed AMENet model in comparison to the ablative models. In general, AMENet generated accurate predictions and outperformed the other models in all the scenes, which is especially visible in coupa3 (a) and bookstore3. All the models predicted plausible trajectories for two agents walking in parallel in coupa3 (b) (denoted by the black box), except the baseline model ENet. Without modeling interactions, the ENet model generated two trajectories that intersected with each other. In hyang6 limited performance can be seen by ENet, AOENet and MENet regarding travel speed and OENet and ACVAE regarding destination for the fast-moving agent. In contrast AMENet kept a good prediction. In nexus6 (a) and (b), only two agents were present, where all the models performed well. More agents were involved in the roundabout scenes, in which the prediction task was more challenging. AMENet generated accurate predictions for most of the agents. However, its performance is limited for the agents that changed speed or direction rapidly from their past movement. We notice one interesting scenario of the two agents that walked towards each other in deathCircle0 (denoted by the black box). In reality, when the right agent changed its heading towards the left agent, the left agent had to decelerate strongly to yield the way. Regarding the interaction and compared with the other models, AMENet generated non-conflict trajectories.

4.6. Trajectory prediction on Benchmark InD

To further investigate the generalization performance of the proposed model, we carried out extensive experiments on a newly published large-scale benchmark InD[3]. It consists of 33 datasets and was collected using drones on four very busy intersections (as shown in Fig. 8) in Germany in 2019 by Bock et al. (2019). Different from Trajnet where most of the environments (i.e., shared spaces Reid, 2009; Robicquet et al., 2016) are pedestrian friendly, the intersections in InD are dominated by vehicles. This makes the prediction task more challenging due to the very different travel speed between pedestrians and vehicles, as well as the direct interactions. We follow the same format as the Trajnet benchmark for data processing (Section 4.2). The performance of AMENet is compared with Social LSTM (Alahi et al., 2016) and Social GAN (Gupta et al., 2018), which are the most relevant ones to our models. As mentioned above, Social LSTM (Alahi et al., 2016) is the first deep learning method that uses occupancy grid for modeling interactions between agents and Social GAN (Gupta et al., 2018) is the closest deep generative model to ours. It is worth mentioning that we trained and tested all the three models using the same data for fair comparison.

The performance is analyzed quantitatively and qualitatively. Table 4 lists the evaluation results measured by ADE/FDE for all the models in each intersection. AMENet predicted more accurate trajectories measured by all the metrics compared with Social LSTM and Social GAN. Fig. 8 shows one scenario in each of the four intersections. AMENet predicted the deceleration of the car approaching the intersection from the right arm, the trajectory of the cross walking pedestrian and the slowing down of the pedestrian on the sidewalk in intersection A. It correctly predicted two cars slowly approaching the intersection area in intersection B and D, and the waiting scenario for pedestrian cross walking in intersection C.

4.7. Discussion of the results

Based on the extensive studies and results, we discuss the advantages and limitations of the AMENet model proposed in this paper.

AMENet demonstrated superior performance over different benchmarks for trajectory prediction. Firstly, the proposed model was able to achieve the state-of-the-art performance on the Trajnet benchmark challenge, which contains various scenes. Secondly, the results of the ablation studies proved that the information of interactions between agents is beneficial for trajectory prediction. However, the performance highly depends on how such information was leveraged. It was difficult for the occupancy grid, which is only based on the positions of the neighboring users, to extract useful information for interaction modeling, because positions change from one step to the next and from one scene to another. Meanwhile, the speed and interaction information

[3] https://www.ind-dataset.com/

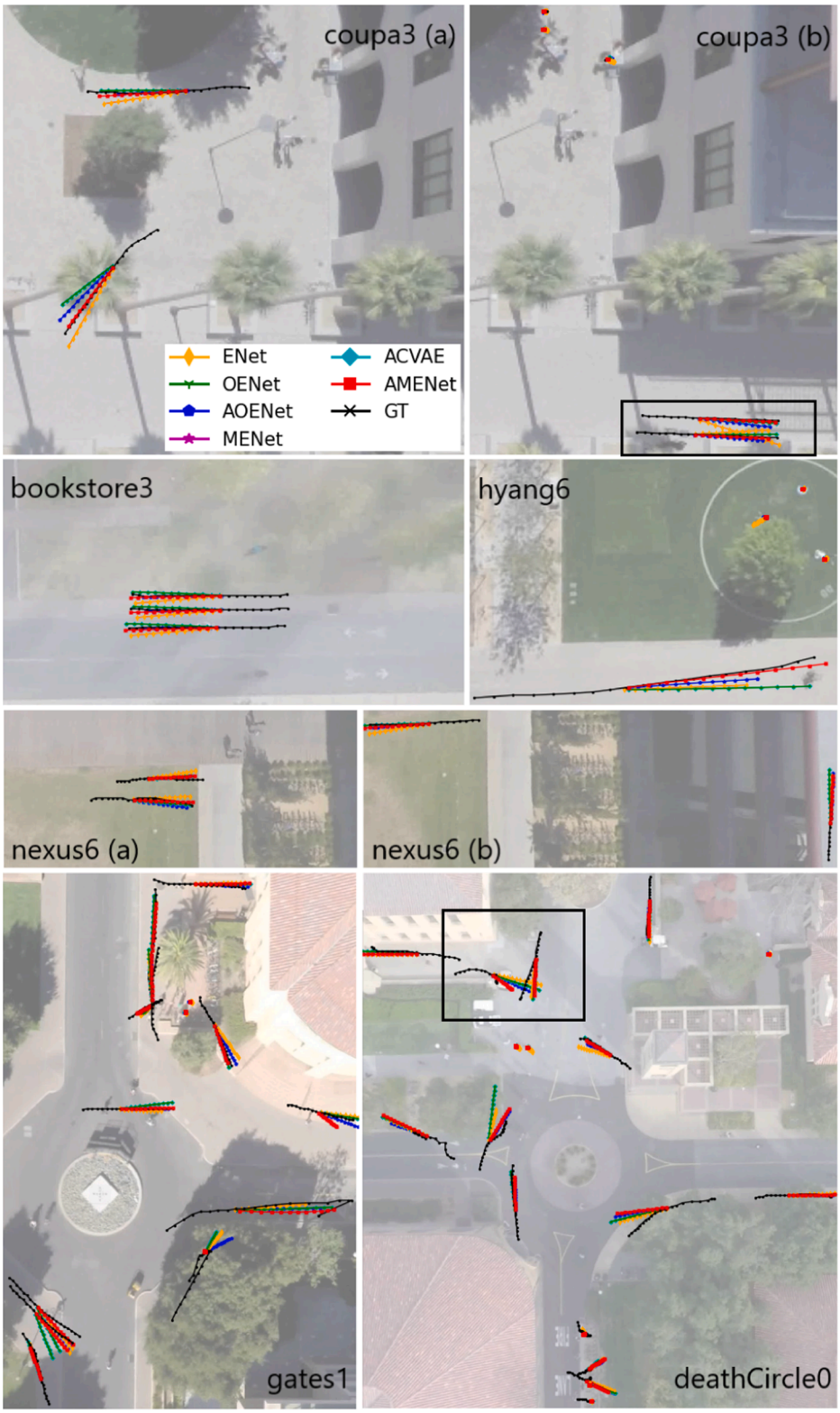


Fig. 7. Trajectories predicted by ENet, OENet, AOENet, MENet, ACVAE and AMENet in comparison with the ground truth (GT) trajectories on Trajnet (Sadeghian et al., 2018).

Fig. 8. Trajectories predicted by AMENet on the InD benchmark (Bock et al. (2019)).

Table 4
Quantitative results of AMENet and the comparative modles on InD (Bock et al., 2019) measured by ADE/FDE. Best values are highlighted in bold face.

Model	S-LSTM	S-GAN	AMENET	S-LSTM	S-GAN	AMENET
InD		@top 10			Most-likely	
Int. A	2.04/4.61	2.84/4.91	**0.95/1.94**	2.29/5.33	3.02/5.30	**1.07/2.22**
Int. B	1.21/2.99	1.47/3.04	**0.59/1.29**	1.28/3.19	1.55/3.23	**0.65/1.46**
Int. C	1.66/3.89	2.05/4.04	**0.74/1.64**	1.78/4.24	2.22/4.45	**0.83/1.87**
Int. D	2.04/4.80	2.52/5.15	**0.28/0.60**	2.17/5.11	2.71/5.64	**0.37/0.80**
Avg.	1.74/4.07	2.22/4.29	**0.64/1.37**	1.88/4.47	2.38/4.66	**0.73/1.59**

is not considered, which may explain why the occupancy grid performed worse than the dynamic maps in the same settings. Thirdly, as interactions change over time, the self-attention mechanism automatically extracted the salient features in the time axis from the dynamic maps.

However, there are several limitations of the model being uncovered throughout the experiments. First, the resolution of the map was approximated according to the experimental data and the size of the neighboring agents was not yet considered. This may limit the model for dealing with big-sized agents, such as buses or trucks. We leave this to future work. Second, from the qualitative results we notice that the model had limited performance for predicting the behavior of the agents that drastically change direction and speed, which is in general a very challenging task without extra information from the agents, such as body posture or eye gaze. Last but not least, in this work, scene context information was not included. The lack of this information may lead to a wrong prediction, e. g., trajectories leading into obstacles or inaccessible areas. Scene context can have a positive effect that a trajectory follows a (curved) path. On the other hand, a strong constraint from the scene context can easily overfit a model for some particular scene layout (Cheng et al., 2020). Hence, a good mechanism for parsing the scene information is needed to balance the trade-off, especially for a model trained in one scene and applied in another.

5. Conclusions

In this paper, we have presented a generative model called Attentive Maps Encoder Network (AMENet) for multi-path trajectory prediction and made the following contributions. (1) The model captures the stochastic properties of road users' motion behaviors after a short observation time via the latent space learned by the X-Encoder and Y-Encoder that encode motion and interaction information, and predicts multiple plausible trajectories. (2) We propose a novel concept–attentive 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 steps. (3) The model targets heterogeneous agents in mixed traffic in various real-world traffic environments. The efficacy of the model was validated on the benchmark Trajnet that contains various datasets in different real-world environments and the InD benchmark for different intersections. The model not only achieved state-of-the-art performance, but also won the first place on the leader board for predicting 12 time-step positions of 4.8 s. Each component of AMENet has been validated via a series of ablation studies.

In future work, we plan to include more information to further improve the prediction accuracy, such as the type and size information of agents and spatial context information. In addition, we will extend our trajectory prediction model for safety analysis, e. g., using the predicted trajectories to calculate time-to-collision (Perkins and Harris, 1968) and to detect abnormal trajectories by comparing the anticipated/predicted trajectories with the actual ones.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

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

Appendix A. Conditional Variational Auto-Encoder

The Conditional Variational Auto-Encoder (CVAE) model is built upon the variational inference and the learning of directed graphical models (Kingma and Welling, 2014; Rezende et al., 2014; Kingma et al., 2014), as denoted by Fig. A.1. CVAE is an extension of the Variational Auto-Encoder (VAE) (Kingma and Welling, 2014). For understanding the solution of the CVAE model, it is necessary to revisit the variational inference and the VAE model.

The VAE assumes that the dataset, $\mathbf{X} = \{\mathrm{X}_1, \cdots, \mathrm{X}_N\}$, contains N independent and identically distributed samples of some continuous or discrete variable X. The dataset can be generated from some unobserved random variables $\mathbf{z}$, the so-called latent variables (Kingma and Welling, 2014). Eq. (A.1) denotes the integral of the marginal likelihood, where $p_\theta(\mathbf{z})$ is the prior distribution of the latent variables and θ are the generative parameters.

$$p_\theta\left(\mathbf{X}\right) = \int p_\theta\left(\mathbf{X}\middle|\mathbf{z}\right) p_\theta\left(\mathbf{z}\right) d\mathbf{z}. \tag{A.1}$$

However, the equation cannot be solved analytically due to the intractable posterior $p_\theta(\mathbf{z}|\mathbf{X}) = p_\theta(\mathbf{X}|\mathbf{z})p_\theta(\mathbf{z})/p_\theta(\mathbf{X})$, or efficiently due to the expensive sampling over a large dataset. To solve the problem, a variational approximation of the true posterior is introduced as the recognition model $q_\phi(\mathbf{z}|\mathbf{X})$, where ϕ is the variational parameter. Then Eq. (A.1) can be rewritten as:

$$\log p_\theta\left(\mathrm{X}_1, \cdots, \mathrm{X}_N\right) = \sum_{i=1}^{N} \log p_\theta\left(\mathrm{X}_i\right), \tag{A.2}$$

$$\log p_\theta(\mathrm{X}_i) = D_{KL}(q_\phi(\mathbf{z}|\mathrm{X}_i)\|p_\theta(\mathbf{z}|\mathrm{X}_i)) + \mathcal{L}(\theta, \phi; \mathrm{X}_i). \tag{A.3}$$

It summarizes over the marginal likelihoods of individual data points. The Kullback–Leibler divergence $D_{KL}(q_\phi(\mathbf{z}|\mathrm{X}_i)\|p_\theta(\mathbf{z}|\mathrm{X}_i))$ measures the error of the approximation to the true posterior. Note that the Kullback–Leibler divergence is always non-negative. Hence,

$$\log p_\theta(\mathrm{X}_i) \geqslant \mathcal{L}(\theta, \phi; \mathrm{X}_i), \tag{A.4}$$

which is called the (variational) *lower bound* on the marginal likelihood of the data point i. With the variational approximation, Bayes' theorem is applied to solve $\mathcal{L}(\theta, \phi; \mathrm{X}_i)$ as follows:

$$\begin{aligned} \mathcal{L}(\theta, \phi; \mathrm{X}_i) &= \log \mathbb{E}_{q_\phi(\mathbf{z}|\mathrm{X}_i)} \frac{p_\theta(\mathrm{X}_i, \mathbf{z})}{q_\phi(\mathbf{z}|\mathrm{X}_i)} \\ &= -D_{KL}\left(q_\phi\left(\mathbf{z}|\mathrm{X}_i\right)\|p_\theta\left(\mathbf{z}\right)\right) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathrm{X}_i)}\left[\log p_\theta\left(\mathrm{X}_i|\mathbf{z}\right)\right], \end{aligned} \tag{A.5}$$

where $-D_{KL}(\cdot)$ is the negative Kullback–Leibler divergence of the approximate posterior from the prior $p_\theta(\mathbf{z})$ and acts as a regularizer. $\mathbb{E}_{q_\phi(\mathbf{z}|\mathrm{X}_i)}(\cdot)$ is an expected negative reconstruction loss. When one optimizes the log likelihood on the left side of Eq. (A.4), the Kullback–Leibler divergence and the reconstruction loss are jointly minimized. Hence, the recognition model parameters ϕ and the generative parameters θ can be learned jointly. The structure of an "Auto-Encoder" framework becomes intuitive in Eq. (A.5): the recognition model $q_\phi(\mathbf{z}|\mathrm{X}_i)$ *encodes* the input into the latent variables and the generative model $\log p_\theta(\mathrm{X}_i|\mathbf{z})$ *decodes* the output from the latent variables. An analytical solution for the Kullback–Leibler divergence of two distributions can be found in the Gaussian case (Kingma and Welling, 2014). The reconstruction error requires sampling, e. g., Monte Carlo sampling. The lower bound estimation of the VAE model is solved as:

$$\mathcal{L}\left(\theta, \phi; \mathrm{X}_i\right) = \frac{1}{2}\sum_{j=1}^{J}\left(1 + \log\sigma_j^2 - \mu_j^2 - \sigma_j^2\right) + \frac{1}{L}\sum_{l=1}^{L}\left(\log p_\theta\left(\mathrm{X}_i\middle|\mathbf{z}_i^l\right)\right). \tag{A.6}$$

There is one remaining problem of the sampling process. Neural networks can be used to parameterize the mapping of θ and ϕ, which works in the forward pass in the VAE model. However, there is no gradient of the sampling when the neural networks have to be optimized via gradient descent in the backpropogation. To solve this problem, a *re-parameterization* trick (Rezende et al., 2014) is introduced to mimic the stochastic property of the latent variables drawn from $g_\phi(,)$ while maintaining the gradients for backpropogation at the same time.

$$\mathbf{z}_i^l = g_\phi\left(\epsilon_i^l, \mathrm{X}_i^l\right), \text{where} \mathbf{z}_i^l \sim q_\phi\left(\mathbf{z}|\mathrm{X}_i^l\right), \tag{A.7}$$

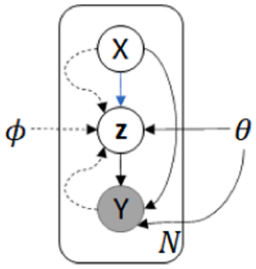


Fig. A.1. Conditional Variational Auto-Encoder graphical model. The generative model $p_\theta(Y|X, \mathbf{z})p_\theta(\mathbf{z})$ is denoted by black solid lines, where $p_\theta(\mathbf{z})$ is the prior of the latent variables. The prior is made independent from the input variables such that $p_\theta(\mathbf{z}|X) = p_\theta(\mathbf{z})$ (Kingma et al., 2014), denoted by blue solid lines. The variational approximation $q_\phi(\mathbf{z}|Y, X)$ to the intractable posterior of $p_\theta(\mathbf{z}|Y, X)$ is denoted .by dashed lines.

ϵ is a noise vector drawn from the distribution of $\mathcal{N}(0, \mathbf{I})$. Assuming the posterior approximation $q_\phi(\mathbf{z}|\mathrm{X}_i) = \mathcal{N}(\mu, \sigma^2)$, a valid function of $g_\phi(\epsilon_i^l, \mathrm{X}_i)$ can be formulated as:

$$\mathbf{z}_i^l = g_\phi\left(\epsilon_i^l, \mathrm{X}_i^l\right) = \mu_i^l + \sigma_i^l \odot \epsilon_i^l \tag{A.8}$$

The Conditional VAE (CVAE) is proposed by Sohn et al. (2015) for structured output prediction. Different from the VAE that reconstructs the input variables with a variational recognition of the posterior, the CVAE generates desirable outputs conditioned on the input variables. To be more specific, given the observation X, the latent variables $\mathbf{z}$ are drawn from $p_\theta(\mathbf{z}|\mathrm{X})$, the output Y is generated from $p_\theta(\mathrm{Y}|\mathrm{X}, \mathbf{z})$. A latent variable z can be drawn multiple times from the distribution $p_\theta(\mathbf{z}|\mathrm{X})$. The multi-sampling process of the latent variables $\mathbf{z}$ allows for modeling multiple modes in the conditional distribution of the output variables $\mathbf{Y}$, the so called one-to-many mapping. Then the integral of the conditional probability is defined as follows:

$$p_\theta\left(\mathbf{Y}\middle|\mathrm{X}\right) = \int p_\theta\left(\mathrm{Y}\middle|\mathrm{X}, \mathbf{z}\right) p_\theta\left(\mathbf{z}\right) d\mathbf{z}. \tag{A.9}$$

Note that in Eq. (A.9), the latent variables $\mathbf{z}$ can be made statistically independent of the input variables such that $p_\theta(\mathbf{z}|\mathrm{X}) = \sum_{\mathrm{X}} p_\theta(\mathbf{z}|\mathrm{X})p(\mathrm{X}) = p_\theta(\mathbf{z})$ (Kingma et al., 2014).

Similar to the VAE, the variational approximation estimation is used to solve Eq. (A.9) for the intractable posterior. At the datapoint i, the log likelihood is denoted by Eq. (A.10), where $q_\phi(\mathbf{z}|\mathrm{Y}_i, \mathrm{X}_i)$ is the variational approximation of the true posterior $p_\theta(\mathbf{z}|\mathrm{Y}_i, \mathrm{X}_i)$.

$$\log p_\theta(\mathrm{Y}_i|\mathrm{X}_i) = D_{KL}(q_\phi(\mathbf{z}|\mathrm{Y}_i, \mathrm{X}_i)||p_\theta(\mathbf{z}|\mathrm{Y}_i, \mathrm{X}_i)) + \mathcal{L}(\theta, \phi; \mathrm{Y}_i, \mathrm{X}_i). \tag{A.10}$$

The lower bound of the log likelihood at the datapoint i is solved analogously to the VAE mentioned above:

$$\log p_\theta(\mathrm{Y}_i|\mathrm{X}_i) \tag{A.11}$$

$$\geqslant \mathcal{L}(\theta, \phi; \mathrm{Y}_i, \mathrm{X}_i), \tag{A.12}$$

$$= -D_{KL}\left(q_\phi\left(\mathbf{z}|\mathrm{X}_i, \mathrm{Y}_i\right)\middle\|p_\theta\left(\mathbf{z}\right)\right) + \mathbb{E}_{q_\phi(\mathbf{z}|\mathrm{X}_i, \mathrm{Y}_i)}\left[\log p_\theta\left(\mathrm{Y}_i|\mathrm{X}_i, \mathbf{z}\right)\right], \tag{A.13}$$

$$\simeq -D_{KL}\left(q_\phi\left(\mathbf{z}\middle|\mathrm{X}_i, \mathrm{Y}_i\right)\middle\|p_\theta\left(\mathbf{z}\right)\right) + \frac{1}{L}\sum_{l=1}^{L}\left(\log p_\theta\left(\mathrm{Y}_i\middle|\mathrm{X}_i, \mathbf{z}_i^l\right)\right). \tag{A.14}$$

Appendix B. The self-attention mechanism

The self-attention module (Vaswani et al., 2017) is trained to assign a weight to the dynamic information of each step (denoted as Values (V)) based on how much the latent state of the current step (denoted as Query (Q)) matches the latent states of the other steps (denote as Key (K)). The latent states of Q, K and V are computed via three learnable linear transformations with the same input separately:

$$\begin{aligned} Q &= \pi(\mathrm{Map})W_Q, W_Q \in \mathbb{R}^{D\times d_q}, \\ K &= \pi(\mathrm{Map})W_K, W_K \in \mathbb{R}^{D\times d_k}, \\ V &= \pi(\mathrm{Map})W_V, W_A \in \mathbb{R}^{D\times d_v}, \end{aligned} \tag{A.15}$$

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 respective dimensionalities of the vector Q, K and V, which are all set to 4 in implementation. The attention module outputs a weighted sum of the values V, where the weight assigned to each value is determined by the dot-product of Q with K:

$$\mathrm{Attention}\left(Q, K, V\right) = \mathrm{softmax}\left(\frac{QK^{\mathbf{T}}}{\sqrt{d_k}}\right)V, \tag{A.16}$$

where $\sqrt{d_k}$ is the scaling factor, d_k is the dimensionality of the vector K and $\mathbf{T}$ stands for transpose. This operation is also called *scaled dot-product attention* (Vaswani et al., 2017).

To improve the performance of the self-attention module, the *multi-head attention* (Vaswani et al., 2017) strategy is applied as a conventional operation, where a head is an independent scaled dot-product attention module. In this way, the information is attended from different representation subspaces at different positions jointly:

$$\begin{aligned} \mathrm{MultiHead}(Q, K, V) &= \mathrm{ConCat}(\mathrm{head}_1, \ldots, \mathrm{head}_h)W_O, \\ \mathrm{head}_i &= \mathrm{Attention}\left(QW_{Qi}, KW_{Ki}, VW_{Vi}\right), \end{aligned} \tag{A.17}$$

where $W_{Qi}, W_{Ki}, W_{Vi} \in \mathbb{R}^{D\times d_{qi}}$ are the same linear transformation parameters as in (A.15) and W_O are the linear transformation parameters for aggregating the extracted information from different heads. Note that $d_{qi} = \frac{d_q}{h}$ must be an aliquot part of d_q. h is the total number of attention heads and we use two heads in the implementation.

Appendix C. AMENet model architecture and hyper-parameters

The detailed graph of AMENet is given in Fig. A.2. In Table A.1, we list the most important training hyper-parameters. The detailed settings can be found in our repository at https://github.com/haohao11/AMENet.

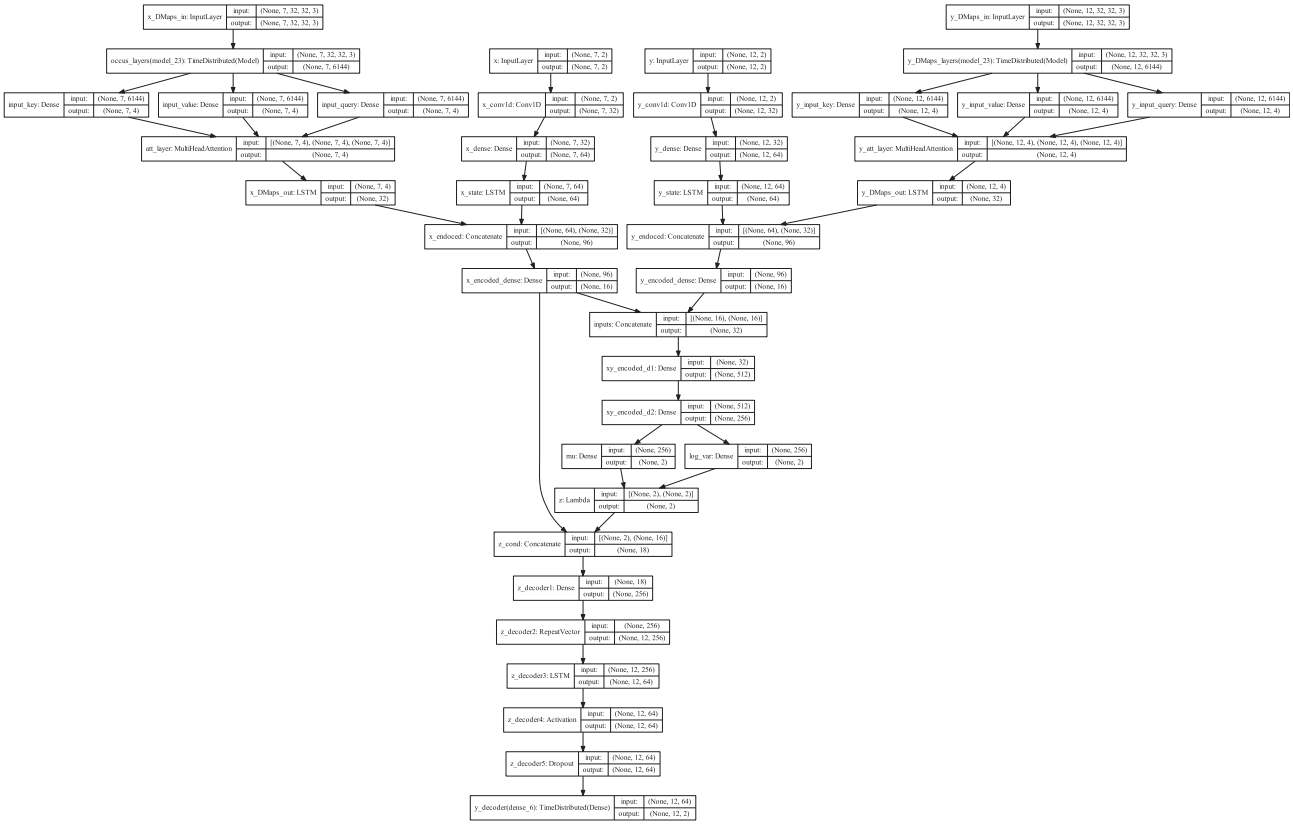


Fig. A.2. Detailed graph of the AMENet model architecture.

Table A.1
Training hyper-parameters for the AMENet model.

Name	Description	Values
z-dim	Size of the latent variable	2
Hidden-size	Size of the LSTM hidden state	32
Query	Query dimensionality for the self-attention layer	4
Key-value	Key&value dimensionality for the self-attention layer	4
h	Number of heads	2
beta	Rate of the re-construction loss	0.70 to 0.85
alpha	Rate of the KL-loss	1 −beta
lr	Learning rate of the Adam optimizer (Kingma et al., 2015)	0.001

References

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

Al-Molegi, A., Jabreel, M., Martinez-Balleste, A., 2018. Move, attend and predict: an attention-based neural model for people's movement prediction. Pattern Recogn. Lett. 112, 34–40.

Amirian, J., Hayet, J.-B., Pettré, J., 2019. Social ways: Learning multi-modal distributions of pedestrian trajectories with gans. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), IEEE, pp. 2964–2972..

Bahdanau, D., Cho, K., Bengio, Y., 2015. Neural machine translation by jointly learning to align and translate. In: ICLR.

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

Bock, J., Krajewski, R., Moers, T., Runde, S., Vater, L., Eckstein, L., 2019. The ind dataset: A drone dataset of naturalistic road user trajectories at german intersections..

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

Cheng, H., Sester, M., 2018. Modeling mixed traffic shared space using lstm with probability density mapping. In: ITSC, pp. 3898–3904.

Cheng, H., Sester, M., 2018. Mixed traffic trajectory prediction using lstm–based models shared space. In: The Annual International Conference on Geographic Information Science, pp. 309–325..

Cheng, H., Liao, W., Yang, M.Y., Sester, M., Rosenhahn, B., 2020. Mcenet: Multi-context encoder network for homogeneous agent trajectory prediction mixed traffic. In: ITSC.

Gérin-Lajoie, C.L., Richards, Martand, McFadyen, B.J., 2005. The negotiation of stationary and moving obstructions during walking: anticipatory locomotor adaptations and preservation of personal space. Motor Control 9 (3), 242–269.

Giuliari, F., Hasan, I., Cristani, M., Galasso, F., 2020. Transformer networks for trajectory forecasting. In: ICPR.

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

Graves, A., 2013. Generating sequences with recurrent neural networks, arXiv preprint arXiv:1308.0850..
Gupta, A., Johnson, L., Fei-Fei, Justand, Savarese, S., Alahi, A., 2018. Social gan: Socially acceptable trajectories with generative adversarial networks. In: CVPR, pp. 2255–2264.
Guzman-Rivera, A., Batra, D., Kohli, P., 2012. Multiple choice learning: learning to produce multiple structured outputs. In: NIPS, pp. 1799–1807.
Harvey, A.C., 1990. Forecasting, Structural Time Series Models and the Kalman Filter. Cambridge University Press.
Hasan, I., Setti, F., Tsesmelis, T., Del Bue, A., Galasso, F., Cristani, M., 2018. Mx-lstm: mixing tracklets and vislets to jointly forecast trajectories and head poses. In: CVPR, pp. 6067–6076.
Helbing, D., Molnar, P., 1995. Social force model for pedestrian dynamics. Phys. Rev. E 51 (5), 4282.
Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Comput. 9 (8), 1735–1780.
Johora, F.T., Cheng, H., Müller, J.P., Sester, M., 2020. An agent-based model for trajectory modelling shared spaces: a combination of expert-based and deep learning approaches, in. In: Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems, pp. 1878–1880.
Kingma, D.P., Ba, J., 2015. Adam: A method for stochastic optimization. In: ICLR.
Kingma, D.P., Welling, M., 2014. Auto-encoding variational bayes. In: ICLR.
Kingma, D.P., Mohamed, S., Rezende, D.J., Welling, M., 2014. Semi-supervised learning with deep generative models. In: NIPS, pp. 3581–3589.
Kitani, K.M., Ziebart, B.D., Bagnell, J.A., Hebert, M., 2012. Activity forecasting. In: ECCV, pp. 201–214.
Klinger, T., Rottensteiner, F., Heipke, C., 2017. Probabilistic multi-person localisation and tracking image sequences. ISPRS J. Photogramm. Remote Sens. 127, 73–88.
LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. Nature 521 (7553), 436.
Lee, N., Choi, W., Vernaza, P., Choy, C.B., Torr, P.H., Chandraker, M., 2017. Desire: Distant future prediction dynamic scenes with interacting agents. In: CVPR, pp. 336–345.
Lerner, A., Chrysanthou, Y., Lischinski, D., 2007. Crowds by example. In: Computer Graphics Forum, vol. 26, Wiley Online Library, pp. 655–664..
Ma, L., Liu, Y., Zhang, X., Ye, G., Yin, Yuanxand, Johnson, B.A., 2019. Deep learning remote sensing applications: a meta-analysis and review. ISPRS J. Photogramm. Remote Sens. 152, 166–177.
Makansi, O., Ilg, E., Cicek, O., Brox, T., 2019. Overcoming limitations of mixture density networks: a sampling and fitting framework for multimodal future prediction. In: CVPR, pp. 7144–7153.
Mohajerin, N., Rohani, M., 2019. Multi-step prediction of occupancy grid maps with recurrent neural networks. In: CVPR, pp. 10600–10608.
Mohanan, M., Salgoankar, A., 2018. A survey of robotic motion planning dynamic environments. Robot. Autonomous Syst. 100, 171–185.
Morris, B.T., Trivedi, M.M., 2008. A survey of vision-based trajectory learning and analysis for surveillance. Trans. Circuits Syst. Video Technol. 18 (8), 1114–1127.
Pellegrini, S., Ess, A., Schindler, K., Van Gool, L., 2009. You'll never walk alone: Modeling social behavior for multi-target tracking. In: ICCV, pp. 261–268..
Perkins, S.R., Harris, J.L., 1968. Traffic conflict characteristics-accident potential at intersections. Highway Res. Rec. (225)..
Reid, S., 2009. DfT Shared Space Project Stage 1: Appraisal of Shared Space. MVA Consultancy.
Rezende, D.J., Mohamed, S., Wierstra, D., 2014. Stochastic backpropagation and approximate inference deep generative models. In: ICML, pp. II-1278–II-1286.
Rinke, N., Schiermeyer, C., Pascucci, F., Berkhahn, V., Friedrich, B., 2017. A multi-layer social force approach to model interactions shared spaces using collision prediction. Transp. Res. Procedia 25, 1249–1267.
Robicquet, A., Sadeghian, A., Alahi, A., Savarese, S., 2016. Learning social etiquette: human trajectory understanding crowded scenes. In: ECCV, pp. 549–565.
Rudenko, A., Palmieri, L., Herman, M., Kitani, K.M., Gavrila, D.M., Arras, K.O., 2020. Human motion trajectory prediction: a survey. Int. J. Robot. Res. 39 (8), 895–935.
Sadeghian, A., Kosaraju, V., Gupta, A., Savarese, S., Alahi, A., 2018. Trajnet: Towards a benchmark for human trajectory prediction, arXiv preprint..
Sadeghian, A., Kosaraju, V., Sadeghian, A., Hirose, N., Savarese, S., 2019. Sophie: An attentive gan for predicting paths compliant to social and physical constraints. In: CVPR, pp. 1349–1358.
Schindler, K., Ess, A., Leibe, B., Van Gool, L., 2010. Automatic detection and tracking of pedestrians from a moving stereo rig. ISPRS J. Photogramm. Remote Sens. 65 (6), 523–537.
Sohn, K., Lee, H., Yan, X., 2015. Learning structured output representation using deep conditional generative models. In: NIPS, pp. 3483–3491.
Tang, C., Salakhutdinov, R.R., 2019. Multiple futures prediction. In: NIPS, pp. 15398–15408.
Tay, M.K.C., Laugier, C., 2008. Modelling smooth paths using gaussian processes. In: Field and Service Robotics, pp. 381–390..
Varshneya, D., Srinivasaraghavan, G., 2017. Human trajectory prediction using spatially aware deep attention models, arXiv preprint arXiv:1705.09436..
Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I., 2017. Attention is all you need. In: NIPS, pp. 5998–6008.
Vemula, A., Muelling, K., Oh, J., 2018. Social attention: modeling attention human crowds. In: ICRA, pp. 1–7.
Xue, H., Huynh, D.Q., Reynolds, M., 2018. Ss-lstm: A hierarchical lstm model for pedestrian trajectory prediction. In: WACV, pp. 1186–1194.
Xu, J., Kelvand Ba, Kiros, R., Cho, K., Courville, A., Salakhudinov, R., Zemel, R., Bengio, Y., 2015. Show, attend and tell: neural image caption generation with visual attention. In: ICML, pp. 2048–2057..
Zhang, P., Ouyang, W., Zhang, P., Xue, J., Zheng, N., 2019. Sr-lstm: State refinement for lstm towards pedestrian trajectory prediction. In: CVPR, pp. 12085–12094.