

Article

Lightweight Graph Embedding Augmentation for Airport Traffic Forecasting

Ahmed Alharbi 

Department of Computer Science, Taibah University, Madinah 42353, Saudi Arabia; atharbi@taibahu.edu.sa

Abstract

Short-term airport traffic forecasting faces a structural gap: temporal ensemble models ignore route-network dependencies that shape hub operations, while deep graph neural networks require synchronised multi-airport operational data streams unavailable to single-airport operators, who have access only to their own operational records and publicly available route topology. To our knowledge, this study provides the first systematic evaluation of three graph representation classes—centrality measures, DeepWalk, and Node2Vec—as structural augmentations to Random Forest (RF), XGBoost, and LightGBM for hourly aircraft movement prediction at King Khalid International Airport (RUH), using a two-hop aviation graph combining RUH operational data with the OpenFlights database. Across all three ensemble families, random-walk graph augmentations consistently reduce MAE by approximately 9–17% relative to temporal-only baselines, whereas handcrafted centrality measures provide smaller and less consistent gains. Diebold–Mariano tests confirm that both RF+DeepWalk and RF+Node2Vec significantly outperform all nine baseline models ($p < 0.05$), while no statistically significant difference is observed between the two embedding methods within any ensemble family, indicating that the benefit arises from the class of random-walk representations rather than a specific algorithm. RF+DeepWalk achieves the lowest observed MAE of 1.810 (RMSE = 2.481, sMAPE = 6.17%). SHAP analysis indicates that graph embedding dimensions rank among the top predictors, suggesting that they capture structural signal absent from temporal features.

Keywords: airport traffic forecasting; graph embeddings; DeepWalk; Node2Vec; random forest; XGBoost; LightGBM; aviation network; Diebold–Mariano test; SHAP interpretability

1. Introduction

Ensemble learners such as Random Forest, XGBoost, and LightGBM have become strong baselines for aviation forecasting, consistently outperforming classical statistical models on flight delay and demand prediction [1–3]. They are computationally efficient and robust on the medium-scale tabular datasets typical of single-airport operations. Accurate short-term traffic forecasts are operationally critical: airport operators rely on them to schedule ground crews, allocate gates, coordinate apron resources, and plan fuel logistics on an hourly basis [4]. The limitation is structural: they treat each airport as an isolated node. A hub such as King Khalid International Airport (RUH) is embedded in a dense global network whose hourly traffic reflects not only recent local patterns but also cascading schedule dependencies from partner hubs and concentrated demand events—including the annual Hajj pilgrimage, one of the largest traffic surges in global aviation [5–7]. Temporal features alone cannot encode these relationships.



Academic Editors: Chengjia Han and Zheng Tong

Received: 29 May 2026

Revised: 24 June 2026

Accepted: 29 June 2026

Published: 3 July 2026

Copyright: © 2026 by the author.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

Deep spatio-temporal graph neural networks do capture network structure, but their deployment requires synchronised operational data streams—real-time or historical traffic readings—from many airports simultaneously, in addition to substantial computational resources [8–10]. Recent aviation-specific graph models, including spatio-temporal networks for airport capacity and traffic flow prediction [11,12], follow this same paradigm. In practice, traffic readings from partner airports are rarely available in a standardised, synchronised format suitable for network-wide forecasting, as each airport is managed by a separate authority under different national regulations. What is publicly available is structural route topology—which airports serve which routes—through databases such as OpenFlights. This distinction is critical: structural topology tells you how airports are connected, but carries no operational signal about what is happening at those airports at any given time. The framework proposed in this study follows exactly this principle: it combines the target airport’s operational observations with publicly available route topology, while requiring no operational data from any other airport.

No prior work has systematically compared lightweight graph representations—from simple connectivity statistics to distributed random-walk embeddings—as augmentations to ensemble learners for single-airport traffic volume forecasting, or established which class of structural representation provides meaningful predictive benefit. Temporal models discard structural information entirely, while graph neural networks demand data that single-airport operators do not have; lightweight pre-computed embeddings represent an unexplored middle ground that could bridge this divide. The present setting is methodologically distinct from three related problem classes. Flight delay prediction models a binary or continuous disruption event, typically aggregating causal features across a network of connected airports [1,5]; here, the target is hourly volume, not delay. Network-wide flight flow forecasting requires synchronised operational observations from multiple airports simultaneously [8,11,13]; here, only the target airport’s own records are available. General transportation forecasting operates on sensor networks where all nodes are observable in real time [14,15]; aviation lacks this infrastructure at the single-operator level. Nevertheless, valuable structural information is still contained in publicly available route topology, including which airports are connected and the relative strength of these connections. The unique challenge is therefore how to exploit this structural signal without access to multi-airport operational data, a setting in which the data requirements of spatio-temporal graph neural networks are difficult to satisfy because they typically rely on synchronised observations from multiple airports. Lightweight pre-computed graph embeddings address this gap directly: they extract structural context from route topology alone, requiring no operational observations from any airport other than the target. RUH is a well-suited testbed for this study: it is a large international hub with a strongly structured seasonal demand profile, a rich set of direct and second-order route partners, and complementary public route data available through the OpenFlights database.

To address this gap, we investigate whether structural network information can improve single-airport traffic forecasting when encoded as lightweight graph representations rather than data-intensive graph neural networks. We develop a graph-augmented forecasting framework that combines the target airport’s operational data with publicly available route topology and systematically compares handcrafted centrality measures with distributed random-walk embeddings (DeepWalk [16] and Node2Vec [17]) across three ensemble learning families (RF, XGBoost, and LightGBM), applied to hourly aircraft movement forecasting at King Khalid International Airport (RUH). The graph is constructed by combining RUH operational flight data with the OpenFlights public routes database, producing a two-hop global aviation network that captures both direct route partners and second-order hub influences.

Our main contributions are as follows:

1. We present the first systematic evaluation of lightweight graph representations—centrality measures, DeepWalk, and Node2Vec—as augmentations to temporal ensemble learners for single-airport traffic forecasting, using a two-hop aviation graph combining RUH operational data with the OpenFlights routes database.
2. We show that random-walk graph embeddings provide substantial and statistically significant forecasting gains (9–17% MAE reduction across ensemble families), whereas handcrafted centrality measures provide smaller and less consistent improvements.
3. We confirm via the Diebold–Mariano test that both RF+DeepWalk and RF+Node2Vec significantly outperform all nine baseline models ($p < 0.05$), while the choice between DeepWalk and Node2Vec does not reach statistical significance within any ensemble family, with RF+DeepWalk achieving the lowest observed MAE of 1.810.
4. We demonstrate that the performance gains generalise across three ensemble families (RF, XGBoost, and LightGBM), five terminals, and five consecutive walk-forward evaluation months, with SHAP analysis confirming that graph embedding dimensions contribute independent predictive signal beyond temporal features.

Unlike deep spatio-temporal graph neural networks, the framework is lightweight and interpretable, offering a practical balance between accuracy, computational cost, and deployment simplicity. The remainder of this paper is organised as follows. Section 2 reviews related work. Section 3 describes the proposed framework. Section 4 outlines the evaluation protocol. Section 5 presents and discusses the experimental results. Section 6 summarises findings and outlines future work.

2. Related Work

2.1. Time-Series Forecasting in Aviation

The most common approach to airport traffic forecasting treats it as a pure time-series problem. Classical models such as ARIMA and SARIMA have been widely used for traffic volume, passenger demand, and delay prediction [18,19]. Their interpretability is a practical advantage, but the assumption of linearity and stationarity rarely holds at a major hub [20].

Machine learning regressors address the nonlinearity. Among them, Random Forest, XGBoost, and LightGBM have become consistently strong performers across aviation prediction tasks. Hatipoğlu et al. [1] demonstrated that gradient boosting methods, including XGBoost and LightGBM, outperform classical statistical approaches for flight delay prediction. Dai [2] showed that Random Forest-based architectures effectively model complex, nonlinear patterns in aviation operational data. Jafari and Lewison [3] applied deep neural networks to air passenger demand forecasting, noting that well-engineered ensemble models remain competitive. In the Saudi aviation context, Alfarhood et al. [7] applied machine learning to predict flight delays for Saudi Arabian Airlines, confirming the importance of data-driven approaches for Gulf-region airports. More broadly, tree-based ensemble methods have shown strong and reproducible performance in transportation demand forecasting [4,21].

2.2. Graph-Based Learning for Transportation Networks

What temporal models cannot capture is structural: airports are not independent. A small number of hub airports dominate global connectivity [22–24], and disruptions propagate along route connections in ways that no time-series model recovers from local history alone [5,25,26]. Representing airports as nodes and flight routes as directed weighted edges makes these dependencies explicit [23].

Graph-based learning techniques have been widely explored across transportation domains [14,15]. Shallow embedding algorithms such as DeepWalk [16] and Node2Vec [17] generate low-dimensional representations that preserve both local and global structural relationships. DeepWalk uses uniform random walks, treating all neighbours equally to capture global community structure, while Node2Vec extends this idea with biased walks that balance homophily and structural equivalence, making both methods suitable for complex infrastructure networks. Compared with deep graph neural networks (GNNs), which require large labelled datasets and extensive hyperparameter tuning, these lightweight embedding methods offer scalable and interpretable alternatives for unsupervised network representation [10].

2.3. Hybrid Temporal–Graph Modelling

The natural extension is to combine temporal and graph learning. In road and rail networks, spatio-temporal architectures—DCRNN [27], STGCN [28], GraphSAGE [29], and graph attention networks [30]—have shown strong results by coupling graph convolutions with sequence models [14,15], with recent work combining dynamic and static graph representations for improved traffic flow modelling [31]. Sequence-based approaches using structured state-space models have also been proposed as graph-free alternatives for traffic flow prediction [32], though they forgo the explicit connectivity structure that network-level propagation requires. Aviation has seen limited adoption: Zheng et al. [8] applied a graph multi-attention network to network-wide delay prediction, Franco et al. [9] modelled holding-maneuvre delays as a graph learning problem, Qian et al. [11] proposed a spatio-temporal graph attention network for airport capacity prediction, Huang et al. [13] developed a dynamic multi-graph convolutional network for airport arrival flow forecasting, Zang et al. [33] combined ResNet, GCN, and LSTM to predict flight flow across a network of airports, and Yan et al. [12] applied graph convolution on a semantic airport network for short-term traffic flow prediction. Tang et al. [34] used cascaded gradient boosting for turnaround prediction, showing that ensemble methods remain relevant even in graph-adjacent aviation tasks. The shared limitation is data: these architectures require synchronised operational observations from multiple airports [10], data that are rarely available in a form suitable for network-wide forecasting [23]. As a result, ensemble learners in aviation are applied without network structure [1,2], while graph architectures remain out of reach for single-airport settings [8,9,11–13,33]. The consequence is a practical vacuum: ensemble learners are deployed without any structural information, and graph architectures that could capture it require data that single-airport operators cannot access. What remains unexplored is whether the structural signal latent in publicly available route topology—which carries no operational observations from other airports—can be extracted through lightweight pre-computed embeddings and transferred into ensemble learners as static features. This is the gap the present work addresses.

3. Methodology

3.1. Framework Overview

We propose a graph-augmented airport traffic forecasting framework that systematically evaluates lightweight graph representations—centrality measures, DeepWalk, and Node2Vec—as structural augmentations to RF, XGBoost, and LightGBM applied to hourly aircraft movement prediction. The framework captures short-term operational dynamics through temporal features and longer-range network dependencies through pre-computed graph embeddings derived from a two-hop global aviation graph. Figure 1 illustrates the overall workflow. RF+DW achieves the lowest observed error; the choice between DeepWalk and Node2Vec does not reach statistical significance within any

ensemble family, and all three ensemble families confirm that the benefit generalises across architectures.

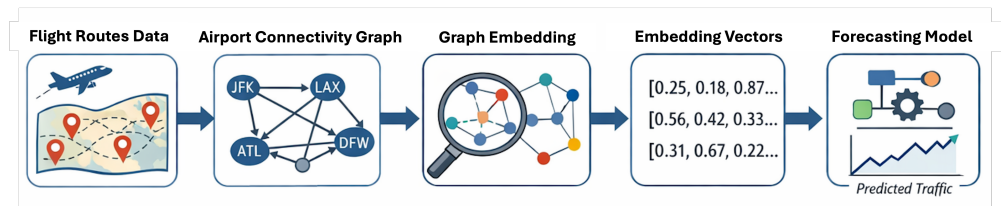


Figure 1. Overview of the proposed graph-augmented airport traffic forecasting framework. RUH flight data and OpenFlights routes are combined to construct a two-hop airport graph, which is embedded using lightweight random-walk graph embeddings (DeepWalk and Node2Vec) and concatenated with temporal features for ensemble forecasting. The embedding panel label and airport codes shown are generic illustrations; ellipses indicate full 64-dimensional vectors truncated for display; node colors represent structural graph communities; arrows indicate the sequential processing pipeline.

3.2. Problem Formulation

Let y_t denote the number of aircraft movements (arrivals and departures) at the target airport during hour t . The goal is to predict next-hour traffic $\hat{y}_{t+1}$ based on historical observations and contextual information:

$$\hat{y}_{t+1} = f(X_{t-k:t}, G) \quad (1)$$

where $X_{t-k:t}$ represents historical temporal observations and airport-specific contextual features from the previous k hours, and G denotes the global airport connectivity graph. This formulation differs from traditional univariate models by incorporating network-level information, recognising that changes at major hubs can propagate to connected airports and influence short-term traffic patterns [5].

3.3. Dataset Description

We use two publicly available data sources.

(1) RUH Flight Movement Dataset. The primary dataset is a publicly available operational flight movement record for King Khalid International Airport (RUH) in Riyadh, Saudi Arabia, sourced from Kaggle (<https://www.kaggle.com/datasets/mohammedalsubaie/king-khalid-international-airport-flights-dataset> (accessed on 1 May 2025)). It contains 153,308 flight movement records spanning approximately seven months of operational data. Each record includes: scheduled timestamp (UTC and local time), origin and destination airports (ICAO/IATA), movement type (arrival or departure), terminal assignment, aircraft model and registration, and airline identifiers. This dataset is used for temporal feature engineering, hourly movement count aggregation (the forecasting target), and the construction of RUH-centric edge weights in the airport graph.

(2) OpenFlights Routes Database. The secondary dataset is the OpenFlights global routes database (<https://openflights.org/data> (accessed on 1 May 2025)), which provides over 67,000 directed routes between airports worldwide. This dataset is used exclusively for graph construction to supply the second-hop connectivity needed to extend the airport graph beyond RUH's direct route partners. OpenFlights data is not used for temporal feature engineering or as forecasting targets.

RUH operates five terminals serving a diverse mix of domestic and international flights. The airport experiences notable seasonal traffic variation driven by summer peak travel and pilgrimage periods (Hajj and Umrah) [6,7], making it a challenging and representative testbed for short-term traffic forecasting.

3.4. Data Preprocessing and Temporal Aggregation

All timestamps are converted to the local time zone to ensure a unified temporal reference. Flight events are aggregated at hourly resolution; for each hour t , the total movement count is

$$y_t = \sum_{i \in \mathcal{F}_t} 1 \quad (2)$$

where $\mathcal{F}_t$ denotes the set of all non-cancelled flight records within hour t . Hours with zero movements are retained to preserve the true operational profile and prevent bias toward high-traffic periods.

3.5. Feature Engineering

We engineer temporal features encoding recent traffic dynamics and recurrent scheduling cycles, complemented by random-walk graph embeddings (DeepWalk and Node2Vec) that provide network-level context.

(1) Lag Features.

Lag features model the autoregressive memory of airport activity. For each hour t ,

$$\text{Lag}_k(t) = y_{t-k}, \quad k \in \{1, 3, 6\} \quad (3)$$

Lag-1h captures immediate temporal memory; Lag-3h captures short-term wave structures; Lag-6h represents longer periodic dependencies.

(2) Rolling Mean Features.

Rolling averages smooth short-term noise and highlight gradual traffic trends:

$$\text{RollMean}_k(t) = \frac{1}{k} \sum_{i=1}^k y_{t-i}, \quad k \in \{3, 6\} \quad (4)$$

RollMean-3h captures transitional periods; RollMean-6h reflects broader operational cycles.

(3) Calendar Features.

Calendar features capture periodic cycles from daily and weekly scheduling patterns. Hour of day is encoded using trigonometric transformations to preserve continuity between hour 23 and hour 0:

$$\begin{aligned} \text{hour_sin}(t) &= \sin\left(\frac{2\pi \cdot \text{hour}(t)}{24}\right), \\ \text{hour_cos}(t) &= \cos\left(\frac{2\pi \cdot \text{hour}(t)}{24}\right) \end{aligned} \quad (5)$$

Additional calendar variables include weekday (0–6), a Saudi weekend indicator (Friday–Saturday), and month of year. All calendar features are exogenous temporal indicators derived from the forecast timestamp and fully observable before the next-hour prediction is generated; they are not computed from the target variable y_{t+1} and therefore do not introduce label leakage. Table 1 summarises all temporal features.

Table 1. Summary of temporal feature sets used in this study.

Feature Set	Feature	Description
Lags	Lag-1h	Flight count one hour earlier; captures immediate traffic memory.
	Lag-3h	Flight count three hours earlier; reflects short flight-bank cycles.
	Lag-6h	Flight count six hours earlier; captures longer daily repeating patterns.

Table 1. Cont.

Feature Set	Feature	Description
RollMeans	MA-3h	Three-hour rolling mean; smooths short spikes and transient fluctuations.
	MA-6h	Six-hour rolling mean; captures broader traffic trend direction.
Calendar	Hour (sin, cos)	Circular encoding of hour of day to model daily periodicity.
	Weekday/Weekend	Weekday index and Saudi weekend flag (Fri–Sat).
	Month	Month of year to capture seasonal variation.

3.6. Airport Graph Construction

Airport networks are scale-free and small-world, with a small number of highly connected hubs dominating global flows [22–24]. We construct the graph in three steps using the two datasets described in Section 3.3.

Step 1: Extract RUH direct route partners (1-hop nodes). All unique origin–destination pairs in the RUH flight dataset are extracted. Airports that appear as either the origin or destination of a flight involving RUH form the set of 1-hop neighbours $\mathcal{N}_1(\text{RUH})$. Directed edges between RUH and each neighbour d are assigned frequency-based weights:

$$w_{\text{RUH},d} = |\{f \in \mathcal{F} : \text{origin}(f) = \text{RUH}, \text{destination}(f) = d\}| \quad (6)$$

To prevent temporal leakage, RUH edge weights were computed exclusively from flights in the training partition (hours 1–3515); validation and test flights were not used in graph construction or embedding generation.

Step 2: Extend to 2-hop neighbours via OpenFlights. For each 1-hop neighbour $d \in \mathcal{N}_1(\text{RUH})$, the OpenFlights routes database is queried to retrieve all airports reachable from d in one hop. These airports form the 2-hop neighbourhood $\mathcal{N}_2(\text{RUH})$, and the corresponding directed edges (d, n) are added with unit weight (OpenFlights does not provide flight frequency). This step incorporates hub-level structural information that reflects cascading propagation patterns and global connectivity roles [5,24].

Step 3: Construct the directed weighted graph. The final graph $G = (V, E)$ consists of all nodes in $\{\text{RUH}\} \cup \mathcal{N}_1 \cup \mathcal{N}_2$ and all directed edges from Steps 1 and 2. The adjacency matrix is

$$A_{ij} = \begin{cases} w_{ij}, & \text{if a route exists between airports } i \text{ and } j, \\ 0, & \text{otherwise.} \end{cases} \quad (7)$$

The two-hop extension adds global hub structure absent from a purely local graph. To prevent temporal leakage, RUH edge weights were computed exclusively from flights in the training partition, while the OpenFlights routes database was used only to define static second-hop connectivity. Both DeepWalk and Node2Vec embeddings were then generated from this fixed graph.

Table 2 summarises the dataset partitioning and graph topology statistics. The 153,308 raw flight records yield 5022 hourly observations after aggregation, split chronologically into 3515 training, 753 validation, and 754 test hours. The two-hop aviation graph contains 1198 nodes and 11,200 directed edges, with 112 airports directly connected to RUH and 1085 s-order neighbours contributing global hub structure.

Table 2. Dataset partitioning and graph topology statistics.

Statistic	Value
Dataset	
Raw flight records	153,308
Total hourly observations	5022
Training hours (70%)	3515
Validation hours (15%)	753
Test hours (15%)	754
Airport graph	
Direct RUH neighbours (1-hop)	112
Two-hop airports	1085
Total graph nodes	1198
Total directed edges	11,200
Random-walk embedding dimension	64

3.7. Graph Embedding Generation

We evaluate two lightweight random-walk embedding methods—DeepWalk and Node2Vec—alongside centrality measures as the three graph representations under study. Both random-walk methods are preferred over deep graph neural networks because they capture local and global connectivity efficiently without labelled supervision, scale well to the 1198-node aviation graph, and produce fixed-length feature vectors directly compatible with tabular ensemble learners [10,16,17].

Both methods share the same objective: maximise the log-likelihood of observing context nodes generated through random walks:

$$\max_{\Phi} \sum_{u \in V} \log P(N_S(u) \mid \Phi(u)) \quad (8)$$

where V is the set of airports, $N_S(u)$ is the neighbourhood of node u obtained via sampling strategy S , and $\Phi : V \rightarrow \mathbb{R}^d$ assigns each airport a d -dimensional embedding.

DeepWalk [16] performs uniform random walks ($p = 1, q = 1$), treating all neighbours equally at each step. This produces embeddings that reflect the global community structure of the network.

Node2Vec [17] introduces two parameters p (return probability) and q (in-out probability) that control the trade-off between depth-first and breadth-first exploration. We set $p = 1.0$ and $q = 0.5$, biasing walks toward deeper neighbourhood exploration. Both methods use identical remaining hyperparameters (dimensions = 64, walk length = 80, walks per node = 10, window = 10, epochs = 5), ensuring a fair comparison where only the walk strategy differs. DeepWalk and Node2Vec were selected because they represent the two canonical random-walk embedding paradigms [16,17] and differ primarily in walk exploration strategy, enabling a controlled comparison of structural representations under otherwise identical conditions. More complex embedding methods (e.g., LINE, NetMF) were deliberately excluded to maintain a clean experimental design focused on the effect of walk bias rather than a broad embedding benchmark.

Centrality features provide a handcrafted baseline for graph-structural information. Four node-level statistics are computed: PageRank, in-degree centrality, out-degree centrality, and betweenness centrality. These are included as an ablation condition to assess whether simple summary statistics capture sufficient structural information, or whether distributed representations are required.

To derive a per-hour graph feature for each embedding method, we compute a mean embedding across all destination airports with RUH departures during hour t :

$$\bar{\mathbf{e}}_t = \frac{1}{|D_t|} \sum_{d \in D_t} \Phi(d) \quad (9)$$

where D_t is the set of destination airports served by RUH departures during hour t . The destination set D_t is constructed from scheduled departure times, which are published in advance and known before hour t begins; therefore, the hourly graph feature is fully observable at forecast issuance and does not rely on realised future operational outcomes. Hours with no departures receive a zero vector. Departure destinations were selected to provide a consistent representation of RUH's outbound network context. Although separate arrival–origin embeddings could provide complementary information, incorporating both arrival and departure representations would substantially increase feature dimensionality and introduce an additional design variable, which falls outside the scope of the present study. Combined arrival–departure aggregation is identified as a direction for future work in Section 5.10. Forecasts are generated at the end of hour t using information available up to and including hour t , with the target defined as traffic during hour $t + 1$. The feature vector for a graph-augmented model concatenates temporal features with the hourly mean embedding:

$$\mathbf{x}_t = [\mathbf{X}_t \parallel \bar{\mathbf{e}}_t] \quad (10)$$

where $\mathbf{X}_t$ is the 10-dimensional temporal feature vector and $\bar{\mathbf{e}}_t \in \mathbb{R}^{64}$, yielding a 74-dimensional input per hour.

3.8. Prediction Models

We organise models into three groups: strong temporal baselines, proposed graph-augmented models, and reference baselines.

(1) Strong Temporal Baselines. Based on prior work demonstrating their strong and consistent performance in aviation prediction tasks [1–3], three tree-based ensemble families are selected as the primary comparison group: Random Forest (RF) [35], XGBoost [36], and LightGBM [37]. Each is trained exclusively on the 10-dimensional temporal feature set $\mathbf{X}_t$. RF constructs multiple independent decision trees on bootstrap samples and aggregates outputs, reducing variance through ensemble averaging. XGBoost builds trees sequentially under a gradient boosting framework, minimising regularised residuals from the preceding model. LightGBM adopts a leaf-wise growth strategy with histogram-based split finding, offering fast training and competitive accuracy on tabular datasets.

(2) Proposed: Graph-Augmented Models. We augment each strong temporal baseline with a 64-dimensional mean graph embedding $\bar{\mathbf{e}}_t$, independently for each embedding method, yielding six graph-augmented models: RF+DW, XGBoost+DW, LightGBM+DW (DeepWalk family) and RF+N2V, XGBoost+N2V, LightGBM+N2V (Node2Vec family). Each is trained on the 74-dimensional feature vector $\mathbf{x}_t = [\mathbf{X}_t \parallel \bar{\mathbf{e}}_t]$. The two embedding families are evaluated under identical conditions—same base models, same hyperparameters, same seeds—so that any performance difference is attributable solely to the walk strategy ($p = q = 1$ for DeepWalk vs. $p = 1, q = 0.5$ for Node2Vec).

(3) Reference Baselines. Six additional models provide broader context. A Long Short-Term Memory (LSTM) network [38] processes a 24 h sliding window of temporal signals through stacked gating layers, representing the best achievable performance from a standard deep sequence model without graph augmentation [4,39]. GRU and deeper LSTM variants were not included as separate baselines; the two-layer LSTM is selected to represent the class of deep sequence models. Empirically, the two-layer LSTM already un-

derperforms the temporal-only RF baseline by a substantial margin (MAE 2.668 vs. 2.028), consistent with prior findings that deep sequential architectures require substantially larger training sets to match well-engineered ensemble methods [3,4]. TCN and Transformer architectures generally require larger training sets than recurrent models and are therefore unlikely to provide a clear advantage under the present data constraints (3515 training hours). Including them would substantially broaden the scope of the study without directly addressing the central comparison between graph representation classes and ensemble learners. Evaluating graph augmentation in conjunction with TCN or Transformer backbones remains an interesting direction for future work. An MLP with two fully connected hidden layers (128 and 64 neurons, ReLU activations, dropout) provides a non-sequential neural network reference. SVR (RBF kernel) and Ridge Regression represent classical regression baselines trained on temporal features [20]. ARIMA(p, d, q) with orders selected via AIC [18,19] and a naive Persistence model ($\hat{y}_{t+1} = y_t$) provide lower-bound statistical references [40]. All reference baselines are trained on temporal features only.

3.9. Model Formulation and Learning Objective

For RF, the prediction averages B independently trained regression trees:

$$\hat{y}_{t+1} = \frac{1}{B} \sum_{b=1}^B T_b(\mathbf{x}_t) \quad (11)$$

For XGBoost and LightGBM, the prediction is an additive ensemble of M weak learners:

$$\hat{y}_{t+1} = \sum_{m=1}^M f_m(\mathbf{x}_t) \quad (12)$$

where each f_m is a regression tree added to minimise a regularised loss:

$$\mathcal{L} = \sum_{t=1}^n (y_{t+1} - \hat{y}_{t+1})^2 + \sum_{m=1}^M \Omega(f_m) \quad (13)$$

where $\Omega(f_m)$ penalises overly complex trees to balance accuracy and generalisation.

4. Experimental Setup

4.1. Environment and Data Partitioning

We ran all experiments in Python 3.12 on Google Colab. Deep learning models (LSTM, MLP) used a Tesla T4 GPU via PyTorch (v2.11.0); all tree-based models (RF, XGBoost, LightGBM), graph construction, and random-walk embedding generation ran on CPU using scikit-learn (v1.6.1), XGBoost (v3.3.0), LightGBM (v4.6.0), and NetworkX (v3.6.1). Data processing used pandas (v2.2.2), NumPy (v1.26.4), and statsmodels (v0.14.6). Reported training and inference times reflect CPU execution.

We divided the dataset into three temporal segments following a strict chronological walk-forward strategy to prevent look-ahead bias: 70% training, 15% validation (for hyperparameter tuning and early stopping), and 15% testing. All stochastic experiments were repeated across five independent random seeds (42, 123, 456, 789, 2024), and mean $\pm$ standard deviation are reported. Deterministic models (SVR, Ridge, Persistence, ARIMA) are reported as single values.

4.2. Model Hyperparameters

We selected hyperparameters via grid search on the validation subset. Table 3 summarises all configurations.

Table 3. Hyperparameter configuration of all evaluated models.

Model	Parameter	Value
RF	n_estimators	300
	max_depth	10
	min_samples_leaf	2
XGBoost	learning rate	0.05
	max_depth	6
	n_estimators	500
	subsample/colsample	0.8
	λ/α	1/1
	early stopping	20 rounds
LightGBM	learning rate	0.05
	max_depth	6
	n_estimators	500
	subsample/colsample	0.8
	λ/α	1/1
	early stopping	20 rounds
LSTM	hidden size/layers	64/2
	dropout	0.2
	sequence length	24 h
	learning rate/batch	0.001/64
MLP	hidden layers	(128, 64)
	dropout	0.2
	learning rate/batch	0.001/64
SVR	kernel/C/ ϵ	RBF/10/0.1
Ridge	α (CV)	{0.01, 0.1, 1, 10}
DeepWalk	dimension/walk length	64/80
	walks per node/window	10/10
	p/q	1.0/1.0
	epochs/learning rate	5/0.025
Node2Vec	dimension/walk length	64/80
	walks per node/window	10/10
	p/q	1.0/0.5
	epochs/learning rate	5/0.025

4.3. Evaluation Metrics

We assess model performance using three complementary metrics [40–42]:

$$\text{MAE} = \frac{1}{n} \sum_{t=1}^n |y_{t+1} - \hat{y}_{t+1}| \quad (14)$$

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{t=1}^n (y_{t+1} - \hat{y}_{t+1})^2} \quad (15)$$

$$\text{sMAPE} = \frac{100}{n} \sum_{t=1}^n \frac{|y_{t+1} - \hat{y}_{t+1}|}{(|y_{t+1}| + |\hat{y}_{t+1}|)/2} \quad (16)$$

MAE and RMSE serve as primary indicators (both expressed in aircraft movements per hour); sMAPE provides a scale-normalised complement. Because sMAPE is sensitive to near-zero values [40], MAE and RMSE are treated as the principal performance measures.

To assess statistical significance, the Diebold–Mariano (DM) test [43] is applied pairwise using squared-error loss differences. Two sets of comparisons are reported: (i) each

graph-augmented model vs. all nine baseline models; (ii) a direct comparison between the DeepWalk and Node2Vec variants within each ensemble family. A two-sided test with $\alpha = 0.05$ is used throughout. The long-run variance is estimated using the Newey–West HAC estimator with a Bartlett kernel and bandwidth $M = \lfloor T^{1/3} \rfloor = 9$, where $T = 754$ is the test-set length.

5. Results and Discussion

5.1. Overall Model Performance

Table 4 shows the performance of all models on the held-out test set. Models are grouped as reference baselines, strong temporal baselines, and proposed graph-augmented models, sorted by MAE within each group. Stochastic models report mean $\pm$ std across five seeds; deterministic models report single values.

Table 4. Comparative performance on the RUH test set. Lower values indicate better performance. Models are grouped as: reference baselines (top), strong temporal baselines (middle), and proposed graph-augmented models (bottom). The best overall result is highlighted in bold. Improvement column reports MAE reduction relative to the corresponding temporal-only baseline (RF, XGBoost, or LightGBM). Dashes (—) indicate that the improvement metric is not applicable for models without graph augmentation.

Model	MAE	RMSE	sMAPE (%)	Improv. vs. Base
Reference baselines (temporal features only)				
Persistence	4.4138	5.7662	15.61	—
ARIMA (3,0,1)	5.7775	6.9440	19.48	—
Ridge	3.4803	4.7076	12.09	—
SVR	2.4107	3.2917	8.28	—
MLP	3.1518 $\pm$ 0.09	4.0211 $\pm$ 0.11	10.75 $\pm$ 0.30	—
LSTM	2.6679 $\pm$ 0.15	3.5030 $\pm$ 0.20	9.08 $\pm$ 0.63	—
Strong temporal baselines (temporal features only)				
LightGBM	2.4354 $\pm$ 0.02	3.3098 $\pm$ 0.02	8.35 $\pm$ 0.09	—
XGBoost	2.3815 $\pm$ 0.02	3.2703 $\pm$ 0.02	8.10 $\pm$ 0.07	—
RF	2.0283 $\pm$ 0.00	2.8884 $\pm$ 0.01	6.93 $\pm$ 0.02	—
Proposed: graph-augmented models (temporal + DeepWalk)				
LightGBM+DW	2.0427 $\pm$ 0.01	2.9239 $\pm$ 0.02	6.90 $\pm$ 0.04	−16.1%
XGBoost+DW	2.0116 $\pm$ 0.03	2.9304 $\pm$ 0.04	6.82 $\pm$ 0.08	−15.5%
RF+DW	1.8098 $\pm$ 0.01	2.4811 $\pm$ 0.02	6.17 $\pm$ 0.02	−10.8%
Proposed: graph-augmented models (temporal + Node2Vec)				
LightGBM+N2V	2.0193 $\pm$ 0.02	2.8669 $\pm$ 0.04	6.88 $\pm$ 0.08	−17.1%
XGBoost+N2V	1.9959 $\pm$ 0.01	2.9348 $\pm$ 0.04	6.79 $\pm$ 0.05	−16.2%
RF+N2V	1.8371 $\pm$ 0.02	2.5077 $\pm$ 0.02	6.30 $\pm$ 0.05	−9.4%

Random-walk graph embeddings improve all three ensemble families consistently across both DeepWalk and Node2Vec variants. RF+DeepWalk achieved the lowest observed MAE of 1.810 (RMSE = 2.481, sMAPE = 6.17%). RF+Node2Vec is closely competitive (MAE = 1.837); the pairwise difference between DeepWalk and Node2Vec does not reach statistical significance within any ensemble family (see Section 5.2). Graph augmentation reduces MAE by approximately 9–11% for the RF family and 15–17% for the XGBoost and LightGBM families.

Among temporal-only models, RF (MAE = 2.028) is the strongest, outperforming XGBoost (2.382) and LightGBM (2.435). RF's bootstrap aggregation is particularly effective on the 3515 training hours available here [4,35]. Despite regularisation, gradient boosting is more susceptible to overfitting at this scale; RF's independent tree construction yields a more stable optimisation landscape. LSTM (MAE = 2.668) and MLP (MAE = 3.152) un-

derperform all three tree-based baselines; deep learning architectures require substantially larger datasets to match well-engineered ensemble methods [3,4]. SVR (MAE = 2.411) and Ridge (MAE = 3.480) trail the ensemble learners. ARIMA (MAE = 5.778) and Persistence (MAE = 4.414) reflect the non-stationary nature of the task.

5.2. Statistical Significance

Table 5 reports the Diebold–Mariano test results comparing RF+DW against all nine baselines [43]. A negative DM statistic indicates that RF+DW has lower squared error than the comparison model. Table 6 reports the direct comparison between the DeepWalk and Node2Vec variants within each ensemble family.

Table 5. Diebold–Mariano test: RF+DW vs. all baselines. Negative DM statistic favours RF+DW. ✓ = significant at $p < 0.05$.

Baseline	DM Stat	<i>p</i> -Value	Sig.
vs. strong temporal baselines			
RF	−2.942	0.003	✓
XGBoost	−4.806	<0.001	✓
LightGBM	−5.560	<0.001	✓
vs. reference baselines			
SVR	−4.371	<0.001	✓
LSTM	−4.933	<0.001	✓
MLP	−9.613	<0.001	✓
Ridge	−8.360	<0.001	✓
Persistence	−9.939	<0.001	✓
ARIMA	−14.933	<0.001	✓

Table 6. Diebold–Mariano test: DeepWalk vs. Node2Vec within each ensemble family. Negative DM statistic favours the DeepWalk variant. Dashes (—) indicate results that did not reach statistical significance ($p \geq 0.05$).

Comparison	DM Stat	<i>p</i> -Value	Sig.
RF+DW vs. RF+N2V	−0.468	0.640	—
XGBoost+DW vs. XGBoost+N2V	−0.139	0.890	—
LightGBM+DW vs. LightGBM+N2V	+1.043	0.297	—

RF+DW significantly outperforms all nine baselines at $\alpha = 0.05$, including the temporal-only RF ($p = 0.003$)—the most challenging comparison since RF already captures the dominant periodic and autoregressive patterns. Random-walk graph embeddings thus provide predictive signal that temporal features alone cannot recover. Similarly, RF+N2V significantly outperforms all nine baselines ($p = 0.016$ vs. RF), confirming that the observed gains are not specific to a single embedding method. The largest DM statistics are against ARIMA (−14.933) and Persistence (−9.939), reflecting the non-stationary nature of the task.

The direct embedding comparison (Table 6) shows that neither DeepWalk nor Node2Vec consistently outperforms the other—none of the three pairwise comparisons reaches statistical significance (RF: $p = 0.640$; XGBoost: $p = 0.890$; LightGBM: $p = 0.297$). Taken together, the evidence supports a finding about the class of representation rather than a specific algorithm: random-walk embeddings consistently outperform temporal-only baselines across all three ensemble families, and the benefit of graph augmentation is robust across both walk strategies regardless of the choice of walk parameterisation.

5.3. Feature Ablation Study

Table 7 reports the ablation results across 11 feature subsets for XGBoost and RF (mean $\pm$ std across five seeds), isolating the contribution of each feature group. LightGBM exhibited a pattern closely matching XGBoost across all subsets and is omitted for conciseness. Table 8 summarises the key progression from temporal-only to graph-augmented representations for RF.

Table 7. Ablation study: mean MAE and RMSE ($\pm$ std across five seeds) for XGBoost and RF. Lower is better. Feature sets are ordered from simplest to most complete. Best results (graph-augmented models) are highlighted in bold.

Feature Set	XGBoost (Mean $\pm$ Std)		RF (Mean $\pm$ Std)	
	MAE	RMSE	MAE	RMSE
Individual feature groups				
Lags only	3.368 $\pm$ 0.011	4.585 $\pm$ 0.027	3.223 $\pm$ 0.010	4.530 $\pm$ 0.013
RollMeans only	4.033 $\pm$ 0.069	5.336 $\pm$ 0.064	3.407 $\pm$ 0.001	4.707 $\pm$ 0.003
Calendar only	2.360 $\pm$ 0.025	2.938 $\pm$ 0.017	1.820 $\pm$ 0.011	2.318 $\pm$ 0.014
Centrality only	4.232 $\pm$ 0.023	5.849 $\pm$ 0.038	4.131 $\pm$ 0.007	5.341 $\pm$ 0.008
DeepWalk only	2.643 $\pm$ 0.016	3.609 $\pm$ 0.023	2.276 $\pm$ 0.009	3.118 $\pm$ 0.022
Node2Vec only	2.595 $\pm$ 0.023	3.561 $\pm$ 0.055	2.253 $\pm$ 0.013	2.990 $\pm$ 0.014
Combined feature sets				
Lags + RollMeans	3.058 $\pm$ 0.059	4.334 $\pm$ 0.065	2.898 $\pm$ 0.006	4.174 $\pm$ 0.007
AllTemporal	2.382 $\pm$ 0.019	3.270 $\pm$ 0.025	2.028 $\pm$ 0.004	2.888 $\pm$ 0.016
Centrality + AllTemporal	2.177 $\pm$ 0.016	3.012 $\pm$ 0.038	2.022 $\pm$ 0.008	2.762 $\pm$ 0.017
DeepWalk + AllTemporal	2.012 $\pm$ 0.030	2.930 $\pm$ 0.050	1.810 $\pm$ 0.006	2.481 $\pm$ 0.017
Node2Vec + AllTemporal	1.996 $\pm$ 0.016	2.935 $\pm$ 0.045	1.837 $\pm$ 0.017	2.508 $\pm$ 0.027

Table 8. RF MAE progression across graph representation types. Centrality provides negligible gain; random-walk embeddings provide substantial improvement. DM p -value is against temporal-only RF. Best result highlighted in bold. Dashes (—) indicate that the metric is not applicable for the reference baseline (AllTemporal).

RF Feature Set	MAE	RMSE	Δ MAE vs. Temporal	DM p -Value
AllTemporal	2.028	2.888	—	—
Centrality + AllTemporal	2.022	2.762	−0.3%	n.s. *
Node2Vec + AllTemporal	1.837	2.508	−9.4%	0.016
DeepWalk + AllTemporal	1.810	2.481	−10.8%	0.003

* n.s. = not significant at $p < 0.05$.

Table 8 captures the central finding of the ablation: centrality features provide negligible improvement over temporal-only RF (Δ MAE = −0.3%), whereas random-walk embeddings yield substantial gains of approximately 9–11%. DeepWalk embeddings alone achieve MAE of 2.276 (RF) and 2.643 (XGBoost), while Node2Vec embeddings alone achieve MAE of 2.253 (RF) and 2.595 (XGBoost), demonstrating that graph structure independently encodes predictive signal. RMSE trends throughout Table 7 are broadly consistent with MAE. The Calendar-only RF baseline is highly competitive (MAE = 1.820); DeepWalk+AllTemporal achieves the lowest MAE overall (1.810, $\Delta \approx 0.5\%$ over Calendar-only). Notably, adding lag and rolling-mean features to calendar variables (AllTemporal RF, MAE = 2.028) leads to worse performance than Calendar-only, indicating that the richer temporal feature set does not automatically translate into better predictive accuracy in this setting. DeepWalk+AllTemporal largely recovers this loss while retaining the full temporal feature set. Taken together, these results suggest that recurring temporal schedules explain a substantial fraction of RUH traffic variability, while graph embeddings provide

complementary structural information that yields the lowest overall MAE. The absolute gain over Calendar-only is modest ($\Delta \approx 0.5\%$); the primary contribution of the embeddings is to offset the performance degradation caused by adding temporal features in this setting. DeepWalk performs better than Node2Vec as a standalone representation, but both improve substantially when combined with temporal features; the main gain comes from random-walk graph representation, while the walk strategy has a modest model-dependent effect.

Calendar features are the most informative single temporal group. For RF, calendar features alone achieve MAE = 1.820, outperforming lags (3.223) and rolling means (3.407). This reflects RUH's highly regular scheduling: hour of day and day of week largely determine traffic volume [6]. However, calendar features alone lack historical memory and may be less robust under irregular traffic conditions, motivating the inclusion of lag, rolling mean, and graph-based representations. For XGBoost, calendar features also dominate (2.360), though the gap is less pronounced.

RF degrades slightly when moving from Calendar only to AllTemporal (1.820 $\rightarrow$ 2.028). Adding five correlated lag and rolling mean features dilutes the probability of selecting calendar features at individual splits [35]. Adding 64 graph embedding dimensions reverses this trend: the embeddings expand the feature space in an uncorrelated direction, raising the fraction of informative features available at each split and yielding the best results across both embedding methods.

Graph centrality features provide negligible benefit over AllTemporal for RF (2.028 $\rightarrow$ 2.022, $\Delta = -0.3\%$) and moderate but substantially smaller benefit for XGBoost (2.382 $\rightarrow$ 2.177, $\Delta \approx -9\%$) compared with random-walk embeddings ($\Delta = -9$ to -17% across families). The gains from centrality are therefore smaller in magnitude and less consistent across model families than those from distributed embeddings. This shows that handcrafted node-level statistics—PageRank, degree centrality, betweenness—are insufficient to capture the relational structure of the aviation network. Each centrality measure collapses the full neighbourhood structure of a node into a single scalar, discarding the distributed positional context that distinguishes one hub from another at a fine-grained level. Distributed random-walk embeddings, by contrast, encode neighbourhood context across multiple hops into a 64-dimensional vector, preserving the relational geometry of the network and producing representations that are both richer and more complementary to temporal features [16,17].

5.4. Embedding Dimension Sensitivity

Table 9 reports the effect of embedding dimension on forecasting performance for RF+DW and RF+N2V (mean $\pm$ std across five seeds). Each model is trained using only the graph embeddings at the specified dimension combined with the full temporal feature set. Performance generally improves as dimension increases from 16 to 128 for both methods. RF+DW outperforms RF+N2V at dimensions 32, 64, and 128, while RF+N2V achieves lower MAE at dimension 16; the differences are small and within approximately one standard deviation. Performance variations remain modest across the tested dimensions, confirming that the results are stable and not an artifact of a single lucky hyperparameter setting. The relative ranking of representation classes remains unchanged across all tested dimensions. A fixed dimension of 64 was retained throughout the main experiments to maintain a consistent and comparable experimental setting; the sensitivity results demonstrate that this choice is conservative rather than tuned to maximise reported performance.

Table 9. Embedding dimension sensitivity analysis for RF+DW and RF+N2V. MAE and RMSE are mean $\pm$ std across five seeds. Lower is better. Best result per method highlighted in bold.

Dim	Method	MAE	RMSE
16	RF+DW	1.822 ± 0.004	2.449 ± 0.017
	RF+N2V	1.799 ± 0.005	2.394 ± 0.011
32	RF+DW	1.824 ± 0.007	2.448 ± 0.017
	RF+N2V	1.829 ± 0.007	2.504 ± 0.021
64	RF+DW	1.781 ± 0.006	2.431 ± 0.030
	RF+N2V	1.821 ± 0.008	2.453 ± 0.021
128	RF+DW	1.778 ± 0.010	2.402 ± 0.025
	RF+N2V	1.781 ± 0.011	2.447 ± 0.026

5.5. SHAP Interpretability

Figure 2 presents SHAP mean absolute feature importance for RF+DW, the best-performing model, computed using TreeExplainer [44] on the seed-42 model instance on the test set.

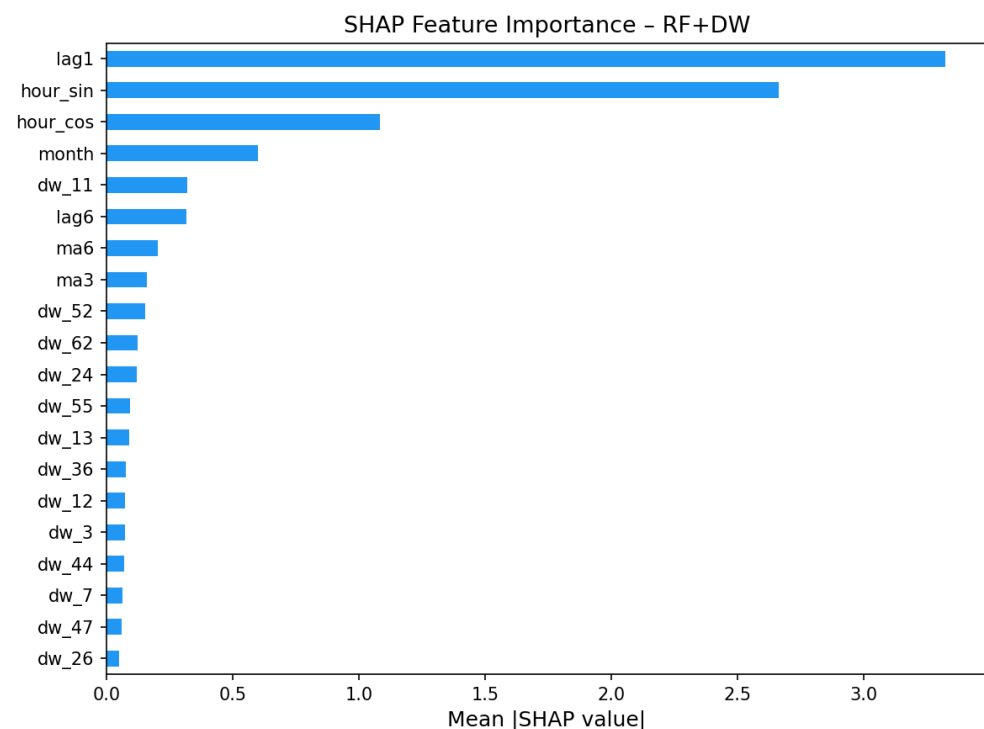


Figure 2. SHAP feature importance for RF+DW (top 20 features, mean $|\text{SHAP value}|$). The dominant predictors are the 1 h lag and time-of-day encodings, followed by month and the DeepWalk dimension dw_{11} . Thirteen of the top 20 features are DeepWalk dimensions, confirming that distributed graph representations contribute independent predictive signal beyond temporal features.

lag_1 is the most influential feature, reflecting the strong autoregressive structure of RUH's traffic. $\text{hour}_{\sin}$ and $\text{hour}_{\cos}$ follow, capturing the daily periodicity of scheduled operations. month ranks fourth, encoding seasonal cycles and pilgrimage-driven demand typical of Gulf aviation [6,7]. dw_{11} ranks fifth—directly after month and ahead of lag_6 , ma_6 , ma_3 , and all remaining temporal features—and 13 of the top 20 features are DeepWalk dimensions. This provides mechanistic evidence that distributed graph representations capture network-structural signal that is absent from, and complementary to, the temporal feature set.

5.6. Residual Distribution and Model Behaviour

Figure 3 presents the residual error distributions for the two best-performing graph-augmented variants: RF+DW and RF+N2V.

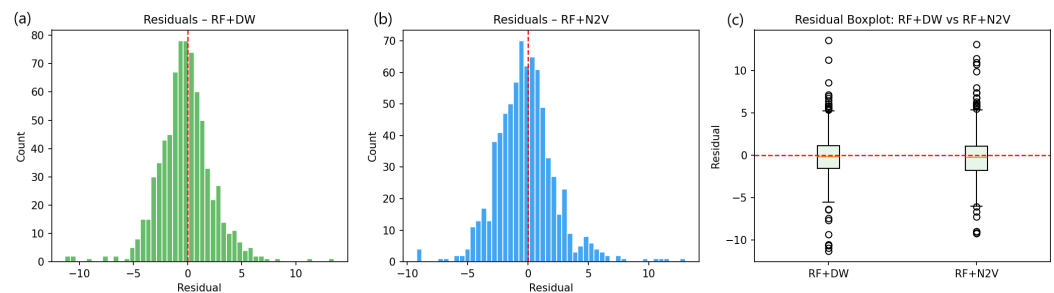


Figure 3. Residual distributions for the two best graph-augmented Random Forest variants. (a) RF+DW histogram; (b) RF+N2V histogram; (c) comparative boxplot. Both models show residuals centred near zero; RF+DW exhibits a marginally tighter distribution with slightly fewer extreme outliers, consistent with its lower observed MAE.

Both histograms are concentrated near zero, indicating largely unbiased predictions. RF+DW and RF+N2V produce visually similar distributions, reflecting the modest magnitude of the actual performance gap. Both benefit from averaging 300 independent trees, which smooths the over- and under-estimation errors typical of sequential boosting and yields a tight, near-symmetric residual spread (−10 to +10) [21,42]. The modest edge of RF+DW in the tails is consistent with its slightly lower MAE (1.810 vs. 1.837) but should not be interpreted as a systematic advantage; it reflects the stochastic sampling variability inherent to random-walk embedding generation [4].

5.7. Terminal-Level Forecasting Performance

Table 10 evaluates RF+DW across all five terminals of RUH.

Table 10. Terminal-level forecasting performance of RF+DW across all five terminals.

Terminal	MAE	RMSE	sMAPE (%)	Hours
Terminal 1	0.307	0.518	6.14	743
Terminal 2	0.069	0.105	5.61	356
Terminal 3	0.278	0.401	6.20	743
Terminal 4	0.245	0.365	6.12	736
Terminal 5	0.946	1.277	6.16	754

sMAPE is consistent across all five terminals (5.61–6.20%), showing that RF+DW’s relative accuracy is stable regardless of terminal type or traffic volume. Terminal 2 achieves the lowest absolute error (MAE = 0.069), reflecting its lower and more regular traffic volume (356 evaluated hours, approximately half the others). Terminal 5—the busiest international terminal—exhibits the highest absolute error (MAE = 0.946, RMSE = 1.277), attributable to greater traffic volume, irregular airline scheduling, and higher sensitivity to external disruptions [5,19]. Despite this, its sMAPE (6.16%) remains within the same narrow band as all other terminals.

5.8. Temporal Robustness Analysis

Figure 4 presents the monthly walk-forward evaluation across five consecutive walk-forward evaluation months (June–October 2025), where each month’s test set is evaluated on a model trained on all preceding data.

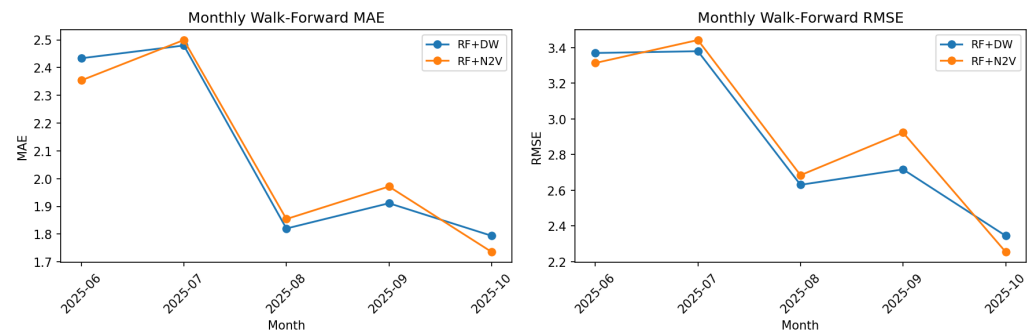


Figure 4. Monthly walk-forward MAE (left) and RMSE (right) for RF+DW and RF+N2V. Both random-walk-augmented variants track each other closely across all five walk-forward evaluation months, with neither method consistently outperforming the other, consistent with the non-significant pairwise Diebold–Mariano test. Both models show a general improving trend as the training window expands, with a temporary performance degradation in July 2025 attributable to seasonal demand surge.

RF+DW and RF+N2V track each other closely across all five walk-forward evaluation months, consistent with the non-significant pairwise Diebold–Mariano test between them and the modest magnitude of the overall performance gap. Both models improve as the training window grows, with RF+DW reaching MAE ≈ 1.79 by October 2025. A temporary degradation appears in July 2025: RUH handles a large share of Saudi Arabia’s pilgrimage traffic, and July coincides with the annual Hajj season—one of the largest concentrated traffic surges in global aviation [6,7]. The model has not yet seen enough data from this high-demand period to calibrate accurately. Recovery from August onward shows that both variants adapt as more seasonal data accumulates, and the embedding-family gap remains stable throughout, underscoring that the benefit derives from the random-walk representation class rather than from any specific walk strategy.

5.9. Computational Cost and Training Time

Table 11 reports representative training and inference times measured using seed 42.

Table 11. Training and inference times for all evaluated models (measured on seed 42, CPU-only). Inference time is the per-sample prediction latency in milliseconds. Embedding generation is a one-time offline cost reported separately.

Model	Train (s)	Infer (ms/Sample)
Temporal-only baselines (10 features)		
RF	0.79	0.116
XGBoost	0.20	0.004
LightGBM	0.17	0.008
Graph-augmented: Node2Vec (74 features)		
RF+N2V	10.73	0.129
XGBoost+N2V	3.42	0.003
LightGBM+N2V	0.92	0.016
Graph-augmented: DeepWalk (74 features)		
RF+DW	10.62	0.132
XGBoost+DW	2.75	0.004
LightGBM+DW	0.90	0.010

Adding 64 graph embedding dimensions increases RF training time from 0.79 s to ≈ 10.6 s (a $13\times$ factor attributable to the expanded split-evaluation space across 300 trees); XGBoost+DW shows a similarly large increase (0.20 s to 2.75 s, $\approx 14\times$), while LightGBM shows a proportionally smaller increase due to its leaf-wise growth strategy. Inference

latency is negligible for all models (<0.15 ms per sample), confirming real-time applicability. Embedding generation is a one-time offline cost: Node2Vec takes ≈ 18 s and DeepWalk ≈ 17 s on the same CPU. End-to-end pipeline execution—including preprocessing, graph construction, embedding generation, and model training across all evaluated models and five seeds—completes in under three minutes, confirming the framework's suitability for operational deployment without specialised hardware.

5.10. Limitations

The proposed framework has several limitations to acknowledge. First, the evaluation is conducted on a single airport (RUH); generalisation to airports with different operational profiles—hub-and-spoke, regional, or highly seasonal—has not been empirically verified. Second, the airport graph is constructed from historical flight connections and represents a static approximation that does not account for seasonal route additions, route suspensions, or other network changes that occur over time; periodic recomputation of embeddings could better reflect evolving network topology and is discussed as a future direction below. Third, external demand drivers such as fuel pricing, major events, or pandemic-related disruptions were not incorporated as features. Fourth, while the choice between DeepWalk and Node2Vec did not reach statistical significance in any ensemble family, additional random-walk variants (e.g., LINE, HARP) and higher-order graph neural network embeddings were not included in the comparison. Fifth, the graph augmentation was applied exclusively to tree-based ensemble learners; whether similar accuracy gains transfer to deep learning architectures such as LSTM or GNN-based models remains an open question for future investigation. Finally, the model is tested over five months; longer evaluation horizons across multiple years would provide stronger evidence of temporal robustness. Additionally, the per-hour graph feature is derived by averaging destination embeddings across all departures in hour t (Equation (9)); this simple aggregation discards distributional information about the departure set—for example, whether traffic is concentrated on a single frequently-served route or spread across many destinations—and weighted or distribution-aware aggregation strategies remain an open direction. Future work may address these limitations by incorporating dynamic graph representations, real-time operational feeds, deep learning graph augmentation, and multi-airport evaluation.

6. Conclusions

This study presents the first systematic evaluation of lightweight graph representations for single-airport traffic forecasting using temporal ensemble learners. We proposed a graph-augmented forecasting framework that augments RF, XGBoost, and LightGBM with three structural representations (centrality measures, DeepWalk, and Node2Vec) derived from a two-hop global aviation graph combining RUH operational data with the OpenFlights routes database.

The central finding is that the class of random-walk distributed representations—not any specific algorithm—drives accuracy gains. Both embedding families improve all three model families numerically, with the strongest statistically validated gains observed for the RF family: RF+DW achieved the lowest observed MAE (1.810, RMSE = 2.481, sMAPE = 6.17%) and RF+N2V achieved a competitive result (MAE = 1.837); the choice between DeepWalk and Node2Vec did not reach statistical significance in any ensemble family ($p \geq 0.297$), while Diebold–Mariano tests confirm both are significantly better than all baseline models ($p \leq 0.016$). Graph augmentation reduced MAE by approximately 9–17% across all three ensemble families, demonstrating that the benefit generalises beyond a single architecture. By contrast, graph centrality features yielded negligible improvement for RF ($\Delta\text{MAE} = -0.3\%$) and moderate but substantially smaller improvement for XGBoost

($\sim -9\%$) compared with random-walk embeddings, confirming that distributed structural representations carry far more forecasting signal than scalar graph statistics.

SHAP analysis revealed that 13 of the top 20 most influential features are DeepWalk embedding dimensions, with the leading dimension (dw_11) ranked fifth overall—directly confirming the contribution of learned network-structural context. Terminal-level evaluation showed consistent relative accuracy across all five RUH terminals (sMAPE 5.61–6.20%), and monthly walk-forward analysis confirmed stable generalisation over five evaluation months, with a predictable July dip attributable to the Hajj traffic surge.

The framework requires no GPU hardware, no real-time data feeds, and completes end-to-end training in under three minutes, requiring only the target airport’s operational records and publicly available route topology—making it a practical candidate for operational integration into airport management and decision-support systems, subject to validation across additional airports and operational conditions. Future work may incorporate real-time flight status feeds, weather covariates, dynamic graph representations, additional embedding architectures, and multi-airport validation. An explicit binary indicator for the Hajj and Umrah pilgrimage periods, which generate concentrated demand surges at RUH that a monthly calendar feature alone cannot fully capture, represents a near-term practical extension. The current framework uses a static two-hop graph constructed from historical route data; periodic graph updates to reflect route network changes—for example, seasonal schedule revisions—represent a natural extension that could further improve robustness over longer deployment horizons.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Data Availability Statement: The primary dataset used in this study is publicly available on Kaggle: <https://www.kaggle.com/datasets/mohammedalsubaie/king-khalid-international-airport-flights-dataset> (accessed on 1 May 2025). The OpenFlights global routes database used for graph construction is publicly available at: <https://openflights.org/data> (accessed on 1 May 2025). The code and scripts used to reproduce all experiments are publicly available at: <https://github.com/a7madtalal123/Lightweight-Graph-Embedding-Augmentation-for-Airport-Traffic-Forecasting> (accessed on 1 May 2025).

Acknowledgments: During the preparation of this manuscript, the author used AI language tools for the purposes of language editing, sentence restructuring, and improving the clarity of the written text. These tools were not used to generate scientific content, data analysis, or conclusions. The author has reviewed and edited all AI-assisted output and takes full responsibility for the content of this publication.

Conflicts of Interest: The author declares no conflicts of interest.

References

1. Hatipoğlu, İ.; Tosun, Ö.; Tosun, N. Flight delay prediction based with machine learning. *LogForum* **2022**, *18*, 97–109. [\[CrossRef\]](#)
2. Dai, M. A hybrid machine learning-based model for predicting flight delay through aviation big data. *Sci. Rep.* **2024**, *14*, 4728. [\[CrossRef\]](#) [\[PubMed\]](#)
3. Jafari, N.; Lewison, M. Forecasting air passenger traffic and market share using deep neural networks with multiple inputs and outputs. *Front. Artif. Intell.* **2024**, *7*, 1429341. [\[CrossRef\]](#) [\[PubMed\]](#)
4. Pinto Neto, E.C.; Baum, D.M.; Almeida, J.R.d.; Camargo, J.B.; Cugnasca, P.S. Deep Learning in Air Traffic Management (ATM): A Survey on Applications, Opportunities, and Open Challenges. *Aerospace* **2023**, *10*, 358. [\[CrossRef\]](#)
5. Fleurquin, P.; Ramasco, J.J.; Eguíluz, V.M. Systemic Delay Propagation in the US Airport Network. *Sci. Rep.* **2013**, *3*, 1159. [\[CrossRef\]](#) [\[PubMed\]](#)
6. Reynolds-Feighan, A.; Zou, L.; Yu, C. Seasonality in European and North American Air Transport Markets: Network Structures and Implications for Airline Performance and Recovery. *Transp. J.* **2022**, *61*, 284–304. [\[CrossRef\]](#)

7. Alfarhood, M.; Alotaibi, R.; Abdulrahim, B.; Einieh, A.; Almousa, M.; Alkhanifer, A. Predicting Flight Delays with Machine Learning: A Case Study from Saudi Arabian Airlines. *Int. J. Aerosp. Eng.* **2024**, *2024*, 3385463. [\[CrossRef\]](#)
8. Zheng, H.; Wang, Z.; Zheng, C.; Wang, Y.; Fan, X.; Cong, W.; Hu, M. A graph multi-attention network for predicting airport delays. *Transp. Res. Part E Logist. Transp. Rev.* **2024**, *181*, 103375. [\[CrossRef\]](#)
9. Franco, J.L.; Machado Neto, M.V.; Verri, F.A.N.; Amancio, D.R. Graph machine learning for flight delay prediction due to holding maneuver. *arXiv* **2025**, arXiv:2502.04233.
10. Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; Yu, P.S. A Comprehensive Survey on Graph Neural Networks. *IEEE Trans. Neural Networks Learn. Syst.* **2021**, *32*, 4–24. [\[CrossRef\]](#) [\[PubMed\]](#)
11. Qian, P.; Zhang, J.; Zhang, H.; Li, X.; Ouyang, J. ST-GTNet: A Spatio-Temporal Graph Attention Network for Dynamic Airport Capacity Prediction. *Aerospace* **2025**, *12*, 811. [\[CrossRef\]](#)
12. Yan, Z.; Yang, H.; Li, F.; Lin, Y. A Deep Learning Approach for Short-Term Airport Traffic Flow Prediction. *Aerospace* **2022**, *9*, 11. [\[CrossRef\]](#)
13. Huang, Y.; Yang, H.; Yan, Z. A Dynamic Multi-Graph Convolutional Spatial-Temporal Network for Airport Arrival Flow Prediction. *Aerospace* **2025**, *12*, 395. [\[CrossRef\]](#)
14. Jiang, W.; Luo, J. Graph Neural Network for Traffic Forecasting: A Survey. *Expert Syst. Appl.* **2022**, *207*, 117921. [\[CrossRef\]](#)
15. Cui, Z.; Henrickson, K.; Ke, R.; Wang, Y. Traffic Graph Convolutional Recurrent Neural Network: A Deep Learning Framework for Network-Scale Traffic Learning and Forecasting. *IEEE Trans. Intell. Transp. Syst.* **2020**, *21*, 4883–4894. [\[CrossRef\]](#)
16. Perozzi, B.; Al-Rfou, R.; Skiena, S. DeepWalk: Online Learning of Social Representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '14)*, New York, NY, USA, 24–27 August 2014; Association for Computing Machinery: New York, NY, USA, 2014; pp. 701–710. [\[CrossRef\]](#)
17. Grover, A.; Leskovec, J. node2vec: Scalable Feature Learning for Networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*, New York, NY, USA, 13–17 August 2016; Association for Computing Machinery: New York, NY, USA, 2016; pp. 855–864. [\[CrossRef\]](#) [\[PubMed\]](#)
18. Box, G.E.P.; Jenkins, G.M.; Reinsel, G.C.; Ljung, G.M. *Time Series Analysis: Forecasting and Control*, 5th ed.; Wiley: Hoboken, NJ, USA, 2015.
19. Zachariah, R.A.; Sharma, S.; Kumar, V. Systematic Review of Passenger Demand Forecasting in Aviation Industry. *Multimed. Tools Appl.* **2023**, *82*, 46483–46519. [\[CrossRef\]](#) [\[PubMed\]](#)
20. Wang, S.; Gao, Y. A Literature Review and Citation Analyses of Air Travel Demand Studies Published between 2010 and 2020. *J. Air Transp. Manag.* **2021**, *97*, 102135. [\[CrossRef\]](#)
21. Manna, S.; Biswas, S.; Kundu, R.; Rakshit, S.; Gupta, P.; Barman, S. A Statistical Approach to Predict Flight Delay Using Gradient Boosted Decision Tree. In *Proceedings of the 2017 International Conference on Computational Intelligence in Data Science (ICCIDS)*, Piscataway, NJ, USA, 2–3 June 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 1–5. [\[CrossRef\]](#)
22. Guimerà, R.; Mossa, S.; Turtschi, A.; Amaral, L.A.N. The Worldwide Air Transportation Network: Anomalous Centrality, Community Structure, and Cities' Global Roles. *Proc. Natl. Acad. Sci. USA* **2005**, *102*, 7794–7799. [\[CrossRef\]](#) [\[PubMed\]](#)
23. Zanin, M.; Lillo, F. Modelling the Air Transport with Complex Networks: A Short Review. *Eur. Phys. J. Spec. Top.* **2013**, *215*, 5–21. [\[CrossRef\]](#)
24. Lordan, O.; Sallan, J.M.; Simo, P.; Gonzalez-Prieto, D. Robustness of the Air Transport Network. *Transp. Res. Part E Logist. Transp. Rev.* **2014**, *68*, 155–163. [\[CrossRef\]](#)
25. Kang, J.; Yang, S.; Shan, X.; Bao, J.; Yang, Z. Exploring Delay Propagation Causality in Various Airport Networks with Attention-Weighted Recurrent Graph Convolution Method. *Aerospace* **2023**, *10*, 453. [\[CrossRef\]](#)
26. Tang, Z.; Niu, Z.; Chen, X.; Huang, S.; Zhu, X. Edge-Based GNN for Network Delay Prediction Enhanced by Flight Connectivity. *Aerospace* **2026**, *13*, 161. [\[CrossRef\]](#)
27. Li, Y.; Yu, R.; Shahabi, C.; Liu, Y. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In *Proceedings of the International Conference on Learning Representations (ICLR)*; Curran Associates, Inc.: Red Hook, NY, USA, 2018.
28. Yu, B.; Yin, H.; Zhu, Z. Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence (IJCAI-18)*. International Joint Conferences on Artificial Intelligence Organization, 13–19 July 2018; AAAI Press: Washington, DC, USA, 2018; pp. 3634–3640. [\[CrossRef\]](#) [\[PubMed\]](#)
29. Hamilton, W.L.; Ying, R.; Leskovec, J. Inductive Representation Learning on Large Graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17)*, Red Hook, NY, USA, 4–9 December 2017; Curran Associates, Inc.: Red Hook, NY, USA, 2017; pp. 1025–1035.
30. Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Liò, P.; Bengio, Y. Graph Attention Networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*; Curran Associates, Inc.: Red Hook, NY, USA, 2018.
31. Dong, D.; Yu, L.; Qin, X.; Zhu, X.; Huang, Z.; Qu, Z. Dynamic-Static Graph Fusion Multi-Head Flow Attention Networks for Traffic Flow Forecasting. *Electronics* **2026**, *15*, 2294. [\[CrossRef\]](#)

32. Lv, H.; Lan, F.; Xiu, W. Mamba-Based Macro–Micro Spatio-Temporal Model for Traffic Flow Prediction. *Electronics* **2026**, *15*, 1327. [[CrossRef](#)]
33. Zang, H.; Zhu, J.; Gao, Q. Deep Learning Architecture for Flight Flow Spatiotemporal Prediction in Airport Network. *Electronics* **2022**, *11*, 4058. [[CrossRef](#)]
34. Tang, X.; Wu, J.; Wu, C.L.; Ding, Y.; Zhang, S. Dynamic prediction of aircraft turnaround milestone times using a cascaded gradient boosting model for improved airport collaborative decision-making. *J. Air Transp. Manag.* **2025**, *128*, 102842. [[CrossRef](#)]
35. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32. [[CrossRef](#)]
36. Chen, T.; Guestrin, C. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*, New York, NY, USA, 13–17 August 2016; Association for Computing Machinery: New York, NY, USA, 2016; pp. 785–794. [[CrossRef](#)]
37. Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; Ye, Q.; Liu, T.Y. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, Long Beach, CA, USA, 4–9 December 2017; Curran Associates Inc.: Red Hook, NY, USA, 2017; Volume 30, pp. 3146–3154.
38. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [[CrossRef](#)] [[PubMed](#)]
39. Yu, B.; Guo, Z.; Asian, S.; Wang, H.; Chen, G. Flight Delay Prediction for Commercial Air Transport: A Deep Learning Approach. *Transp. Res. Part E Logist. Transp. Rev.* **2019**, *125*, 203–221. [[CrossRef](#)]
40. Tofallis, C. A Better Measure of Relative Prediction Accuracy for Model Selection and Model Estimation. *J. Oper. Res. Soc.* **2015**, *66*, 1352–1362. [[CrossRef](#)]
41. Hyndman, R.J.; Koehler, A.B. Another Look at Measures of Forecast Accuracy. *Int. J. Forecast.* **2006**, *22*, 679–688. [[CrossRef](#)]
42. Makridakis, S.; Spiliotis, E.; Assimakopoulos, V. The M4 Competition: Results, Findings, Conclusion and Way Forward. *Int. J. Forecast.* **2018**, *34*, 802–808. [[CrossRef](#)]
43. Diebold, F.X.; Mariano, R.S. Comparing Predictive Accuracy. *J. Bus. Econ. Stat.* **1995**, *13*, 253–263. [[CrossRef](#)]
44. Lundberg, S.M.; Lee, S.I. A Unified Approach to Interpreting Model Predictions. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS)*, Long Beach, CA, USA, 4–9 December 2017; Curran Associates Inc.: Red Hook, NY, USA, 2017; Volume 30, pp. 4765–4774.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.