Residual Correction in Real-Time Traffic Forecasting

Daejin Kim∗
kiddj@kaist.ac.kr
KAIST AI
Republic of Korea

Youngin Cho∗
choyi0521@kaist.ac.kr
KAIST AI
Republic of Korea

Dongmin Kim
tommy.dm.kim@kaist.ac.kr
KAIST AI
Republic of Korea

Cheonbok Park
cbok.park@navercorp.com
NAVER Corp.
Republic of Korea

Jaegul Choo
jchoo@kaist.ac.kr
KAIST AI
Republic of Korea

ABSTRACT
Predicting traffic conditions is tremendously challenging since every road is highly dependent on each other, both spatially and temporally. Recently, to capture this spatial and temporal dependency, specially designed architectures such as graph convolutional networks and temporal convolutional networks have been introduced. While there has been remarkable progress in traffic forecasting, we found that deep-learning-based traffic forecasting models still fail in certain patterns, mainly in event situations (e.g., rapid speed drops). Although it is commonly accepted that these failures are due to unpredictable noise, we found that these failures can be corrected by considering previous failures. Specifically, we observe autocorrelated errors in these failures, which indicates that some predictable information remains. In this study, to capture the correlation of errors, we introduce ResCAL, a residual estimation module for traffic forecasting, as a widely applicable add-on module to existing traffic forecasting models. Our ResCAL calibrates the prediction of the existing models in real time by estimating future errors using previous errors and graph signals. Extensive experiments on METR-LA and PEMS-BAY demonstrate that our ResCAL can correctly capture the correlation of errors and correct the failures of various traffic forecasting models in event situations.

CCS CONCEPTS
• Computer systems organization → Real-time system architecture; • Computing methodologies → Neural networks: Supervised learning by regression; • Mathematics of computing → Time series analysis; • Information systems → Sensor networks.

KEYWORDS
Traffic forecasting, Spatial-temporal forecasting, Graph neural networks, Residual autocorrelation

∗Both authors contributed equally to this research.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

CIKM ’22, October 17–21, 2022, Atlanta, GA, USA
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9236-5/22/10...
https://doi.org/10.1145/3511808.3557432

1 INTRODUCTION
Despite its high practicality, traffic forecasting is a complex task since the speeds of all nodes are highly dominated by their historical signals as well as the conditions of the neighboring nodes. To handle spatial-temporal datasets, recent studies [17, 18, 21, 26, 27, 30] introduce deep-learning-based models in traffic forecasting and consider the graph structure in the training process. While these studies have made great progress in the traffic forecasting task,
little attention has been given to analyzing the errors in the traffic forecasting models.

In this work, we analyze the errors in the traffic forecasting models and observe that recent models still produce relatively large errors in certain patterns regardless of their high average performance. While these failures are considered unpredictable, we found that we can estimate how models will fail using current errors. In the real world, it is presumable that correlations among successive errors (i.e., failures) exist. However, previous errors have been ignored in the existing deep-learning-based traffic forecasting methods. Based on our findings, we highlight the necessity of utilizing previous errors in traffic forecasting. To explicitly handle the correlation of errors, we utilize historical errors of predictions, i.e., residuals, to make the next prediction. This can correct the failures of the traffic forecasting models and improve their performance in unexpected situations. Consequently, the critical mistakes that are crucial in a real-time setting can be minimized.

In addition to the previous predictions within a single sensor node, the previous predictions of the neighboring nodes are also highly correlated with the current prediction of each sensor node. To consider both spatial and temporal residual correlations, we propose a simple residual estimation module called ResCAL that estimates the expected residuals in the current prediction, i.e., how forecasting models will fail. Our method adopts the spatial-temporal layers conducted with a gated temporal convolutional network (Gated TCN) and a graph convolutional layer (GCN), as proposed in [26]. Furthermore, we analyze the patterns of failures since high errors occur in certain patterns. To this end, we introduce vector quantization in ResCAL and provide the justification of the calibrations. Vector quantization also allows our method to handle the noise and the outliers that appear in residuals.

Fig. 1 depicts how ResCAL corrects the failures in the real-time setting. Several attempts have been made to consider the residuals in graph classification [9, 13]; however, to the best of our knowledge, this study is the first attempt to capture the temporal correlation of residuals in traffic forecasting. In our experiments, we confirm that the residuals of each node are highly correlated with its previous residuals as well as that of neighboring nodes. Here, we introduce a simple synthetic dataset and validate the correctness of our ResCAL both qualitatively and quantitatively. Subsequently, we conducted extensive experiments in the two most representative traffic datasets: METR-LA and PEMS-BAY. In both datasets, we calibrated the predictions from various traffic forecasting models such as STGCN [27], DCRNN [17], Graph WaveNet [26], and STAWnet [21]. Specifically, we focus on whether our ResCAL accurately calibrates the predictions in the time areas where the existing models struggle. Despite its simplicity, our ResCAL consistently improves performance in event situations. Along with correcting failures, we validate the effectiveness of the vector quantization approach with a qualitative analysis. In this analysis, we verify that our ResCAL can provide meaningful justification for the calibration by examining the residual patterns in the unobserved data.

Our contributions can be summarized as follows:

- We propose a novel method called ResCAL as a widely-applicable add-on module, which estimates the future residuals and corrects the failures of models.
- Extensive experiments demonstrate that our ResCAL consistently improves the baselines in event situations while significantly reducing the correlation of residuals.

2 RELATED WORKS

2.1 Traffic Forecasting

Traffic forecasting is a challenging task due to the complicated spatial and temporal dependencies among the sensor nodes. To capture the dynamics of traffic conditions, data-driven approaches based on deep learning have received considerable attention. In spatial and temporal modeling, approaches based on graph convolutional networks (GCNs) [1, 16, 28] are promising in capturing spatial dependencies among roads. Several studies [5, 25, 29] have proposed applying recurrent neural networks (RNNs) and convolutional neural networks (CNNs) to capture the temporal dependencies along the sequence in traffic forecasting.

Recent studies [17, 18, 21, 26, 27, 30] have shown that graph modeling is a key factor to achieve state-of-the-art performance in traffic forecasting. Particularly, DCRNN [17] demonstrates its impressive performance against statistical approaches by incorporating diffusion graph convolutional neural networks [1] into RNNs. STGCN [27] utilizes only convolution-based approaches for modeling spatial and temporal dependencies in the road network. Graph WaveNet [26] introduces a self-adaptive adjacency matrix to overcome the limitation of applying fixed adjacency information and uses dilated convolutions [22] to efficiently handle long sequences. As another approach, STAWNet [21] applies a self-attention mechanism [24] to capture spatial dependencies between roads and uses self-learned node embedding to eliminate the need of prior knowledge of the graph structure. While these studies have shown remarkable progress in solving the traffic forecasting problem, methods to consider the errors of the forecasting models have been under-explored. In the line of traffic forecasting research, our work can improve the performance of the existing forecasting models by calibrating predictions of those models in real time.

2.2 Residual Correction

In statistical approaches, residual is defined as the difference between the ground truth values and the predicted ones. Statistical time-series models such as the autoregressive model and moving-average model represent future values as a linear combination of observed values and residuals at previous times steps. The autoregressive integrated moving average (ARIMA) [2] model considers residuals with autoregressive terms. However, these approaches suffer from expressing non-linearity because the residual term is expressed as a finite linear combination of a white-noise sequence.

Fitting residuals in the regression problem is an effective technique in machine learning. For example, a gradient boosting algorithm [6] and its variants, such as XGBoost [3] and LightGBM [14], recursively capture the residuals to improve the performance. Moreover, graph neural network architectures to model residual correlation have also been proposed. [13] suggested modeling of the joint distribution of the residuals to obtain information from both the
input feature and the output correlation in the graph structure. [9] proposes a procedure called “Correct and Smooth”, which models error correlation to correct the base prediction from simple architectures such as multi-layer perceptron (MLP). In contrast to these works, our ResCAL considers both spatial and temporal residuals to enhance the performance of traffic forecasting models.

2.3 Discrete Representations

Utilizing the discrete representation can provide interpretability and reduce the noise for a given data [4, 12, 23]. One of the discretization approaches is vector quantization [7], which is suggested as a system for mapping a signal into a digital sequence. VQ-VAE [23] utilizes this quantization mechanism to model the categorical distribution, and the latent variable is represented as the combination of the embedding vectors. The vector quantization layer formulates a latent space, called codebook, and clusters vectors according to a given distance metric (e.g., L2 distance). To learn a discrete representation by the backpropagation algorithm, [12] suggests Gumbel-Softmax which can approximate categorical samples by a differentiable sampling mechanism. In this work, we utilize discrete representation to enhance the interpretability of the calibration module and to reduce the noise introduced in traffic forecasting.

3 RESIDUAL DIAGNOSTICS IN TRAFFIC FORECASTING

High Errors in Traffic Forecasting. In the real world, a few critical errors have a huge influence on traffic conditions. Therefore, predicting such errors is essential for the traffic forecasting task. On a widely used traffic dataset, METR-LA [17], we observed that top-20% errors account for about 60% of the total absolute errors, e.g., 63.0% for STGCN and 62.8% for DCRNN. In traffic dynamics, these abnormal cases are mainly caused by sparse events. In this work, we specifically focus on time steps with top-20% errors and denote them as event situations since a large magnitude of error indicates the failure of the prediction.

Residual Autocorrelation in Traffic Forecasting. Autocorrelation is the correlation of a time series and its delayed copy. Given an input sequence \( y_1, y_2, \ldots, y_T \), the autocorrelation function (ACF) \( r_k \) measures the degree of the linear relationship between \( y_t \) and \( y_{t+k} \) where \( k \) is a time lag:

\[
    r_k = \frac{\sum_{t=k+1}^{T} (y_t - \bar{y})(y_{t-k} - \bar{y})}{\sum_{t=1}^{T} (y_t - \bar{y})^2}.
\]

High autocorrelation indicates a high potential of performance improvement; any forecasting model of which residuals are correlated or residuals have a non-zero mean can be improved by estimating the future residuals [10]. The residuals of the traffic forecasting model are commonly autocorrelated; therefore, we explicitly capture this relationship to further enhance the model performance.

Fig. 2 (b) shows the ACF of residuals for the traffic forecasting models. Before calibration, the residuals have a high autocorrelation, meaning that predictable information remains in the residuals. Fig. 2 (c) represents the Pearson Correlation between the current residual of each sensor node and the previous residuals of neighboring sensor nodes on METR-LA. The brighter points in the heatmap indicate a high correlation between these nodes, indicating that ResCAL can capture both temporal and spatial dependencies to correct the predictions.
The goal of traffic forecasting is to learn a mapping function \( X \rightarrow \hat{Y} \) that can be observed when time is greater than or equal to \( t + i \). For this, we consider a real-time traffic forecasting problem in which the current prediction of the model can be corrected a model-agnostic way.

\[
X^{(i)}(t - T_u + t) \rightarrow \hat{Y}^{(i)}(t - T_u + t)
\]

where \( T_u \) is the input time steps of the observed residuals. Given the estimated residuals \( \hat{R}^t \) for \( t \), we can learn \( g \) with a relatively lightweight model to estimate the residuals. This allows hyperparameter tuning with a small budget and increases the reusability of the model in real-world situations. (iii) Since information about \( f \) is not needed to predict the residuals, the performance of the base forecasting model can be improved in a model-agnostic way.

4 RESIDUAL CORRECTION IN TRAFFIC FORECASTING

In this section, we introduce a simple model-agnostic framework to boost the performance of traffic forecasting models in the real-time setting. To this end, we first formally describe the problem setting considered in traffic forecasting. Next, the model architecture and the residual prediction mechanism of our ResCAL are introduced.

4.1 Real-Time Traffic Forecasting

The aim of traffic forecasting is to predict the future traffic speed observed at \( N \) correlated sensors on the road network. Let \( \mathcal{G} = (V, E) \) be a graph representing the spatial relationship between sensors where \( V \) is a set of nodes (\(|V| = N\)) and \( E \) is a set of edges. Following the convention of the traffic prediction problem, \( X^t \in \mathbb{R}^{N \times 2} \) is denoted as the graph signal obtained at time \( t \). \( X^t \) and \( X^t \) represent the speed and the timestamp features, respectively. The goal of traffic forecasting is to learn a mapping function \( f \) from past graph signals and a graph \( \mathcal{G} \) to future traffic speeds:

\[
[X^{(t - T_u + t)}, \mathcal{G}] \xrightarrow{f} X^{(t + i)}(t + T_u),
\]

where \( T_u \) and \( T_y \) are the input and output sequence lengths, respectively. In particular, we consider a real-time traffic forecasting problem in which the current prediction of the model can be corrected using continuous historical predictions. Let \( Y^t = X^{(t + i)}(t + T_u) \in \mathbb{R}^{N \times T_y} \) be the ground truth speed so that \( Y^t \) represents the ground truth speed of \( i \) step ahead prediction at time \( t \), and \( \hat{Y}^t \) is its estimated values, i.e., \( Y^t = f(X^{(t - T_u + t)}, \mathcal{G}) \). Then, the residual of traffic prediction is defined as \( R^t = Y^t - \hat{Y}^t \). Note that the ground truth of \( R^t \) can be observed when time is greater than or equal to \( t + i \). Let \( U^t = \{U^{(t - T_u + t)}, U^{(t - T_u + t)}, \ldots, U^{(t - T_u + t)}\} \in \mathbb{R}^{N} \) be the newly observed residuals at time \( t \). Here, our main problem is learning a mapping function \( g \) which predicts the residual at time \( t \), given past graph signals, observed residuals, and a graph \( \mathcal{G} \):

\[
[X^{(t - T_u + t)}, U^{(t - T_u + t)}, \mathcal{G}] \xrightarrow{g} \hat{R}^t,
\]

where \( T_u \) is the time steps of the observed residuals. Given the estimated residuals \( \hat{R}^t \), accurate predictions can be made by taking \( \hat{Y}^t + \hat{R}^t \) as a final output. Fig. 3 illustrates our problem setting.

Our proposed problem setting in real-time traffic forecasting has several advantages, as follows: (i) The correlation between successive residuals can be explicitly modeled. We may consider resizing the window size of input to handle this correlation in the time series model. However, this can be tricky depending on the model design and is time-consuming to retrain due to its high complexity. (ii) As most parts of the complex modeling is done by \( g \), we can learn \( g \) with a relatively lightweight model to estimate the residuals. This allows hyperparameter tuning with a small budget and increases the reusability of the model in real-world situations. (iii) Since information about \( f \) is not needed to predict the residuals, the performance of the base forecasting model can be improved in a model-agnostic way.

4.2 ResCAL

To explicitly model the residuals, we designed a light-version of the traffic forecasting model called ResCAL consisting of spatial-temporal layers with the self-adjacency matrix, as proposed by [26]. The overall architecture of our ResCAL is depicted in Fig. 4. The encoder consists of \( K \) spatial-temporal layers, each conducted by a gated temporal convolutional layer (Gated TCN) and a graph convolutional layer (GCN). Following the conventional setting of the base traffic forecasting models, we let \( T = T_x = T_y \). The encoder produces a latent \( Z \in \mathbb{R}^{N \times d_h} \) by taking both a graph signal \( X \in \mathbb{R}^{N \times 2 \times T} \) and a residual \( U \in \mathbb{R}^{N \times T} \):

\[
Z = \text{Encoder} (\text{Concat} (X, U)),
\]

where \( \text{Concat} \) is a concatenation operation along the second axis and \( d_h \) is the number of hidden dimensions.

Here, we want to make a more accurate prediction by capturing useful information from \( Z \) and simultaneously provide a reason for the judgment. However, it is difficult to directly analyze \( Z \) since we do not put any strict restrictions on generating \( Z \). Moreover, the noise introduced by the baseline model or data further hinders the analysis of \( Z \). Previous works [12, 23] introduce discrete latent vectors into an autoencoder to increase the interpretability and stabilize training on the noisy data. However, we observe that using only these discrete forms sometimes reduces overall performance of the model. Instead, we use a hybrid approach of combining discrete and continuous representations. We take the sum of the outputs of two different branches from \( Z \): the regression branch and the quantization branch. The regression branch provides a
will show that the baselines fail similarly for similar events. Using 𝑛, where 𝑔, while the quantization branch provides interpretable and denoised 
dependencies of the residuals as our ResCAL does.

By predicting the residuals, we can easily calibrate the models 
other temporal dependencies of the residuals as our ResCAL does.

5 EXPERIMENTAL SETTINGS

5.1 Traffic Dataset

We examined our ResCAL on the two representative traffic datasets: 
PEMS-BAY and METR-LA introduced by [17]. PEMS-BAY, collected 
by California Transportation Agencies (CalTrans) Performance 
Measurement System (PeMS), contains 325 sensor data from the 
Bay Area. In our experiments, 6 months of data collected from Jan 
1st, 2017 to May 31st, 2017 was selected. METR-LA contains 
the traffic speed data recorded by 207 sensors on the highways of Los 
Angeles County [11]. The sensor locations of METR-LA and PEMS-
BAY are shown in Fig. 5. Both datasets were preprocessed following 
[17]. For METR-LA, 4 months of data collected from Mar 1st, 2012 to 
Jun 30th, 2012 was selected for our experiments. For both datasets, 
the traffic speed is aggregated into 5 minute windows, and Z-score 
normalization is applied to the input. As [19] proposed, adjacency 
matrix of both datasets was conducted using road distances with a 
threshold Gaussian kernel. A detailed description of each dataset is 
provided in Table 1.

| Data | # Nodes | # Edges | # Time steps |
|------|---------|---------|--------------|
| Synthetic | - | - | 10000 |
| METR-LA | 207 | 1515 | 34272 |
| PEMS-BAY | 325 | 2369 | 52116 |

Figure 5: Sensor locations of (a) METR-LA and (b) PEMS-BAY.
Figure 6: Calibrated predictions using ResCAL. (a) On the synthetic dataset, Seq2Seq shows similar failures for each event, and ResCAL correctly calibrates the predictions. (b) On METR-LA, ResCAL accurately corrects the failures of the baseline models.

5.2 Baseline Methods
To widely validate the correctness and effectiveness of our ResCAL, we examined several baselines commonly used for traffic forecasting. We additionally provide the reported results for the basic models such as ARIMA and FC-LSTM to help compare the performance of various models. In this work, we choose the mean squared error (MSE) for training the base models. For all baselines, the PyTorch implementation was utilized as denoted in their footnotes, and their default training strategies were followed.

- **ARIMA.** The auto-regressive integrated moving average model with a Kalman filter, which is the most representative regression model for time series data.
- **FC-LSTM [20].** Basic deep-learning-based regression model for time series data conducted with long short-term memory (LSTM) [8] and fully-connected layers.
- **DCRNN [17].** Diffusion convolutional recurrent neural network consisting of the graph convolutional networks and recurrent neural networks.
- **STGCN [27].** Spatial-temporal graph convolutional network which is conducted with the graph convolutional layer and 1D convolutional layers.
- **Graph WaveNet [26].** Traffic forecasting model which combines the dilated 1D convolutional layers and graph convolutional networks.
- **STAWnet [21].** Spatial-temporal attention network with temporal convolution and spatial attention mechanism to capture dynamic spatial dependencies.

| Time steps | Models       | MAE   | RMSE  | MAPE  |
|------------|--------------|-------|-------|-------|
| 1 step     | Seq2Seq      | 0.029 | 0.047 | 7.37% |
|            | + Calibration| 0.015 | 0.022 | 2.95% |
| 6 step     | Seq2Seq      | 0.061 | 0.153 | 20.34%|
|            | + Calibration| 0.025 | 0.093 | 10.29%|
| 12 step    | Seq2Seq      | 0.123 | 0.294 | 51.70%|
|            | + Calibration| 0.067 | 0.205 | 27.56%|
| 24 step    | Seq2Seq      | 0.174 | 0.367 | 80.64%|
|            | + Calibration| 0.122 | 0.291 | 53.47%|

5.3 Training Settings
For both METR-LA and PEMS-BAY, we used the same training strategy for the simplicity. \( k = 4 \) spatial-temporal layers were used for the encoder, and \( d_c = 32, n_c = 16, \) and \( d_c = 16 \) were set for the quantization branch in the decoder. Since the baselines predict the next \( T = 12 \) steps in units of 5 minutes and each step has errors of 12 horizons, the input residuals and predictions have a size of \( 12 \times 12 \) and \( 12 \times 1 \), respectively. Our ResCAL is trained with the mean absolute error (MAE) and a batch size of 256. An Adam optimizer with a learning rate of 0.001, \( \beta_1 = 0.9 \) and \( \beta_2 = 0.999 \) is also used. Each dataset is split into a training set, validation set, and test set with a ratio of 7:1:2 and the model with the best validation score is selected in all experimental evaluations.

6 EXPERIMENTAL RESULTS
6.1 Synthetic Dataset
To validate the correctness of our ResCAL, we first construct a simple synthetic dataset where similar events are occurring at random
Table 3: Quantitative results in event situations where the absolute error of the base forecasting model falls within the top 20%. The performances of the forecasting models were measured with and without our ResCAL on the METR-LA and PEMS-BAY datasets. The results are reproduced as mentioned in Section 5. Note that the high error region depends on the base forecasting model.

| Data      | Models          | 15min MAE | 15min RMSE | 15min MAPE | 30min MAE | 30min RMSE | 30min MAPE | 60min MAE | 60min RMSE | 60min MAPE | Average MAE | Average RMSE | Average MAPE |
|-----------|----------------|-----------|------------|------------|-----------|------------|------------|-----------|------------|------------|-------------|--------------|--------------|
|           | METR-LA        |           |            |            |           |            |            |           |            |            |             |              |              |
|           | DCRNN [17]     | 14.53     | 16.31      | 48.63%     | 16.98     | 18.93      | 60.79%     | 20.03     | 22.00      | 75.27%     | 17.18       | 19.08       | 61.56%       |
|           | + Calibration  | 13.46     | 15.44      | 44.22%     | 16.26     | 18.38      | 57.37%     | 19.45     | 21.54      | 73.13%     | 16.39       | 18.46       | 58.24%       |
|           | STGCN [27]     | 12.77     | 15.07      | 43.55%     | 15.79     | 18.18      | 57.50%     | 19.10     | 21.44      | 73.93%     | 15.88       | 18.23       | 56.33%       |
|           | + Calibration  | 13.28     | 14.99      | 41.45%     | 16.03     | 17.93      | 54.43%     | 18.94     | 20.85      | 68.34%     | 16.08       | 17.93       | 54.74%       |
|           | Graph WaveNet  | 13.36     | 15.30      | 43.75%     | 16.15     | 18.10      | 56.00%     | 19.00     | 21.00      | 68.44%     | 16.24       | 18.13       | 56.06%       |
|           | + Calibration  | 13.17     | 14.98      | 42.24%     | 15.87     | 17.89      | 54.88%     | 18.80     | 20.84      | 68.21%     | 15.95       | 17.90       | 55.11%       |
|           | STAWnet [21]   | 4.41      | 6.07       | 10.51%     | 5.95      | 8.45       | 15.36%     | 7.50      | 10.53      | 20.39%     | 5.95        | 8.35        | 15.42%       |
|           | + Calibration  | 4.25      | 5.93       | 10.07%     | 5.65      | 8.15       | 14.37%     | 6.94      | 10.00      | 18.60%     | 5.61        | 8.03        | 14.34%       |
| PEMS-BAY  | DCRNN [17]     | 6.42      | 7.95       | 15.88%     | 7.50      | 9.44       | 19.39%     | 8.74      | 11.03      | 23.50%     | 7.56        | 9.47        | 19.59%       |
|           | + Calibration  | 3.88      | 5.91       | 9.55%      | 5.41      | 8.00       | 14.36%     | 6.84      | 9.78       | 18.94%     | 5.38        | 7.90        | 14.29%       |
|           | STGCN [27]     | 4.36      | 5.89       | 10.45%     | 5.72      | 7.97       | 14.46%     | 6.89      | 9.50       | 18.07%     | 5.66        | 7.79        | 14.33%       |
|           | + Calibration  | 4.28      | 5.87       | 10.07%     | 5.63      | 7.93       | 14.21%     | 6.81      | 9.46       | 18.02%     | 5.57        | 7.75        | 14.10%       |
|           | Graph WaveNet  | 4.38      | 5.96       | 10.34%     | 5.71      | 7.91       | 14.51%     | 6.75      | 9.29       | 18.07%     | 5.61        | 7.72        | 14.31%       |
|           | + Calibration  | 4.27      | 5.87       | 10.07%     | 5.62      | 7.88       | 14.23%     | 6.70      | 9.27       | 17.77%     | 5.53        | 7.67        | 14.02%       |

Concretely, the synthetic dataset with the lengths of 10k time steps contains a periodic sine wave with a randomly generated zero signal, as depicted in Fig. 6 (a). The period of the sine wave is set to 50 steps and each period with zero values is randomly substituted with a probability of 0.1 to reflect the traffic dynamics. Similar to the traffic datasets, the synthetic dataset is divided into three parts: the training set, validation set, and test set with a ratio of 7:1:2. With our synthetic dataset, we examine the correctness of the following two assumptions essential for residual correction: (i) a deep-learning-based prediction model likely generates similar errors in similar types of events, and (ii) when the residuals are correlated, it is possible to improve the performance of the base prediction model by estimating the residuals that will occur, i.e., how the model fails.

For the base prediction model, we build a simple sequence-to-sequence model (Seq2seq). Seq2seq is designed to get an input sequence of length 24 and generate predictions on the next 24 steps. The encoder and decoder of Seq2seq are conducted with GRU units with a hidden feature size of 128 and a single recurrent layer. The output of the decoder is passed to a multi-layer perceptron (MLP) consisting of ReLU activations and fully connected layers of size 128–16–1. The model is trained using an Adam optimizer [15] with a learning rate of 0.001, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and a batch size of 100. The model is trained for 50 epochs and the Z-score normalization is applied to preprocess the input.

Fig. 6 (a) shows the predictions of the Seq2seq model, the estimated residuals from our ResCAL, and the calibrated prediction, respectively. As we expected in the first assumption, the prediction model always made similar errors for similar types of events. This implies that the residuals do not occur randomly and are also predictable. To assess the second assumption, we trained our ResCAL to predict the residuals occurring in Seq2seq. For the synthetic dataset without the graph structure, the spatial-temporal layers in the encoder were replaced with 1D convolutional layers. We set $d_c = 32$, $n_c = 16$, and $d_{enc} = 16$ for the quantization branch in the decoder. For training, an Adam optimizer with a learning rate of 0.001 and a batch size of 128 was used. Since the length of Seq2seq is $T = 24$ steps, our ResCAL gets 24 × 24 residuals with the original time series data for the input and outputs the 24 × 1 size of the residuals. Table 2 shows the results of Seq2seq with and without our ResCAL. Our ResCAL is shown to greatly enhance the performance of Seq2seq in every step of the predictions. This indicates that our ResCAL accurately predicts the residuals to occur. Notably, our ResCAL further improves the MAE of Seq2seq by 0.014, 0.036, 0.056, and 0.001 for 1 step, 6 steps, 12 steps, and 24 steps, respectively. Through a simple simulation, we validate that our assumptions and the proposed method are quite presumable in the time series data. The calibrated results in Fig. 6 (a) demonstrate that repeated errors can be estimated by our ResCAL. Moreover, the experiments on the synthetic data show that ResCAL allows the model to rapidly adapt to unexpected changes.

6.2 Traffic Dataset

In this section, we demonstrate that our ResCAL can correct the failure of the existing traffic forecasting models on the METR-LA
and PEMS-BAY datasets. Our ResCAL is trained with the settings as described in Section 5.

**Residual Correction in Event Situations.** To reflect the nature of the real-world setting, we examine our ResCAL in regions where critical errors occur. Table 3 shows the calibration results on time steps where the absolute error of the base forecasting model falls within the top 20%. Surprisingly, we can observe large gaps in performance before and after calibration. In the event situations on METR-LA, our ResCAL improves the performance of DCRNN by 0.79, 0.62 and 3.32% for MAE, RMSE, and MAPE, respectively. Even for STAWnet, the most advanced traffic forecasting model, our ResCAL shows an improvement of 0.29, 0.23 and 0.95% for MAE, RMSE, and MAPE on METR-LA, respectively. Consequently, our ResCAL can calibrate the prediction models more effectively in the cases of critical errors, which highlights the practicality of our ResCAL in real-time traffic forecasting.

As shown in Fig. 2, our ResCAL can accurately estimate the residuals and drastically reduce the autocorrelation of residuals both temporally and spatially. Fig. 6 (b) shows the qualitative results of prediction models with and without our proposed ResCAL on METR-LA. When the speed drops rapidly (i.e., an anomalous event occurs), STGCN and DCRNN cannot adapt to the changes; thus, they show poor prediction performance in the changed regions (orange lines). For the same case, our ResCAL successfully captures what those models tend to overestimate and corrects them accurately (green lines). Consequently, as a model-agnostic add-on module, our proposed ResCAL allows the prediction model to adapt to the fluctuation of data and consistently improves prediction performance regardless of the prediction model.

**Residual Correction in Overall Situations.** While our residual correction clearly shows its effectiveness in situations where residuals are correlated, we can also examine its performance when the existing models already correctly predict speeds. Here, we examine our method in overall situations for both METR-LA and PEMS-BAY to demonstrate its consistency. Table 4 shows that our ResCAL consistently improves the baselines on METR-LA and PEMS-BAY. This indicates that deep-learning-based models generate the correlated residuals even with their large number of parameters, and performance improvement in a high error region also leads to improvement in overall situations. For STGCN, our ResCAL achieves average improvements of 0.28 in MAE, 0.41 in RMSE, and 1.01% in MAPE on METR-LA. Even with the most recent model STAWnet, our ResCAL shows improvements for MAE, RMSE, and MAPE in both METR-LA and PEMS-BAY.

**Running Time Analysis.** The average computation time required for calibrating the outputs of the base models (e.g., Graph-WaveNet) was measured in real-time inference. It was tested on METR-LA and 12 sequences of Graph-WaveNet outputs were calibrated. A PC with an Intel Xeon Silver 4210R 2.40GHz CPU and a Titan RTX GPU was used in our analysis. On average, the calibration was performed within 9.05ms. That is, the running time of ResCAL is fast enough for practical usage. Note that the inference time of our ResCAL does not depend on the base model.
Table 5: Tuning analysis of ResCAL with Graph WaveNet.

| Models              | 15min MAE | 15min RMSE | 15min MAPE | 30min MAE | 30min RMSE | 30min MAPE | 60min MAE | 60min RMSE | 60min MAPE |
|--------------------|-----------|------------|------------|-----------|------------|------------|-----------|------------|------------|
| STGCN [27]         | 3.31      | 2.77       | 8.92%      | 3.74      | 4.65       | 10.19%     | 4.12      | 7.32       | 13.32%     |
| STGCN + Ours       | 3.41      | 2.87       | 9.09%      | 3.85      | 6.85       | 10.84%     | 4.40      | 7.95       | 12.94%     |
| Graph WaveNet [26] | 3.46      | 5.84       | 9.00%      | 3.90      | 6.75       | 10.56%     | 4.46      | 7.87       | 12.77%     |
| STGCN-LH           | 2.98      | 5.28       | 7.73%      | 3.47      | 6.26       | 9.46%      | 4.10      | 7.36       | 11.66%     |
| STGCN + Ours       | 2.88      | 5.10       | 7.30%      | 3.33      | 6.03       | 8.93%      | 3.91      | 7.61       | 10.84%     |

6.3 Pattern Analysis and Interpretability

Here, we show the qualitative results for the residuals assigned in the same categorical variables to examine the role of the quantization branch of ResCAL. Concretely, we extract the quantized vectors with the dimensions of the number of categorical variables $D = 32$ from the Gumbel-Softmax operation and assign the two residuals in the same pattern if both vectors have identical values for 32 categorical variables. Fig. 7 shows the prediction results of STGCN on METR-LA for the two different input sequences, each from the train dataset and test dataset. The quantized vectors for each pair in Fig. 7 (a), (b), and (c) belong to the same pattern. Notable implication comes from the fact that the residuals assigned to the same category show a similar pattern of the sequences. The results for the train dataset (top) and test dataset (bottom) share a similar ground truth with a similar strategy of calibration. That is, our ResCAL can provide the interpretable evidences for calibration. This is crucial for real-time traffic forecasting since the calibration of residuals is mostly needed in the case of unintended failures, and case studies are essential for maintenance of the system.

6.4 Tuning Analysis

In the tuning analysis, we show that the performance gain of ResCAL is mostly based on the consideration of residuals rather than on external factors (e.g., introducing additional parameters). As an add-on module, our ResCAL introduces additional parameters compared to solely using a baseline. In addition, ResCAL uses the residuals from previous predictions, thus observing a wide range of input sequences. For scrutiny, we conducted an experiment to check whether the performance gains can be obtained by simply increasing the model complexity or increasing the input length without using ResCAL. We introduced two variations of the baseline: one with larger parameters (LH) and one with longer input sequences (LI). Concretely, we extend the channels of the convolutional layers to increase the number of parameters as much as our ResCAL (~ 170k) for LH models and provide 2× longer input sequences for the LI models. Table 5 shows the results of the two variations and the original with our ResCAL on METR-LA. Interestingly, even with the larger parameters and wide ranges of usable data, the baselines cannot reproduce the results of our ResCAL. In Graph WaveNet-LH, introducing the larger parameters seems to improve the performance in terms of MAPE. However, the performance of MAE and RMSE significantly decreased, even compared to the original Graph WaveNet. In contrast, the model with our ResCAL shows a stable performance improvement in all cases.

7 CONCLUSION

In this work, we introduce a real-time setting in traffic forecasting and show that the residuals occurring from deep-learning-based models are highly correlated both spatially and temporally. To fully consider the autocorrelation of the residuals, we present a model-agnostic add-on module named ResCAL, which calibrates the predictions by estimating the residuals. Without having to fix the original architectures, which may entail high computational cost, our ResCAL module can effectively capture spatial-temporal dependencies of the residuals. On METR-LA and PEMS-BAY, our ResCAL consistently shows performance improvements for the existing forecasting models in event situations. Furthermore, we demonstrate the high practicality of our ResCAL in real-time traffic forecasting by providing the interpretability for calibration and effectively calibrating the predictions with significant errors. While we focus on traffic datasets, our proposed approach can be freely adopted to various tasks with autocorrelated residuals. We hope that our findings will shed light on future research for other forecasting problems as well as traffic forecasting.

ACKNOWLEDGMENTS

This work was supported by the Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korean government (MSIT) (No. 2019-0-00075, Artificial Intelligence Graduate School Program (KAIST)), and the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (No. NRF-2022R1A2B5B02001913). This work was also partially supported by NAVER Corp.
