Stepwise Goal-Driven Networks for Trajectory Prediction

Chuhua Wang* Yuchen Wang* Mingze Xu† David J. Crandall
Luddy School of Informatics, Computing, and Engineering
Indiana University, Bloomington, IN 47408
{cw234,wang617,mx6,djcran}@indiana.edu

Abstract

We propose to predict the future trajectories of observed agents (e.g., pedestrians or vehicles) by estimating and using their goals at multiple time scales. We argue that the goal of a moving agent may change over time, and modeling goals continuously provides more accurate and detailed information for future trajectory estimation. In this paper, we present a novel recurrent network for trajectory prediction, called Stepwise Goal-Driven Network (SGNet). Unlike prior work that models only a single, long-term goal, SGNet estimates and uses goals at multiple temporal scales. In particular, the framework incorporates an encoder module that captures historical information, a stepwise goal estimator that predicts successive goals into the future, and a decoder module that predicts future trajectory. We evaluate our model on three first-person traffic datasets (HEV-I, JAAD, and PIE) as well as on two bird's eye view datasets (ETH and UCY), and show that our model outperforms the state-of-the-art methods in terms of both average and final displacement errors on all datasets. Code has been made available at: https://github.com/ChuhuaW/SGNet.pytorch.

1. Introduction

Predicting the future behavior of other agents is crucial in the real world [33]: safe driving requires predicting future movements of other cars and pedestrians, for example, while effective social interactions require anticipating the actions of social partners. However, predicting another agent's future actions is challenging because it depends on numerous factors including the environment and the agent's internal state (*e.g.*, its intentions and goals). Recent work [12, 28, 32, 43, 48, 50] has explored goal-driven methods for future trajectory prediction, which explicitly estimate the goal state (*e.g.*, destination) of an object to help guide future trajectory estimation.

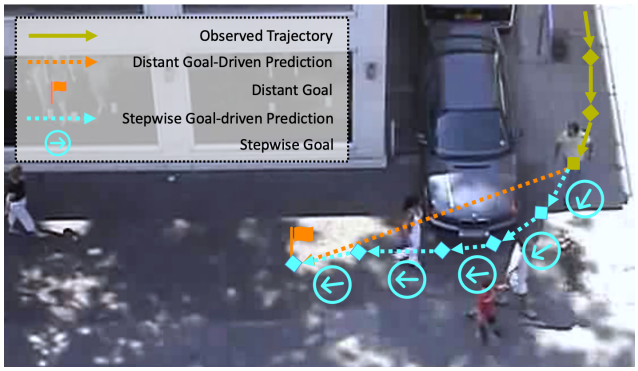

Figure 1: Comparison between SGNet and other goal-driven methods for trajectory prediction. Unlike prior work that estimates only a distant goal to guide the prediction (dotted orange line), our approach estimates multiple goals to better model the future context (dotted cyan line).

Although these recent models make an important step forward, they make the simplistic assumption that agents' trajectories are based only on a long-term goal that they are working towards. However, work in psychology and cognitive science shows that people base their actions not on a single long-term goal, but instead on series of goals at different time scales [10, 14, 37, 38]. For example, a pedestrian wishing to cross the street needs to solve multiple smaller problems along the way: finding the crosswalk, waiting for cars to stop, avoiding other pedestrians, etc.

Based on this intuition from psychology, in this paper, we propose new techniques for future trajectory prediction. These techniques are based on three main hypotheses: (1) estimating the future at multiple time spans, even on a coarse level, provides guidance in decoding the trajectory (as shown in Figure 1); (2) inspired by [41], these future goals can be used as supplementary input to aid in disentangling and encoding the current and past information, resulting in more discriminative representations that implicitly model the intention of the agent; and (3) inaccu-

*Equal contribution
†Currently at Amazon/AWS AI

rate future goals may mislead the prediction, so adaptively learned weights can help the model to distinguish the importance of each stepwise goal.

To validate these hypotheses, we present a new recurrent encoder-decoder architecture, Stepwise Goal-Driven Network (SGNet), to address trajectory prediction. SGNet consists of three main components: (1) A stepwise goal estimator that predicts coarse successive goals into the future. Unlike prior work that models only a single, long-term goal, we use a lightweight module to estimate sequences of goals at multiple temporal scales. Attention mechanism is employed to enhance the important goal features. (2) An encoder module that captures historical information along with predicted stepwise goals, to embed a better hidden representation, and better understand what will happen in the future. (3) A decoder module that takes advantage of goal information to predict future trajectories. SGNet can also incorporate a conditional variational autoencoder (CVAE) module that learns future trajectory distributions conditioned on the observed trajectories through a stochastic latent variable, and outputs multiple proposals. We evaluate our model on multiple first-person and third-person datasets, including both vehicles and pedestrians, and compare to an extensive range of existing work ranging from deterministic to stochastic approaches. We achieve the state-of-the-art performance on all benchmarks.

The contributions of this paper are three-fold. First, our work highlights a novel direction for goal-driven trajectory prediction by using predicted stepwise goals. Second, we show how to effectively incorporate these goals into both encoder and decoder with attention mechanism. Our stepwise goal estimator is a very general module that can be used in numerous scenarios. Finally, our method achieves the state-of-the-art results on multiple benchmarks, including both first- and third-person viewpoints and different agents (*i.e.*, car and pedestrians).

2. Related Work

Trajectory prediction from first-person views simultaneously models the motion of the observed object and ego-camera. Bhattacharyya *et al.* [3] propose the Bayesian LSTMs to model observation uncertainty and predict the distribution of future locations. Yagi *et al.* [42] use multi-modal data, such as human pose, scale, and ego-motion, as cues in a convolution-deconvolution (Conv1D) framework to predict future pedestrian locations. Yao *et al.* [44] introduce a multi-stream encoder-decoder that separately captures both object location and appearance. Makansi *et al.* [27] estimate a reachability prior for objects from the semantic map and propagate them into the future.

Trajectory prediction from a bird's eye view simplifies the problem by removing ego-motion. Alahi *et al.* [1] propose Social-LSTM to model pedestrians' trajectories and interactions. Their social pooling module is improved by [13] to capture global context. SoPhie [34] applies generative models to model the uncertainty of future paths. Lee *et al.* [21] use RNNs with conditional variational autoencoders (CVAEs) to generate multi-modal predictions. Recent work [16, 17, 19, 39] also proposes graph-based recurrent models, simultaneously predicting potential trajectories of multiple objects, while [35] exploits more dynamic and heterogeneous inputs. PLOP [6] and Argoverse [7] use the ego trajectory in a bird's eye view map. Simaug and SMARTS [25, 52] take advantage of simulation data to train the prediction model. Others [5, 8, 11, 24, 26, 49] have explored multimodal inputs, such as Lidar [9, 31, 36, 47], to aid in trajectory prediction.

Goal-driven trajectory prediction incorporates estimated future goals. Rhinehart *et al.* [32] anticipate multi-modal semantic actions as the goal and conduct conditional forecasting using imitative models. Deo *et al.* [12] estimate goal states and fuse the results with past trajectories using maximum entropy inverse reinforcement learning. PECNet [28] infers distant destinations to improve long-range trajectory prediction. TNT [50] decomposes the prediction task into three stages: predicting potential target states, generating trajectory state sequences, and estimating trajectory likelihoods. BiTraP [43] uses a bi-directional decoder on the predicted goal to improve long-term trajectory prediction.

In contrast to the above methods that only estimate the final goal (*i.e.*, the destination), our technique estimates goals at multiple temporal scales. Our extensive experiments show that this idea significantly improves the state-of-the-art in future trajectory prediction.

3. Stepwise Goal-Driven Network (SGNet)

At time step t, given an object's observed trajectory in the last ℓ_e steps, $\mathbf{X}_t = \{\mathbf{x}_{t-\ell_e+1}, \mathbf{x}_{t-\ell_e+2}, \ldots, \mathbf{x}_t\}$, where $\mathbf{x}_t$ includes its bounding box position (*i.e.*, centroid position, width, and height) and motion properties (*e.g.*, optical flow, velocity, and acceleration), our goal is to predict its future positions $\mathbf{Y}_t = \{\mathbf{y}_{t+1}, \mathbf{y}_{t+2}, \ldots, \mathbf{y}_{t+\ell_d}\}$ in the next ℓ_d frames. Unlike state-of-the-art methods [28, 43] that estimate and use only an object's distant target, we hypothesize that explicitly incorporating both long- and short-term goals at multiple time steps provides more comprehensive contextual information. We propose a new recurrent encoder-decoder architecture, Stepwise Goal-Driven Network (SGNet), which predicts stepwise goals to provide guidance during trajectory prediction as well as supplementary features to help capture historical information.

Fig. 2 presents an overview of SGNet. In particular, the stepwise goal estimator (SGE) predicts an object's future locations from $t+1$ to $t+\ell_d$ as stepwise goals, and embeds

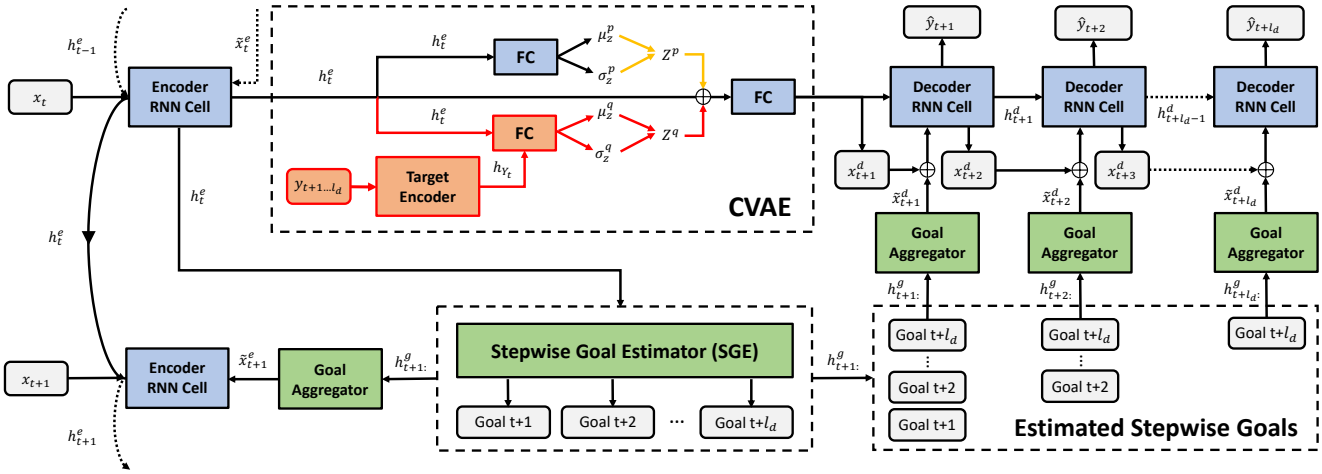

Figure 2: Visualization of SGNet. Arrows in red, yellow, and **black** indicate connections during training, inference, and both training and inference, respectively. To obtain the deterministic results, we replace the CVAE with a non-linear embedding.

them as input to the decoder in an incremental manner to ensure trajectory prediction is guided without receiving any redundant information. SGE also fuses and feeds all predicted goals into the encoder for the next time step, which we believe could help encode a better representation with intention-guided past and current information. In general, SGNet decomposes trajectory prediction into three stages: encoder, SGE, and decoder (summarized in Alg. 1).

3.1. Encoder

The encoder captures an agent's movement behavior as a latent vector by embedding its historical trajectory $\mathbf{X}_t$ using a single fully-connected layer. If additional motion features (*e.g.*, optical flow) are available, they are also embedded using a separate fully-connected layer and concatenated with the trajectory representations,

$$\mathbf{x}_t^e = ReLU(\mathbf{W}_\alpha^T \mathbf{X}_t + \mathbf{b}_\alpha). \tag{1}$$

The input feature $\mathbf{x}_t^e$ is then concatenated with the aggregated goal information $\widetilde{\mathbf{x}}_t^e$ from the previous time step $t-1$, and then the new hidden state h_t^e is updated through a recurrent cell,

$$h_{t+1}^e = \text{RNNCell}_e([\mathbf{x}_{t+1}^e, \widetilde{\mathbf{x}}_{t+1}^e], h_t^e), \tag{2}$$

where the brackets indicate concatenation. h_{t+1}^e and $\widetilde{\mathbf{x}}_{t+1}^e$ are set to zero for the first time step. We next discuss how to obtain the aggregated goal input.

3.2. Stepwise Goal Estimator (SGE)

The main idea of SGE is to generate coarse stepwise future goals to assist the trajectory prediction in a coarse-to-fine manner. These goals also help the network better understand what probably will happen next. Consequently, we design SGE to predict and convey the predicted coarse stepwise goals to both encoder and decoder. For encoder, at each time step t, a set of stepwise goals from $t-1$ is concatenated with the input to serve as supplementary features, helping the encoder learn a more discriminative representation of the sequence. Since inaccurate future goals may mislead the prediction, we use a goal aggregator that adaptively learns the importance of each stepwise goal by using an attention mechanism. Meanwhile, for decoder, at each time step $t+i$ ($i \in [1, \ell_d]$), a subset of stepwise goals $h_{t+i:}$ serves as coarse guidance to help trajectory prediction, and a goal aggregator again gathers and weights the selected goals.

We define a generic SGE module f_{SGE} as,

$$h_{t+1:}^g = f_{SGE}(ReLU(\mathbf{W}_\gamma^T h_t^e + \mathbf{b}_\gamma)) \tag{3}$$

where encoder hidden state h_t^e is the input to SGE at time t. Then SGE module outputs a sequence of stepwise goals, $h_{t+1:}^g = \{h_{t+1}^g, h_{t+2}^g, \ldots, h_{t+\ell_d}^g\}$. We found that SGE is open to different implementations including recurrent, convolution, and fully-connected layers, as will be described in Sec. 4.4. To regularize SGE to generate goals that contain precise future information, we regress the goal position $\widehat{\mathbf{Y}}_t^g$ using the same regressor defined in Sec. 3.4 to minimize the distance between goal position and the ground truth.

Goal Aggregator for Encoder. We hypothesize that anticipated goals can serve as supplementary features to help the encoder learn a more discriminative representation, and that each individual goal has different impact on the prediction. We thus define an attention function f_{attn}^e that takes stepwise goals $h_{t+1:}^g = \{h_{t+1}^g, h_{t+2}^g, ..., h_{t+\ell_d}^g\}$, calculates corresponding weights over estimated goals, and outputs $\widetilde{\mathbf{x}}_{t+1}^e$ as additional input for the encoder,

$$\widetilde{\mathbf{x}}^e_{t+1} = f^e_{attn}(h^g_{t+1:}) = \sum_{s=t+1}^{\ell_d} w^e_s h^g_s \tag{4}$$

$$w^e = \text{Softmax}(\mathbf{W}^T_\zeta \text{Tanh}(h^g_{t+1:}) + \mathbf{b}_\zeta), \tag{5}$$

where w^e is an attention vector corresponding to the probability distribution over ℓ_d estimated goals, and w^e_s is the weight for each individual goal h^g_s.

Goal Aggregator for Decoder. Stepwise goals are fed into the decoder at each time step in an incremental manner. For example, at time step $t+i$, the decoder receives goals between time step $t+i$ and $t+\ell_d$, and the goals before $t+i$ are ignored, because these are redundant and have been encoded in the historical information. Similar to how we aggregating goals for encoder, we implement an attention network to learn the weights of each subset of goals adaptively. The decoder input $\widetilde{\mathbf{x}}^d_{t+i}$ is defined as,

$$\widetilde{\mathbf{x}}^d_{t+i} = f^d_{attn}(h^g_{t+i:}) = \sum_{s=t+i}^{\ell_d} w^d_s h^g_s \tag{6}$$

$$w^d = \text{Softmax}(\mathbf{W}^T_\eta \text{Tanh}(h^g_{t+i:}) + \mathbf{b}_\eta), \tag{7}$$

where w^d is an attention vector corresponding to the probability distribution over $\ell_d - i + 1$ estimated goals, and w^d_s is the weight for each individual goal h^g_s.

3.3. Conditional Variational Autoencoder (CVAE)

SGNet can also include a Conditional Variational Autoencoder (CVAE) module to learn and predict the distribution of future trajectory $\mathbf{Y}_t$ conditioned on the observed trajectory $\mathbf{X}_t$, by introducing a Gaussian latent variable $z \sim N(\mu_z, \sigma_z)$. Following most prior work [3, 28, 43], our CVAE consists of three main components: recognition network $Q_\phi(z|\mathbf{X}_t, \mathbf{Y}_t)$, prior network $P_\nu(z|\mathbf{X}_t)$, and generation network $P_\theta(h^d_t|\mathbf{X}_t, z)$, where ϕ, ν, θ denote the parameter of these three networks, and h^d_t is the trajectory encoded by the generation network that will be fed into the decoder for trajectory prediction. In particular, recognition, prior, and generation networks are all realized by fully-connected layers.

During training, the ground truth future trajectory $\mathbf{Y}_t$ is fed into the target encoder to output the hidden state h_{Y_t}. To capture the dependencies between observed and ground truth trajectories, the recognition network takes encoder hidden states h^e_t and h_{Y_t} and predicts the distribution mean μ^q_z and standard deviation σ^q_z. The prior network takes h^e_t only and predicts μ^p_z and σ^p_z. Kullback–Leibler divergence (KLD) loss is used to regularize the distribution between $N(\mu^q_z, \sigma^q_z)$ and $N(\mu^p_z, \sigma^p_z)$. We sample z from $N(\mu^q_z, \sigma^q_z)$ and concatenate it with h^e_t to generate h^d_t with the generation network. **During testing**, the ground truth future trajectory $\mathbf{Y}_t$ is not available, so we directly sample z from $N(\mu^p_z, \sigma^p_z)$. The generation network concatenates z and h^e_t and then outputs h^d_t.

Algorithm 1 Workflow of SGNet

Input: Concatenated input feature $\mathbf{x}^e_t$ and predicted goal $\widetilde{\mathbf{x}}^e_t$, hidden state h^e_{t-1}

Output: Future trajectory $\widehat{\mathbf{Y}}_t = \{\widehat{\mathbf{y}}_{t+1}, \widehat{\mathbf{y}}_{t+2}, \ldots, \widehat{\mathbf{y}}_{t+\ell_d}\}$, hidden state h^e_t, and predicted goal $\widetilde{\mathbf{x}}^e_{t+1}$

```
for each encoder step t do
    Update h^e_t with (x^e_t, x̃^e_t) and h^e_{t-1}
    Compute h^g_t from h^e_t by linear embedding
    Construct goal states vector h^g_{t+1:}
    CVAE takes h^e_t and sample K proposals of h^d_t
    for each decoder step i do
        for each proposal k do
            Compute x̃^d_{t+i} from h^g_{t+i:} as in (6)
            Compute x^d_{t+i} from h^d_{t+i} as in (9)
            Update decoder states h^d_{t+i} as in Eq. (8)
            Output trajectory prediction ŷ_{t+i} from h^d_{t+i}
        end for
    end for
    Compute x̃^e_{t+1} from h^g_{t+1:} as in (4)
    Gather h^g_{t:} and concatenate x̃^e_{t+1} with x^e_{t+1}, and
  feed into the encoder at next time step t + 1
end for
return Ŷ_t
```

3.4. Decoder

Our recurrent decoder outputs the final trajectory $\mathbf{Y}_t = \{\mathbf{y}_{t+1}, \mathbf{y}_{t+2}, \cdots, \mathbf{y}_{t+\ell_d}\}$ with a trajectory regressor. Given h^d_t and the estimated goal input $\widetilde{\mathbf{x}}^d_{t+1}$, it produces a new hidden state for the next time step,

$$h^d_{t+1} = \text{RNNCell}_d([\mathbf{x}^d_{t+1}, \widetilde{\mathbf{x}}^d_{t+1}], h^d_t) \tag{8}$$

$$\mathbf{x}^d_{t+1} = ReLU(\mathbf{W}^T_\psi h^d_t + \mathbf{b}_\psi). \tag{9}$$

Our trajectory regressor is a single fully-connected layer that takes hidden states h^d_{t+i} and computes a trajectory $\widehat{\mathbf{y}}_{t+i}$ at each time step,

$$\widehat{\mathbf{y}}_{t+i} = ReLU(\mathbf{W}^T_\chi h^d_{t+i} + \mathbf{b}_\chi). \tag{10}$$

3.5. Loss Functions

We train our model in an end-to-end manner. We choose to use the Root Mean Square Error (RMSE) as loss function to supervise trajectory prediction from our decoder. For our stochastic model using CAVE, we follow [3, 43] and adapt best-of-many (BoM) approach to minimize the distance between our best prediction $\widehat{\mathbf{Y}}_t$ and target $\mathbf{Y}_t$. This approach leads to more accurate and diverse predictions and encourages the model to capture the true variation in data. To

Method	JAAD			PIE		
	MSE ↓ (0.5s/1.0s/1.5s)	C_{MSE} ↓ (1.5s)	CF_{MSE} ↓ (1.5s)	MSE ↓ (0.5s/1.0s/1.5s)	C_{MSE} ↓ (1.5s)	CF_{MSE} ↓ (1.5s)
SGNet-ED-MLP	84 / 338 / 1098	1032	4376	39/153/496	464	1967
SGNet-ED-CNN	88 / 344 / 1079	1016	4170	37/145/470	441	1859
SGNet-ED-GRU	**82 / 328 / 1049**	**996**	**4076**	**34/133/442**	**413**	**1761**

Table 1: Explore different instantiations of SGE on JAAD and PIE. SGNet-ED-GRU will be used our best model in the following sections. ↓ denotes lower is better.

Method	ADE ↓ (0.5s / 1.0s / 1.5s)	FDE ↓ (1.5s)	FIOU ↑ (1.5s)
FOL-X [45]	6.70 / 12.60 / 20.40	44.10	0.61
SGNet-E	6.88 / 12.72 / 20.93	46.69	0.60
SGNet-D	6.43 / 12.04 / 19.49	42.56	0.62
SGNet-ED	**6.28 / 11.35 / 18.27**	**39.86**	**0.63**

Table 2: Deterministic results on HEV-I. ↑ denotes higher is better, and ↓ denotes lower is better.

ensure SGE to predict accurate goal states, we also optimize the prediction from SGE using RMSE between goal prediction $\widehat{\mathbf{Y}}_t^g$ and ground truth $\mathbf{Y}_t$. Finally, we add KL-divergence loss (KLD) to optimize the prior network in the CVAE. Thus, for each training sample, our final loss is summarized as follows,

$$\begin{aligned}\mathcal{L}_{total} = \min_{\forall k \in K} \text{RMSE}(\widehat{\mathbf{Y}}_t^k, \mathbf{Y}_t) + \text{RMSE}(\widehat{\mathbf{Y}}_t^g, \mathbf{Y}_t) \\ + \text{KLD}(Q_\phi(z|\mathbf{X}_t, \mathbf{Y}_t), P_\nu(z|\mathbf{X}_t)),\end{aligned} \tag{11}$$

where $\widetilde{\mathbf{Y}}_t^k$ is the k-th trajectory hypothesis from CVAE, $\widetilde{\mathbf{Y}}_t^g$ is the predicted stepwise goal location, and $\mathbf{Y}_t$ is the object's ground truth location at time t.

4. Experiments

We evaluated SGNet against multiple state-of-the-art methods on three first-person datasets (Joint Attention for Autonomous Driving (JAAD) [20], Pedestrian Intention Estimation (PIE) [30], and Honda Egocentric View Intersection (HEV-I) [45]) and two bird's eye view datasets (ETH [29] and UCY [22]). These diverse datasets with different viewpoints (first- and third-person) and agent types (pedestrians and vehicles) challenge our model's robustness in complex, dynamic situations.

4.1. Datasets

JAAD and PIE include egocentric videos of pedestrians in traffic scenes and dynamic conditions. JAAD contains 2,800 trajectories, while PIE's includes 1,835 trajectories. All videos are captured 1920 × 1080 pixels at 30fps. We follow [30] and sample tracks with an overlap ratio of 0.5, discard tracks with length below 2 sec, and use 0.5 sec of observed data as input to predict future trajectories of length 0.5, 1.0, and 1.5 sec.

HEV-I contains 230 first-person videos of vehicles at 10 fps. Following [45], we split the data into 40,000 train (70%) and 17,000 test (30%) samples. HEV-I does not include human annotations, so we follow [45] and use Mask-RCNN [15] and Sort [2] to generate "ground truth" trajectories. We use 1.6 sec of observed data as input to predict future trajectories of length 0.5, 1.0, and 1.5 sec.

ETH and UCY focus on pedestrians with multi-human interaction scenarios from a bird's eye view, and contain diverse behaviors such as couples walking together, groups passing by each other, and groups forming and dispersing. The dataset is captured at 25 fps and annotated at 2.5 fps, and includes 1536 pedestrians in 5 sets of data with 4 unique scenes. Following prior work [35], we use a leave-one-out strategy to split the train and test partitions. We use 3.2 sec as observed input data to predict 4.8 sec future trajectories.

4.2. Implementation Details

We implement our networks in PyTorch, and perform all experiments with Nvidia Titan Xp Pascal GPUs. We use Gated Recurrent Units (GRUs) as the backbone for both encoder and decoder with 512 hidden size. The length of the observation ℓ_e is determined by the default setting of each benchmark: ℓ_e is 16 on HEV-I, 15 on JAAD, 15 on PIE, and 8 on ETH-UCY. Object bounding boxes are taken as inputs for JAAD and PIE, and following [44], optical flow is also included on HEV-I. We follow [35] to use object centroid, velocities, and accelerations as inputs for ETH and UCY. We use the Adam [18] optimizer with default parameters and initial learning rate 5×10^{-4}, which is dynamically reduced based on the validation loss. Our models are optimized end-to-end with batch size 128 and the training is terminated after 50 epochs.

4.3. Evaluation Protocols

Our main evaluation metrics include **average displacement error (ADE)**, which measures accuracy along the whole trajectory, and **final displacement error (FDE)**, which measures accuracy only at the trajectory end point. We use **mean squared error (MSE)** to evaluate our per-

Method	JAAD			PIE		
	MSE ↓ (0.5s/1.0s/1.5s)	C_{MSE} ↓ (1.5s)	CF_{MSE} ↓ (1.5s)	MSE ↓ (0.5s/1.0s/1.5s)	C_{MSE} ↓ (1.5s)	CF_{MSE} ↓ (1.5s)
LSTM [30]	289 / 569 / 1558	1473	5766	173 / 330 / 911	837	3352
Bayesian-LSTM [3]	159 / 539 / 1535	1447	5615	101 / 296 / 855	811	3259
FOL-X [45]	147 / 484 / 1374	1290	4924	47 / 183 / 584	546	2303
PIE_{traj} [30]	110 / 399 / 1280	1183	4780	58 / 200 / 636	596	2477
BiTraP-D [43]	93 / 378 / 1206	1105	4565	41 / 161 / 511	481	1949
SGNet-E	90 / 372 / 1176	1117	4497	40 / 154 / 496	464	1939
SGNet-D	87 / 350 / 1121	1065	4355	37 / 148 / 478	450	1891
SGNet-ED	**82 / 328 / 1049**	**996**	**4076**	**34 / 133 / 442**	**413**	**1761**

Table 3: Deterministic results on JAAD and PIE in terms of MSE/C_{MSE}/CF_{MSE}. ↓ denotes lower is better.

Method	ADE (4.8s) ↓ / FDE (4.8s) ↓					
	ETH	HOTEL	UNIV	ZARA1	ZARA2	Avg
LSTM [35]	1.09 / 2.41	0.86 / 1.91	0.61 / 1.31	0.41 / 0.88	0.52 / 1.11	0.70 / 1.52
Social-LSTM [1]	1.09 / 2.35	0.79 / 1.76	0.67 / 1.40	0.47 / 1.00	0.56 / 1.17	0.72 / 1.54
Social-GAN [13]	1.13 / 2.21	1.01 / 2.18	0.60 / 1.28	0.42 / 0.91	0.52 / 1.11	0.74 / 1.54
MATF [51]	1.33 / 2.49	0.51 / 0.95	0.56 / 1.19	0.44 / 0.93	0.34 / 0.73	0.64 / 1.26
FvTraj [4]	0.62 / 1.23	0.53 / 1.10	0.57 / 1.19	0.42 / 0.89	0.38 / 0.79	0.50 / 1.04
STAR-D [46]	**0.56 / 1.11**	0.26 / 0.50	0.52 / 1.15	0.41 / 0.90	0.31 / 0.71	0.41 / 0.87
Trajectron++ [35]	0.71 / 1.68	**0.22 / 0.46**	0.41 / 1.07	0.30 / 0.77	0.23 / 0.59	0.37 / 0.91
SGNet-E	0.74 / 1.71	0.32 / 0.75	0.44 / 1.05	0.29 / 0.72	0.23 / 0.56	0.40 / 0.96
SGNet-D	0.76 / 1.68	0.30 / 0.70	0.43 / 1.03	0.28 / 0.70	0.22 / 0.55	0.40 / 0.93
SGNet-ED	0.63 / 1.38	0.27 / 0.63	**0.40 / 0.96**	**0.26 / 0.64**	**0.21 / 0.53**	**0.35 / 0.83**

Table 4: Deterministic results on ETH and UCY in terms of ADE/FDE. ↓ denotes the lower the better.

formance on JAAD and PIE, calculated according to the upper-left and lower-right coordinates of the bounding box. We also adopt **Center mean squared error (C_{MSE})** and **center final mean squared error (CF_{MSE})** as evaluation metrics on JAAD and PIE. These two error metrics are similar to MSE except they are computed based on the bounding box centroids. To fairly compare with prior work on HEV-I dataset, we use the upper-left coordinates of bounding boxes to calculate ADE and FDE. In the bird's eye view ETH and UCY datasets, we follow prior work to use the centroid points to calculate ADE and FDE. To compare to the state-of-the-art method [45] in HEV-I, we also use **final intersection over union (FIOU)**, the overlap between the predicted bounding box and ground truth at the final step, which measures the model's ability to predict both the scale and location of bounding boxes in the long-term. All results metrics used for HEV-I, JAAD, and PIE dataset are in pixels, while for ETH and UCY we compute the ADE and FDE in Euclidean space.

4.4. Exploration Study

We begin with experiments for design trade-offs and training strategies of our architecture with JAAD and PIE.

How to implement SGE? We considered three instantiations of f_{attn} in the SGE module. **First,** we implement SGE with GRUs (named SGNet-ED-GRU). The hidden size is set to 128, since we only need a lightweight module to produce a coarse goal prediction. h^g_{t+i} is defined to be the hidden state at time $t+i$, which is initialized by using the encoder hidden state h^e_t after a linear transformation. The GRU input $\mathbf{x}^g_{t+1}$ is initialized with a zero vector and updated using the auto-regressive strategy, and the sequence of the output hidden state at each time step is used as the stepwise goals. **Second,** we try SGE with a multilayer perceptron (MLP) that takes the encoder hidden states h^e_t as input, and directly outputs ℓ_d goals of size 128. We call this SGNet-ED-MLP. **Third,** we follow [42] and implement a convolution-deconvolution framework for SGE to predict the stepwise goals, called SGNet-ED-CNN.

As shown in Table 1, all of the above variants achieve the state-of-the-art results, indicating that the SGE module is not sensitive to the choice of f_{attn}. The results also suggest that using temporal models, such as GRU, as SGE is more effective and robust in future trajectory prediction. Since SGNet-ED-GRU achieves the best performance, we use this version in the remaining experiments (named SGNet-ED).

What if decoder does not receive predicted goals? To evaluate the importance of using predicted goals in the de-

Method	JAAD			PIE		
	MSE ↓ (0.5s/1.0s/1.5s)	C_{MSE} ↓ (1.5s)	CF_{MSE} ↓ (1.5s)	MSE ↓ (0.5s/1.0s/1.5s)	C_{MSE} ↓ (1.5s)	CF_{MSE} ↓ (1.5s)
BiTrap-GMM [43]	153/250/585	501	998	38/90/209	171	368
BiTrap-NP [43]	38/94/222	177	565	23/48/102	81	261
SGNet-ED (20)	**37/86/197**	**146**	**443**	**16/39/88**	**66**	**206**

Table 5: Stochastic results on JAAD and PIE in terms of MSE/C_{MSE}/CF_{MSE}. ↓ denotes lower is better.

Method	ADE (4.8s) ↓ / FDE (4.8s) ↓					
	ETH	HOTEL	UNIV	ZARA1	ZARA2	Avg
Social-GAN [13]	0.81 / 1.52	0.72 / 1.61	0.60 / 1.26	0.34 / 0.69	0.42 / 0.84	0.58 / 1.18
Sophie [34]	0.70 / 1.43	0.76 / 1.67	0.54 / 1.24	0.30 / 0.63	0.38 / 0.78	0.54 / 1.15
CGNS [23]	0.62 / 1.40	0.70 / 0.93	0.48 / 1.22	0.32 / 0.59	0.35 / 0.71	0.49 / 0.97
MATF GAN [51]	1.01 / 1.75	0.43 / 0.80	0.44 / 0.91	0.26 / 0.45	0.26 / 0.57	0.48 / 0.90
FvTraj [4]	0.56 / 1.14	0.28 / 0.55	0.52 / 1.12	0.37 / 0.78	0.32 / 0.68	0.41 / 0.85
DSCMP [40]	0.66 / 1.21	0.27 / 0.46	0.50 / 1.07	0.33 / 0.68	0.28 / 0.60	0.41 / 0.80
PECNet [28]	0.54 / 0.87	0.18 / 0.24	0.35 / 0.60	0.22 / 0.39	0.17 / 0.30	0.29 / 0.48
STAR [46]	0.36 / **0.65**	0.17 / 0.36	0.31 / 0.62	0.26 / 0.55	0.22 / 0.46	0.26 / 0.53
Trajectron++ [35]	0.43 / 0.86	**0.12 / 0.19**	0.22 / 0.43	0.17 / 0.32	0.12 / 0.25	0.21 / 0.41
BiTrap-GMM [43]	0.40 / 0.74	0.13 / 0.22	0.19 / 0.40	0.14 / 0.28	0.11 / 0.22	0.19 / 0.37
BiTrap-NP [43]	0.37 / 0.69	**0.12** / 0.21	**0.17 / 0.37**	0.13 / 0.29	**0.10 / 0.21**	**0.18 / 0.35**
SGNet-ED (20)	**0.35 / 0.65**	**0.12** / 0.24	0.20 / 0.42	**0.12 / 0.24**	**0.10 / 0.21**	**0.18 / 0.35**

Table 6: Stochastic results on ETH and UCY in terms of ADE/FDE. ↓ denotes lower is better.

coder, we implement a baseline (SGNet-E) without the connection between SGE and the decoder. Table 3 shows that our best model significantly outperforms this baseline, indicating that using predicted goals in the decoder is important for achieving more accurate results.

What if encoder does not receive predicted goals? To evaluate the importance of using predicted goals in the encoder, we remove the connection between SGE and encoder (SGNet-D). As shown in Table 3, the performance of our best model is also better than this baseline on both datasets. These results confirm the importance of using predicted goals in the encoder.

Does SGE need supervision? To investigate whether supervised stepwise goals offer more accurate information for trajectory prediction, we train our network without the goal module loss, RMSE($\widehat{\mathbf{Y}}_t^g, \mathbf{Y}_t$). This change increases the error of our best model by 13/68/208 and 15/57/181 on JAAD and PIE, respectively, in terms of 0.5s/1.0s/1.5s MSE. This suggests that supervision on SGE leads to better goal representation to help predicting trajectory, and thus using loss to optimize SGE is important in our model.

To conclude, our full model, called SGNet-ED, uses GRU as the backbone for SGE, and the output stepwise goals are fed into both encoder and decoder. Our final loss function includes the SGE loss, BoM trajectory loss, and KLD loss, as shown in Eq. 11.

4.5. Comparison with the State of the Art

In this section, we compare our best model (SGNet-ED) with the state-of-the-art methods under two different settings: deterministic, in which the model returns a single trajectory, and stochastic, in which it generates a set of K possible trajectories and we report the best-performing sample. More specifically, to obtain the deterministic results, we replace the CVAE module with a non-linear embedding to predict a single trajectory. Both results are compared with an extensive set of published baselines.

4.5.1 Deterministic Results

First-person benchmarks. We start by evaluating our model's performance on predicting pedestrian trajectory in two first-person datasets. As shown in Table 3, our best model (SGNet-ED) significantly outperforms the state-of-the art on the first-person pedestrian detection datasets. On JAAD, our model reduces the MSE error by 11, 50, and 157 pixels for 0.5s, 1.0s and 1.5s prediction, respectively, compared to [43]. On PIE, our model reduces MSE error by 7, 28, and 69 for 0.5s, 1.0s, and 1.5s prediction, respectively. As the prediction length increases (and thus the problem becomes harder), comparing to prior work, our improvement is even larger, suggesting that SGE is especially helpful for

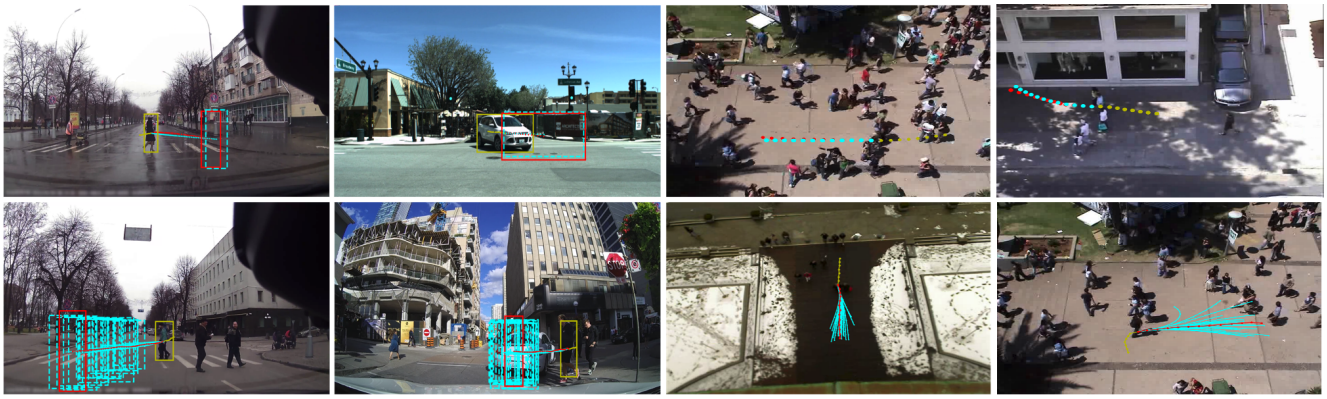

Figure 3: Qualitative results of deterministic (top row) and stochastic (bottom row) prediction on first- and third-person datasets. The yellow color indicates the observed trajectory, the red color indicates the ground truth future trajectory, and cyan color indicates the predictions from our SGNet-ED model (better in color).

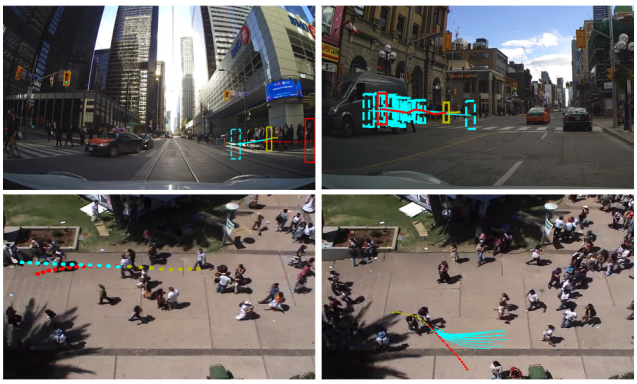

Figure 4: Failure cases of deterministic (left column) and stochastic (right column) predictions on first- and third-person datasets. The yellow color indicates the observed trajectory, the red color indicates the ground truth future trajectory, and cyan color indicates the predictions from our SGNet-ED model (better in color).

long-term prediction. We get similar results by using other evaluation metrics, as shown in the Table 3.

Table 2 presents comparisons between SGNet and prior work [45] on vehicle trajectory prediction in HEV-I. SGNet-ED outperforms this baseline by 0.42, 1.25, and 2.13 in terms of 0.5s, 1.0s, and 1.5s Average Displacement Error (ADE) and 4.24 in terms of Final Displacement Error (FDE). Moreover, the improvement in 1.5s FIOU shows that our model can also accurately predict the scale of objects.

Third-person benchmarks. Table 4 shows that, on average, our model outperforms the state-of-the-art methods by more than 10% in terms of ADE and FDE on the bird's eye view datasets. The results demonstrate that our model predicts persistent and stable future trajectories. As with the first-person datasets, our model leads to a larger improvement as the prediction length increases, suggesting that estimated stepwise goals offer better temporal information.

4.5.2 Stochastic Results

First-person benchmarks. To fairly compare with [43], we generate $K = 20$ possible proposals and report the best-performance sample during evaluation. For the first-person datasets, our method outperforms the state-of-the-art by 14% on average on JAAD and 25% on average on PIE, as shown in Table 5. We do not report results on HEV-I since there is no existing stochastic benchmark to our knowledge.

Third-person benchmarks. For ETH-UCY, we follow the leave-one-out evaluation protocol with $K = 20$ by following the prior work in Table 6. SGNet-ED outperforms the current state-of-the-art stochastic model [43] by 5% on average on ETH, ZARA1 and ZARA2, and achieves comparable results on HOTEL and UNIV. Our model does not explicitly model interaction, but has a relatively strong temporal model, which may explain why our model is better on less complex scenes (ETH, ZARA1 and ZARA2). Including interaction and map features may help our model to improve on crowd scenes.

4.6. Qualitative Results

The first row of Fig. 3 shows four examples of our best model's deterministic predictions. the first example, the ego-vehicle is stopping at the intersection and a pedestrian is walking along the marked crosswalk. In the second example, the shape of the marked vehicles change as they are making turns, making it challenging to predict their size and distance. In the third and fourth examples, we show our model's deterministic prediction result on ETH and UCY.

The second row of Fig. 3 shows four examples of our best model's stochastic predictions. We show all $K = 20$ stochastic trajectories generated by our best model. Most of the predictions are close to the ground truth trajectory and bounding box, demonstrating the stability of our stochastic prediction model.

4.7. Failure Cases

Fig. 4 studies four failure cases of our model. In the top-left example, the vehicle pulls to the right suddenly and drives forward in the scene, but the observed trajectory of the pedestrian is nearly still. Since we only use the bounding boxes to represent the agents' states, it is difficult for our model to predict the agent's relative movement caused by an unexpected and large motion of the ego-vehicle. However, using ego-motion information in our model may help to improve this. In the bottom-left example, the marked person is descending the stairs in the near future, but our model fails to predict the correct trajectory due to the lack of context information and interaction between pedestrians. In the top-right example, most of the stochastic predictions show the pedestrian walking along the road, but one prediction indicates the possibility that the pedestrian may cross the road along the zebra crossing. However, this error may be a "blessing in disguise", for example helping an autonomous driving system to prepare for a potential future risk. In the bottom-right example, our stochastic model fails to predict the future when the target pedestrian interacts with another. We believe taking object interaction into consideration is a good direction for our future work.

5. Conclusion

In this paper, we presented Stepwise Goal-Driven Network (SGNet) to tackle the trajectory prediction problem. Unlike most existing goal-driven models that only estimates final destination or distant goals, SGNet predicts both long- and short-term goals and explicitly incorporates these estimated future states into both encoder and decoder. We show that these goals can help to predict the trajectory as well as to better capture the historical observations. Our Stepwise Goal Estimator (SGE) can be used as a general plugin module, and is robust to various ways to be implemented. We conducted extensive experiments on three first-person and two bird's eye view datasets to evaluate both the effectiveness and robustness of the proposed approach. Experimental results show that our models outperform the state-of-the-art methods. In the future, we plan to include object interaction and environment information to enhance our model, and to consider ego-motion which would help our model in first-person trajectory prediction.

6. Acknowledgment

This work was supported in part by the Indiana University Office of the Vice Provost for Research, the College of Arts and Sciences, and the Luddy School of Informatics, Computing, and Engineering through the Emerging Areas of Research Project Learning: Brains, Machines, and Children. We also would like to thank Yu Yao for valuable discussions and IU Computer Vision Lab for helpful feedback.

The views and conclusions contained in this paper are those of the authors and should not be interpreted as representing the official policies, either expressly or implied, of the U.S. Government, or any sponsor.

References

[1] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social LSTM: Human trajectory prediction in crowded spaces. In *CVPR*, 2016. 2, 6

[2] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *ICIP*, 2016. 5

[3] Apratim Bhattacharyya, Mario Fritz, and Bernt Schiele. Long-term on-board prediction of people in traffic scenes under uncertainty. In *CVPR*, 2018. 2, 4, 6

[4] Huikun Bi, Ruisi Zhang, Tianlu Mao, Zhigang Deng, and Zhaoqi Wang. How can i see my future? fvtraj: Using first-person view for pedestrian trajectory prediction. In *ECCV*, 2020. 6, 7

[5] Bruno Brito, Hai Zhu, Wei Pan, and Javier Alonso-Mora. Social-vrnn: One-shot multi-modal trajectory prediction for interacting pedestrians. *arXiv:2010.09056*, 2020. 2

[6] Thibault Buhet, Emilie Wirbel, and Xavier Perrotton. Plop: Probabilistic polynomial objects trajectory planning for autonomous driving. *arXiv:2003.08744*, 2020. 2

[7] Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, et al. Argoverse: 3d tracking and forecasting with rich maps. In *CVPR*, 2019. 2

[8] Chiho Choi, Joon Hee Choi, Jiachen Li, and Srikanth Malla. Shared cross-modal trajectory prediction for autonomous driving. *CVPR*, 2021. 2

[9] Chiho Choi, Srikanth Malla, Abhishek Patil, and Joon Hee Choi. Drogon: A trajectory prediction model based on intention-conditioned behavior reasoning. *arXiv:1908.00024*, 2019. 2

[10] Jennifer Creek and Lesley Lougher. *Occupational therapy and mental health*. Elsevier Health Sciences, 2011. 1

[11] Nachiket Deo and Mohan M Trivedi. Multi-modal trajectory prediction of surrounding vehicles with maneuver based lstms. In *IV*, 2018. 2

[12] Nachiket Deo and Mohan M Trivedi. Trajectory forecasts in unknown environments conditioned on grid-based plans. *arXiv:2001.00735*, 2020. 1, 2

[13] Agrim Gupta, Justin Johnson, Li Fei-Fei, Silvio Savarese, and Alexandre Alahi. Social GAN: Socially acceptable trajectories with generative adversarial networks. In *CVPR*, 2018. 2, 6, 7

[14] Judith M Harackiewicz, Kenneth E Barron, John M Tauer, Suzanne M Carter, and Andrew J Elliot. Short-term and long-term consequences of achievement goals: Predicting interest and performance over time. *Journal of educational psychology*, 2000. 1

[15] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, 2017. 5

[16] Boris Ivanovic, Amine Elhafsi, Guy Rosman, Adrien Gaidon, and Marco Pavone. Mats: An interpretable trajectory forecasting representation for planning and control. *arXiv:2009.07517*, 2020. 2

[17] Boris Ivanovic and Marco Pavone. The trajectron: Probabilistic multi-agent trajectory modeling with dynamic spatiotemporal graphs. In *ICCV*, 2019. 2

[18] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *ICLR*, 2015. 5

[19] Vineet Kosaraju, Amir Sadeghian, Roberto Martín-Martín, Ian Reid, Hamid Rezatofighi, and Silvio Savarese. Social-bigat: Multimodal trajectory forecasting using bicycle-gan and graph attention networks. In *NeurIPS*, 2019. 2

[20] Iuliia Kotseruba, Amir Rasouli, and John K Tsotsos. Joint attention in autonomous driving (JAAD). *arXiv:1609.04741*, 2016. 5

[21] Namhoon Lee, Wongun Choi, Paul Vernaza, Christopher B Choy, Philip HS Torr, and Manmohan Chandraker. Desire: Distant future prediction in dynamic scenes with interacting agents. In *CVPR*, 2017. 2

[22] Alon Lerner, Yiorgos Chrysanthou, and Dani Lischinski. Crowds by example. In *Computer graphics forum*, 2007. 5

[23] Jiachen Li, Hengbo Ma, and Masayoshi Tomizuka. Conditional generative neural system for probabilistic trajectory prediction. *IROS*, 2019. 7

[24] Xiao Li, Guy Rosman, Igor Gilitschenski, Jonathan DeCastro, Cristian-Ioan Vasile, and Sertac Karaman Daniela Rus. Differentiable logic layer for rule guided trajectory prediction. In *CoRL*, 2020. 2

[25] Junwei Liang, Lu Jiang, and Alexander Hauptmann. Simaug: Learning robust representations from simulation for trajectory prediction. In *ECCV*, 2020. 2

[26] Junwei Liang, Lu Jiang, Juan Carlos Niebles, Alexander G Hauptmann, and Li Fei-Fei. Peeking into the future: Predicting future person activities and locations in videos. In *CVPR*, 2019. 2

[27] Osama Makansi, Ozgun Cicek, Kevin Buchicchio, and Thomas Brox. Multimodal future localization and emergence prediction for objects in egocentric view with a reachability prior. In *CVPR*, 2020. 2

[28] Karttikeya Mangalam, Harshayu Girase, Shreyas Agarwal, Kuan-Hui Lee, Ehsan Adeli, Jitendra Malik, and Adrien Gaidon. It is not the journey but the destination: Endpoint conditioned trajectory prediction. *ECCV*, 2020. 1, 2, 4, 7

[29] Stefano Pellegrini, Andreas Ess, Konrad Schindler, and Luc Van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *ICCV*, 2009. 5

[30] Amir Rasouli, Iuliia Kotseruba, Toni Kunic, and John K. Tsotsos. PIE: A large-scale dataset and models for pedestrian intention estimation and trajectory prediction. In *ICCV*, 2019. 5, 6

[31] Nicholas Rhinehart, Kris M Kitani, and Paul Vernaza. R2p2: A reparameterized pushforward policy for diverse, precise generative path forecasting. In *ECCV*, 2018. 2

[32] Nicholas Rhinehart, Rowan McAllister, Kris Kitani, and Sergey Levine. Precog: Prediction conditioned on goals in visual multi-agent settings. In *ICCV*, 2019. 1, 2

[33] Andrey Rudenko, Luigi Palmieri, Michael Herman, Kris M Kitani, Dariu M Gavrila, and Kai O Arras. Human motion trajectory prediction: A survey. *The International Journal of Robotics Research*, 2020. 1

[34] Amir Sadeghian, Vineet Kosaraju, Ali Sadeghian, Noriaki Hirose, Hamid Rezatofighi, and Silvio Savarese. Sophie: An attentive gan for predicting paths compliant to social and physical constraints. In *CVPR*, 2019. 2, 7

[35] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Multi-agent generative trajectory forecasting with heterogeneous data for control. *ECCV*, 2020. 2, 5, 6, 7

[36] Meet Shah, Zhiling Huang, Ankit Laddha, Matthew Langford, Blake Barber, Sidney Zhang, Carlos Vallespi-Gonzalez, and Raquel Urtasun. Liranet: End-to-end trajectory prediction using spatio-temporal radar fusion. *arXiv:2010.00731*, 2020. 2

[37] Piers Steel and Cornelius J König. Integrating theories of motivation. *Academy of management review*, 2006. 1

[38] Shu Hua Sun and Michael Frese. Multiple goal pursuit. 2013. 1

[39] Yichuan Charlie Tang and Ruslan Salakhutdinov. Multiple futures prediction. *arXiv:1911.00997*, 2019. 2

[40] Chaofan Tao, Qinhong Jiang, Lixin Duan, and Ping Luo. Dynamic and static context-aware lstm for multi-agent motion prediction. *ECCV*, 2020. 7

[41] Mingze Xu, Mingfei Gao, Yi-Ting Chen, Larry S. Davis, and David J. Crandall. Temporal recurrent networks for online action detection. In *ICCV*, 2019. 1

[42] Takuma Yagi, Karttikeya Mangalam, Ryo Yonetani, and Yoichi Sato. Future person localization in first-person videos. In *CVPR*, 2018. 2, 6

[43] Yu Yao, Ella Atkins, Matthew Johnson-Roberson, Ram Vasudevan, and Xiaoxiao Du. Bitrap: Bi-directional pedestrian trajectory prediction with multi-modal goal estimation. *RA-L*, 2021. 1, 2, 4, 6, 7, 8

[44] Yu Yao, Mingze Xu, Chiho Choi, David J Crandall, Ella M Atkins, and Behzad Dariush. Egocentric vision-based future vehicle localization for intelligent driving assistance systems. In *ICRA*, 2019. 2, 5

[45] Yu Yao, Mingze Xu, Yuchen Wang, David J Crandall, and Ella M Atkins. Unsupervised traffic accident detection in first-person videos. In *IROS*, 2019. 5, 6, 8

[46] Cunjun Yu, Xiao Ma, Jiawei Ren, Haiyu Zhao, and Shuai Yi. Spatio-temporal graph transformer networks for pedestrian trajectory prediction. In *ECCV*, 2020. 6, 7

[47] Wenyuan Zeng, Wenjie Luo, Simon Suo, Abbas Sadat, Bin Yang, Sergio Casas, and Raquel Urtasun. End-to-end interpretable neural motion planner. In *CVPR*, 2019. 2

[48] Lingyao Zhang, Po-Hsun Su, Jerrick Hoang, Galen Clark Haynes, and Micol Marchetti-Bowick. Map-adaptive goal-based trajectory prediction. *arXiv:2009.04450*, 2020. 1

[49] Pu Zhang, Wanli Ouyang, Pengfei Zhang, Jianru Xue, and Nanning Zheng. Sr-lstm: State refinement for lstm towards pedestrian trajectory prediction. In *CVPR*, 2019. 2

[50] Hang Zhao, Jiyang Gao, Tian Lan, Chen Sun, Benjamin Sapp, Balakrishnan Varadarajan, Yue Shen, Yi Shen, Yuning

Chai, Cordelia Schmid, et al. Tnt: Target-driven trajectory prediction. *CoRL*, 2020. 1, 2

[51] Tianyang Zhao, Yifei Xu, Mathew Monfort, Wongun Choi, Chris Baker, Yibiao Zhao, Yizhou Wang, and Ying Nian Wu. Multi-agent tensor fusion for contextual trajectory prediction. In *CVPR*, 2019. 6, 7

[52] Ming Zhou, Jun Luo, Julian Villela, Yaodong Yang, David Rusu, Jiayu Miao, Weinan Zhang, Montgomery Alban, Iman Fadakar, Zheng Chen, et al. Smarts: Scalable multi-agent reinforcement learning training school for autonomous driving. *arXiv:2010.09776*, 2020. 2