Using Foundation Models MOMENT and OpenCity for Crowd Flow Prediction

Wan Rong Shen*
Manuel Pérez Belizón*
s2400804@vuw.leidenuniv.nl
m.d.perez.belizon@umail.leidenuniv.nl
Leiden University
Leiden, The Netherlands

ABSTRACT

In the era of big data, foundation models have emerged as one of the most relevant tools in the field of artificial intelligence by making good use of the vast amount of data available. Their great ability to generalize makes them ideal for urban tasks like crowd flow prediction. One of the biggest challenges of this task lies in its spatio-temporal nature. In this paper, we propose two different foundation models MOMENT and OpenCity can outperform one of the leading model in the field ST-ResNet, a model that outperforms most of the conventional methods. Experiments on two types of crowd flows in Beijing and New York City (NYC) demonstrate that both foundation models were able to outperform the ST-ResNet model.

KEYWORDS

Crowd Flow Prediction, Time-Series Foundation Model, MOMENT, OpenCity

ACM Reference Format:
Wan Rong Shen and Manuel Pérez Belizón. 2024. Using Foundation Models MOMENT and OpenCity for Crowd Flow Prediction. In . ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Forecasting crowd flow traffic in urban areas is essential to ensure public safety and optimize traffic management. However, traditional crowd flow prediction models are usually strongly dependent on training data. Although these models are effective within their task scope, they often lack the scalability to handle the dynamic nature of real-world crowd flow, such as adapting to weather changes or new public events.

In contrast, recently published time-series foundation models, such as MOMENT and OpenCity, have shown promising results in predicting general time-series tasks. These models were pre-trained on diverse time-series and spatial datasets, offer the potential for greater adaptability and generalization across a variety of scenarios. We aim to apply these models such as MOMENT and OpenCity to the crowd flow prediction task and to explore whether they can provide a more efficient and adaptable approach.

Our research focuses on the following key objectives:

- Evaluate MOMENT and OpenCity in crowd flow prediction against the traditional, domain-specific models like ST-ResNet.
- Investigate whether excluding external data like weather and holidays, time-series foundation models such as MOMENT and OpenCity can still achieve accurate predictions and provide a more adaptable solution.

2 RELATED WORK

Crowd Flow Prediction. This paper is motivated by the paper published by Zhang et al. [3], in this paper they introduced a novel model called ST-ResNet, a model that aims to do citywide crowd flow predictions.

ST-ResNet model employs convolution-based residual networks to model nearby and distant spatial dependencies between any two regions in a city.

ST-ResNet has 4 main components as shown in Figure 1, 1 taking into account external features such as weather or holidays, and the other 3 components model three different temporal properties: closeness, period and trend, respectively. The first three components share the same network structure with a convolutional neural network followed by a Residual Unit sequence. This structure captures the spatial dependency between nearby and distant regions.

Deep Learning Prediction Models. Of course there are many other models with different architectures for flow predictions. Recurrent Neural Networks, Temporal Convolutional Networks [5], and attention networks [7], have all proven to be able to make predictions in urban tasks such as crowd flow prediction.

Foundation Models. Foundation models are great because of their abilities to generalize and quickly adapt to diverse tasks, that is why more and more foundation models are emerging to solve diverse urban challenges [1, 6]. In this paper we will be working two novel foundation models MOMENT [2] and OpenCity [4].

*Both authors contributed equally to this research.

Conference'17, July 2017, Washington, DC, USA
© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/10.1145/nnnnnnn.nnnnnnn

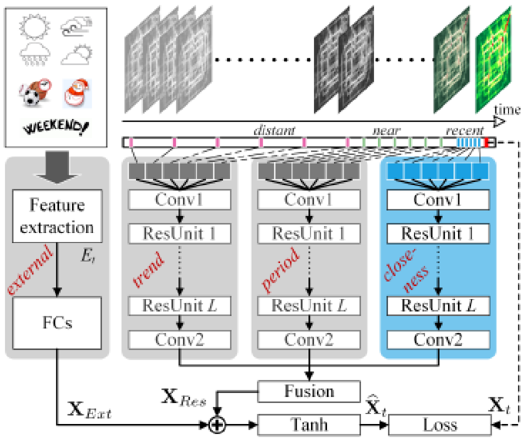

Figure 1: ST-ResNet model architecture

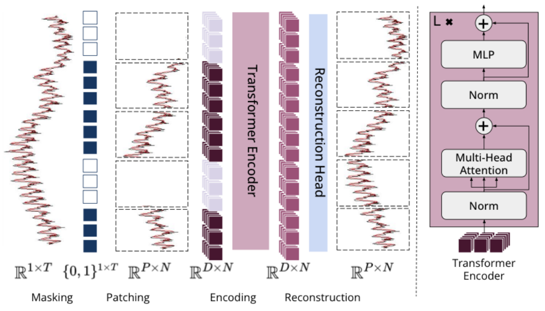

Figure 2: MOMENT model architecture

3 METHODS

In this section, we describe the models used in this study, explain how the foundation models are applied to predict crowd flow and the evaluation methods used to assess their performance.

3.1 Models

We focus on two foundation models, MOMENT and OpenCity, and use ST-ResNet for performance comparison.

3.1.1 MOMENT. MOMENT is an open-source foundational model designed for multi-purpose time series tasks, proposing by Goswami et al. (2024) [2]. It was pre-trained on a diverse collection of public time series datasets, such as the Informer long-horizon forecasting datasets, the Monash Time Series Forecasting Archive, the UCR/UEA Classification Archive, and the TSB-UAD Anomaly Benchmark. Thus, it is capable of handling a wide range of time-series tasks without requiring expensive fine-tuning.

MOMENT has four modes to handle different tasks: forecasting, classification, anomaly detection, and imputation. In this research, we only use the forecasting mode, which is designed for predicting future values based on historical time series data.

As shown in Figure 2, during pre-training, MOMENT first standardize the input time series data into a fixed length of $\mathbf{T} = 512$ time steps. A small portion of the input is then masked to enable the model to predict the missing values. MOMENT then breaks the time series into **N** fixed-**P**-length subsequences , called patches, and represents each patch with **D**-dimensional patch embeddings.

These patch embeddings are passed into the transformer encoder, as the self-attention mechanism helps to capture the temporal dependencies. The reconstruction head is then used to predict the masked portions of the time series.

For the forecasting task, while fine-tuning, the embedding and encoder layers remain frozen, and only the prediction head is updated. After the transformer encoder, the prediction head generates predictions for the next time steps in the sequence.

3.1.2 OpenCity. Opencity is a one for all foundation model meaning that can be used for many different tasks with few changes, it can effectively capture spatio-temporal dependencies from diverse data, with powerful zero-shot generalization across diverse urban environments.

OpenCity was pre-trained on large-scale datasets from a wide range of forecasting scenarios like speeding, trajectory, or flow predictions.

The architecture of the model can be divided into 3 main components:

- An Spatio-Temporal Embedding Layer that uses in-context normalization for zero-shot generalization and patch embedding for efficient long-term prediction.
- An ST Context Encoding layer to capture temporal and spatial context.
- An ST Dependencies Modeling layer to capture periodic and dynamic traffic patterns.

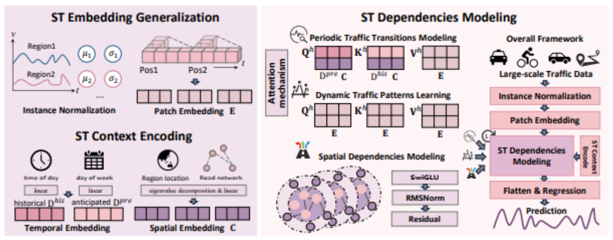

Figure 3: OpenCity model architecture

3.2 Data Preprocessing

To represent the inflows and outflows of crowd traffic, we use a grid-based representation, which we compare with the ST-ResNet research [3].

We divide the city into an $\mathbf{N} \times \mathbf{M}$ grid map based on its latitude and longitude, where each cell corresponds to an area in the city.

The inflow and outflow traffic for each region are then calculated by tracking the number of people entering or leaving each grid cell, as depicted in Figure 4. Each grid cell contains aggregated traffic counts within specific time intervals. We set the time frame for the time-series data to 30 minutes.

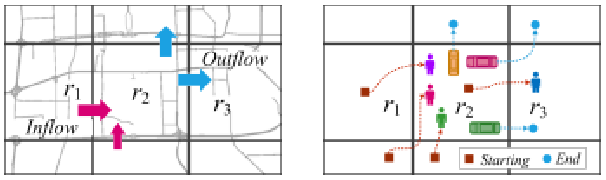

Figure 4: Grid-based flows representation

3.3 Model Configuration

3.3.1 MOMENT. We use the pre-trained MOMENT-large model with the following configurations:

- *task_name*: set this parameter to *'forecasting'* to enable forcasting mode.
- *context_length*: defines the number of timesteps used as model input and it was set to '512'.
- *forecast_horizon*: defines the number of future time steps the model will forecast and it was set to '480' for 10 days.
- *batch_size*: defines the number of sequences used in each batch.

For model input, Each $\mathbf{N} \times \mathbf{M}$ grid map data of a 30-minute-interval is flattened into $\mathbf{N} \times \mathbf{M}$ feature vectors. MOMENT has three parameters for the input (batch_size, n_channels, context_length). For example, for one of our experiments, we set the batch size to 4, used 576 channels (comes from 24 * 24 grids), and a 512 context length to predict a forecast horizon of 48 future timesteps.

3.3.2 OpenCity. OpenCity is a newer model compared to MOMENT. Although the model itself is complete, its usability is not as optimized. For instance, it is not well-adapted for use with Google Colab.

We use pretrained OpenCity model to avoid training the entire model from scratch. For the input data, we convert the data to a compressed numpy file `.npz` format and an adjacency matrix, which captures the spatial relationships between regions, in a `.npy` file.

Once the necessary files are in place, we execute the model by running the `Run.py` script. This script runs the model with the configurations specified in the `config` files. OpenCity uses `config` files to modify the model parameters or datasets, while also allowing for modifications via command line arguments. For example, the validation dataset ratio could be adjusted in the `config` file or in the command using the argument *–val_ratio*. Below we listed the important configuration files we adjusted:

- *global_baselines.conf*: In this file we control the batch size, validation and test set ratios, learning rate and early stop.
- *pretrain.conf*: In this file we control the dataset that the model will be using.
- *OpenCity.conf*: Here we control the input and output window, the embeddings dimensionality, and some other parameters we will cover in the fine tuning like the dropout ratio.

The most important configuration parameters used for OpenCity are:

- *input_window* and *output_window*: defines number of timesteps of either the input or the output sequence.
- *enc_depth*: defines the encoder depth.
- *batch_size*: defines the number of samples per batch.
- *val_ratio* and *test_ratio*: defines the proportion for the train validation test splits.

3.4 Evaluation metrics

In order to evaluate and compare the three different models, we will use RMSE and MAE as the main evaluation metrics, as this are the metrics used in the ST-ResNet paper [3]. In addition, we compare MOMENT and OpenCity using Mean Absolute Error (MAE) because it provides a measure of the prediction errors without amplifying the impact of outliers. In crowd prediction, large deviations can occur due to a lack of data points, but these outliers may not always be of primary concern when capturing the overall patterns of the crowd traffic.

4 EXPERIMENTAL SETUP

4.1 Hardware Settings

All experiments were conducted on Google Colab, using a NVIDIA Tesla T4 GPU with 16GB of RAM.

4.2 Dataset

We use a grid-map-based crowd flow dataset to train the models, similar to the one used by Zhang et al. [3], but excluding meteorological and holiday data such as temperature, wind speed, weather types, and holiday dates. The goal is to investigate whether the foundation models can still perform well without these location specific features.

NYC Shared Bike Trajectory Dataset, this dataset includes trip data for shared City Bike in New York City from April 2014 to 30th September 2014, with detailed information about each bike ride, including the start and end times, start and end station names and IDs, as well as the latitude and longitude of the stations. Additionally, the dataset has been processed to remove data for trips under 60 seconds (which are likely false starts).
The New York City was divided into 16x8 grid for spatial analysis. However, we also experimented with more refined grid sizes such as 24x24 to achieve better results.

The training data consists of all the data points from April to August (7344 timesteps), which is approximately 84% of the data, and the remaining 1440 timesteps from September are used as the test data.

For OpenCity we followed the suggestion made in the OpenCity paper [4] for the splits, having 20% training set, 20% validation set and 60% test set.

4.3 Evaluation Setups

4.3.1 MOMENT. The model uses the Adam (Adaptive Moment Estimation) optimizer with a learning rate of 0.0001, and is trained for either 1 or 10 epochs.
To address the issue of negative predictions, we increased epochs to 10 and experimented two types of loss functions:

- **MSE**: use the Mean Squared Error (MSE) as loss function, which is defined as:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2$$

  where Y_i represents the predicted values, $\hat{Y}_i$ is the true values, and n is the number of data points. MSE measures the average squared difference between the predicted and true values. The goal of this loss function is to minimize this difference, making the predictions as close as possible to the actual values.

- **MSE with Negative Penalty**: use a custom loss function that combines MSE with an additional penalty. This penalty is calculated by multiplying a penalty factor of 0.001 by the count of negative predictions. The penalty is then added to the MSE loss, resulting in the total loss:

$$Loss = MSE + 0.00001 \times \text{number of negative predictions}$$

  This loss encourages the model to minimize the number of negative predictions during training.

4.3.2 Opencity zero-shot. For OpenCity model we have two different training setups, a zero-shot and a fine-tune setup.

The zero-shot setup is directly used for testing without any training, while in the fine-tune setup the prediction head of the model(the last linear layer) is updated for a maximum of 3 training epochs.

Both setups utilize the Adam optimizer with a learning rate of 0.001, a batch size of 16, and an encoder depth of 6, achieving a balance between model complexity and computational efficiency.

For the fine-tune setup we set a dropout rate to 0.1 to prevent over-fitting and use the absolute error as the loss function that can be defined as:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|Y_i - \hat{Y}_i\right|$$

this loss function is very similar to the MSE loss function described above, the main difference between these two loss functions is that MAE measures the average absolute difference between predicted values and actual values while MSE measures the average squared difference between the predicted and the true value, this makes MSE more sensitive to outliers and MAE less sensitive to outliers.

To run each of the setups we used the following commands where we run the Run.py script:

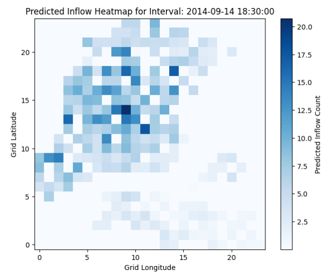

Figure 5: Predicted Inflow at 6:30 p.m. (1 epoch)

```
!python OpenCity/model/Run.py -mode test -model OpenCity
-load_pretrain_path OpenCity-plus.pth
  !python OpenCity/model/Run.py -mode eval -model OpenCity
-load_pretrain_path OpenCity-plus.pth -epochs 3
```

These commands do not include all the parameters as they were already mentioned above, the main difference between the commands is the mode parameter(*test* for zero-shot and *eval* for fine-tune) and the addition of the epochs parameter for the fine-tune.

5 RESULTS

The ST-ResNet has a Test RMSE of 6.33 and Test MSE of 40. We observed that for NYC Bike dataset both MOMENT and OpenCity effectively capture the crowd flow better than ST-ResNet as shown in Table 1 and 2.

Grid Size	Epochs	Negative Penalty incl.	Test RMSE	Test MAE
32x32	1	No	31.63	18.25
16x8	1	No	5.82	2.93
24x24	1	No	5.28	2.00
24x24	10	No	**1.62**	0.63
24x24	10	Yes	1.83	**0.62**

Table 1: MOMENT performance

5.1 MOMENT

From the result table (Table 1), we can see that MOMENT outperformed ST-ResNet by approximately 7.72% in RMSE even with just one training epoch, without requiring additional location, weather, or holiday data. Surprisingly, even without spatial features, MOMENT is still able to capture peak hour crowd patterns.

Also, increasing the number of epochs from 1 to 10 leads to a significant improvement in both RMSE and MAE. Comparing Figure 5 (Predicted Inflow Heatmap for 1 Epoch) with Figure 6 (Predicted Inflow Heatmap for 10 Epochs), we observe that after 10 epochs, the predicted crowd flow becomes more accurate and better aligns with the true values shown in Figure 7 (True Inflow Heatmap).

One challenge of the model is that it may predict negative values for crowd flow data, even though crowd flow should never be less than 0, as shown in Figure 8. However, after training for 10 epochs, the model demonstrates an improvement in learning this characteristic, especially in the peak inflow areas, as shown in Figure 9.

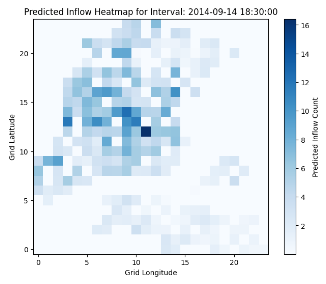

Figure 6: Predicted Inflow at 6:30 p.m. (10 epochs)

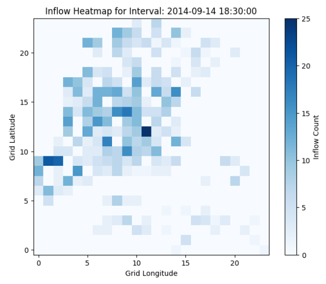

Figure 7: True Inflow at 6:30 p.m.

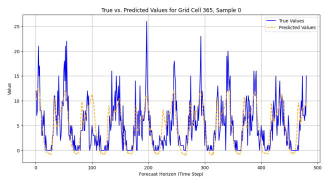

Figure 8: True and Predicted Inflow for High Activity Area (1 epoch)

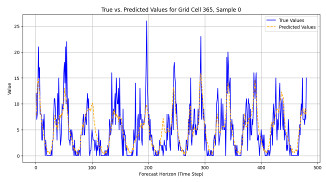

Figure 9: True and Predicted Inflow for High Activity Area (10 epochs)

The results indicate that 10 epochs are sufficient for the model to accurately predict crowd flow in the high-activity areas. Additionally, this training process is lightweight and efficient compared to models like ResNet.

Nevertheless, we observed that the model still has limitations in predicting low-activity areas and during off-peak times. For example, Figure 10 shows that the model still predicts negative values when there is a lack of data, such as during time intervals such as 2 a.m. as shown in Figure 11.

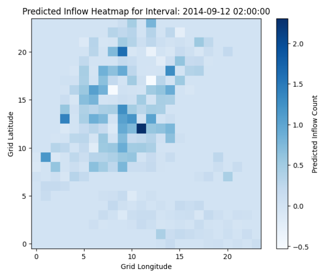

Figure 10: Predicted Inflow at 2 a.m. (10 epochs)

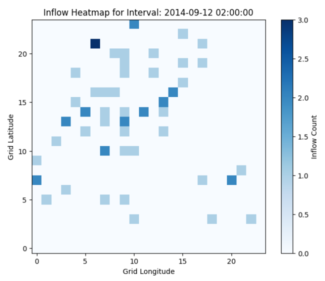

Figure 11: True Inflow at 2 a.m.

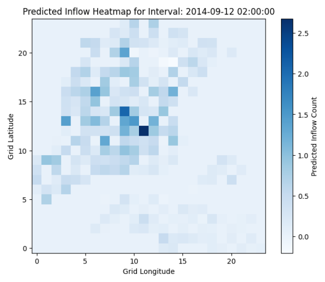

Figure 12: Predicted Inflow at 2 a.m. with Fine-Tuning (10 epochs)

After introducing a custom loss function with a penalty for negative predictions, this situation improved as shown in Figure 12. Yet, it is still unable to perfectly capture the real flow due to the limited availability of data points.

5.2 OpenCity

From Table 2 we can see that both setups outperform ST-ResNet MAE performance but not the RMSE performance. This could be because OpenCity was pre-trained using MAE instead of RMSE. The results are still really good if we take in consideration that the zero-shot setup did not train at all proving OpenCity great ability to generalize.

OpenCity seems to do worse predictions in general compared to MOMENT when its trained for 10 epochs as shown in Figure 13. But

Number of Regions	Epochs	Setup	Test RMSE	Test MAE
540	0	Zero-shot	11.61	6.33
540	1	Fine-tune	10.46	5.82
540	2	Fine-tune	8.28	4.20
540	3	Fine-tune	**6.45**	**3.10**

Table 2: OpenCity performance

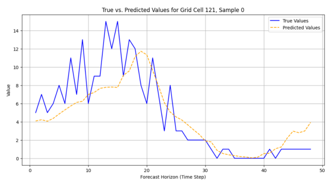

Figure 13: Predicted Inflow at 6 p.m. without training

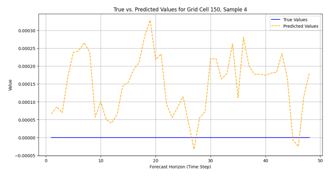

Figure 14: Predicted Inflow at 2 a.m. with training

we noticed that when trained OpenCity can do better low-activity areas predictions than MOMENT as shown in Figure 14

As it is shown in Table 2 it looks like the more epochs the better the performance for OpenCity but the computational complexity increases drastically making it impossible for us to reach more epochs with our resources, this also aligns with the experiments performed in the OpenCity paper [4] where they also only provided results for up to 3 epochs.

6 CONCLUSION

Our study demonstrates that MOMENT performs well in predicting crowd flow, especially in high-activity areas. MOMENT's performance significantly improves after increasing the number of training epochs to 10, even with minimal training and without the need for additional external data such as weather or holidays, the predictions closely aligned with the ground truth.

While MOMENT can handle peak inflow areas well, it does face challenges in predicting low-activity periods, particularly when data is sparse. To fill this gap, OpenCity shows better performance with sparse data by including spatial features, especially when it predicts values close to 0 in grids with no data.

In this study, we have shown how OpenCity excels at generalizing, particularly in the zero-shot setup. Without requiring any additional training, the zero-shot setup achieves impressive performance. A notable strength of OpenCity is its ability to accurately predict low-activity hours, a critical yet often challenging aspect of flow prediction tasks.

These results highlight the potential of OpenCity as a powerful foundation model for traffic-related tasks, reducing the need for extensive retraining while maintaining high predictive accuracy. This study underscores the value of leveraging pre-trained models like OpenCity to address real-world challenges efficiently.

In general, in this study we evaluated the performance of the foundation models OpenCity and MOMENT in crowd flow prediction tasks compared to traditional, domain-specific models such as ST-ResNet. The results highlight the significant advantages of foundation models, particularly MOMENT, which outperformed all other models by a substantial margin when fine-tuned for just 10 epochs, even without relying on external data like weather and holidays. These findings underscore the adaptability and scalability of time-series foundation models, which can provide accurate predictions while reducing dependency on domain-specific external features. This makes models like MOMENT and OpenCity more versatile and practical for diverse real-world applications, setting a new benchmark for crowd flow prediction tasks.

In future work, we will evaluate the performance of MOMENT and OpenCity on additional datasets, including those from diverse urban environments, to better understand their generalization capabilities.

REFERENCES

[1] Pasquale Balsebre, Weiming Huang, Gao Cong, and Yi Li. 2023. City Foundation Models for Learning General Purpose Representations from OpenStreetMap. *arXiv preprint arXiv:2310.00583* (2023).
[2] Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. 2024. MOMENT: A Family of Open Time-series Foundation Models. arXiv:2402.03885 [cs.LG]
[3] Dekang Qi Junbo Zhang, Yu Zheng. 2017. *Deep Spatio-Temporal Residual Networks for Citywide Crowd Flows Prediction.* https://arxiv.org/pdf/1610.00081
[4] Zhonghang Li, Long Xia, Lei Shi, Yong Xu, Dawei Yin, and Chao Huang. 2024. OpenCity: Open Spatio-Temporal Foundation Models for Traffic Prediction. arXiv:2408.10269 [cs.LG]
[5] Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. Graph WaveNet for Deep Spatial-Temporal Graph Modeling. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*. International Joint Conferences on Artificial Intelligence Organization, 1907–1913. https://doi.org/10.24963/ijcai.2019/264
[6] Congxi Xiao, Jingbo Zhou, Yixiong Xiao, Jizhou Huang, and Hui Xiong. 2024. ReFound: Crafting a Foundation Model for Urban Region Understanding upon Language and Visual Foundations. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (Barcelona, Spain) *(KDD '24)*. Association for Computing Machinery, New York, NY, USA, 3527–3538. https://doi.org/10.1145/3637528.3671992
[7] Chuanpan Zheng, Xiaoliang Fan, Cheng Wang, and Jianzhong Qi. 2020. GMAN: A Graph Multi-Attention Network for Traffic Prediction. *Proceedings of the AAAI Conference on Artificial Intelligence* 34, 01 (Apr. 2020), 1234–1241. https://doi.org/10.1609/aaai.v34i01.5477