CrowdFlowTransformer:Capturing Spatio-Temporal Dependence for Forecasting Human Mobility

Tomoki Choya
Graduate school of Engineering
Nagoya University
Aichi, Japan
choya@nuee.nagoya-u.ac.jp

Naoki Tamura
Graduate school of Engineering
Nagoya University
Aichi, Japan
tam@nuee.nagoya-u.ac.jp

Shin Katayama
Graduate school of Engineering
Nagoya University
Aichi, Japan
shinsan@nuee.nagoya-u.ac.jp

Kenta Urano
Graduate school of Engineering
Nagoya University
Aichi, Japan
urano@nagoya-u.jp

Takuro Yonezawa
Graduate school of Engineering
Nagoya University
Aichi, Japan
takuro@nagoya-u.jp

Nobuo Kawaguchi
Graduate school of Engineering
Nagoya University
Aichi, Japan
kawaguti@nagoya-u.jp

***Abstract*— Crowd flow forecasting is expected to have a wide range of applications such as human resource allocation, guidance design, marketing, disaster mitigation and congestion prediction for avoiding epidemic such as COVID-19. Crowd flow forecasting is challenging because it requires considering both the task of capturing the temporal dependency of data and capturing the spatial dependence. To address these challenges, in this paper, we propose a mechanism for referencing time-series features that are important for forecasting and incorporating graph convolution into Transformer, and we introduce CrowdFlowTransformer(CF-Transformer), a deep learning model based on Google's Transformer framework captures the Spatio-temporal dependency of time series. CF-Transformer captures the time series dependency by extracting important local time series from the past time series, inputting them to the decoder of Transformer, and encoding critical features into the model's input. We adapted CF-Transformer to a real-world crowd flow dataset. We evaluated it by comparing its forecasting accuracy with conventional models, and the results demonstrate that our model outperforms the conventional models.**

Index Terms*— crowd flow, Transformer, graph convolution, time series forecasting, human mobility

I. Introduction

Crowd flow forecasting is the prediction of the subsequent flow of people by analyzing various other information such as past crowd flow and event information. Accurate crowd flow forecasting has a wide range of applications such as human resource allocation, guidance design, marketing, disaster mitigation and congestion prediction to avoid epidemics such as COVID-19. It is one of the issues of urban computing, which aims to solve urban problems by computer science[9].

When we consider time series forecasting in crowd flow, we can divide the task into two parts.

(1) Task of capturing the temporal dynamics of changing crowd flow

(2) Task of capturing spatial dependence in a complex traffic network

For capturing temporal dependence of (1), long-term forecasting is difficult due to the non-stationarity of the time series caused by the presence of events such as the arrival of trains and the effects of weather. Fig.1 shows the actual traffic flow at two certain points of the airport terminal. The time series is non-stationary and the traffic flow changes varies from place to place. For capturing spatial dependency of (2), it is not easy because it requires modeling traffic conditions at various scales[12, 17]. For example, the volume of traffic in a certain corridor is affected by the volume of traffic upstream and downstream. The magnitude of this influence varies from corridor to corridor, and it is difficult to model the relationships that influence each other. Simple regression models such as ARIMA (Auto-Regressive Integrated Moving Average) or VAR (Vector Auto-Regressive) models are ineffective for forecasting capturing these complex dependencies because these models assume stationarity in the time series.

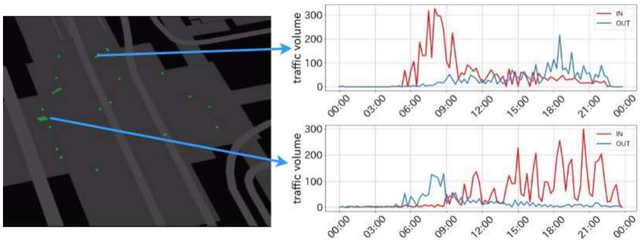

Figure 1: Actual Traffic Flow at Two Points.

As a deep learning method, recursive neural networks (RNNs) such as Gated Recurrent Unit (GRU) and Long Short Term Memory (LSTM) have been used to model the time dependency in (1)[15, 18]. Li et al.[9] and Lei et al.[7] proposed a method to model temporal dependence using GRU and spatial dependence using Graph Convolutional Networks (GCNs).In these RNN-based models, the time series are fed into the model sequentially, thus capturing the temporal sequential dependency. However, it is not possible to directly model the periodicity of a time series because different time steps in a time series are treated equally[3, 13]. This is a drawback since crowd flow has hourly, daily, weekly or seasonal periodicity.

On the other hand, Transformer, an encoder-decoder model, was proposed in 2017[14]. Transformer uses the multi-head attention mechanism and the positional encoding and does not require sequential input of series data and processes them in parallel. Thus, some research to forecast the time series using Transformer was conducted [8] since the positional encoding adds the order information of the time series to input. In addition, a hybrid model combining GCNs and Transformer was also proposed for traffic flow prediction [10, 16]. Ling et al. proposed Traffic Transformer [1], which uses a single neural layer as GCNs to capture spatial dependency, and concatenates two more intercepted time series segments, a daily component, and a weekly component, along the time axis as the input to model temporal dependency. Then, Traffic Transformer connects a local time series exactly one day or one week before the time to be predicted to the input. However, what is important for forecasting is that the crowd flow is similar to that at the time to be predicted, and it is not necessarily that there is such a similar crowd flow exactly one day or one week before. In addition, the input features of the model do not sufficiently take into account the features necessary for forecasting crowd flow, such as absolute time information and nearby event information.

In order to solve these problems, we incorporate the graph convolution into Transformer to capture the spatial dependence and take two approaches to capture the temporal dependence. There are crucial local time series for prediction in the data of approximately one day or one week ago since the crowd flow has a cycle of at least one day. Then, as the first approach to capture the temporal dependence, we present a mechanism to extract these crucial time series and input them into the decoder of Transformer. As the second approach to capture the temporal dependence, we combine the features involved in crowd flow prediction with the input data. We call the hybrid structure CF-Transformer (Crowd Flow Transformer), and CF-Transformer in an end-to-end model that learns spatio-temporal dependencies and captures abrupt changes. This approach enables us to take into account not only the information of the passage of people, but also any other information. We applied CF-Transformer to a real-world cloud flow dataset with large variations in traffic volume and evaluated it by comparing its forecasting accuracy with conventional models. Although the data was originally aggregated at the airport, our model can be adapted to other environments by changing the input features.

II. Related Work

Many deep learning models have been proposed for crowd flow forecasting. Many of the models are composed of RNNs, which compute time series by recursive processing and store the temporal dependence of the time series. RNNs capture only the forward dependence of the time series, but Cui et al. proposed a bidirectional LSTM model that captures the backward dependence[18].

However, although these RNN-based models capture the temporal dependency, the models do not capture the spatial dependency, including geographical information such as sensor locations. Zhang et al.[5] proposed a model that takes both temporal and spatial proximity into account for prediction. Similarly, hybrid models combined CNN or GNN with RNN-based models were also proposed to capture both temporal and spatial dependence. Wu et al.[17] used 1D convolutional neural networks to learn spatial dependency and LSTM to learn temporal dependency for traffic flow prediction. Li et al.[9] proposed DCRNN, which is an encoder-decoder model that combines GCNs with GRU to handle the graph structure in the model instead of CNN.

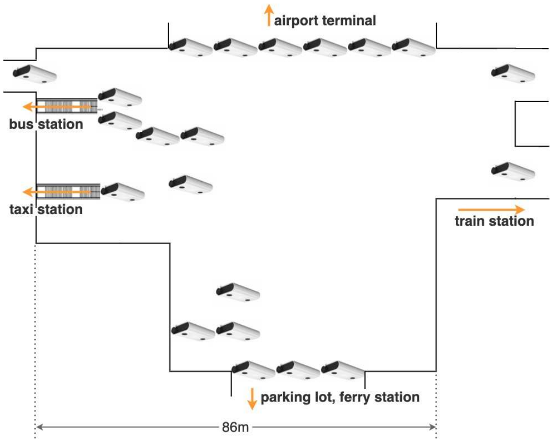

Figure2: Target environment (airport terminal)

To solve the problem that GCNs require a predefined adjacency matrix, Lei et al.[7] proposed AGCRN. ARCRN consists of GCN and GRU and is trained with the graph's adjacency matrix as a parameter. In other words, it is not necessary to predefine the adjacency matrix. However, the RNN models, which are based on sequential input and recursive processing, only capture sequential information of the time series and cannot directly model periodicity because it equally treats different time series time steps. Another drawback is that it is difficult to parallelize the processing, which makes learning and prediction inefficient, and long-term memory difficult[6].

On the other hand, Transformer has achieved significant success in text generation and natural language processing[2, 4]. Furthermore, several recent studies have used Transformer for time series forecasting due to the high versatility of the model. For traffic flow forecasting that captures both temporal and spatial dependencies, Xu et al.[10] proposed a model combining GCNs and Transformer. The model treats location embedding as a learnable parameter. Moreover, Ling et al.[1] proposed Traffic Transformer using GCN and Transformer, and also proposed a method for encoding the input. However, it only embeds temporal information and does not consider other significant information (e.g., train schedules, weather, presence of nearby events, etc.).

To summarize, with both spatial and temporal modeling, we built CF-Transformer. The model architecture of CF-Transformer is shown in Fig.4. CF-Transformer can capture Spatio-temporal dependencies among time series. We applied CF-Transformer to the original crowd flow dataset and evaluated it. We describe the environment in which the data was obtained in Section 3.

III. Problem Definition

We define the crowd flow forecasting problem in this study. In Section 3.A, we describe the environment to be forecasted. In Section 3.B, we define and formalize the forecasting problem.

A. Environment

We predict the crowd flow in a part of the international airport terminal in Aichi, Japan. Fig.2 shows the environment.

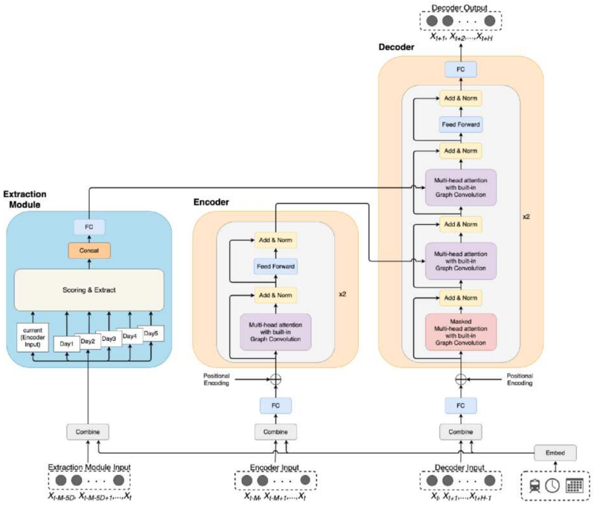
Decoder Output
$X_{t+1}, X_{t+2}, \dots, X_{t+H}$
Decoder
FC
Add & Norm
Feed Forward
Add & Norm
Multi-head attention with built-in Graph Convolution
Add & Norm
Multi-head attention with built-in Graph Convolution
Add & Norm
Masked Multi-head attention with built-in Graph Convolution
x2
Extraction Module
FC
Concat
Scoring & Extract
current (Encoder Input)
Day1
Day2
Day3
Day4
Day5
Encoder
Add & Norm
Feed Forward
Add & Norm
Multi-head attention with built-in Graph Convolution
x2
Positional Encoding
FC
Positional Encoding
FC
Combine
Combine
Combine
Embed
Extraction Module Input
$X_{t-M-5D}, X_{t-M-5D+1}, \dots, X_t$
Encoder Input
$X_{t-M}, X_{t-M+1}, \dots, X_t$
Decoder Input
$X_t, X_{t+1}, \dots, X_{t+H-1}$

Figure3: Overview of CF-Transformer

The space has corridors leading to the airport terminal, train station, bus station, taxi station, parking lot and ferry station.

Thus, a lot of people pass through this space. This environment is installed with 21 people counters and is the same environment used in the study of person flow estimation by Nagata et al[11]. The people counters are VC-3D made by Vitracom(R). The counters are installed on the ceiling at the position shown in Fig.2 and measure ingress (IN) and egress it took to pass through the sensor's measurement range. In this study, we use the information of the passing time and the passing direction to forecast crowd flow. In addition, there is a train platform adjacent to this environment. Thus, the traffic volume fluctuates greatly depending on the arrival and departure of trains. Therefore, we use the train timetable for forecasting. In another feature of these 3D people counters, passengers' privacy can be secured because images acquired by the sensors are not transmitted or stored outside the sensors. When sensing in a social environment, it is essential to secure the privacy of individuals. (OUT) to space. Moreover, the sensor also measures when the person passed through, the height of the person, and the time

B. *Forecasting Task*

The aim of crowd flow forecasting is to predict the future traffic volume using the past traffic information measured by the sensors. We solve a multivariate time series forecasting problem with 42 time series = 21 (number of sensors) × 2 (IN-OUT direction) since each sensors record two transit directions (IN-OUT). To model the spatial dependence, we represent the complex sensor network as a weighted undirected graph $G = (V, W)$ where V is a set of time series for each sensor's IN-OUT with $|V| = \mathrm{N} = 42$, and $W \in \mathbb{R}^{N \times N}$ the adjacency matrix that stores the stores the strength of the correlation between the time series. $X^t \in \mathbb{R}^{N \times P}$ denotes the feature matrix of the graph observed at time t, where P is the number of features. We formalize the forecasting problem as a learning of a mapping function from previously observed features matrices to future feature matrices on the premise of a network G;

$$X_{t+1}^{t+H} = F\left(G; X_{t-(M-1)}^{t}\right) \qquad (1)$$

where X_i^{i+n} denotes an array of feature matrices from time stamp i to $i + n$: $[X^i, X^{i+1}, \dots, X^{i+n}]$.

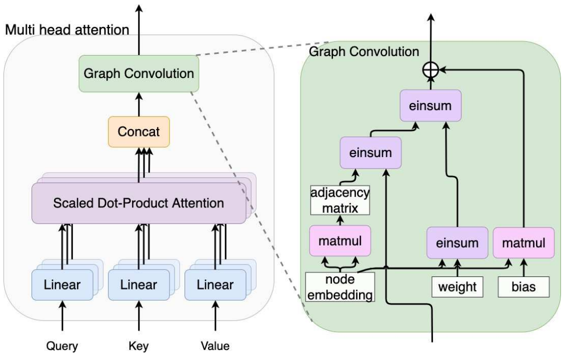

Figure 4: Multi-head attention with built in graph convolution.

IV. Methodology

In this section, we present the strategies for capturing spatio-temporal dependencies for crowd flow. In Section 4.A, we describe an overview of CF-Transformer(Fig.3). Next, in section 4.B, we describe the mechanism of time series forecasting using Transformer. And next, in Section 4.C, we describe the structure of the graph convolution used in CF-Transformer, and in Section 4.D, we present the extraction module outside Transformer. Finally, in Section 4.E, we describe the features of the input.

A. *CF-Transformer Architecture*

Transformer is organized in an encoder-decoder manner. The encoder encodes the source and captures the features of the source. The decoder decodes the target based on the encoded information. Then, we applied the structure to forecast the time series. The encoder encodes the features of the past time series and captures temporal dependency, and the decoder decodes the future time series based on the features of the past time series. Transformer has attention mechanisms that calculate which to focus on in the series. We replaced part of the linear transformation in the attention mechanism with the graph convolution. This replacement enables us to predict with capturing spatial dependency. In addition, CF-Transformer has an extraction module. The module goes through for local time series in the past that are similar to the current time series just before the prediction and extracts the subsequent time series for the input of the decoder. Finally, CF-Transformer has a mechanism to encode important features for prediction into the input to the model. This mechanism enables us to predict with rich information.

B. *Transformer-based time series forcasting*

Transformer is an encoder-decoder model upon attention mechanisms. Unlike RNNs, which are processed sequentially, Transformer can access any part of a time series regardless of its distance to the target. Thus, Transformer can directly model the periodicity of a time series and handle old information without collapsing. Therefore, Transformer is also useful in time series processing. The encoder consists of a multi-head self-attention layer and a position-wise feedforward layer, and the decoder has an encoder-decoder attention layer between the self-attention layer and the feedforward layer. In addition, CF-Transformer has an extraction-decoder attention layer with the same structure as the encoder-decoder attention layer.

The aim of the multi-head attention layers is to attach different importance of a time series to each other from multiple heads. The outputs of those different heads are then concatenated and linearly transformed to aggregate all the information. When a time series $x = (x1, x2, \ldots, xn)$ is input to the multi-head attention, the time series is updated using a weighted sum of the values at any time after being passed through a linear transformation. The weights are called attention scores and are assigned by their similarities. We formulate the attention mechanism:

$$y_i = \sum_{j=1}^{n} a_{ij}\left(x_j W_V\right) \quad (2)$$

where y_i is the updated x_i, and a_{ij} is the attention score, measuring the similarity between x_i and x_j, calculate as,

$$a_{ij} = \frac{\exp\left(e_{ij}\right)}{\sum_{k=1}^{n} \exp\left(e_{ik}\right)} \quad (3)$$

Where e_{ij} measures the compatibility of two linearly transformed x_i and x_j, calculated by using the scaled dot product,

$$e_{ij} = \frac{\left(x_i W_Q\right)\left(x_j W_K\right)^T}{\sqrt{h}} \quad (4)$$

where h is the dimension of the output, W_V, W_Q, $W_K$are three linear transformation matrices to strengthen the expressiveness of Transformer.

In the encoder, first, the input features are encoded, and a fully connected neural network is employed to strengthen the expressiveness of the model. Next, the result is passed through the multi-head self-attention to aggregate the traffic impact at other time-steps on that at time-step t. The multi-head attention in CF-Transformer also aggregates the spatial dependency over nearby nodes since CF-Transformer has the graph convolution in the multi-head attention. Finally, the result is passed through the feedforward layer. After the multi-head attention layer and the feedforward layer, an add and norm layer performs residual skip connection and normalization. The residual skip connection solves the vanishing gradient problem. As for the decoder, it has a similar structure except that Transformer decoder cell has one more encoder-decoder attention block, and another dense layer is leveraged before outputting the forecasts. The dense layer maps the output of the decoder cell to the traffic volume. In the decoder's self-attention mechanism, the decoder trains by masking the future values to avoid referencing future values. In the inference phase, the value at time-step t is input to the decoder, and the predicted value is used as the input at time t+1.

We introduce graph convolution to capture the spatial dependence of time series for crowd flow forecasting. The graph convolution operation can be well-approximated by 1st order Chebyshev polynomial expansion and generalized to high-dimensional graph convolution. With the graph convolution, the weight pool and bias pool are shared by all nodes. Because of this, only features common to all time series are learned without learning the features of individual time series. For example, it can learn common features such as the traffic volume increases during the daytime and decreases at night. We replace part of the multi-head attention mechanism with graph convolution. This structure allows attention with spatial dependency. Fig.4 shows an overview of the multi-head attention mechanism with built-in graph convolution.

C. Extraction Module

There is at least a one-day cycle in the crowd flow data since people act on a day unit. Thus, local time series similar to the time series to be forecasted exist one day or one week in advance, and data in that vicinity are effective for forecasting. However, inputting a week's worth of data into a deep learning model significantly increases the computation time and requires rich computational resources. Hence, in this study, we present the extraction module that goes through local time series in the past that are similar to the current time series just before the prediction, and extracts the subsequent time series. Extraction Module in Fig.3 shows an overview of this module. In this module, data for the past several days is input in addition to the time series data that is input to the encoder. When people intuitively predict time series, they may refer to time series that have shown similar fluctuation in the past. For example, if the traffic volume increased sharply at a particular time yesterday and the day before, they will probably predict that it will increase at that time today. Inputting the output of the extraction module into the decoder enables us to incorporate human intuition into deep learning explicitly.

D. Input Features

RNNs process the input sequentially, so the order relationship of the time series is guaranteed. On the other hand, Transformer handles the input time series in parallel, so the order relationship of the time series is not guaranteed. Then, Transformer adds relative positional information to the time series by applying the positional encoding strategy. In addition to the relative ordering relationship, absolute time is also important for predicting the crowd flow. For example, at 8:00 a.m., there is a rush to work, and traffic is heavy. Because of this, in this study, we combine time delay embedding, absolute time, day of the week, and train arrival/departure information into the time series, in addition to the positional encoding.

V. Evaluation

A. Dataset and Evaluation Metrics

We used the crowd flow dataset that is aggregated in the environment described in Section 3.A. The period of the data is weekdays from Jun.1, 2020, to Nov.18, 2020, and the portion of the data that was not collected due to unforeseen reasons was removed beforehand. The dataset is sorted by time in ascending order (from the past to the present) and is split into three parts for training (60%), validation (20%), and testing(20%). During training, the training data and the validation data were shuffled.

We used four metrics: MAE(Mean Absolute Error), RMSE(Root Mean Square Error), RRSE(Root Relative Squared Error), CORR(Correlation Coefficient), in order to evaluate and compare the performance of different models.

B. Evaluation Experiment

We compare proposed CF-Transformer with multiple baselines using the dataset and metrics described in Section 5.A. We used the following baseline models.

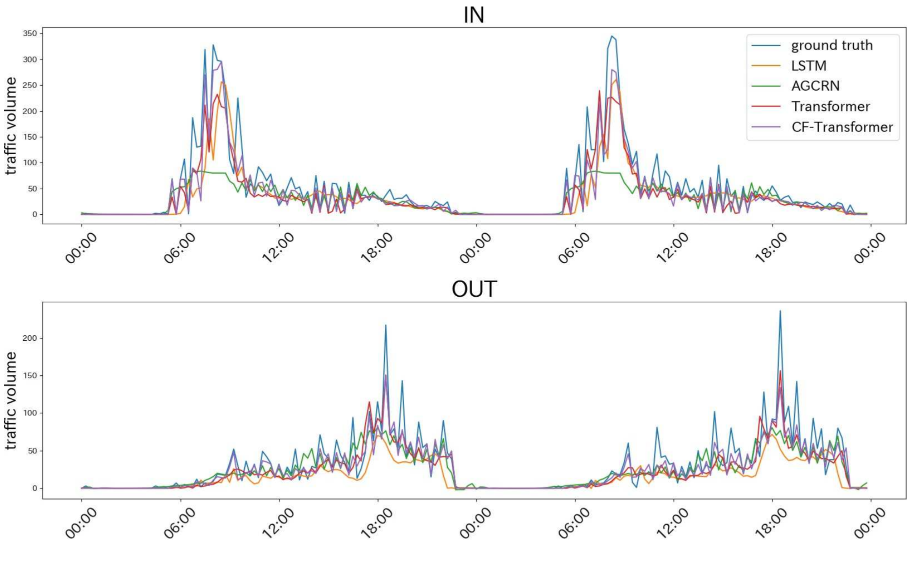

Figure 5: Example of prediction result.

- LSTM: Long Short-Term Memory, which is a typical recursive neural network model, and composed of two LSTM layers.
- AGCRN: Adaptive Graph Convolutional Recurrent Network, proposed by Lei et al. (2020). This model captures node- specific spatial and temporal correlations in time-series data automatically without a predefined graph.
- Transformer: proposed by Google Research (2017), which re- lies entirely on an attention mechanism to draw global dependencies between input and output.

Table I Parameters of a deep learning model

Parameters	Values
Input length to Encoder	8, 4, 2
Output length to Decoder	8, 4, 2
Multi Head	8
Epoch	1000
Node Embedding Dimension	20
Optimization	Adam
Adam beta1	0.9
Adam beta2	0.98
Chebyshev Order	2
Dropout	0.1
Coefficient of Penalty Term	0.001

We input 25 hours of data, which is longer than the cycle in a day, and predicted up to 2 hours ahead. Besides, we experimented with data from different sampling periods of 15, 30, and 60 minutes. We used the parameters shown in Table I to train the model, and added the L2 regularization of the learning parameters as a penalty term to the loss function in order to stabilize the learning.

C. Result

Table II shows the prediction performance of different methods in sampling periods of 15, 30, and 60 minutes. There are several discoveries from this result. First, it is clear that the prediction performance of CF-Transformer is much better than the other models on the crowd flow dataset in terms of all the evaluation metrics. This improvement shows that our presented architecture is effective in forecasting crowd flow, capturing both complex temporal and spatial dependencies. In addition, the RNN-based models of LSTM and AGCRN have larger prediction errors. This is because RNN-based models process sequentially and cannot directly model the periodicity of a time series. Furthermore, the performance of Transformer is better than that of such RNN models. This result shows that the encoder-decoder structure that treats time series equally by attention mechanism is effective for time series forecasting. Fig. 5 shows the example of prediction results of each method on the test data. The horizontal axis of the figure is the time, and the vertical axis is the number of people passing through. Overall, it can be seen that our model fits better and predicts more accurately. Under this sensor, it is difficult to predict the traffic volume because the volume varies greatly. Even in such a situation, our model is able to predict the traffic volume accurately. In particular, the difference is obvious at the large

Table II Evaluate Comparison

T	Metric	LSTM	AGCRN	Transformer	CF-Transforme
15min	MAE	6.21	5.86	4.85	**4.41**
	RMSE	17.35	16.82	13.68	**12.10**
	RRSE	0.90	0.94	0.62	**0.53**
	CORR	0.48	0.52	0.57	**0.58**
30min	MAE	8.44	9.40	7.91	**7.32**
	RMSE	23.50	25.56	21.66	**18.62**
	RRSE	0.61	0.67	0.52	**0.44**
	CORR	0.62	0.56	0.64	**0.65**
60min	MAE	15.16	16.73	13.01	**11.78**
	RMSE	40.31	50.59	33.52	**28.89**
	RRSE	0.56	0.72	0.42	**0.35**
	CORR	0.69	0.63	**0.72**	**0.72**

Table III Comparison of Each Presented Mechanism

Metric	Transformer	Graph Convolution	Extraction Module	Encoding Input	CF-Transformer
MAE	4.85	4.86	4.85	**4.57**	**4.41**
RMSE	13.68	13.78	13.78	**13.61**	**12.10**
RRSE	0.62	0.63	0.64	**0.58**	**0.53**
CORR	0.57	0.56	0.57	**0.58**	**0.58**

rise near 08:00 in the OUT direction, and only the proposed model predicts a large increase in traffic.

Table 3 shows the prediction performance of each of the proposed mechanisms in sampling periods of 15 minutes. Graph Convolution, Extraction Module, and Encoding Input in Table 3 are models that combine the mechanisms described in Sections 4.3, 4.4, and 4.5 with the Transformer. The results from Transformer and CF-Transformer are also included for comparison. The results show that the extraction module contributes significantly to the prediction. There was an approximate improvement of -5.8% in MAE, -4.3% in RMSE, -6.5% in RRSE, and +1.75% in CORR. However, on the other hand, there is no improvement in prediction performance by the graph convolution. To sum up, although the overall prediction error will be a little larger, it is important to use the graph convolution to capture spatial dependence in order to make abrupt changes in the traffic volume.

VI. Conclusion

In this paper, we presented CF-Transformer that captures both temporal and spatial dependencies. Specifically, we incorporate the graph convolution into the multi-head attention of Transformer to capture spatial dependency. In addition, we further presented an extraction module that extracts important local time series from past time series, and encodes the features necessary for time series forecasting of crowd flow into the input to capture the time dependency. When evaluated on the real-world crowd flow dataset, our approach obtained significantly better prediction than baselines. For future work, we will model spatial dependencies that change over time.

Acknowledgment

This research is partially supported by JST CREST and NICT B5G project.

References

[1] Ling Cai, Krzysztof Janowicz, Gengchen Mai, Bo Yan, and Zhu Rui. 2020. Traffic transformer: Capturing the continuity and periodicity of time series for traffic forecasting. Transactions in GIS 24, 03 (2020), 736–755. https://doi.org/10.1111/ tgis.12644

[2] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Salakhut- dinov Ruslan. 2019. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In Proceedings of the 57th Annual Meeting of the Associ- ation for Computational Linguistics. Association for Computational Linguistics, Florence, Italy, 2978–2988. https://doi.org/10.18653/v1/P19-1285

[3] Yao Huaxiu, Tang Xianfeng, Wei Hua, Zheng Guanjie, and Zhenhui Li. 2019. Revisiting Spatial-Temporal Similarity: A Deep Learning Framework for Traffic Prediction. Proceedings of the AAAI Conference on Artificial Intelligence 33, 5668– 5675.

[4] Devlin Jacob, Chang Ming-Wei, Lee Kenton, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:arXiv:1810.04805

[5] Zhang Junbo, Zheng Yu, Qi Dekang, Li Ruiyuan, and Xiuwen Yi. 2016. DNN- based prediction model for spatio-temporal data. Proceedings of the 24th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems 92, 1–4. https://doi.org/10.1145/2996913.2997016

[6] Urvashi Khandelwal, He He, Peng Qi, and Dan Jurafsky. 2018. Sharp Nearby, Fuzzy Far Away: How Neural Language Models Use Context. Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics 1, 284–294. https://doi.org/10.18653/v1/P18-1027

[7] Bai Lei, Yao Lina, Li Can, Wang Xianzhi, and Can Wang. 2020. Adaptive Graph Convolutional Recurrent Network for Traffic Forecasting. arXiv:2007.02842

[8] Shiyang Li, Xiaoyong Jin, Yao Xuan, Xiyou Zhou, Wenhu Chen, Yu-Xiang Wang, and Xifeng Yan. 2019. Enhancing the Locality and Breaking the Memory Bottle- neck of Transformer on Time Series Forecasting. In Advances in Neural Informa- tion Processing Systems, Vol. 32. Curran Associates, Inc.

[9] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. 2018. Diffusion convolutional recurrent neural network: Data driven traffic forecasting. In ICLR (2018).

[10] Xu Mingxing, Dai Wenrui, Liu Chunmiao, Gao Xing, Lin Weiyao, Qi Guo-Jun, and Hongkai Xiong. 2020. Spatial-Temporal Transformer Networks for Traffic Flow Forecasting. arXiv:arXiv:2001.02908

[11] Yoshiteru Nagata, Takuro Yonezawa, and Nobuo Kawaguchi. 2020. Person- Flow Estimation with Preserving Privacy using Multiple 3D People Counters. Science and Technologies for Smart Cities 2021 (Transaction of 5th EAI International Conference on IoT in Urban Space) (2020).

[12] Davis Neema, Raina Gaurav, and Jagannathan Krishna. 2020. Grids versus graphs: Partitioning space for improved taxi demand-supply forecasts. arXiv:1902.06515

[13] Guo Shengnan, Lin Youfang, Feng Ning, Song Chao, and Huaiyu Wan. 2019. Attention Based Spatial-Temporal Graph Convolutional Networks for Traffic Flow Forecasting. Proceedings of the AAAI Conference on Artificial Intelligence 33, 01, 922–929. https://doi.org/10.1609/aaai.v33i01.3301922

[14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In Advances in Neural Information Processing Systems, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[15] Jin Wenwei, Lin Youfang, Wu Zhihao, and Huaiyu Wan. 2018. Spatio-temporal recurrent convolutional networks for citywide short-term crowd flows prediction. In ICCDA (2018).

[16] Li Yang and José M. F., Moura. 2019. Forecaster: A Graph Transformer for Forecasting Spatial and Time-Dependent Data. in European Conference on Artificial Intelligence (ECAI), pp. 1293 - 1300, 2020. (2019). https://doi.org/10. 3233/FAIA200231 arXiv:arXiv:1909.04019

[17] Wu Yuankai, Tan Huachun, Qin Lingqiao, Ran Bin, and Jiang Zhuxi. 2018. A hybrid deep learning based traffic flow prediction method and its understanding. Transportation Research Part C: Emerging Technologies 90 (2018), 166–180.

[18] Cui Zhiyong, Ke Ruimin, Pu Ziyuan, and Yinhai Wang. 2018. Deep bidirectional and unidirectional LSTM recurrent neural network for network-wide traffic speed prediction. In UrbComp (2018)