

Human trajectory forecasting using a flow-based generative model

Bo Zhang^a, Tao Wang^a, Changdong Zhou^a, Nicola Conci^b, Hongbo Liu^{a,*}

^a*College of Artificial Intelligence, Dalian Maritime University, No.1, Linghai Road, 116026, Dalian City, China.*

^b*Department of Information Engineering and Computer Science, University of Trento, No.5, Via Sommarive, 38123, Trento, Italy.*

Abstract

In this article, we present a flow-based framework for multi-modal trajectory prediction, which is able to provide an accurate and explicit inference of the latent representations on trajectory data. Differently from other typical generative models (such as GAN, VAE, etc.), the flow-based models aim at learning data distribution explicitly through an invertible network, which can convert a complicated distribution into a tractable form via invertible transformations. The whole framework is built upon the standard encoder-decoder architecture, where the LSTM is exploited as the fundamental block to capture the temporal structure of a trajectory. As a core module, we incorporate an invertible network that can learn the multi-modal distributions of trajectory data and further generate plausible future paths by sampling tricks from the standard Gaussian distribution. Extensive experiments carried out on synthetic and realistic datasets demonstrate the effectiveness of the proposed approach, and show the advantages as compared to the GAN-based and the VAE-based prediction frameworks.

Keywords: Path forecasting, Invertible networks, Flow-based generative models, Multi-modal prediction

*Corresponding author

Email addresses: bzhang@dlmu.edu.cn (Bo Zhang), wangtao_@dlmu.edu.cn (Tao Wang), changdong0901@dlmu.edu.cn (Changdong Zhou), nicola.conci@unitn.it (Nicola Conci), 1hb@dlmu.edu.cn (Hongbo Liu)

1. Introduction

The task of predicting future trajectories has drawn great attentions in the field of computer vision, which covers a wide range of applications, including visual surveillance, autonomous driving, navigation, and traffic dispersion, to name a few. A good predictive model does not only need proper long-term sequential modeling mechanisms for accurate prediction, but also an efficient way to learn the multi-modal characteristics of trajectory data, for the purpose of handling the problem of behavior uncertainty. With the rapid development of modern machine learning techniques, generative models such as generative adversarial networks (GAN) [1] and variational auto-encoders (VAE) [2] have shown great potentials in multi-modal data generation, and further been extended to deal with sequential modeling [3] [4] [5] [6]. The GAN-based models usually consist of a generator and a discriminator, where the former is used to generate new samples and the later is used to distinguish samples from real distribution and fake ones produced by the generator. Although GAN can generate the expected data properly, it can not infer the latent representations of original data directly, due to the lack of an encoder. Moreover, they usually suffer from the problem of mode collapse [7] as well, which implies that the generator may only provide limited numbers of modes which appear in the training set. As for the VAE-based approaches, they utilize probabilistic graphical models to learn the underlying structure of data, which can approximate the latent representations of original data by optimizing the lower bound on the data log-likelihood. However, this kind of approximation is non-deterministic and inaccurate, since the reparameterization trick requires a random noise vector for data generation, thus may produce unrealistic samples.

Comparing to the wide applications of GAN and VAE, another branch of generative models, namely the flow-based models (such as NICE [8], RealNVP [9], and Glow [10]) have become popular in the recent years. The flow-based models are supposed to learn the data distribution explicitly. To this aim, they exploited a chain of invertible transformations that can convert a complicated

distribution into a tractable form. Since these transformations are reversible, it can be straightforward to learn the target distribution from a simple one following the *change of variables* theorem [10]. The flow-based models have their distinct advantages, allowing: (1) explicit and accurate inference of the latent variable; (2) the log-likelihood optimization on real data; (3) efficiency in parallelization; and (4) potential downstream tasks in the latent space.

Motivated by the advantages mentioned above, we provide a flow-based architecture in this work for human trajectory prediction, where an invertible network serves as the core part, accounting for the multi-modal characteristics of trajectory data. Unlike GAN-based models, the proposed framework can learn the multi-modal distribution explicitly. In contrast to VAE-based models, the proposed framework can provide a deterministic and accurate inference of the latent representation, thus ensuring high-quality data generation. In the experimental section, we will further compare the prediction performances with the standard GAN-based and VAE-based prediction frameworks, and show the strength of the flow-based model. Moreover, the proposed model has a simple yet efficient architecture. The time consumption for both training and inference is much lower than the GAN-based framework.

To sum up, the main contributions of this work are listed as follows:

- (1) We introduce a flow-based invertible network to model multi-modal trajectory distributions, where plausible future paths can be generated by sampling from the standard Gaussian distribution.
- (2) The proposed framework can not only produce multi-modal future trajectories, but also generate trajectories with a pre-defined motion modality.
- (3) We conduct extensive experiments on synthetic data and realistic benchmark datasets, which demonstrate that the proposed framework can achieve better prediction performances as compared to other GAN-based and VAE-based schemes.

The rest of the paper is organized as follows. In Section 2, we review the recent literature in trajectory forecasting. In Section 3, we present the over-

all framework, together with the detailed model training and path forecasting pipelines. Experimental results are reported in Section 4, to evaluate the performances in terms of multi-modal trajectory generation and prediction accuracy. Conclusions are drawn in Section 5.

65 2. Related work

Trajectory prediction has emerged as a relevant topic in computer vision, with a wide range of applications, including video surveillance [11], abnormal event detection [12] [13], and crowd analysis [14] [15] [16], to name a few. In the following, we will present the recent progress in detail.

70 In its early stages, the research in this area has focused on simple yet effective solutions, such as the first-order Markov model. However, the limitation of handling only one-step predictions turned out to be insufficient in many realistic scenarios [17]. With the emergence of recurrent neural networks (RNN), long-term future prediction has become applicable. Moreover, thanks to the help of
75 attention mechanisms [18], social interactions can be taken into considerations as well. In [19], Alahi et al. presented the so-called social-LSTM for trajectory prediction, where the social pooling layer was introduced to capture the importance of each person in the neighborhood. In [20], Bartoli et al. proposed a context-aware recurrent neural network, which can jointly model human-human
80 and human-space interactions for trajectory forecasting. In [21], Huang et al. used a spatial-temporal attention network for trajectory prediction, where mutual interactions were modeled by the spatial attention mechanism, while the temporal correlations of interactions were modeled using another extra LSTM block. In [22], Vineet et al. exploited the graph attention networks to model
85 el interactions among pedestrians. In [23], Mohamed et al. presented a novel convolutional neural network which was built upon the social spatio-temporal graph, allowing interaction modeling through graphical models. In [24], Zhang et al. proposed a hybrid spatio-temporal embedding network for trajectory forecasting, where group influences were taken into account as well.

90 Most approaches presented above can merely provide uni-modal prediction.
 In order to capture the uncertainty of future behaviors, generative models were
 widely exploited, due to their capability of understanding the underlying struc-
 ture of data in the physical world and the promising performances in the appli-
 cations of sequential data modeling, such as image generation [25] [26], audio
 95 synthesis [27], and text analysis [28], etc. In [3], Gupta et al. provided a
 novel path prediction framework, which combined sequence modeling and gen-
 erative adversarial networks together. The overall framework was trained in
 a generative-adversarial manner, where the generator G was able to produce
 plausible future locations, and the discriminator D was used to evaluate the
 100 quality of predicted trajectories. In [6], Amirian et al. exploited Info-GAN for
 multi-modal trajectory prediction, in order to deal with the problem of mod-
 e collapse. In [4], the so-called Sophie model was proposed, which adopted a
 similar GAN-based mechanism. In [5], Lee et al. proposed the DESIRE frame-
 work for multi-modality modeling, where the core sample generation module was
 105 implemented using a conditional variational auto-encoder (CVAE), which can
 produce multiple plausible trajectories. In addition to generative models, other
 approaches have been investigated as well. In [29], Makansi et al. proposed
 a sampling and fitting framework for multi-modal future trajectory prediction,
 which can overcome the limitation of traditional mixture density networks. In
 110 [30], Li et al. presented a generic trajectory forecasting framework based on
 evolving interaction graphs, which was able to jointly forecast the multi-modal
 trajectory distribution of multiple agents in the scene. In [31], Rhinehart et
 al. proposed a re-parameterized push-forward policy that can be applied under
 the autonomous driving background, achieving very promising performances on
 115 the KITTI dataset [32]. In [33], a goal-conditioned forecasting strategy was
 proposed for autonomous vehicles. By exploiting the past position and LIDAR
 data, it is able to predict multi-modal future trajectories in a multi-agent set-
 ting. In [34], Yuan et al. proposed a novel transformer for multi-agent trajectory
 forecasting, which was able to model the time and social dimensions simultane-
 120 ously. In [35], Song, et al. presented a model-based planning framework, which

aims at predicting the future trajectories of on-road vehicles. In [36], Salzmann et al. designed a graph-structured recurrent model called *Trajectron++* for diverse trajectory prediction, which integrated with robotic planning and decision making strategies. In [37], Liang et al. proposed a feature pyramid network for
125 trajectory prediction, which can learn rich temporal information at multiple scales.

3. Methodology

In the following, we will first present the annotations used in this work:

- Let $\mathbf{X}=\{x_1, x_2, \dots, x_{T_{obs}}\}$ and $\mathbf{Y}=\{y_1, y_2, \dots, y_{T_{pred}}\}$ be the observation
130 sequence and the ground truth of the prediction sequence of a trajectory, respectively, where T_{obs} and T_{pred} indicate the corresponding sequence lengths. Each x_i (or y_i) is a 2D coordinate that indicates the position of a pedestrian on the ground plain.
- Let $\hat{\mathbf{Y}}=\{\hat{y}_1, \hat{y}_2, \dots, \hat{y}_{T_{pred}}\}$ be the prediction sequence.
- x_t^h (or y_t^h) indicates the hidden representation of a trajectory point at a
135 given time step t .
- z indicates the latent variable generated from the Gaussian distribution, which is used to capture the diversity of the future path.

The whole framework consists of an encoder, an decoder, and an invertible
140 network, which aims at estimating the future path $\hat{\mathbf{Y}}$ given the past motion history $\mathbf{X}$ (as shown in Figure 1). The encoder and the decoder are implemented using the standard LSTM module, aiming at modeling the temporal evolution of trajectory data. Specifically, we use two independent LSTMs to model the observation and the prediction sequences, respectively. The invertible network
145 is supposed to learn the distribution of future trajectories, where the hidden representation of a future trajectory can be converted into a latent variable that subjected to the standard Gaussian distribution. Once the model is fully trained, the future path can be readily generated by sampling tricks.

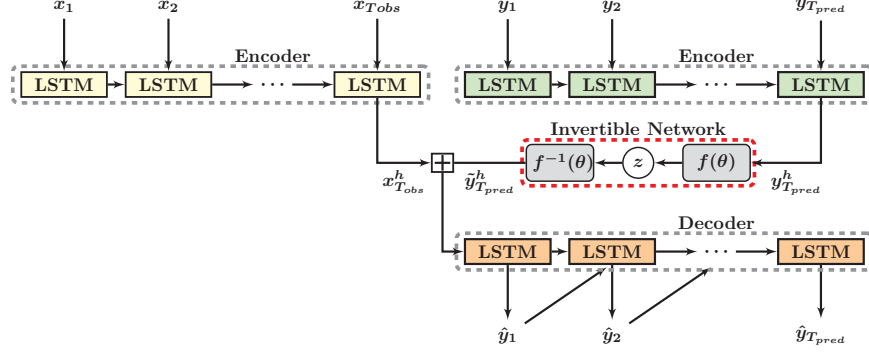


Figure 1: The overall framework is comprised by an encoder and a decoder. Specifically, we introduce an invertible network in the encoder, which aims at modeling the distributions of the future trajectories. $\boxplus$ represents the concatenation operation.

3.1. Invertible network

For the hidden representation of a trajectory point in the prediction sequence, namely, y_t^h , we build an invertible network, which is able to map y_t^h into a random variable z (usually sampled from a tractable distribution). We use θ to indicate the parameters of the invertible network, as presented in Eq. (1):

$$\begin{aligned} \text{forward: } z &= f(y_t^h; \theta); y_t^h \sim P_h(\cdot). \\ \text{reverse: } y_t^h &= f^{-1}(z; \theta); z \sim P_z(\cdot). \end{aligned} \quad (1)$$

Following the *change of variables* theorem, we have:

$$P_h(y_t^h) = P_z(z) \cdot \left| \det \left(\frac{dz}{dy_t^h} \right) \right|, \quad (2)$$

150 where $\det(\cdot)$ computes the determinant of a matrix.

For a given training set of N trajectories, we would like to maximize the joint likelihood of $\{y_{t,i}^h\}$ ($i=1, 2, 3, \dots, N$), which equals to minimize its negative log likelihood (NLL). Therefore, the objective function $L(\theta)$ can be formulated as:

$$\begin{aligned} L(\theta) &= - \sum_{i=1}^N \log P_h(y_{t,i}^h) \\ &= - \sum_{i=1}^N [\log P_z(z_i) + \log \left| \det \left(\frac{dz_i}{dy_{t,i}^h} \right) \right|]. \end{aligned} \quad (3)$$

According to Eq. (1), we have $z_i = f(y_{t,i}^h)$, where $z_i = \{z_{i,1}, z_{i,2}, \dots, z_{i,d}\}$ is a d -dimensional vector. Assume each element in z_i is independent, and follows the standard Gaussian distribution, then the term $\log P_z(z_i)$ in Eq. (3) can be rewritten as:

$$\begin{aligned} \log P_z(z_i) &= \log \prod_{j=1}^d P_z(z_{i,j}) = \sum_{j=1}^d \log P_z(z_{i,j}) \\ &= \sum_{j=1}^d \log \left(\frac{1}{\sqrt{2\pi}} e^{-\frac{z_{i,j}^2}{2}} \right) = -\frac{1}{2} \sum_{j=1}^d (z_{i,j}^2 + \log(2\pi)). \end{aligned} \quad (4)$$

As for the second term in the brackets in Eq. (3), we exploit the affine coupling layer as presented in [9] to compute the Jacobian matrix. For any y_t^h , we first split it into two separated parts, A and B , which will be involved in the affine operation according to the following steps (see Eq. (5)).

$$\begin{aligned} (A, B) &= \text{split}(y_t^h); \\ s &= \text{NN}_s(B), t = \text{NN}_t(B); \\ z_a &= s \cdot A + t, z_b = B; \\ z &= \text{concate}(z_a, z_b). \end{aligned} \quad (5)$$

In the above equations, $\text{NN}_s(\cdot)$ and $\text{NN}_t(\cdot)$ can be any complicated neural networks, whose weight matrices are the main parameters of the invertible network. This operation can guarantee: (1) the transformation $y_t^h \rightarrow z$ is invertible; and (2) the Jacobian matrix is lower triangular, whose determinant is easy to compute. Therefore, we can simplify the computation as in Eq. (6):

$$\log |\det(\frac{dz_i}{dy_{t,i}^h})| = \text{sum}(\log |\text{diag}(\frac{dz_i}{dy_{t,i}^h})|), \quad (6)$$

where $\text{diag}(\cdot)$ takes the diagonal of the Jacobian matrix, $\log(\cdot)$ computes the element-wise logarithm of a vector, and $\text{sum}(\cdot)$ indicates the summation of all the elements in a vector.

Obviously, Eqs. (4) and (6) are merely the functions of θ . We substitute them into Eq. (3), and obtain the final form of the objective function.

As for y_t^h , we set $t = T_{pred}$ in this work. Since T_{pred} represents the last time step in the prediction sequence, $y_{T_{pred}}^h$ can preserve the motion information

with respect to the overall prediction sequence. Considering all the N training trajectories, we can obtain a set of $\{y_{T_{pred},i}^h\}$ ($i=1, 2, \dots, N$) from the LSTM model. Standard stochastic gradient descent algorithms can be used to optimize the objective function, in order to estimate the parameters θ of the invertible network.

3.2. Model training

For model training, our targets are: (1) learning parameters of the LSTM blocks in the encoder and decoder, respectively, which are used to generate future locations sequentially; (2) learning parameters of the invertible networks, which enables the bidirectional mapping: $y_{T_{pred}}^h \leftrightarrow z$, where z is restricted to the standard Gaussian distribution. This constraint implies that it can be easy to infer the future path by sampling. Particularly, $y_{T_{pred}}^h$ and z should have the same dimension.

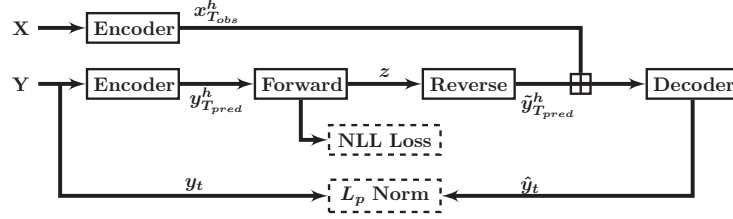


Figure 2: The training diagram of the proposed framework.

The training diagram is presented as in Figure 2, which can be summarized in Eq. (7). The encoder consists of two separated LSTMs, which model the observation and the prediction sequences, respectively. We use $x_{T_{obs}}^h$ and $y_{T_{pred}}^h$ to indicate the hidden representations of the two sequences at the corresponding time steps. The function $\text{invert}(\cdot)$ indicates the mapping modeled by the invertible network: $y_{T_{pred}}^h \rightarrow z$, and $z \rightarrow \tilde{y}_{T_{pred}}^h$. With the training procedure going on, $\tilde{y}_{T_{pred}}^h$ will gradually approximate to $y_{T_{pred}}^h$. The concatenation of $\tilde{y}_{T_{pred}}^h$ and $x_{T_{obs}}^h$ serves as the input of the decoder, which will then generate the future

locations sequentially.

$$\begin{aligned}
x_{T_{obs}}^h &= \text{encoder}(\mathbf{X}), \\
y_{T_{pred}}^h &= \text{encoder}(\mathbf{Y}), \\
\tilde{y}_{T_{pred}}^h &= \text{invert}(y_{T_{pred}}^h), \\
\hat{y} &= \text{decoder}(\text{concat}(x_{T_{obs}}^h, \tilde{y}_{T_{pred}}^h)).
\end{aligned} \tag{7}$$

The overall loss (see Eq. (8)) is comprised by two parts: (1) the reconstruction error, which exploits the L_p norm between y and $\hat{y}$; and (2) the NLL loss of the invertible network.

$$\mathcal{L} = \sum_{t=1}^{T_{pred}} \|y_t - \hat{y}_t\|_p + L(\theta). \tag{8}$$

When training recurrent networks, two different training modes can be exploited, namely the *free-running* mode and the *teacher-forcing* mode. As for the *free-running* mode, the output of the current state serves as the input of the next state directly. On the contrary, in the *teacher-forcing* mode, the output of the current state is replaced by the ground truth as the input of the next state. In our implementation, we use the *teacher-forcing* mode for path prediction, since it can achieve better prediction performances which have been verified in the experimental section.

3.3. Trajectory prediction

Once the training process is complete, the invertible network is able to generate proper $y_{T_{pred}}^h$ from a random variable z sampled from the standard Gaussian distribution. Therefore, only the ‘reverse’ direction is exploited in path forecasting. The prediction procedure is formulated as in Eq. (9).

$$\begin{aligned}
x_{T_{obs}}^h &= \text{encoder}(\mathbf{X}), \\
y_{T_{pred}}^h &= f^{-1}(z), z \sim N(0, 1), \\
\hat{y} &= \text{decoder}(\text{concat}(x_{T_{obs}}^h, y_{T_{pred}}^h)).
\end{aligned} \tag{9}$$

180 4. Experiments

In the following paragraphs, we will present the experimental results. First, we introduce the standard datasets used in this work, together with the evaluation protocols and the hyper-parameters of the framework. Then, we validate the prediction performances in terms of *precision* and *diversity* on the synthetic data. Next, we carry out our experiments on realistic data, and compare the performances with other baseline methods. Finally, we discuss about some other characteristics of the proposed model.

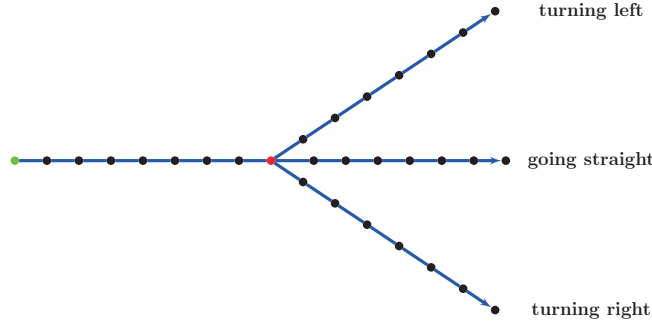


Figure 3: The synthetic data configuration.

4.1. Benchmarks

4.1.1. Synthetic data

190 We construct a synthetic dataset for path prediction and multi-modal trajectory generation. Examples of the training samples can be found in Figure 3. The green dot represents the starting point of a trajectory. The red dot represents the intersection point, where an agent starts to choose its future directions, namely: turning left, going straight, and turning right. The portion between
195 the green and the red dots indicate the observation sequence, which consists of 8 discrete trajectory points. Arrows indicate potential motion directions of the future movements, each of which is comprised 8 discrete trajectory points as well, demonstrating multiple motion trends. The interval between two consecutive trajectory points is set to 1. Each point is mixed with random deviations in

Table 1: The hyper-parameters adopted in our experiments.

parameter	value
batch size	64
learning rate	5e-4
epoch	200
hidden dimension of the LSTM (x_t^h)	64
hidden dimension of the LSTM (y_t^h)	64
hidden dimension of the LSTM (decoder)	128

the horizontal and vertical directions, thus allowing generating diverse samples with arbitrary numbers.

4.1.2. ETH and UCY datasets

The ETH [38] and the UCY [39] datasets are standard benchmarks in the field of trajectory analysis, which contain five subsets in total, namely *eth*, *hotel*, *zara-1*, *zara-2*, and *univ*. These datasets are collected from the bird-eye view camera in realistic environments, including varied behaviors such as group movements, walking together, and collision avoidance, etc.

4.2. Evaluation protocols and hyper-parameter setting

We adopt the average displacement error (ADE) and the final displacement error (FDE) metrics to evaluate the prediction accuracy quantitatively. The Adam algorithm [40] is adopted as the optimizer. The hyper-parameters used in the experiments are listed in Table 1. The invertible variable z has the same size with $y_{T_{pred}}^h$, which is a 1×64 vector.

4.3. Results

4.3.1. Evaluations on the capability of path generation using synthetic data

First, we validate that the proposed model is able to generate plausible future paths. For this purpose, we organize three groups of tests with respect to each motion direction, separately.

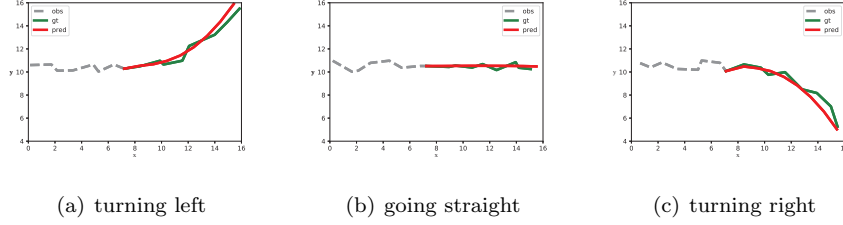


Figure 4: Visualizations on synthetic data. The red curve indicates the predicted trajectory. The green curve represents the ground truth with respect to the prediction portion. The dash line indicates the observation. As can be seen from the figures, the proposed approach is able to generate future paths with specific motion directions properly.

Table 2: Prediction performances on each motion direction using synthetic data (ADE/FDE).

Synthetic data	RNN	LSTM	Flow-based
turning left	0.53/0.49	0.53/0.49	0.45/0.46
going straight	0.52/0.52	0.52/0.51	0.46/0.46
turning right	0.54/0.56	0.54/0.55	0.43/0.44
AVG	0.53/0.52	0.53/0.52	0.45/0.45

- turning left: 1,000 trajectories for training, and another 1,000 for test;
- going straight: 1,000 trajectories for training, and another 1,000 for test;
- turning right: 1,000 trajectories for training, and another 1,000 for test.

We compare our method with other recurrent networks (such as standard RNN and LSTM) in the prediction task. The obtained results can be found in Table 2, from where we can find that the flow-based model can achieve superior performances. As compared to RNN and LSTM, the average ADE decreases by 15% and the average FDE decreases by 13.5%. We visualize the prediction performances in Figure 4, from where we can observe that the flow-based model is able to generate the future paths properly.

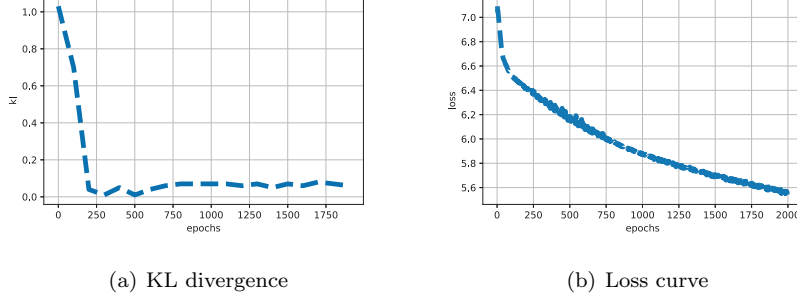
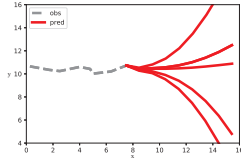


Figure 5: When the KL distance is close to zero, the distribution of the predicted trajectories almost approximates to the real data distribution, which implies that the model is trained properly.

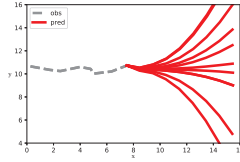
Second, we validate that the proposed model is able to generate multi-modal
 230 future trajectories successfully. In this experiment, we use 3,000 trajectories in
 total as the training samples, including 1,000 turning left, 1,000 going straight,
 and 1,000 turning right, demonstrating multi-modal motion characteristics. In
 the test phase, given another 3,000 observation sequences that exhibit the going
 straight pattern, the model is supposed to generate varied motion patterns at the
 235 intersection point (note: these 3,000 observation sequences are slightly different
 from each other). In Table 3, we record the number of predicted trajectories with
 respect to each direction, and use the KL divergence to measure the difference
 between the distributions of the predicted trajectories and the corresponding
 ground truths at the intersection point. From the experimental results we can
 240 find that the flow-based model is able to approximate trajectory distributions
 at the intersection point properly when the KL divergence decreases to 0.01. In
 Figure 5 (a), we present the values of the KL divergence at different iterations.
 In Figure 5 (b), we show the corresponding training loss. In additional, we
 visualize the prediction performances in Figure 6. For the same observation
 245 sequence, we sample M times (M is set to 5, 10, and 15, accordingly) from
 the standard Gaussian distribution, and then produce multiple plausible future
 paths with varied motion patterns.

Table 3: Multi-modal prediction performances on synthetic data. We record the number of trajectories in every motion direction at the intersection point for varied iterations.

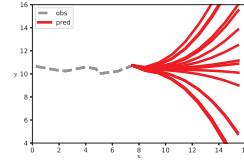
iteration	left	straight	right	KL divergence
#100	85	2,719	196	1.03
#200	240	2,561	199	0.70
#300	644	1,215	1,141	0.04
#400	811	1,033	1,156	0.01
#500	635	976	1,389	0.05
#600	968	820	1,212	0.01
#700	1,094	640	1,266	0.04



(a) $M=5$



(b) $M=10$



(c) $M=15$

Figure 6: Visualizations of multi-modal trajectory predictions on synthetic data. We did not use more values of M , in order to avoid overlapping when visualizing the results. We can find that the proposed framework is able to generate multi-modal trajectories properly with respect to the same observation.

Table 4: The ablation study on synthetic data. We compare the prediction accuracy (in terms of ADE/FDE) by removing the flow-based mechanism from our framework. k indicates the sampling numbers in the test phrase.

models	sampling number		
	$k = 1$	$k = 5$	$k = 20$
Without the flow-based mechanism	1.69/4.32	1.58/3.94	1.51/3.69
Using the flow-based mechanism	1.61/4.22	0.76/1.30	0.50/0.47

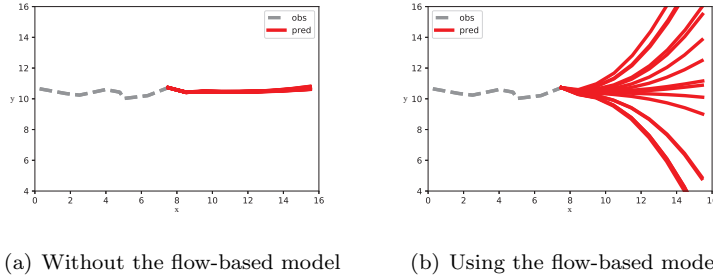


Figure 7: Visualization of the ablation study. When removing the flow-based model from the framework, only one motion modality can be generated. On the contrary, multi-modal trajectories can be generated if the flow-based mechanism is exploited.

In order to better demonstrate the effectiveness of the proposed model, we further conduct an ablation experiment on synthetic data by removing the flow-based mechanism from our framework, where the hidden representation of the observation sequence is concatenated with a random noise z directly, and then processed by the decoder to generate the future positions. It can be seen clearly in Table 4 that the prediction accuracy will decrease dramatically when removing the flow-based mechanism. Particularly, when the sampling number is set to 20, the average ADE will increase from 0.50 to 1.51, and the average FDE will increase from 0.47 to 3.69. Moreover, as shown in Figure 7, the model will lose the capability of generating multi-modal trajectories if the flow-based mechanism is not exploited.

Finally, we will show that the proposed framework is able to generate conditional sample trajectories as well given the pre-defined motion modes, for examples, generating the *turning left* trajectories. This is similar to the conditional GAN (CGAN) [41] and the conditional VAE (CVAE) [42]. We use the term *label* to indicate varied motion modes, which is defined as follows:

$$label = \begin{cases} +1, & \text{turning left;} \\ 0, & \text{going straight;} \\ -1, & \text{turning right.} \end{cases} \quad (10)$$

As shown in Figure 8, the motion direction can be determined by a vector OP , where O is the last point in the observation sequence (namely, $x_{T_{obs}}$). The coordinate P is obtained through Eq. (11), where τ is a random variable that follows a uniform distribution of $U(-0.4, +0.4)$.

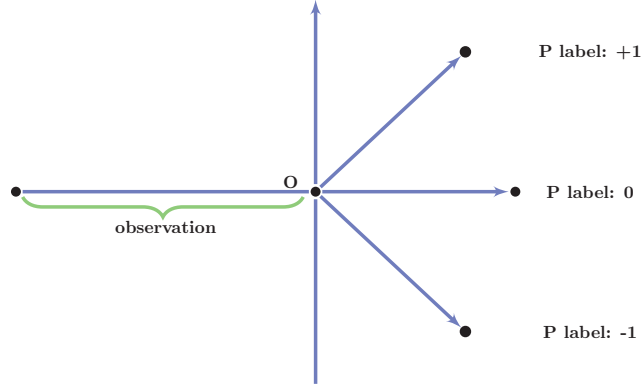


Figure 8: Conditional trajectory generation.

$$\begin{cases} P_x = O_x + 1, \\ P_y = O_y + label + \tau. \end{cases} \quad (11)$$

For a pre-defined motion mode (indicated by *label*), we first compute the coordinate of P , and then sent it to the encoder and the invertible network

265 accordingly. Given the observation sequences, we follow the same trajectory generation procedure to obtain the future path with respect to the preferred motion direction. The corresponding experimental results are demonstrated in Figure 9.

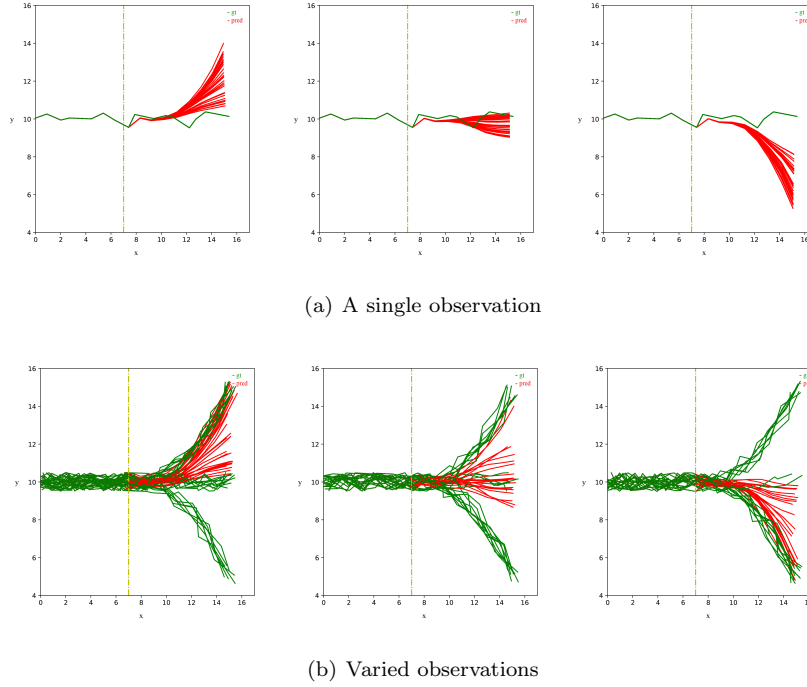


Figure 9: Conditional trajectory generation. (a) for the same observation, the model can generate multiple plausible trajectories according to the pre-defined motion direction. (b) for varied observations, we generate a single mode of the future trajectory with respect to each observation, given the preferred motion direction.

270 4.3.2. Evaluations on the performances of path prediction using realistic data

In this section, we validate the proposed model on realistic benchmarks, namely, the ETH and the UCY datasets. For comparison, the LSTM model, the social-LSTM [19], the VAE-based model, and the social-GAN [3] are used as the baseline approaches. We set $T_{obs}=8$ and $T_{preds}=12$. For evaluation, the 5-fold

cross-validation strategy is used as previously adopted in [3]. The quantitative results are reported in Table 5, showing that the proposed framework is able to achieve very promising performances as compared to the standard LSTM-based strategies and the other two generative model-based frameworks. Specifically, as compared to the social-GAN model, the average ADE decreases by 37.9% and the average FDE decreases by 39.0%. As compared to the VAE-based model, the average ADE decreases by 28.0% and the average FDE decreases by 34.5%.

Table 5: Trajectory prediction performances using the ADE/FDE metrics on realistic data (the ETH and the UCY datasets). Note: for the baseline approaches, we use the results reported in [3] for comparison. S-LSTM indicates social-LSTM; S-GAN indicates social-GAN. The VAE-based framework is implemented according to [5].

dataset	LSTM	S-LSTM	S-GAN	VAE-based	Flow-based
zara1	0.41/0.88	0.47/1.00	0.34/0.69	0.38/0.83	0.27/0.52
zara2	0.52/1.11	0.56/1.17	0.42/0.84	0.29/0.64	0.25/0.46
eth	1.09/2.41	1.09/2.35	0.81/1.52	0.88/1.95	0.64/1.39
hotel	0.86/1.91	0.79/1.76	0.72/1.61	0.37/0.81	0.28/0.55
univ	0.61/1.31	0.67/1.40	0.60/1.26	0.59/1.27	0.36/0.70
AVG	0.70/1.52	0.72/1.54	0.58/1.18	0.50/1.10	0.36/0.72

Furthermore, we compare the proposed model with other typical non-deep learning algorithms, including the constant velocity model (CVM [43]), Kalman filter, and liner regression [44]. Experimental results are reported in Table 6, showing that the flow-based framework performs much better than the non-deep learning approaches.

In addition, we conduct an ablation experiment to evaluate the impact of the flow-based mechanism on realistic data. The corresponding results are listed in Table 7, from where we can observe that the ADE and the FDE metrics will increase by 32.1% and 37.9% when the invertible network is removed from the proposed framework, further verifying the effectiveness of the flow-based

Table 6: Comparisons with typical non-deep learning algorithms.

	linear regression	CVM	Kalman	Flow-based
eth	1.33/2.94	1.00/2.23	0.94/2.10	0.64/1.39
univ	0.82/1.59	0.62/1.38	0.74/1.51	0.36/0.70
hotel	0.39/0.72	0.32/0.62	0.23/0.44	0.28/0.55
zara1	0.62/1.21	0.43/0.96	0.52/1.08	0.27/0.52
zara2	0.77/1.48	0.33/0.73	0.40/0.82	0.25/0.46
AVG	0.79/1.59	0.54/1.18	0.57/1.19	0.36/0.72

mechanism.

Table 7: The ablation study on realistic data (the ETH and the UCY datasets). We compare the average prediction accuracy (in terms of ADE/FDE) by removing the flow-based mechanism from our framework. k indicates the sampling numbers in the test phrase.

models	sampling number		
	$k = 1$	$k = 5$	$k = 20$
Without the flow-based mechanism	0.57/1.23	0.55/1.19	0.53/1.16
Using the flow-based mechanism	0.95/1.98	0.53/1.11	0.36/0.72

Finally, we visualize the prediction performances qualitatively on the above realistic scenarios in Figures 10 and 11, respectively. The dash line indicates the observation sequence, the red solid line indicates the prediction sequence, and the green solid line corresponds to the ground truth. It can be seen clearly that the proposed framework is able to generate plausible future paths successfully.

4.3.3. Discussion

In the following, we will provide a comprehensive evaluation on different aspects of the proposed model. First, we compare the computational complexity of the proposed model with the other two generative model-based prediction

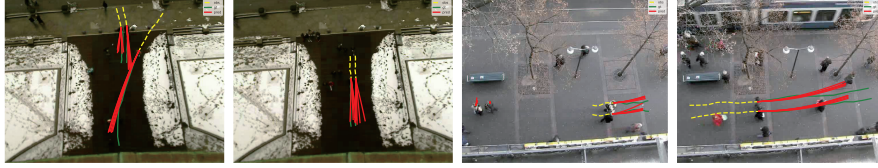


Figure 10: Visualizations of the prediction performances on the *eth* and *hotel* subsets. The dash line indicates the observation sequence, the red solid line indicates the prediction sequence, and the green solid line corresponds to the ground truth.



Figure 11: Visualizations of the prediction performances on the *zara1* and *zara2* subsets. The dash line indicates the observation sequence, the red solid line indicates the prediction sequence, and the green solid line corresponds to the ground truth.

frameworks. Then, we evaluate the prediction performances which use L_1 and
 305 L_2 norms in the loss function, respectively. Next, we investigate on different
 training strategies. Finally, we discuss about how varied lengths of observation
 sequences influence on the prediction performance.

Computational complexity: We provide the time consumption on each sce-
 310 nario of the ETH/UCY datasets (in average) in Table 8, where the GAN-based,
 the VAE-based, and the flow-based prediction frameworks are used for compar-
 ison. It can be seen clearly that the computational complexity of the proposed
 model is much lower as compared to the GAN-based framework (namely S-
 GAN). Using a single GeForce RTX 3080 card, training the proposed model
 315 with 200 epochs on each scenario needs around 5 minutes. On the contrary,
 training the S-GAN model needs around 110 minutes. Furthermore, the pro-
 posed model runs faster in the inference as well. As compared to the VAE-based
 model, their time consumption are more or less the same, but the flow-based
 model can obtain better prediction performances.

Table 8: Time consumption on each scenario of the ETH/UCY datasets in average (in sec-
 onds). Note: a single GeForce RTX 3080 card is exploited for model training, and we train
 each model using 200 epochs.

	S-GAN	VAE-based model	Flow-based model
training	6621.1	240.6	272.4
inference	83.8	1.3	1.6

320 **Varied loss functions:** We compare the performances that use different norms
 (L_1 vs. L_2) in the loss function. As shown in Table 9, the adoption of L_1 norm
 demonstrates better prediction accuracy on almost all the five subsets in real-
 istic environments. The average ADE and the FDE metrics decrease by 14.3%
 325 and 13.3%, respectively.

Table 9: Prediction performances (ADE/FDE) using L_1 and L_2 norms on realistic data.

dataset	zara1	zara2	eth	hotel	univ	AVG
L_1 norm	0.27/0.52	0.25/0.46	0.64/1.39	0.28/0.55	0.36/0.70	0.36/0.72
L_2 norm	0.42/0.87	0.38/0.79	0.59/1.13	0.37/0.74	0.32/0.60	0.42/0.83

Table 10: Evaluations on different training modes on realistic data (ADE/FDE).

dataset	zara1	zara2	eth	hotel	univ	AVG
teacher forcing	0.27/0.52	0.25/0.46	0.64/1.39	0.28/0.55	0.36/0.70	0.36/0.72
free-running	0.34/0.70	0.33/0.65	0.52/0.84	0.47/0.93	0.36/0.68	0.40/0.76

Training strategies: We compare the performances that use different training modes (*teacher forcing* vs. *free-running*) for path forecasting. The corresponding results are shown in Table 10. It can be seen clearly that the adoption of *teacher forcing* mode demonstrates better performances, where the average ADE and FDE metrics will decrease by 10.0% and 5.3% as compared to using the free-running strategy.

Varied lengths of the observation sequences: Intuitively, longer observation sequences may lead to more accurate prediction. However, considering the benchmark datasets at hand (ETH and UCY), the sample trajectories are almost linear in nature. In this case, we argue that only the most recent observations may contribute to the forecasting task, whereas earlier observations may not play significant roles. To this end, we use varied observation lengths for comparison. The experimental results are demonstrated in Table 11, from where we can find that the latest observation with the length of three time steps is more than enough to obtain promising results. This is also in line with the

Table 11: Prediction performances (ADE/FDE) using different lengths of observation sequences on realistic data, where T_{obs} is set to 3, 5, and 8 accordingly.

dataset	$T_{obs}=3$	$T_{obs}=5$	$T_{obs}=8$
zara1	0.24/0.44	0.25/0.45	0.27/0.52
zara2	0.25/0.49	0.27/0.52	0.25/0.46
eth	0.62/1.35	0.61/1.27	0.64/1.39
hotel	0.27/0.56	0.27/0.54	0.28/0.55
univ	0.36/0.71	0.36/0.71	0.36/0.70
AVG	0.35/0.71	0.35/0.70	0.36/0.72

conclusion of the recent work [43].

In addition, we would like to point out that in real applications, the motion
 345 history is usually obtained via object detection and tracking. Thus, the prediction accuracy is also relevant to the performances of the selected trackers. In some extreme cases, such as dense crowds with frequent mutual occlusions, trackers are not reliable, thus the prediction results may be not that perfect as expected.

350

Ablation study: In the following paragraphs, we will investigate on the potentials of incorporating other popular tricks to promote the prediction performances. To this aim, we exploit the proposed flow-based generative model as the backbone, and then introduce the graph attention mechanism [21] and the
 355 trajectory clustering strategy [45] into the framework for further evaluation.

Experimental results are reported in Table 12, from where we can find that by exploiting the graph attention mechanism, the ADE/FDE metrics will be improved from 0.36/0.72 to 0.30/0.56. By further integrating the trajectory clustering strategy, the ADE/FDE metrics can reach to 0.28/0.50. This ablation
 360 study validates that the naive flow-based models are flexible enough to integrate other tricks (such as clustering strategies, attention mechanisms, etc.), and the

prediction performances can be promoted correspondingly.

Table 12: Prediction performances (ADE/FDE) by incorporating the graph attention mechanism and the trajectory clustering strategy.

dataset	flow-based	graph attention	graph attention + clustering
eth	0.64/1.39	0.49/0.80	0.47/0.75
univ	0.36/0.70	0.31/0.57	0.29/0.51
hotel	0.28/0.55	0.29/0.59	0.25/0.51
zara1	0.27/0.52	0.20/0.39	0.19/0.35
zara2	0.25/0.46	0.22/0.43	0.21/0.38
AVG	0.36/0.72	0.30/0.56	0.28/0.50

5. Conclusions

In this paper, we present a trajectory prediction framework using a flow-based generative model. In order to build the distributions of future trajectories, an invertible network is exploited, which enables explicit bidirectional transformations between the hidden representation of a future trajectory and a latent variable z that subjected to the standard Gaussian distribution. This allows generating multiple and plausible future paths by sampling from a tractable distribution. We validate our approach on both synthetic and realistic datasets. Experimental results demonstrate that the flow-based models can achieve better prediction performances as compared to the standard GAN-based and VAE-based models. Moreover, the trajectory diversity can be maintained as well. As for the future work, we would like to further investigate the influences of environmental visual cues on the path forecasting task.

6. Acknowledgment

This work is supported by the National Natural Science Foundation of China (Grant No. 61702073, 61772102, 62176036), the China Postdoctoral Science

Foundation (Grant No. 2019M661079), the Liaoning Collaborative Fund (Grant
380 No. 2020-HYLH-17), and the Fundamental Research Funds for the Central
Universities (Grant No. 3132022227).

References

- [1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley,
S. Ozair, A. Courville, Y. Bengio, Generative adversarial nets, in: Pro-
385 ceedings of Advances in Neural Information Processing Systems, 2014, pp.
2672–2680.
- [2] D. P. Kingma, M. Welling, Auto-encoding variational bayes, in: Proceed-
ings of the International Conference on Learning Representations, 2014.
- [3] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, A. Alahi, Social GAN: So-
390 cially acceptable trajectories with generative adversarial networks, in: Pro-
ceedings of the IEEE International Conference on Computer Vision and
Pattern Recognition, IEEE, 2018, pp. 2255–2264.
- [4] A. Sadeghian, V. Kosaraju, A. Sadeghian, N. Hirose, S. Savarese, Sophie:
An attentive GAN for predicting paths compliant to social and physical
395 constraints, in: Proceedings of the IEEE International Conference on Com-
puter Vision and Pattern Recognition, IEEE, 2019, pp. 1349–1358.
- [5] N. Lee, W. Choi, P. Vernaza, C. B. Choy, P. H. Torr, M. Chandraker, DE-
SIRE: Distant future prediction in dynamic scenes with interacting agents,
in: Proceedings of the IEEE International Conference on Computer Vision
400 and Pattern Recognition, IEEE, 2017, pp. 336–345.
- [6] J. Amirian, J. B. Hayet, J. Pettre, Social Ways: Learning multi-modal
distributions of pedestrian trajectories with GANs, in: Proceedings of the
IEEE International Conference on Computer Vision and Pattern Recogni-
tion Workshop, IEEE, 2019, pp. 2964–2972.

- 405 [7] A. Srivastava, L. Valkov, C. Russell, M. U. Gutmann, C. Sutton, VEEGAN: Reducing mode collapse in GANs using implicit variational learning, in: Proceedings of the Neural Information Processing Systems, 2017, pp. 3310–3320.
- [8] L. Dinh, D. Krueger, Y. Bengio, NICE: non-linear independent components estimation, in: Proceedings of the International Conference on Learning
410 Representations, 2015.
- [9] L. Dinh, J. Sohl-Dickstein, S. Bengio, Density estimation using real NVP, in: Proceedings of the International Conference on Learning Representations, 2017.
- 415 [10] D. P. Kingma, P. Dhariwal, Glow: Generative flow with invertible 1x1 convolutions, in: Proceedings of Advances in Neural Information Processing Systems, 2018, pp. 10215–10224.
- [11] S. A. Ahmed, D. P. Dogra, S. Kar, P. P. Roy, Trajectory-based surveillance analysis: A survey, IEEE Transactions on Circuits and Systems for Video
420 Technology 29 (7) (2019) 1985–1997.
- [12] S. Coşar, G. Donatiello, V. Bogorny, C. Garate, L. O. Alvares, F. Bremond, Toward abnormal trajectory and event detection in video surveillance, IEEE Transactions on Circuits and Systems for Video Technology 27 (3) (2017) 683–695.
- 425 [13] R. Rodrigues, N. Bhargava, R. Velmurugan, S. Chaudhuri, Multi-timescale trajectory prediction for abnormal human activity detection, in: Proceedings of the IEEE Winter Conference on Applications of Computer Vision, 2020, pp. 2626–2634.
- [14] J. Shao, C. C. Loy, X. Wang, Learning scene-independent group descriptors
430 for crowd understanding, IEEE Transactions on Circuits and Systems for Video Technology 27 (6) (2017) 1290–1303.

- [15] T. Li, H. Chang, M. Wang, B. Ni, R. Hong, S. Yan, Crowded scene analysis: A survey, *IEEE Transactions on Circuits and Systems for Video Technology* 25 (3) (2015) 367–386.
- 435 [16] D. Wang, Q. Ma, N. Wang, X. Fan, M. Lu, H. Liu, AONet: Active offset network for crowd flow prediction, *Engineering Applications of Artificial Intelligence* 97 (2021) 104022.
- [17] P. Rathore, D. Kumar, S. Rajasegarar, M. Palaniswami, J. C. Bezdek, A scalable framework for trajectory prediction, *IEEE Transactions on Intelligent Transport Systems* 20 (10) (2019) 3860–3874.
- 440 [18] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, in: *Proceedings of Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008.
- [19] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, S. Savarese, 445 Social LSTM: Human trajectory prediction in crowded spaces, in: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 961–971.
- [20] F. Bartoli, G. Lisanti, L. Ballan, A. D. Bimbo, Context-aware trajectory prediction, in: *Proceedings of the IEEE International Conference on Pattern Recognition*, IEEE, 2018, pp. 1941–1946.
- 450 [21] Y. Huang, H. Bi, Z. Li, T. Mao, Z. Wang, STGAT: Modeling spatial-temporal interactions for human trajectory prediction, in: *Proceedings of the IEEE International Conference on Computer Vision*, IEEE, 2019, pp. 6272–6281.
- 455 [22] V. Kosaraju, A. Sadeghian, R. Martn-Martn, I. Reid, S. H. Rezatofighi, S. Savarese, Social-BiGAT: Multimodal trajectory forecasting using bicycle-GAN and graph attention networks, in: *Proceedings of the Neural Information Processing Systems*, 2019, pp. 137–146.

- [23] A. Mohamed, K. Qian, M. Elhoseiny, C. Claudel, Social-STGCNN: A social
460 spatio-temporal graph convolutional neural network for human trajectory
prediction, in: Proceedings of the IEEE International Conference on Com-
puter Vision and Pattern Recognition, IEEE, 2020, pp. 14424–14432.
- [24] B. Zhang, C. Yuan, T. Wang, H. Liu, STENet: A hybrid spatio-temporal
465 embedding network for human trajectory forecasting, Engineering Appli-
cations of Artificial Intelligence 106 (2021) 104487.
- [25] T. Karras, S. Laine, T. Aila, A style-based generator architecture for gen-
erative adversarial networks, in: Proceedings of the IEEE International
Conference on Computer Vision and Pattern Recognition, IEEE, 2019, pp.
4396–4405.
- [26] A. V. den Oord, N. Kalchbrenner, O. Vinyals, L. Espeholt, A. Graves,
470 K. Kavukcuoglu, Conditional image generation with PixelCNN decoders,
in: Proceedings of Advances in Neural Information Processing Systems,
2016, pp. 4790–4798.
- [27] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves,
475 N. Kalchbrenner, A. Senior, K. Kavukcuoglu, WaveNet: A generative mod-
el for raw audio (2016). [arXiv:1609.03499](https://arxiv.org/abs/1609.03499).
- [28] L. Yu, W. Zhang, J. Wang, Y. Yu, SeqGAN: sequence generative adversarial
nets with policy gradient, in: Proceedings of AAAI Conference on Artificial
Intelligence, 2017, pp. 2852–2858.
- [29] O. Makansi, E. Ilg, O. Cicek, T. Brox, Overcoming limitations of mixture
480 density networks: A sampling and fitting framework for multimodal fu-
ture prediction, in: Proceedings of the IEEE International Conference on
Computer Vision, IEEE, 2019, pp. 7144–7153.
- [30] J. Li, F. Yang, M. Tomizuka, C. Choi, Evolvegraph: Multi-agent trajectory
485 prediction with dynamic relational reasoning (2020). [arXiv:2003.13924](https://arxiv.org/abs/2003.13924).

- [31] N. Rhinehart, K. Kitani, P. Vernaza, R2P2: a reparameterized pushforward policy for diverse, precise generative path forecasting, in: Proceedings of the European Conference on Computer Vision, Springer, 2018, pp. 772–788.
- 490 [32] A. Geiger, P. Lenz, R. Urtasun, Are we ready for autonomous driving? the KITTI vision benchmark suite, in: Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition, 2012, pp. 2852–2858.
- 495 [33] R. M. N. Rhinehart, K. Kitani, S. Levine, PRECOG: Prediction conditioned on goals in visual multi-agent settings, in: Proceedings of the IEEE International Conference on Computer Vision, IEEE, 2019, pp. 2821–2830.
- [34] Y. Yuan, X. Weng, Y. Ou, K. M. Kitani, AgentFormer: Agent-aware transformers for socio-temporal multi-agent forecasting, in: Proceedings of the IEEE International Conference on Computer Vision, 2021, pp. 9813–9823.
- 500 [35] H. Song, D. Luan, W. Ding, M. Y. Wang, Q. Chen, Learning to predict vehicle trajectories with model-based planning, in: Proceedings of the 5th Conference on Robot Learning, PMLR, 2022, pp. 1035–1045.
- 505 [36] T. Salzmann, B. Ivanovic, P. Chakravarty, M. Pavone, Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data, in: Proceedings of the European Conference on Computer Vision, Springer, 2020, pp. 683–700.
- [37] R. Liang, Y. Li, X. Li, Y. Tang, J. Zhou, W. Zou, Temporal pyramid network for pedestrian trajectory prediction with multi-supervision, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2021, pp. 2029–2037.
- 510 [38] S. Pellegrini, A. Ess, K. Schindler, L. V. Gool, You’ll never walk alone: Modeling social behavior for multi-target tracking, in: Proceedings of the IEEE International Conference on Computer Vision, IEEE, 2009, pp. 261–268.

- [39] A. Lerner, Y. Chrysanthou, D. Lischinski, Crowds by example, Computer Graphics Forum 26 (2007) 655–664.
- [40] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: Proceedings of the International Conference on Learning Representations, 2015.
- [41] M. Mirza, S. Osindero, Conditional generative adversarial nets (2014). **arXiv:1411.1784**.
- [42] K. Sohn, H. Lee, X. Yan, Learning structured output representation using deep conditional generative models, in: Proceedings of Advances in Neural Information Processing Systems, 2015, pp. 3483–3491.
- [43] C. Scholler, V. Aravantinos, F. Lay, A. Knoll, What the constant velocity model can teach us about pedestrian motion prediction, IEEE Robotics and Automation Letters 5 (2) (2020) 1696–1703.
- [44] R. E. Kalman, A new approach to linear filtering and prediction problems, Journal of Basic Engineering 82 (1) (1960) 35–45.
- [45] H. Zhao, J. Gao, T. Lan, C. Sun, B. Sapp, B. Varadarajan, Y. Shen, Y. Shen, Y. Chai, C. Schmid, C. Li, D. Anguelov, TNT: Target-driven trajectory prediction (2020). **arXiv:2008.08294**.