This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Robust Traffic Prediction From Spatial–Temporal Data Based on Conditional Distribution Learning

Zeng Zeng, *Senior Member, IEEE*, Wei Zhao, Peisheng Qian, Yingjie Zhou, *Member, IEEE*, Ziyuan Zhao, *Member, IEEE*, Cen Chen, and Cuntai Guan, *Fellow, IEEE*

***Abstract*—Traffic prediction based on massive speed data collected from traffic sensors plays an important role in traffic management. However, it is still challenging to obtain satisfactory performance due to the complex and dynamic spatial-temporal correlations among the data. Recently, many research works have demonstrated the effectiveness of graph neural networks (GNNs) for spatial–temporal modeling. However, such models are restricted by conditional distribution during training, and may not perform well when the target is outside the primary region of interest in the distribution. In this article, we address this problem with a stagewise learning mechanism, in which we redefine speed prediction as a conditional distribution learning followed by speed regression. We first perform a conditional distribution learning for each observed speed class, and then obtain speed prediction by optimizing regression learning, based on the learned conditional distribution. To effectively learn the conditional distribution, we introduce a mean–residue loss, consisting of two parts: 1) a mean loss, which penalizes the differences between the mean of the estimated conditional distribution and the ground truth and 2) a residue loss, which penalizes residue errors of the long tails in the distribution. To optimize the subsequent regression based on distribution information, we combine the mean absolute error (MAE) as another part of the loss function. We also incorporate a GNN-based architecture with our proposed learning mechanism. Mean–residue loss is employed to supervise the hidden speed representation in the network at each time interval, followed by a shared layer to recalibrate the hidden temporal dependencies in the conditional distribution. The experimental results based on three public traffic datasets have demonstrated that the effectiveness of the proposed method outperforms state-of-the-art methods.**

***Index Terms*—Deep neural network, graph neural networks (GNNs), mean–residue loss, spatial–temporal data, traffic prediction.**

I. Introduction

TRAFFIC prediction is a fundamental task for various urban management tasks. For example, in the growth of smart cities, outcomes of traffic prediction motivate various urban studies to enhance the efficiency and safety of the transportation system [1]. At the same time, speed forecast is a very challenging task. Due to the highly dynamic nature of traffic, there are complicated and diverse spatial–temporal relations among the traffic data. To discover such relations for accurate traffic modeling, efforts have been made to capture and model the dynamic characteristics and interdependencies in traffic data [2]–[4].

Recent research explores deep learning methods [5]–[8], especially spatial–temporal graph networks, to model the complex and dynamic spatial–temporal correlations [9]–[11]. They have shown promising results in traffic forecasting. Spatial–temporal graph modeling assumes that a node's future information is dependent on its historical information as well as its neighbors' historical information. This approach is also widely applied in taxi demand prediction [12], human action recognition [13], etc. In traffic prediction, there are two main categories of spatial–temporal graph modeling methods, that is: 1) recurrent neural networks (RNN)-based methods [14], [15] and 2) convolution neural networks (CNNs)-based methods [2], [13]. Both of them integrate graph convolution networks (GCNs) into their base architectures. These methods have effectively captured spatial and temporal traffic relations in their networks, and have incremental progress on public traffic datasets. While advances have been achieved with these methods, there are major limitations. Specifically, these studies simply use mean absolute error (MAE) alone as the loss function to solve the traffic prediction problem, which may lead to overfitting on a particular set of speeds [16]. Therefore, regression-based methods are limited by the narrow conditional distribution during training [9]. The performance can drop significantly, especially for extreme values and outliers, for example, the speed at the peak curvature points.

We speculate that the aforementioned problem can be addressed by a framework including two stages, which forms a progressive solution for traffic prediction. First, the framework learns the probability distribution among the speed classes.

Manuscript received November 2, 2020; revised September 03, 2021 and October 29, 2021; accepted November 24, 2021. This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 61801315; in part by the Singapore–China NRF-NSFC under Grant NRF2016NRF-NSFC001-111; and in part by 111 Project under Grant B21044. This article was recommended by Associate Editor N. Zhang. *(Corresponding author: Yingjie Zhou.)*

Zeng Zeng, Peisheng Qian, and Cen Chen are with the Institute for Infocomm Research, Agency for Science, Technology and Research (A*STAR), Singapore (e-mail: zengz@i2r.a-star.edu.sg; qianp@i2r.a-star.edu.sg; chenc@i2r.a-star.edu.sg).

Wei Zhao and Yingjie Zhou are with the College of Computer Science, Sichuan University, Chengdu 610065, China (e-mail: 2019223049289@stu.scu.edu.cn; yjzhou09@gmail.com).

Ziyuan Zhao is with the Institute for Infocomm Research, Agency for Science, Technology and Research (A*STAR), Singapore, and also with the School of Computer Science and Engineering, Nanyang Technological University, Singapore (e-mail: zhaoz@i2r.a-star.edu.sg).

Cuntai Guan is with the School of Computer Science and Engineering, Nanyang Technological University, Singapore (e-mail: ctguan@ntu.edu.sg).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCYB.2021.3131285.

Digital Object Identifier 10.1109/TCYB.2021.3131285

2168-2267 © 2021 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Nanyang Technological University. Downloaded on December 18,2021 at 14:16:30 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

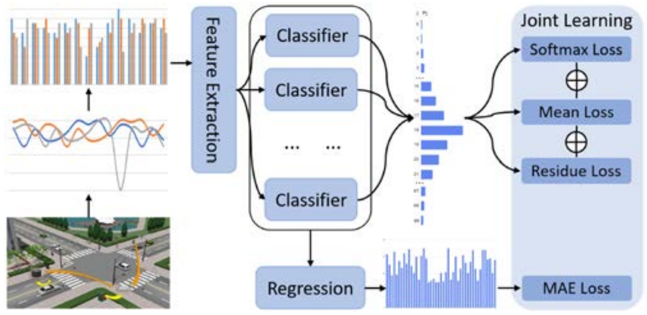


Fig. 1. Illustration of our proposed 2-stage traffic prediction. Traffic signals are converted as discrete values. The multiple classification branches represent conditional distributions at different time horizons, followed by the final fine-grained speed prediction. We propose a mean–residue loss to be used together with softmax and MAE loss as the training target, which regularizes the conditional distribution in hidden representations.

For the classification task, the objective is to learn the features of different speed classes and represent the probability distribution among them. Different from other label distribution learning (LDL) tasks, the target distribution learning in our article is defined as a conditional distribution learning, which is determined by time, location, and the history of each speed value. Each conditional distribution is corresponding to a certain speed class with a narrow speed range between two consecutive integers. Second, a regression process can achieve fine-grained speed prediction based on the hidden representation tuned by the classification task. The framework is inspired by analog-digital conversion [17], in which the signal is first discretized and digitized, and then recalibrated to the continuous value. In the first stage, the classification procedure can supervise the backbone neural networks to learn more features belonging to different speed classes and represent the probability distribution among the classes. Based on the extra information provided by the classification procedure, in the second stage, the regression network can discover more from the spatial–temporal data, leading to improved prediction. The approach is conceptually similar to other mutually supplementing classification and regression frameworks, for example, double-task CNN (DTCNN) for image ordinal estimation [18], but is sequential rather than parallel.

As discussed above, the hidden representation of conditional distribution among different speed classes is critical to the model performance. In this work, we propose a novel loss function, called mean–residue loss, that can supervise the neural network to learn the features of different speed classes efficiently and represent the conditional distribution among the speed classes properly, to enhance the traffic prediction. As shown in Fig. 1, the loss function consists of a mean loss, which penalizes the differences between the mean of the estimated conditional distribution and the ground truth, and a residue loss, which penalizes the residue error of the long tails in the distribution. As a result, the mean–residue loss can concentrate the speed representation at the target value, and shift the distribution to the target when they do not match. Furthermore, mean loss reinforces MAE loss at the regression stage. The probability distribution of speed, optimized by mean–residue loss, is crucial for speed forecast at a finer level, and is the input of our regression stage, as described above. The mean–residue loss can effectively capture the probability distribution for the speed prediction.

To demonstrate the effectiveness of the proposed mean–residue loss, we incorporate the loss into a neural network architecture based on Graph-WaveNet [9], for speed prediction. The network consists of multiple branches for speed classification, speed predictions for different time intervals, and shared regression layers for final predictions. Moreover, classification and regression losses are calculated and backpropagated at respective layers and branches in the training process. Appropriate scaling factors and training configurations have been applied to ensure model convergence [2], [9], [10], [15], [19]. The experimental results on three public traffic datasets have demonstrated the superiority of the proposed method over state-of-the-art approaches in the literature.

The main contributions of the work can be summarized as follows.

1) We propose a stagewise learning mechanism, including: a) a conditional distribution learning and b) a regression learning. We first perform a conditional distribution learning for each observed speed class, and then obtain speed prediction by optimizing regression learning, based on the learned speed distribution. The approach redefines speed prediction as conditional distribution based speed regression. It effectively addresses target speed that is outside the primary region of interest in the distribution.
2) To fulfill this learning mechanism, we propose a novel mean–residue loss function for distribution learning and combine it with MAE for speed regression. The mean–residue loss consists of: a) a mean loss, which penalizes the differences between the mean of the estimated conditional distribution and the ground truth and b) a residue loss, which penalizes the residue errors of the long tails in the distribution. We also present a graph neural network (GNN)-based model that could incorporate with our proposed loss for speed prediction.
3) We evaluate the method on three widely used traffic datasets, that is: 1) METR-LA [3], [15]; 2) PMES-BAY [15]; and 3) PeMSD7 [20]. Both theoretical analysis and experimental results have demonstrated the superiority of the proposed method over state-of-the-art approaches.

The remainder of this work is organized as follows. In Section II, we present an overview of the related work in GNNs for traffic prediction and label distribution learning. Our proposed method is elaborated in Section III. Sections IV and V demonstrate the dataset and experimental settings, and evaluates the proposed method with respect to different metrics. Finally, we conclude our work and present some future work in Section VI.

II. Related Work

Our work is based on speed data from traffic sensors. The proposed method benefits both from recent work on GCNs and from the field of distribution learning. In the following sections, we provide a brief overview of related work in traffic IoT technologies and algorithms.

Authorized licensed use limited to: Nanyang Technological University. Downloaded on December 18,2021 at 14:16:30 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

A. Traffic Prediction from Spatial–Temporal Data

Traffic prediction has attracted research attention in recent years. Zhang *et al.* [21] investigated the forecasting of electric vehicle load based on traffic flow prediction. Liu *et al.* [22] employed the attention-based network for large-scale transportation demand prediction. Zhang *et al.* [23] demonstrated an exclusive cyber-physical system (CPS) for velocity profile prediction. Despite extensive research efforts, how to discover the spatial–temporal correlations among time-series data collected from sensors remains a big challenge in the domain of traffic prediction. Recently, GCNs-based approaches have achieved state-of-the-art performance, which can work as the main building blocks in constructing complex graph representations of the data [2], [11]. In the literature, there are numerous applications of GCNs for traffic prediction, including node embedding [24], node classification [25], node clustering [26], graph classification [27], and link prediction [28]. In GCNs, multiple graph convolutional layers are stacked to obtain node representations from their own features and their neighbors' features.

A large number of approaches based on deep learning using graph representations have been proposed in recent years, most of which fall into two broad categories, that is: 1) spectral-based methods and 2) spatial-based methods. In spectral-based approaches, noises in the signals of the node's input are smoothed via graph convolution operations [14]. While motivated by conventional convolutions on images, graph convolutions based on a node's spatial dependencies are defined in spatial-based approaches [29]. In these approaches, the adjacency matrix is generated based on the original data, which stores nodes' representation by aggregating information from neighborhoods. In general, the adjacency matrix is extracted as prior knowledge with Gaussian kernels [30], attention mechanisms [31], or an adaptive path layer [32], in which the adjacency matrix is fixed throughout training. In contrast, Liu *et al.* proposed to explore the relationships of nodes' neighborhoods in an adaptive manner for improved stability [9], in which the proposed adaptive path layer learns the importance of different sized neighborhoods and signals aggregated from neighbors of different hops away for breadth and depth exploration, respectively. Although these methods can learn the adjacency matrix to a certain extent, they are still based on the predefined graph-structure data, which is unstable for spatial–temporal graph modeling.

Spatial–temporal graph modeling is typically carried out using spatial–temporal GNNs (STGNNs). With no assumptions on static graph structures or graph inputs, [11] used dynamic interdependent node inputs in STGNNs based on RNNs or CNNs. Seo *et al.* [14] used a recurrent unit in graph convolution to filter node inputs and hidden states. To enhance model performance, diffusion convolution [15] and attention mechanisms [28] are combined with GNNs. Jain *et al.* [33] combined node-level RNNs and edge-level RNNs for better sequence learning in a structured graph. While RNN-based methods suffer from potential gradient explosion in long sequences with GCN, CNN-based methods combine graph convolution with 1D convolution for more computational efficiency [2], [13]. Despite this, they need to stack many layers to capture long sequences, resulting in the linear increase in the size of receptive fields with an increasing number of hidden layers. It is well noted that Wu *et al.* [9] integrated WaveNet [10] into GCNs for spatial–temporal graph modeling. With dilation convolution, it can cover the entire sequence with smaller layers efficiently. Ge *et al.* [20] took external factors, such as weather conditions and holidays, into account with graph models, and validated the effectiveness of extra information in traffic prediction.

B. Label Distribution Learning

LDL is proposed to address the challenges due to label ambiguity [34]. Tan *et al.* [35] employed multioutput regression and manifold learning to reduce the computational cost of multilabel data by dimension reduction and exploiting manifold's topological relationship. Different from multilabel learning, which is used to handle the problem where an instance can be associated with multiple labels simultaneously, LDL can assign a label distribution to an object and leverage the relative relationship among a sequence of values in label space, which can lead to more robust estimation. LDL has been utilized on many computer vision tasks, such as facial emotion recognition [36], head position estimation [37], age prediction [38]. He *et al.* [38] proposed a series of LDL approaches for age estimation and demonstrated the effectiveness in resolving the issues, such as data-dependent modeling. It has been demonstrated that one face image can contribute to not only the learning of the age but also the learning of the adjacent ages [39]. Geng and Xia [37] utilized multivariate label distribution to enhance the estimation accuracy without increasing the total size of the training dataset.

It is well noted that in [40], multiple loss functions for deep learning model training have been adopted and a novel mean–variance loss is proposed for facial age prediction, in which mean loss attempts to penalize the distance between the estimated age and the ground-truth age, while variance loss aims at minimizing the variance of the estimated age distribution. Hence, the curve of the age distribution can be sharpened by variance loss. However, the sharper distribution is not necessary to lead to a more accurate prediction directly, which may easily result in overfitting. Furthermore, in some cases, mean loss and variance loss suppress each other for an optimal distribution.

III. METHODOLOGY

In this section, we first present the mathematical definition of the traffic forecasting problem. Next, we introduce our speed prediction that has been redefined as a distribution learning followed by speed regression, where we propose the mean–residue loss for optimizing the target distribution learning. Finally, we introduce a graph-based architecture that incorporates the proposed loss for traffic prediction.

A. Problem Definition

Given observed speed data from sensors, the task of traffic prediction is to estimate the future speed at different sensor

Authorized licensed use limited to: Nanyang Technological University. Downloaded on December 18,2021 at 14:16:30 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

locations. The sensors and roads can be modeled in a graph $G = (V, E)$. A node $v_i \in V$ represents a sensor, and an edge $(v_i, v_j) \in E$ represents a road. A trainable feature matrix $X^t \in R^{N\times D}$ is associated to G, where X is the matrix, and t refers to time. Given the graph G and S steps of historical traffic data, the task is to learn a mapping function f, which forecasts traffic speed at the next T steps. The function f is presented as

$$[X^{(t-S):t}, G] \xrightarrow{f} X^{(t+1):(t+T)}. \tag{1}$$

A speed value is often taken as a single value in previous research [9], [19], [41], and the modeling of speed is considered as a regression task. However, regression-based methods, along with the corresponding loss function, for example, MAE, fail to utilize the ordinal correlations among different speed values. To overcome the limitations of MAE loss, we divide the continuous speed into multiple classes to learn the conditional distribution. Specifically, we use speed classification that rounds different speed values into classes representing speed values from zero up to the maximum rounded speed, with an interval of 1 mph. Since we have transformed the regression problem into a classification problem, we use softmax loss to minimize the difference between the ground-truth class and the prediction. Meanwhile, we further represent the distribution with a novel mean–residue loss, which will be discussed in the following sections. Finally, we perform regression based on classification and minimize the MAE loss. The loss functions in the classification task and the regression task are jointly used to update the model parameters [5].

B. Mean–Residue Loss

Normally, for classification problems, there is no particular requirement on the arrangement of labels, and the classes can be labeled arbitrarily, since all the classes are assumed to be independent and orthogonal [6], [42], [43]. However, in the case of traffic prediction where the speed value is continuous, there are strong correlations among different speed types. For example, if the speed can be classified as *low speed*, *middle speed*, and *high speed*, it is impossible for a vehicle to speed up from *low speed* to *high speed* without through a state of *middle speed*. To capture the correlations among the speed classes, we use $y_i \in \{0,\ 1, \dots,\ L-1\}$ to denote the corresponding speed label of the ith sample. The target conditional distribution is concentrated in one of the y_i classes, while the distribution is close to, or equals 0 in the rest of the classes. The shape of the sharp target distribution is different from common label distributions in label distribution learning. $L-1$ refers to the highest integer speed in the dataset, and the unit is mph. x_i denotes the feature vector, and $f(x_i) \in \mathbb{R}^{N\times M}$ denotes the output of the layer ahead of the classification branches. Let $z \in \mathbb{R}^{N\times L}$ denote the output of the classification and then, we have $p \in \mathbb{R}^{N\times L}$, a typical softmax probability as follows:

$$z = f(x_i)\cdot\theta^T, \quad p_{i,j} = \frac{e^{z_{i,j}}}{\sum_{l=1}^{L} e^{z_{i,l}}} \tag{2}$$

where $\theta \in \mathbb{R}^{L\times M}$ is the trainable parameter of the linear layer, $j \in \{0, 1, \dots, L-1\}$ denotes the class label as well as the corresponding speed, and $z_{i,j}$ is an element of z for the ith sample

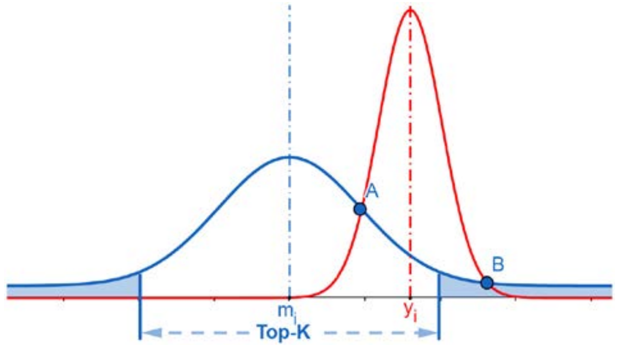


Fig. 2. Conditional distribution during training and the ideal distribution for the ith sample are illustrated by the blue curve and the red curve, respectively. After top-K pooling operation, two long tails are left as indicated by the shadowed areas. redPoint A and B are two points in which the values of two distributions are the same.

and speed j. Hence, p_i can denote the estimated conditional distribution for sample i over all the L classes, and then, $p_{i,j}$ can denote the probability that sample i may belong to speed j.

Based on (2), we can have the estimated mean speed of sample i, m_i, as follows:

$$E(\text{speed}_i) = m_i = \sum_{j=1}^{L} j * p_{i,j}. \tag{3}$$

We also can add a learnable parameter w_b into the traffic prediction to make some final adjustment and (3) can be transformed to

$$E(\text{speed}_i) = m_i = \sum_{j=1}^{L} j * p_{t,j} + w_b. \tag{4}$$

We then carry out the top-K pooling operation on p_i and, one or two tails with $L-K$ classes left as shown in Fig. 2. The residue entropy in the tail(s) can be calculated as

$$r_i = -\sum_{j=1,\ j\notin\text{top-}K}^{L} p_{i,j} * \log p_{i,j}. \tag{5}$$

To help the model concentrate more on the top-K classes, we need to reduce r_i. Once K is determined, a smaller r_i is helpful to obtain a smaller confidence interval with a higher confidence level.

As shown in Fig. 2, the proposed mean–residue loss aims to penalize not only the difference between the mean of an estimated conditional distribution and the ground-truth speed but also residue error existing in the long tails.

Mean Loss: Similar to [40], we have the mean loss component in the proposed mean–residue loss that can penalize the difference between the mean of an estimated conditional distribution and the ground-truth speed. Based on (3), we can have the mean loss as

$$L_m = \frac{1}{2N}\sum_{i=1}^{N}(m_i - y_i)^2 = \frac{1}{2N}\sum_{i=1}^{N}\left(\sum_{j=1}^{L} j * p_{i,j} - y_i\right)^2 \tag{6}$$

where N is the batch size in training process. Unlike the softmax loss, which is widely used for classification tasks, the

Authorized licensed use limited to: Nanyang Technological University. Downloaded on December 18,2021 at 14:16:30 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

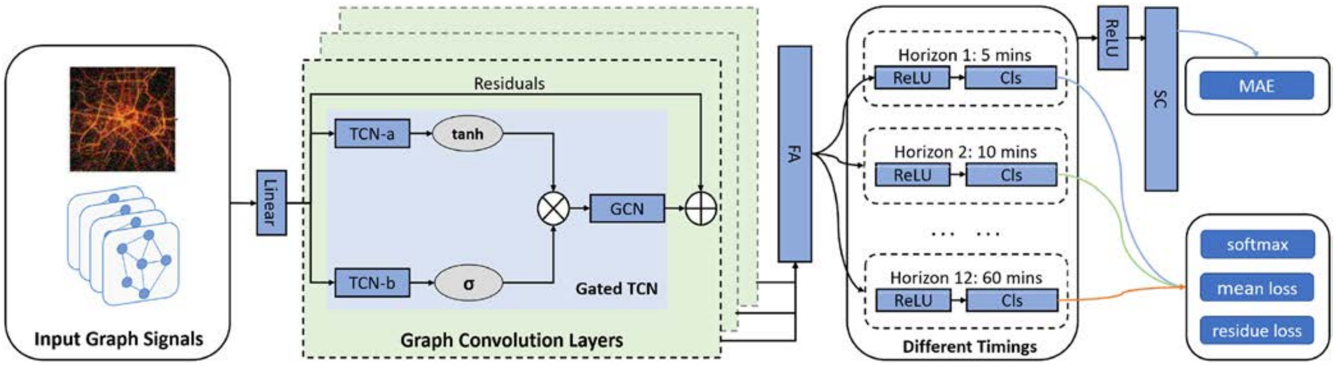


Fig. 3. Our proposed speed prediction architecture. We follow the graph network in Graph-WaveNet as the feature extractor, which is attached to 12 classification branches. Each branch handles prediction over a specific time horizon. A shared regression layer outputs the final speed with high precision. During training, softmax and mean–residue loss are computed at the classification level, while MAE loss is calculated at the regression level. Abbreviations: TCN (temporal convolution layer), GCN (graph convolution networks), FA (feature aggregation), SC (speed calculation), and Cls (classifier).

mean loss focuses on regression tasks. We can use L_2 distance to measure the difference between the mean of an estimated conditional distribution and the ground-truth speed. Hence, the proposed mean loss is complementary to the softmax loss. Besides, the mean loss is also an reinforcement to the MAE loss in the speed regression, discussed in the next section.

Residue Loss: The residue loss in the proposed mean–residue loss penalizes the residue error in the tails that exists in an estimated conditional distribution after top-K pooling operation. In the training process, top-K and the value of K can be fixed to a fixed value, for example, top-5. Based on (5), the residue entropy loss can be computed as follows:

$$L_r = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1,\ j\notin \text{top-}K}^{L} p_{i,j} * \log p_{i,j}. \tag{7}$$

Under the joint effect of mean and residue loss, the conditional distribution is moved to class $\lfloor y_i \rfloor$, and narrowed to this class closely. Ideally, all the probabilities of the classes beyond y_i shall be zero, and p_{i,y_i} shall be maximized to be 1. The expected shape of the distribution is known, unlike arbitrary shapes in general LDL tasks. In the proposed residue loss L_r, it can be guaranteed that y_i falls in the top-K classes, when K is large enough to include y_i in top-K.

C. Mean–Residue Loss Implementation

Speed estimation is a complicated and challenging process due to the dynamic traffic conditions. In this work, we focus on the loss function and assume that the graph network architecture has been predetermined, as shown in Fig. 3. We choose Graph-WaveNet as our basic feature representation learning to model the speed estimation process. Especially, we can combine the softmax loss, denoted as L_s and mean–residue loss in the classification task jointly as the supervision signal in the following:

$$\begin{aligned} L_{mr} &= L_s + \lambda_1 L_m + \lambda_2 L_r \\ &= \frac{1}{N}\sum_{i=1}^{N}\left[-\log p_{i,y_i} + \frac{\lambda_1}{2}(m_i - y_i)^2 + \lambda_2 r_i\right] \end{aligned} \tag{8}$$

where L_s denotes the softmax loss. L_m and L_r are the mean loss and residule loss, respectively. λ_1 and λ_2 are scaling factors for L_m and L_r for model optimization. In Section V, various combinations of λ_1 and λ_2 are experimented to optimize the model. Based on (8), we can have the final MAE loss of our method as the training objective as shown in Fig. 3, which has been used in many recent research work on traffic prediction [9], [15]. The final MAE loss can be calculated as

$$L_{\text{MAE}} = \frac{1}{N}\sum_{i=1}^{N}|m_i - y_i|. \tag{9}$$

The final loss function is presented in 10, and the value of λ_3 is set to 1 in the experiments. λ_3 scales the ratio between the mean residue loss and the MAE loss for model optimization. The MAE loss optimizes the regression process following the speed classification:

$$L_{\text{final}} = L_{mr} + \lambda_3 L_{\text{MAE}}. \tag{10}$$

D. Implementation for Structured Prediction

We implement a graph-based architecture to fulfill the proposed loss for speed prediction. As illustrated in Fig. 3, our design of the network is based on Graph-WaveNet [9], to capture conditional distribution in hidden representations of the network. A classification layer for each time horizon represents the conditional distribution, which is then calibrated to speed prediction.

The scheme is presented in Algorithm 1. Given the spatial–temporal feature matrix X_t, the algorithm learns the speed label distribution from classification results c, and speed regression results p at each time interval from 1 to 12. Next, mean residue loss L_{mr} and MAE L_{mae} are calculated, respectively, to update the model parameters. The process iterates until the maximum number of epochs is reached.

Specifically, we reuse its dynamic adjacency matrix and graph network components, denoted as F_{tl}, in Algorithm 1, featuring dilated TCNs with gating mechanisms and skip connections to extract spatial and temporal features. Next, we arrange our classification and regression as follows. We create 12 branches after the representation layer of the graph network to predict the speed at all locations in the next 12 time horizons, where each branch corresponds to one time

Authorized licensed use limited to: Nanyang Technological University. Downloaded on December 18,2021 at 14:16:30 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Algorithm 1: Training Scheme Incorporating Mean–Residue Loss and MAE

```
Input: Graph $G = (V, E)$, feature matrix $X_t \in R^{N\times D}$,
       spatial temporal learning function $\mathcal{F}_{tl}$, classifier
       $\mathcal{F}_c$, forecast $\mathcal{F}_f$
Initialization: classification result c, classification results
C, predict result p, predict results P
for $epoch = 1$ to $epochs$ do
    $C = \{\}$, $P = \{\}$, $L_{mr} = 0$, $L_{mae} = 0$
    //Learning spatial and temporal features
    $ft \leftarrow \mathcal{F}_{tl}(G, Xt)$;
    for $t = 1$ to $\mathcal{T} = 12$ do
        // Target distribution learning
        $ft^{'}, c \leftarrow \mathcal{F}_c(ft)$;
        Add c to classification results C
        // Regression learning
        $p \leftarrow \mathcal{F}_f\left(ft^{'}\right)$;
        Add p to predict results P
    end
    $L_{mr} \leftarrow$ update $L_{mr}$ on $C$;
    $L_{mae} \leftarrow$ update $L_{mae}$ on $P$;
end
```

horizon. Each branch consists of a ReLU activation followed by a classification layer called F_c in Algorithm 1, which transforms the feature into an intermediate representation of size [*batch_size*, *number_of_class*, *number_of_sensor*]. For each sensor at each time interval, the input feature is converted to a probability distribution over a range of speed classes, rounded to integers. The range is from zero to the maximum speed (rounded) in the dataset. The intermediate features are then forwarded into the last regression network with shared weights across the branches, corresponding to F_f in Algorithm 1. The approach to sharing weights exploits invariant spatiotemporal patterns in conditional distribution, which could benefit feature extraction. Based on the distribution learning results, the regression layer calculates a single value that is a continuous speed prediction for each sensor at each time step.

We design a new training scheme to incorporate the mean–residue loss, which is described in Algorithm 1. At the classification level, softmax loss and mean–residue loss, as shown in (8), are computed. At the regression level, MAE loss is calculated. The final loss includes both of them. All components in the final loss are backpropagated together, by which the model parameters are updated. The branches are trained with targets at their respective time horizons, while the graph network and the regression layer are updated by all the training instances. The loss functions and their corresponding network components are also illustrated in Fig. 3. There are 12 values in each target of the dataset, corresponding to traffic speed in the next hour with 5-min intervals. The 12 classifier network matches the format of the desired output. The architecture also makes it more comparable with similar multibranch networks, for example, Graph WaveNet.

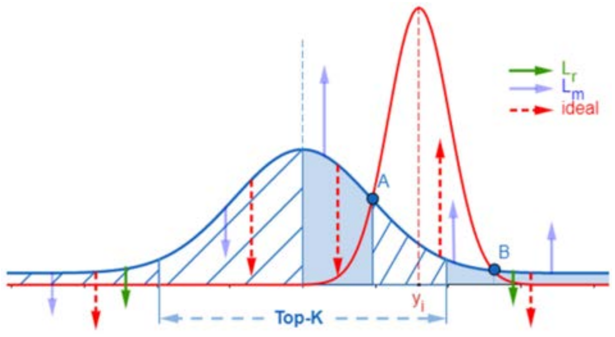


Fig. 4. Gradient analysis on the proposed mean–residue loss. Areas with hatching lines indicate the ranges that L_m and L_r can work together toward the optimum conditional distribution, while blue areas indicate the ranges that L_m and L_r fail to work jointly toward the optimum conditional distribution. Points A and B are two points in which the values of two distributions are the same.

IV. Gradient Analysis

As shown in Fig. 4, mean loss attempts to reduce the difference between the mean speed of an estimated conditional distribution and the ground-truth speed. According to (3), we can have the gradient of mean loss L_m, with respect to $p_{i,j}$ as follows:

$$\frac{\partial L_m}{\partial p_{i,j}} = \frac{1}{N} * j * (m_i - y_i). \tag{11}$$

Based on (2), we can obtain the gradient of $p_{i,j}$ with respect to $z_{i,l}$ as

$$\frac{\partial p_{i,j}}{\partial z_{i,l}} = \begin{cases} \frac{e^{z_{i,l}}}{\sum_{j=1}^{L} e^{z_{i,j}}} - \left(\frac{e^{z_{i,l}}}{\sum_{j=1}^{L} e^{z_{i,j}}}\right)^2 = p_{i,l} - p_{i,l}^2, & \text{if } l = j \\ \frac{e^{z_{i,l}}}{\left(\sum_{j=1}^{L} e^{z_{i,j}}\right)^2} * e^{z_{i,l}} = -p_{i,l} * p_{i,j}, & \text{if } l \neq j. \end{cases} \tag{12}$$

Based on (11) and (12), the gradient of L_m with respect to $z_{i,j}$ can be computed as follows:

$$\begin{aligned}\frac{\partial L_m}{\partial z_{i,j}} &= \frac{m_i - y_i}{N}\left(j * \left(p_{i,j} - p_{i,j}^2\right) - \sum_{l=1,\ l\neq j}^{L} j * p_{i,l} * p_{i,j}\right) \\ &= \frac{(m_i - y_i)}{N} p_{i,j} * (j - m_i).\end{aligned} \tag{13}$$

According to (13), if the mean value m_i of an estimated distribution is less than y_i, the probabilities of the classes $j \geq \lceil m_i \rceil$ will be increased via their negative gradients, while the probabilities of the classes $j \leq \lfloor m_i \rfloor$ will decrease via the positive gradients when the network is updated. Hence, m_i of the estimated conditional distribution will become greater and the distance between m_i and y_i will become smaller. On the contrary, if $m_i > y_i$, the probabilities of the conditional distribution will be updated to achieve smaller m_i and push m_i closer to y_i too. Furthermore, if $m_i = y_i$, the gradients will be 0 and the network will do nothing to update the speed probability distribution. The residue loss aims at reducing the probability of class j, if $j \notin$ top$-K$. Based on (7), we can have

Authorized licensed use limited to: Nanyang Technological University. Downloaded on December 18,2021 at 14:16:30 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

the gradient of residue loss L_r with respect to $p_{i,j}$ as follows:

$$\begin{aligned}\frac{\partial L_r}{\partial p_{i,j}} &= -\frac{1}{N}\sum_{l}\frac{\partial\big(p_{i,l} * \log p_{i,l}\big)}{\partial p_{i,j}}\\ &= -\frac{1}{N}\sum_{l}\left(\log p_{i,l} * \frac{\partial p_{i,l}}{\partial p_{i,j}} + p_{i,l} * \frac{\partial \log p_{i,l}}{\partial p_{i,l}} * \frac{\partial p_{i,l}}{\partial p_{i,j}}\right)\\ &= -\frac{1}{N}\sum_{l}\big(\log p_{i,l} + 1\big)\frac{\partial p_{i,l}}{\partial p_{i,j}} \qquad (14)\end{aligned}$$

where $l = 0, 1, \ldots, L-1$ & $l \notin$ top-K. According to (2) and (14), we can have

$$\frac{\partial L_r}{\partial z_{i,j}} = \frac{\partial L_r}{\partial p_{i,j}}\frac{\partial p_{i,j}}{\partial z_{i,j}} = -\frac{1}{N}\sum_{l}\big(\log p_{i,l} + 1\big)\frac{\partial p_{i,l}}{\partial z_{i,j}}. \qquad (15)$$

Assume $p_i^* = \max\{p_{i,l},\ l = 1, 2, \ldots, L$ & $l \notin$ top-$K\}$. From (12) and (15), we have the following inequation:

$$\begin{aligned}\frac{\partial L_r}{\partial z_{i,j}} &> -\frac{1}{N}\sum_{l}\big(\log p_i^* + 1\big)\frac{\partial p_{i,l}}{\partial z_{i,j}}\\ &= -\frac{\log p_i^* + 1}{N} * p_{i,j} * \left(1 - \sum_{l\notin \text{top-}K} p_{i,l}\right)\\ &> 0. \qquad (16)\end{aligned}$$

Obviously, if $K \geq 2$, we have $p_i^* \leq (1/3) < e^{-1}$ and hence, $\log p_i^* + 1 < 0$ and (16) holds. Let K_i denote the value of top-K and R_{y_i} denote the ranking of y_i, for the ith sample in a batch training. Then, we can fix K_i to be some value.

Equation (16) illustrates the network will be updated to decrease the probability of speed j close to 0, for any j out of top-K speed values, via the positive gradient. L_r does not touch the probabilities of top-K classes, but indirectly increases the probability of top-K classes as a whole. So, L_r can help L_s to focus more on y_i and boost L_m to push m_i closer to y_i. The joint effect of the mean and residue loss is summarized as follows, in the increasing order of speed, that is, the following points correspond to regions in Fig. 4 from left to right.

1) For classes smaller than m_i and outside top-K, both loss functions decrease the probabilities, which ideally should decrease. They optimize the model in the same, correct direction.
2) For classes smaller than m_i and inside top-K, the mean loss decreases the probabilities, and the residue loss is 0. These probabilities ideally should decrease. Therefore, the loss functions optimize the model in the same, correct direction.
3) For classes between m_i and point A in Fig. 4, which is the smallest class (exclusive) where the predicted probability is smaller than the ground truth, the mean loss increases the probabilities, and the residue loss is 0. These probabilities ideally should decrease. Therefore, the loss functions optimize the model in the same, but incorrect direction. If A is outside top-K, the residue loss could be 0 (to the left of A) or decreases the probabilities (to the right of A). In this case, the loss functions optimize the model in different directions, and the joint effect could be positive or negative.
4) For classes between point A (inclusive) and the boundary class (inclusive) of top-K, the mean loss increases the probabilities, and the residue loss is 0. Ideally, the probabilities should increase. Therefore, the loss functions optimize the model in the same, correct direction.
5) For classes between the boundary class of top-K and point B (exclusive), which is the smallest class where predicted probability is greater than the ground truth, the mean loss increases the probabilities, but the residue loss decreases the probabilities. Ideally, the probabilities should increase. Therefore, the loss functions conflict with each other, and the joint effect is unknown. If B is within top-K, the residue loss is 0, and the joint effect would be in the wrong direction.
6) For classes to the right of point B, the mean loss increases the probabilities, while the residue loss decreases the probabilities. Ideally, the probabilities should decrease. Therefore, the joint effect could be positive or negative.

V. Experimental Results

In this section, we shall carry out experiments to compare our proposed method with state-of-the-art baselines for traffic prediction. In the experiments, we use MAE, root mean squared error (RMSE), and mean absolute percentage error (MAPE) as our evaluation metrics [2], [9], [10], [15], [19], which are presented as follows:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \hat{y}_i\right| \qquad (17)$$

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\big(y_i - \hat{y}_i\big)^2} \qquad (18)$$

$$\text{MAPE} = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \qquad (19)$$

where n is the number of instances, $\hat{y}_i$ is the prediction result, and y_i is the ground truth. Missing values are ignored both during training and testing. The above-mentioned metrics are evaluated at 12 horizons, from 5 to 60 min, with a 5-min interval. The overall metrics are the average of all metrics at the 12 horizons.

A. Datasets

In our experiments, we evaluate our approach on three popular public traffic speed datasets, which are widely used in the traffic prediction domain, as shown in Fig. 5. In the following, we shall give a brief introduction to the datasets.

1) *METR-LA [15]:* This dataset contains traffic speed data recorded by sensors on Los Angeles highways. We select data from 207 sensors in a period of four months from March 1, 2012 to June 30, 2012 in our work. The total number of speed data points is 6 519 002. The sensor distribution is illustrated in Fig. 5(a).
2) *PeMS-BAY [15]:* This dataset contains traffic speed information collected by California Transportation

Authorized licensed use limited to: Nanyang Technological University. Downloaded on December 18,2021 at 14:16:30 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

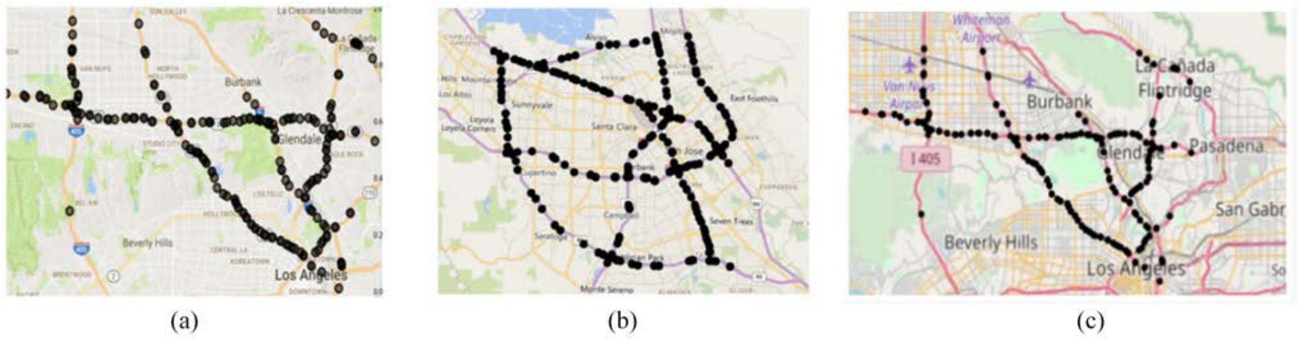


Fig. 5. Illustration of sensor distributions in METRA-LA, PeMS-Bay, and PeMSD7 datasets. (a) METR-LA dataset. (b) PeMS-Bay dataset. (c) PeMSD7 dataset.

Agencies (CalTrans) Performance Measurement System (PeMS) in the San Francisco Bay area. We select data from 325 sensors in the six-month range. The total number of speed data points is 16 937 179. The sensor distribution of the dataset is shown in Fig. 5(b).

3) *PeMSD7 [20]:* This dataset contains data collected from sensors on the highways in Los Angeles County, California. It is also collected real time by PeMS at over 39 000 sensor locations [2]. Following the configurations in [2], 228 sensor locations are selected, and the time spans the weekdays in May and June 2012. Fig. 5(c) presents the sensor distribution of the dataset.

All datasets contain real-world traffic data with large spatial–temporal scales and are widely adopted for traffic analysis [15]. We follow the same data-processing steps described in [9] to aggregate data and construct the adjacency matrix. The datasets are split into 70% for training, 10% for validation, and 20% for testing.

B. Baselines

We compare our method with the following approaches (as shown in Table I), which can be considered as state-of-the-art baselines.

1) *FC-LSTM:* The model is a variation of an RNN with fully connected LSTM hidden units [15].
2) *WaveNet:* A convolution network architecture for sequential data, first applied on audio generation, text, and speech analysis [10].
3) *Diffusion Convolution RNN (DCRNN) [15]:* It models traffic flow as a diffusion process. The network uses an encoder–decoder architecture and combines graphical convolution networks with RNNs.
4) *Graph Gated Recurrent Unit Network (GGRU) [19]:* The approach is recurrent based. It uses a convolutional subnetwork to guide each attention head's importance and is integrated with spatial–temporal graph networks.
5) *Spatial–Temporal Graph Attention Network (STGCN) [2]:* The model combines graph convolution with 1-D convolution to extract spatial–temporal features.
6) *Graph-WaveNet [9]:* The architecture features an adaptive adjacency matrix and dilated 1-D convolutions with exponentially larger receptive fields. The method is capable of handling very long sequences.

C. Experimental Setups

Our models are trained and evaluated on a single GPU device (NVIDIA RTX 2080Ti and NVIDIA Quadro P5000). Following the method adopted in Graph-WaveNet [9], we use the Adam method to optimize the parameters of the neural network, with an initial learning rate of 0.001. Adopting the technique in [44], a 3% decay in learning rate is added during training. There is a dropout rate of 0.3 in the GCN. For fine-tuning experiments, we first reproduce the experimental results of Graph-WaveNet, using the settings described in [9], and load the model weights in the graph convolution part of the network. Next, the model is fine-tuned for 100 epochs. For experiments without pretrained models, we randomly initialize the network and node embedding with a uniform distribution of size 10.

D. Hyperparameter Analysis

From our gradient analysis, we can observe that the hyperparameters, K and λ, can affect the performance of the proposed method, and different combinations of K and λ may lead to different results. Hence, we first search for the optimal combination of the hyperparameters, and then carry out our experiments based on the optimal values of the hyperparameters. In the following, we present the detailed analysis on the hyperparameters K and λ, respectively.

1) Analysis on Hyperparameter K: In the proposed mean–residue loss, the residue error is penalized by varying degrees with different K. Specifically, a large K sets a big confidence interval, which leads to a low confidence level. On the other hand, a small K may lead to an opposite direction away from the ground-truth speed once the ground-truth speed falls out of top $-K$. Therefore, a suitable K can guide loss convergence in the right direction. In Fig. 6, we present the study on the value selection of K in the residue loss and the influence on the model performance. It can be inferred from the figure that our proposed loss function can optimize the network better than the original MAE loss in [9], even with a nonoptimal K value. It is also demonstrated that the model performance at $K = 11$ is better than the rest, and the performance generally drops when K deviates from 11. Comparing Fig. 6 and

Authorized licensed use limited to: Nanyang Technological University. Downloaded on December 18,2021 at 14:16:30 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE I
PERFORMANCE COMPARISON OF OUR APPROACH AND BASELINE MODELS. OUR APPROACH REACHES THE BEST RESULTS ON METR-LA AND PEMSD7, AND HAS THE BEST OVERALL METRICS ON PEMS-BAY. THE RESULTS REPORTED ARE FROM MODELS WITH RANDOMLY INITIALIZED WEIGHTS. THE LAST COLUMN REFERS TO RESULTS AT 60 MIN FOR META-LA AND PEMS-BAY, AND 45 MIN FOR PEMSD7

Data	Models	15 min			30 min			45/60 min		
		MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
METR-LA	FC-LSTM [15]	3.44	6.30	9.60%	3.77	7.23	10.90%	4.37	8.69	13.20%
	WaveNet [10]	2.99	5.89	8.04%	3.59	7.28	10.25%	4.45	8.93	13.62%
	STGCN [2]	2.88	5.74	7.62%	3.47	7.24	9.57%	4.59	9.40	12.70%
	DCRNN [15]	2.77	5.38	7.30%	3.15	6.45	8.80%	3.60	7.60	10.50%
	GGRU [19]	2.71	5.24	6.99%	3.12	6.36	8.56%	3.64	7.65	10.62%
	Graph-WaveNet [9]	2.69	5.15	6.90%	3.07	6.22	8.37%	3.53	7.37	10.01%
	Ours	**2.59**	**4.86**	**6.80%**	**2.97**	**5.79**	**8.18%**	**3.37**	**6.70**	**9.68%**
PeMS-BAY	FC-LSTM [15]	2.05	4.19	4.80%	2.20	4.55	5.20%	2.37	4.96	5.70%
	WaveNet [10]	1.39	3.01	2.91%	1.83	4.21	4.16%	2.35	5.43	5.87%
	STGCN [2]	1.36	2.96	2.90%	1.81	4.27	4.17%	2.49	5.69	5.79%
	DCRNN [15]	1.38	2.95	2.90%	1.74	3.97	3.90%	2.07	4.74	4.90%
	Graph-WaveNet [9]	1.30	2.74	2.73%	1.63	3.70	3.67%	1.95	4.52	4.63%
	Ours	**1.30**	**2.55**	**2.70%**	**1.62**	**3.31**	**3.58%**	**1.92**	**3.94**	**4.45%**
PeMSD7	FC-LSTM [15]	3.57	6.20	8.60%	3.94	7.03	9.55%	4.16	7.51	10.10%
	GGRU [19]	2.37	4.21	5.54%	3.31	5.96	8.06%	4.01	7.13	9.99%
	STGCN [2]	2.26	4.07	5.24%	3.09	5.77	7.39%	3.79	7.03	9.12%
	Graph-WaveNet [9]	2.15	4.05	5.04%	2.77	5.45	6.93%	3.10	6.16	7.96%
	Ours	**2.13**	**3.95**	**5.01%**	**2.73**	**5.30**	**6.86%**	**3.04**	**5.95**	**7.86%**

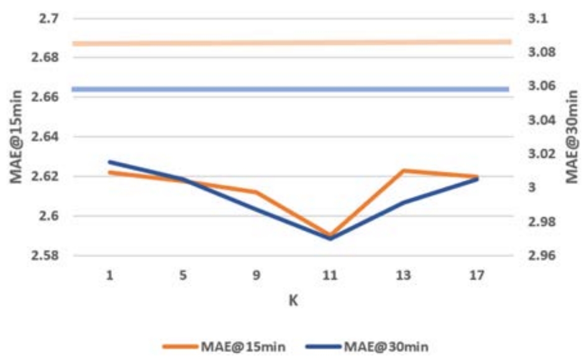


Fig. 6. Analysis on model performance with different K values in the residual loss measured by test MAE, on the dataset METR-LA. The colored horizontal lines indicate the performance of Graph-WaveNet.

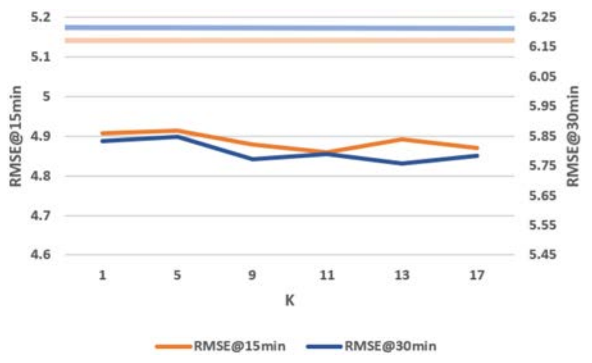


Fig. 7. Analysis on model performance with different K values in the residual loss measured by test RMSE, on the dataset METR-LA. The colored horizontal lines indicate the performance of Graph-WaveNet.

Fig. 7, we can observe that the optimal K only has a small difference under different evaluation metrics. In addition, the choice of other experiment settings and hyperparameters also leads to tiny fluctuations on the choice of K. Based on the above observation, we fix $K = 11$ for all the experiments on the three datasets.

2) Analysis on Hyperparameter λ: We also provide an ablation study on λ values in the loss function (8). In particular, the ratio of the combination of loss functions is a crucial set of parameters that affect model performance. The λ values scale the losses to the same magnitude level and assign them different weights in the optimization process. We did a grid search for λ_1 and λ_2, which scales mean loss and residue loss, respectively. The experimental settings are described in Section V-C, and particularly, we set $\lambda_3 = 1$. The results are presented in Table III. The model is optimized when λ_1 and λ_2 are set to 1 and 0.01, respectively. In addition, the selection of λ_3 is presented in Table II, when λ_1 and λ_2 are fixed at 1 and 0.01, respectively. λ_3 scales the MAE loss during the regression stage. In Table II, the model is optimized when $\lambda_3 = 1$. Based on the experimental results above, we set the values of λ_1, λ_2, and λ_3 to be 1, 0.01, and 1, respectively, for all the experiments on the three datasets.

E. Experimental Results and Discussions

In our experiments, we choose fixed hyperparameters in the loss function. We keep a fixed value of $K = 11$ for the top$-K$ selection in residual loss. The scale factors, λ_1 for mean loss and λ_2 for residual loss, are set to 1 and 0.01, respectively.

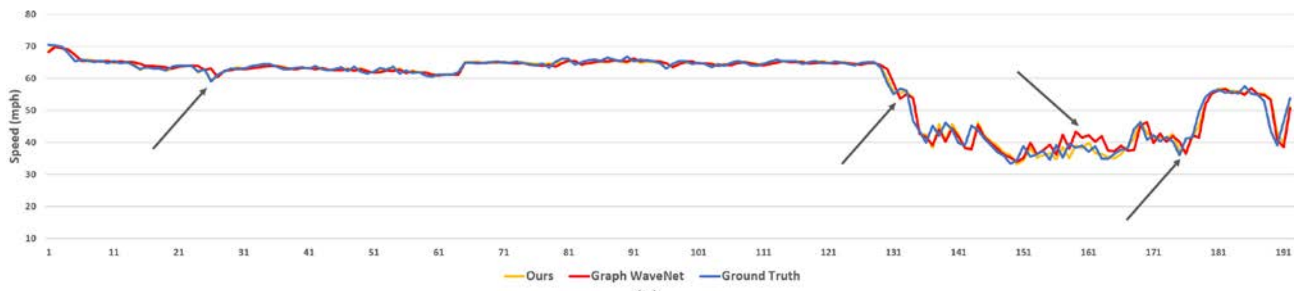
Based on the optimal combination of K and λ, we carry out our experiments on the three datasets. In Table I, we compare the performance of the proposed approach with the baseline models. More specifically, we present evaluation results of different metrics at 15-, 30-, and 60-min time intervals on METR-LA and PeMS-BAY datasets and at 15-, 30-, and 45-min time intervals on the PeMSD7 dataset. The proposed approach has achieved the best results among these methods with respect to MAE, RMSE, and

Authorized licensed use limited to: Nanyang Technological University. Downloaded on December 18,2021 at 14:16:30 UTC from IEEE Xplore. Restrictions apply.

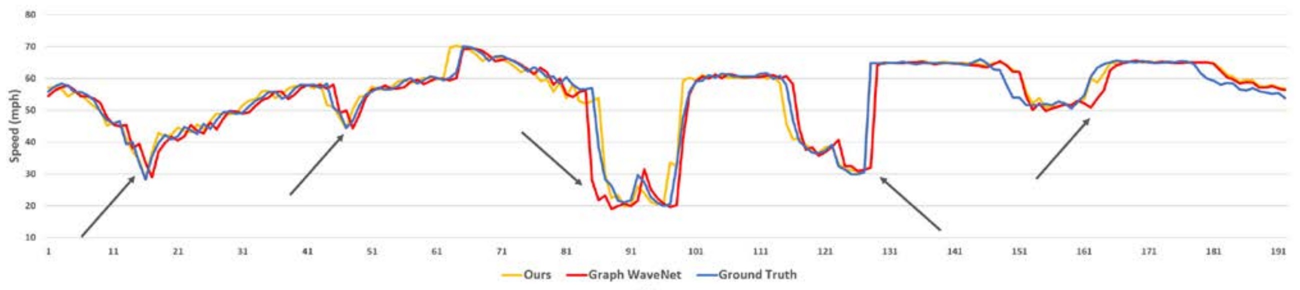
This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE II
EXPERIMENTAL RESULTS WITH DIFFERENT λ_3, ON THE DATASET METR-LA

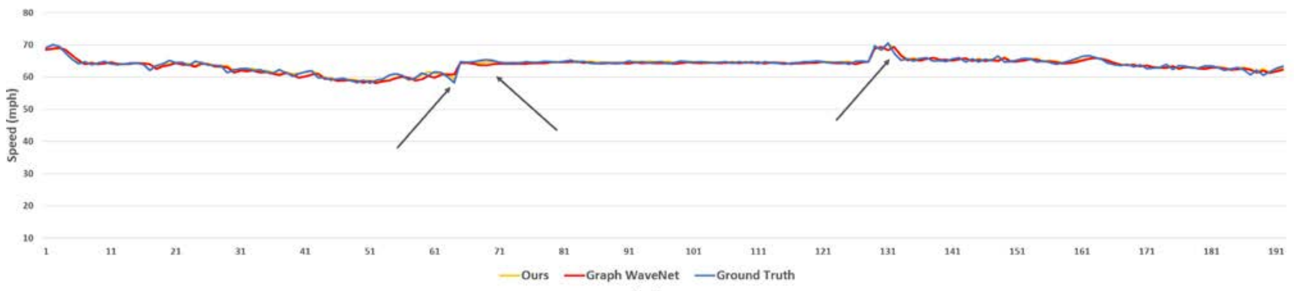
λ_3		15 min			30 min			60 min	
	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
0.5	2.85	5.03	7.30%	3.46	6.37	9.15%	3.45	7.25	11.32%
1	**2.59**	**4.86**	**6.80%**	**2.97**	**5.79**	**8.18%**	**3.37**	**6.70**	**9.68%**
1.5	2.85	5.04	7.28%	3.18	6.30	9.10%	3.44	7.23	11.39%
2	2.95	5.54.	7.72%	3.57	6.78	9.97%	4.41	8.24.	13.28%



(a)



(b)



(c)

Fig. 8. Comparisons of prediction curves between the proposed method (yellow) and Graph-WaveNet (red) for 5 min ahead prediction on a snapshot of the test datasets of METR-BAY, PeMS-BAY, and PeMSD7. The arrows indicate where our result is clearly closer to the ground truth. Performance comparison (a) on METR-LA dataset, (b) on PeMS-BAY dataset, and (c) on PeMSD7 dataset.

MAPE metrics. Our approach outperforms temporal models, including FC-LSTM and WaveNet, as well as both convolution and recurrent-based spatial–temporal models, that is, STGCN, DCRNN, and GGRU. In comparison to Graph-WaveNet for different time horizons, we can observe that better performance has been achieved on larger time horizons, which proves that the proposed method can achieve improved performance on capturing information from longer sequences. We have also reached a noticeable performance increment from Graph WaveNet, compared to its major baseline, DCRNN [15], at all time horizons. Hence, it has been proven that our proposed approach with supervision on both classification and regression tasks can improve the performance of traffic prediction significantly.

TABLE III
EXPERIMENTAL RESULTS WITH DIFFERENT COMBINATIONS OF λ_1 AND λ_2. THE TEST MAE IS AT THE 15-MIN INTERVAL, ON THE DATASET METR-LA

λ_1 \ λ_2	0	0.005	**0.01**	0.015	0.02	0.1
0	2.65	2.61	2.61	2.63	2.65	2.66
0.5	2.64	2.62	2.60	2.63	2.64	2.65
1	2.63	2.62	**2.59**	2.62	2.63	2.64
1.5	2.64	2.62	2.61	2.63	2.65	2.65
2	2.65	2.63	2.62	2.63	2.65	2.66
2.5	2.66	2.66	2.63	2.63	2.66	2.668

For more intuitive comparison, we visualize the results of 5-min-ahead predicted values versus real values of Graph-WaveNet and the proposed method on a snapshot of

Authorized licensed use limited to: Nanyang Technological University. Downloaded on December 18,2021 at 14:16:30 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE IV
EXPERIMENTAL RESULTS ON DIFFERENT CONDITIONAL DISTRIBUTIONS. METRICS ARE EVALUATED AT THE 60-MIN INTERVAL, ON THE DATASET METR-LA

Model	**Graph WaveNet**			**Ours**		
Evaluation	**MAE**	**RMSE**	**MAPE**	**MAE**	**RMSE**	**MAPE**
Distribution 1	3.53	7.37	10.01%	**3.37**	**6.70**	**9.68%**
Distribution 2	4.93	9.70	15.37%	**4.79**	**8.88**	**14.69%**
Distribution 3	4.79	9.43	14.97%	**4.68**	**8.59**	**14.77%**

TABLE V
HYPOTHESIS TEST RESULTS ON THE METR-LA DATASET, AT THREE TIME INTERVALS, WITH ε COMPARED WITH AVERAGED GRAPH WAVENET RESULTS

Time interval	μ	ε	$\sigma^2(1e-4)$	τ_t	$1-\alpha$
15 min	2.59	$2.61 < 2.69$	0.11	1.88	0.79
30 min	2.97	$3.01 < 3.07$	0.27	2.13	0.84
60 min	3.37	$3.49 < 3.53$	10.51	1.34	0.67

TABLE VI
TIME COMPLEXITY ANALYSIS. THE TRAIN AND TEST TIME ARE MEASURED IN 1 EPOCH, ON THE DATASET METR-LA, FOR ONE TIME HORIZON

Method	**Train time**	**Test time**
Graph WaveNet	17.209	1.54
Ours	42.96	1.56

TABLE VII
EXPERIMENTAL RESULTS ON EXCLUSION OF LOSS FUNCTIONS, THE TEST MAE IS AT THE 15-MIN INTERVAL, ON THE DATASET METR-LA

Experiments	**MAE**	**MAPE**	**RMSE**
Graph-WaveNet baseline	2.69	5.15	6.9%
With mean loss only	2.62	4.92	6.8%
With residue loss only	2.61	4.88	6.8%
With mean and residue loss together	**2.59**	**4.86**	**6.8%**

TABLE VIII
EXPERIMENTAL RESULTS ON EXCLUSION OF LOSS FUNCTIONS, THE TEST MAE IS AT THE 30-MIN INTERVAL, ON THE DATASET METR-LA

Experiments	**MAE**	**MAPE**	**RMSE**
Graph-WaveNet baseline	3.12	6.36	8.37%
With mean loss only	3.02	5.90	8.34%
With residue loss only	3.00	5.84	**8.08%**
With mean and residue loss together	**2.97**	**5.79**	8.18%

TABLE IX
EXPERIMENTAL RESULTS ON EXCLUSION OF LOSS FUNCTIONS, THE TEST MAE IS AT THE 60-MIN INTERVAL, ON THE DATASET METR-LA

Experiments	**MAE**	**MAPE**	**RMSE**
Graph-WaveNet baseline	3.53	7.37	10.01%
With mean loss only	3.44	6.86	9.88%
With residue loss only	3.46	6.88	9.72%
With mean and residue loss together	**3.37**	**6.70**	**9.68%**

the test datasets on randomly selected time horizons in Fig. 8, which presents the predicted values and the ground truth on METR-LA, PeMS-BAY, and PeMSD7 datasets, respectively. In Fig. 8, we can observe that the proposed approach generates more accurate predictions than Graph WaveNet, especially at where the arrows point out. Particularly, when the ground-truth speed changes abruptly, corresponding to sharp spikes in data, our predictions are well adapted to data fluctuations and match closely with the real values at all time horizons. Meanwhile, Graph WaveNet has difficulty in predicting such speed changes.

We also investigate the effects of some implementation details. We find that a ReLU activation after the classification results significantly boosts the model performance. We also verify that additional fully connected layers in SC offer no performance enhancement. Moreover, we confirm that with enough training epochs, randomly initialized models slightly outperform models with pretrained Graph-WaveNet weights. Therefore, our approach can be performed in an end-to-end manner with random initialization.

To validate the robustness of our approach, we follow the method suggested in [45] and compare the performance of Graph WaveNet and our approach on different conditional distributions in Table IV. The distributions are generated as follows.

1) *Distribution 1:* The same distribution in the METR-LA dataset.
2) *Distribution 2:* In the METR-LA dataset, we observe that most high-speed values are in the range [50, 70] mph. Therefore, we rank data points by the number of speed values that fall in the range of [50, 70] from low to high. Next, we select 4000 consecutive data points starting from the *lowest*.
3) *Distribution 3:* Similar to the steps in Distribution 2, we select 4000 consecutive data points starting from the *highest*.

The results in Table IV show that our method is clearly superior to Graph WaveNet with all the distributions and in all evaluation metrics. The result verifies the robustness and wide applicability of our method when the distribution changes.

To show the significance of prediction performance, we follow practice in [46] and [47] and present hypothesis test results in Table V. In Table V, most ε are smaller than the result of Graph WaveNet, indicating that the model has statistical significance over the baseline. The results in the other two datasets are similar.

Table VI presents time complexity results. To accurately measure the time for training and testing, results in Table VI are measured with a single branch prediction in both methods. As shown in Table VI, it takes a longer time for our method to train and test. The result is expected, as there are more layers in our model, and the loss function is more complicated. We believe it is the tradeoff between complexity and performance of the model. Besides, it is noticeable that the test time is very close in both experiments, indicating that both models would be similarly fast if deployed.

Authorized licensed use limited to: Nanyang Technological University. Downloaded on December 18,2021 at 14:16:30 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

TABLE X
EXPERIMENTAL RESULTS WITH DIFFERENT LE APPROACHES, ON THE DATASET METR-LA

Experiments	15 min MAE	15 min RMSE	15 min MAPE	30 min MAE	30 min RMSE	30 min MAPE	60 min MAE	60 min RMSE	60 min MAPE
Mean-Variance Loss [40]	2.94	5.54	7.76%	3.56	6.75	10.01%	4.37	8.14	13.35%
KL divergence [34]	2.88	5.45	7.65%	3.46	6.67	9.78%	4.28	8.08	13.13%
Ours	**2.59**	**4.86**	**6.80%**	**2.97**	**5.79**	**8.18%**	**3.37**	**6.70**	**9.68%**

F. Ablation Studies

We also provide ablation studies on the performance of mean loss and residue loss, respectively, to evaluate their independent performance for traffic prediction. In these experiments, we set $\lambda_1 = 0$ that can remove mean loss from the proposed framework and set $\lambda_2 = 0$ that can remove residue loss from the neural network, and keep other configurations unchanged. We present the experimental results on the three datasets in Tables VII–IX, respectively, from which we can observe that either mean or residue loss can improve the performance significantly, while their combination, the mean–residue loss, can improve the performance further.

In Table VII, we present the experimental results for ablation study on the METR-LA dataset, at the 15-min interval. From these results, we can observe that without mean and residue loss functions, MAE of our baseline framework is 2.69. When mean loss is included in the framework, MAE can be improved to 2.62. When residue loss is included in the framework, MAE is also improved to 2.61. When we combine them, the proposed mean–residue loss can further reduce MAE to 2.59, which is 4% performance improvement. We carry out the same experiments at the intervals of 30 and 60 min on METR-LA dataset, and present the results in Tables VIII and IX, respectively. From Table VIII, we can observe that for the middle-term traffic prediction, 30-min interval, MAPE of Graph-WaveNet is 5.15, while that of our proposed methods with mean loss and residue loss only are 4.92 and 4.88, respectively. They are both better than the baseline, proving the effectiveness of mean and residue losses with regard to the metric of MAPE. Our proposed mean–residue loss can further improve MAPE to 4.86 that is 5.6% improvement over the baseline.

With regard to the metric of RMSE for long-term prediction, 60-min interval, we can observe that in Table IX, the performance of Graph-WaveNet is 10.01%, while that of approaches with mean loss and residue loss only are 9.88% and 9.72%, respectively. Their combination, our proposed mean–residue loss, can achieve 9.68%, the best one in the comparison.

It is worth noting that in Table VIII, at the interval of 30 min, RMSE of our approach with residue loss only is slightly better than our proposed mean–residue loss. As shown in Fig. 4, our gradient analysis has indicated that in the blue areas, mean loss and residue loss may fail to work jointly toward the optimum conditional distribution. In this case, mean loss has some negative effect on residue loss. Referring to Fig. 4 again, we can observe that the areas of positive joint effect are much bigger than the negative joint areas, which means the chance that the negative joint effect of mean–residue loss may dominate the performance is much lower than positive joint effect. However, it may happen since the probability distribution may not follow standard deviation strictly in the real implementations, although the negative effect as a whole is very small, and we should address this issue in our future work.

In Table X, model performance with our mean residue loss and other LE approaches is compared. The mean–variance loss is similar to the proposed loss function, except that the variance loss is replaced with the residue loss. Therefore, we keep the same label enhancement procedure as in our experiments, which is a narrow conditional distribution at the target class. The experiment settings are also the same as those in our experiments, except that we keep $\lambda_1 = 0.2$ and $\lambda_2 = 0.05$ for the mean and variance loss, respectively, following the original settings [40]. In the experiment with KL divergence, we generate the speed distribution from the ground truth using the softmax function. Next, we calculate the KL divergence between the logarithm of the predicted speed distribution and the generated speed distribution. A scaling factor of 0.05 is multiplied to KL divergence, and other experimental settings remain unchanged. It is evident in Table X that the proposed mean residue loss can better optimize the model than other LE approaches. Empirically, the proposed mean residue loss restricts the conditional distribution narrowly at the target class, which is a small range for the regression stage to optimize. Such distribution is specific to the traffic prediction task and is not addressed by other measurements, including mean–variance loss and KL divergence.

It is clear that the ablation studies have proven our theoretical findings in the work.

VI. CONCLUSION AND FUTURE WORK

In this work, we proposed a stagewise learning mechanism, including a conditional distribution learning followed by a regression learning. We introduce a novel mean–residue loss function for distribution learning and combine it with MAE for speed regression. The mean–residue loss consists of a mean loss and a residue loss, which effectively moves the conditional distribution towards the target and squeezes it around the center. The mean loss penalizes the difference between the mean of an estimated conditional distribution and the ground-truth speed and the residue loss penalizes residue errors in the tails that exist in an estimated conditional distribution after top-K pooling operation. We construct a graph-based architecture, which represents speed in discrete classes in the hidden layers, followed by a regression network. Our approach effectively learns the conditional distribution and produces more fine-grained prediction results in the output even with

Authorized licensed use limited to: Nanyang Technological University. Downloaded on December 18,2021 at 14:16:30 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

sharp spikes in the data. Extensive experiments are carried out and the results illustrate that the proposed architecture achieves state-of-the-art performance on three publicly available traffic datasets. For future work, we intend to study the conditional distribution learning in sequential anomaly detection [48] and weakly anomaly detection [49].

References

[1] Y. Zheng, L. Capra, O. Wolfson, and H. Yang, "Urban computing: Concepts, methodologies, and applications," *ACM Trans. Intell. Syst. Technol.*, vol. 5, no. 3, pp. 1–55, 2014.

[2] 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. (IJCAI)*, Stockholm, Sweden, Jul. 2018, pp. 3634–3640. [Online]. Available: https://doi.org/10.24963/ijcai.2018/505

[3] C. Chen *et al.*, "Gated residual recurrent graph neural networks for traffic prediction," in *Proc. AAAI Conf. Artif. Intell.*, Jul. 2019, pp. 485–492.

[4] C. Chen, K. Li, S. G. Teo, X. Zou, K. Li, and Z. Zeng, "Citywide traffic flow prediction based on multiple gated spatio-temporal convolutional neural networks," *ACM Trans. Knowl. Discov. Data*, vol. 14, pp. 1–23, Jul. 2020.

[5] Z. Zeng, N. Liang, X. Yang, and S. Hoi, "Multi-target deep neural networks: Theoretical analysis and implementation," *Neurocomputing*, vol. 273, pp. 634–642, Jan. 2018.

[6] Z. Zeng, Y. Xulei, Y. Qiyun, Y. Meng, and Z. Le, "$SeSe-Net$: Self-supervised deep learning for segmentation," *Pattern Recognit. Lett.*, vol. 128, pp. 23–29, Dec. 2019.

[7] Z. Zhao *et al.*, "An image-text consistency driven multimodal sentiment analysis approach for social media," *Inf. Process. Manage.*, vol. 56, no. 6, 2019, Art. no. 102097. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0306457319304546

[8] Y. Wu *et al.*, "Interactive multi-camera soccer video analysis system," in *Proc. 27th ACM Int. Conf. Multimedia*, 2019, pp. 1047–1049.

[9] 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. (IJCAI)*, Macau, China, Jul. 2019, pp. 1907–1913. [Online]. Available: https://doi.org/10.24963/ijcai.2019/264

[10] A. V. D. Oord *et al.*, "WaveNet: A generative model for raw audio," 2016, *arXiv:1609.03499*.

[11] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 4–24, Jan. 2021.

[12] H. Yao *et al.*, "Deep multi-view spatial-temporal network for taxi demand prediction," in *Proc. 32nd AAAI Conf. Artif. Intell.*, New Orleans, LA, USA, 2018, pp. 2588–2595.

[13] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Proc. AAAI Conf. Artif. Intell.*, New Orleans, LA, USA, 2018, pp. 7444–7452.

[14] Y. Seo, M. Defferrard, P. Vandergheynst, and X. Bresson, "Structured sequence modeling with graph convolutional recurrent networks," in *Neural Information Processing*, L. Cheng, A. C. S. Leung, and S. Ozawa, Eds. Siem Reap, Cambodia: Springer Int., 2018, pp. 362–373.

[15] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: Data-driven traffic forecasting," in *Proc. Int. Conf. Learn. Represent.*, Vancouver, BC, Canada, 2018, pp. 1–16. [Online]. Available: https://openreview.net/forum?id=SJiHXGWAZ

[16] Z. Zhang and M. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," in *Advances in Neural Information Processing Systems 31*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds. Red Hook, NY, USA: Curran Assoc., Inc., 2018, pp. 8778–8788. [Online]. Available: http://papers.nips.cc/paper/8094-generalized-cross-entropy-loss-for-training-deep-neural-networks-with-noisy-labels.pdf

[17] D. Rowlands, *Analog-to-Digital Conversion*. Cham, Switzerland: Springer, 04 2020.

[18] C. Zhang, C. Zhu, Y. Liu, H. Wen, and Z. Wang, "Image ordinal estimation: Classification and regression benefit each other," in *Proc. Asia–Pac. Signal Inf. Process. Assoc. Annu. Summit Conf. (APSIPA ASC)*, Kuala Lumpur, Malaysia, 2017, pp. 275–278.

[19] J. Zhang, X. Shi, J. Xie, H. Ma, I. King, and D.-Y. Yeung, "GaAN: Gated attention networks for learning on large and spatiotemporal graphs," 2018, *arXiv:1803.07294*.

[20] L. Ge, S. Li, Y. Wang, F. Chang, and K. Wu, "Global spatial-temporal graph convolutional network for urban traffic speed prediction," *Appl. Sci.*, vol. 10, no. 4, p. 1509, 2020.

[21] X. Zhang, K. W. Chan, H. Li, H. Wang, J. Qiu, and G. Wang, "Deep-learning-based probabilistic forecasting of electric vehicle charging load with a novel queuing model," *IEEE Trans. Cybern.*, vol. 51, no. 6, pp. 3157–3170, Jun. 2021.

[22] Z. Liu, Y. Liu, C. Lyu, and J. Ye, "Building personalized transportation model for online taxi-hailing demand prediction," *IEEE Trans. Cybern.*, vol. 51, no. 9, pp. 4602–4610, Sep. 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9133476

[23] Y. Zhang, L. Chu, Y. Ou, C. Guo, Y. Liu, and X. Tang, "A cyber-physical system-based velocity-profile prediction method and case study of application in plug-in hybrid electric vehicle," *IEEE Trans. Cybern.*, vol. 51, no. 1, pp. 40–51, Jan. 2021.

[24] S. Pan, R. Hu, G. Long, J. Jiang, L. Yao, and C. Zhang, "Adversarially regularized graph autoencoder for graph embedding," in *Proc. 27th Int. Joint Conf. Artif. Intell. (IJCAI)*, Stockholm, Sweden, Jul. 2018, pp. 2609–2615. [Online]. Available: https://doi.org/10.24963/ijcai.2018/362

[25] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. 5th Int. Conf. Learn. Represent.*, Toulon, France, 2017, pp. 1–14. [Online]. Available: https://openreview.net/forum?id=SJU4ayYgl

[26] C. Wang, S. Pan, G. Long, X. Zhu, and J. Jiang, "MGAE: Marginalized graph autoencoder for graph clustering," in *Proc. ACM Conf. Inf. Knowl. Manage. (CIKM)*, 2017, pp. 889–898.

[27] Z. Ying, J. You, C. Morris, X. Ren, W. Hamilton, and J. Leskovec, "Hierarchical graph representation learning with differentiable pooling," in *Advances in Neural Information Processing Systems 31*. Red Hook, NY, USA: Curran Assoc., Inc., 2018, pp. 4800–4810.

[28] M. Zhang and Y. Chen, "Link prediction based on graph neural networks," in *Advances in Neural Information Processing Systems 31*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds. Montreal, QC, Canada: Curran Assoc., Inc., 2018, pp. 5165–5175. [Online]. Available: http://papers.nips.cc/paper/7763-link-prediction-based-on-graph-neural-networks.pdf

[29] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in Neural Information Processing Systems*. Long Beach, CA, USA: Curran Assoc., Inc., 2017, pp. 1024–1034.

[30] F. Monti, D. Boscaini, J. Masci, E. Rodolá, J. Svoboda, and M. M. Bronstein, "Geometric deep learning on graphs and manifolds using mixture model CNNs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, Jul. 2017, pp. 5425–5434.

[31] P. Velič ković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *Proc. Int. Conf. Learn. Represent.*, Vancouver, BC, Canada, 2018, pp. 1–12. [Online]. Available: https://openreview.net/forum?id=rJXMpikCZ

[32] Z. Liu *et al.*, "GeniePath: Graph neural networks with adaptive receptive paths," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33. Honolulu, HI, USA, 2019, pp. 4424–4431.

[33] 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.*, Las Vegas, NV, USA, 2016, pp. 5308–5317.

[34] N. Xu, Y.-P. Liu, and X. Geng, "Label enhancement for label distribution learning," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 4, pp. 1632–1643, Apr. 2021.

[35] C. Tan, S. Chen, G. Ji, and X. Geng, "Multilabel distribution learning based on multioutput regression and manifold learning," *IEEE Trans. Cybern.*, early access, Oct. 23, 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9238428

[36] Y. Zhou, H. Xue, and X. Geng, "Emotion distribution recognition from facial expressions," in *Proc. 23rd ACM Int. Conf. Multimedia*, New York, NY, USA, 2015, pp. 1247–1250. [Online]. Available: https://doi.org/10.1145/2733373.2806328

[37] X. Geng and Y. Xia, "Head pose estimation based on multivariate label distribution," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Columbus, OH, USA, Jun. 2014, pp. 1837–1842.

[38] Z. He *et al.*, "Data-dependent label distribution learning for age estimation," *IEEE Trans. Image Process.*, vol. 26, pp. 3846–3858, 2017.

[39] X. Geng, C. Yin, and Z.-H. Zhou, "Facial age estimation by learning from label distributions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 10, pp. 2401–2412, Oct. 2013.

[40] H. Pan, H. Han, S. Shan, and X. Chen, "Mean-variance loss for deep age estimation from a face," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Salt Lake City, UT, USA, Jun. 2018, pp. 5285–5294.

Authorized licensed use limited to: Nanyang Technological University. Downloaded on December 18,2021 at 14:16:30 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

[41] R. Li, S. Wang, F. Zhu, and J. Huang, "Adaptive graph convolutional neural networks," in *Proc. AAAI Conf. Artif. Intell.*, vol. 32, 2018, pp. 3546–3553.

[42] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 770–778.

[43] G. Huang, Z. Liu, L. V. D. Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Honolulu, HI, USA, Jul. 2017, pp. 2261–2269.

[44] S. Shleifer, C. McCreery, and V. Chitters, "Incrementally improving graph WaveNet performance on traffic prediction," 2019, *arXiv:1912.07390*.

[45] Y. Wu *et al.*, "Dynamic Gaussian mixture based deep generative model for robust forecasting on sparse multivariate time series," in *Proc. 35th AAAI Conf. Adv. Artif. Intell. (AAAI)*, 2021, pp. 651–659. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/16145

[46] Y. Qin, D. Song, H. Cheng, W. Cheng, G. Jiang, and G. W. Cottrell, "A dual-stage attention-based recurrent neural network for time series prediction," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Melbourne, VIC, Australia, 2017, pp. 2627–2633.

[47] J. Hu and W. Zheng, "A deep learning model to effectively capture mutation information in multivariate time series prediction," *Knowl. Based Syst.*, vol. 203, Sep. 2020, Art. no. 106139.

[48] Y. Zhou, X. Song, Y. Zhang, F. Liu, C. Zhu, and L. Liu, "Feature encoding with autoencoders for weakly supervised anomaly detection," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jun. 25, 2021, doi: 10.1109/TNNLS.2021.3086137.

[49] J. Liu *et al.*, "Deep anomaly detection in packet payload," *Neurocomputing*, to be published. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231221016349

Zeng Zeng (Senior Member, IEEE) received the Ph.D. degree in electrical and computer engineering from the National University of Singapore, Singapore, in 2004.

He currently works as a Senior Scientist and the Program Head, Institute for Infocomm Research, Agency for Science, Technology and Research, Singapore. From 2011 to 2014, he worked as a Senior Research Fellow with the National University of Singapore. From 2005 to 2011, he worked as a Professor with the Computer and Communication School, Hunan University, Changsha, China. His research interests include distributed/parallel computing systems, data stream analysis, deep learning, multimedia storage systems, and wireless sensor networks.

Wei Zhao received the B.Eng degree in communication engineering from the Chengdu University of Information technology, Chengdu, China, in 2019. He is currently pursuing the master's degree with the College of Software Engineering, Sichuan University, Chengdu.

His research interests include data analysis and deep learning applications.

Peisheng Qian received the B.Sc. degree in computer science from the National University of Singapore, Singapore, in 2019.

His research interests include computer vision and deep learning.

Yingjie Zhou (Member, IEEE) received the Ph.D. degree from the School of Communication and Information Engineering, University of Electronic Science and Technology of China, Chengdu, China, in 2013.

He is currently an Assistant Professor with the College of Computer Science, Sichuan University, Chengdu. He was a Visiting Scholar with the Department of Electrical Engineering, Columbia University, New York, NY, USA. His current research interests include network management, behavioral data analysis, and resource allocation.

Ziyuan Zhao (Member, IEEE) received the Master of Technology degree from the National University of Singapore, Singapore, in 2019.

He is currently a Research Engineer with the Institute for Infocomm Research, Agency for Science, Technology and Research, Singapore. His current research interests include deep learning, multimedia analysis, medical imaging, and bioinformatics.

Cen Chen received the Ph.D. degree in computer science from Hunan University, Changsha, China, in 2019.

He currently works as a Scientist II with the Institute for Infocomm Research, Agency for Science, Technology and Research, Singapore. He has published several research articles in international conference and journals of machine learning algorithms and parallel computing, such as IEEE TRANSACTIONS ON COMPUTERS, IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, IEEE TRANSACTIONS ON CYBERNETICS, *ACM Transactions on Knowledge Discovery from Data*, AAAI, ICDM, ICPP, and many more. His research interests include parallel and distributed computing, machine learning, and deep learning.

Cuntai Guan (Fellow, IEEE) received the Ph.D. degree in electrical and electronic engineering from Southeast University, Nanjing, China, in 1993.

He is currently a Professor of Computer Science and Engineering and the Program Director, Strategic Collaboration, with the School of Computer Science and Engineering, Nanyang Technological University (NTU), Singapore. He is the Lead of the Artificial Intelligence Strategic Working Group, NTU. His research interests are in the fields of brain–computer interfaces, machine learning, neural signal and image processing, data analytics, and artificial intelligence.

Prof. Guan is a member of the Management Committee of the Singapore Health Technologies Consortium (HealthTEC), Singapore. He is on the Steering Committee of Ageing Research Institute for Society and Education, NTU.

Authorized licensed use limited to: Nanyang Technological University. Downloaded on December 18,2021 at 14:16:30 UTC from IEEE Xplore. Restrictions apply.