YouBike Is All We Need

邵子軒
ID: R13250004

王本偉
ID: R13250007

張宥婷
ID: R13250012

翁德軒
ID: R13250020

1 Motivation

YouBike bike-sharing is a frequently used mode of transportation for Taipei residents, especially around National Taiwan University (NTU) campus. According to official statistics[1], 捷運公館站 (2 號出口) and 捷運公館站 (3 號出口) are the two stations with the highest number of rentals in Taipei. Frequent YouBike users around NTU have likely encountered two common situations: rushing to borrow a bike only to find none available, or arriving near the destination and discovering that nearby stations have no empty docks for returns. Since YouBike currently only provides real-time station availability, we believe that being able to anticipate—at a future time—the number of bikes available to rent or docks available to return at a specified station would help users plan their trips more effectively. It would also enable YouBike to better decide when to redistribute bikes and when to remove them.

2 Literature Review

Long Short-Term Memory Network The Long Short-Term Memory (LSTM) Network [1] is a tool for modeling time series data. With its two states and three gates structure, it learns long-term pattern of data. The hidden state, which is the valuable information of long-term pattern at the current time can be used to concatenate LSTM network and other neural network for prediction of time series data.

Encoder Decoder and Attention The Encoder-Decoder architecture is a model designed to address sequence-to-sequence problems, such as translation or multi-step time series forecasting. In the context of time series forecasting, it is standard practice to implement both the encoder and decoder using LSTMs. However, within a basic LSTM framework, the forecasting information for multiple future steps relies entirely on the final hidden state output of the encoder. This creates a bottleneck, making it difficult for the model to learn precise alignments between the input and output sequences. Hence, in the field of Neural Machine Translation (NMT), [2] first introduced the Encoder-Decoder with Attention, which was further developed by [3] with the proposal of Global and Local Attention. This concept has subsequently been widely applied to traffic flow and bike-sharing demand prediction. Consequently, our approach is primarily inspired by the Global Attention mechanism proposed in [3].

Graph Neural Networks For our YouBike data, Graph Neural Networks (GNNs) are also considered since bike stations are interdependent. However, standard GNNs do not explicitly handle temporal dynamics. Our task therefore falls under spatial-temporal forecasting. Under this setting, we first consider the Adaptive Graph Convolutional Recurrent Network (AGCRN) [4], which integrates a Data Adaptive Graph Generation (DAGG) module, a Node Adaptive Parameter Learning (NAPL) mechanism, and Gated Recurrent Units (GRU) to capture both spatial and temporal dependencies. AGCRN learns a unified node embedding matrix $E \in \mathbb{R}^{N\times d}$ shared across modules, and constructs an adaptive adjacency (or its normalized variant) via an embedding-similarity transformation,

$$A = \mathrm{softmax}(\mathrm{ReLU}(EE^\top)),$$

[1] `https://www.youbike.com.tw/region/taipei/operation/`

so that spatial correlations are inferred from data rather than relying on a pre-defined graph. In addition, NAPL generates node-specific parameters from a shared parameter pool conditioned on the node embeddings, enabling the model to capture fine-grained, node-dependent patterns. In the original paper, AGCRN is evaluated on real-world traffic datasets and compared with several representative spatial-temporal models, including Attention Based Spatial-Temporal Convolutional Networks (ASTGCN) [5], and it reports superior forecasting performance.

While AGCRN [4] represents a recent line of work that learns adaptive graph structure and node-specific parameters from node embeddings, it may not be well suited to our data because its learned adjacency matrix is symmetric, whereas the influence from station A to station B may differ from that of B to A. Therefore, we also consider ASTGCN [5], which introduces spatial and temporal attention mechanisms to learn dynamic dependencies and combines graph convolution with temporal convolution.

3 Model Description

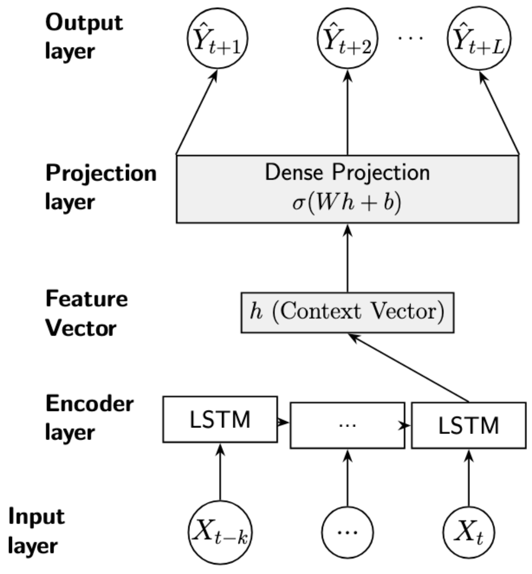
3.1 LSTM Based Networks

We used two LSTM based networks, LSTM and LSTM_FCNN. LSTM is a combination of LSTM and Fully Connected Network (FCNN). LSTM is the deeper version of LSTM. See Figure 1 for their network structures.
For LSTM, the size of hidden state vectors is 64, one hidden layer in FCNN layer and the number of neurons in its FCNN Layer is 64.
We use last hidden state in LSTM network as the input of the following FCNN. The output layer goes through a sigmoid activation function if we are predicting the percentage of rent-available bikes and a softmax activation if we are predicting whether a station is empty or full (Check Section 5.2 for details of the definition of empty and full)
The number of neurons in output layer is set to be the number of multiple steps prediction. For example, if we want to make predictions in the following 6 steps, the output layer is set to have 6 neurons. Each corresponds to the step of prediction in the following 6 steps.
LSTM_FCNN is the deeper version of LSTM. The size of hidden state vectors is 256, two hidden layers in FCNN layer and the number of neurons in its FCNN Layer is 256 and 128 in two layers.
The two networks are trained by "sliding window" technique. It uses the idea of stride in training of Convolution Neural Network (CNN). We use Mini-batch to train network, batch size is set as 64.
In each batch, at the time t, we input LSTM with the previous 12 steps (data in last 1 hour, our data is interpolated to have record every 5 minutes). For 6 steps prediction, we compare the value in output layer with the ground-truth in the next 6 steps.
After that, we move to the time $t + 1$ and repeat the above work flow.
Loss function is Mean Absolute Error (MAE) for the prediction of percentage of rent-available bikes and Mean Binary Cross Entropy for empty or full at a station.

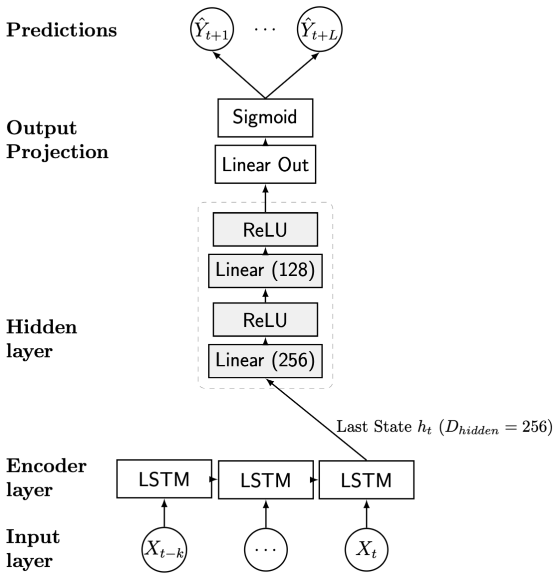
3.2 Encoder Decoder and Attention

Since multi-step forecasting combined with a look-ahead window can be fundamentally viewed as a sequence-to-sequence problem, the LSTM-based model mentioned in Section 3.1 relies solely on the hidden state h_t from the final time step to predict the subsequent six steps. However, it is anticipated that generating predictions for the next 6 steps may require accessing specific information from individual steps within the preceding 12-step observation window. Consequently, referencing [3], we propose the **Encoder(LSTM)-Decoder(LSTM) with Attention (LLA)** architecture, which incorporates a Global Attention mechanism and station-specific embeddings. The detailed model architecture is illustrated in Figure 2. Distinct from the approaches in Section 3.1 and the upcoming Section 3.3, LLA processes the past 12 time steps (containing 6 features) of a single station at a time, rather than using data from all stations simultaneously. Therefore, we utilize embeddings to differentiate between stations, enabling the model to discern the distinct bike usage patterns associated with each specific station.

The process operates as follows: First, each station is assigned a unique identifier which is transformed via a learnable embedding layer. Second, the temporal information from the input 12 time steps is processed by the Encoder LSTM to produce a final hidden state. This hidden state is replicated six times and fed into the Decoder LSTM to generate the initial latent representations for the next 6 steps. These decoder outputs are then processed via a global attention mechanism using additive attention to calculate alignment scores



(a) LSTM



(b) LSTM-FCNN

Figure 1: Diagrams of the LSTM based network structures.(a) LSTM+FCNN (b) Deeper version of (a)

and derive a weighted average (context vector). Finally, this context vector is concatenated with the original decoder output and passed through a fully connected dense layer to yield the final output predictions.

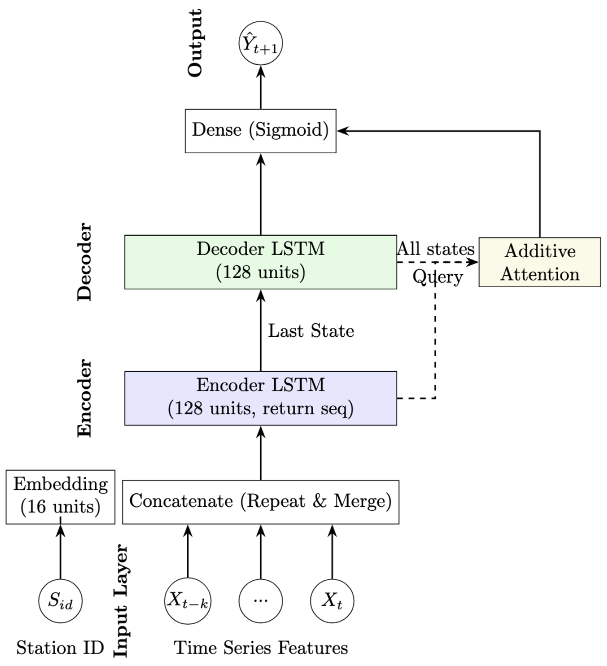


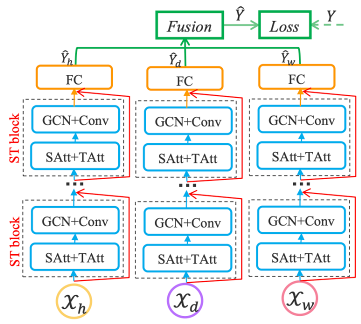
Figure 2: LLA structure

3.3 Attention Based Spatial-Temporal Dependence

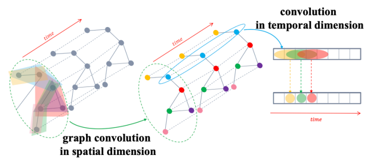
In our implementation, we found that setting the embedding dimension to 10 leads to a relatively large number of parameters in AGCRN, which slows down the decrease of training error and ultimately degrades forecasting performance. We also experimented with adjusting the learning rate and reducing the embedding dimension, but observed no noticeable improvement. Therefore, we adopt ASTGCN as the primary graph-based model in our study.

In ASTGCN, each spatial-temporal (ST) block contains a spatial attention (SAtt) module, a temporal attention (TAtt) module, and a spatial-temporal convolution module (see Figure 3a). SAtt estimates node-to-node importance by aggregating information over the input time window, while TAtt estimates temporal importance by aggregating information over the spatial dimension; neither module explicitly models the other dependency during attention estimation. The subsequent spatial-temporal convolution then captures local spatial and temporal patterns via graph convolution over neighbors and temporal convolution over nearby timestamps (see Figure 3b), and passes the output to the next ST block.

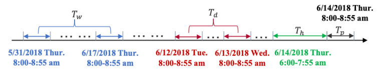
In the original ASTGCN framework, three parallel components with the same block design are used to model different temporal ranges of historical observations, namely recent ($\mathcal{X}_h$), daily-periodic ($\mathcal{X}_d$), and weekly-periodic ($\mathcal{X}_w$) patterns (see Figure 3c), and their outputs are fused for the final prediction. In our study, we only use the recent component with a lookback window of 12 time steps, which corresponds to one hour in our data setting. Therefore, our model does not include the multi-component fusion module; instead, the output from the last ST block is directly connected to the prediction head and optimized by the loss function.



(a) ASTGCN structure


convolution in temporal dimension
graph convolution in spatial dimension
time

(b) Spatial-Temporal Convolution


T_w
T_d
T_h
T_p
6/14/2018 Thur. 8:00-8:55 am
5/31/2018 Thur. 8:00-8:55 am
6/17/2018 Thur. 8:00-8:55 am
6/12/2018 Tue. 8:00-8:55 am
6/13/2018 Wed. 8:00-8:55 am
6/14/2018 Thur. 6:00-7:55 am

(c) An example of the input segments

Figure 3: Detail of ASTGCN

4 Data Analysis and Feature Input

The input features are percentage of rent-available bikes, precipitation, index of weekday (what day is the record of data, e.g Monday, Tuesday, etc.) and the timestamp (our data is of 5-minutes level, so we have 288 data for a day with raw timestamp of $1\tilde{2}88$. The timestamp used in our analysis is the sine and cosine transformed value of the raw timestamp)
For all models used in our analysis, we aggregate the aforementioned features into a vector to serve as the model input. However, a key distinction is that the LLA model processes input on a station-by-station basis, whereas the LSTM and ASTGCN models ingest information from all stations simultaneously.

5 Results

5.1 Future Bike Prediction

From Table 1, we first compare the first two models. The main differences are that the second model uses two fully connected layers instead of one, and the LSTM hidden size is increased from 64 to 256. As a result, LSTM-FCNN has roughly ten times more parameters than the baseline LSTM, yet exhibits a higher prediction error. This suggests that the substantially larger parameter count makes the model more complex and flexible, but also harder to optimize and converge under the same amount of training data.

Our third model, LLA, incorporates an encoder–decoder LSTM architecture with an attention mechanism. Despite having only about one-third of the parameters of the baseline LSTM, it achieves a much lower error, indicating that the architectural inductive bias is more effective than simply increasing model capacity. Finally, ASTGCN models not only temporal dependence but also spatial dependence among stations. Overall, it delivers the best prediction performance among the compared models, and we therefore further analyze its forecasting results in the next section.

Table 1: One-step and multi-step prediction error for different models

Model	**1-step prediction**		**Multi-step prediction**	
	#Params	MAE	#Params	MAE
LSTM	51,029	2.3212	87,054	2.7116
LSTM-FCNN	608,365	2.5407	678,670	3.2491
LLA	214,625	0.5477	214,625	**1.1459**
ASTGCN	112,835	**0.3370**	116,680	1.2338

Figure 4 shows the 1-step ahead forecast for 捷運公館站 (3 號出口). As we can see, our model performs as well as normal time series model. Notably, our model performs comparably to standard time-series baselines and predicts most points accurately. However, when the ground truth exhibits a sharp change (e.g., around time step 80), the model reacts with a one-step delay and tends to overshoot the magnitude. This suggests that it does not fully capture temporal uncertainty, a limitation that is also evident in the multi-step forecasting results.

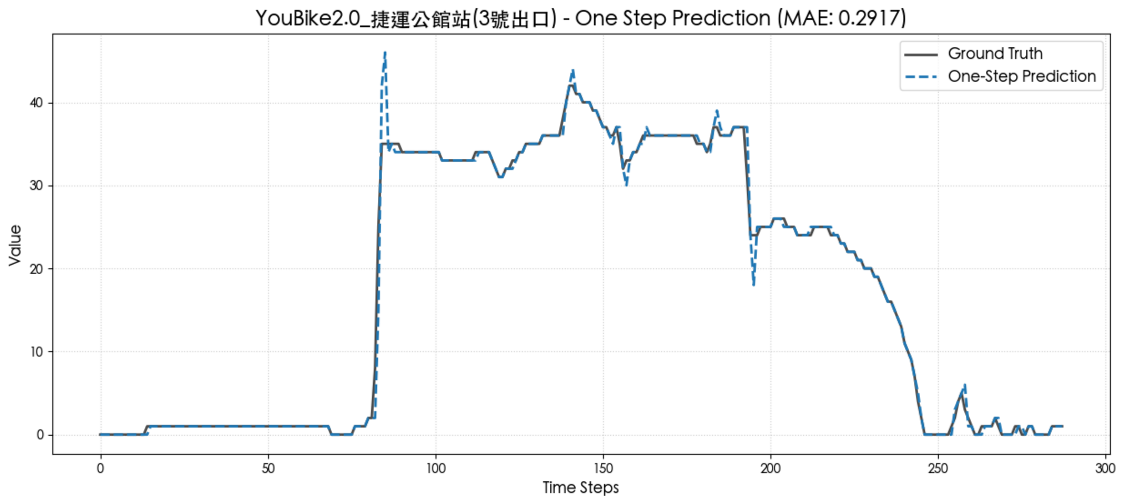


Figure 4: One-step ahead prediction

Figure 5 shows different results of multi-step ahead prediction. The top two plots show that the model tracks the trend well when the series is either perfectly flat (often when no bikes are available) or exhibits a steady upward pattern. However, in the lower-left plot, where the ground truth contains an abrupt V-shaped turning point, the model fails to capture such rapid dynamics and produces a more conservative forecast. In contrast, the lower-right plot shows the opposite behavior: although the ground truth remains stable, the model incorrectly predicts a downward trend. These errors likely reflect the strong randomness in our data. While daily patterns are broadly similar, sudden fluctuations vary from day to day, possibly due to inconsistent bike rebalancing times and variations in users' schedules. As a result, the series does not exhibit a stable periodicity, making it difficult for the model to learn and predict these dynamic structures reliably.

Since ASTGCN follows a GNN-based design, we can extract its learned adjacency (attention) matrix and visualize it as a heatmap. Because the matrix is learned with attention and equipped with a sigmoid function, it is asymmetric with entries in [0,1], and can be interpreted as a directed weight matrix. The heatmap shows that most large weights concentrate near the diagonal, indicating that self-dependence is stronger than cross-station influence.

For 捷運公館站 (3 號出口), the top five related locations (by weight) are 捷運公館站 (2 號出口), 捷運科技大樓站, 和平復興路口西北側, 臺大農業陳列館北側, and 臺大小小福西南側. Among them, 捷運公館站 (2

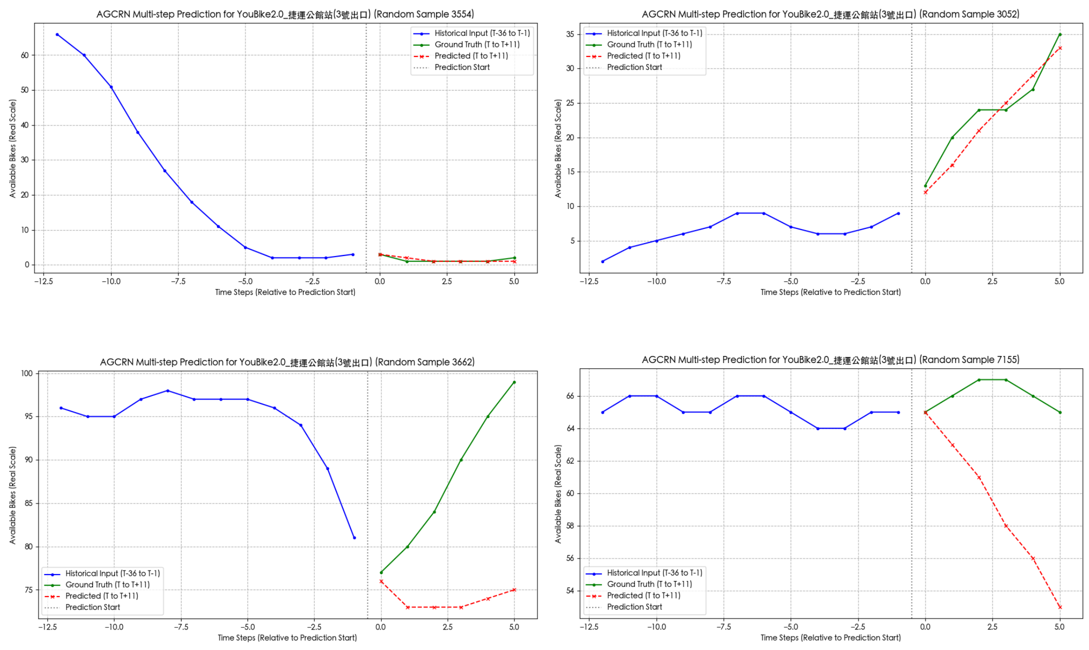


Figure 5: Different result for multi-step-ahead prediction

號出口) dominates, which is expected given its proximity. Interestingly, 捷運科技大樓站 also shows a strong influence despite not being nearby, possibly because both are metro exits with similar commuting patterns.

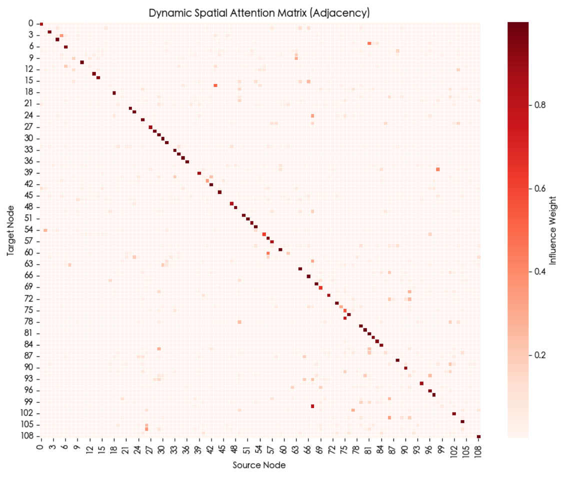


Figure 6: Heatmap of the adjacency matrix learned by the model

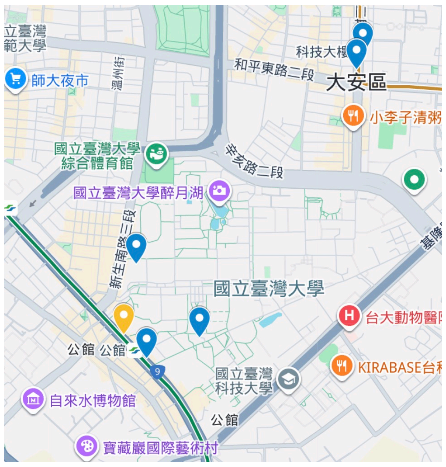


Figure 7: Top five locations with the largest weights

5.2 Classification

Because the YouBike data do not exhibit a stable periodic pattern and are influenced by substantial human-driven factors, we found that the multi-step forecasts produced by the aforementioned models still deviate noticeably from the ground truth. Therefore, we revisited what problem we truly aim to solve. Rather than precisely predicting how many bikes are available to rent or how many docks are available for returns at a given station, we may care more about when a station will run out of bikes to rent or run out of docks for returns. Accordingly, we redefine two variables as follows:

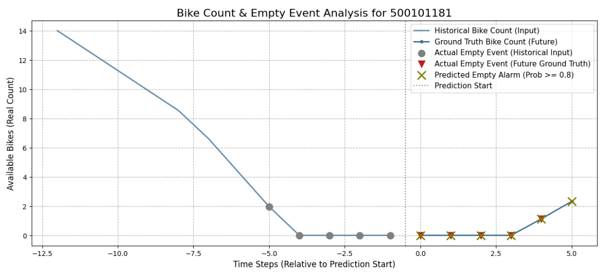
$$\text{empty} = \begin{cases} 1, & \text{if available rent} < 2, \\ 0, & \text{otherwise,} \end{cases} \qquad \text{full} \ = \begin{cases} 1, & \text{if available return} < 2, \\ 0, & \text{otherwise.} \end{cases}$$

Table 2 presents the classification results of our models. Except for the LSTM without covariates, all other models incorporate additional features such as temporal and weather variables. ASTGCN achieves the best performance for predicting *Empty* events, whereas LLA performs best for predicting *Full* events.

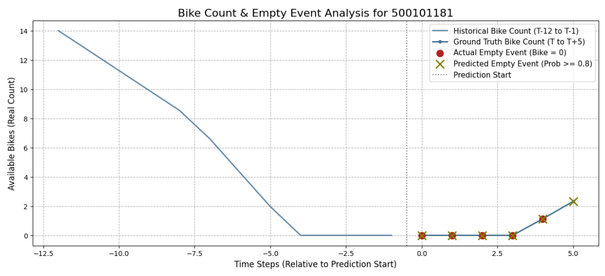
Table 2: Classification performance for predicting *empty* and *full* events.

Model	**Empty**			**Full**		
	Recall	Precision	F1 score	Recall	Precision	F1 score
LSTM (without covariates)	0.5296	0.7665	0.6264	0.6033	0.3900	0.4737
LSTM	0.5463	0.7619	0.6363	0.6080	0.4441	0.5133
LSTM-FCNN	0.6426	0.8610	0.7359	0.8608	0.4991	0.6318
LLA	0.8248	**0.8992**	0.8603	0.8544	**0.6567**	**0.7426**
ASTGCN	**0.8282**	**0.8992**	**0.8723**	**0.8753**	0.6209	0.7265

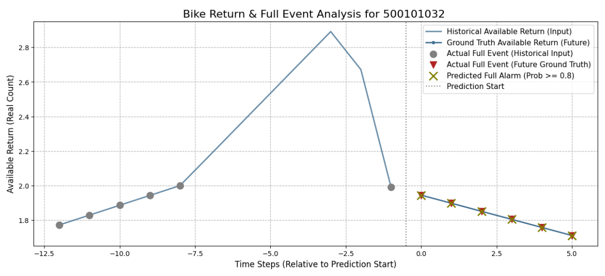
Figures 8 shows randomly selected time windows comparing the 6-step ahead forecasts of two models against the ground-truth number of available bikes and available docks. Figures 9 illustrate the 6th step predictions for 捷運公館站 (3 號出口) from 2025-05-14 08:00:00 to 2025-05-15 20:00:00 (i.e., each point corresponds to a value predicted 30 minutes earlier). We observe that at time points when the station is truly in an empty-bike or full-dock state, the predicted probabilities of *Empty* or *Full* are consistently higher. This indicates that reformulating the task as a classification problem yields predictions that better align with real-world conditions, compared with directly forecasting the exact numbers of available bikes or available docks.



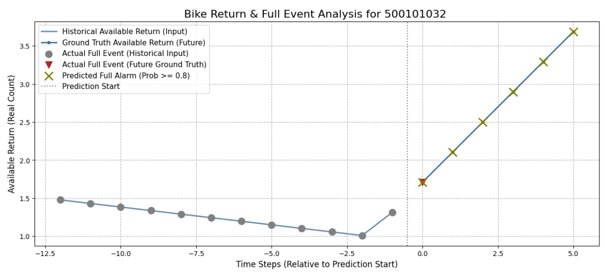
(a) ASTGCN Multi-step prediction for Empty Event -1



(b) ASTGCN Multi-step prediction for Empty Event -2



(c) LLA Multi-step prediction for Full Event -1



(d) LLA Multi-step prediction for Full Event -2

Figure 8: Predictions of **ASTGCN** and **LLA** for *empty* and *full* events under a 6-step ahead setting (focusing on the last 12 steps). In the two upper panels, the y-axis denotes the number of available bikes for rent, while in the two lower panels, the y-axis denotes the number of available docks for returns. Circles indicate event time points within the 12 steps, red triangles mark the ground-truth event occurrences in the six steps, and green crosses represent the event time points predicted by the model.

6 Discussion

6.1 Model Performance and Architectural Trade-offs

In this study, we explored various deep learning architectures for multi-step bike availability forecasting. Our results demonstrate that increasing model complexity does not strictly guarantee better performance. As observed in Section 5.1, the *LSTM-FCNN* model, despite having ten times the parameters of the baseline LSTM, resulted in higher prediction errors[cite: 170]. This suggests that simply increasing capacity without appropriate inductive biases can lead to optimization difficulties given the limited training data[cite: 171].

In contrast, the *LLA* model, which utilizes an encoder-decoder structure with Global Attention, achieved significantly lower errors with fewer parameters than the baseline[cite: 173]. We can try use Local Attention to reduce the parameters of LLA model. Nevertheless, this indicates that the attention mechanism effectively captures the relevant temporal dependencies within the 12-step look-back window[cite: 56]. Furthermore, *ASTGCN* outperformed other models in general metrics [cite: 178], confirming that explicitly modeling both spatial and temporal dependencies is crucial for bike-sharing networks where station statuses are interdependent.

6.2 Spatial Dependency and Commuting Patterns

The adjacency matrix learned by ASTGCN provides interpretable insights into the spatial dynamics of the YouBike network. While self-dependence remains the strongest factor[cite: 209], the model successfully identified non-local correlations. For instance, 捷運公館站 *(3* 號出口*)* showed strong connectivity with 捷運科技大樓站. Although these stations are not geographically adjacent, they share similar commuting characteristics as metro exits. This validates the effectiveness of the graph-based approach in capturing latent traffic patterns that purely distance-based or single-station time-series models would miss.

6.3 The Challenge of Randomness and Temporal Lag

A persistent challenge observed across all models is the "one-step delay" phenomenon, particularly when the ground truth exhibits sharp changes[cite: 183]. As noted in the results, models tend to be conservative or overshoot when facing abrupt V-shaped turning points[cite: 202]. This limitation stems from the inherent randomness of the YouBike system, which lacks stable periodicity due to inconsistent rebalancing schedules and variable user behavior. Because the models rely heavily on recent history (previous 1 hour), they struggle to anticipate sudden spikes caused by external factors (e.g., a rebalancing truck refilling a station) that are not present in the input features.

6.4 Revisiting the Problem: From Forecasting to Classification

One of the most significant findings of this work is the shift in problem formulation. While exact numerical forecasting (Regression) is difficult due to the aforementioned noise, the Classification approach yielded more practically useful results. Users typically do not require the exact number of bikes; rather, they need to know if a station is "Empty" (cannot rent) or "Full" (cannot return). Our classification experiments (Table 2) showed that ASTGCN and LLA could predict these critical events with high F1 scores[cite: 256]. Visualizations in Figure 9 confirm that the predicted probabilities align well with actual empty/full states[cite: 261]. This suggests that for real-world applications, treating bike sharing prediction as a risk assessment task (probability of failure) is more robust than deterministic count forecasting.

6.5 Future Work

To address the limitations of temporal lag, future work could incorporate "bike rebalancing schedules" as an exogenous variable, as sudden supply changes are often operator-driven rather than user-driven. Additionally, expanding the graph to include a wider range of stations could help capture broader flow dynamics. Finally, investigating hybrid loss functions that penalize "missed events" (false negatives on empty/full status) more heavily than numerical errors could further improve the user experience.

References

[1] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[3] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[4] Lei Bai, Lina Yao, Can Li, Xianzhi Wang, and Can Wang. Adaptive graph convolutional recurrent network for traffic forecasting. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[5] Shengnan Guo, Youfang Lin, Ning Feng, Chao Song, and Huaiyu Wan. Attention based spatial-temporal graph convolutional networks for traffic flow forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 922–929, 2019.

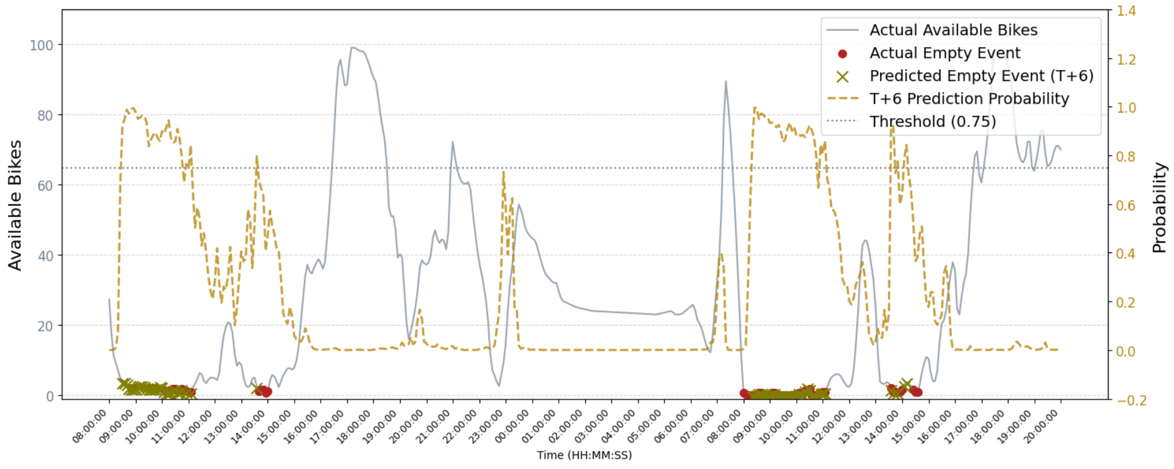
[6] George Kour and Raid Saabne. Real-time segmentation of on-line handwritten arabic script. In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 417–422. IEEE, 2014.

[7] George Kour and Raid Saabne. Fast classification of handwritten on-line arabic characters. In *Soft Computing and Pattern Recognition (SoCPaR), 2014 6th International Conference of*, pages 312–318. IEEE, 2014.

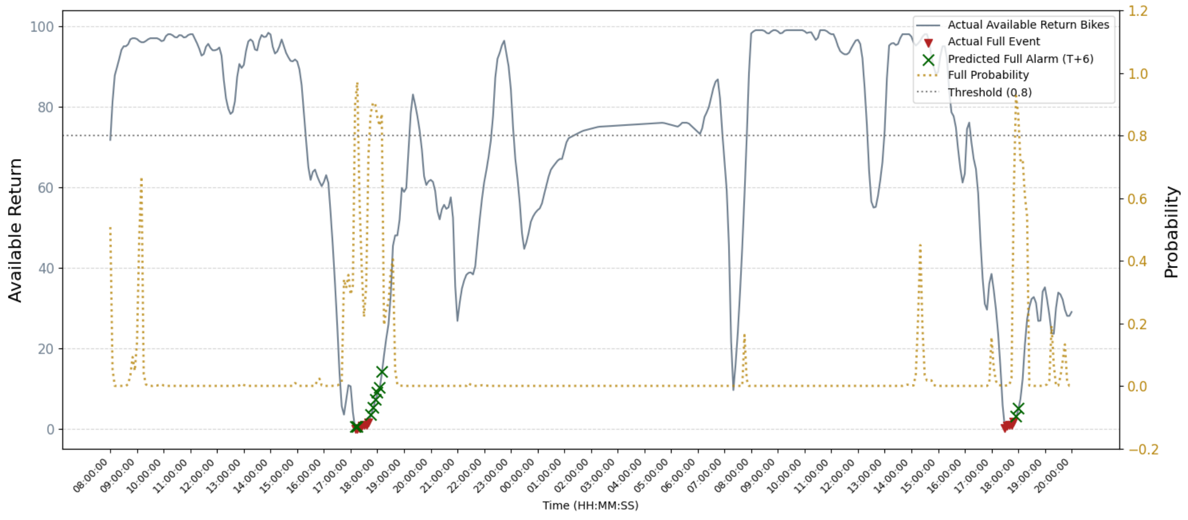
[8] Guy Hadash, Einat Kermany, Boaz Carmeli, Ofer Lavi, George Kour, and Alon Jacovi. Estimate and replace: A novel approach to integrating deep neural networks with existing applications. *arXiv preprint arXiv:1804.09028*, 2018.

[9] tses89214. YouBike Historical Data. `https://github.com/tses89214/youbike-historical-data`, 2025. Accessed: 2025-12-24.

Appendix



(a) ASTGCN’s 6-step ahead prediction. The left y-axis indicates the number of available bikes for rent, corresponding to the gray solid line. The right y-axis shows the model-predicted probability of an *empty* event, corresponding to the yellow dashed line.



(b) LLA’s 6-step ahead prediction is shown in the figure. The left y-axis indicates the number of available docks for return, corresponding to the gray solid line. The right y-axis shows the model-predicted probability of an *full* event, corresponding to the yellow dashed line.

Figure 9: The 6th-step predictions for 捷運公館站 (3 號出口) from 2025-05-14 8AM to 2025-05-15 8PM

分配

組員	負責內容
R13250004 邵子軒	LLA and Discussion
R13250007 王本偉	ASTGCN and Section 5.1
R13250012 張宥婷	Section 5.2 and Motivation
R13250020 翁德軒	LSTM Based Models and Data Analysis