CE290 Final Report: Short-term Airport Throughput Prediction with GA-LSTM

Joey Cai[a], **Albert Cao**[a], **Yusheng Wang**[a], **Jing Yang**[a]
[a]Department of Civil and Environmental Engineering, University of California, Berkeley

1 Introduction

With increasing traffic at airports, forecasting airport throughput with high accuracy is integral to operation planning and decision making. Airport staffing and equipment planning can be assisted by throughput forecasts which improve the resiliency and efficiency of airport operations. Another motivation behind the project is the unprecedented disruption that COVID-19 created in the air transportation system. During the recovery process, predicting short-term air traffic has become a major challenge that airlines and airport operators face. Traditionally, minor adjustments are implemented to accommodate the year-over-year increase in demand but few models have been built to model the trend of recovery.

To better model and predict the short-term airport throughput, we developed a deep learning model, Graph-Assisted Long Short Term Memory (GA-LSTM), by superimposing the graph topology of the air transportation network on time-series inputs. We found using the graph structure of the airport yields statistically significant better forecasts compared to the benchmark methods.

Leveraging the trained LSTM model, we employ a greedy algorithm to identify the optimal distribution of flight increase that would yield the greatest increase in future profit. We focus on a hub and spoke system within the Aviation System Performance Metrics (ASPM) 77 airport network.

2 Literature Review

This literature review provides an overview of the theoretical basis of our model in the context of deep learning, time-series forecasting, and graph-assisted methods.

Long Short-Term Memory (LSTM), introduced by Hochreiter and Schmidhuber,[1] is a type of Recurrent Neural Network (RNN) architecture designed to address the limitations of conventional RNNs when learning long-range dependencies in sequence data. It has become popular for various sequence-based tasks such as natural language processing, recognition and time series prediction.

In the field of predicting network flow, the theory of applying convolution methods and other deep-learning neural networks has been continuously proposed and renovated.[2] applied a spatiotemporal recurrent convolutional networks (SRCNs) model that retains the fine-scale structure and model spatio-temporal dependencies in urban traffic networks. In literature,[3] a GAT-LSTM model that combines graph attention neural networks with LSTM units to accurately predict short-term airport throughput over a national air traffic network was used, tested with real-world data from 65 major airports in China. Other research[4][5] using convolutional neural network extract Euclidean features in traffic networks and presented either a 2D or 3D approach to the topic.

Given the complex connections and dependencies that exist within airport networks as well as urban road traffic networks, incorporating graph information in flow forecasting for airport networks is crucial. Considering spatial weighted recurrent neural networks and graph convolution

Fig 1 Total Throughput at ASPM 77 Airports in 2020

recurrent neural networks to enhance flight delay and traffic prediction accuracy is applied in some recent research[6,7] [8].

Building upon the research that combines deep learning and graph information, we have opted for a simpler and more direct approach to model the airport network by utilizing an adjacency matrix to represent spatial connections and assigning weights based on quantity. This streamlined methodology aims to efficiently capture the intricate relationships within the airport network while still offering accurate predictions and insights for informed decision-making in the aviation industry.

3 Methodology

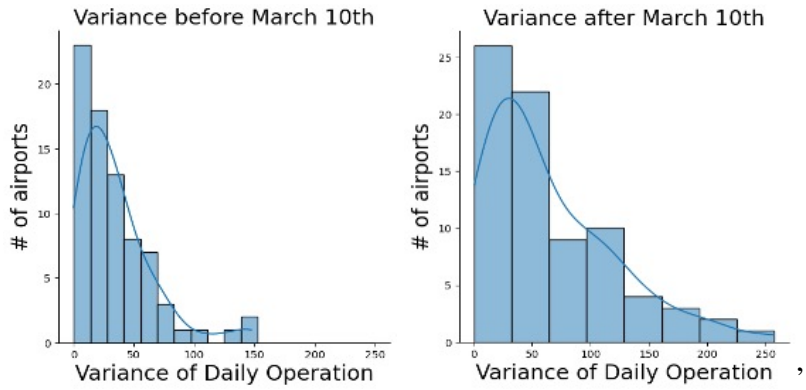
3.1 Data Acquisition

We download the daily segment traffic for ASPM 77 airports from FAA's Aviation System Performance Metrics (ASPM) database. We select the domestic, jet, carrier flights in the year 2020. We further define the daily throughput at an airport as the sum of the number of arrivals and departures in a single day.

3.2 Exploratory Data Analysis and Standardization

The domestic air transportation system experienced a drastic decline in traffic in March 2020, and gradually recovered after July. Figure 1 shows the total throughput at the 77 airports, which dropped from 14,000 to less than 3,000 within days. It is also noted that the system has large spatial and temporal variability, which makes the neural network unable to identify and decode the trends in time-series data. Figure 2 shows the variance of daily throughput at each airport before and after March 10^{th}, when the lockdown was announced. The number of airports with variance more than 100 grew significantly after the lockdown, revealing an increase in temporal variability in the system

Furthermore, because the degrees of airports in the air transportation network exhibit a power law distribution, there also exists large spatial variability in our data.[9] Figure 3 illustrates 3 airports that are selected to reveal the large spatial variability. The figure shows that the throughput at Atlanta Hartsfield-Jackson International Airport (ATL) is twice the throughput of San Francisco International Airport (SFO) and 4-5 times more when compared to Austin-Bergstrom International Aiprort (AUS).



(a) (b)

Fig 2 Temporal Variability in the ASPM 77 System

Fig 3 Spatial Variability in ASPM 77 Airports in 2020

As a result, we implement the following standardization scheme to scale the throughput data of the model. We define an operator $Z_k(\cdot)$ such that:

$$Z_{k,t} = \frac{X_{k,t} - \bar{X}_k}{\sigma_k} \quad \forall k \in K$$

where K is the set of 77 ASPM airports and σ_k denotes the standard deviation of throughput at airport k in the training set and $\bar{X}_k$ denotes the average throughput of airport k in the training set. $X_{k,t}$ represents the throughput of airport k at t time step of the training set. Hence, $Z_{k,t}$ is the input to the GA-LSTM model. Similarly, we define an inverse standardization operator $Z_k^{-1}(\cdot)$ such that:

$$\hat{Y}_{k,t'} = \hat{Z}_{k,t'} \cdot \sigma_k + \bar{X}_k$$

where $\hat{Y}_{k,t'}$ is the predicted throughput of airport k at time step $t\prime$ in the output data and $\hat{Z}_{k,t'}$ is the direct output of the machine learning model. Both σ_k and $\bar{X}_k$ are inherited from the operator $Z_k(\cdot)$ we define previously on the input data set.

3.3 GA-LSTM

Our GA-LSTM model is consists of one LSTM layer, two hidden layers with ReLu and linear activation functions respectively, and one dropout unit to prevent over-fitting. The number of nodes

chosen for the LSTM layer and the second hidden layer is n_pred * 77 where n_pred denotes the number of days to forecast in the future. The number of nodes in the first hidden layer is $77//2$ where $//$ is the floor operator. The probability of dropout is set to 0.2.

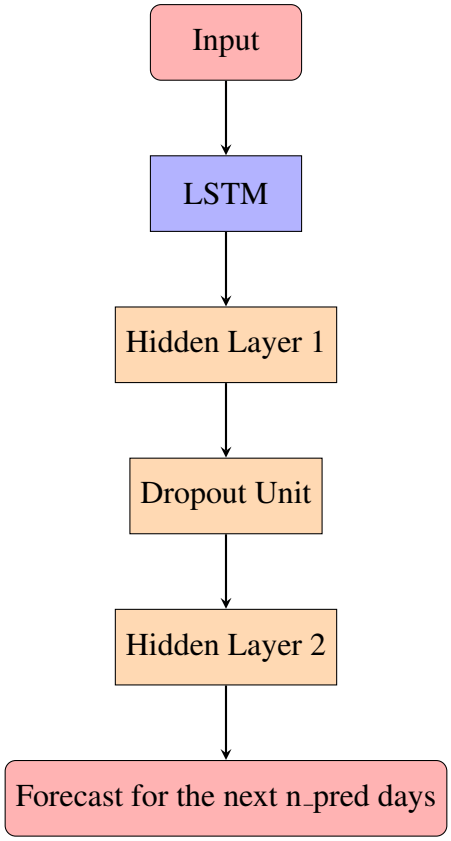


Table 1 Parameters in GA-LSTM

Parameter	Brief description
start_day	Start day of the training set
n_timesteps	Number of days used as the input in each batch for training and testing
n_training	Number of days used as features in training
n_pred	Number of days to forecast for

Table 1 shows the parameters used in constructing the GA-LSTM model and their respective definition. Figure 3.3 shows the process of constructing the training and testing batches using the sliding window mechanism. In this example, start_day is set to 0, n_timestep is set to 21 days, n_training is set to 56, and n_pred is set to 7 days. We use the first 56 days to construct 28 training batches, each with an input time series of 21 days and an output time series of 7 days. Then we slide the window forward by one day to create 28 training batches. To predict the throughput from day 63 to day 70, an additional day is added to the last 7 training batches. The 7^{th} day of the output

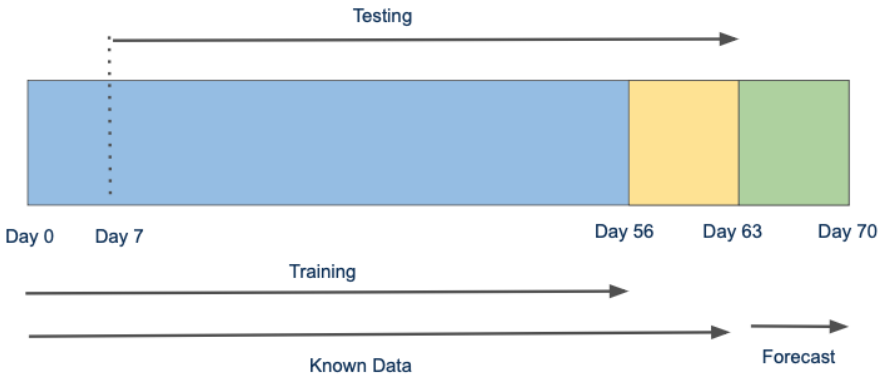


Fig 4 Sliding Window Batch Construction

from each batch is taken to reconstruct the time-series. Hence, a 7 day buffer is needed, which is the yellow region, between the forecast and the training data. To incorporate spatial information in the model, we define a spatial correlation matrix $A \in \mathbb{R}^{K \times K}$, where $A_{i,j}$ denotes the fraction of days with flights from airport j to i within the n_timesteps of each batch. The input matrix $X \in \mathbb{R}^{n\_timesteps \times N}$ is further multiplied by the correlation matrix. Thus, the input matrix X'_{batch} is defined as:

$$X'_{batch} = X_{batch} \cdot A_{batch}$$

3.4 Optimization Setup

The optimization process focuses on determining the optimal distribution of n flights to a hub and spoke system, where n is the total number of added flights each day. In a hub and spoke system, the hub is a connection point where passengers can transfer from one flight to another and spoke a route from or to the hub. We selected Salt Lake City International Airport (SLC), a Delta Airline hub with 70 percent Delta flights, as the hub in the system because Delta Airlines plans to expand this hub in the next 20 years.[10] For the spokes in the system, We selected 14 flight routes with greater than 60 flights and less than 123 flights to or from SLC.

3.5 Greedy Formulation

The process of finding the optimal distribution of n added flights involves using a greedy algorithm to break the problem into smaller sub-problems and find the optimal solution for each sub-problem based on the rewards generated from each action. The sub-problem in this case is to determine the placement of 1 additional flight to the hub and spoke system. The action for each step is to add 1 flight each day for 7 consecutive days to one of the spoke. In the greedy algorithm, one flight is added to both the hub airport and the spoke airport (the non-hub airport in a spoke) to indicate that the additional flight is between the hub and the spoke airport. The reward for each action is calculated as the change in model predicted number of flights times a discount rate times 10,000 dollar in profit per each additional flight.

$$Reward = \Delta\ Model\ Predicted\ Number\ of\ Flights \cdot (\gamma^x) \cdot (10,000\ dollars)$$

where γ is a discount between 0.1 to 1 (inclusive) and x is the number of flights added to the spoke before the current action. We introduced a discount rate of γ to the x power because marginal profit decrease exponentially with each additional flights.

3.6 Optimization Process

The greedy algorithm proceeds as the flowchart shown in Figure 5, and a detailed description is provided below.

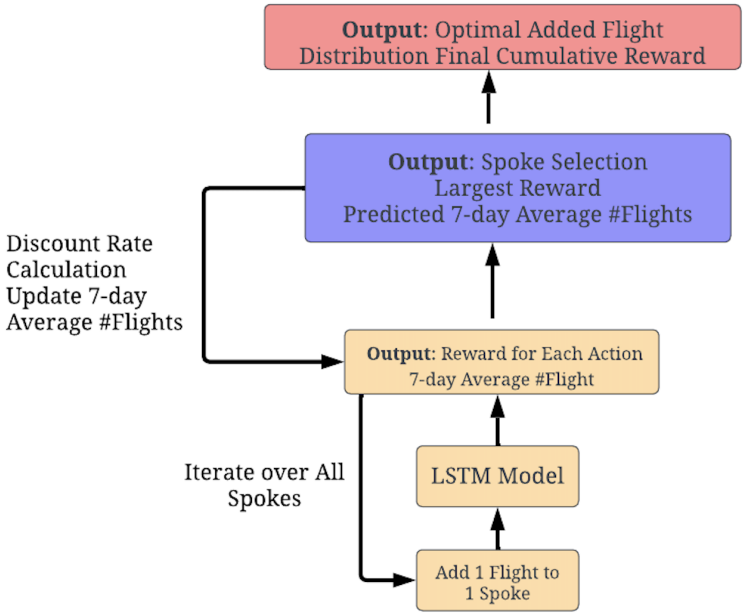


Fig 5 Flowchart for Greedy Algorithm

First, testing data was input into the trained LSTM model to predict the next 7-day average number of flights at SLC airport.

Next, for each of the 14 spokes, the testing data was modified by adding one flight trip from the hub to the spoke. The LSTM model was then used to predict the increase in the next 7-day average number of flights for each modified testing data, and a reward was calculated for each spoke.

The optimal spoke was then determined by selecting the one with the highest reward. The corresponding flight was added to the optimal spoke, and the testing data and the next 7-day average number of flights were updated accordingly. This process was repeated for each additional flight until the optimal spoke for the last flight was obtained.

Finally, the optimal flight distribution was recorded, and the cumulative reward was calculated.

4 Results

4.1 Sample Forecast Results

Figure 6 contains 4 sample forecast results at Newark Liberty International Airport (EWR), Dallas-Fort Worth International Airport (DFW), and Las Vegas Harry Reid International Airport (LAS). We select mid-June as the forecast period to display GA-LSTM's performance on predicting recovery of the air transportation system from COVID-19 lockdowns. Overall, the output of GA-LSTM closely resembles the training data. For the last 7 days in the testing set, which are the forecasted days, it shows that as the prediction extends further into the future, the error becomes larger.

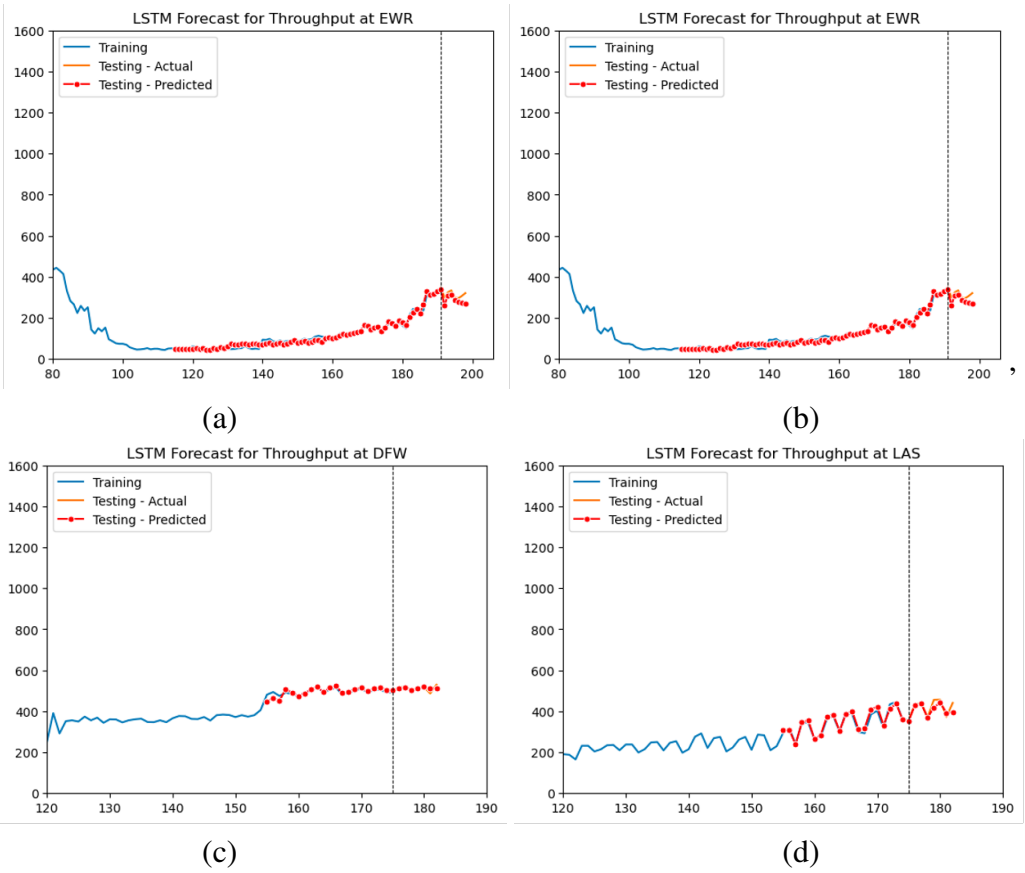


Fig 6 Sample Forecast Results

4.2 Error Analysis

Three benchmark models are created to compare the forecast accuracy of GA-LSTM. Extreme Gradient Boosting (XGBoost) performs time series forecasting by reconstructing the historical time series data set using the sliding window mechanism and predicting future values based on gradient-boosted trees. Auto-regressive Integrated Moving Average (ARIMA) uses various parameters to reconstruct a regression on time series data, including the number of lags, the order of the moving average components, and the number of past forecast errors. ARIMA can be used to identify seasonality and trends, and facilitate forecasts. A simple LSTM, where the input is not multiplied by the spatial correlation matrix, is also used. It is recognized that both LSTM and GA-LSTM are stochastic models. Hence, a sample of 10 runs is collected for both methods for comparison.

We define two metrics, Root Mean Square Error (RMSE) and Mean Absolute Error (MAE), as errors for evaluation:

$$RMSE = \sqrt{\frac{\sum_{s\in S}\sum_{t\in T}\sum_{k\in K}(y_{s,t,k} - \hat{y}_{s,t,k})^2}{|T||K||S|}}$$

$$MAE = \frac{\sum_{s\in S}\sum_{t\in T}\sum_{k\in K}|y_{s,t,k} - \hat{y}_{s,t,k}|}{|T||K||S|}$$

T is the set of forecast. In the example, $|T|$ = 7. K is the set of airports and $|K|$ = 77. S is the set of start day of predictions and $S = \{120, 150, 180, \ldots, 330\}$. A wide range of dates of forecast is chosen to account for the temporal variability. Table 2 displays the errors for the benchmark models and GA-LSTM.

Table 2 Error Metrics

	RMSE	**MAE**
ARIMA	33.006	2.763
XGBoost	10.648	2.891
LSTM[1]	9.147 (0.126)	1.741 (0.048)
GA-LSTM[2]	8.808 (0.219)	1.634 (0.065)

[1] RMSE and MAE for LSTM and GA-LSTM are computed as an average of 10 samples
[2] Difference in errors are statistically significant at $\alpha = 0.05$

4.3 Optimization Result

The optimal flight distribution to spokes and total cumulative reward determined using the greedy algorithm is compared with results from 4 random action trials. For each random trail, the distribution of each additional flight to the spoke is randomly chosen. Table 3 and 4 shows the optimal distribution of 15 additional flight to the system with a discount factor of 0.8 using random actions and the greedy algorithm respectively.

Table 3 Distribution of the 15 Flights for each Random Action Trail

	ABQ	**AUS**	**BOS**	**BWI**	**DTW**	**EWR**	**FLL**	**MCI**	**MCO**	**ONT**	**PSP**	**SAT**	**STL**	**TUS**
Random Action 1	0	1	1	0	1	0	0	1	3	2	1	1	4	0
Random Action 2	1	0	2	1	3	1	1	1	0	0	2	1	0	2
Random Action 3	2	2	1	2	0	1	0	0	3	2	1	0	0	1
Random Action 4	0	1	2	1	2	0	1	1	0	1	2	3	1	0

Table 4 Distribution of the 15 Flights Using the Greedy Algorithm

	ABQ	**AUS**	**BOS**	**BWI**	**DTW**	**EWR**	**FLL**	**MCI**	**MCO**	**ONT**	**PSP**	**SAT**	**STL**	**TUS**
Greedy Algorithm	3	0	0	0	0	0	0	2	0	5	1	0	2	2

Table 5 and 6 shows the total cumulative reward gained from using each method method. As shown in the reward tables, the total cumulative reward is the largest when using the greedy algorithm to determine flight distribution.

Table 5 Final Cumulative Reward for each Random Action Trial

	Reward
Random Action 1	1517
Random Action 2	1168
Random Action 3	1648
Random Action 4	1184

Table 6 Distribution of the 15 Flights using the Greedy Algorithm

	Reward
Greedy Algorithm	2504

5 Discussion

With spatial correlation matrix superimposed on the original input matrix, which is the historical throughput at 77 airports, the accuracy of the model improves significantly. Figure 2 shows a statistically significant decrease is observed for both RMSE and MAE.

5.1 Sensitivity Analysis

Using the optimization method described earlier, we investigate the impact of discount factors on the optimal action. Figure 7 displays the optimal actions for different discount factors, where each row corresponds to a specific discount factor. Our findings indicate that a high discount factor tends to distribute flights evenly among spokes, while a low discount factor favors concentrating flights in a single spoke.

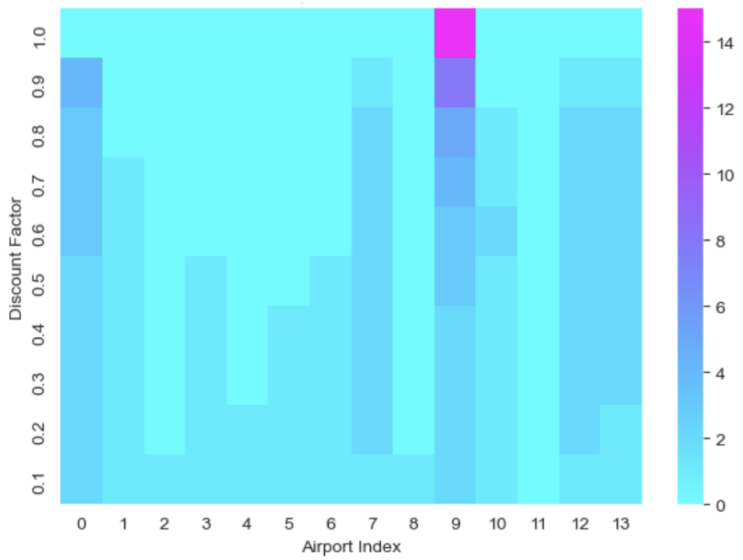


Fig 7 Optimal Action at Different γ

6 Conclusion

In this project, we incorporated a spatial correlation matrix in the LSTM model to predict short-term airport throughput in 2020. We find that adding spatial information to the neural network model significantly improves the accuracy of the forecast. We also find that the model performs better when forecasting for airports with high throughput volume and that the number of days used as input is negatively correlated with errors.

We develop a greedy algorithm to determine the optimal distribution of flights to a hub and system that maximize profit gain from these additional flights. In addition, we conduct a sensitivity analysis on the reward discount factor and find that a high discount factor tends to distribute flights evenly among spokes, while a low discount factor favors concentrating flights in a single spoke.

For future research, a rigorous mathematical interpretation of the spatial correlation matrix should be developed. Although the spatial correlation matrix improves the accuracy of the forecast, a deeper understanding is needed to exploit the spatial correlation between airports in an air transportation network. Additionally, using the GA-LSTM model as a foundation, building a deep learning model that can predict the emergence of new edges or connections between two airports is also of great interest.

7 Appendix

Acknowledgments

Junhui Su also contributed to this project.

References

1 S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation* **9**(8), 1735–1780 (1997).

2 H. Yu, Z. Wu, S. Wang, *et al.*, "Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks," *Sensors (Basel, Switzerland)* **17**(7), 1501– (2017).

3 X. Zhu, Y. Lin, Y. He, *et al.*, "Short-term nationwide airport throughput prediction with graph attention recurrent neural network," *Frontiers in artificial intelligence* **5**, 884485–884485 (2022).

4 D. Chai, L. Wang, and Q. Yang, "Bike flow prediction with multi-graph convolutional networks," in *Proceedings of the 26th ACM SIGSPATIAL International Conference on advances in geographic information systems*, *SIGSPATIAL '18*, 397–400, ACM (2018).

5 D. Tran, L. Bourdev, R. Fergus, *et al.*, "Learning spatiotemporal features with 3d convolutional networks," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 4489–4497, IEEE (2015).

6 K. Guo, Y. Hu, Z. Qian, *et al.*, "Optimized graph convolution recurrent neural network for traffic prediction," *IEEE transactions on intelligent transportation systems* **22**(2), 1138–1149 (2021).

7 X. Zhu and L. Li, "Flight time prediction for fuel loading decisions with a deep learning approach," *Transportation research. Part C, Emerging technologies* **128**, 103179– (2021).

8 Y. J. Kim, S. Choi, S. Briceno, *et al.*, "A deep learning approach to flight delay prediction," in *2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*, 1–6, IEEE (2016).

9 A. T. R. Guimera, S. Mossa and L. A. N. Amaral, "The worldwide air transportation network: Anomalous centrality, community structure, and cities' global roles," *PNAS* **102**, 7794–7799 (1005).

10 "Delta's salt lake city expansion plan moves forward; delta is constantly raising the bar to further serve the needs of salt lake city's travelers," *ENP Newswire* (2023).

List of Figures

List of Tables