Journal of Computer Science & Technology

Indexed in: SCIE, EI, INSPEC, CBST, DBLP, etc.

Sponsored by: ICT, CAS & CCF

JCST Papers

Only for academic and non-commercial use

Thanks for reading!

JCST WeChat Subscription Account

JCST URL: https://jcst.ict.ac.cn

SPRINGER URL: https://www.springer.com/journal/11390

E-mail: jcst@ict.ac.cn

Online Submission: https://mc03.manuscriptcentral.com/jcst

Twitter: JCST_Journal

LinkedIn: Journal of Computer Science and Technology

SURVEY

PRINT ISSN: 1000-9000
ONLINE ISSN: 1860-4749
CODEN JCTEEM

Domain Adaptation for Graph Representation Learning: Challenges, Progress, and Prospects

Bo-Shen Shi, Yong-Qing Wang, Fang-Da Guo, Bing-Bing Xu, Hua-Wei Shen, and Xue-Qi Cheng

JCST

Journal of Computer Science & Technology

Vol.40 No.2 Mar. 2025

INSTITUTE OF COMPUTING TECHNOLOGY
CHINESE ACADEMY OF SCIENCES

CHINA COMPUTER FEDERATION

SCIENCE PRESS

SPRINGER

Qian TW, Wang Y, Xu YJ *et al.* A model-agnostic hierarchical framework towards trajectory prediction. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY, 40(2): 322-339, Mar. 2025. DOI: 10.1007/s11390-023-3013-4, CSTR: 32374.14.s11390-023-3013-4

A Model-Agnostic Hierarchical Framework Towards Trajectory Prediction

Tang-Wen Qian[1, 2] (钱塘文), Yuan Wang[1, 2] (王　元), Yong-Jun Xu[1, 2] (徐勇军), *Member, CCF*
Zhao Zhang[1, 2] (张　钊), *Member, CCF*, Lin Wu[1, 2] (吴　琳), Qiang Qiu[2, 3] (邱　强)
and Fei Wang[1, 2, *] (王　飞), *Member, IEEE*

[1] *Domain-Oriented Intelligent System Research Center, Institute of Computing Technology, Chinese Academy of Sciences Beijing 100190, China*

[2] *University of Chinese Academy of Sciences, Beijing 100190, China*

[3] *CAS Key Laboratory of Network Data Science and Technology, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China*

E-mail: qiantangwen@ict.ac.cn; wangyuan21s@ict.ac.cn; xyj@ict.ac.cn; zhangzhao2021@ict.ac.cn; wulin@ict.ac.cn
qiuqiang@ict.ac.cn; wangfei@ict.ac.cn

Received December 6, 2022; accepted June 6, 2023.

Abstract Predicting the future trajectories of multiple agents is essential for various applications in real life, such as surveillance systems, autonomous driving, and social robots. The trajectory prediction task is influenced by many factors, including the individual historical trajectory, interactions between agents, and the fuzzy nature of the observed agents' motion. While existing methods have made great progress on the topic of trajectory prediction, they treat all the information uniformly, which limits the effectiveness of information utilization. To this end, in this paper, we propose and utilize a model-agnostic framework to regard all the information in a two-level hierarchical view. Particularly, the first-level view is the inter-trajectory view. In this level, we observe that the difficulty in predicting different trajectory samples varies. We define trajectory difficulty and train the proposed framework in an "easy-to-hard" schema. The second-level view is the intra-trajectory level. We find the influencing factors for a particular trajectory can be divided into two parts. The first part is global features, which keep stable within a trajectory, i.e., the expected destination. The second part is local features, which change over time, i.e., the current position. We believe that the two types of information should be handled in different ways. The hierarchical view is beneficial to take full advantage of the information in a fine-grained way. Experimental results validate the effectiveness of the proposed model-agnostic framework.

Keywords spatial-temporal data mining, trajectory prediction, hierarchical framework, model-agnostic

1 Introduction

The study of sequential patterns derived from individual or multiple trajectories for the purpose of predicting future movement is a major focus in research fields such as urban flow prediction[1] and travel time prediction[2]. However, multi-agent trajectory prediction differs from other trajectory prediction types due to the social interactions between the observed agent and its neighbors. This unique characteristic requires specialized modeling techniques to effectively predict future trajectories.

Predicting future trajectories of multiple agents is essential for various applications in real life, such as

Regular Paper

A preliminary version of the paper was published in the Proceedings of ACM Multimedia 2022.

This work is supported by the Youth Innovation Promotion Association of Chinese Academy of Sciences under Grant No. 2023112, and the National Natural Science Foundation of China under Grant No. 62206266. Zhao Zhang is supported by the China Postdoctoral Science Foundation under Grant No. 2021M703273.

*Corresponding Author

surveillance systems, autonomous driving, and social robots. Surveillance systems used for city planning must be able to understand how crowds will move and interact to better manage the infrastructure. Autonomous vehicles such as self-driving cars, and social robots such as delivery vehicles are supposed to understand human movement to avoid collisions. In the above applications, accurate trajectory prediction has become a crucial component. However, trajectory prediction is still a challenging task because of several factors, e.g., individual historical trajectory, interactions between agents, and the fuzzy nature of the observed agents' motion[3-5]. Modeling individual historical trajectory is complicated as trajectory prediction methods should focus on memorizing both spatial and temporal variations in a unified memory pool. Interactions between agents are complex as they contain interactions between not only moving agents, but also moving and stationary agents. To get to the observed agent's expected destination, there may exist more than one feasible path that an agent could move along in the future, which is the fuzzy nature of the observed agents' motion.

Recent years have witnessed tremendous progress on trajectory prediction. For modeling individual historical trajectory, many studies modify Recurrent Neural Network (RNN)[6, 7] or the Transformer network[8, 9] to seize spatial and temporal information in a unified memory pool. In order to capture interactions between agents, the model architecture is changed from social pooling[3, 5] to graph structure[10–12] to better capture the complex interactions. To model the fuzzy nature of the observed agents' motion, current studies are mostly based on Conditional Generative Adversarial Networks (CGAN)[13, 14] or Conditional Variational Autoencoders (CVAE)[12, 15] to facilitate the generation of multiple feasible paths rather than relying on a single deterministic trajectory. While existing methods have tackled several previously listed challenges, they treat all the information uniformly, which hinders the full utilization of information.

In this paper, we propose to treat all the information in a two-level hierarchical view.

The first-level view is the inter-trajectory view. In this level, we observe that the difficulty in predicting different trajectory samples varies. As shown in Fig.1, the green agent and her friend, the orange agent, are walking together when they encounter a stranger, the blue agent, at the position marked by the red diamond in their trajectories. Due to their distinct comfortable interactive distances, the green agent adjusts her path, resulting in a more challenging trajectory prediction task, while the blue agent maintains a nearly straight path. As a consequence, the trajectory of the green agent is more complex than that of the blue agent. However, existing studies[3, 5, 12, 14] ignore the difficulty in predicting different trajectories. In order to distinguish between trajectory samples, we define trajectory difficulty and train the proposed framework in an "easy-to-hard" schema.

The second-level view is the intra-trajectory view.

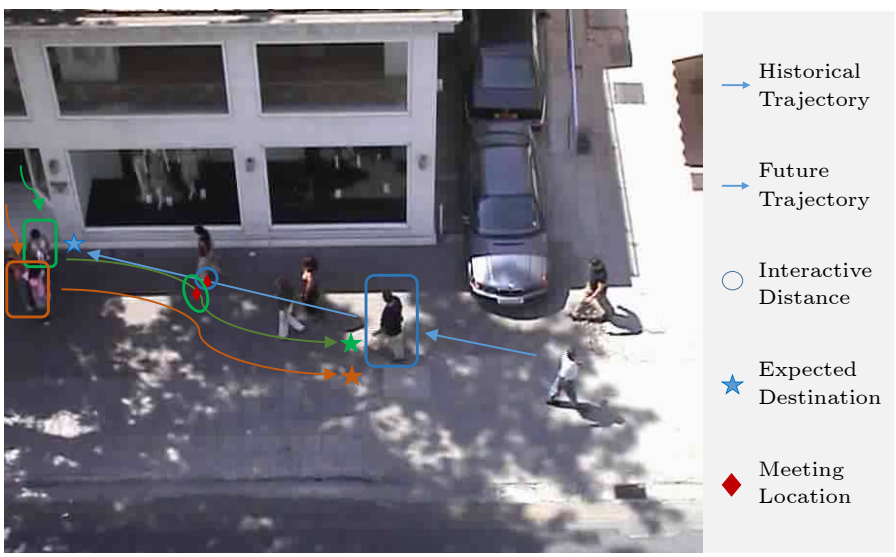


Fig.1. Illustration of the trajectory prediction task. The green agent walks with her friend, the orange agent. They meet with the stranger blue agent at the position, i.e, the red diamond marks in the blue and green trajectories. Agents will consider all neighbors in future movements.

In this level, we find the influencing factors for a particular trajectory can be divided into two parts, global features and local features. The definition is shown as follows.

• Global features indicate the features that are stable within a trajectory, such as comfortable interactive distance and expected destination of agents.

• Local features denote the features that change over time, such as the observed agents' positions and their neighbors' positions.

In Fig.1, the green agent and the orange agent change their paths due to their own positions and neighbors' positions, and finally they arrive at the expected destinations, which do not change within a trajectory. Existing studies[8, 11, 13, 15] do not explicitly distinguish between these two kinds of information, and treat all the information in a uniform way. Different from existing models, we believe that the two types of information should be handled in different ways. The two-level hierarchical view enables our framework to take full advantage of all the information in a thorough and fine-grained way, which is beneficial to achieve better performance. It is worth noticing that our framework outperforms existing methods even without using visual or contextual information, which are not always available under some circumstances in real-world scenes.

To the best of our knowledge, this paper presents a novel approach to modeling the difficulty of trajectory samples, marking the first attempt to explicitly distinguish between global and local features within a trajectory. Additionally, it introduces a pioneering framework that embraces a hierarchical perspective in order to comprehensively consider all available information. Specifically, in this paper, we define a difficulty function to dynamically measure each trajectory sample's difficulty with the development of the framework's capability, and then train the framework in an "easy-to-hard" schema to adaptively improve the framework's learning ability. Furthermore, considering the fact that different features, even inside a trajectory sample, have different influence on the future trajectory, we propose local and global feature extraction modules to divide features inside a trajectory into local and global parts, respectively, and thoroughly explore relationships between them. We highlight that we propose a model-agnostic framework based on the hierarchical view to achieve better performance by distinguishing all the information in a fine-grained way.

Our major contributions are summarized as follows.

• We propose to treat the information in the trajectory prediction task in a two-level hierarchical view. The first-level view is the inter-trajectory view, and the second-level view is the intra-trajectory view. In the first level, we define trajectory difficulty and train the proposed framework in an "easy-to-hard" schema. In the second level, we distinguish between global and local features, and propose to handle them in different ways.

• We propose and utilize a model-agnostic framework based on the hierarchical view and achieve better performance by distinguishing all the information in a fine-grained way.

• Compared with the original conference paper[16], we make a thorough analysis of the two-level hierarchical view and make it a simply model-agnostic framework which can be easily applied to any baseline prediction methods. The model-agnostic hierarchical framework consistently improves the performance on multiple trajectory prediction benchmarks.

The rest of the paper is organized as follows. Section 2 discusses related work. Section 3 gives a formal definition of the trajectory prediction problem. The proposed hierarchical framework is described in Section 4, and Section 5 gives the experimental results of the hierarchical framework. Finally, Section 6 concludes the paper.

2 Related Work

In this section, we first highlight the differences compared with the original conference paper[16], and then discuss existing work on the topic of trajectory prediction. Furthermore, we present relevant work that distinguishes between different samples and distinguishes between features within a sample.

2.1 Differences Compared with Original Conference Paper

This manuscript is an enhanced version of our previous work[16]. Firstly, besides the proposed two-level hierarchical view in [16], we propose and utilize a novel model-agnostic framework based on the hierarchical view to achieve better performance by distinguishing all the information in a fine-grained way.

Secondly, we make a thorough analysis of the proposed framework by adopting a state-of-the-art method as the basic model. Finally, in addition to the major benchmark ETH[17]&UCY[18], the experiments are conducted on a larger-scale dataset SDD[19]. Evaluation metrics are also computed over the top 20 predictions, besides the top 1 prediction in [16].

2.2 Trajectory Prediction

Research in trajectory prediction can be roughly grouped as modeling individual historical trajectory, interactions between agents, and the fuzzy nature of the observed agents' motion. For modeling individual historical trajectory, Recurrent Neural Network (RNN) was designed to tackle sequential problems and performs well in time-series modeling. Social LSTM[3] analyzes individual historical trajectory through RNN. However, there are two crucial aspects in individual historical trajectory: spatial correlation and temporal dynamics. Vanilla RNN focuses more on modeling the temporal aspect. In order to extract spatial and temporal aspects simultaneously, several studies[6, 7, 20] modify RNN to better model spatiotemporal sequence. For instance, PredRNN[20] models spatial and temporal representations in a unified memory cell and conveys the memory both vertically across layers and horizontally over states. However, RNN is still weak and time-consuming in capturing long-range dependency due to its modeling and optimization mechanism. Some studies replace RNN with the Transformer network[9] and Bidirectional Transformer (BERT)[8] in the trajectory prediction task and perform better.

Interactions between agents are complicated since agents are governed by social norms, such as yielding the right-of-way and respecting personal space. Existing algorithms focus on making full use of the interactions between agents. Early studies[21] model the interactions by hand-crafted features like several force terms. However, their performance is limited by the quality of manually designed features, and data-driven social pooling based methods have demonstrated their powerful performance. For example, Social LSTM[3] proposes local social pooling to capture interactions between spatially proximal agents. Social GAN[5] modifies local social pooling to global social pooling to consider subtle cues for all agents involved in a scene. While social pooling based studies model only the pairwise interactions between agents, which simplifies the interactions between agents. Current studies use the graph structure to grasp complicated interactions as the topology of graph is a natural way to represent interactions between agents[11, 14]. For instance, StarNet[22] uses a star topology, which includes a unique hub network and multiple host networks, to consider the collective influence among agents. Social BiGAT[14] introduces a flexible graph attention network to allow all agents in a scene to interact with each other. RSBG[23] proposes a recursive social behavior graph to recursively extract social representations supervised by group-based annotations. Trajectron++[12] adds physical attention and social attention through graph-structured neighbors.

Agents' motion is fuzzy: given the historical trajectory of an agent, there are many feasible paths the agent could move along in the future. Most studies use a sequence architecture with a latent variable model, such as Conditional Variational Autoencoder (CVAE), to explicitly encode multimodality[15, 24], or Conditional Generative Adversarial Network (CGAN) to implicitly do so[4, 5, 11, 13, 14]. For instance, PECNet[24] uses the endpoint estimation VAE for sampling the future endpoints and a trajectory prediction module to use the sampled endpoints to predict future trajectories. Sophie[4] uses the Long Short-Term Memory (LSTM) based GAN module, which takes the highlighted features from a designed attention module, to generate a sequence of plausible and realistic future paths for each agent. While existing methods have tackled several previously listed challenges, they treat all the information uniformly, which limits the effectiveness of information utilization.

2.3 Distinguishing Between Different Samples

Recently drawing a distinction between different samples has received growing research interests in natural language processing[25] and computer vision[26]. Researchers design difficulty functions to estimate the overall difficulty of samples, and then adopt curriculum learning[27] to enable models to gradually proceed from easy samples to more complex ones in training. To the best of our knowledge, no other study has previously distinguished between different trajectory samples' difficulty in the trajectory prediction task. In this paper, we propose a trajectory difficulty function to dynamically measure each trajectory sample's difficulty as the framework's capability evolves.

2.4 Research in Distinguishing Between Features Inside a Sample

Previous studies implicitly encode all features into a dense vector and use it to generate future trajectories[3, 5]. Recently some studies have explicitly extracted the observed agent's expected destination from individual historical trajectories and use the observed agent's expected destination as a uniform factor in the trajectory generation module[12, 24]. To the best of our knowledge, this paper is the first work to explicitly distinguish between global and local features inside a trajectory sample and thoroughly explore relationships between them.

3 Problem Definition

There exists a time-varying number of interacting agents $1, \ldots, N(t)$ in a scene. Here, $N(t)$ is the number of agents at timestamp t. Each agent i has its historical trajectory, e.g., its location (x_i^t, y_i^t) in the scene. Therefore, the historical and current trajectories in the scene are represented by the ordered set of multiple agents' locations:

$$\boldsymbol{X}_i^{1:\,t} = \{(x_i^\tau, y_i^\tau) | \tau = 1, \ldots, t\}.$$

Here, $\forall i \in \{1, \ldots, N(t)\}$, and t represents the current time instant. Our goal is to predict future trajectories of multiple agents based on the historical and current trajectories:

$$\boldsymbol{Y}_i^{t+1:\,T} = \{(x_i^\tau, y_i^\tau) | \tau = t+1, \ldots, T\}.$$

4 Two-Level Hierarchical View

Compared with the original conference paper[16], we further propose and utilize a model-agnostic framework based on the hierarchical view to achieve better performance. All the information is extracted from spatial-temporal trajectories of multiple agents. We treat all the information from a two-level hierarchical view. The first level is the inter-trajectory view, where we define trajectory difficulty, one kind of information, as a means to measure the difficulty of each trajectory sample. The second level is the intra-trajectory view, where we differentiate between local and global information to better explore the relationships among the observed agent's positions, neighbors' positions, the observed agent's expected destination, and the observed agent's comfortable interactive distance. By doing so, we can more thoroughly analyze the information inside a trajectory sample.

The two-level hierarchical view takes full advantage of the information in a fine-grained way for the trajectory prediction task. The first-level view is the inter-trajectory view. In this level, a trajectory difficulty measure module is proposed to define trajectory difficulty to train the proposed framework in an "easy-to-hard" schema. The second-level view is the intra-trajectory view. In this level, a local feature extraction module, a global feature extraction module, and a trajectory generation module are proposed to distinguish between global and local features in each trajectory sample, and handle them in different ways. The architecture of our proposed framework is visualized in Fig.2. In the following subsections, we provide the details of the above components in our proposed model-agnostic framework.

4.1 Model-Agnostic Hierarchical Framework

We further propose and utilize a model-agnostic framework based on the two-level hierarchical view. This model-agnostic framework, which consists of two key modules, can be used to boost a multitude of trajectory prediction models.

- Fir*: this represents the first module of the proposed framework, which defines trajectory difficulty to train the basic model in an "easy-to-hard" schema.
- Sec*: this represents the second module of the proposed framework, which distinguishes between global and local features in each trajectory sample, and handles them in different ways.

In order to verify the performance of the above abstract framework, we choose one basic model to equip with it. In this paper, we choose PECNet[24] as the basic model because it is one of the state-of-the-art studies and a representative study of CVAE-based models. Vanilla PECNet[24] uses the endpoint estimation VAE for sampling the future endpoints and a trajectory generation module to utilize the sampled endpoints to predict future trajectories. Vanilla PECNet consists of a local feature extraction module, a global feature extraction module, and a trajectory generation module.

The detailed architecture of equipping vanilla PECNet with the above framework is visualized in Fig.3. Firstly, we introduce the trajectory difficulty measure module for a comprehensive implementation of Fir*. In this module, we define a dynamic difficulty function to quantify the difficulty of predicting

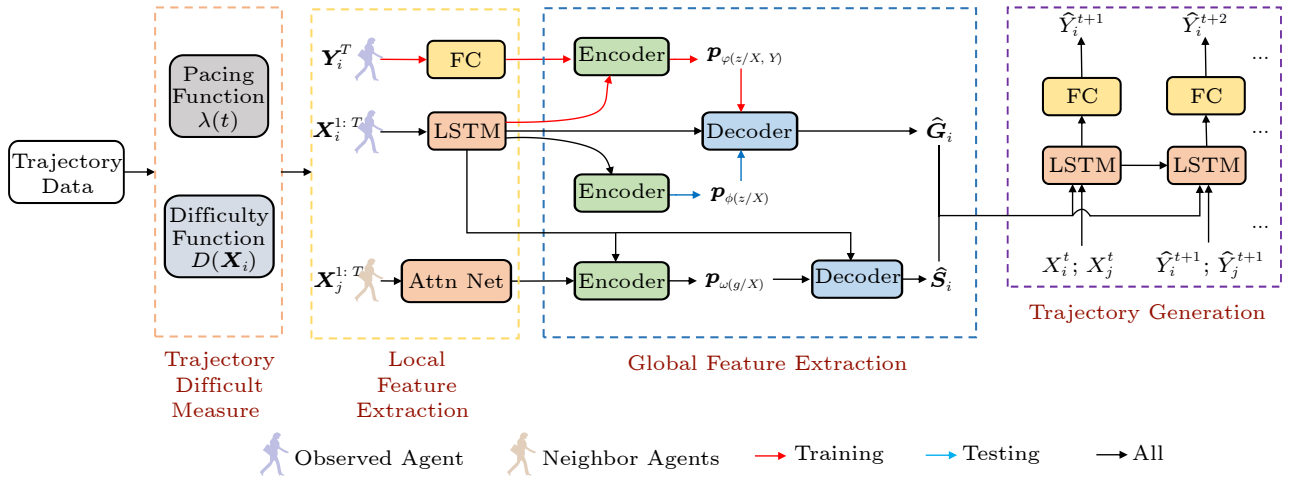


Fig.2. Architecture of our two-level model-agnostic hierarchical framework, which includes the inter-trajectory view and the intra-trajectory view. It mainly consists of four modules, from left to right, the trajectory difficulty measure module, local feature extraction module, global feature extraction module, and trajectory generation module. The leftmost module, i.e., the trajectory difficulty measure module, belongs to the first-level view. The other three modules belong to the second-level view. Red connections are only used in the training stage, blue connections are only used in the evaluation stage, and black connections are used in both stages.

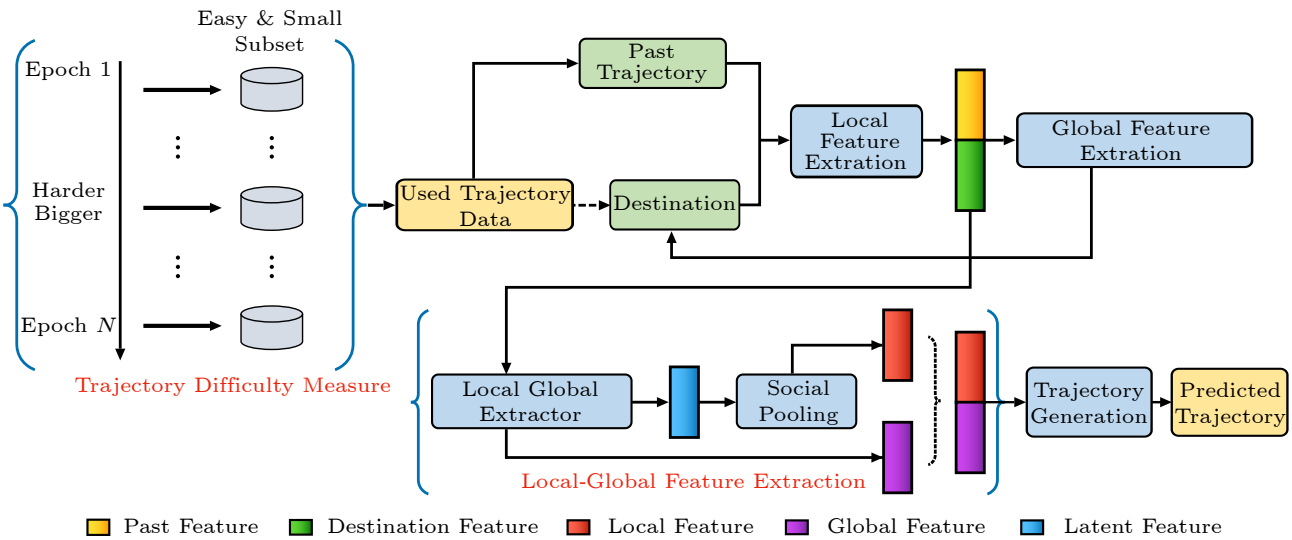


Fig.3. Architecture of our proposed model-agnostic framework adopting one of the state-of-the-art trajectory prediction methods, PECNet, as the basic model. The differences compared with vanilla PECNet consist of two modules which are enclosed in braces. The leftmost module, i.e., the trajectory difficulty measure module, belongs to the first module of the proposed framework. The other module, i.e., the local-global feature extraction module, belongs to the second module of the proposed framework.

each trajectory sample. Then we train the proposed framework in an "easy-to-hard" schema. Secondly, we introduce the local-global feature extraction module for a comprehensive implementation of Sec*. In this module, we distinguish between global and local features and make global features maintained while local features change during trajectory generation.

4.2 Inter-Trajectory View

In the view, the trajectory difficulty measure module is proposed. It is used to dynamically measure the difficulty of trajectories, and enable the framework to be trained with trajectories under the difficulty threshold.

In the trajectory difficulty measure module, we define a dynamic difficulty function to quantify the difficulty of predicting each trajectory sample. A simpler trajectory has a lower difficulty score. Then we train the proposed framework in an "easy-to-hard" schema. We set a difficulty threshold for all the trajectories. The framework's training process starts up with a low threshold, i.e., at the beginning, we train the framework with easy samples. Then we gradually increase the threshold until taking all the samples into account according to a pacing function. In this

way, the framework's ability in finding the mismatching information is progressively strengthened and will be more robust in real-world scenarios[25].

The dynamic difficulty function in our framework is defined as below,

$$D(\boldsymbol{X}_i^{1:\,t}) = \frac{\mathcal{L}_2(\boldsymbol{Y}_i^{t+1:\,T},\ \hat{\boldsymbol{Y}}_i^{t+1:\,T}) - D_{\min}}{D_{\max} - D_{\min}}.$$

Here, $\boldsymbol{Y}_i^{t+1:\,T}$ and $\hat{\boldsymbol{Y}}_i^{t+1:\,T}$ represent the ground truth and predicted future trajectories respectively. $\mathcal{L}_2$ denotes $L2$ norm, e.g., the Euclidean distance between the ground truth and the predicted future trajectories. $D_{\min}$ and $D_{\max}$ denote the minimum and maximum $L2$ norm for all trajectories, respectively, used to standardize each trajectory sample's difficulty into the range of $[0,\ 1]$. 0 means the simplest trajectory, and 1 means the toughest one.

The difficulty function is designed to measure the degree of difficulty for modeling different samples. For instance, when an agent undergoes sudden changes in its intended next location to evade potential conflicts with other agents, this leads to the emergence of large variation points within the trajectory. Consequently, such instances significantly elevate the complexity and difficulty of trajectory prediction tasks. In the above situation, where large variation points exist, the L2 norm tends to be large due to the farther distance between $\boldsymbol{Y}_i^{t+1:\,T}$ and $\hat{\boldsymbol{Y}}_i^{t+1:\,T}$, resulting in a large value of $D(\boldsymbol{X}_i^{1:\,t})$. Besides, with the enhancement of the model's capability during training, the difficulty on trajectory prediction for one certain sample is constantly changing. Therefore, we choose $\hat{\boldsymbol{Y}}_i^{t+1:\,T}$ as prediction results from the last training round to dynamically measure each trajectory sample's difficulty during the training process.

The pacing function adjusts the training data at every epoch by varying the difficulty threshold. The pacing function used in our framework is a general root function. [28] shows that the root function is the most beneficial predefined training scheduler for curriculum learning.

$$\lambda(e) = \min\left(1,\ \sqrt{\frac{1-\lambda_0^2}{T_{\text{grow}}} \times e + \lambda_0^2}\right).$$

Pacing function $\lambda(e)$ maps training epoch number e to a scalar $\lambda \in (0, 1]$, which means top $\lambda(e)$ proportion easy training examples are available at the e-th epoch. Here λ_0 and T_{grow} are hyper-parameters. λ_0 is the initial proportion at the first epoch. T_{grow} denotes the first T_{grow} epochs in the training stage. At the training warm-up stage (first T_{grow} epochs), we learn a basic model with an easy subset of the training data. After T_{grow} epochs (i.e, epoch number $e \geqslant T_{\text{grow}}$), $\lambda(e) = 1$ and the model can then freely access the entire training data.

4.3 Intra-Trajectory View

In the view, the local feature extraction module, global feature extraction module, and trajectory generation module are proposed. The first module encodes local features including the observed agent's positions and neighbors' positions. The second module extracts global features including the observed agent's expected destination and the observed agent's comfortable interactive distance. The third module is fed with local and global features to generate feasible paths.

It is worth noting that different methods adopt different definitions of neighbors. Some methods (e.g., [3]) consider spatially proximal agents as neighbors, with a predefined distance threshold. In contrast, other methods (e.g., [5, 14, 22]) consider all agents involved in a scene as neighbors, with the distance threshold being effectively infinite. In our study, we adopt the latter approach to define neighbors.

4.3.1 Local Feature Extraction Module

In our paper, local features are extracted from locations of agents and neighbors. In order to consider the influence of all neighbors, we use the aggregation operation and attention mechanism to obtain neighbors' local feature.

For an agent's local feature $\boldsymbol{e}_L^A$, we use Long Short-Term Memory (LSTM) to model the observed historical trajectory of the agent for the LSTM's strong performance on sequential summarization tasks.

$$\boldsymbol{e}_L^A = LSTM(\boldsymbol{X}_i^{1:\,t-1}).$$

For neighbors' local feature $\boldsymbol{e}_L^N$, we use LSTM to model the observed agent's neighbors' historical trajectories firstly, and then we aggregate neighbors' features and the observed agent's current location. Aggregated features are fed into the attention mechanism to obtain the local neighborhood feature, which represents the influence of all neighbors on the observed agent.

$$\boldsymbol{e}_L^N = ATTN([LSTM(\boldsymbol{X}_j^{1:\ t-1}),\ f(\boldsymbol{X}_i^t)]).$$

Here, $j \in N(t) \setminus i$ and $N(t) \setminus i$ denotes the collection of all $N(t)$ agents excluding the observed agent i. f represents spatial embedding function to turn current location into a high dimensional vector. Finally, the observed agent's local feature and local neighborhood feature are concatenated to produce local features $\boldsymbol{e}_L$.

$$\boldsymbol{e}_L = [\boldsymbol{e}_L^A,\ \boldsymbol{e}_L^N].$$

4.3.2 Global Feature Extraction Module

In our paper, global features consist of the observed agent's expected destination and the observed agent's comfortable interactive distance. The former is extracted from the historical trajectory of the observed agent, and the latter is extracted from not only the historical trajectory of the observed agent but also the historical interactions between the observed agent and its neighbors.

For the observed agent's expected destination, we adopt the Conditional Variational Autoencoder (CVAE) concepts to obtain its distribution. In this paper, $\boldsymbol{G}_i$ is a high-dimensional vector of an agent's ground truth destination $\boldsymbol{Y}_i^T$. We encode the expected destination $\boldsymbol{G}_i$ of the observed agent i with an encoder E_G to represent the observed agent's ground truth destination in neural networks, which is independent for all agents. Here the encoder E_G is implemented with a multi-layer perception (MLP). The representations are concatenated with the observed agent's local features and passed into the latent encoder $E_{z;\ X,\ Y}$ to learn the distribution of the observed agent's ground truth destination upon both the observed agent's local features and ground truth destination $p_\psi(z|\boldsymbol{X},\ \boldsymbol{Y})$. Here z represents a latent variable, and ψ denotes parameters of the distribution.

$$p_\psi(z|\boldsymbol{X},\ \boldsymbol{Y}) \leftarrow E_{z;\ X,\ Y}([E_G(\boldsymbol{Y}_i^T),\ \boldsymbol{e}_L^A]).$$

The left part of $\leftarrow$ is characterized by the right part. We sample possible latent destination $\hat{\boldsymbol{Y}}_i^T$ from the distribution $p_\psi(z|\boldsymbol{X},\ \boldsymbol{Y})$.

$$\hat{\boldsymbol{Y}}_i^T \sim p_\psi(z|\boldsymbol{X},\ \boldsymbol{Y}).$$

As the ground truth $\boldsymbol{G}_i$ is unavailable at test time, we also learn the distribution of the observed agent's expected destination $p_\phi(z|\boldsymbol{X})$ by feeding the latent encoder $E_{z;\ X}$ only with the observed agent's local features. Here ϕ denotes parameters of the distribution.

$$p_\phi(z|\boldsymbol{X}) \leftarrow E_{z;\ X}(\boldsymbol{e}_L^A).$$

We expect these two distributions $p_\psi(z|\boldsymbol{X},\ \boldsymbol{Y})$ and $p_\phi(z|\boldsymbol{X})$ to be as similar as possible. In other words, we expect the KL divergence between them to be as small as possible. After that, we sample possible latent destination $\hat{\boldsymbol{Y}}_i^T$ from the distribution $p_\phi(z|\boldsymbol{X})$ during the evaluation phase.

$$\hat{\boldsymbol{Y}}_i^T \sim p_\phi(z|\boldsymbol{X}).$$

After we get sampled latent destination $\hat{\boldsymbol{Y}}_i^T$, we concatenate it with the observed agent's local features and decode the concatenation using a decoder D_G to yield our prediction for expected destination $\hat{\boldsymbol{G}}_i$. Here the decoder D_G is implemented with an MLP.

$$\hat{\boldsymbol{G}}_i = D_G([\hat{\boldsymbol{Y}}_i^T,\ \boldsymbol{e}_L^A]).$$

The global expected destination extraction module is illustrated in Fig.2. Notice that the red connections are only used in the training phase, the blue connections are only used in the evaluation phase, and the black connections are used in both phases.

For the observed agent's comfortable interactive distance, in our end-to-end architecture, it is a hidden representation, which is non-human interpretable. The differentiated prediction of each agent indicates the differentiated comfortable interactive distance of each agent. We encode neighbors' local features with an encoder E_S to represent the observed agent's comfortable interactive distance, which is independent for all agents. Here the encoder E_S is implemented with an MLP. The representations are concatenated with the observed agent's local features and passed into the latent encoder $E_{z;\ X}$ to learn the distribution of the latent variable $p_\omega(g|\boldsymbol{X})$. Here g represents a latent variable, and ω denotes parameters of the distribution.

$$p_\omega(g|\boldsymbol{X}) \leftarrow E_{z;\ X}([E_S(\boldsymbol{e}_L^N),\ \boldsymbol{e}_L^A]).$$

We sample possible latent comfortable interactive distance $\boldsymbol{S}_i$ from the distribution $p_\omega(g|\boldsymbol{X})$.

$$\boldsymbol{S}_i \sim p_\omega(g|\boldsymbol{X}).$$

Finally, we concatenate the sampled latent comfortable interactive distance with the observed agent's local features and decode the concatenation using a decoder D_S to yield our prediction for $\hat{\boldsymbol{S}}_i$. Here the

decoder D_S is implemented with an MLP.

$$\hat{\boldsymbol{S}}_i = D_S([\boldsymbol{S}_i,\ \boldsymbol{e}_L^A]).$$

There are two differences between the expected distance extraction and comfortable interactive distance extraction. The first one is that expected distance is extracted from the historical trajectory of the observed agent, and the comfortable interactive distance is extracted from not only the historical trajectory of the observed agent but also the historical interactions between the agent and its neighbors. The second difference lies in the evaluation phase, where the availability of information differs. Specifically, the ground truth destination of the observed agent is solely accessible during the training phase, while the comfortable interactive distance of the observed agent is accessible in both the training and evaluation phases.

Finally, the observed agent's expected destination and comfortable interactive distance are concatenated to produce the observed agent's global features $\boldsymbol{e}_G$.

$$\boldsymbol{e}_G = [\hat{\boldsymbol{G}}_i,\ \hat{\boldsymbol{S}}_i].$$

4.3.3 *Trajectory Generation Module*

In this module, the observed agent's local features $\boldsymbol{e}_L$, global features $\boldsymbol{e}_G$, and current location $\boldsymbol{X}_i^t$ are concatenated and fed into LSTM to generate a location at next timestamp $\hat{\boldsymbol{Y}}_i^{t+1}$.

$$\hat{\boldsymbol{Y}}_i^{t+1} = LSTM([\boldsymbol{e}_L,\ f(\boldsymbol{X}_i^t),\ \boldsymbol{e}_G]).$$

Here f is spatial embedding function described in the local feature extraction module.

When predicting the location at the $t+2$ timestamp, the observed agent's global features remain consistent throughout the whole prediction process, while other features undergo a change. Specifically, the observed agent's local features change from $\boldsymbol{e}_L$ to $\boldsymbol{e}_L^{'}$, and the current location changes from $\boldsymbol{X}_i^t$ to $\hat{\boldsymbol{Y}}_i^{t+1}$. They are concatenated into LSTM to generate locations at the $t+2$ timestamp.

$$\hat{\boldsymbol{Y}}_i^{t+2} = LSTM([\boldsymbol{e}_L',\ f(\hat{\boldsymbol{Y}}_i^{t+1}),\ \boldsymbol{e}_G]).$$

It is worth noticing that global features are maintained while local features change during trajectory generation.

4.4 Loss Function

To train our framework, the loss function consists of three parts,

$$L = L_{\text{Recon}} + L_{\text{G}} + L_{\text{KL}}.$$

L_{Recon} calculates the reconstruction loss between the generated trajectories and ground truth. L_{G} calculates the reconstruction loss between ground truth destination $\boldsymbol{G}_i$ and predicted destination $\hat{\boldsymbol{G}}_i$. Besides, L_{KL} calculates the KLD regularization loss, which measures how close the sampled distribution is to the distribution of latent variables.

In our framework, L_{KL} consists of two parts,

$$L_{\text{KL}} = \sum_q L_{\text{KL}}(q) + L_{\text{KL}}(p_\phi(z|\boldsymbol{X}), p_\psi(z|\boldsymbol{X}, \boldsymbol{Y})).$$

The first part $L_{\text{KL}}(q)$ calculates the KL divergence between the prior distribution and the q distribution. The task of finding a suitable predictive distribution for high-dimensional, real-valued data is a formidable challenge. To overcome this challenge, we adopt the Gaussian distribution as a prior distribution, a common approach shared by existing methods[3, 12]. This choice is motivated by the fact that the Gaussian distribution can accurately approximate any desired density function, provided that its parameters (means and variances) are appropriately chosen[29]. In the paper, the prior distribution is $N(0,1)$, and $q \in \{p_\psi(z|\boldsymbol{X},\ \boldsymbol{Y}),\ p_\phi(z|\boldsymbol{X}),\ p_\omega(g|\boldsymbol{X})\}$.

The second part $L_{\text{KL}}(p_\psi(z|\boldsymbol{X},\ \boldsymbol{Y}),\ p_\phi(z|\boldsymbol{X}))$ calculates the KL divergence between $p_\psi(z|\boldsymbol{X},\ \boldsymbol{Y})$ and $p_\phi(z|\boldsymbol{X})$, as we expect these two distributions to be as similar as possible. The former distribution $p_\psi(z|\boldsymbol{X},\ \boldsymbol{Y})$ is learned during the training phase to model the ground truth destination of agents based on their local features and ground truth destination, while the latter distribution $p_\phi(z|\boldsymbol{X})$ is learned during the evaluation phase to model the expected destination of agents based only on their local features. The greater the similarity between the two distributions is, the more the confidence we can have when sampling latent destinations from the distribution $p_\phi(z|\boldsymbol{X})$ during evaluation.

5 Experiments

This section gives experimental details firstly. Then, we compare our framework's performance against various baselines. We also conduct ablation studies to verify proposed contributions. Different from the original conference paper[16], we conduct experiments for the comprehensive evaluation of the

model-agnostic hierarchical framework. Finally, we present case studies of our framework.

5.1 Experimental Details

In this subsection, we give experimental details of our work, including datasets, baselines, evaluation metrics, and implementation details.

5.1.1 Datasets

We evaluate our framework on well-established public datasets: the ETH[17]&UCY[18] datasets and Stanford Drone dataset (SDD)[19].

The ETH&UCY datasets, which are usually used together in literature, are the major benchmark for pedestrian trajectory prediction. These datasets capture real pedestrian trajectories with rich multi-agent interaction scenarios at 2.5 Hz, e.g., 0.4 s, and consist of five datasets, i.e., ETH-eth, ETH-hotel, UCY-zara1, UCY-zara2, and UCY-univ. They are collected from bird-eye-view cameras in four scenes with 1 536 pedestrian trajectories. The trajectories are extracted in the world space (unit: meters).

The Stanford Drone dataset (SDD), which is a larger-scale dataset in bird's eye view, has been widely used for benchmarking the performance of trajectory prediction models in recent years. It is comprised of more than 11 000 pedestrians across 20 scenes captured in the Stanford university campus. The trajectories are extracted in the image pixel space.

Each dataset includes top-view images, group annotations, and locations of multiple agents. In our paper, we focus on locations of multiple agents and disregard visual and contextual information, which are not always available in real-world scenes.

5.1.2 Baselines

We compare the proposed framework with a range of trajectory prediction models. The compared baselines adopt a discriminative or generative approach to model individual historical trajectories, interactions between agents, and the fuzzy nature of agents' motion. For modeling individual historical trajectories, the recurrent neural network (RNN) and Transformer network are modified to extract spatial correlation and temporal dynamics inside spatial-temporal trajectories. In order to capture interactions between agents, data-driven social pooling based and graph structure based methods are designed to grasp complicated interactions. To model the fuzzy nature of agents' motion, the generative approach is proposed. Most generative baselines use a sequence architecture with a latent variable model, such as a conditional variational autoencoder (CVAE) to explicitly encode multimodality, or a conditional adversarial network (CGAN) to implicitly do so. We briefly describe these representative baselines.

• *Linear*: a naive discriminative model that fits the historical and future trajectories by linear regression.

• *LSTM*: a classical model that captures the agents' historical trajectories by the vanilla LSTM.

• *S-LSTM*[3]: a discriminative model that combines LSTM with a social pooling layer.

• *SGAN*[5]: two GAN-based models (SGAN-V and SGAN-P) that apply generative modeling to S-LSTM.

• *Sophie*[4]: a GAN-based model that utilizes attention mechanisms to model both with social and visual attentions.

• *RSBG*[23]: a discriminative model that incorporates the agent context information into the learning-based method.

• *Trajectron++*[12]: a CVAE-based model that incorporates the agent dynamics and semantic maps.

• *Counter*[30]: a counterfactual analysis method to investigate the causality between the predicted trajectories and input clues and alleviate the negative effects brought by the environment bias.

• *CausalMotion*[31]: a causal method which casts the trajectory prediction task as a dynamic process with three groups of latent variables, namely invariant variables, style confounders, and spurious features.

5.1.3 Evaluation Metrics

There are two common kinds of application scenarios. The first one, e.g., vehicle navigation, gives users several most possible candidate predictions to choose, which cares more about top@K predictions ($K \geqslant 2$)[12, 24]. The other one, e.g., the self-driving vehicle, gives vehicles the most precise instruction, which cares more about top@1 prediction[3–5, 12]. Previous studies[3, 5, 14] mainly focus on only one scenario. In this paper, we focus on both two scenarios.

We employ two commonly used evaluation metrics, minimum ADE/FDE over top k predictions (k=1 or 20), in the trajectory prediction task. Lower

results indicate better performances.

• Average displacement error (ADE): the average Euclidean distance between ground truth locations and predicted locations over all time instants, which is used to evaluate the deviation of overall trend of predicted future trajectories.

• Final displacement error (FDE): the Euclidean distance between the ground truth final location and predicted final location at time instant T, which describes the deviation of destination of predicted future trajectories.

5.1.4 *Implementation Details*

There are two distinct tasks, the 8-8 task and 8-12 task. The difference between them is setting with different prediction horizons. The former task is to predict the next eight frames using past eight frames. The latter one is to predict the next 12 frames using the past eight frames.

On the ETH&UCY datasets, we follow the leave-one-out approach with four sets for training and the remaining set for testing in prior studies[5, 12]. Similar to most prior studies, we predict the next eight or 12 frames (3.2 or 4.8 seconds) based on observed trajectories of eight frames (3.2 s). We report results with metrics computed using top@1 and top@20 predictions. Models are trained for 100 epochs using the Adam optimizer.

On the Stanford Drone dataset (SDD), following [24], we use the standard train-test split and predict the next 12 frames (4.8 s) based on observed trajectories of eight frames (3.2 s). All metrics are computed using top@20 predictions. Models are trained for 650 epochs using the Adam optimizer.

5.2 Quantitative Results

The quantitative results on the ETH&UCY datasets are illustrated in Table 1 and Table 2, which include detailed results of average displacement error (ADE) and final displacement error (FDE). The ADE metric is used to evaluate the deviation about the overall trend of predicted future trajectories, while FDE metric describes the deviation about the destination of predicted future trajectories. In Table 1 and Table 2, the evaluation metrics are the minimum ADE/FDE over top@1 prediction. The difference between these two tables is setting with different prediction horizons.

As seen in Table 1 and Table 2, statistical results of eight frames are better than those of 12 frames, indicating that it is harder to predict farther trajectories into the future. The LSTM method performs better than the linear method upon its ability to capture nonlinearity. The SGAN method achieves better performance than the LSTM method upon the maximum social pooling mechanism. The Sophie method

Table 1. Minimum ADE/FDE (m) over Top@1 Next Eight Frames Prediction on the ETH&UCY Datasets

Method	ETH	HOTEL	UNIV	ZARA1	ZARA2	AVG
Linear	0.84/1.60	0.35/0.60	0.56/1.01	0.41/0.74	0.53/0.95	0.54/0.98
LSTM	0.70/1.45	0.55/1.17	0.36/0.77	0.25/0.53	0.31/0.65	0.43/0.91
S-LSTM	0.73/1.48	0.49/1.01	0.41/0.84	0.27/0.56	0.33/0.70	0.45/0.91
SGAN-P	0.60/1.19	0.52/1.02	0.44/0.84	0.22/0.43	0.29/0.58	0.41/0.81
SGAN-V	0.61/1.22	0.48/0.95	0.36/0.75	0.21/0.42	0.27/0.54	0.39/0.78
Ours	**0.44/0.91**	**0.20/0.38**	**0.20/0.49**	**0.16/0.39**	**0.13/0.32**	**0.23/0.50**

Note: The best results are displayed in bold. AVG indicates the average result of all datasets.

Table 2. Minimum ADE/FDE (m) over Top@1 Next 12 Frames Prediction on the ETH&UCY Datasets

Method	ETH	HOTEL	UNIV	ZARA1	ZARA2	AVG
Linear	1.33/2.94	0.39/0.72	0.82/1.59	0.62/1.21	0.77/1.48	0.79/1.59
LSTM	1.09/2.41	0.86/1.91	0.61/1.31	0.41/0.88	0.52/1.11	0.70/1.52
S-LSTM	1.09/2.35	0.79/1.76	0.67/1.40	0.47/1.00	0.56/1.17	0.72/1.54
SGAN	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	0.70/1.43	0.76/1.67	0.54/1.24	0.30/0.63	0.38/0.78	0.54/1.15
RSBG	0.80/1.53	0.33/0.64	0.59/1.25	0.40/0.86	0.30/0.65	0.48/0.99
Trajectron++	0.71/1.68	**0.22/0.46**	0.41/1.07	0.30/0.77	0.23/0.59	0.37/0.91
Counter	0.89/1.96	0.65/1.44	0.70/1.53	0.64/1.44	0.49/1.08	0.67/1.49
CausalMotion	1.21/2.34	0.72/1.52	0.72/1.56	0.75/1.63	0.63/1.34	0.81/1.68
Ours	**0.53/1.07**	0.26/0.50	**0.40/0.96**	**0.29/0.71**	**0.23/0.56**	**0.34/0.76**

Note: The best results are displayed in bold. AVG indicates the average result of all datasets.

performs better than the SGAN method as it adds a physical attention mechanism to capture visual information of top-view images. The RSBG method achieves better performance than the SGAN method as it incorporates contextual information. Trajectron++ has achieved state-of-the-art results recently as it adds physical attention and social attention through graph-structured neighbors.

Overall, our framework achieves state-of-the-art performance on both the ADE and FDE metrics. All values of compared methods summarized in Table 1 and Table 2 are reported by their authors. For predicting next eight frames, our framework reduces the ADE/FDE of SGAN-V, a better variant of SGAN than other variants, achieving an increase of 41.03%/35.90% on average. For predicting next 12 frames, our framework improves over Trajectron++, one of current state-of-the-art methods, on the average ADE/FDE by 8.11%/16.48%.

It is worth noticing that our framework outperforms existing methods even without using visual or contextual information, which is not always available under some circumstances in real-world scenes. It is also worth noting that our framework achieves larger improvement on a more challenging task, the 8-12 task. The observation indicates that the hierarchical view enables our framework with a stronger capability for farther trajectory prediction tasks.

We also compare the training time of our framework and Trajectron++, one of state-of-the-art methods. As shown in Table 3, it is notable that our framework shows an improvement up to 2.6 times on average faster than the current state-of-the-art work Trajectron++, which shows that our framework has superiority in training efficiency, indicating less computation resource cost.

5.3 Ablation Results

As shown in Table 4 and Table 5, we evaluate various ablations of our framework.

- *Local*: this represents an ablation of our framework with only local features.
- *C-L*: this represents an ablation of our framework with local features and utilization of trajectory difficulty function.
- *C-LnG*: this represents our complete framework, which not only consists of global and local features, but also utilizes the trajectory difficulty function to train our framework.

As shown in Table 4 and Table 5, the local variant performs the worst among all variants as it does not treat features differently. The C-L variant performs better than the local variant, achieving an increase of average 13.33%/11.50% (ADE/FDE) and 8.93%/8.04% (ADE/FDE) on predicting next eight and 12 timestamps respectively. The comparison between the C-L variant and local variant proves the effect of inter-trajectory view. C-LnG outperforms the C-L variant by average 41.03%/33.77% (ADE/FDE)

Table 3. Training Time (min) of Different Methods for 100 Epochs over Top@1 Next 12 Frames Prediction on the ETH&UCY Datasets

Method	ETH	HOTEL	UNIV	ZARA1	ZARA2	AVG
Trajectron++	220	215	160	180	165	188.0
Ours	90	70	83	70	40	70.6

Note: Methods are evaluated on a computer with a 2080Ti GPU. AVG indicates the average results of all datasets.

Table 4. Minimum ADE/FDE (m) of Different Views over Top@1 Next Eight Frames Prediction on the ETH&UCY Datasets

Method	ETH	HOTEL	UNIV	ZARA1	ZARA2	AVG
Local	0.63/1.25	0.60/1.11	0.46/0.91	0.26/0.52	0.29/0.56	0.45/0.87
C-L	0.61/1.21	0.50/0.94	0.34/0.70	0.25/0.51	0.23/0.47	0.39/0.77
C-LnG	**0.44/0.93**	**0.21/0.42**	**0.21/0.49**	**0.16/0.38**	**0.13/0.32**	**0.23/0.51**

Note: The best results are displayed in bold. AVG indicates the average results of all datasets.

Table 5. Minimum ADE/FDE (m) of Different Views over Top@1 Next 12 Frames Prediction on the ETH&UCY Datasets

Method	ETH	HOTEL	UNIV	ZARA1	ZARA2	AVG
Local	0.74/1.47	0.50/0.98	0.71/1.40	0.44/0.90	0.42/0.85	0.56/1.12
C-L	0.70/1.37	0.40/0.81	0.67/1.34	0.43/0.88	0.37/0.74	0.51/1.03
C-LnG	**0.52/1.17**	**0.28/0.57**	**0.40/0.96**	**0.29/0.71**	**0.23/0.56**	**0.34/0.79**

Note: The best results are displayed in bold. AVG indicates the average results of all datasets.

and 33.33%/23.30% (ADE/FDE) on predicting next eight and 12 timestamps respectively upon the utilization of intra-trajectory view. The observation indicates that the global feature extraction module in the intra-trajectory view has even more significant influence over long sequence of trajectory prediction. These two comparisons indicate the effectiveness of our two-level hierarchical framework.

Considering noise in input data, i.e., the observed agent's and neighbors' trajectories, we conduct experiments to analyze how noise affects our framework. We add Gaussian noise when modeling the observed agent's and neighbors' historical trajectories. The results are shown in Table 6 and Table 7. The results of noisy dataset prediction for C-L are generally worse than those of the original datasets, while with the utilization of global features, the results of noisy dataset prediction for C-LnG are better than those of original datasets. The observation indicates that the global features in the intra-trajectory view endow models with robustness.

We also conduct ablation experiments to compare the random and the "easy-to-hard" training strategy:

• Random: making models' training process with randomly selected trajectories;

• Easy-to-Hard: making models' training process start with simpler trajectories, gradually increasing the difficulty until whole trajectories.

As shown in Table 8 and Table 9, the random strategy performs worse than the easy-to-hard strategy for the easy-to-hard strategy fits well with the neural model whose competence increases gradually. This proves the superiority of the easy-to-hard strategy.

For a comprehensive evaluation of the model-agnostic hierarchical framework, we conduct several experiments. An observation length of eight frames and a prediction horizon of 12 frames are used for evaluation.

Experimental results on the ETH&UCY datasets are shown in Table 10. The vanilla PECNet performs the worst among all variants as it does not treat features differently. Both the Fir* variant and the Sec* variant outperform the vanilla model. These findings confirm the effectiveness of the proposed Level 1 view and Level 2 view. The improvement is especially large on relatively complex scenes, such as the ETH-hotel scene, attributing to the trajectory difficulty measure module in the framework that helps the prediction model to distinguish trajectory difficulty in more complex scenes. The Fir-Sec* variant outperforms the

Table 6. Minimum ADE/FDE (m) over Top@1 Next 8 Frames Prediction on the ETH&UCY Datasets

Method	ETH*	HOTEL*	UNIV*	ZARA1*	ZARA2*	AVG*
C-L	0.60/1.19	0.62/1.16	0.42/0.81	0.28/0.55	0.26/0.51	0.44/0.84
C-LnG	**0.44/0.91**	**0.20/0.38**	**0.20/0.49**	**0.16/0.39**	**0.13/0.32**	**0.23/0.50**

Note: The best results are displayed in bold. * indicates adding noise in datasets. AVG indicates the average result of all datasets.

Table 7. Minimum ADE/FDE (m) over Top@1 Next 12 Frames Prediction on the ETH&UCY Datasets

Method	ETH*	HOTEL*	UNIV*	ZARA1*	ZARA2*	AVG*
C-L	0.67/1.25	0.44/0.96	0.74/1.46	0.43/0.90	0.38/0.77	0.53/1.07
C-LnG	**0.53/1.07**	**0.26/0.50**	**0.40/0.96**	**0.29/0.71**	**0.23/0.56**	**0.34/0.76**

Note: The best results are displayed in bold. * indicates adding noise in datasets. AVG indicates the average result of all datasets.

Table 8. Minimum ADE/FDE (m) of Different Training Strategies over Top@1 Next 8 Frames Prediction on the ETH&UCY Datasets

Method	ETH	HOTEL	UNIV	ZARA1	ZARA2	AVG
Random	0.63/1.25	0.60/1.11	0.46/0.91	0.26/0.52	0.29/0.56	0.45/0.87
Easy-to-Hard	**0.61/1.21**	**0.50/0.94**	**0.34/0.70**	**0.25/0.51**	**0.23/0.47**	**0.39/0.77**

Note: The best results are displayed in bold. AVG indicates the average result of all datasets.

Table 9. Minimum ADE/FDE (m) of Different Training Strategies over Top@1 Next 12 Frames Prediction on the ETH&UCY Datasets

Method	ETH	HOTEL	UNIV	ZARA1	ZARA2	AVG
Random	0.74/1.47	0.50/0.98	0.71/1.40	0.44/0.90	0.42/0.85	0.56/1.12
Easy-to-Hard	**0.70/1.37**	**0.40/0.81**	**0.67/1.34**	**0.43/0.88**	**0.37/0.74**	**0.51/1.03**

Note: The best results are displayed in bold. AVG indicates the average result of all datasets.

Table 10. Minimum ADE/FDE (m) of the Model-Agnostic Hierarchical Framework over Top@20 Next 12 Frames Prediction on the ETH&UCY Datasets

Method	ETH	HOTEL	UNIV	ZARA1	ZARA2	AVG
Vanilla-PECNet	1.94	0.93	1.45	1.00	0.74	1.21
Fir*-PECNet	1.80	0.70	1.33	0.92	0.72	1.09
Sec*-PECNet	1.77	0.77	1.29	0.89	0.68	1.08
Fir-Sec*-PECNet	**1.74**	**0.68**	**1.26**	**0.88**	**0.63**	**1.04**

Note: The best results are displayed in bold. AVG indicates the average result of all datasets.

Table 11. Minimum ADE/FDE (m) of the Model-Agnostic Hierarchical Framework over Top@20 Next 12 Frames Prediction on the SDD Dataset

Method	ADE	FDE
Vanilla-PECNet	17.94	35.90
Fir*-PECNet	17.72	34.44
Sec*-PECNet	17.66	34.40
Fir-Sec*-PECNet	**10.26**	**16.08**

Note: The best results are displayed in bold.

Fir* variant due to the utilization of the local-global feature extraction module in the framework. Above comparisons indicate the effectiveness of the proposed model-agnostic framework.

The scenes in the SDD dataset are known to be more complex due to the larger number of moving agents. The results on the SDD dataset reported in Table 11 are consistent with those on the ETH&UCY datasets, where both the Fir* and the Sec* variants outperform the vanilla model. Interestingly, the improvement of the Fir*-Sec* variant, which combines the Level 1 view and Level 2 view, is more significant on the SDD dataset, indicating that our proposed two-level hierarchical view enhances the predictive capability of models in more complex scenes.

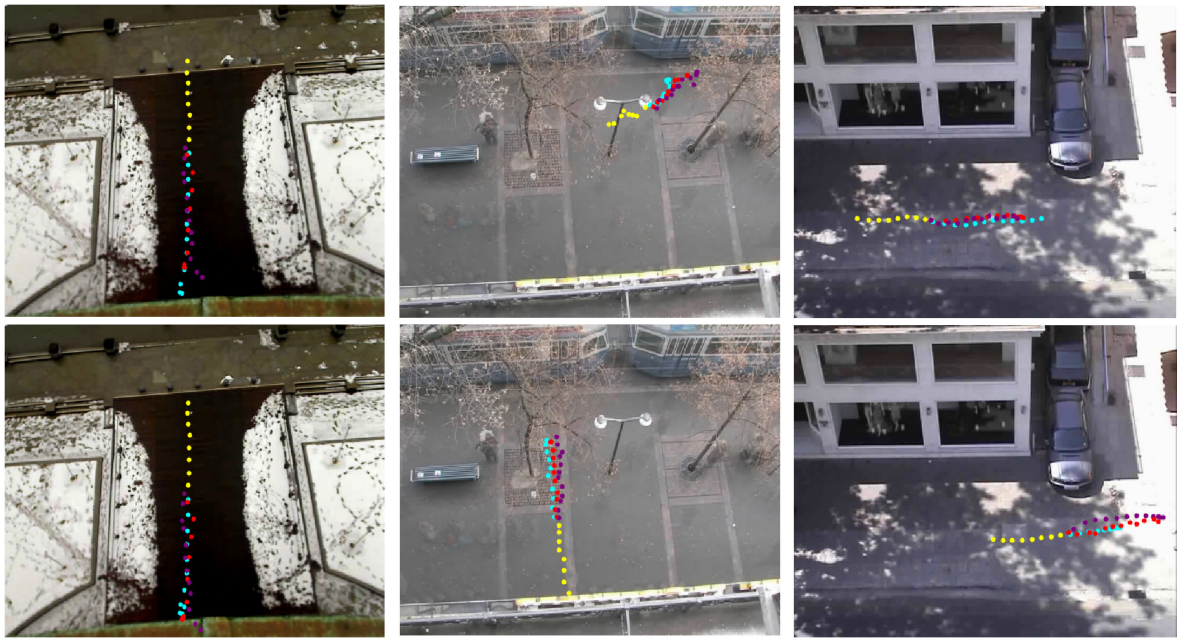
5.4 Case Studies

To thoroughly analyze the limitation of the proposed method, case studies are conducted not only on our framework but also on the model-agnostic framework, as shown in Fig.4 and Fig.5 respectively.

In Fig.4, we represent several visualized prediction results of our framework. In Fig.4(a) depicting the ZARA scene, the ground truth future trajectories almost overlap with the predicted future trajectories, which shows that our framework has the capability of predicting future trajectories which have the same direction and pace with the ground truth. In Fig.4(b) and Fig.4(c) which depicting the Hotel and ETH scenes, respectively, we observe that there is a slight difference between the ground truth and predicted trajectory. Comparison above is consistent with quantitative results, which indicates our framework performs the best in the ZARA scene among all scenes. In Fig.4(b), compared with the ground truth (cyan dots), each predicted trajectory point in the future (red dots) has the same direction with a smaller pace, which indicates that the velocity of the predicted tra-

(a) (b) (c)

Fig.4. Visualized predictions of our framework. Yellow dots represent the trajectory for past 3.2 s while red and cyan dots represent the trajectory of the predicted and the ground truth future, respectively, for next 4.8 s. (a) ZARA. (b) Hotel. (c) ETH.



(a) (b) (c)

Fig.5. Visualized predictions of the proposed framework adopting PECNet as the basic models. Yellow dots represent the trajectory for past 3.2 s, and cyan dots represent the trajectory of the ground truth future for next 4.8 s, while red and purple dots represent the future trajectory predicated by our model and the vanilla PECNet, respectively, for next 4.8 s. (a) ETH. (b) Hotel. (c) ZARA.

jectory is slower than the ground truth one. The slower velocity leads to the deviation of the predicted trajectory in this scene. In Fig.4(c), compared with the ground truth (cyan dots) in the first row, each predicted trajectory point in the future (red dots) has the same direction with a larger pace, which indicates the velocity of the predicted trajectory is faster than the ground truth one. In the second row, the velocity of the predicted trajectory is slower than the ground truth one. Either slower or faster velocity leads to the deviation of the predicted trajectory in this scene. We plan to capture velocity in our future work for different agents in different scenes to achieve more accurate prediction results.

In Fig.5, we represent several visualized prediction results of our proposed framework, which adopts PECNet as the basic model. In Fig.5(a), compared with the ground truth (cyan dots) in the first row, each predicted trajectory point in the future (red dots) has the same direction with a smaller pace, which indicates the velocity of the predicted trajectory is slower than ground truth one. In the second row, the predicted future trajectory (red dots) is close to the ground truth future trajectory, which shows that our framework has the capability of predicting future trajectories which have the same direction and pace with the ground truth. We observe that some predicted trajectory points of vanilla PECNet in the future (purple dots) are beyond the scene. We also observe that the predicted trajectory point of our hierarchical perspective in the future (red dots) is closer to the ground truth than the predicted trajectory point of vanilla PECNet (purple dots) in all columns. Comparison above is consistent with quantitative results, which indicates our hierarchical perspective performs better than vanilla PECNet. In Fig.5(b), compared with ground truth (cyan dots) in the first row, each predicted trajectory point in the future (red dots) has the same direction with a larger pace, which indicates that the velocity of the predicted trajectory is faster than the ground truth one. The faster velocity leads to the deviation of the predicted trajectory in this scene. In the second row, the predicted future trajectory (red dots) is close to the ground truth future trajectory, which shows that our framework has the capability of predicting future trajectories which have the same direction and pace with ground truth. In Fig.5(c), compared with the ground truth (cyan dots) in the first row, each predicted trajectory point in the future (red dots) has the same direction with a smaller pace, which indicates that the velocity of the predicted trajectory is slower than the ground truth one. In the second row, the velocity of the predicted trajectory is faster than the ground truth one. Above observations indicate that either slower or faster velocity leads to the deviation of the predicted trajecto-

ry in this scene. We plan to capture velocity and consider boundary problems in our future work for different agents in different scenes to achieve more accurate prediction results.

6 Conclusions

The results of extensive experiments on several benchmarks show that our model-agnostic framework enhances the performance of multi-agent trajectory prediction. Furthermore, the global feature extraction module has an even more significant influence over long sequence prediction and endows our framework with robustness in case of noise interference. Case studies indicated that our framework predicts future trajectories considering the expected destinations.

In the future, we may consider discovering more information of agents, especially velocity, to better explore the trajectory prediction task. We may also incorporate visual and contextual information for comprehensive multi-modal modeling and further validate our framework on other datasets, e.g., Trajnet++.

Conflict of Interest The authors declare that they have no conflict of interest.

References

[1] Fang Z, Wu D, Pan L. When transfer learning meets cross-city urban flow prediction: Spatio-temporal adaptation matters. In *Proc. the 31st International Joint Conference on Artificial Intelligence*, Jul. 2022, pp.2030–2036. DOI: 10.24963/ijcai.2022/282.

[2] Wan F, Li L, Wang K, Chen L, Gao Y, Jiang W, Pu S. MTTPRE: A multi-scale spatial-temporal model for travel time prediction. In *Proc. the 30th International Conference on Advances in Geographic Information Systems*, Nov. 2022, Article No. 51. DOI: 10.1145/3557915.3560986.

[3] Alahi A, Goel K, Ramanathan V, Robicquet A, Fei-Fei L, Savarese S. Social LSTM: Human trajectory prediction in crowded spaces. In *Proc. the 2016 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2016, pp.961–971. DOI: 10.1109/CVPR.2016.110.

[4] Sadeghian A, Kosaraju V, Sadeghian A, Hirose N, Rezatofighi H, Savarese S. SoPhie: An attentive GAN for predicting paths compliant to social and physical constraints. In *Proc. the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun. 2019, pp.1349–1358. DOI: 10.1109/CVPR.2019.00144.

[5] Gupta A, Johnson J, Fei-Fei L, Savarese S, Alahi A. Social GAN: Socially acceptable trajectories with generative adversarial networks. In *Proc. the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun. 2018, pp.2255–2264. DOI: 10.1109/CVPR.2018.00240.

[6] Zhang P, Ouyang W, Zhang P, Xue J, Zheng N. SR-LSTM: State refinement for LSTM towards pedestrian trajectory prediction. In *Proc. the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun. 2019, pp.12077–12086. DOI: 10.1109/CVPR.2019.01236.

[7] Xu Y, Liu X, Cao X, Huang C, Liu E, Qian S, Liu X, Wu Y, Dong F, Qiu C W, Qiu J, Hua K, Su W, Wu J, Xu H, Han Y, Fu C, Yin Z, Liu M, Roepman R, Dietmann S, Virta M, Kengara F, Zhang Z, Zhang L, Zhao T, Dai J, Yang J, Lan L, Luo M, Liu Z, An T, Zhang B, He X, Cong S, Liu X, Zhang W, Lewis J P, Tiedje J M, Wang Q, An Z, Wang F, Zhang L, Huang T, Lu C, Cai Z, Wang F, Zhang J. Artificial intelligence: A powerful paradigm for scientific research. *The Innovation*, 2021, 2(4): 100179. DOI: 10.1016/j.xinn.2021.100179.

[8] Giuliari F, Hasan I, Cristani M, Galasso F. Transformer networks for trajectory forecasting. In *Proc. the 25th International Conference on Pattern Recognition*, Jan. 2021, pp.10335–10342. DOI: 10.1109/ICPR48806.2021.9412190.

[9] Qian T, Wang F, Xu Y, Jiang Y, Sun T, Yu Y. CABIN: A novel cooperative attention based location prediction network using internal-external trajectory dependencies. In *Proc. the 29th International Conference on Artificial Neural Networks*, Sept. 2020, pp.521–532. DOI: 10.1007/978-3-030-61616-8_42.

[10] Shao Z, Zhang Z, Wang F, Xu Y. Pre-training enhanced spatial-temporal graph neural network for multivariate time series forecasting. In *Proc. the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, Aug. 2022, pp.1567–1577. DOI: 10.1145/3534678.3539396.

[11] Mohamed A, Qian K, Elhoseiny M, Claudel C G. Social-STGCNN: A social spatio-temporal graph convolutional neural network for human trajectory prediction. In *Proc. the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun. 2020, pp.14412–14420. DOI: 10.1109/CVPR42600.2020.01443.

[12] Salzmann T, Ivanovic B, Chakravarty P, Pavone M. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *Proc. the 16th European Conference on Computer Vision*, Aug. 2020, pp.683–700. DOI: 10.1007/978-3-030-58523-5_40.

[13] Amirian J, Hayet J B, Pettré J. Social ways: Learning multi-modal distributions of pedestrian trajectories with GANs. In *Proc. the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, Jun. 2019, pp.2964–2972. DOI: 10.1109/CVPRW.2019.00359.

[14] Kosaraju V, Sadeghian A, Martín-Martín R, Reid I, Rezatofighi S H, Savares S. Social-BiGAT: Multimodal trajectory forecasting using bicycle-GAN and graph attention networks. In *Proc. the 33rd International Conference on Neural Information Processing Systems*, Dec. 2019, Article No. 13. DOI: 10.5555/3454287.3454300.

[15] Lee N, Choi W, Vernaza P, Choy C B, Torr P H S, Chandraker M. DESIRE: Distant future prediction in dynamic scenes with interacting agents. In *Proc. the 2017 IEEE Conference on Computer Vision and Pattern Recognition*, Jul. 2017, pp.2165–2174. DOI: 10.1109/CVPR.2017.233.

[16] Qian T, Xu Y, Zhang Z, Wang F. Trajectory prediction from hierarchical perspective. In *Proc. the 30th ACM International Conference on Multimedia*, Oct. 2022, pp.6822–6830. DOI: 10.1145/3503161.3548092.

[17] Pellegrini S, Ess A, Schindler K, van Gool L. You'll never walk alone: Modeling social behavior for multi-target tracking. In *Proc. the 12th IEEE International Conference on Computer Vision*, Sept. 29 -Oct. 2, 2009, pp.261–268. DOI: 10.1109/ICCV.2009.5459260.

[18] Lerner A, Chrysanthou Y, Lischinski D. Crowds by example. *Computer Graphics Forum*, 2007, 26(3): 655–664. DOI: 10.1111/j.1467-8659.2007.01089.x.

[19] Robicquet A, Sadeghian A, Alahi A, Savarese S. Learning social etiquette: Human trajectory understanding in crowded scenes. In *Proc. the 14th European Conference on Computer Vision*, Oct. 2016, pp.549–565. DOI: 10.1007/978-3-319-46484-8_33.

[20] Wang Y, Wu H, Zhang J, Gao Z, Wang J, Yu P S, Long M. PredRNN: A recurrent neural network for spatiotemporal predictive learning. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2023, 45(2): 2208–2225. DOI: 10.1109/TPAMI.2022.3165153.

[21] Yamaguchi K, Berg A C, Ortiz L E, Berg T L. Who are you with and where are you going? In *Proc. the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2011, pp.1345–1352. DOI: 10.1109/CVPR.2011.5995468.

[22] Zhu Y, Qian D, Ren D, Xia H. StarNet: Pedestrian trajectory prediction using deep neural network in star topology. In *Proc. the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nov. 2019, pp.8075–8080. DOI: 10.1109/IROS40897.2019.8967811.

[23] Sun J, Jiang Q, Lu C. Recursive social behavior graph for trajectory prediction. In *Proc. the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun. 2020, pp.657–666. DOI: 10.1109/CVPR42600.2020.00074.

[24] Mangalam K, Girase H, Agarwal S, Lee K H, Adeli E, Malik J, Gaidon A. It is not the journey but the destination: Endpoint conditioned trajectory prediction. In *Proc. the 16th European Conference on Computer Vision*, Aug. 2020, pp.759–776. DOI: 10.1007/978-3-030-58536-5_45.

[25] Su Y, Cai D, Zhou Q, Lin Z, Baker S, Cao Y, Shi S, Collier N, Wang Y. Dialogue response selection with hierarchical curriculum learning. In *Proc. the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, Aug. 2021, pp.1740–1751. DOI: 10.18653/v1/2021.acl-long.137.

[26] Liu F, Ge S, Wu X. Competence-based multimodal curriculum learning for medical report generation. In *Proc. the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing*, Aug. 2021, pp.3001–3012. DOI: 10.18653/v1/2021.acl-long.234.

[27] Bengio Y, Louradour J, Collobert R, Weston J. Curriculum learning. In *Proc. the 26th Annual International Conference on Machine Learning*, Jun. 2009, pp.41–48. DOI: 10.1145/1553374.1553380.

[28] Wang X, Chen Y, Zhu W. A survey on curriculum learning. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 2022, 44(9): 4555–4576. DOI: 10.1109/TPAMI.2021.3069908.

[29] Bishop C M. Mixture density networks. Technical Report, Aston University, 1994. https://research.aston.ac.uk/en/publications/mixture-density-networks, Jan. 2025.

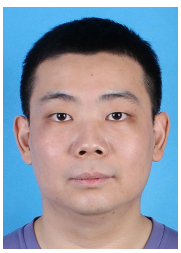
[30] Chen G, Li J, Lu J, Zhou J. Human trajectory prediction via counterfactual analysis. In *Proc. the 2021 IEEE/CVF International Conference on Computer Vision*, Oct. 2021, pp.9804–9813. DOI: 10.1109/ICCV48922.2021.00968.

[31] Liu Y, Cadei R, Schweizer J, Bahmani S, Alahi A. Towards robust and adaptive motion forecasting: A causal representation perspective. In *Proc. the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Jun. 2022, pp.17060–17071. DOI: 10.1109/CVPR52688.2022.01657.

Tang-Wen Qian is currently pursuing her Ph.D. degree in computer architecture from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing. She received her B.E. degree in computer science and technology from Beijing Institute of Technology, Beijing, in 2017. Her research interests include spatial-temporal data mining and multi-agent trajectory prediction.

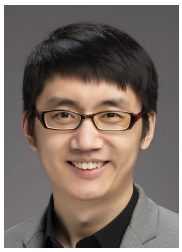
Yuan Wang is currently pursuing his Master degree in computer architecture from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing. He received his B.E. degree in computer science and technology from China University of Petroleum, Beijing, in 2020. His research interests include spatial-temporal data mining and time series forecasting.

Yong-Jun Xu is a professor at the Institute of Computing Technology, Chinese Academy of Sciences, Beijing. He received his B.E. and Ph.D. degrees in computer communication from Xi'an Institute of Posts and Telecoms, Xi'an, in 2001, and the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, in 2006, respectively. His current research interests include artificial intelligence systems and big data processing.



Zhao Zhang is a research associate at the Institute of Computing Technology, Chinese Academy of Sciences, Beijing. He received his B.E. degree in computer science and technology from Beijing Institute of Technology, Beijing, in 2015, and his Ph.D. degree from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, in 2021. His current research interests include data mining and knowledge graphs.

Lin Wu is an associate researcher at the Institute of Computing Technology, Chinese Academy of Sciences, Beijing. He received his Ph.D. degree in computer architecture from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, in 2017. His research interest is multi-source information fusion especially in the domain of maritime situation assessment.



Qiang Qiu is an associate professor at the Institute of Computing Technology, Chinese Academy of Sciences, Beijing. He received his B.S. degree in software engineering from Shandong University, Jinan, in 2010, and his Ph.D. degree in computer application technology from the University of Chinese Academy of Sciences, Beijing, in 2015. His research interests include big data analysis and deep learning system.

Fei Wang is an associate professor at the Institute of Computing Technology, Chinese Academy of Sciences, Beijing. He received his B.S. degree in computer science from the Beijing Institute of Technology, Beijing, in 2011. He received his Ph.D. degree in computer architecture from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, in 2017. His main research interests include spatio-temporal data mining, time series analysis, and AI for science.

Editorial Board of Young Scientists

JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY

Volume 40, Number 2, March 2025

Content

JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY

《计算机科学技术学报(英文)》

Volume 40 Number 2 2025 (Bimonthly, Started in 1986)

Indexed in: **SCIE, EI, Scopus, INSPEC, DBLP, etc.**

Edited by: Editorial Board of Journal of Computer Science and Technology

Editor-in-Chief: Zhi-Wei Xu; Managing Editor: Feng-Di Shu; P.O. Box 2704, Beijing 100190, P.R. China

E-mail: jcst@ict.ac.cn; Available Online: https://link.springer.com/journal/11390, https://jcst.ict.ac.cn

Sponsored by: Institute of Computing Technology, CAS & China Computer Federation

Supervised by: Chinese Academy of Sciences Undertaken by: Institute of Computing Technology, CAS

Published by: Science Press, Beijing, China Printed by: Beijing Baochang Color Printing Co. Ltd

Distributed by:

China: All Local Post Offices *Others*: Springer Nature Customer Service Center GmbH, Germany

国内统一刊号: CN11-2296/TP 国内邮发代号: 2-578 RMB ¥160.00