2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

SWIFT: Strategic Weather-informed Image-based Forecasting for Trajectories

Youya Xia[1], Jose Nino[2], Yutao Han[3] and Mark Campbell[2]

***Abstract*— Predicting agents' trajectories in complex environments is critical for achieving safe autonomous robot navigation. Empirically, agents' decisions and preferences are susceptible to changes in environmental factors (*e.g.*, interactions with other agents, weather conditions, traffic rules). State-of-the-art methods rely on High-Definition (HD) or semantic maps to model the environment, but do not take into account unpredictable factors such as complex weather conditions. In addition, since HD maps are nontrivial to obtain, those methods are limited in the scope of environments they can be applied in. We propose a more flexible graph based trajectory prediction model that uses only images to model the environment, without requiring expensive map information. We experimentally validate our proposed model, demonstrating robust performances in trajectory prediction compared to state-of-the-art methods, and outperform in complex environments that cannot be modeled with purely map based methods, such as diverse weather conditions.**

I. Introduction

Predicting agents' future trajectories accurately is essential for developing safe self-driving autonomous systems. However, agents' future motions are not only based on their past trajectories but also on other environmental factors (*e.g.*, weather conditions, surrounding agents, traffic rules). Since those environmental factors are difficult to model, prediction uncertainties tend to grow exponentially with variations of those factors.

In order to support predicting future trajectories properly, many existing state-of-the-art methods aim to understand complete scene context by extracting features from rasterized top-view representations [17], [29], [36] from view limited camera images. Since agents tend to obey traffic rules such as lane lines, other methods such as LaPred [17] and LaneGCN [20] infer the relationship between agents and road lanes by building a lane graph. In both of these, however, obtaining a top-view representation and accurate lane labels require High-Definition (HD) Maps, which are expensive to acquire and are only included in a few self-driving datasets (*e.g.*, Nuscene [2] and Argoverse [5]). Thus, it is not practical to apply methods that rely solely on semantic and/or HD map information when these are not readily available.

Although methods such as BEVFormer [19] and FIERY [13] aim to address this problem by only relying on camera images, they are mainly focused on predicting future instance segmentation in the Bird-eye View (BEV); they fail to predict actual future motions (*e.g.*, locations, accelerations, velocities). Therefore, they are not suitable for solving the real-world trajectory prediction problems.

In this paper, our goal is to achieve the state-of-the-art trajectory prediction performance **without using HD or semantic maps** and in **varying weather conditions**. Specifically, we propose a Conditional Variational Encoder (CAVE) network with a constructed node graph of the current scene as input. Our proposed model-SWIFT, obtains detailed semantic information in scenes by using an LSTM [12] to extract scene features from only camera image sequences. In addition, in order to provide occupancy details in scenes (*e.g.*, shapes of vehicles and roads), we extract edge features in scenes to model scenes' structural information. The image and edge feature extractors provide both static and dynamic contextual information in scenes.

We also directly address weather and lighting dependencies by training a conditioned network using different weather and lighting conditions [8]. Importantly, agents' trajectories are inclined to be affected by weather and lighting conditions. For example, as demonstrated in Figure 1, under snowy weather conditions, agents avoid driving on road regions where snow accumulates, even though those regions are drivable during sunny conditions. Likewise, during rainy weather or night conditions, agents typically will drive slower to avoid sliding and unexpected conditions, compared to driving the same regions during sunny conditions. Thus, in our model SWIFT, we also incorporate weather factors into the trajectory forecasting process using camera images. We append a classification layer to the image feature extractor so that we can obtain weather label information in SWIFT's encoder. We improve the robustness of trajectory prediction under diverse weather conditions, by incorporating the predicted weather label and the generation of latent intent and future trajectories. Finally, we propose displacement and intention losses to optimize prediction accuracy in our proposed model.

Due to the importance of weather conditions on trajectory prediction, we use the Ithaca365 [8] dataset, which collects a series of driving sequences of the same locations under different weather conditions, as one of our main evaluation datasets.

Our contribution is summarized as follows:

- A novel trajectory prediction model-SWIFT, that does not rely on HD maps and uses only camera images for scene context. Thus, it can be applied effectively under diverse scenarios where HD or semantic maps are unavailable.
- Our model is the **first** trajectory prediction method that

[1]Youya Xia is with Computer Science Department, Cornell University yx454@cornell.edu
[2]Jose Nino, Mark Campbell are with the Department of Mechanical Engineering, Cornell University jan268, mc288@cornell.edu
[3]Yutao Han is with OPPO Research yh675@cornell.edu

Fig. 1 This example from the Ithaca365 dataset [8] shows the drivable area (*i.e.*, potential trajectory choices) shrinks with the degradation of weather conditions (sunny → snowy). The blue overlay indicates the save driving area. The red curved line indicates a possible future trajectory for the agent.

directly factors in weather and lighting conditions of the self-driving area.

- We evaluate SWIFT's and other state-of-the-art trajectory prediction models performance on the Ithaca365 dataset [8], empirically showing that our method has higher prediction accuracy than other state-of-the-art models in complex environments with diverse weather conditions.
- Even compared with state-of-the-art methods that use HD maps for prediction, our method still shows similar or better performances when using Nuscenes dataset [2] for evaluation.

II. Related Work

A. Physics based Predictions

The early trajectory prediction methods [11], [14], [21], [25] take the velocity, acceleration, and steering angles as input and predict the position for the next time step. Monte Carlo Sampling [30], Kalman filters [15] are classical physics-based methods. When scenes contain limited interactions between agents, those methods can be effectively applied for predictions with **short-time horizon** (*e.g.*, 1 second). When predicting with a longer time-horizon, however, those methods tend to lose accuracy due to increasing maneuver complexity.

More recently, many deep learning-based methods integrate agent kinematics into their models for more physically grounded predictions. [1], [18], [23], [26], [32]. These methods include Reinforcement Learning based models [18], [26], and LSTM based models [1], [23], [32]. Vehicle and pedestrian kinematics are used to either design cost functions in RL or combine Gaussian distributions with LSTM [12], for predicting the future trajectories. Learning-based algorithms can model higher level maneuver complexity and their prediction of the future trajectories have higher accuracy than traditional Monte Carlo Sampling [30] and Kalman filters [15]. However, due to lack of consideration of scene context, many learning based methods fail under more complex environments (such as crossroads, traffic jams and roads in snowy conditions)

B. Semantic-aware based Prediction

State-of-the-art trajectory prediction methods predict future trajectories using the semantic information of the surrounding environment [4], [9], [17], [20], [22], [29], [36]. Earlier work uses either top-view camera images or Lidar sensors to obtain information on the structure of the scene. Car-Net [28] leverages the attention mechanism to learn where to focus in top-view images. Similar to Car-Net [28], Sophie [27] uses top-view cameras as input to the model and takes advantage of GANs [10] and attention mechanism to produces future agents' positions.

Other methods such as [4], [34] use a rasterized top-down representation of scene context as map input to their models. For example, MultiPath [4] generates probabilities over K predefined anchor trajectories based on the cropped agent-centric view of feature representation using a rasterized top-down scene representation.

Methods such as FRM [24], LaPred [17], and LaneGCN [20] aim to learn the lane that the target agent should follow and the impact of agents following nearby lanes on the target agent. Although by leveraging lane information, those methods provide more accurate prediction than methods that use a rasterized top-down representation, they often require HD maps to acquire precise lane labels, which is not trivial to obtain and very few self-driving datasets (nuScenes [2] and Argoverse [5]) have HD maps.

None of the methods mentioned above consider the impact of weather conditions on future agent trajectories. Thus, potential weather information that could be inferred from camera images is never exploited. Our model proposes to take advantage of scene structure and weather information derived from rgb images.

C. Image-Based Prediction

To address the problem of predicting future agents' locations without using any map information, many state-of-the-art methods such as BEVFormer [19] and FIERY [13] have been proposed. Those methods aims to use the multi-camera image sequences and agents' past locations to predict future instance segmentation results (*i.e.*, future agents' locations in BEV space). However, they fail to predict future agents' real-world motion information (*e.g.*, locations, velocities, and

accelerations). Therefore, they are not suitable for tackling real-world trajectory prediction problems.

D. Generative Prediction Approaches

To predict future agent positions, many state-of-the-art trajectory prediction methods [17], [29], [36] use GANs [10] and CAVEs [31]. Recurrent networks (such as RNNs and LSTMs [12]) are chosen as the backbone of most proposed networks. However, few of them consider dynamic constraints during the trajectory generation stage. Thus, trajectories generated from them might be dynamically infeasible. To address this problem, methods such as Trajectron++ [29] only produce control variable distributions and integrate agents' dynamics with the produced control variable to generate future positions.

III. Methodology

A. Problem Setup

Given past agents' trajectories collected over the time sequence $T = \{1, 2, 3, ..., t\}$ (t is defined as the current agent's time step for our paper), we aim to predict agent trajectories over future time steps $\{t+1, t+2, .., t+n\}$. We first assign each agent with one of the two semantic class labels (*i.e.*, vehicle or pedestrian). Then at time t, by incorporating the past positions $x^{(\sigma)}$ ($\sigma \in T$) and additional contextual information (*i.e.,* weather factor w and encoded image feature I), our model predicts future position distribution $y^{(k)} = p(y^{(k)}|x, w, I)$ ($k \in \{t+1, t+2, .., t+n\}$) for N future timesteps over all interacting agents (x is defined as the positions over the sequence T).

B. Map Representation

HD maps contain semantics such as lane information which is useful for trajectory prediction, since agent motion tends to follow lanes. Most state-of-the-art methods [17], [20], [24] extract semantic and interaction features from HD maps. However, collecting HD maps is a time-consuming task and thus HD maps do not exist in many self-driving datasets. To develop a robust and more adaptable trajectory prediction model, compared models that rely on HD maps, we address the trajectory prediction problem **without using any HD map information** while **demonstrating state-of-the-art performance**. Instead we use camera images to (*i.e.*, understand scene context). Camera images provide dense semantic and occupancy information in the scenes, while also being relatively cheap to acquire or a variety of different environments.

C. SWIFT's Structure

In the following sections, we describe the trajectory prediction network.

1) Overall Architecture: Our proposed network-SWIFT is shown in Figure 2. We build upon Trajectron++ [29] and use a directed spatiotemporal graph $G = (V, E)$ to describe the current scene to model interactions between different types of agents. Each node V in the graph represents either a vehicle or a pedestrian in the scene. An edge $E = (V_i, V_j)$ exists if agent V_i affects agent V_j's trajectory. We follow the convention defined in [29] such that there is an edge between agent V_i and V_j if $|e_i - e_j| \leq d_{c_j}$ (e_i and e_j represents the position of agent i and agent j, d_{c_j} refers to the visual range of an agent with semantic label c_j).

The directed graph is input into the encoder stage of the trajectory prediction architecture. For modeling previous trajectory history and interactions with other agents, we first encode agent trajectory histories and agent interaction features using two LSTM blocks respectively.

2) Node History Block: Since the input to the encoder is the constructed scene graph, we can obtain the current and past states of agents. Thus, similar as [29], to extract features from agents' past states, an LSTM network (See Figure 2) is exploited to extract node history features from the history sequences.

3) Edge History Block: Agents' trajectories are not only affected by their planning decisions but also by the surrounding agents' trajectories. Thus, to model the interactions between agents, similar to [29], we also use LSTMs to extract features lying in graphs' edges. Specifically, we feed edge into LSTM blocks (See Figure 2) to obtain different kinds of interaction features. The weights of the LSTM blocks are shared among all edges that have the same semantic types. For example, all LSTMs that extract Vehicle-Vehicle edge features are shared with the same weights. Finally, we use the attention mechanism to combine features from different edge types and feed the weighted features into a later stage.

Then, to extract scene context and environmental cues, we intentionally add two image-based feature extraction blocks in the encoder stage. The first block extracts RGB image features I and the second block extracts occupancy and structural information O in the current scene. We describe the details of the two added blocks below.

4) Image Feature Extractor: We use camera images to provide semantic information normally obtained from HD maps. Specifically, we add an additional LSTM block to the encoder stage. The input to the LSTM block is the sequence of previous images collected from the previous time sequences T. The output of the image feature extraction block is appended to features extracted from the node history and the edge influence blocks.

5) Structural Information Extractor: Structural information for the scene (*e.g.*, positions of lanes, infrastructure, shapes of roads, cars, and pedestrians) is important for agents to understand the scene's occupancy and interactions with lanes and other agents. It's essential to extract structural information during the encoding stage, for understanding the underlying interactions and scene context. Edge detectors such as the Canny edge detector [3] are efficient tools for obtaining structural scene information and we use it on top of the camera images to attain 'edge' image sequences at the previous time sequences T. Then, in the encoding stage, 'edge' image sequences are input into an LSTM block to obtain structure feature E.

6) Weather Classification: Agents' behaviors are usually influenced by changes in lighting and weather conditions.

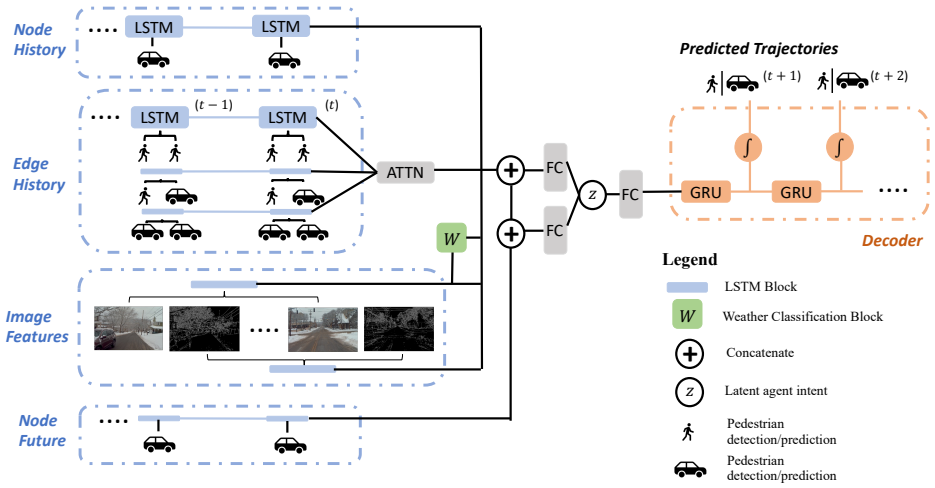

Fig. 2 The network architecture for our proposed trajectory prediction model (SWIFT). Besides node and edge history blocks, we include the image features in the encoder by using both raw image and edge detection sequences (obtained by using Canny Edge detector [3]). The predicted weather label w is also obtained in the image feature block such that in the later stage (*i.e.,*, between encoder and decoder), the generation of agents' latent intent z will consider the forecasted weather information. In the decoder stage, the final future agents' positions are predicted by using the encoded features, latent intent, weather information, and the previous states.

For example, during snowy conditions, vehicles tend to avoid snow-accumulated regions and slippery areas (See Figure 1). Likewise, during driving scenarios at night, drivers tend to avoid regions with poor lighting conditions. We incorporate weather factor classification in the encoding stage of our proposed model since weather conditions affect the decision-making process of agents. Specifically, after the image feature extractor LSTM, we add a fully connected layer and softmax layer for predicting the current scene's weather and lighting conditions. In our case, we divide the scene's weather and lightning conditions into four categories: sunny, snowy, night, and rainy. The softmax layer described above predicts the probability distribution $p(w|I)$ among those four weather labels. The predicted weather label w is then incorporated later into the decoding stage of our proposed model. Note that our weather classification model can not only classify single weather condition, under the case that the dataset contains enough mixed weather condition labels, our proposed model can also be applied potentially for the mixed-weather conditions (*e.g.,*, we can also create weather classification such as night+snowy, night+rainy)

7) Latent Intent Classification: Agent intent, which is defined as discrete decision making (*e.g.*, turning around or going forward) significantly affects future trajectories. Thus, similar to other state-of-the-art trajectory prediction models [29], we use a categorical latent discrete variable $z \in Z$. The latent variable z can be regarded as a latent indication of agents' intent (*e.g.*, turn right, turn left, go straight).

Since agents' intent and decisions are affected by not only previous states and interactions, but also current weather conditions, we also incorporate the weather factor w into the prediction model. Thus, the predicted trajectory y can be expressed as $p(y|x) = \sum_{z \in Z} p_\phi(y|x, z, w) p_\zeta(z|x, w)$ (ϕ and ζ are parameters for the described distributions respectively). We constrain $|Z| = 25$.

To optimize the latent intent distribution estimation during the training process, the ground-truth latent behavior distribution is estimated by $q_\gamma(z_{gt}|x, y_{gt}, w_{gt})$ where y_{gt} represents encoded ground truth future trajectory vector and w_{gt} represents ground truth weather labels.

8) Predicting Future Trajectories: Since deep learning models tend to fail considering the system dynamics, the trajectories directly generated by them might be infeasible for agents to traverse. Thus, similar to [29], in the decoding stage of SWIFT, the encoded features f_x, predicted latent intent z, and weather label w are input into the Gated Recurrent Units (GRU) module [6] to generate control variable a^k for the future timesteps. To trajectories suitable for agents' systems, the predicted control variable a^k is then input into one of the two system models (*i.e.*, unicycle and single integrator) according to the agent's type (unicycle model for vehicles and single integrator for pedetrians). By classifying agents into different dynamic models, the generated future trajectories will be feasible for agents to traverse.

D. Training Details

We introduce loss functions used during the training process.

a) Weather Classification Loss: Since we incorporate the predicted weather label w in our model, to optimize the weather classification block and improve the label prediction precision, we first adopt the Cross-Entropy loss for the weather classification block and denote it as $\mathcal{L}_w$.

b) KL Divergence Loss using Latent Intent: Before the decoding stage, the ground truth latent intent is produced by $q_\gamma(z_{gt}|x, w_{gt}, y_{gt})$. The latent intent $z^{'}$ conditioned on the predicted future trajectory y should be similar with z conditioned on the ground truth y_{gt}. Thus, to optimize

the predicted trajectory, we first estimate $z^{'}$ by using the predicted weather label w, current state x, and the **predicted** future trajectory y (*i.e.*, $q_\gamma(z^{'}|x,w,y)$). Then, we use KL divergence loss to reduce the distance between z and $z^{'}$. Specifically, the loss $\mathcal{L}_z$is given by:

$$\mathcal{L}_z = D_{KL}(q_\gamma(z^{'}|x,w,y)|q_\gamma(z_{gt}|x,w_{gt},y_{gt})) \quad (1)$$

c) Displacement Loss: We employ a displacement loss to enforce $||y - y_{gt}||_2 \simeq 0$. Specifically, we use Average Displacement Error (ADE) for computing the displacement loss. To compute the ADE, we sample 1 predicted trajectory during the training. The displacement loss function $\mathcal{L}_d$ is defined as follows:

$$\mathcal{L}_d = \frac{1}{N}\sum_{k=t+1}^{t+n}(||y^k - y^k_{gt}||_2) \quad (2)$$

where $v^{f_{gt},i}$ refers to to the groudtruth future position at timestep i.

d) Total training objective: Besides the described three loss functions above, we also use the infoVAE [29], [35] objective function. The total training objective function can be described as follows:

$$\begin{aligned}\mathcal{L}_{total} = & max_{\phi,\zeta,\gamma} \\ & \mathbb{E}_{z\sim q_\gamma(z_{gt}|x,y_{gt},w_{gt})}[\log p_\phi(y|x,z,w)] \\ & - \delta D_{KL}((q_\gamma(z_{gt}|x,y_{gt},w_{gt})|p_\zeta(z|x,w)) \\ & + \epsilon I_q(x;z) + \mathcal{L}_w + \mathcal{L}_d + \mathcal{L}_z\end{aligned} \quad (3)$$

where I_q is the mutual information between x and z based on distribution q_γ. We compute I_q by approximating $q_\gamma(z_{gt}|x,y_{gt},w_{gt})$ with $p_\zeta(z|x,w)$ [29].

IV. Experiments

In the following experimental section, we evaluate the performance of our proposed model on the Ithaca365 [8] and Nuscenes [2] datasets. Although SWIFT can be potentially applied for mixed-weather labels, due to the **limited number of mixed weather scenarios** in both of the two datasets (*e.g.,*, rainy+night, night+snowy), in the following experiment section, we only discuss the experimental results for single weather labels.

A. Experiments on Ithaca365 Dataset

1) Introduction: The Ithaca365 dataset [8] is a publicly-released dataset collected across four diverse weather conditions (*i.e.*, sunny, rainy, snowy and night) and four diverse scenes (*i.e.*, rural, highway, urban and campus area) along a repeated 15km route. Since it contains multiple driving traversals along the exact same routes with different weather conditions (*i.e.*, sunny, snowy, rainy, and night) it is a uniquely suitable dataset to evaluate our image based agent trajectory prediction method with weather factor classification.

Ithaca365 dataset [8] contains 40 labeled traversals annotated at 10 Hz, and Table I shows the number of traversals collected under each of the four weather conditions respectively. To ensure the data we use for training and testing does not contain imbalanced classes, we use the same amount of traversals **8** per weather condition, which means that we use 32 out of 40 traversals for training and testing. Finally, for each traversal, we choose 25% of the traversal for testing (*i.e.*, 3.75 km) and 75% of the traversal for training (*i.e.*, 11.25 km). Each route traversal in the Ithaca365 dataset contains around 7500 images for training and around 2500 images for testing.

TABLE I The number of traversals collected under different weather conditions

	sunny	rainy	night	snowy
# of traversals	15	8	8	9

2) Training and Evaluation Metrics:

a) Training: SWIFT was trained on one 3090 GPU for 20 epochs on the Ithaca365 [8] dataset.

b) Evaluation Metrics: Following the convention of prior work [29], we evaluated our proposed model using two popular metrics-*average displacement error* (ADE) and *final displacement error*(FDE). ADE is defined as the minimum error of the l_2 distance between the predicted **most likely** trajectory and the ground truth trajectory. FDE is defined as following: At target time stamp t, the minimum error of the l_2 distance between the predicted final **most likely** position and the ground truth final position. To evaluate the trajectory prediction accuracy with the increasing time horizon, we evaluate predicted future trajectories at $t = \{2, 3, 4\}$ s using the described ADE and FDE metrics.

3) Main Results: Since the Ithaca365 [8] dataset does not contain any map information, we only compare our proposed model with the state-of-the-art models that perform trajectory prediction **without** any semantic maps. Specifically, the four state-of-the-art models we choose to compare with are AgentFormer [33], FQA [16], S-LSTM [1], Trajectron++ [29]. We evaluate our models quantitatively and qualitatively to show our image based model improves trajectory prediction results under adverse weather conditions.

From the table II, since our method has the lowest ADE and FDE results among the five evaluated methods, we can conclude that generally, our proposed model outperforms all other state-of-the-art methods which do not require HD map information as a necessary component.

Also, from the FDE and ADE evaluation under four weather conditions (*i.e.*, sunny, rainy, night, and snowy) (Table II), we have three additional findings:

- Compared with sunny and night conditions, FDE and ADE errors are larger under rainy and snowy conditions. Thus, those two weather conditions cause the most challenging scenarios for trajectory prediction.
- Compared with four state-of-the-art methods, under adverse weather conditions (*i.e.*, rainy and snowy), the ADE and FDE errors of our proposed model are even lower than the ADE and FDE errors of other baselines under benign weather conditions (*i.e.*, sunny). Thus,

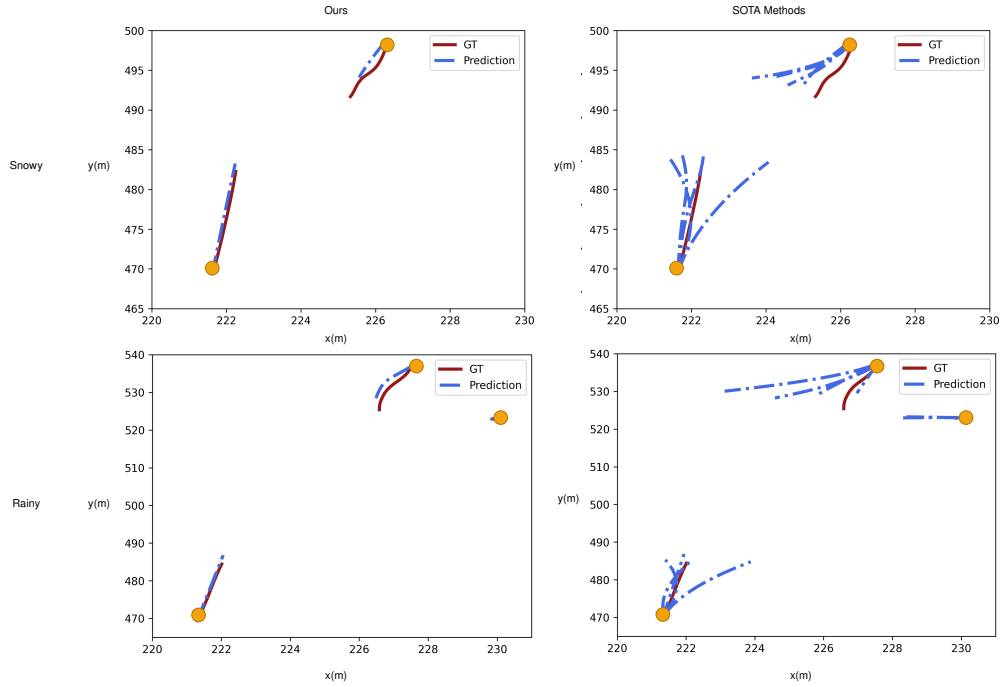

Fig. 3 Visual comparisons of predicted trajectories produced by our proposed model and the four SOTA methods (FQA [16], S-LSTM [1], Trajectron++ [29], AgentFormer [33]) on the Ithaca365 [8] dataset. Columns demonstrate our model versus the four SOTA models while rows demonstrate different weather conditions (snowy, rainy). Our proposed model predicts the most accurate future trajectories compared with the other methods.The yellow dots in the picture indicate the starting location of agents.

TABLE II Performance comparison between our proposed model and the four baselines evaluated on the **entire** Ithaca365 [8] test set and Ithaca365 [8] test set under four different weather conditions(*i.e.*, snowy, rainy, night and sunny)

Prediction Horizon (ADE/FDE)					
Entire Ithaca365 [8] Dataset					
Model	Trajectron++ [29]	AgentFormer [33]	S-LSTM [1]	FQA [16]	Ours
2s	0.337/0.940	0.329/0.932	0.543/1.431	0.403/1.047	***0.268/0.693***
3s	0.609/1.831	0.591/1.618	0.912/2.197	0.829/1.828	***0.476/1.349***
4s	0.956/2.957	0.939/2.872	1.216/3.871	1.001/3.217	***0.741/2.171***
Snowy Conditions					
2s	0.369/0.994	0.362/0.990	0.681/1.554	0.453/1.125	***0.281/0.722***
3s	0.644/1.893	0.627/1.867	1.006/2.392	0.883/1.910	***0.495/1.389***
4s	0.993/2.769	0.984/2.967	1.471/4.002	1.131/3.416	***0.765/2.222***
Rainy Conditions					
2s	0.367/1.001	0.361/0.982	0.670/1.545	0.450/1.134	***0.278/0.719***
3s	0.651/1.912	0.631/1.885	0.998/2.377	0.882/1.905	***0.492/1.395***
4s	1.002/3.109	0.987/3.005	1.438/3.892	1.164/3.411	***0.759/2.235***
Night Conditions					
2s	0.334/0.935	0.314/0.928	0.547/1.434	0.401/1.043	***0.257/0.680***
3s	0.601/1.639	0.583/1.600	0.916/2.200	0.824/1.819	***0.461/1.327***
4s	0.935/2.943	0.919/2.825	1.221/3.877	0.982/3.109	***0.722/2.144***
Sunny Conditions					
2s	0.313/0.841	0.309/0.804	0.408/1.22	0.366/0.928	***0.227/0.595***
3s	0.560/1.682	0.545/1.500	0.850/2.078	0.747/1.660	***0.419/1.207***
4s	0.889/2.843	0.882/2.779	0.725/3.569	0.871/3.046	***0.666/1.981***

our proposed model shows state-of-the-art performances across all weather conditions.

- With regard to the ADE metric, for our proposed method, the ADE of the adverse weather conditions is about 3% higher than the ADE results of the average results across all weather conditions. Since this percentage is the lowest across all evaluated models, our method shows a smaller variance between adverse weather conditions (*i.e.*, snowy and rainy) and the average results calculated across all weather conditions (Table II). Thus, our method demonstrates robust performances even under adverse weather conditions.

Qualitative evaluation of the trajectory prediction results under snowy and rainy conditions is shown in Figure 3. From the figure, we can see that with the increasing of prediction length, other four baselines tend to gradually deviate from the groudtruth labels. However, the predicted trajectories produced by our method are consistently follow the groudtruth labels. Besides when the vehicles are comparatively static (*e.g.*, short trajectories on the Figure 3), our method can correctly identify the status of the current vehicle while other method tend to generate trajectories that do not actually exist in the future. Thus, comparing with other state-of-the-art methods, we can conclude that our method has the minimum errors from the ground truth trajectories, which also means that our proposed model-SWIFT is robust under adverse

weather conditions.

B. Nuscene [2] Dataset

We conduct additional evaluation on the popular Nuscenes dataset [2] which features HD maps. We use the same metrics as described in Section IV-A.2.b and evaluate all methods at $t = \{2, 4, 6\}$ s. Four state-of-the-art methods are chosen for comparison-Trajectron++ [29], PGP [7], LaPred [17] and AgentFormer [33]. Note that since the four SOTA methods are focused on predicting vehicles' future trajectories. Therefore, in the following experimental table III, we only discuss the ADE and FDE results for the predictions of vehicles' trajectories.

1) Training Details: Since the Nuscene dataset [2] is sampled at 2 Hz, it contains a limited number of images to train a good weather classifier. Thus, we pretrained the entire image feature extractor module on the Ithaca365 dataset [8] to obtain a robust weather classifier. Then, before the training, we initialize the weights of the image feature extractor from the pretrained module and finetune its weights during training. The SWIFT has been trained using 20 epochs on Nuscene dataset [2].

TABLE III Performance comparison between our proposed model and other four baselines evaluated on the Nuscene [2] test set (vehicles).

Prediction Horizon (ADE/FDE)	2s	4s	6s
Trajectron++ (w/ map) [29]	0.21/0.45	0.82/2.20	1.99/5.81
AgentFormer [33]	0.21/0.42	0.78/1.99	1.86/3.89
LaPred [17]	-	-	1.41/2.65
PGP [7]	-	-	***1.32/2.48***
SWIFT(ours)	***0.17/0.33***	***0.50/1.42***	1.35/2.57

2) Main Results: From table III, the ADE and FDE results of our proposed model are better than all listed state-of-the-art methods except PGP [7], which means our model demonstrates the state-of-the-art performances even compared to methods that use HD map information. Therefore, using image modules to replace the HD map information is not only cheaper and more flexible, but also does not degrade performance and actually performs favorably compared to methods using HD maps.

C. Ablation Study on Ithaca365 [8] Dataset

1) Overall Ablation Study: We perform ablation studies to evaluate the improvements obtained from the proposed components. First, we construct the baseline model **A** by removing the image feature extractor, structural information module, displacement loss, and KL divergence intent loss (*i.e.*, the baseline Trajectron++ model without using HD maps [29]). Then we construct an improved model **B** by adding the image feature extractor (**without** weather classification layer) in the encoder stage. **On top of** model **B**, we construct a further improvement by appending the weather classification layer to the image feature extractor (we call it model **C**). We conduct a further improvement **D** by adding the structural feature extractor module. Model **E** is constructed by adding the displacement loss during training on top of model **D**. Model **F** is constructed by adding the KL divergence loss during training on top of model **E**.

From the ablation evaluation Table IV, there are three findings. First, all proposed components facilitate the improvement of trajectory prediction to different degrees. Second, from the improvement of ADE and FDE results, model **B** demonstrates the largest percentage of improvement (10% improvement from model **A** to model **B**). Thus, the addition of an image module feature extractor plays a more important role in our proposed trajectory prediction model. Finally, because considering weather conditions can help drivers arrive at the same location while avoiding water-accumulated or snow-accumulated regions, compared with other design choices, the addition of a weather classification layer improves the ADE results more obviously.

TABLE IV Ablation Study on the Ithaca365 [8] test set.

Prediction Horizon (ADE/FDE)	2s	3s	4s
A	0.337/0.940	0.609/1.831	0.956/2.957
B	0.327/0.840	0.589/1.632	0.925/2.657
C	0.280/0.773	0.537/1.472	0.771/2.392
D	0.275/0.730	0.517/1.405	0.759/2.281
E	0.269/0.696	0.500/1.366	0.753/2.250
F(ours)	0.268/0.693	0.476/1.349	0.741/2.171

2) Evaluation of Weather Classification: We evaluate the accuracy of the weather classification block of our model on the Ithaca365 test set. From the results in Table V, we see that the sunny and night have the highest classification accuracy since they have the most distinguishable features. The classification accuracy of the raining scenario is the lowest because sometimes light rain scenarios and sleet cause challenges in differentiating between rainy and other weather conditions (*e.g.*, snowy and sunny).

TABLE V Accuracy of Weather Classification

Weather	sunny	Snowy	Rainy	Night
Accuracy(%)	98.03%	93.85%	91.21%	97.64%

V. Conclusion

We propose SWIFT, a flexible trajectory prediction model that can be applied effectively even under scenarios where HD map information does not exist. By leveraging image based features, our model empirically outperforms other state-of-the-art baselines, even without using HD map information. SWIFT takes weather conditions into consideration when using camera image sequences, and demonstrates robustness for trajectory prediction under diverse weather conditions (*i.e.*, sunny, rainy, night, snowy), which is not accounted for in other trajectory prediction methods. SWIFT is a flexible image based trajectory prediction model, and easily applied to a wide variety of scenes and environmental conditions.

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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 961–971, 2016.

[2] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020.

[3] John Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698, 1986.

[4] Yuning Chai, Benjamin Sapp, Mayank Bansal, and Dragomir Anguelov. Multipath: Multiple probabilistic anchor trajectory hypotheses for behavior prediction. *arXiv preprint arXiv:1910.05449*, 2019.

[5] 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8748–8757, 2019.

[6] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *Empirical Methods in Natural Language Processing*, 2014.

[7] Nachiket Deo, Eric Wolff, and Oscar Beijbom. Multimodal trajectory prediction conditioned on lane-graph traversals. In Aleksandra Faust, David Hsu, and Gerhard Neumann, editors, *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 203–212. PMLR, 08–11 Nov 2022.

[8] Carlos A. Diaz-Ruiz, Youya Xia, Yurong You, Jose Nino, Junan Chen, Josephine Monica, Xiangyu Chen, Katie Luo, Yan Wang, Marc Emond, Wei-Lun Chao, Bharath Hariharan, Kilian Q. Weinberger, and Mark Campbell. Ithaca365: Dataset and driving perception under repeated and challenging weather conditions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21383–21392, June 2022.

[9] Liangji Fang, Qinhong Jiang, Jianping Shi, and Bolei Zhou. Tpnet: Trajectory proposal network for motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6797–6806, 2020.

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[11] Dirk Helbing and Peter Molnar. Social force model for pedestrian dynamics. *Physical review E*, 51(5):4282, 1995.

[12] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[13] Anthony Hu, Zak Murez, Nikhil Mohan, Sofía Dudas, Jeffrey Hawke, Vijay Badrinarayanan, Roberto Cipolla, and Alex Kendall. Fiery: Future instance prediction in bird's-eye view from surround monocular cameras. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15273–15282, 2021.

[14] Nico Kaempchen, Bruno Schiele, and Klaus Dietmayer. Situation assessment of an autonomous emergency brake for arbitrary vehicle-to-vehicle collision scenarios. *IEEE Transactions on Intelligent Transportation Systems*, 10(4):678–687, 2009.

[15] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. 1960.

[16] Nitin Kamra, Hao Zhu, Dweep Kumarbhai Trivedi, Ming Zhang, and Yan Liu. Multi-agent trajectory prediction with fuzzy query attention. *Advances in Neural Information Processing Systems*, 33:22530–22541, 2020.

[17] ByeoungDo Kim, Seong Hyeon Park, Seokhwan Lee, Elbek Khoshimjonov, Dongsuk Kum, Junsoo Kim, Jeong Soo Kim, and Jun Won Choi. Lapred: Lane-aware prediction of multi-modal future trajectories of dynamic agents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14636–14645, 2021.

[18] Namhoon Lee and Kris M Kitani. Predicting wide receiver trajectories in american football. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1–9. IEEE, 2016.

[19] Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Yu Qiao, and Jifeng Dai. Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In *European conference on computer vision*, pages 1–18. Springer, 2022.

[20] Ming Liang, Bin Yang, Rui Hu, Yun Chen, Renjie Liao, Song Feng, and Raquel Urtasun. Learning lane graph representations for motion forecasting. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 541–556. Springer, 2020.

[21] Chiu-Feng Lin, A Galip Ulsoy, and David J LeBlanc. Vehicle dynamics and external disturbance estimation for vehicle path prediction. *IEEE Transactions on Control Systems Technology*, 8(3):508–518, 2000.

[22] Kaouther Messaoud, Nachiket Deo, Mohan M Trivedi, and Fawzi Nashashibi. Trajectory prediction for autonomous driving based on multi-head attention with joint agent-map representation. In *2021 IEEE Intelligent Vehicles Symposium (IV)*, pages 165–170. IEEE, 2021.

[23] Jeremy Morton, Tim A Wheeler, and Mykel J Kochenderfer. Analysis of recurrent neural networks for probabilistic modeling of driver behavior. *IEEE Transactions on Intelligent Transportation Systems*, 18(5):1289–1298, 2016.

[24] Daehee Park, Hobin Ryu, Yunseo Yang, Jegyeong Cho, Jiwon Kim, and Kuk-Jin Yoon. Leveraging future relationship reasoning for vehicle trajectory prediction. In *The Eleventh International Conference on Learning Representations*, 2023.

[25] Romain Pepy, Alain Lambert, and Hugues Mounier. Reducing navigation errors by planning with realistic vehicle model. In *2006 IEEE Intelligent Vehicles Symposium*, pages 300–307. IEEE, 2006.

[26] Nicholas Rhinehart, Kris M Kitani, and Paul Vernaza. R2p2: A reparameterized pushforward policy for diverse, precise generative path forecasting. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 772–788, 2018.

[27] 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 *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1349–1358, 2019.

[28] Amir Sadeghian, Ferdinand Legros, Maxime Voisin, Ricky Vesel, Alexandre Alahi, and Silvio Savarese. Car-net: Clairvoyant attentive recurrent network. In *Proceedings of the European conference on computer vision (ECCV)*, pages 151–167, 2018.

[29] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16*, pages 683–700. Springer, 2020.

[30] Alexander Shapiro. Monte carlo sampling methods. *Handbooks in operations research and management science*, 10:353–425, 2003.

[31] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28, 2015.

[32] Anirudh Vemula, Katharina Muelling, and Jean Oh. Social attention: Modeling attention in human crowds. In *2018 IEEE international Conference on Robotics and Automation (ICRA)*, pages 4601–4607. IEEE, 2018.

[33] Ye Yuan, Xinshuo Weng, Yanglan Ou, and Kris M Kitani. Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9813–9823, 2021.

[34] Lingyao Zhang, Po-Hsun Su, Jerrick Hoang, Galen Clark Haynes, and Micol Marchetti-Bowick. Map-adaptive goal-based trajectory prediction. In *Conference on Robot Learning*, pages 1371–1383. PMLR, 2021.

[35] Shengjia Zhao, Jiaming Song, and Stefano Ermon. Infovae: Balancing learning and inference in variational autoencoders. In *Proceedings of the aaai conference on artificial intelligence*, volume 33, pages 5885–5892, 2019.

[36] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12126–12134, 2019.