Contents lists available at ScienceDirect

Physica A

journal homepage: www.elsevier.com/locate/physa

Graph machine learning for flight delay prediction due to holding maneuver

Jorge L. Franco [a,b,c], Manoel V. Machado Neto [b], Filipe A.N. Verri [b,c], Diego R. Amancio [a,*]

[a] Institute of Mathematics and Computer Science, University of São Paulo – USP, Av. Trabalhador São-carlense, 400, São Carlos, 13566-590, SP, Brazil
[b] Aeronautics Institute of Technology, Computer Science Division – ITA, Praça Marechal Eduardo Gomes, 50, São José dos Campos, 12228-900, SP, Brazil
[c] Instituto Curvelo, Av. Marília, 1000, Galpão 27, Arujá, 07429-825, SP, Brazil

ARTICLE INFO

Keywords:
Graph neural networks
Machine learning
Complex networks

ABSTRACT

Flight delays due to holding maneuvers are a critical and costly phenomenon in aviation, driven by the need to manage air traffic congestion and ensure safety. Holding maneuvers occur when aircraft are instructed to circle in designated airspace, often due to factors such as airport congestion, adverse weather, or air traffic control restrictions. This study models the prediction of flight delays due to holding maneuvers as a graph problem, leveraging advanced Graph Machine Learning (Graph ML) techniques to capture complex interdependencies in air traffic networks. Holding maneuvers, while crucial for safety, cause increased fuel usage, emissions, and passenger dissatisfaction, making accurate prediction essential for operational efficiency. Traditional machine learning models, typically using tabular data, often overlook spatial–temporal relations within air traffic data. To address this, we model the problem of predicting holding as edge feature prediction in a directed (multi)graph where we apply both CatBoost, enriched with graph features capturing network centrality and connectivity, and Graph Attention Networks (GATs), which excel in relational data contexts. Our results indicate that CatBoost outperforms GAT in this imbalanced dataset, effectively predicting holding events and offering interpretability through graph-based feature importance. Additionally, we discuss the model's potential operational impact through a web-based tool that allows users to simulate real-time delay predictions. This research underscores the viability of graph-based approaches for predictive analysis in aviation, with implications for enhancing fuel efficiency, reducing delays, and improving passenger experience.

1. Introduction

The aviation industry increasingly relies on data-driven approaches to improve operational efficiency and reduce delays. Among the pressing challenges in air traffic management is the prediction of 'holding' maneuvers, where aircraft are instructed to delay landing due to factors such as airport congestion, adverse weather, or airspace limitations. While holding patterns are necessary for safety, they contribute to increased fuel consumption, emissions, and passenger dissatisfaction. This study aims to enhance

* Corresponding author.
E-mail addresses: jorge.luiz@usp.br (J.L. Franco), manoelmvmn@ita.br (M.V.M. Neto), filipe.verri@institutocurvelo.org.br (F.A.N. Verri), diego@icmc.usp.br (D.R. Amancio).

https://doi.org/10.1016/j.physa.2026.131318
Received 2 July 2025; Received in revised form 5 January 2026

Available online 23 January 2026

the accuracy of holding predictions using machine learning (ML) models based on graph-structured data, specifically employing advanced methodologies in Graph Machine Learning (Graph ML) and Graph Neural Networks (GNNs).

Traditional machine learning applications in aviation have primarily focused on flight delay prediction and air traffic flow management. For instance, delay predictions based on weather conditions, airport congestion, and flight schedules have been widely studied [1,2]. Recent comprehensive reviews have highlighted the evolution of machine learning approaches in aviation delay prediction, identifying class imbalance as a persistent challenge across studies [3], while network-based approaches have emerged as a promising paradigm for capturing the interconnected nature of air traffic systems [4]. However, these models often rely on tabular data representations, which limit their ability to capture complex relational patterns among airports and other influencing factors. Additionally, research specifically focused on holding maneuvers is limited and generally lacks machine learning and network-based approaches that can model the spatial and temporal dependencies intrinsic to air traffic data [5,6]. The relationship between departure delays and en-route conflicts is particularly relevant to holding maneuver prediction, as holds often occur to resolve predicted conflicts in congested airspace [7].

The use of graph-based machine learning methods is rapidly advancing in the field of intelligent transportation, where graph structures effectively capture complex spatial and temporal relationships across networks. A recent survey on GNNs in intelligent transportation systems, Rahmani et al. [8], highlights their application across a variety of domains, including traffic forecasting, demand prediction, and urban planning. This survey underscores the power of GNNs in applications where data is inherently interconnected, such as autonomous vehicle routing and intersection management. By organizing studies within these domains, they identify distinct opportunities and challenges, particularly in multi-modal models and reinforcement learning applications. Similarly, Zhao et al. [9] demonstrates the value of GNNs in the specific context of real-time traffic forecasting with their T-GCN (Temporal Graph Convolutional Network) model. By combining Graph Convolutional Networks (GCN) and Gated Recurrent Units (GRU), the T-GCN model captures both spatial and temporal dependencies, achieving state-of-the-art accuracy in urban traffic prediction tasks. These studies exemplify the increasing role of GNNs in transportation-related decision-making, showing potential for improved accuracy and efficiency in complex, dynamic systems.

The study employs two main approaches:

1. *Tabular-based approach:* We use the CatBoost model, leveraging graph features – such as centrality and connectivity metrics – that capture the significance of directed edges (flights) within the network [10].
2. *Graph Attention Network (GAT) approach:* We compare with GATs that have proven effective in applications where relational data is essential, making them a promising choice for capturing the interconnected nature of air traffic [11].

The contributions of this study are twofold. First, we demonstrate the application of graph-based ML models for predicting holding events, offering a detailed perspective on airport interdependencies. Second, by comparing the CatBoost model with the GAT, we assess which method better captures the graph topology of air traffic and achieves superior predictive performance. This research has potential implications for improved fuel efficiency, reduced delays, and enhanced passenger experiences by refining model selection and feature engineering strategies tailored to aviation applications. Our work explores graph-based ML approaches for predicting flight delays due to holding maneuvers, leveraging airport network topology.

The structure of this work is as follows. In Section 2, we review relevant literature and build the theoretical background needed. Section 3 details the dataset and the modeling. Section 4 discusses the experimental setup, with results and comparison analysis presented and model deployment.

2. Theoretical framework and related works

Graph machine learning encompasses learning on graph-structured data [12–14], including diverse tasks such as node/edge classification, link prediction, and network embeddings [15–17]. This work focuses on supervised edge classification and regression, dividing the field into `traditional` graph learning (standard ML with topological features) and `deep` graph learning based on graph neural networks [18–21].

2.1. Traditional graph learning

Traditional approaches extract graph-based features for integration into standard ML models [22,23]. Centrality measures quantify node importance [24] and extend to edges [25,26]. PageRank [27] extends eigenvector centrality with damping:

$$PR(v) = \frac{1-d}{N} + d \sum_{u \in \mathcal{N}(v)} \frac{PR(u)}{\deg(u)},$$

where d is the damping factor and $\mathcal{N}(v)$ represents neighbors. However, spectral centralities have limitations: $\mathcal{O}(n^2 d)$ complexity and node-centric focus, lacking direct edge extensions crucial for directed graphs.

2.2. Deep graph learning

Graph neural networks revolutionized graph ML through two main categories: spectral-based and spatial-based approaches [19,20].

2.2.1. Spectral-based GNNs

Spectral methods use graph Laplacian $\mathcal{L} = D - A$ for Fourier-like transforms [28]. GCN [20] simplified earlier approaches: $g_\theta \star f \approx \theta(I_n + \widetilde{A})f$, where $\widetilde{A} = D^{-1/2}AD^{-1/2}$. However, spectral methods require undirected graphs and are node-centric, excluding edge features.

2.2.2. Spatial-based GNNs

Spatial GNNs perform neighborhood aggregation through message passing. Graph Attention Networks (GATs) [11] learn attention weights:

$$\alpha_{ij} = \frac{\exp(\mathrm{LeakyReLU}(\mathbf{a}^T[\mathbf{W}\mathbf{h}_i \parallel \mathbf{W}\mathbf{h}_j]))}{\sum_{k\in\mathcal{N}(i)}\exp(\mathrm{LeakyReLU}(\mathbf{a}^T[\mathbf{W}\mathbf{h}_i \parallel \mathbf{W}\mathbf{h}_k]))},$$

enabling weighted aggregation that captures varying relationship importance in heterogeneous networks.

3. Materials and methods

This section details the materials and methods used in our study, providing a fluid overview of the datasets and aviation features employed in our predictive modeling. Our objective is to predict whether a given aircraft will experience a delay due to a holding maneuver by leveraging both a CatBoost model enhanced with graph-derived features and a Graph Attention Network (GAT) approach.

The study utilizes two datasets comprising 42,336 flight operations within the Brazilian airspace system spanning June 1, 2022 to February 16, 2023 (260 days of operations). Each observation represents a single flight with comprehensive meteorological, geographical, and operational features [29]. The binary classification dataset predicts holding occurrence (720 positive samples representing flights that experienced holding, and 41,616 negative samples representing flights without holding, yielding 1.7% class imbalance), while the regression dataset predicts holding duration in seconds for delayed flights.[1]

Aviation features in our datasets can be understood through three intertwined dimensions. First, meteorological features – sourced from both METAR and METAF reports – provide real-time weather observations and complementary forecast elements. METAR (Meteorological Terminal Aviation Routine Weather Report) delivers data on wind direction and speed, visibility, temperature, and cloud coverage, while METAF (combined meteorological data from METAR and TAF, which provides short-term forecasts) reports offer additional context that deepens the understanding of atmospheric conditions that might influence holding maneuvers. Second, geographical features offer spatial context by capturing the geodesic flight distance between departure and arrival airports, along with details such as airport altitudes and precise latitude and longitude coordinates. This spatial information is critical for analyzing how physical location and distance affect delay events. Finally, flight-specific features capture operational nuances such as the flight hour and indicators of runway activity, including previous runway head changes and recent alterations in runway configuration. Together, these features provide a holistic view of the operational environment, allowing for a more nuanced prediction of holding maneuvers.

3.1. Data preprocessing

The dataset contains minimal missing observations. For meteorological variables, standard forward-filling approaches were applied using the most recent valid observation, consistent with aviation operational practices. Graph-based features are computed from the complete network structure and therefore contain no missing values.

For model training and evaluation, we employed a two-stage random split approach: first, an 80–20 split to separate test data (20%), followed by splitting the remaining 80% into training (70% of total) and validation (10% of total) sets using a fixed random seed for reproducibility. This results in a final 70-10-20 train-validation-test distribution. Additionally, we conducted comprehensive temporal validation experiments to ensure the robustness of our methodology for operational deployment, as detailed in Appendix A.1.

In our modeling approaches, we first apply CatBoost – a gradient boosting decision tree model enhanced with graph-derived features from the airport network – which is well-suited to handle class imbalances and offers transparency through explainable AI techniques. In parallel, we explore Graph Attention Networks (GATs), which use attention mechanisms to model the intricate relationships within graph-structured data. Although GATs are promising in capturing relational patterns, our experiments reveal that they struggle with class imbalance, often leading to overfitting in deeper architectures.

The following sections provide further details on the specific implementations and experimental configurations for both the CatBoost and GAT models.

[1] The datasets and problem formulation originate from the Data Science Challenge [29] proposed by ICEA (Institute of Airspace Control) and ITA (Aeronautics Institute of Technology) at the Engineering Education for The Future (EEF) 2024 event. The challenge and datasets are publicly available at: https://www.kaggle.com/competitions/data-science-challenge-at-eef-2024/

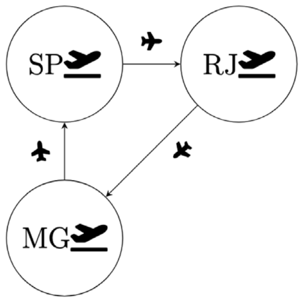


Fig. 1. Example of airports and directed flights.

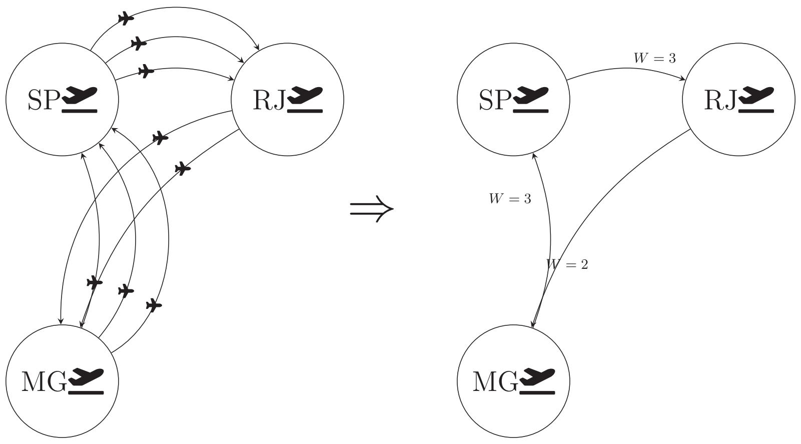


Fig. 2. Transformation of a multigraph of flights into a weighted directed graph. The multigraph (left) represents multiple flights between airports. In the weighted graph (right), edges are aggregated to show total flights as weights.

3.2. CatBoost with graph features

This study employs the CatBoost model, a high-performance gradient boosting library, chosen specifically for its ability to handle categorical features and class imbalance effectively, as well as for its robust handling of noisy data [10]. CatBoost has been widely recognized for its superior performance in structured data problems, particularly when compared to other boosting algorithms like XGBoost and LightGBM, thanks to its unique techniques such as ordered boosting and categorical feature encoding. These innovations help prevent overfitting and enhance generalization in class unbalanced problems.

Here, we describe how CatBoost is combined the graph-based features that are extracted of our modeled airports network. These features are derived from the weighted directed graph and encoded as tabular features that are used as input to the model as we describe in the following sections.

3.2.1. Graph representation of the flight network

To model the interactions in flight data, we represented the problem as a directed graph, depicted in Fig. 1, where each node represents an airport, here we represent the airports as: SP (São Paulo), MG (Minas Gerais), RJ (Rio de Janeiro). In this network, nodes represent airports. Directed edges represent flights, with each edge directed from the departure airport to the destination airport. Holding maneuvers occur between departure and arrival airports during a flight. Thus, we model each flight as a directed edge in the airport graph, where edge features predict the occurrence/duration of holding. Given the frequent occurrence of multiple flights between the same pairs of airports (i.e., multiedges), we have in fact a multigraph, however we abstract it into a weighted directed graph as shown in 2. Here, each edge's weight corresponds to the total number of flights between a specific pair of airports, transforming multiple directed edges into a single weighted edge. This abstraction allows us to calculate key network metrics more easily, which we then used as features in the CatBoost model.

3.2.2. Graph-based features

The graph-based features encode essential structural information about the flight network, capturing connectivity, centrality, and robustness. These features are crucial for understanding the influence of each airport within the network and its potential impact on flight holding patterns.

Although we have already made this simplification of the multigraph, transforming it into a weighted directed graph, we still need to extract the features from the graph and encode them as tabular data. However, this is not straightforward, as the graph measures are not directly compatible with the model.

The modeling will impact dramatically in the resulting graph-based features. For instance, we need to calculate edge measures, but this is not so explored as node measures, so the lack of possibilities is a challenge to be overcome. Another challenge is the direction, that is, we have to create edge measures in a directed weighted graph, which is hard, as we detailed in Section 2.1, because most of the complex networks measures proposed are 'node centric' and for undirected graphs.

With this in mind, we can observe why the weighted graph transformation was so important, since the measures available for our setting are strongly dependent to the weight (as we will detail later), and our graph is almost totally connected, so in undirected unweighted setting they would be approximately equal, leaving no information. The following graph metrics were calculated from the weighted directed graph:

1. *Betweenness Centrality:* Captures the relative importance of each airport in terms of the routes it controls within the network. Higher values indicate airports that serve as critical transit points.
2. *Flow Betweenness:* Highlights the flow dynamics of connections, showing how flights tend to route through certain airports, which may correlate with congestion.
3. *Edge Connectivity:* Indicates the robustness of airport connections, with higher values signifying more resilient routes between airports that could better handle rerouting needs.
4. *Degree Difference:* Measures the disparity between in-degrees and out-degrees at each node, helping to identify key hubs or spokes in the network.
5. *Google Matrix:* Based on PageRank centrality, the Google matrix provides a probabilistic transition representation for each airport, which reflects both local and global connectivity.

The Google Matrix formulation we employ adapts the PageRank algorithm to edge-level importance in directed weighted graphs. For a weighted directed graph with adjacency matrix A where a_{ij} represents the weight (number of flights) from airport i to airport j, we first construct the transition probability matrix P where each element is:

$$p_{ij} = \frac{a_{ij}}{\sum_k a_{ik}} \tag{1}$$

representing the probability of transitioning from node i to node j in a random walk on the flight network. The Google Matrix G is then defined as:

$$G = \alpha P + (1-\alpha)\frac{ee^T}{n} \tag{2}$$

where $\alpha \in (0,1)$ is the damping factor (we use $\alpha = 0.85$ following the standard PageRank formulation), e is the n-dimensional vector of ones, n is the number of airports, and ee^T/n is the uniform teleportation matrix ensuring ergodicity. To obtain edge-level features we assign to each directed edge (i,j) its position in G. This adaptation allows us to leverage the global connectivity insights provided by PageRank while focusing on edge-level characteristics relevant for predicting holding patterns.

As we can see, these features are not commonly used in the literature. Here is where the weighted network plays a crucial role, edge betweenness centrality [30] is constructed using shortest paths in the network, thus the weight will be crucial part of it, since without it the graph is almost fully connected, the shortest path will be almost the same for all pairs of nodes. The same happens with flow betweenness centrality [31], that is a measure based on electrical circuits Kirchhoff law, more specifically, instead of working with shortest paths, it use the maximum flow that pass through each edge and the weight visualized as capacity will be crucial to calculate it.

The edge connectivity is a measure of the minimum number of edges that must be removed to disconnect the graph, and the weight will be crucial to calculate it. The degree difference we stated here as a measure of the difference between the in-degree and out-degree of a node. The Google matrix is a way we derived to keep using PageRank for edges. In fact, as we detailed in Section 2.1, although the PageRank centrality could be applied in our graph, since it satisfies the Perron theorem as it is always positive and strongly connected, it is a node measure, so we have to adapt it to edges, and the Google matrix is a way to do it.

These features enhance the CatBoost model by embedding graph-theoretic insights into its predictive capabilities, ultimately enabling a more nuanced understanding of how network dynamics relate to flight holding patterns.

3.3. Graph attention network

As we previously described, the GAT model in Section 2.2.2 has a large range of applications, from drug discovery to fake news detection [32]. The GAT model leverages the underlying graph structure but does not rely on explicitly computed graph-derived features like the CatBoost model does. Instead, it learns node representations in an end-to-end manner, enabling the model to capture the relationships between airports and flights directly from the data.

The modeling of a GNN for our problem is a challenging task, as we have to adapt the model to predict edge features, since 'holding' is an edge feature in our setting. In Section 2.2.1 we detailed why the spectral-based GNNs are not suitable for our setting, as they are not able to handle edge features and direction, due to their 'node-centric' approach based on the adjacency matrix. Although spatial-based GNNs can handle direction in their majority, they are not able to handle edge features in general, since they need to create a way to aggregate the edge features with the neighbors' features.

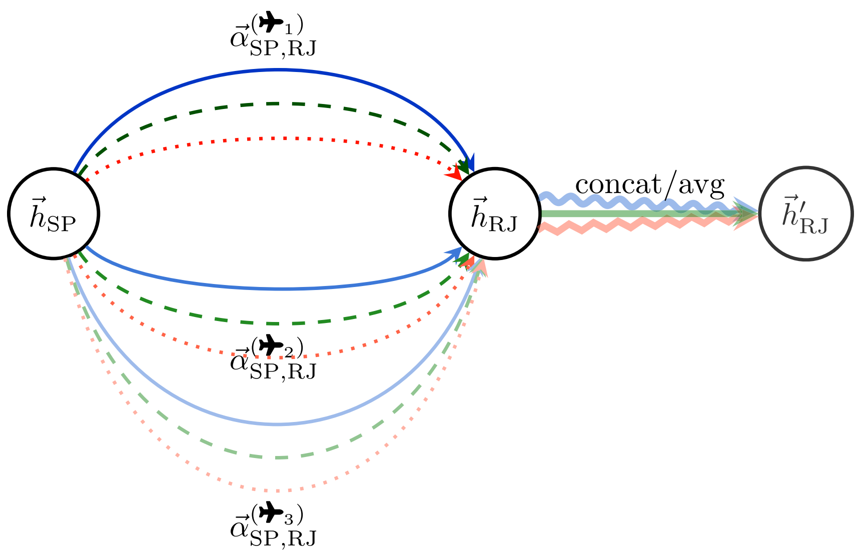

$\vec{\alpha}^{(✈_1)}_{\mathrm{SP,RJ}}$
$\vec{h}_{\mathrm{SP}}$
$\vec{h}_{\mathrm{RJ}}$
concat/avg
$\vec{h}'_{\mathrm{RJ}}$
$\vec{\alpha}^{(✈_2)}_{\mathrm{SP,RJ}}$
$\vec{\alpha}^{(✈_3)}_{\mathrm{SP,RJ}}$

Fig. 3. Airport multigraph GAT Layer with multi-head attention for three different flights between nodes (SP,RJ), with alternating colors opacity for each flight.

The GAT model is so used because it is highly adaptable in practically any graph setting. As we will show, the attention mechanism detailed in Section 2.2.2 can be generalized to handle edge features, and the model can be adapted to predict edge features. In fact, a simple concatenation (‖) in the attention formula already gives us this power,

$$\alpha_{ij} = \sigma(\phi_1(\mathbf{a}^T[Wh_i \| Wh_j \| W_2 e_{ij}])), \tag{3}$$

where e_{ij} are the edge features, h_i and h_j are the node features, and W and W_2 are the weight matrices. This formula allows the model to focus on the relevant neighboring nodes, making it ideal for relational data. In our case, the edge features are the tabular data features with holding being part of them, which is the target we want to predict.

Our GAT implementation uses hidden dimension of 32 units with configurable layer depths (1–30 layers). We employ Focal Loss with $\alpha = 0.1$ and $\gamma = 7.5$ for class imbalance, Adam optimizer with learning rate 0.01, and 100 training epochs with early stopping.

Node features are initialized by concatenating one-hot encoding of airport identifiers with geographical features (latitude, longitude, altitude) and degree centrality statistics. The edge features comprise the complete set of meteorological, temporal, and operational features described in the data section. For the final prediction, we concatenate learned node embeddings with edge features and pass through a multi-layer perceptron.

Furthermore, the directed multigraph setting we described in Section 3.2 is not a problem for the GAT model, since it can handle multiple edges between the same pair of nodes, as we will show in the following sections. We show how we model the GAT to be a directed multigraph representing the flights and their features in Fig. 3.

The GAT's handling of multiple edges between the same node pairs requires careful attention mechanism design. Unlike the weighted graph abstraction used for CatBoost feature extraction, our GAT implementation preserves individual flight edges, allowing the attention mechanism to compute separate attention coefficients for each flight between airports i and j. For multiple edges $e_{ij}^{(k)}$ where $k = 1, 2, \ldots, K$ represents different flights, the attention mechanism computes:

$$\alpha_{ij}^{(k)} = \frac{\exp\left(\sigma\left(\mathbf{a}^T[Wh_i \| Wh_j \| W_2 e_{ij}^{(k)}]\right)\right)}{\sum_{l\in\mathcal{N}_i}\sum_{m=1}^{K_{il}} \exp\left(\sigma\left(\mathbf{a}^T[Wh_i \| Wh_l \| W_2 e_{il}^{(m)}]\right)\right)} \tag{4}$$

where $\mathcal{N}_i$ represents the neighborhood of node i, and K_{il} is the number of edges between nodes i and l. This formulation ensures that the attention weights for all edges from node i (including multiple edges to the same destination) sum to unity, while allowing each individual flight to contribute distinct attention scores based on its specific characteristics.

The aggregated message for node i from all its neighbors becomes:

$$\mathbf{h}_i^{(l+1)} = \sigma\left(\sum_{j\in\mathcal{N}_i}\sum_{k=1}^{K_{ij}} \alpha_{ij}^{(k)} \cdot Wh_j^{(l)}\right) \tag{5}$$

This updates the node embeddings at each GAT layer l, allowing nodes to aggregate information from all connected flights. The model differentiates between flights with similar origins and destinations but different operational characteristics, providing a more nuanced representation of airport connectivity patterns than simple weighted edge approaches.

Table 1
Performance metrics for various GAT layer configurations and CatBoost with graph features.

Model	Test accuracy	Precision	Recall	F1-Score
CatBoost	0.90	**0.09**	0.58	**0.16**
1 GAT Layer	**0.95**	0.03	0.06	0.04
3 GAT Layers	0.52	0.01	0.40	0.03
5 GAT Layers	0.57	0.01	0.30	0.02
10 GAT Layers	0.91	0.02	0.08	0.03
30 GAT Layers	0.02	0.02	**0.99**	0.03

Finally, the last layer of our predictor uses the updated node embeddings $h_i^{(L)}$ and $h_j^{(L)}$ (where L is the final layer) along with the edge feature $e_{ij}^{(k)}$ of flight k to predict holding. The final prediction concatenates these representations:

$$\hat{y}_k = \sigma(\mathrm{MLP}(h_i^{(L)} \| h_j^{(L)} \| e_{ij}^{(k)})) \quad .$$

4. Results

This section presents a comprehensive evaluation of the models applied in this study, with a focus on comparing the CatBoost and Graph Attention Network (GAT) models, as well as examining the regression capabilities and interpretability of the CatBoost approach.

4.1. Model performance and interpretability

The performance of each model in predicting flight delays due to holding maneuvers was evaluated using a range of metrics, including accuracy, precision, recall, and F1-score. Table 1 summarizes the results for various GAT layer configurations alongside the CatBoost model. While the single-layer GAT model achieved the highest accuracy, its precision and F1-score were markedly low. In contrast, the CatBoost model provided a balanced performance across all metrics, which is particularly beneficial for the imbalanced dataset at hand. This balanced performance indicates that CatBoost is more effective at capturing both delayed and non-delayed flights, even if it sometimes errs on the side of caution in predicting delays.

Beyond classification, the CatBoost model was also applied to a regression task aimed at predicting continuous delay values. Figs. 5 and 6 illustrate the distributions of the predicted and actual delay values, respectively. The overall alignment between these distributions suggests that CatBoost is effective at capturing the underlying trends in flight delay data, though deviations at extreme values highlight potential areas for further refinement. While the distribution-level agreement demonstrates that the model captures meaningful relationships between features and delay patterns, we acknowledge that point-wise prediction accuracy metrics such as MAE, RMSE, and R^2 would strengthen the regression analysis and provide complementary insights, particularly for operational deployment where specific delay duration predictions would be needed. A comprehensive regression analysis with these quantitative metrics and residual analysis is provided in Appendix A.2.

Interpretability is another key strength of the CatBoost approach. By leveraging graph-based features, the model not only achieves robust performance but also offers transparency regarding feature importance. As shown in Fig. 4, Explainable AI (XAI) techniques help elucidate how specific graph-based metrics contribute to the model's predictions, thereby providing valuable insights into the decision-making process. While graph features like betweenness appeared in the top 10 predictors, operational features (e.g., METAF, flight time) dominated importance. This suggests airport connectivity supplements but does not override flight-specific factors.

4.2. Deployment, implementation, and discussion

The practical applicability of our approach is further demonstrated through the development of a web-based simulation tool. The source code for this project is available on GitHub at https://github.com/graph-learning-ita/airnet-holding-ml/. The simulation tool, named *Airdelay*, is implemented using Folium and Streamlit and is accessible at https://airdelay.manoel.dev. This tool visualizes flight delays as predicted by the CatBoost model in real time, enabling users to simulate various operational scenarios and assess the potential impact of holding maneuvers.

Fig. 7 shows a screenshot of the *Airdelay* interface, which allows users to interact with the map, explore different scenarios, and gain a deeper understanding of the model's performance through dynamic visualizations.

In summary, our results underscore the advantages of the CatBoost model for this application. Despite the high accuracy observed with some GAT configurations, issues such as overfitting and unstable performance – particularly in terms of precision and F1-score – limit their effectiveness in an imbalanced setting. The CatBoost model, enhanced by graph-based features, not only achieves superior performance in both classification and regression tasks but also offers interpretability through XAI techniques.

Nevertheless, our study is not without limitations. The scope of hyperparameter tuning was restricted, and the challenges posed by class imbalance remain significant. Future research should focus on refining the GAT architecture, exploring additional graph-based features, and potentially incorporating alternative models such as Support Vector Machines (SVM) to further improve predictive performance. Moreover, emerging GNN architectures, such as the provably powerful graph neural network for directed multigraphs introduced in Egressy et al. [33], hold promise for addressing class imbalance issues and enhancing minority-class performance.

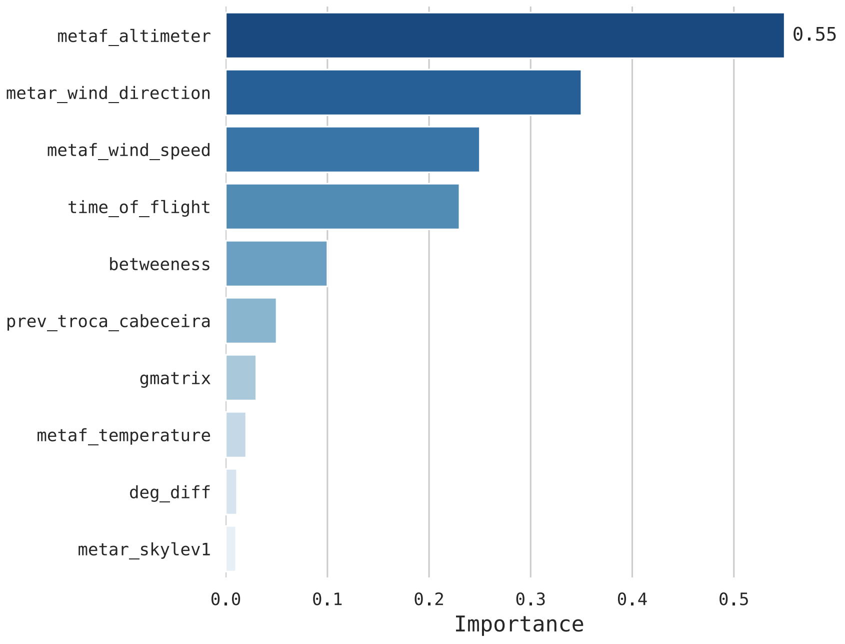


Fig. 4. Feature importance for the CatBoost model on the airport network dataset, highlighting the relevance of graph-based features.

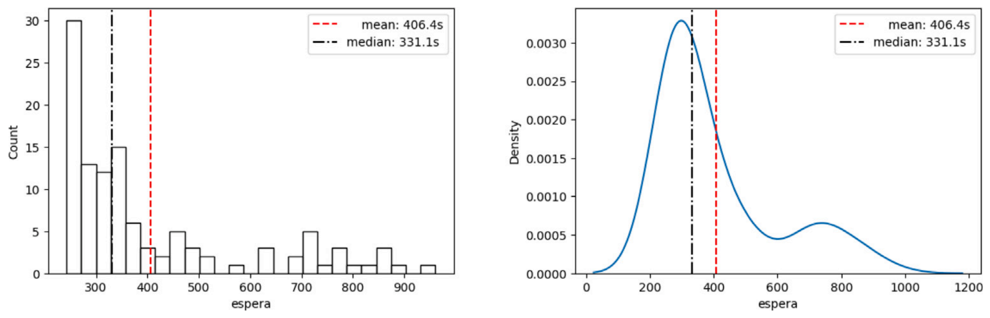


Fig. 5. Distribution of predicted delay values (y_{pred}) for the regression task using CatBoost.

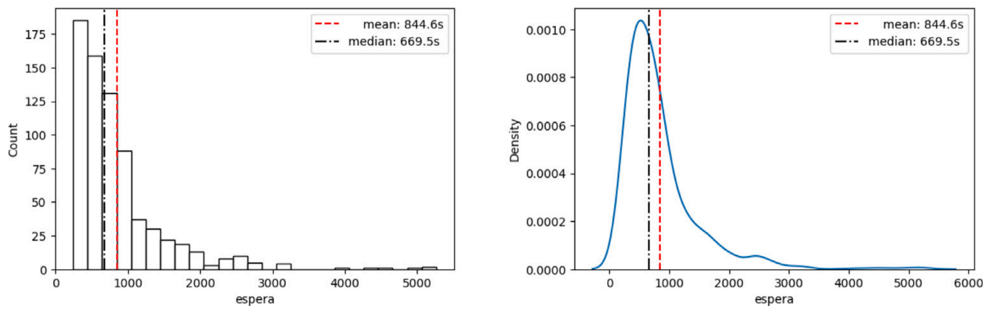
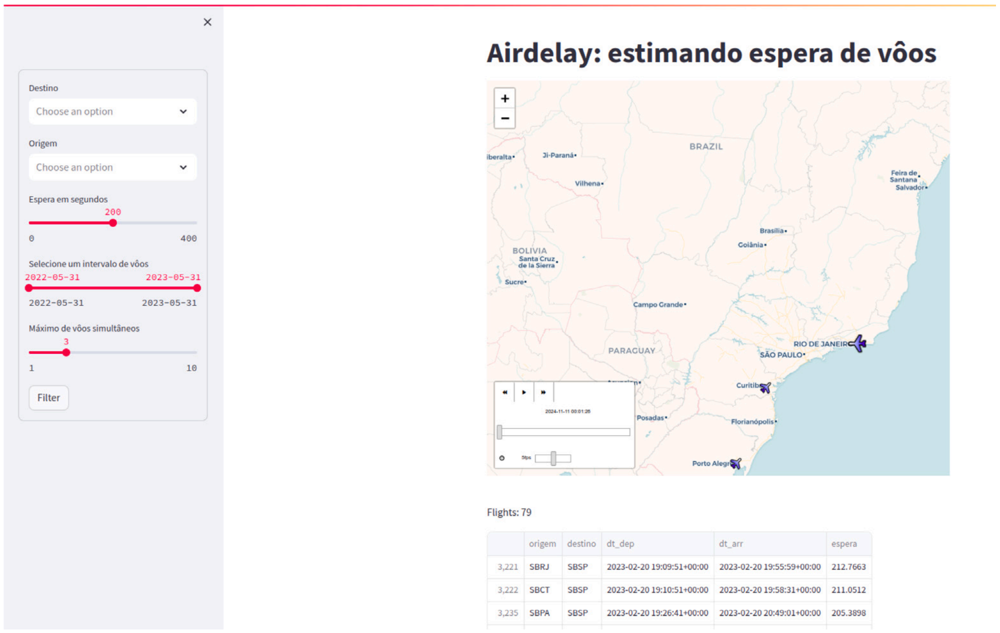


Fig. 6. Distribution of actual delay values (y_{test}) for the test set in the regression task.

5. Conclusion

This study examined the prediction of flight delays caused by holding maneuvers using graph machine learning techniques. By modeling air traffic as a directed network, we applied both CatBoost, which integrates graph-based features, and GATs, which capture relational dependencies. Our results demonstrate that while GATs provide a flexible framework for learning from structured data, CatBoost achieved better performance on this imbalanced dataset, highlighting the effectiveness of graph-based feature engineering


Airdelay: estimando espera de vôos

Destino: Choose an option
Origem: Choose an option
Espera em segundos: 200 (0–400)
Selecione um intervalo de vôos: 2022-05-31 – 2023-05-31
Máximo de vôos simultâneos: 3 (1–10)

Flights: 79

	origem	destino	dt_dep	dt_arr	espera
3,221	SBRJ	SBSP	2023-02-20 19:09:51+00:00	2023-02-20 19:55:59+00:00	212.7663
3,222	SBCT	SBSP	2023-02-20 19:10:51+00:00	2023-02-20 19:58:31+00:00	211.0512
3,235	SBPA	SBSP	2023-02-20 19:26:41+00:00	2023-02-20 20:49:01+00:00	205.3898

Fig. 7. *Airdelay* web-based simulation tool, showing predicted flight delays due to holding maneuvers.

in structured tabular models. Additionally, our web-based simulation tool illustrates the practical implications of these predictive models for real-time decision-making in air traffic management. The comprehensive evaluation methodology presented in Appendix A provides additional evidence of the robustness and operational readiness of our approach.

Despite these advances, several areas remain open for improvement. One limitation of our approach is the lack of explicit data imputation techniques for handling class imbalance. Future work could explore oversampling methods such as GraphSMOTE [34] to enhance the representation of underrepresented holding events in the dataset. Furthermore, since this is an edge-based classification problem, incorporating topological deep learning techniques could offer new insights by leveraging higher-order structures in the flight network. Methods from topological graph learning, such as persistent homology-based representations [35] or topological message passing [36,37] techniques tailored for edge features, may provide a richer understanding of the underlying flight delay dynamics.

Additionally, future research could explore hybrid models that combine GNNs and gradient boosting to leverage both graph-based feature learning and tabular-based decision trees. The integration of real-time air traffic and weather data could further enhance predictive accuracy, making these models more robust for operational deployment. As airspace management continues to evolve, these approaches hold promise for optimizing flight scheduling, reducing delays, and improving overall passenger experience.

CRediT authorship contribution statement

Jorge L. Franco: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Investigation, Formal analysis, Data curation, Conceptualization. **Manoel V. Machado Neto:** Writing – review & editing, Writing – original draft, Methodology, Investigation, Conceptualization. **Filipe A.N. Verri:** Writing – review & editing, Supervision, Methodology, Conceptualization. **Diego R. Amancio:** Writing – review & editing, Supervision, Investigation, Formal analysis.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001. Diego R. Amancio acknowledges financial support from CNPq (Grant no. 311074/2021-9) and FAPESP (Grant no. 2020/06271-0).

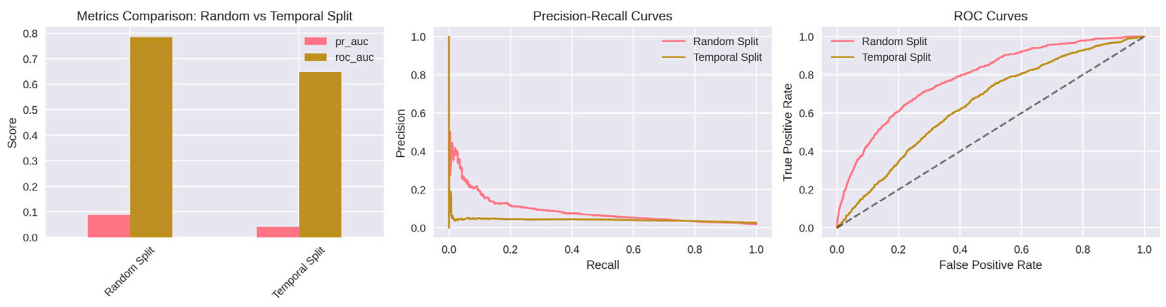


Fig. A.8. Temporal validation framework for operational robustness assessment. Left: Performance metrics comparison demonstrating realistic operational expectations under temporal constraints. Center: Precision-Recall curves showing authentic predictive capability for real-world deployment. Right: ROC curves confirming robust performance under stringent temporal validation. Our temporally-aware validation provides conservative, operationally achievable performance estimates essential for reliable aviation systems deployment.

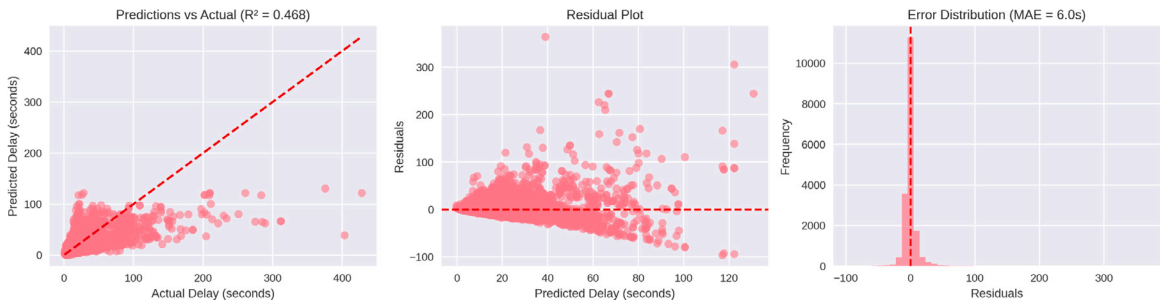


Fig. A.9. Enhanced regression evaluation for continuous delay prediction. Left: Scatter plot of predicted vs. actual delays showing moderate correlation (R^2 = 0.468). Center: Residual analysis revealing unbiased predictions with increasing variance for longer delays. Right: Distribution of prediction errors demonstrating normal error characteristics. MAE = 6.00s, RMSE = 12.57 s.

Appendix A. Additional experiments

This appendix presents a comprehensive experimental analysis demonstrating the methodological rigor and operational readiness of our approach. Our enhanced evaluation framework addresses critical concerns regarding evaluation metrics, temporal validation protocols, and model interpretability, providing robust evidence for the reliability and transparency required in aviation applications.

A.1. Temporal validation for operational robustness

To ensure the reliability of our methodology for real-world aviation applications, we implemented rigorous temporal validation protocols that simulate authentic operational deployment conditions. This analysis demonstrates our commitment to methodological rigor and provides realistic performance expectations for operational systems (see Fig. A.8).

A.2. Evaluation regression metrics

Fig. A.9 presents a comprehensive analysis of the regression performance for predicting continuous delay values. The left panel shows the correlation between predicted and actual delay values, revealing a moderate but consistent relationship (R^2 = 0.468). While the model captures the general trend of delay magnitudes, systematic deviations are observed, particularly for extreme delay values exceeding 20 s.

A.3. Implications for operational deployment

The temporally-aware evaluation reported above indicates conservative but operationally relevant predictive skill for delay estimation; however, consistent with FAA guidance the transition to operations requires embedding the model within formal verification, validation and human-factors controls [38,39]. Specifically, documented acceptance criteria tied to temporally-aware benchmarks, calibrated probabilistic outputs that map to explicit operational risk tolerances, traceable data provenance and reproducible preprocessing, and continuous monitoring for distributional drift are prerequisites for field trials. Integration should preserve human authority via human-in-the-loop controls and be preceded by scenario-based simulation and domain expert review; these measures align deployment with the FAA’s AI safety assurance principles and human-factors expectations.

Data availability

Data will be made available on request.

References

[1] M. Lambelho, M. Mitici, S. Pickup, A. Marsden, Assessing strategic flight schedules at an airport using machine learning-based flight delay and cancellation predictions, J. Air Transp. Manag. 82 (2020) 101737.
[2] G. Gui, F. Liu, J. Sun, J. Yang, Z. Zhou, D. Zhao, Flight delay prediction based on aviation big data and machine learning, IEEE Trans. Veh. Technol. 69 (1) (2019) 140–150.
[3] S. Wandelt, X. Chen, X. Sun, Flight delay prediction: A dissecting review of recent studies using machine learning, IEEE Trans. Intell. Transp. Syst. 26 (4) (2025) 4283–4297, http://dx.doi.org/10.1109/TITS.2025.3528536.
[4] S. Wandelt, X. Chen, X. Sun, Networks for flight delay analysis: A scoping review and research agenda, IEEE Trans. Netw. Sci. Eng. 12 (2) (2025) 1250–1266, http://dx.doi.org/10.1109/TNSE.2025.3526850.
[5] D.h. Lee, C.J. Kim, M.J. Heo, J.W. Hwang, H.G. Lyu, J.Y. Lee, Development of real-time maneuver library generation technique for implementing tactical maneuvers of fixed-wing aircraft, Int. J. Aerosp. Eng. 2020 (1) (2020) 7025374.
[6] A.P. Smith, H. Bateman, Management of holding patterns: A potential ADS-b application, in: 2008 IEEE/AIAA 27th Digital Avionics Systems Conference, IEEE, 2008, pp. 3–D.
[7] A. Chandra, A. Verma, To delay or not to delay? A hybrid relationship between departure delay, en-route conflict probability, and number of conflicts, J. Air Transp. Res. Soc. 4 (2025) 100053, http://dx.doi.org/10.1016/j.jatrs.2024.100053.
[8] S. Rahmani, A. Baghbani, N. Bouguila, Z. Patterson, Graph neural networks for intelligent transportation systems: A survey, IEEE Trans. Intell. Transp. Syst. 24 (8) (2023) 8846–8885, http://dx.doi.org/10.1109/TITS.2023.3257759.
[9] L. Zhao, Y. Song, C. Zhang, Y. Liu, P. Wang, T. Lin, M. Deng, H. Li, T-GCN: A temporal graph convolutional network for traffic prediction, IEEE Trans. Intell. Transp. Syst. 21 (9) (2020) 3848–3858, http://dx.doi.org/10.1109/TITS.2019.2935152.
[10] L. Prokhorenkova, G. Gusev, A. Vorobev, A.V. Dorogush, A. Gulin, CatBoost: unbiased boosting with categorical features, Adv. Neural Inf. Process. Syst. 31 (2018).
[11] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, Y. Bengio, Graph Attention Networks, Int. Conf. Learn. Represent. (2018) URL https://openreview.net/forum?id=rJXMpikCZ.
[12] I. Chami, S. Abu-El-Haija, B. Perozzi, C. Ré, K. Murphy, Machine learning on graphs: A model and comprehensive taxonomy, J. Mach. Learn. Res. 23 (89) (2022) 1–64, URL http://jmlr.org/papers/v23/20-852.html.
[13] F.A.N. Verri, L. Zhao, High level data classification based on network entropy, in: The 2013 International Joint Conference on Neural Networks, IJCNN, 2013, pp. 1–5, http://dx.doi.org/10.1109/IJCNN.2013.6707042.
[14] J. You, X. Ma, Y. Ding, M.J. Kochenderfer, J. Leskovec, Handling missing data with graph representation learning, Adv. Neural Inf. Process. Syst. 33 (2020) 19075–19087.
[15] E.A. Corrêa Jr., D.R. Amancio, Word sense induction using word embeddings and community detection in complex networks, Phys. A 523 (2019) 180–190.
[16] D.R. Amancio, Network analysis of named entity co-occurrences in written texts, Europhys. Lett. 114 (5) (2016) 58005.
[17] A. Semeraro, S. Vilella, R. Improta, E.S. De Duro, S.M. Mohammad, G. Ruffo, M. Stella, EmoAtlas: An emotional network analyzer of texts that merges psychological lexicons, artificial intelligence, and network science, Behav. Res. Methods 57 (2) (2025) 77.
[18] L.d.F. Costa, Characterization of complex networks: A survey of measurements, Adv. Phys. 56 (1) (2007) 167–242.
[19] J. Bruna, W. Zaremba, A. Szlam, Y. LeCun, Spectral networks and locally connected networks on graphs, 2013, arXiv preprint arXiv:1312.6203.
[20] T.N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, 2016, arXiv preprint arXiv:1609.02907.
[21] W. Hamilton, Z. Ying, J. Leskovec, Inductive representation learning on large graphs, Adv. Neural Inf. Process. Syst. 30 (2017).
[22] M. Newman, Networks, Oxford University Press, 2018.
[23] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, D.-U. Hwang, Complex networks: Structure and dynamics, Phys. Rep. 4 (424) (2006) 175–308.
[24] P. Bonacich, Power and centrality: A family of measures, Am. J. Sociol. 92 (5) (1987) 1170–1182.
[25] L. Lu, M. Zhang, Edge betweenness centrality, in: Encyclopedia of Systems Biology, Springer New York, New York, NY, 2013, pp. 647–648, http://dx.doi.org/10.1007/978-1-4419-9863-7_874.
[26] T. Bröhl, K. Lehnertz, Centrality-based identification of important edges in complex networks, Chaos: Interdiscip. J. Nonlinear Sci. 29 (3) (2019).
[27] S. Brin, The PageRank citation ranking: bringing order to the web, Proc. ASIS, 1998 98 (1998) 161–172.
[28] D. Bo, X. Wang, Y. Liu, Y. Fang, Y. Li, C. Shi, A survey on spectral graph neural networks, 2023, arXiv:2302.05631. URL https://arxiv.org/abs/2302.05631.
[29] F. Verri, Data science challenge at EEF 2024, 2024, Kaggle, https://kaggle.com/competitions/data-science-challenge-at-eef-2024.
[30] M. Newman, M. Girvan, Finding and evaluating community structure in networks, Phys. Rev. E 69 (2) (2004) 026113.
[31] L.C. Freeman, S.P. Borgatti, D.R. White, Centrality in valued graphs: A measure of betweenness based on network flow, Soc. Netw. 13 (2) (1991) 141–154.
[32] M. Caravanti de Souza, M.P. Silva Gôlo, A. Mário Guedes Jorge, E. Carvalho Freire de Amorim, R. Nuno Taborda Campos, R. Marcondes Marcacini, S. Oliveira Rezende, Keywords attention for fake news detection using few positive labels, Inf. Sci. 663 (C) (2024) http://dx.doi.org/10.1016/j.ins.2024.120300.
[33] B. Egressy, L. Von Niederhäusern, J. Blanuša, E. Altman, R. Wattenhofer, K. Atasu, Provably powerful graph neural networks for directed multigraphs, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 38, (10) 2024, pp. 11838–11846.
[34] T. Zhao, X. Zhang, S. Wang, Graphsmote: Imbalanced node classification on graphs with graph neural networks, in: Proceedings of the 14th ACM International Conference on Web Search and Data Mining, 2021, pp. 833–841.
[35] J. Immonen, A. Souza, V. Garg, Going beyond persistent homology using persistent homology, Adv. Neural Inf. Process. Syst. 36 (2024).
[36] C. Bodnar, F. Frasca, Y. Wang, N. Otter, G.F. Montufar, P. Lio, M. Bronstein, Weisfeiler and lehman go topological: Message passing simplicial networks, in: International Conference on Machine Learning, PMLR, 2021, pp. 1026–1037.
[37] J.L. Franco, G. Duarte, A.V. Nikitin, M.A. Ponti, D. Mesquita, A.H. Souza, Differentiable lifting for topological neural networks, in: Non-Euclidean Foundation Models: Advancing AI beyond Euclidean Frameworks, 2025, URL https://openreview.net/forum?id=qL6sgVeOaI.
[38] Federal Aviation Administration, Roadmap for Artificial Intelligence Safety Assurance, Technical Report, Federal Aviation Administration, 2024, URL https://www.faa.gov/media/82891. (Accessed 03 January 2026).
[39] Federal Aviation Administration, NextGen Human Factors Division, Human Factors Guidance for the Integration of Artificial Intelligence/Machine Learning in FAA Systems, Technical Report, U.S. Department of Transportation, Federal Aviation Administration, 2025, URL https://rosap.ntl.bts.gov/view/dot/86694/dot_86694_DS1.pdf. (Accessed 03 January 2026).