Online Multi-Agent Forecasting With Interpretable Collaborative Graph Neural Networks

Maosen Li, *Student Member, IEEE*, Siheng Chen, *Member, IEEE*, Yanning Shen, *Member, IEEE*, Genjia Liu, Ivor W. Tsang, *Fellow, IEEE*, and Ya Zhang, *Member, IEEE*

***Abstract*— This article considers predicting future statuses of multiple agents in an online fashion by exploiting dynamic interactions in the system. We propose a novel collaborative prediction unit (CoPU), which aggregates the predictions from multiple collaborative predictors according to a collaborative graph. Each collaborative predictor is trained to predict the agent status by integrating the impact of another agent. The edge weights of the collaborative graph reflect the importance of each predictor. The collaborative graph is adjusted online by multiplicative update, which can be motivated by minimizing an explicit objective. With this objective, we also conduct regret analysis to indicate that, along with training, our CoPU achieves similar performance with the best individual collaborative predictor in hindsight. This theoretical interpretability distinguishes our method from many other graph networks. To progressively refine predictions, multiple CoPUs are stacked to form a collaborative graph neural network. Extensive experiments are conducted on three tasks: online simulated trajectory prediction, online human motion prediction, and online traffic speed prediction, and our methods outperform state-of-the-art works on the three tasks by 28.6%, 17.4%, and 21.0% on average, respectively; in addition, the proposed CoGNNs have lower average time costs in one online training/testing iteration than most previous methods.**

***Index Terms*— Collaborative graph, collaborative predictor, online multi-agent forecasting, theoretical regret analysis.**

Manuscript received May 18, 2021; revised January 2, 2022; accepted February 9, 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 62171276, in part by the Science and Technology Commission of Shanghai Municipal under Grant 21511100900, in part by the National Key Research and Development Program of China under Grant 2019YFB1804304, in part by the Shanghai Municipal Commission of Economy and Informatization (SHEITC) under Grant 2018-RGZN-02046, in part by the 111 Plan under Grant BP0719010, in part by the Shanghai "Science and Technology Innovation Plan" Key Research Program of Artificial Intelligence under Grant 21511100900, in part by the Science and Technology Commission of Shanghai Municipality (STCSM) under Grant 18DZ2270700, and in part by the State Key Laboratory of Ultra High Definition (UHD) Video and Audio Production and Presentation. *(Corresponding authors: Siheng Chen; Ya Zhang.)*

Maosen Li, Siheng Chen, and Ya Zhang are with the Cooperative Medianet Innovation Center, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: maosen_li@sjtu.edu.cn; sihengc@sjtu.edu.cn; ya_zhang@sjtu.edu.cn).

Yanning Shen is with the Samueli School of Engineering, University of California at Irvine, Irvine, CA 92697 USA (e-mail: yannings@uci.edu).

Genjia Liu is with the Cooperative Medianet Innovation Center, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: lgj1zed@sjtu.edu.cn).

Ivor W. Tsang was with the Australian Artificial Intelligence Institute, University of Technology Sydney, Sydney, NSW 2000, Australia. He is now with the A*STAR Centre for Frontier AI Research (CFAR), Singapore 138632 (e-mail: ivor_tsang@ihpc.a-star.edu.sg).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TNNLS.2022.3152251.

Digital Object Identifier 10.1109/TNNLS.2022.3152251

I. Introduction

DYNAMIC multi-agent systems depict a group of co-evolving agents which perform interactions to activate or constrain the pattern of each other. Dynamic systems are critical in many real-world scenarios, such as transportation networks and smart grids [1]–[3]. In many applications, we need to predict future measurements of the systems, such as the traffic flows within hours for route planning [4].

To forecast the systematic status, previous works usually train models with finite training datasets and evaluate on unseen test sets in offline settings. The offline models essentially assume the training and testing samples are stationary and share the same distribution, thus fixed models could generalize to arbitrary dynamics [5]–[7]. We sketch the offline learning paradigms in Fig. 1(a). However, limited training sets are still hard to depict complicated real-world cases [8], and the multi-agent systems could be highly dynamic, resulting in the infinitely possible states changing over time. To make reliable predictions for the dynamic multi-agent systems with streaming states, we need an online model to adapt to varying environments in real-time. Taking streaming data captured along time as inputs, the online multi-agent forecasting model is tested and trained simultaneously step by step and never stopped. The online learning methods never fix the model parameters, while they always tune the models to capture the dynamic even infinite systemic states along with iteration. The online learning paradigm is sketched in Fig. 1(b).

Moreover, to learn the patterns of multiple agents, previous methods usually regard every single agent as an isolated object, such as autoregressive models [9], hidden Markov models [10], Kalman filter [11], and deep sequence-to-sequence models [12]; however, for multi-agent forecasting, these methods might underestimate potential interactions among the co-evolving agents. For example, traffic flows on neighboring streets are mutually dependent; and human body parts might constrain each other during actions. To leverage such relations, some works build graphs, whose edges are predefined or automatically updated during training [13]–[16]. However, such graphs are typically constructed through trials and errors, lacking direct guidance and theoretical interpretation in terms of optimization. In this work, we propose interpretable update steps to learn a graph that captures the collaboration among agents. We further provide the theoretical justification of this graph.

This work aims to design an online forecasting model, which captures the dynamic interactions among agents for real-time prediction. The core component of our method is

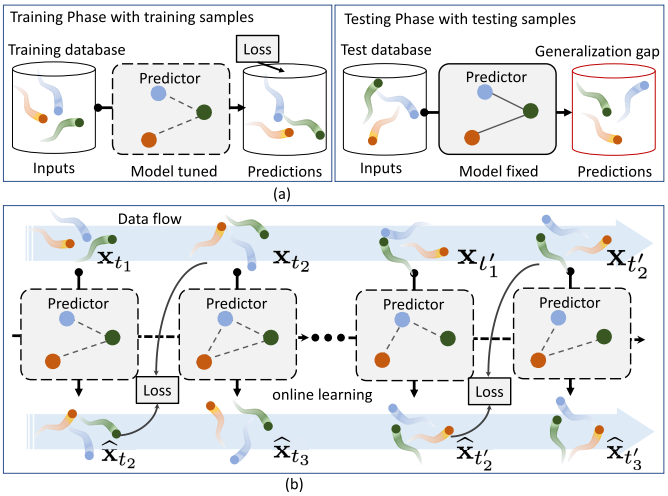


Fig. 1. Comparison of the graph-based predictors in offline and online settings. (a) For offline multi-agent forecasting, a training database is used to tune the predictor, followed by an isolated test phase with a test database. In the test phase, the model and graph construction mechanism is fixed; however, the highly dynamic data and possibly outlier test distribution might cause a generalization gap. We use a dashed box with a graph to represent trainable models and a solid box with a graph for fixed model and graph construction. (b) For online learning, the predictor is tested and tuned simultaneously along with unlimited iterations. The model could dynamically adapt to time-varying data for real-time prediction. Note that predictions are supervised by data flow at delayed iterations.

a collaborative prediction unit (CoPU), which automatically builds a trainable collaborative graph. This collaborative graph includes two types of nodes: agents and collaborative pairs. A collaborative pair, essentially formed by a pair of agents, refers to the mutual collaboration between the two agents. Fed with collaborative pairs, a collaborative predictor is built in CoPU to forecast each agent's status under the inter-agent impacts, thus the collaborative predictors produce multiple possible measurements for each agent affected by other agents. Finally, we learn the edge weights of the collaborative graph, aggregating all collaborative predictors for the final predictions. The edge weights reflect the influence level of each collaborative pair. Notably, in CoPU, the collaborative predictor is trained through online gradient descent, while, in a different way, the edge weights are adjusted by an online multiplicative update step by step to reflect the dynamic topology. The update strategy can be motivated by solving an explicit objective function. To understand CoPU, we conduct regret analysis and prove that the performance of CoPU can converge to that of the best single collaborative predictor.

One advantage of the proposed method is its interpretability, which comes from the following three aspects.

1) *Algorithm*: The edge weights of the collaborative graph are obtained by solving an explicit optimization, while previous works are based on heuristic designs [5], [13], [17]–[20].
2) *Theory*: We prove that the gap between the predictions based on our collaborative graph and the optimal collaborative pair is within a bounded range. In other words, our collaborative graph describes the information and importance carried by different collaborative pairs. In comparison, previous works have no theoretical supports.
3) *Visualization*: Our interpretability is reflected based on human understanding. In experiments, we visualize the learned collaborative graphs changing over time, reflecting reasonable and dynamic topologies in the online setting.

To produce more precise predictions, we further stack a series of CoPUs to build a collaborative graph neural network (CoGNN), progressively refining the predictions. In CoGNN, each CoPU with a collaborative graph is directly supervised by the ground-truth targets, thus all CoPUs predict agents' statuses in the original measurement space. Skip-connections across CoPUs are leveraged to stabilize prediction. Compared to many spatio-temporal graph neural networks (GNNs) [14], [17], [21], the advantages of CoGNN include its interpretability and simplicity, because the information aggregation is performed according to the importance of collaborative pairs in the clear output space, while previous methods generate graph features in the hidden space.

We conduct extensive experiments to validate the proposed CoGNNs on three important tasks: online simulated trajectory prediction, online human motion prediction, and online traffic speed prediction. We note that we formulate a new online setup that simultaneously trains and tests models, and we adapt previous offline state-of-the-art methods to the online setting. According to the results, CoGNNs significantly outperform state-of-the-art methods.

The main contributions are as follows.

1) We propose a novel CoPU, which predicts the future statuses of multiple correlated agents based on a collaborative graph. The collaborative graph contains two types of nodes, that is, individual agents and collaborative pairs, which represent the agent dynamics and collaboration status between agents. In this way, an agent could share customized information with each neighboring agent via the collaborative pair.
2) We employ an online multiplicatively update algorithm to learn the edge weights of the collaborative graph on the fly to solve an explicit objective function [see (7)]. In this way, the graph tuned and tested simultaneously in an online setting effectively adapts to the changing states of the dynamic data flows. Moreover, the collaborative graph can be theoretically justified through regret analysis.
3) We propose a novel CoGNN, which consists of a sequence of CoPUs and progressively refines multi-agent forecasting.
4) We conduct extensive experiments to validate the effectiveness, interpretation, and convergence of our method, which obtains superior and reasonable results.

The rest of this article is organized as follows. In Section II, we introduce some works related to multi-agent forecasting, graph deep learning, and online learning for prediction. In Section III, we define and formulate the problem. In Section IV, we construct the proposed CoPU, introduce the online training method, and conduct the theoretical regret analysis. In Section V, we construct the entire CoGNN model. Finally, the experiments validating the advantages of our model and the conclusion of this article are provided in Sections VI and VII.

II. Related Works

In this section, we introduce critical methods related to our works from aspects of: 1) multi-agent time-series forecasting; 2) graph deep learning; and 3) online learning for prediction.

A. Multi-Agent Forecasting of Time-Series

Multi-agent forecasting has attracted huge attention and brought wide applications. In the early era, state models were studied [10], [22]–[24]. Recently, data-driven models extract deep representations. DeepState [25], Res-sup. [26], AGED [27], and LSTNet [28] design recurrent networks for stability. TCN [29] proposes feed-forward convolution for temporal dependencies. However, the methods do not consider the informative interactions among agents to benefit forecasting.

Recently, some works construct graphs to explicitly exploit the agent relations. Neural relation inference (NRI) [5] infers graphs via an autoencoder for prediction. Structural recurrent neural network (S-RNN) [30] builds factor graphs to propagate agents' information. Diffusion convolutional recurrent neural network (DCRNN) [18] and spatial-temporal graph convolutional network (ST-GCN) [14] incorporate distance-based graphs for traffic forecasting. GraphWaveNet [31] and Traj-GCN [15] use adaptive topologies to capture potential dependencies. Dynamic multiscale graph neural network (DMGNN) [17] builds multiscale graphs. However, most graphs are built with limited human priors or designed through trial and errors, lacking theoretical guarantees and interpretation. In this work, we develop a novel framework for multi-agent forecasting, which infers a collaborative graph with theoretical justification.

B. Graph Deep Learning

The expressiveness of graphs contributes to various scenarios such as social networks [32], bioinformatics networks [33] and human behaviors [19], [20]. GNNs [34]–[38], which expand deep leading to the non-Euclidean domain, have attracted explosive interests. GNNs can be mainly categorized into the spectral-domain-based [34], [39], [40] and vertex-domain-based [35], [41]–[46], which respectively learn the patterns from the graph Fourier representations or the raw topologies. In this work, we learn interpretable graphs for multi-agent forecasting, which captures dynamic and complex correlations for precise prediction.

C. Online Learning for Prediction

A dynamic system evolves with potentially infinite states over time, performing on online streaming data [47]. Tailored for the data flows, online prediction models have been designed to update in real-time [48]. Traditional kernel-based works develop budgeted kernel learning [49]–[52], RF approximations [53]–[55], and multi-kernel learning methods [56]. Recently, some deep-learning-based methods are proposed for online prediction [57]–[59], whose network structures are specifically designed. Besides, the reinforcement learning paradigm can also be employed in this problem [60]. In this work, we train graph-based multi-agent forecasting models in an online setting and optimize graphs to depict the agents' interactions in real-time.

III. Problem Formulation

With the time-series measurements collected by multiple agents in real-time, the task of online multi-agent forecasting essentially aims to predict the future statuses of each agent at each online time stamp based on the corresponding historical states. Mathematically, for the pth agent $v^{(p)}$, at time stamp t, let $\mathbf{x}^{(p)}_{t-\delta:t} \in \mathbb{R}^{\delta\times d}$ be the measurements of the observed clip with length δ and feature dimension d, recording $v^{(p)}$'s states within time interval $(t-\delta, t]$; and $\mathbf{x}^{(p)}_{t:t+\delta} \in \mathbb{R}^{\delta\times d}$ be the measurements of the corresponding ground-truth clip in the future, reflecting $v^{(q)}$'s states within $(t, t+\delta]$. We note that the length of historical and future sequences could be different, while we use δ to simplify notation. Our online predictor $f_t(\cdot)$ aims to produce

$$\left\{\widehat{\mathbf{x}}^{(p)}_{t:t+\delta}\right\}_{p=1}^{N} = f_t\left(\left\{\mathbf{x}^{(p)}_{t-\delta:t}\right\}_{p=1}^{N}\right)$$

to approximate the ground-truth $\{\mathbf{x}^{(p)}_{t:t+\delta}\}_{p=1}^{N}$, where the input data $\mathbf{x}^{(p)}_{t-\delta:t}$ is captured from streaming data flow at time stamp t. Note that the model $f_t(\cdot)$ indexed by t is updated online. The online setting is crucial because the online predictor can effectively adapt to the unseen, or highly dynamic data distribution.

In contrast, previous works [5], [15], [16], [20], [61] mostly consider an offline setting, which includes training and testing phases. In the training phase, a predictor is optimized based on a training dataset; in the testing phase, the predictor is fixed and deployed on a testing dataset. Mathematically, let $f(\cdot)$ be an offline predictor, $\mathcal{X}_{\text{train}}$ be the training dataset, and $\mathcal{X}_{\text{test}}$ be the testing dataset, the training and testing phases work as

$$\text{Training}: \left\{\widehat{\mathbf{x}}^{(p)}_{0:\delta}\right\}_{p=1}^{N} = f\left(\left\{\mathbf{x}^{(p)}_{-\delta:0}\right\}_{p=1}^{N}\right), \quad \mathbf{x}\sim\mathcal{X}_{\text{train}}$$

$$\text{Testing}: \left\{\widehat{\mathbf{y}}^{(p)}_{0:\delta}\right\}_{p=1}^{N} = f\left(\left\{\mathbf{y}^{(p)}_{-\delta:0}\right\}_{p=1}^{N}\right), \quad \mathbf{y}\sim\mathcal{X}_{\text{test}}.$$

These offline methods assume the training and testing time series should share the same data distribution and the resulting offline predictors would be hardly adapt to domain shift in the testing phase.

To design an online predictor, our core strategy is to exploit the dynamic mutual relations of agents and capture their evolution status through online learning. To model pairwise collaborations, we introduce a key concept, collaborative pair, which groups two agents and can be used to reflect the effect from one agent to another in a multi-agent system. For example, the pth agent $v^{(p)}$ and the qth agent $v^{(q)}$ form a collaborative pair $(v^{(p)}, v^{(q)})$ to indicate the directed effect from $v^{(q)}$ to $v^{(p)}$. In this way, to predict the future status of the pth agent, we consider $\widehat{\mathbf{x}}^{(p)}_{t:t+\delta} = f_t(\{\mathbf{x}^{(p)}_{t-\delta:t}, \mathbf{x}^{(q)}_{t-\delta:t}\}_{q=1}^{N})$, which exploits information from all the collaborative pairs of $v^{(p)}$ to benefit its prediction. This model considers each collaborative pair as a basic element, which is different from previous works that regard each individual agent as a basic element [15], [16].

IV. Collaborative Prediction Unit

To develop an online multi-agent forecasting model, we propose a CoPU as a basic module, which predicts the status of each agent from associated collaborative pairs via a collaborative graph.

A. Collaborative Graph

To exploit the interactions among agents, we propose a collaborative graph that explicitly models the

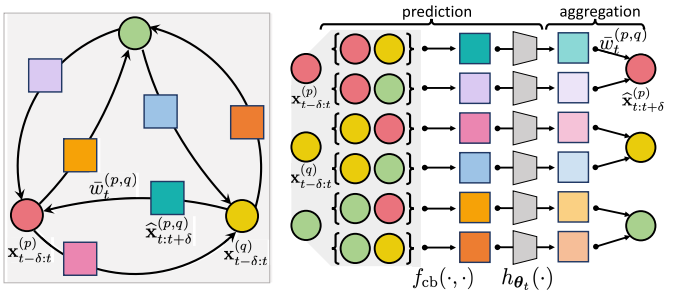


Fig. 2. CoPU sketched with three agents for example. Left: CoPU based on a collaborative graph to learn and aggregate information from collaborative pairs, where the circled nodes represent the agents and the squared nodes represents the collaborative pairs. Right: Detailed computation, where the collaborative pairs enable the late aggregation of agents' predictions.

pairwise correlations. Let $G(\mathcal{V},\mathcal{F},\mathbf{W}_t)$ be a collaborative graph, where $\mathcal{V}=\{v^{(1)},v^{(2)},\ldots,v^{(N)}\}$ is the agent set with $v^{(p)}$ modeling the pth agent, $\mathcal{F}=\{(v^{(p)},v^{(q)})\}_{p,q=1}^{N}$ is the set of collaborative pairs with $(v^{(p)},v^{(q)})$ modeling the (p,q)th collaborative pair; and $\mathbf{W}_t\in\mathbb{R}^{N\times N}$ is the collaborative graph adjacency matrix trained online, whose (p,q)th element $w_t^{(p,q)}$ reflects the influence level from $v^{(q)}$ to $v^{(p)}$ at time stamp t. Note that this collaborative graph is asymmetric and dynamic, that is, each collaborative edge represents a directional relationship between agents and each collaborative edge weight is time-varying during the online training process.

Different from many ordinary graphs, the collaborative graph has two types of nodes: agent and collaborative pair. Agents are actual nodes, while collaborative pairs are virtual nodes as they are naturally obtained when agents are given, which have directions. To predict the future status of each agent, we rely on each of its associated collaborative pairs to produce a possible future status for such an agent and then, the final status is obtained by averaging all the possible statuses based on the weighted collaborative edges.

Different from many previous graph-convolution-based and message-passing-based models [5]–[7], [15], [16], [61], where each node shares the same node feature to all of its neighbors without considering the demand of each one of its neighbors, our model, based on a collaborative pair, $v^{(q)}$ helps the prediction of $v^{(p)}$ by sharing customized information according to $v^{(p)}$'s status.

B. Model Design

Based on the collaborative graph, to construct the CoPU, we consider two key operations in series: collaborative-pair-based prediction, which predicts an agent's future status based on an associated collaborative pair, which generate all possible predictions with the collaborative effects between agents, and weighted collaborative-graph-based aggregation, which obtains the final prediction. Fig. 2 illustrates the CoPU, where the left plot shows the modeling thoughts, and the right plot shows the detailed computation.

1) Collaborative-Pair-Based Prediction: To explicitly form a collaborative pair, which represent the collaborative states of two agent in history, we propose a combination function $f_{\mathrm{cb}}(\cdot,\cdot)$ to directedly combine the measurements of two agents. Given the observed measurements $\mathbf{x}_{t-\delta:t}^{(p)},\mathbf{x}_{t-\delta:t}^{(q)}\in\mathbb{R}^{\delta\times d}$, the state of the collaborative pair $(v^{(p)},v^{(q)})$ is

$$\mathbf{x}_{t-\delta:t}^{(p,q)}=f_{\mathrm{cb}}\left(\mathbf{x}_{t-\delta:t}^{(p)},\mathbf{x}_{t-\delta:t}^{(q)}\right)\in\mathbb{R}^{\delta\times d'}. \tag{1}$$

The collaborative state $\mathbf{x}_{t-\delta:t}^{(p,q)}$ carries the directed effects from the agent $v^{(q)}$ to the agent $v^{(p)}$.

Based on the collaborative pairs and their states, we propose a collaborative predictor, which embeds any collaborative pair to predict an agent's future status under the corresponding effects. Let the collaborative predictor be $h_{\boldsymbol{\theta}_t}(\cdot)$, where $\boldsymbol{\theta}_t$ is the parameter trained online at time t. To predict $v^{(p)}$ under the collaborative effects from $v^{(q)}$, the predicted status is

$$\widehat{\mathbf{x}}_{t:t+\delta}^{(p,q)}=h_{\boldsymbol{\theta}_t}\left(\mathbf{x}_{t-\delta:t}^{(p,q)}\right)\in\mathbb{R}^{\delta\times d}. \tag{2}$$

The physical meaning of $\widehat{\mathbf{x}}_{t:t+\delta}^{(p,q)}$ is one possible future trajectory for $v^{(p)}$ by considering the influence from $v^{(q)}$.

For a collaborative pair, the aggregation function $f_{\mathrm{cb}}(\cdot,\cdot)$ and the predictor $h_{\boldsymbol{\theta}_t}(\cdot)$ could be designed in numerous forms. Here, we specifically consider three typical forms as follows.

1) *Autoregressive model:* Let $f_{\mathrm{cb}}(\cdot,\cdot)$ concatenate the multi-order differences of two agents, that is, $\mathbf{x}_{t-\delta:t}^{(p,q)}=[\mathbf{x}_{t-\delta:t}^{(p)},\ \mathbf{x}_{t-\delta:t}^{(q)}-\mathbf{x}_{t-\delta:t}^{(p)},\ldots,(\mathbf{x}_{t-\delta:t}^{(q)}-\mathbf{x}_{t-\delta:t}^{(p)})^{D}]\in\mathbb{R}^{\delta\times(D+1)d}$, which provides a polynomial distance feature. We then vectorize $\mathbf{x}_{t-\delta:t}^{(p,q)}$ by reshaping it along time and directly set a linear $h_{\boldsymbol{\theta}_t}(\cdot)$ to learn $\widehat{\mathbf{x}}_{t:t+\delta}^{(p,q)}$.
2) *LSTM model:* Let $f_{\mathrm{cb}}(\cdot,\cdot)$ concatenate the measurements of the two agents, that is, $\mathbf{x}_{t-\delta:t}^{(p,q)}=[\mathbf{x}_{t-\delta:t}^{(p)},\mathbf{x}_{t-\delta:t}^{(q)}]\in\mathbb{R}^{\delta\times 2d}$. The concatenation of the raw measurements reduces the dimensions and encourages more flexible information. We employ a one-layer long-short temporal memory (LSTM) as $h_{\boldsymbol{\theta}_t}(\cdot)$ on $\mathbf{x}_{t-\delta:t}^{(p,q)}$ to extract the temporal dynamics.
3) *Temporal convolution model:* We also consider $f_{\mathrm{cb}}(\cdot,\cdot)$ as the raw measurement concatenation. We then perform two layers of 1-D convolution along time to capture sequential dynamics. A $\tanh(\cdot)$ function is used as the activation after the first convolution to enhance flexibility.

The collaborative predictor should only consider every single collaborative pair, thus single collaborative effects are provided to the following collaborative-graph-based aggregation. A more plausible predictor structure could be explored in the future. Each agent has N collaborative pairs, so we predict N possible future statuses for one agent. To obtain the final prediction, we need to adaptively aggregate the N possibilities.

2) Collaborative-Graph-Based Aggregation: Given the possible future status for an agent, which is predicted based on each associated collaborative pair, we aggregate these statuses with different weights on the edge of the collaborative graph, where the edge weights are updated online and reflect the influence level from one agent to another. Mathematically, let the graph adjacency matrix at iteration step t be $\mathbf{W}_t$, whose the (p,q)th element is $w_t^{(p,q)}$. The final prediction of $v^{(p)}$ under the effects from all the associated collaborative pairs is formulated as

$$\widehat{\mathbf{x}}_{t:t+\delta}^{(p)}=\frac{\sum_{q=1}^{N}w_t^{(p,q)}\widehat{\mathbf{x}}_{t:t+\delta}^{(p,q)}}{\sum_{q'=1}^{N}w_t^{(p,q')}} \tag{3}$$

where the edge weights are normalized for any agent $v^{(p)}$. In all, the feed-forward operations of the CoPU are

formulated as

$$\widehat{\mathbf{x}}_{t:t+\delta}^{(p)} = \sum_{q=1}^{N} \bar{w}_t^{(p,q)} h_{\boldsymbol{\theta}_t}\left(f_{\text{cb}}\left(\mathbf{x}_{t-\delta:t}^{(p)}, \mathbf{x}_{t-\delta:t}^{(q)}\right)\right) \tag{4}$$

where $\bar{w}_t^{(p,q)} = w_t^{(p,q)} / \sum_{q'=1}^{N} w_t^{(p,q')}$ denotes the normalized collaborative edge weights.

Compared to most common graph-based models, which bridge the same types of nodes with a scalar edge weight, the proposed CoPU considers the heterogeneous nodes of agents and collaborative pairs, carrying individual dynamics the interactive effects in a hybrid manner. The CoPU essentially aggregates the comprehensive information from the collaborative pairs with different importance. Moreover, the proposed CoPU has a similar framework with the online expert mixture algorithm [56], [62], as the different collaborative predictors associated with an agent could be regarded as experts; however, each collaborative predictor explicitly exploits the customized information for the associated agent based on the corresponding collaborative pair, instead of producing an untargeted prediction independently like most expert mixture methods.

C. Optimization

CoPU enables a complete multi-agent forecasting pipeline, and the internal parameters could be trained given the prediction targets. To train the CoPU online, at each iteration t, we update the model parameters to minimize the loss function

$$\mathcal{L} = \sum_{t=1}^{T} \sum_{p=1}^{N} \mathcal{L}_t\left(\sum_{q=1}^{N} \bar{w}_t^{(p,q)} h_{\boldsymbol{\theta}_t}\left(\mathbf{x}_{t-\delta:t}^{(p,q)}\right), \mathbf{x}_{t:t+\delta}^{(p)}\right) \tag{5}$$

where $\mathcal{L}_t(\cdot)$ denotes a fixed and convex loss function evaluated at time t and T is the length of iterations before the current stamp. Since $\mathcal{L}_t(\cdot)$ is a convex function (e.g., ℓ_2 loss) with respect to $h_{\boldsymbol{\theta}_t}(\mathbf{x}_t^{(p,q)})$, we consider an upper bound of it

$$\mathcal{L} := \sum_{t=1}^{T} \sum_{p,q=1}^{N} \bar{w}_t^{(p,q)} \mathcal{L}_t\left(h_{\boldsymbol{\theta}_t}\left(\mathbf{x}_{t-\delta:t}^{(p,q)}\right), \mathbf{x}_{t:t+\delta}^{(p)}\right). \tag{6}$$

To minimize (6), instead of using common online gradient descent to update $\bar{w}_t^{(p,q)}$ and $\boldsymbol{\theta}_t$ together, we consider different updating mechanisms, which bring theoretical guarantees. We separately optimize the parameters $\boldsymbol{\theta}_t$ and $\bar{w}_t^{(p,q)}$ with different steps. First, at time t, the parameter $\boldsymbol{\theta}_t$ could be trained by online gradient decent, that is,

$$\boldsymbol{\theta}_{t+1} \leftarrow \boldsymbol{\theta}_t - \eta \nabla \mathcal{L}_t\left(h_{\boldsymbol{\theta}_t}^{(p,q)}\right)$$

where we use $\mathcal{L}_t(h_{\boldsymbol{\theta}_t}^{(p,q)})$ to simplify $\mathcal{L}_t(h_{\boldsymbol{\theta}_t}(\mathbf{x}_{t-\delta:t}^{(p,q)}), \mathbf{x}_{t:t+\delta}^{(p)})$; $\eta \in (0, 1)$ is the learning rate. Next, for $\bar{w}_t^{(p,q)}$, we employ a multiplicative exponentiated update, which is inspired from the randomized weighted majority of online learning [62]. We assume the optimal edge weight to be $w^{(p,q)} \in [0, 1]$, and $\bar{w}_t^{(p,q)}$ is updated through

$$\begin{aligned} w_{t+1}^{(p,q)} \leftarrow & \arg\min_{w^{(p,q)}} \eta \, \mathcal{L}_t\left(h_{\boldsymbol{\theta}_t}^{(p,q)}\right)\left(w^{(p,q)} - \bar{w}_t^{(p,q)}\right) \qquad (7) \\ & + \mathcal{D}_{\text{KL}}(w^{(p,q)} \| \bar{w}_t^{(p,q)}) \\ = & \ \bar{w}_t^{(p,q)} \exp\left(-\eta \, \mathcal{L}_t\left(h_{\boldsymbol{\theta}_t}^{(p,q)}\right)\right) \qquad (8) \end{aligned}$$

where $\mathcal{D}_{\text{KL}}(w^{(p,q)} \| \bar{w}_t^{(p,q)})$ denotes the KL-divergence between the optimized $w^{(p,q)}$ and the collaborative weight at the last iteration step. We tune $w^{(p,q)}$ based on two factors: 1) the prediction loss of the corresponding collaborative predictor, that is, $h_{\boldsymbol{\theta}_t}^{(p,q)}$ and 2) the edge weights of the collaborative graph at the last iteration step. Thus, the optimization ensures two facts. First, the lower prediction loss $\mathcal{L}_t$ and the better collaborative predictor leads to a higher weight between the considered two agents. Second, the continuity and stability of the collaborative graph structure are promoted by minimizing the element-wise distances and distribution divergence. We also normalize the new collaborative edge weights by $\bar{w}_{t+1}^{(p,q)} = w_{t+1}^{(p,q)} / \sum_{p=1}^{N} w_{t+1}^{(p,q)}$.

In previous GNNs [15]–[17], [21], graphs are obtained through either heuristic designs or end-to-end learning in a black-box, which lack a clear objective and theoretical justification. Compared to these works, the proposed method obtains the edge weights of the collaborative graph through optimizing an explicit objective function (7), which could be derived as an update function based on the input features and historical states in an online setting.

D. Comparison With Previous Works

The proposed CoPU is seemingly similar to spatio-temporal-graph convolution operations [14], [17], [21] and neural-message-passing operations [5]–[7]. Here, we clarify the differences.

Compared to spatio-temporal-graph convolution operations [14], [17], [21], the proposed CoPU is novel from the following three aspects.

1) *Interpretable graph learning:* Previous models usually build non-grid structures by using predefined graphs or adjusting the graph in an intractable black box. Each CoPU contains a graph explicitly optimized according to the performance of single predictors and structural continuity during training, reflecting more observable, controllable, and interpretable graphs.
2) *Customized feature propagation:* Previous works consider each node shares the same feature to all of its neighbors without considering the demands of neighbors. In CoPU, a node shares a customized feature to each neighbor via the collaborative pairs.
3) *Late fusion with collaborative graphs:* The late fusion considers the detailed collaborative effects, and it describes the specific role of each agent pair in the collaborative prediction in an accurate and informative manner.
4) *Interpretable output:* Previous methods build deep graph networks to learn the agent's dynamics in the hidden high-dimensional spaces, lacking the interpretation of feature representation, while each CoPU predicts agents' future statuses in the original measurement space, showing clearer physical meanings.

Previous neural-message-passing-based works [5]–[7] include two steps to propagate information: v2e, which transforms the information of a pair of nodes to an edge, and e2v, which aggregates the information of edges to a node. Compared to these methods, the proposed CoPU is novel from the following three aspects.

1) *Interpretable graph learning:* Previous works either build predefined graphs based on physical constraints or learn graph structures with errors and trials in black boxes, while the edge weights of our collaborative graph are obtained by optimizing an explicit objective function and has theoretical justification.
2) *Meaning of v2e:* In the v2e step, previous works update the embedding of each edge, which is an intermediate-feature-level fusion, while our CoPU directly generates the prediction of a node conditioned on another node, which is an output-level fusion.
3) *Meaning of e2v:* In the e2v step, previous works consider the mean aggregation, which assigns the same weight for each one of the neighbors, while our CoPU adopts the weighted aggregation, where the weights are the edge weights in the collaborative graph, obtained through multiplicative updating.

E. Theoretical Interpretation

To understand CoPU, we conduct regret analysis. We consider the best single collaborative predictor as our regret baseline and analyze the gap from our CoPU to this predictor. The best predictor has a straightforward meaning and its corresponding collaborative pair makes the most significant effect in predicting an agent's future status. We aim to demonstrate that the regret tends to be finite in the online setting, which indicates the properties of our CoPU and the training algorithm from the following two aspects.

1) The CoPU trained online could converge to a statistically stable state during the training process.
2) The performance of CoPU achieves a limited gap from the best single collaborative pair in statistics, reflecting that CoPU captures significant and reasonable collaborations for effective and interpretable prediction.

Here, we define the collaborative pair in the best collaborative predictor. For the pth agent, it has N collaborative pairs, $\{(v^{(p)}, v^{(q)})\}_{q=1}^{N}$. The collaborative pair $(v^{(p)}, v^{(q^*)})$ for $v^{(p)}$ in the best collaborative predictor is obtained via

$$q^* = \arg\min_{q} \sum_{t=1}^{T} \mathcal{L}_t\left(h_{\boldsymbol{\theta}^*}\left(\mathbf{x}_{t-\delta:t}^{(p,q)}\right)\right) \tag{9}$$

where $\boldsymbol{\theta}^*$ denotes the parameters of the best collaborative predictor at iteration step T. However, it is hard to implement an online model by directly determining $(v^{(p)}, v^{(q^*)})$ and meanwhile optimizing $\boldsymbol{\theta}^*$ that is why CoPU builds a collaborative graph with trainable weights to achieve a soft collaborative mechanism [see (4)].

To measure the performance gap between the best collaborative predictor and CoPU, we define the static regret based on the difference between the accumulated loss of CoPU and that of the best collaborative predictor in hindsight

$$\mathrm{R}_T = \frac{1}{T}\sum_{t=1}^{T}\left(\mathcal{L}_t\left(\sum_{q=1}^{N} \bar{w}_t^{(p,q)} h_{\boldsymbol{\theta}_t}^{(p,q)}\right) - \mathcal{L}_t\left(h_{\boldsymbol{\theta}^*}^{(p,q^*)}\right)\right) \tag{10}$$

where we simplify $h_{\boldsymbol{\theta}_t}(\mathbf{x}_t^{(p,q)})$ as $h_{\boldsymbol{\theta}_t}^{(p,q)}$. The regret reflects the gap between the CoPU and the best collaborative predictor, as well as the convergence rate of online learning.

To study the CoPU and the performance gap, we aim to calculate an upper bound of the regret. To facilitate the closed-form theoretical analysis, we consider a basic predictor (a linear autoregressive model) and propose the following assumptions that are commonly satisfied by our model.

(AS1) *For the collaborative predictor at time t, $h_{\boldsymbol{\theta}_t}$, given any collaborative pair, the loss $\mathcal{L}_t(h_{\boldsymbol{\theta}_t})$ is convex with respect to $\boldsymbol{\theta}_t$.*

(AS2) *For any bounded $\boldsymbol{\theta}_t$, where $\|\boldsymbol{\theta}_t\|_2 \le C_{\boldsymbol{\theta}}$, and any collaborative pair, we have a bounded loss, $\mathcal{L}_t(h_{\boldsymbol{\theta}_t}) \in [0, 1]$, and a bounded gradient, $\|\nabla\mathcal{L}_t(h_{\boldsymbol{\theta}_t})\|_2 \le L$.*

Then, we propose two lemmas, which help to derive the upper bounds of regret. We first analyze the loss gap between arbitrary collaborative predictors and the best collaborative predictor.

Lemma 1: Let $h_{\boldsymbol{\theta}^*}^{(p,q^*)}$ and $h_{\boldsymbol{\theta}_t}^{(p,k)}$ be the predictions of the best collaborative predictor and arbitrary one. Under **AS1** and **AS2**, for any collaborative pair $(v^{(p)}, v^{(k)})$, we have

$$\frac{1}{T}\sum_{t=1}^{T}\left(\mathcal{L}_t\left(h_{\boldsymbol{\theta}_t}^{(p,k)}\right) - \mathcal{L}_t\left(h_{\boldsymbol{\theta}^*}^{(p,q^*)}\right)\right) \le \frac{C_{\boldsymbol{\theta}}^2}{2\eta T} + \frac{\eta L^2}{2}$$

where η is the learning rate, $C_{\boldsymbol{\theta}}$ is the upper bound of the norm of parameters, and L is the Lipschitz constant.

Proof: Given any determined parameter $\boldsymbol{\theta}$ of the collaborative predictor, for any collaborative pair $(v^{(p)}, v^{(k)})$, we have

$$\|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}\|_2^2 = \|\boldsymbol{\theta}_t - \eta\nabla^\top\mathcal{L}_t\left(h_{\boldsymbol{\theta}_t}^{(p,k)}\right) - \boldsymbol{\theta}\|_2^2. \tag{11}$$

Considering the convexity property of the loss function (see **AS1**), the gradient of the loss function satisfies

$$\nabla^\top\mathcal{L}_t\left(h_{\boldsymbol{\theta}_t}^{(p,k)}\right) \ge \frac{\mathcal{L}_t\left(h_{\boldsymbol{\theta}_t}^{(p,k)}\right) - \mathcal{L}_t\left(h_{\boldsymbol{\theta}}^{(p,k)}\right)}{\boldsymbol{\theta}_t - \boldsymbol{\theta}}. \tag{12}$$

Plugging inequality (12) into (11), we could easily obtain the following inequality by rearranging some terms:

$$\begin{aligned}&\mathcal{L}_t\left(h_{\boldsymbol{\theta}_t}^{(p,k)}\right) - \mathcal{L}_t\left(h_{\boldsymbol{\theta}}^{(p,k)}\right)\\ &\quad\le \frac{\eta}{2}\left\|\nabla\mathcal{L}_t\left(h_{\boldsymbol{\theta}_t}^{(p,k)}\right)\right\|_2^2 + \frac{\|\boldsymbol{\theta}_t - \boldsymbol{\theta}\|_2^2 - \|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}\|_2^2}{2\eta}.\end{aligned} \tag{13}$$

We further consider the assumption **AS2** that points out the upper bound of $\|\boldsymbol{\theta}_t\|_2$ and the Lipschitz constant of the gradient $\nabla\mathcal{L}_t(h_{\boldsymbol{\theta}_t})$, that is, for any collaborative predictor, we have $\|\boldsymbol{\theta}_t\|_2 \le C_{\boldsymbol{\theta}}$ and $\|\nabla\mathcal{L}_t(h_{\boldsymbol{\theta}_t})\|_2 \le L$. We then summing inequality (13) over $t = 1, \ldots, T$, we could easily derive

$$\sum_{t=1}^{T}\left(\mathcal{L}_t\left(h_{\boldsymbol{\theta}_t}^{(p,k)}\right) - \mathcal{L}_t\left(h_{\boldsymbol{\theta}}^{(p,k)}\right)\right) \le \frac{C_{\boldsymbol{\theta}}^2}{2\eta} + \frac{\eta L^2 T}{2} \tag{14}$$

where we assume the initial parameter $\boldsymbol{\theta}_1$ to have a very small value, leading to $\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}\|_2^2 \approx \|\boldsymbol{\theta}\|_2^2 = C_{\boldsymbol{\theta}}^2$, and consider $\|\boldsymbol{\theta}_{T+1} - \boldsymbol{\theta}\|_2^2$ to be non-negative.

Additionally, we divide by the number of iterations T at the both side of inequality (14) to complete the proof

$$\frac{1}{T}\sum_{t=1}^{T}\left(\mathcal{L}_t\left(h_{\boldsymbol{\theta}_t}^{(p,k)}\right) - \mathcal{L}_t\left(h_{\boldsymbol{\theta}^*}^{(p,q^*)}\right)\right) \le \frac{C_{\boldsymbol{\theta}}^2}{2\eta T} + \frac{\eta L^2}{2}. \tag{15}$$

□

Note that Lemma 1 holds for any collaborative pair, so that, in hindsight, we could choose $\boldsymbol{\theta}$ to be the parameter at iteration step T and use the predicted status $h_{\boldsymbol{\theta}^*}^{(p,q^*)}$, which is output from the collaborative predictor corresponding to the single optimal collaborative pair $(v^{(p)}, v^{(q^*)})$.

We next analyze the gap between CoPU and an arbitrary collaborative predictor.

Lemma 2: Under **AS1** and **AS2**, for any collaborative pair $(v^{(p)}, v^{(k)})$ fed into the predictor $h_{\boldsymbol{\theta}_t}(\cdot)$, it holds that

$$\frac{1}{T}\sum_{t=1}^{T}\left(\sum_{q=1}^{N}\bar{w}_t^{(p,q)}\mathcal{L}_t\left(h_{\boldsymbol{\theta}_t}^{(p,q)}\right)-\mathcal{L}_t\left(h_{\boldsymbol{\theta}_t}^{(p,k)}\right)\right)\leq\eta+\frac{\log N}{\eta T}$$

where N is the number of agents.

Proof: We first let $W_t^{(p)}$ be the sum of edge weights associated with agent $v^{(p)}$, that is, $W_t^{(p)}=\sum_{q=1}^{N}w_t^{(p,q)}$. Therefore, $W_t^{(p)}$ adjusted through exponentiated update at the time $t+1$ could be formulated as

$$W_{t+1}^{(p)}=\sum_{q=1}^{N}\bar{w}_t^{(p,q)}\exp\left(-\eta\mathcal{L}_t\left(h_{\boldsymbol{\theta}_t}^{(p,q)}\right)\right). \tag{16}$$

We represent $\mathcal{L}_t\left(h_{\boldsymbol{\theta}_t}^{(p,q)}\right)$ as $\mathcal{L}_t$ for simplification. We consider the inequality $\exp(-\eta x)\leq 1-\eta x+\eta^2x^2$ for $\forall|\eta|\leq 1$, and inequality (16) satisfies

$$W_{t+1}^{(p)}\leq 1-\eta\sum_{q=1}^{N}\bar{w}_t^{(p,q)}\mathcal{L}_t+\eta^2\sum_{q=1}^{N}\bar{w}_t^{(p,q)}\mathcal{L}_t^2.$$

Furthermore, according to $1+x\leq e^x$, we have

$$W_{t+1}^{(p)}\leq\exp\Big(-\eta\sum_{q=1}^{N}\bar{w}_t^{(p,q)}\mathcal{L}_t+\eta^2\sum_{q=1}^{N}\bar{w}_t^{(p,q)}\mathcal{L}_t^2\Big). \tag{17}$$

And we then consider the updated situation of inequality (17) along time when $t=1,\ldots,T$, we then obtain

$$W_{T+1}^{(p)}\leq\exp\Big(\sum_{t=1}^{T}\eta^2\sum_{q=1}^{N}\bar{w}_t^{(p,q)}\mathcal{L}_t^2-\eta\sum_{q=1}^{N}\bar{w}_t^{(p,q)}\mathcal{L}_t\Big). \tag{18}$$

On the other hand, for the agent $v^{(p)}$ and any agent $v^{(k)}$, we have the inequality

$$\begin{aligned}W_{T+1}^{(p)}&\geq w_{T+1}^{(p,k)}\\&=w_1^{(p,k)}\exp\left(-\eta\sum_{t=1}^{T}\mathcal{L}_t\left(h_{\boldsymbol{\theta}_t}^{(p,k)}\right)\right).\end{aligned} \tag{19}$$

Combining inequality (18) and inequality (19), we arrive at

$$\begin{aligned}&\exp\Big(-\eta\sum_{t=1}^{T}\sum_{q=1}^{N}\bar{w}_t^{(p,q)}\mathcal{L}_t+\eta^2\sum_{t=1}^{T}\sum_{q=1}^{N}\bar{w}_t^{(p,q)}\mathcal{L}_t^2\Big)\\&\qquad\geq w_1^{(p,k)}\exp\left(-\eta\sum_{t=1}^{T}\mathcal{L}_t\left(h_{\boldsymbol{\theta}_t}^{(p,k)}\right)\right).\end{aligned} \tag{20}$$

Taking the logarithm on both sides of inequality (20) and meanwhile considering $w_1^{(p,k)}=1/N$, we could obtain

$$\sum_{t=1}^{T}\sum_{q=1}^{N}\bar{w}_t^{(p,q)}\mathcal{L}_t\left(h_{\boldsymbol{\theta}_t}^{(p,q)}\right)-\sum_{t=1}^{T}\mathcal{L}_t\left(h_{\boldsymbol{\theta}_t}^{(p,k)}\right)\leq\eta T+\frac{\log N}{\eta} \tag{21}$$

where we note that $\mathcal{L}_t(h_{\boldsymbol{\theta}_t}^{(p,q)})\in[0,1]$ according to **AS2**, and Lemma 2 has been proved, where we could divide T on both sides of the inequality (21). □

Lemma 2 shows that the performance gap between an ensemble of collaborative predictors with the trainable weight $\bar{w}_t^{(p,q)}$ and an arbitrary collaborative predictor fed with any collaborative pairs $(v^{(p)}, v^{(k)})$, including the best predictor, can be upper bounded. Combining Lemmas 1 and 2, we derive the upper bound of the regret as in Theorem 1.

Theorem 1: Under **AS1** and **AS2**, and with the analysis of Lemmas 1 and 2, the regret of the proposed CoPU satisfies the following bound:

$$\begin{aligned}&\frac{1}{T}\left(\sum_{t=1}^{T}\mathcal{L}_t\left(\sum_{q=1}^{N}\bar{w}_t^{(p,q)}h_{\boldsymbol{\theta}_t}^{(p,q)}\right)-\sum_{t=1}^{T}\mathcal{L}_t\left(h_{\boldsymbol{\theta}^*}^{(p,q^*)}\right)\right)\\&\qquad\leq\frac{\log N}{\eta T}+\frac{C_{\boldsymbol{\theta}}^2}{2\eta T}+\frac{\eta L^2}{2}+\eta.\end{aligned}$$

Setting $\eta=\mathcal{O}(1/(T)^{1/2})$, the static regret converges with $\mathrm{R}_T=\mathcal{O}(1/(T)^{1/2})$.

Proof: Combining inequalities (15) and (20), we could obtain that

$$\begin{aligned}&\frac{1}{T}\left(\sum_{t=1}^{T}\sum_{q=1}^{N}\bar{w}_t^{(p,q)}\mathcal{L}_t\left(h_{\boldsymbol{\theta}_t}^{(p,q)}\right)-\sum_{t=1}^{T}\mathcal{L}_t\left(h_{\boldsymbol{\theta}^*}^{(p,q^*)}\right)\right)\\&\qquad\leq\frac{\log N}{\eta T}+\frac{C_{\boldsymbol{\theta}}^2}{2\eta T}+\frac{\eta L^2}{2}+\eta.\end{aligned} \tag{22}$$

Note that the loss function $\mathcal{L}_t(h_{\boldsymbol{\theta}_t}^{(p,q)})$ is convex with respect to any $h_{\boldsymbol{\theta}_t}^{(p,q)}$, thus the loss function $\mathcal{L}_t(\sum_{q=1}^{N}\bar{w}_t^{(p,q)}h_{\boldsymbol{\theta}_t}^{(p,q)})$ has an upper bound $\sum_{q=1}^{N}\bar{w}_t^{(p,q)}\mathcal{L}_t(h_{\boldsymbol{\theta}_t}^{(p,q)})$, that is,

$$\mathcal{L}_t\left(\sum_{q=1}^{N}\bar{w}_t^{(p,q)}h_{\boldsymbol{\theta}_t}^{(p,q)}\right)\leq\sum_{q=1}^{N}\bar{w}_t^{(p,q)}\mathcal{L}_t\left(h_{\boldsymbol{\theta}_t}^{(p,q)}\right) \tag{23}$$

understanding that inequality (21) satisfies all the collaborative pairs in the online system, we could easily prove Theorem 1 to represent the upper bound of the static regret. Plugging (23) in (22), we could easily prove Theorem 1. □

Therefore, the regret of CoPU is upper bounded during online training. Additionally, when T becomes large, the regret converges to a very small value, reflecting the finite gap between the CoPU and the best single collaborative predictor. We also validate Theorem 1 by computing the regret along time in our experiment [see Fig. 6(a)].

V. Collaborative Graph Neural Networks

Though one CoPU can tackle the prediction task, in the experiments, we find that stacking multiple CoPUs can lead to even better performance. Therefore, we develop a deep architecture called CoGNNs, which employs multiple CoPUs as computational building blocks.

A. Network Architecture

We sketch the architecture of the proposed CoGNN in Fig. 3, where we stack three CoPUs in cascade, for example. Each CoPU in CoGNN progressively fine-tunes the predictions and approximates the target measurements step-by-step.

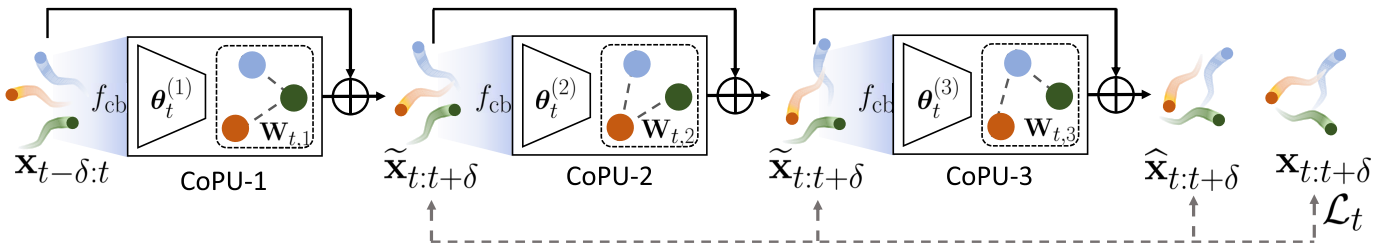


Fig. 3. Architecture of the CoGNN. The model takes multiple time-series as inputs and aims to generate the future series. Each graph-temporal prediction unit (CoPU) in the model performs three operations for progressive prediction. The entire model is trained online for real-time prediction. Here, we show three CoPUs to sketch the model architecture.

According to the exemplar Fig. 3, the first and the second CoPUs produce the immediate prediction results $\widetilde{\mathbf{x}}_{t:t+\delta}$ for all the agents from the input clips or the first CoPU; the last CoPU generates the final prediction $\widehat{\mathbf{x}}_{t:t+\delta}$ from the results of the second CoPU. For each CoPU, we apply a residual connection between the input and the output, reflecting that the internal modules estimate the displacements from the input data to the target.

To train the network, we apply the same form of loss function on each CoPU, because each CoPU tackles a complete prediction task and could be trained directly (see the dashed lines in gray in Fig. 3). Moreover, the collaborative edge weights in each CoPU could be learned to directly optimize an explicit objective function, (7), through the exponentiated update.

B. Comparison With Previous Works

Compared to previous graph-based prediction networks [5]–[7], [14], [17], [21], our model contributes from two aspects.

First, to our best knowledge, the proposed CoGNN is the first online-learning method, which is learned and tested along with the streaming data and never stops. This is crucial because an online method could capture the highly dynamic patterns of the real-time series and reduce the training/testing gaps between the training and testing samples in practice. Our CoGNN is scalable and easy to train, which is suitable for online learning. However, previous methods only sample the multi-agent systems as sets of independent clips but neglect the real-time dynamics and complexity.

Second, the proposed CoGNN is more interpretable. The output of each layer in the proposed CoGNN predicts agents' future statuses in the original measurement space, reflecting a clear physical meaning. With a cascaded structure, our CoGNN can progressively fine-tune the time-series prediction, achieving effective and stable performance. However, previous methods use hierarchical graph propagation to extract hidden representations without interpretable gradual error reduction.

Finally, according to Theorem 1, we theoretically demonstrate the convergence and potential performance of the model with a finite upper bound in online learning. However, previous methods do not guarantee their convergence and derive the regret values after a long period of training.

C. Analysis of Computational Complexity

CoGNNs consider the collaborative effects between any pairs of agents in the entire dynamic system. This design would cause $O(N^2)$ spatial complexity given N agents, which limits the computation in very large-scale systems. However, here we emphasize that when a system does not have a predefined graph, that is, a flexible and implicit graph needs to be inferred from the data, most existing models suffer from $O(N^2)$ computational complexity. For example, some neural-message-passing methods use neural networks to model connection features of any two agents [5]–[7]. Some kernel-based or self-attention-based methods calculate the affinity between any two agents [13], [14], [21], [31]; and so on. To address this problem of the heavy costs in large systems, we could utilize some approximation methods to constrain the system scopes. The first possible operation is agent downsampling. We could randomly sample percentages of agents multiple times to construct subgraphs to approximate the system. Each subgraph has $N' \ll N$ agents to reduce the complexity. The second method is to introduce some spatial prior, because, in many physical systems, closer neighbors would provide higher influence. For example, in the task of vehicle trajectory prediction, a vehicle would mainly consider the nearby vehicles to guide its control. Therefore, we could part the system into multiple local clusters according to the spatial distribution, and we learn the dynamics of these sub-systems and further fuse them for the final prediction.

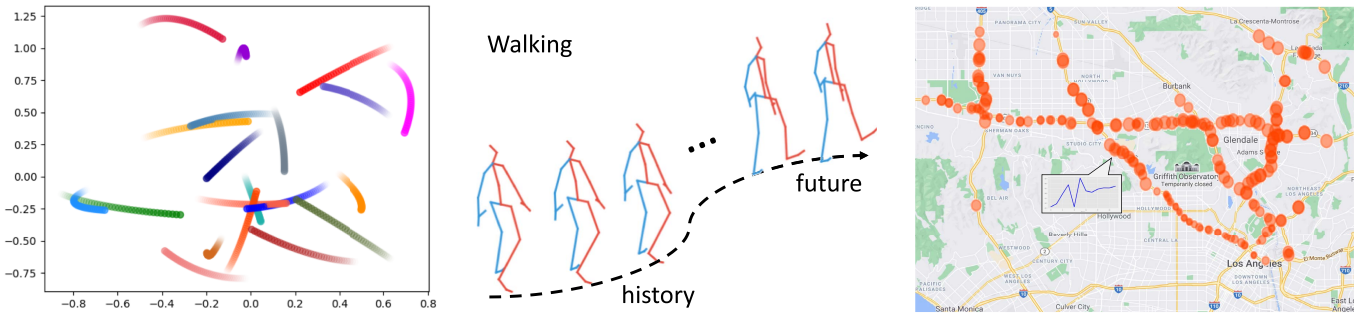
VI. Experiments

To evaluate the proposed CoGNNs, we conduct extensive experiments on three important multi-agent forecasting tasks in an online setting, that is, online simulated trajectory prediction, online human motion prediction and online traffic speed prediction, where the corresponding scenarios and data examples are sketched in Fig. 4.

In CoGNNs, we, respectively, use three forms of collaborative predictors $h_{\boldsymbol{\theta}_t}(\cdot)$ in CoPUs (see Section IV-B). We note the three forms of models as "CoGNN-AR," "CoGNN-LSTM," and "CoGNN-TC," respectively. In CoGNN-AR, to compute the state, $\mathbf{x}_{t-\delta:t}^{(p,q)}$, of any collaborative pair $(v^{(p)}, v^{(q)})$, the order of measurement difference $D = 10$; for CoGNN-LSTM and CoGNN-AR, the hidden dimension of $h_{\boldsymbol{\theta}_t}(\cdot)$ is 64. The entire CoGNNs are built with two CoPUs, which enable precise and stable forecasting. We train the models on one Nvidia Tesla V100 GPU with PyTorch 1.4 framework. The learning rates are from 0.01 to 0.075 for different forms of CoGNNs. We also apply gradient clipping to constrain the absolute values of gradients within 10 for robustness.

A. Online Simulated Trajectory Prediction

We first evaluate the proposed CoGNNs in online simulated particle systems, which contains several moving particles



(a) (b) (c)

Fig. 4. Data examples of three scenarios of online multi-agent forecasting. (a) Simulated interaction systems: we generate systems based on 20 moving agents with various interactions, and we forecast the trajectory of each agent given the historical movements in an online setting. (b) Human motion: the human motion sequences consist of 3-D skeleton-based human poses performing a certain action, for example, walking, and we generate the future poses given the historical poses online. (c) Traffic speed distribution in cities: at different locations, there are sensors recording the traffic speed in real-time, that is, the traffic speed at each point is a continuous time-series (see the curve), while the larger sizes of the red circles denote the higher traffic speed at a certain time step. We aim to predict the traffic speed given the historical speed online.

whose positions and velocities are determined by their interactions.

1) Dataset: In our simulated particle systems, there are 20 particle agents, whose initial states (positions and velocities) are set randomly, moving in a 2-D space with dynamic mutual interaction. The interactions among the agents could be represented as a graph structure with symmetric adjacency matrix. Here, we use a simple interaction rule called "spring" to construct the complex systems, that is, each agent has attractions with several other agents that are randomly selected. One example of our simulated system is illustrated in Fig. 4(a). This system was similarly used by [5], where only five agents are set in a simple system. However, different from the previous work [5] which generates a set of independent trajectory sequences with fixed and known interactions, in our simulated systems, agents have randomly changed interactions with different agents to reflect dynamic and changing states online. In each interactive simulation, we change the interaction structures for 20 times randomly, that is, the interaction adjacency matrix is randomly adjusted for 20 times. Moreover, we conduct ten times of different simulation to demonstrate the effectiveness and robustness. In our experiments, along the streaming data, our CoGNNs are fed with each 10-frame clip one by one along time and generate the next 10-frame clip for prediction.

2) Baselines and Evaluation Metrics: We compare CoGNNs with several methods on trajectory prediction, including NRI [5], dynamic NRI (DNRI) [6], and EvolveGraph [7]. These methods are designed for the offline scenarios; thus, we modify these works to be run online with the same inputs and outputs as our CoGNNs. We also adjust their hyper-parameters to be reasonable. Besides being compared to the state-of-the-art works, CoGNNs are also compared to two degenerated variants in the following.

1) *CoGNNs (no $\mathbf{W}_t$):* We forecast by using only h_{θ_t} on individual agents without considering the collaborative pairs and collaborative graphs.
2) *CoGNNs (e2e $\mathbf{W}_t$):* We use the collaborative graphs, however, the edge weight matrix $\mathbf{W}_t$ is trained via online gradient descent through the back-propagation end-to-end, instead of the exponential update.

To evaluate various methods, we employ the average mean square errors (MSEs) between the predicted locations and ground-truth of all the agents over the training/testing process, as well as we test the running time of each method.

3) Results: Table I presents the average forecasting results over the online learning processes, where we show the prediction performance at five prediction steps of the generated 10-frame clips, respectively. Since all the models are run on ten different interaction systems, the mean values and standard deviations of the prediction MSEs are presented for comparison. Moreover, Table I presents the average time costs of various methods at one online training/testing iteration. We see that: 1) compared to the state-of-the-art methods, the proposed CoGNNs achieve more precise forecasting and outperform the baselines, especially CoGNN-LSTM and CoGNN-TC achieve much more effective prediction at various prediction steps; 2) compared to the model variants, "no $\mathbf{W}_t$" and "e2e $\mathbf{W}_t$," CoGNNs consistently outperforms these two variants, demonstrating the importance and effectiveness of our interpretable collaborative graphs and the exponentiated update strategy; and 3) compared to the previous models, CoGNNs achieve much lower time costs at each iteration, reflecting the lightweight, efficiency, and deployability in an online setting.

B. Online Human Motion Prediction

We next conduct experiments on online human motion prediction to verify the effectiveness of our CoGNN.

1) Dataset: Various models are trained and tested on two motion capture datasets: Human3.6M (H3.6M) [63] and CMU Mocap.[1] H3.6M contains 15 classes of activities. There are 32 body joints with their 3-D coordinates. An exemplar motion clip in H3.6M is shown in Fig. 4(b). CMU Mocap contains 8 activities with 38 joints. Following the previous settings [15], [26], [64], we downsample all motion sequences along time by two for both datasets. We take the motion clips of the past 400 ms (ten frames) as input and forecast the future 400 ms.

2) Baselines and Evaluation Metrics: We compare our model to state-of-the-art works, including two non-graph methods, Res-sup [26], and convolutional sequence-to-sequence model (CSM) [64], and six graph-based methods,

[1] http://mocap.cs.cmu.edu/

TABLE I

PREDICTION RESULTS (MEAN VALUES AND STANDARD DEVIATIONS OF MSES) ON THE TASKS OF ONLINE SIMULATED PREDICTION, WHERE WE CONSIDER TEN INTERACTION SYSTEMS WITH 20 TRAJECTORIES. WE ALSO COMPARE THE VARIANTS OF COGNNS: COGNNS (NO $\mathbf{W}_t$), WHICH PREDICT EACH AGENT BASED ON THIS AGENT ITSELF WITHOUT USING THE COLLABORATIVE GRAPH; AND COGNNS (E2E $\mathbf{W}_t$), WHICH TRAIN THE COLLABORATIVE GRAPH THROUGH STANDARD GRADIENT DESCENT IN AN END-TO-END SETTING

	Methods	Prediction steps 1	2	5	8	10	Time/Iter (ms)
Baselines	ZeroV	13.43 ± 1.94	13.43 ± 1.95	13.48 ± 2.08	13.58 ± 2.35	13.69 ± 2.64	-
	NRI [5]	0.56 ± 0.16	0.58 ± 0.15	0.60 ± 0.18	1.03 ± 0.30	1.59 ± 0.46	14.06
	DNRI [6]	0.26 ± 0.10	0.28 ± 0.12	0.31 ± 0.12	0.51 ± 0.15	0.77 ± 0.19	20.33
	EvolveGraph [7]	0.22 ± 0.10	0.24 ± 0.09	0.28 ± 0.10	0.44 ± 0.13	0.68 ± 0.15	22.84
no $\mathbf{W}_t$	CoGNN-AR (no $\mathbf{W}_t$)	0.23 ± 0.06	0.24 ± 0.07	0.41 ± 0.12	0.75 ± 0.23	1.09 ± 0.32	6.73
	CoGNN-LSTM (no $\mathbf{W}_t$)	0.16 ± 0.04	0.16 ± 0.05	0.20 ± 0.05	0.40 ± 0.14	0.61 ± 0.16	11.96
	CoGNN-TC (no $\mathbf{W}_t$)	0.19 ± 0.08	0.23 ± 0.11	0.24 ± 0.11	0.42 ± 0.16	0.63 ± 0.18	8.52
e2e $\mathbf{W}_t$	CoGNN-AR (e2e $\mathbf{W}_t$)	0.20 ± 0.04	0.21 ± 0.06	0.37 ± 0.15	0.74 ± 0.22	1.06 ± 0.34	6.88
	CoGNN-LSTM (e2e $\mathbf{W}_t$)	0.17 ± 0.06	0.16 ± 0.06	0.19 ± 0.07	0.37 ± 0.13	0.60 ± 0.18	12.08
	CoGNN-TC (e2e $\mathbf{W}_t$)	**0.15** ± 0.05	0.18 ± 0.07	0.20 ± 0.07	0.36 ± 0.12	0.58 ± 0.17	8.71
Ours	CoGNN-AR	0.19 ± 0.06	0.20 ± 0.06	0.37 ± 0.12	0.70 ± 0.21	1.01 ± 0.30	6.86
	CoGNN-LSTM	**0.15** ± 0.05	**0.15** ± 0.06	0.19 ± 0.05	**0.35** ± 0.10	0.59 ± 0.17	12.11
	CoGNN-TC	0.16 ± 0.06	0.18 ± 0.05	**0.18** ± 0.06	**0.35** ± 0.12	**0.56** ± 0.16	8.60

TABLE II

AVERAGE MPJPES OF COGNNS AND BASELINES FOR MOTION PREDICTION ON FOUR REPRESENTATIVE ACTIONS OF H3.6M. WE ALSO COMPARE TWO TYPES OF VARIANTS OF COGNNS, THAT IS, "NO $\mathbf{W}_t$" AND "E2E $\mathbf{W}_t$"; SEE THE DEFINITION OF THESE TWO VARIANTS IN THE CAPTION OF TABLE I

	Motions	Walking				Eating				Smoking				Discussion				Time/Iter (ms)
	Methods	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400	
Baselines	Res-sup [26]	23.0	32.1	37.8	44.4	20.6	30.2	37.1	46.0	23.2	27.6	35.8	42.7	30.1	41.8	46.8	52.0	14.35
	NRI [5]	21.6	30.4	36.0	42.4	19.5	28.7	36.4	42.5	22.4	26.8	33.2	40.1	28.7	39.3	44.9	51.3	16.72
	CSM [64]	22.1	31.5	39.4	43.3	19.9	29.1	39.7	43.9	21.3	26.5	32.4	38.6	31.2	38.0	44.4	50.8	22.36
	Traj-GCN [15]	22.9	30.1	38.9	43.6	19.0	24.9	35.8	38.6	19.7	25.3	28.2	35.9	24.1	36.2	43.4	47.3	69.55
	DNRI [6]	22.3	31.8	39.2	43.7	19.3	27.6	36.7	40.5	20.5	26.3	29.1	36.7	26.8	37.2	43.8	49.0	24.51
	DMGNN [17]	22.6	31.1	39.2	44.1	18.2	24.2	32.5	39.2	19.9	24.3	27.8	35.7	24.6	35.8	43.4	49.5	63.18
	HisRep [65]	22.6	29.0	37.3	42.1	17.3	22.5	33.6	37.3	19.2	25.4	29.9	36.3	22.8	33.6	40.7	46.5	76.77
	EvolveGraph [7]	22.2	30.7	38.3	43.5	18.1	23.8	34.3	38.9	20.4	26.0	29.7	37.1	24.2	34.7	42.9	48.4	28.25
no $\mathbf{W}_t$	CoGNN-AR (no $\mathbf{W}_t$)	26.2	39.9	62.3	65.4	20.3	27.8	43.0	48.7	23.2	30.5	46.0	52.7	24.8	35.6	55.7	63.7	9.04
	CoGNN-LSTM (no $\mathbf{W}_t$)	23.3	23.9	**29.8**	40.1	19.8	20.0	26.4	35.0	20.3	20.8	28.0	33.9	29.5	30.0	40.6	48.9	13.48
	CoGNN-TC (no $\mathbf{W}_t$)	25.8	27.3	37.2	47.2	19.6	20.1	30.1	38.3	17.8	19.1	29.7	36.3	22.2	25.3	36.7	48.3	10.25
e2e $\mathbf{W}_t$	CoGNN-AR (e2e $\mathbf{W}_t$)	24.9	35.1	50.5	57.9	17.6	26.8	38.7	40.3	22.3	28.2	43.6	48.7	24.9	34.3	51.7	58.6	9.09
	CoGNN-LSTM (e2e $\mathbf{W}_t$)	23.5	23.7	33.7	40.9	18.4	17.0	27.6	34.1	19.4	20.0	27.3	24.2	28.4	29.6	40.1	47.5	13.64
	CoGNN-TC (e2e $\mathbf{W}_t$)	22.6	25.2	35.3	41.7	18.3	19.9	29.2	35.7	18.0	18.8	28.4	35.5	22.0	26.7	36.1	47.5	10.38
Ours	CoGNN-AR	22.3	33.7	51.5	56.8	**16.6**	23.0	36.5	41.5	19.5	27.7	41.6	45.4	24.1	30.1	50.4	58.0	9.11
	CoGNN-LSTM	22.6	23.3	33.6	40.7	18.2	19.2	**26.1**	**33.7**	19.7	19.4	**25.2**	33.8	28.7	29.3	38.9	47.3	13.60
	CoGNN-TC	**21.5**	**21.4**	31.0	**39.7**	16.8	**16.2**	28.7	36.0	**16.4**	**16.6**	26.6	**33.6**	**19.9**	**18.5**	**34.1**	**45.2**	10.36

NRI [5], Traj-GCN [15], DNRI [6], DMGNN [17], HisRep [65], and EvolveGraph [7]. For these offline methods, we also adjust them to take the same streaming data as our CoGNNs and generate future poses as effectively as possible in an online setting. For evaluation, we calculate the mean per joint position error (MPJPE) between each predicted pose and the corresponding ground-truth.

3) Results: We first evaluate various methods on H3.6M. Besides being compared to the state-of-the-art works, CoGNNs are also compared to two degenerated variants: that is, "CoGNN (no $\mathbf{W}_t$)" and "CoGNN (e2e $\mathbf{W}_t$)"; see introduction in the experiments of online simulated trajectory prediction. Table II shows the forecasting MPJPEs on four representative actions: "walking," "eating," "smoking," and "discussion," presenting the average MPJPEs at the future 80, 160, 320, and 400 ms. Also, Table II shows the running time costs; that is, the average time at each training/testing iteration during online learning. We see that: 1) compared to the state-of-the-art methods, CoGNNs achieve more precise forecasting and outperform the baselines; 2) compared to the model variants, "no $\mathbf{W}_t$" and "e2e $\mathbf{W}_t$," CoGNNs consistently outperforms them, demonstrating the importance of our interpretable collaborative graphs and the exponentiated update strategy; and 3) at each iteration of online learning, CoGNNs spend much less running time than baselines. See the prediction of another 11 actions in the Appendix.

To further qualitatively evaluate our CoGNNs, we visualize the future poses produced by various methods for human motion prediction on the H3.6M dataset. We illustrate the motions of a clip of "walking," which are predicted by different models at certain iteration step, respectively, during the online learning. We compare one existing model, DMGNN [17] and two variants of CoGNN: CoGNN-LSTM and CoGNN-TC (see Fig. 5). We see that, for the baseline model, DMGNN, there are large errors after the 160th ms of the prediction time, where the left arm on the human body cannot bend flexibly. As for the CoGNN-LSTM and CoGNN-TC, the generated poses are much closer to the ground truths than the DMGNN model.

Additionally, we test various methods on another large-scale dataset, CMU Mocap, for online motion prediction. The average prediction results (MPJPE) across all the actions at the future 80, 160, 320, and 400 ms as well as the running time costs are presented in Table III. We see that: 1) the proposed CoGNNs outperform the state-of-the-art methods with a large margin in terms of prediction performance and running

TABLE III

AVERAGE MPJPEs OF CoGNNs AND BASELINES FOR MOTION PREDICTION ON CMU MOCAP. WE ALSO COMPARE TWO TYPES OF VARIANTS OF CoGNNs, THAT IS, "NO $\mathbf{W}_t$" AND "e2e $\mathbf{W}_t$"; SEE THE DEFINITION OF THESE TWO VARIANTS IN THE CAPTION OF TABLE I

	Motions	Average				Time/Iter (ms)
	Methods	80	160	320	400	
Baselines	Res-sup [26]	22.9	30.7	40.4	47.4	16.38
	CSM [64]	21.0	28.5	35.6	41.7	24.14
	NRI [5]	19.7	26.7	33.6	41.1	19.56
	Traj-GCN [15]	17.3	24.5	31.8	37.9	71.92
	DNRI [6]	18.3	25.5	32.4	38.9	28.08
	DMGNN [17]	16.8	23.8	30.6	37.7	66.15
	HisRep [65]	15.6	21.9	28.7	35.2	81.05
	EvolveGraph [7]	17.0	24.1	33.4	38.6	29.73
no $\mathbf{W}_t$	CoGNN-AR (no $\mathbf{W}_t$)	27.1	34.7	52.7	57.0	8.33
	CoGNN-LSTM (no $\mathbf{W}_t$)	18.6	21.0	30.0	37.2	10.29
	CoGNN-TC (no $\mathbf{W}_t$)	17.0	19.1	28.4	34.7	9.02
e2e $\mathbf{W}_t$	CoGNN-AR (e2e $\mathbf{W}_t$)	23.1	30.6	45.4	51.2	8.51
	CoGNN-LSTM (e2e $\mathbf{W}_t$)	16.9	18.7	24.6	30.2	10.44
	CoGNN-TC (e2e $\mathbf{W}_t$)	16.4	18.2	25.4	31.4	9.09
Ours	CoGNN-AR	21.6	28.5	43.5	50.3	8.49
	CoGNN-LSTM	16.7	17.7	**23.3**	29.0	10.37
	CoGNN-TC	**15.4**	**17.2**	23.8	**28.5**	9.13

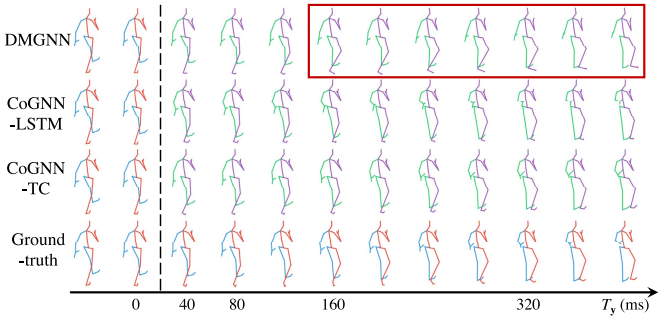


Fig. 5. Predicted samples for human motion prediction on the action of "walking" on H3.6M dataset.

efficiency and 2) the proposed CoGNNs achieve consistently better performances than the other two model variants to show the effectiveness of our collaborative mechanism. The detailed predictions of each activity are presented in the Appendix.

C. Online Traffic Speed Prediction

Furthermore, the proposed CoGNNs are also evaluated on the tasks of online traffic speed prediction.

1) Datasets: We conduct experiments on four large-scale traffic datasets in the real world, including the traffic data of Los Angeles county [67] or different regions of the California highway [68] as follows.

1) Los Angeles Metropolitan Transportation (METR-LA) contains traffic speed collected by 207 sensors around Los Angeles from March to June in 2012. 2) Performance measurement system at bay (PeMS-BAY) contains traffic speed at Bay Area collected by 325 sensors on CalTrans performance measurement system (PeMS) from January to May in 2017. 3) PeMS-D4 contains traffic speed collected by 307 sensors at San Francisco from January to February in 2018. 4) PeMS-D8 contains traffic speed collected by 170 sensors in San Bernardino from July to August in 2016.

We illustrate the traffic speed distribution of METR-LA in Fig. 4(c) as an example, where we denote that each sensor captures the dynamic traffic speed along time. For these

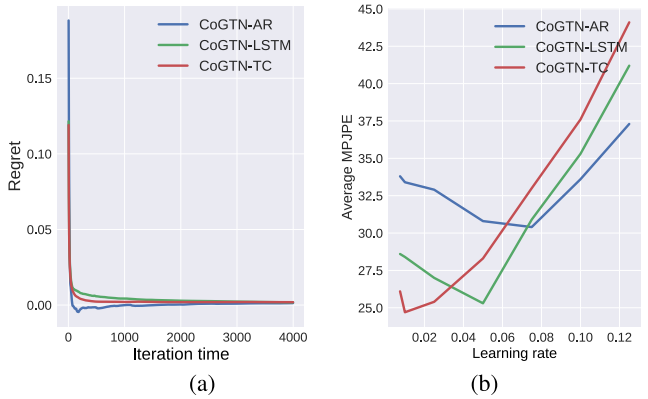


Fig. 6. Regret curves and the performance with various learning rates of CoGNNs for online human motion prediction. (a) Regret curves. (b) Effects of learning rates.

datasets, we aggregate the data sequence into 5-min intervals from 30-s frequency. We take the past 12-frame-length (1 h) sequence as the input and forecast the next 12 frames.

2) Baselines and Evaluation Metrics: To evaluate the effectiveness our CoGNNs on the online traffic speed prediction, we introduce several state-of-the-art models for comparison, including ST-GCN [14], DCRNN [18], attention-based spatial-temporal graph convolutional network (ASTGCN) [13], G-WaveNet [31], spatial-temporal synchronous graph convolutional network (STSGCN) [4], and adaptive graph convolutional recurrent network (AGCRN) [66]. To test various methods, we utilize three metrics, which are mean absolute error (MAE), root MSE (RMSE), and mean absolute percentage error (MAPE) over the whole iterative training and test process.

3) Results: We compare CoGNNs to various algorithms on four datasets. We also test the degenerated variants of CoGNNs without collaborative graphs, that is, CoGNNs (no $\mathbf{W}_t$). We evaluate on the mean MAE, mean RMSE, and mean MAPE as shown in Table IV. We see that: 1) compared to the previous works, our CoGNNs obtain the lowest errors on all the datasets and 2) compared to the model variants without collaborative graphs, the complete CoGNNs also achieve the much better prediction performances. The detailed traffic forecasts at different prediction times are presented in the Appendix.

D. Model Analysis

1) Regrets of Various Forms of CoGNNs: According to our theoretical regret analysis, the static regrets converge with $\mathrm{R}_T = \mathcal{O}(1/(T)^{1/2})$. Here, we compute and visualize the static regrets of different forms of CoGNNs to verify our analysis. For online motion prediction on H3.6M, we illustrate the regret curves along time within 4500 iterations [see Fig. 6(a)]. The regrets of three forms of CoGNNs could converge to very low values within approximately 2500 iterations. Additionally, the regret curves reflect that CoGNNs quickly close the performance gaps with the models that only consider the best collaborative pairs, showing the effectiveness and interpretation of our CoPUs.

2) Effect of Learning Rate η: The learning rate affects the training of the collaborative predictor and the collaborative graph in each CoPU. Here, we vary the learning rate η

TABLE IV

COMPARISONS BETWEEN OUR COGNNS AND STATE-OF-THE-ART METHODS FOR ONLINE TRAFFIC SPEED FORECASTING ON FOUR DATASETS BASED ON THREE METRICS. WE ALSO PRESENT THE DEGENERATED VARIANTS OF COGNNS WHICH ABANDON THE COLLABORATIVE GRAPHS, THAT IS, "NO $\mathbf{W}_t$"

	Datasets	METR-LA			PeMS-BAY			PeMS-D4			PeMS-D8		
	Methods	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAP
Baselines	ST-GCN [14]	5.68	8.92	13.0%	3.09	6.57	7.7%	37.28	46.44	30.5%	30.49	39.57	24.8%
	DCRNN [18]	5.45	8.76	12.6%	3.07	6.69	7.8%	37.67	46.95	31.1%	31.31	39.35	25.4%
	ASTGCN [13]	4.66	7.43	11.7%	2.51	4.29	6.0%	32.15	43.72	28.8%	28.85	36.97	23.0%
	G-WaveNet [31]	4.42	7.10	11.2%	2.31	4.07	5.3%	28.58	39.59	25.6%	27.81	36.03	19.8%
	STSGCN [4]	4.73	7.62	11.6%	2.44	4.31	6.8%	23.82	30.85	18.9%	24.47	33.11	16.4%
	AGCRN [66]	4.19	6.40	8.3%	2.16	3.74	4.7%	19.65	28.54	16.0%	22.72	29.03	13.4%
no $\mathbf{W}_t$	CoGNN-AR (no $\mathbf{W}_t$)	4.42	6.61	8.8%	1.72	2.65	4.1%	19.37	28.02	17.3%	16.92	26.61	12.7%
	CoGNN-LSTM (no $\mathbf{W}_t$)	6.08	8.35	14.6%	2.74	3.68	4.4%	27.45	35.14	19.3%	20.12	29.73	13.7%
	CoGNN-TC (no $\mathbf{W}_t$)	3.61	5.89	8.0%	1.43	2.41	3.0%	20.27	29.48	16.1%	16.99	24.14	11.9%
Ours	CoGNN-AR	4.09	6.22	8.3%	1.59	2.42	3.8%	19.07	**26.94**	15.3%	15.75	23.74	11.4%
	CoGNN-LSTM	5.04	7.28	13.4%	2.21	3.52	4.2%	26.95	32.31	18.9%	19.62	28.87	13.1%
	CoGNN-TC	**3.46**	**5.75**	**7.8%**	**1.36**	**2.24**	**2.8%**	**18.93**	27.71	**14.7%**	**15.51**	**22.35**	**10.8%**

TABLE V

AVERAGE MPJPE OF COGNNS WITH DIFFERENT NUMBERS OF COPUS FOR ONLINE HUMAN MOTION PREDICTION

Methods	1 CoPU	2 CoPUs	3 CoPUs	4 CoPUs
CoGNN-AR	38.16	**34.42**	34.46	35.15
CoGNN-LSTM	28.48	**26.73**	27.47	26.92
CoGNN-TC	26.07	**24.39**	24.74	25.03

from 0.0075 to 0.125 to test the model performance for online motion prediction on H3.6M. The mean MPJPEs of different forms of CoGNNs are illustrated in Fig. 6(b). We see that, CoGNN-AR, CoGNN-LSTM, and CoGNN-TC achieve their best performance when η is 0.075, 0.05, and 0.01, respectively. CoGNN-AR tends to be more stable with various η than others; CoGNN-LSTM and CoGNN-TC obtain better performance given the reasonable η.

3) Effect of Number of CoPUs in CoGNNs: Here, we analyze how different numbers of CoPUs in the CoGNN architectures affect the performance. We test different forms of CoGNNs with 1–4 CoPUs for online human motion prediction on H3.6M. The mean MPJPEs are presented in Table V. For three forms of CoGNNs, two CoPUs help to achieve the lowest MPJPEs, which outperform the models with only one CoPU to different extents, showing the effectiveness of the progressively refinement. For 3 or 4 CoPUs, the model performance shows sub-par results due to the slight over-smoothing, but it still converges to stable values.

4) Visualization of the Learned Collaborative Graphs: To show that our model effectively captures the implicit systematic dependencies by learning collaborative graphs, here we visualize the learned collaborative graphs on the various tasks of online simulated trajectory prediction, human motion prediction and traffic speed prediction, respectively.

We first visualize the collaborative graphs for online simulated trajectory prediction. We compare the learned collaborative graphs and the predefined interactions that are used to simulate the particle interaction systems. We illustrate the adjacency matrices of both simulated graphs and learned graphs. To reflect that our model could adapt to the changing states of the interaction systems, we show the learned graphs at different iteration steps during the online learning process.

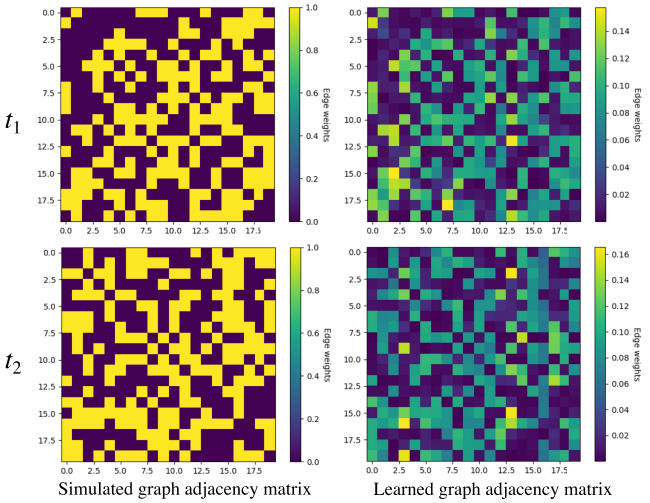


Fig. 7. Visualization of the adjacency matrices of the collaborative graph on the task of online simulated trajectory prediction. The two rows denote the different interactions at different time, while the first and the second columns denote the adjacency matrices of the simulated graphs and the learned graphs, respectively.

The adjacency matrices are illustrated in Fig. 7. We see that, at one iteration step, t_1 or t_2, the learned collaborative graph captures similar topologies to the simulated graphs with weighted edges to different extents. At different iteration steps, the collaborative graphs could adapt to the changing simulated interaction structures. Note that the element values of the graph adjacency matrices are not similar due to the normalization in our weight update algorithm, while we focus more on the visual similarity of topologies. In this way, we verify that the collaborative graph could reasonably depict the implicit interactions in the complex systems.

Here, we visualize the learned collaborative graphs on the H3.6M dataset. We use red segments to plot the graph edges whose weights are larger than 0.2 on human poses. Since any edge weights range from 0 to 1, and most of them are lower than 0.2, thus the plotted edges show the important collaborative pairs, that is, the important influence between two agents. The graphs are illustrated in Fig. 8. In Fig. 8(a), different actions have different graphs, indicating distinct relations on specific actions; for example, arms and legs affect

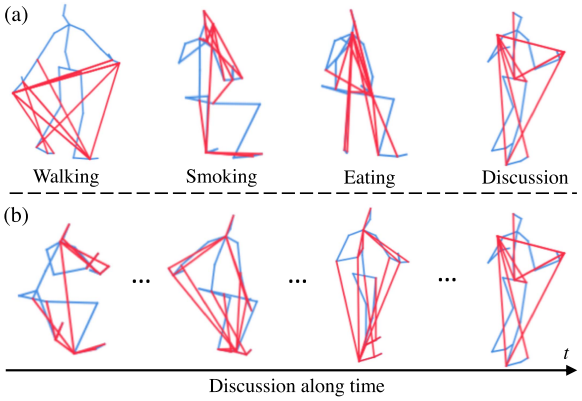


Fig. 8. Visualization of the collaborative graphs on the task of online human motion prediction. The edges whose weights are larger than 0.2 are plotted in red. (a) Learned graphs for different actions, showing distinct interactions. (b) Graphs evolves along time, showing dynamic interactions.

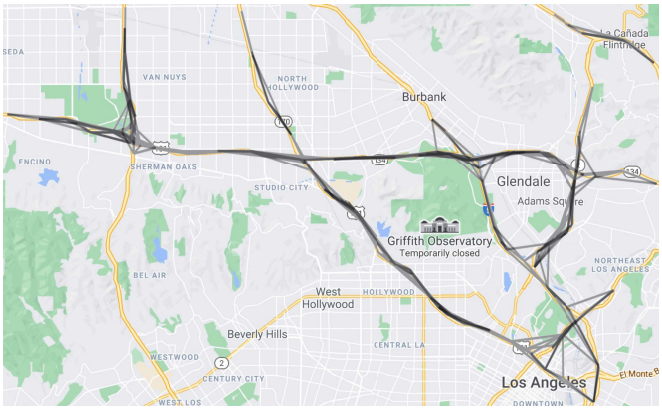


Fig. 9. Learned collaborative graph for online traffic speed prediction on METR-LA dataset. The edges whose collaborative weights are larger than 0.1 are illustrated, reflecting the most important collaborative effects among the 207 nodes.

across left and right for walking, while hands are highly related to other joints for eating and smoking. In Fig. 8(b), we show the dynamic graphs changing along time during discussion.

We also show the learned collaborative graph for online traffic speed prediction. As an example, we show the learned graph on METR-LA dataset. We illustrate the edges whose collaborative weights are larger than 0.1 in Fig. 9, which reflects the most important collaborative effects among the 207 nodes to some extent. We see that, the traffic speed data collected by different sensors are mainly affected by their geographically nearby traffic. For example, a certain node on a single long highway is mainly affected by other nodes on the same highway, because a single long highway could be regarded as an isolated systems, in which the vehicles move and stop collaboratively. Moreover, near the intersections or the traffic circles, one node tends to be affected by the traffic around across roads, since many vehicles perform agglomeration effects in these local regions. Fig. 9 shows a reasonable structure to depict the collaborative effects. Different from the human motion data, in a traffic scenario, the interactions and relations between two nodes with a relatively long distance is usually weak. That is because traffic data has very strong spatial location attributes, and higher synergy tends to appear in more local areas, such as local congestion during peak hours; and two nodes farther apart are difficult to affect each other because of the decay of collaborative effects along distances.

VII. Conclusion

We develop a novel method to predict the future statuses of a multi-agent system online. We propose a novel CoPU, which uses a collaborative graph to aggregate multiple collaborative predictors that learn dynamics from collaborative pairs. The collaborative graph depicts the influence level of collaborative pairs, which are adjusted with the guidance from an explicit objective. The regret analysis show that our method achieves similar performance with the best single collaborative predictor. Multiple CoPUs are stacked in cascade as a CoGNN. Experiments demonstrate the effectiveness of our method for various multi-agent forecasting tasks.

References

[1] A. Jahangiri and H. A. Rakha, "Applying machine learning techniques to transportation mode recognition using mobile phone sensor data," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 5, pp. 2406–2417, Oct. 2015.

[2] J. Wojtusiak, T. Warden, and O. Herzog, "Machine learning in agent-based stochastic simulation: Inferential theory and evaluation in transportation logistics," *Comput. Math. With Appl.*, vol. 64, no. 12, pp. 3658–3665, Dec. 2012.

[3] W. Yu, D. An, D. Griffith, Q. Yang, and G. Xu, "Towards statistical modeling and machine learning based energy usage forecasting in smart grid," *ACM SIGAPP Appl. Comput. Rev.*, vol. 15, no. 1, pp. 6–16, Mar. 2015.

[4] C. Song, Y. Lin, S. Guo, and H. Wan, "Spatial-temporal synchronous graph convolutional networks: A new framework for spatial-temporal network data forecasting," in *Proc. AAAI Conf. Artif. Intell. (AAAI)*, 2020, pp. 914–921.

[5] T. Kipf, E. Fetaya, K.-C. Wang, M. Welling, and R. Zemel, "Neural relational inference for interacting systems," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2018, pp. 2688–2697.

[6] C. Graber and A. G. Schwing, "Dynamic neural relational inference," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 8513–8522.

[7] J. Li, F. Yang, M. Tomizuka, and C. Choi, "EvolveGraph: Multi-agent trajectory prediction with dynamic relational reasoning," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2020, pp. 19783–19794.

[8] S. C. H. Hoi, D. Sahoo, J. Lu, and P. Zhao, "Online learning: A comprehensive survey," 2018, *arXiv:1802.02871*.

[9] E. Zivor and J. Wang, "Vector autoregressive models for multivariate time series," in *Modeling Financial Time Series With S-PLUS*. Springer, 2006, pp. 385–429.

[10] A. M. Lehrmann, P. V. Gehler, and S. Nowozin, "Efficient nonlinear Markov models for human motion," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1314–1321.

[11] K. P. Vishwakarma, "Prediction of economic time-series by means of the Kalman filter," *Int. J. Syst. Sci.*, vol. 1, no. 1, pp. 25–32, Oct. 1970.

[12] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[13] S. Guo, Y. Lin, N. Feng, C. Song, and H. Wan, "Attention based spatial-temporal graph convolutional networks for traffic flow forecasting," in *Proc. AAAI Conf. Artif. Intell. (AAAI)*, 2019, pp. 922–929.

[14] B. Yu, H. Yin, and Z. Zhu, "Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Jul. 2018, pp. 3634–3640.

[15] W. Mao, M. Liu, M. Salzmann, and H. Li, "Learning trajectory dependencies for human motion prediction," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9489–9497.

[16] Q. Cui, H. Sun, and F. Yang, "Learning dynamic relationships for 3D human motion prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6519–6527.

[17] M. Li, S. Chen, Y. Zhao, Y. Zhang, Y. Wang, and Q. Tian, "Dynamic multiscale graph neural networks for 3D skeleton based human motion prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 214–223.

[18] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–16.

[19] M. Li, S. Chen, X. Chen, Y. Zhang, Y. Wang, and Q. Tian, "Actional-structural graph convolutional networks for skeleton-based action recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3595–3603.

[20] Y. Hu, S. Chen, Y. Zhang, and X. Gu, "Collaborative motion prediction via neural motion message passing," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6319–6328.

[21] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Two-stream adaptive graph convolutional networks for skeleton-based action recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 12026–12035.

[22] B. Williams and L. Hoel, "Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results," *J. Transp. Eng.*, vol. 129, no. 6, pp. 664–672, Nov./Dec. 2003.

[23] J. Wang, A. Hertzmann, and D. Fleet, "Gaussian process dynamical models," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2006, pp. 1441–1448.

[24] G. W. Taylor and G. E. Hinton, "Factored conditional restricted Boltzmann machines for modeling motion style," in *Proc. 26th Annu. Int. Conf. Mach. Learn. (ICML)*, 2009, pp. 1025–1032.

[25] S. S. Rangapuram, M. W. Seeger, J. Gasthaus, L. Stella, Y. Wang, and T. Januschowski, "Deep state space models for time series forecasting," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2018, pp. 7785–7794.

[26] J. Martinez, M. J. Black, and J. Romero, "On human motion prediction using recurrent neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 4674–4683.

[27] L.-Y. Gui, Y.-X. Wang, X. Liang, and J. M. F. Moura, "Adversarial geometry-aware human motion prediction," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 786–803.

[28] G. Lai, W.-C. Chang, Y. Yang, and H. Liu, "Modeling long- and short-term temporal patterns with deep neural networks," in *Proc. 41st Int. ACM SIGIR Conf. Res. Develop. Inf. Retr. (SIGIR)*, Jun. 2018, pp. 95–104.

[29] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," 2018, *arXiv:1803.01271*.

[30] A. Jain, A. R. Zamir, S. Savarese, and A. Saxena, "Structural-RNN: Deep learning on spatio-temporal graphs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 5308–5317.

[31] Z. Wu, S. Pan, G. Long, J. Jiang, and C. Zhang, "Graph WaveNet for deep spatial-temporal graph modeling," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 1907–1913.

[32] M. Li, S. Chen, Y. Zhang, and I. Tsang, "Graph cross networks with vertex infomax pooling," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2020, pp. 14093–14105.

[33] P. D. Dobson and A. J. Doig, "Distinguishing enzyme structures from non-enzymes without alignments," *J. Mol. Biol.*, vol. 330, no. 4, pp. 771–783, Jul. 2003.

[34] T. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017, pp. 1–14.

[35] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–12.

[36] S. Liu, M. Sun, L. Feng, H. Qiao, S. Chen, and Y. Liu, "Social neighborhood graph and multigraph fusion ranking for multifeature image retrieval," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 3, pp. 1389–1399, Mar. 2021.

[37] L. C. B. Torres, C. L. Castro, F. Coelho, and A. P. Braga, "Large margin Gaussian mixture classifier with a Gabriel graph geometric representation of data set structure," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 3, pp. 1400–1406, Mar. 2020.

[38] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Jan. 2008.

[39] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2014, pp. 1–14.

[40] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 3844–3852.

[41] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 1024–1034.

[42] M. Niepert, M. Ahmed, and K. Kutzkovl, "Learning convolutional neural networks for graphs," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2016, pp. 2014–2023.

[43] Y. Li, D. Tarlow, M. Brockschmidt, and R. Zemel, "Gated graph sequence neural networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2016, pp. 1–20.

[44] H. Dai, B. Dai, and L. Song, "Discriminative embeddings of latent variable models for structured data," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2016, pp. 2702–2711.

[45] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019, pp. 1–17.

[46] X. Xu, T. Wang, Y. Yang, A. Hanjalic, and H. T. Shen, "Radial graph convolutional network for visual question generation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 4, pp. 1654–1667, Apr. 2021.

[47] S. Thrun, "Lifelong learning algorithms," in *Learning to Learn*. Springer, 1998, pp. 181–209.

[48] J. Lu, D. Sahoo, P. Zhao, and S. C. H. Hoi, "Sparse passive-aggressive learning for bounded online kernel methods," *ACM Trans. Intell. Syst.*, vol. 9, no. 4, pp. 216–242, 2018.

[49] J. Kivinen, A. J. Smola, and R. C. Williamson, "Online learning with kernels," *IEEE Trans. Signal Process.*, vol. 52, no. 8, pp. 2165–2176, Aug. 2004.

[50] O. Dekel, S. Shalev-Shwartz, and Y. Singer, "The Forgetron: A kernel-based Perceptron on a budget," *SIAM J. Comput.*, vol. 37, no. 5, pp. 1342–1372, 2008.

[51] Z. Wang, K. Crammer, and S. Vucetic, "Breaking the curse of kernelization: Budgeted stochastic gradient descent for large-scale SVM training," *J. Mach. Learn. Res.*, vol. 13, pp. 3103–3131, Oct. 2012.

[52] J. Zhang, H. Ning, X. Jing, and T. Tian, "Online kernel learning with adaptive bandwidth by optimal control approach," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 5, pp. 1920–1934, May 2021.

[53] J. Lu, S. C. Hoi, J. Wang, P. Zhao, and Z.-Y. Liu, "Large scale online kernel learning," *J. Mach. Learn. Res.*, vol. 17, no. 47, pp. 1–43, 2016.

[54] P. Bouboulis, S. Chouvardas, and S. Theodoridis, "Online distributed learning over networks in RKH spaces using random Fourier features," *IEEE Trans. Signal Process.*, vol. 66, no. 7, pp. 1920–1932, Apr. 2018.

[55] Y. Ding, C. Liu, P. Zhao, and S. C. H. Hoi, "Large scale kernel methods for online AUC maximization," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2017, pp. 91–100.

[56] Y. Shen, T. Chen, and G. B. Giannakis, "Random feature-based online multi-kernel learning in environments with unknown dynamics," *J. Mach. Learn. Res.*, vol. 20, no. 22, pp. 1–36, 2019.

[57] Y. Xiu, J. Li, H. Wang, Y. Fang, and C. Lu, "Pose flow: Efficient online pose tracking," in *Proc. Brit. Mach. Vis. Conf. (BMVC)*, 2018, pp. 1–12.

[58] J. Butepage, H. Kjellstrom, and D. Kragic, "Anticipating many futures: Online human motion prediction and generation for human-robot interaction," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 4563–4570.

[59] F. Marchetti, F. Becattini, L. Seidenari, and A. Del Bimbo, "MANTRA: Memory augmented networks for multiple trajectory prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 7143–7152.

[60] S. Hu, F. Zhu, X. Chang, and X. Liang, "UPDeT: Universal multi-agent RL via policy decoupling with transformers," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2021, pp. 1–15.

[61] Y. Cai *et al.*, "Learning progressive joint propagation for human motion prediction," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2020, pp. 226–242.

[62] E. Hazan, "Introduction to online convex optimization," *Found. Trends Mach. Learn.*, vol. 2, nos. 3–4, pp. 157–325, 2016.

[63] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu, "Human3.6M: Large scale datasets and predictive methods for 3D human sensing in natural environments," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 7, pp. 1325–1339, Jul. 2014.

[64] C. Li, Z. Zhang, W. S. Lee, and G. H. Lee, "Convolutional sequence to sequence model for human dynamics," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 5226–5234.

[65] W. Mao, M. Liu, and M. Salzmann, "History repeats itself: Human motion prediction via motion attention," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2020, pp. 474–489.

[66] L. Bai, L. Yao, C. Li, X. Wang, and C. Wang, "Adaptive graph convolutional recurrent network for traffic forecasting," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, 2020, pp. 17804–17815.

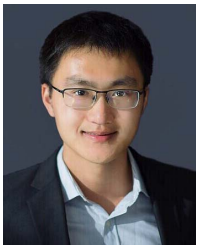
[67] H. V. Jagadish *et al.*, "Big data and its technical challenges," *Commun. ACM*, vol. 57, no. 7, pp. 86–94, Jul. 2014.

[68] C. Chen, K. Petty, A. Skabardonis, P. Varaiya, and Z. Jia, "Freeway performance measurement system: Mining loop detector data," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 1748, no. 1, pp. 96–102, Jan. 2001.

Maosen Li (Student Member, IEEE) received the B.E. degree in optical engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2017. He is currently pursuing the Ph.D. degree with the Cooperative Medianet Innovation Center, Shanghai Jiao Tong University, Shanghai, China.

His research interests include computer vision, machine learning, graph representation learning, and video analysis.

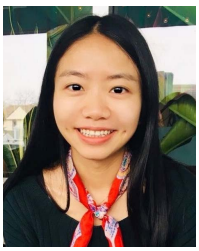
Mr. Li is a Reviewer of some prestigious international journals and conferences, including IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE (TPAMI), IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS (TNNLS), *International Journal of Computer Vision* (IJCV), IEEE TRANSACTIONS ON MULTIMEDIA (TMM), *Pattern Recognition* (PR), International Conference on Machine Learning (ICML), Conference on Neural Information Processing Systems (NeurIPS), and AAAI Conference on Artificial Intelligence.



Siheng Chen (Member, IEEE) received the bachelor's degree in electronics engineering from the Beijing Institute of Technology, Beijing, China, in 2011, and the master's degree in electrical and computer engineering and machine learning and the Ph.D. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, USA, in 2016.

He was a Post-Doctoral Research Associate with Carnegie Mellon University. He was an Autonomy Engineer with the Uber Advanced Technologies Group, Pittsburgh, working on the perception and prediction systems of self-driving cars. He was a Research Scientist with Mitsubishi Electric Research Laboratories (MERL), Cambridge, MA, USA. He is currently an Associate Professor with Shanghai Jiao Tong University, Shanghai, China. His research interests include graph signal processing, graph neural networks, and autonomous driving.

Dr. Chen was a recipient of the 2018 IEEE Signal Processing Society Young Author Best Paper Award. His coauthored article received the Best Student Paper Award at IEEE GlobalSIP 2018. He organized the special session "Bridging Graph Signal Processing and Graph Neural Networks" at ICASSP 2020.



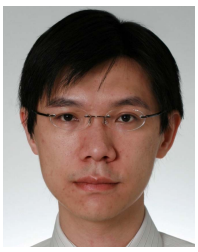
Yanning Shen (Member, IEEE) received the Ph.D. degree from the University of Minnesota (UMN), Minneapolis, MN, USA, in 2019.

Her research interests include machine learning, network science, data science, and statistical-signal processing.

Dr. Shen was a finalist for the Best Student Paper Award at the 2017 IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing and the 2017 Asilomar Conference on Signals, Systems, and Computers. She was selected as a Rising Star in EECS by Stanford University in 2017 and received the UMN Doctoral Dissertation Fellowship in 2018.

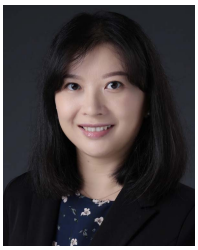
Genjia Liu received the bachelor's degree from Shanghai Jiao Tong University, Shanghai, China, in 2021, where he is currently pursuing the Ph.D. degree with the Cooperative Medianet Innovation Center.

His research interests include graph signal processing and graph representation learning.



Ivor W. Tsang (Fellow, IEEE) was a Professor of artificial intelligence with the University of Technology Sydney (UTS), Sydney, NSW, Australia, and the Research Director of the Australian Artificial Intelligence Institute (AAII), UTS. He is currently the Director of the A*STAR Centre for Frontier AI Research (CFAR), Singapore. He is working at the forefront of big data analytics and artificial intelligence. His research focuses on transfer learning, deep generative models, and learning with weakly supervision. His work is recognized internationally for its outstanding contributions to those fields.

Dr. Tsang received an ARC Future Fellowship for his outstanding research on big data analytics and large-scale machine learning in 2013. In 2019, his *Journal of Machine Learning Research* article titled "Towards ultrahigh dimensional feature selection for big data" received the International Consortium of Chinese Mathematicians Best Paper Award. In 2020, he was recognized as the AI 2000 AAAI Conference on Artificial Intelligence/IJCAI Most Influential Scholar in Australia for his outstanding contributions to the field between 2009 and 2019. His research on transfer learning was awarded the Best Student Paper Award at CVPR 2010 and the 2014 IEEE TMM Prize Paper Award. In addition, he received the IEEE TNN Outstanding 2004 Paper Award in 2007 for his innovative work on solving the inverse problem of nonlinear representations. He was conferred the IEEE Fellow for his outstanding contributions to large-scale machine learning and transfer learning. He serves as the Editorial Board Member for the *Journal of Machine Learning Research*, *Machine Learning*, *Journal of Artificial Intelligence Research*, IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, IEEE TRANSACTIONS ON ARTIFICIAL INTELLIGENCE, IEEE TRANSACTIONS ON BIG DATA, and IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTATIONAL INTELLIGENCE. He serves as a Senior Area Chair/Area Chair for Conference on Neural Information Processing Systems (NeurIPS), International Conference on Machine Learning (ICML), AAAI, and International Joint Conference on Artificial Intelligence (IJCAI), and the Steering Committee of Asian Conference on Machine Learning (ACML).



Ya Zhang (Member, IEEE) received the bachelor's degree from Tsinghua University, Beijing, China, in 2000, and the Ph.D. degree in information sciences and technology from Pennsylvania State University, State College, PA, USA, in 2005.

She was an Assistant Professor at The University of Kansas, Lawrence, KS, USA, with a research focus on machine learning applications in bioinformatics and information retrieval. She was a Research Manager at Yahoo! Labs, Sunnyvale, CA, USA, where she led a research and development team of researchers with strong backgrounds in data mining and machine learning to improve the web search quality of Yahoo international markets. She is currently a Professor at the Cooperative Medianet Innovation Center, Shanghai Jiao Tong University, Shanghai, China. She has published more than 70 refereed articles in prestigious international conferences and journals, including IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE (TPAMI), IEEE TRANSACTIONS ON IMAGE PROCESSING (TIP), IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS (TNNLS), IEEE International Conference on Data Mining (ICDM), IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE International Conference on Computer Vision (ICCV), European Conference on Computer Vision (ECCV), and European Conference on Machine Learning (ECML). She currently holds five U.S. patents and four Chinese patents and has nine pending patents in the areas of multimedia analysis. She was appointed as the Chief Expert for the "Research of Key Technologies and Demonstration for Digital Media Self-Organizing" project under the 863 Program by the Ministry of Science and Technology of China. Her research interest is mainly in machine learning with applications to multimedia and healthcare.