Second US-Europe Air Transportation Research and Development Symposium (ATRDS2026)

Temporal Residual Learning for Real-Time Air Traffic Complexity Forecasting

Go Nam Lui, Guglielmo Lulli
Management School, Lancaster University
Lancaster, United Kingdom
{g.n.lui, g.lulli}@lancaster.ac.uk

M. Florencia Lema-Esposto
Centro de Referencia I+D+i en ATM (CRIDA)
Madrid, Spain
mflema@e-crida.enaire.es

***Abstract*—As the Air Traffic Management (ATM) landscape evolves with increasing traffic demand and emerging users, accurate short-term forecasting of air traffic complexity becomes more important in the development of next-generation airspace system. This paper proposes a new Temporal Residual Recurrent Neural Network (TR-RNN) architecture designed to capture the non-linear temporal dynamics of sector complexity. Using a dataset from the Madrid Area Control Center (ACC) comprising over 2 million rows of state vectors, we benchmark the TR-RNN against state-of-the-art tree-based ensemble methods (Random Forest, XGBoost, LightGBM). Our results reveal a performance crossover: while tree-based models benefit from static sector constraints at longer horizons (15–60 minutes), the proposed TR-RNN significantly outperforms them in the short term (5–10 minutes) by effectively modeling the momentum of traffic complexity changes with only univariate information. With sub-millisecond inference latency, the TR-RNN demonstrates the potential for integration into real-time digital assistants and automated conflict detection tools.**

***Keywords*—Air Traffic Management; Complexity Prediction; Recurrent Neural Networks; Machine Learning**

I. Introduction

Air traffic controllers (ATCO) are facing heightened pressure from increasing traffic volumes, adverse weather patterns, and emerging airspace users. To mitigate these challenges, SESAR's "Dynamic Airspace Configuration" (DAC) has been identified as a vital tool for optimizing capacity and reducing delays [1]. As a cornerstone of both the Airspace Architecture Study and the SMARTS project [2], DAC's success relies on the development of more precise air traffic complexity definition as the proxy of ATCO workload to support future demand-capacity balancing.

Historically, ATM has relied on simplistic metrics, such as aircraft occupancy counts, to approximate ATCO workload. However, contemporary research suggests that "sector count" fails to capture the latent complexity of converging trajectories, speed differentials, and weather-induced rerouting [3–5]. While the shift from workload to complexity metrics provides deeper operational insights, a research gap remains: while many studies define what complexity *is*, few provide a framework for how to *estimate* it effectively using large scale historical data.

This paper addresses this gap by introducing a methodology for short-term complexity estimation. We frame the challenge as both a scalar regression problem and a time-series prediction task proposing a new variant of Recurrent Neural Network (RNN) architectures called temporal residual RNN (TR-RNN). By benchmarking these approaches, we establish a performance baseline for data-driven complexity estimation that accounts for the fluid nature of modern airspace.

The proposed methodology is designed for integration into real-time conflict detection tools and digital assistants for air traffic assignment, which also suitable for real time DAC task. By applying advanced data-driven techniques, this work offers a scalable tool for managing the inherent disorder of modern traffic flows, particularly in the context of flexible sector selections.

It is important to distinguish between two operationally different prediction regimes that this work addresses. Long-horizon forecasts (15–60 minutes) support strategic demand–capacity balancing and DAC, where airspace is reconfigured on timescales of tens of minutes to an hour. Short-horizon forecasts (5–10 minutes), by contrast, are not intended to drive sector reconfiguration; rather, they target tactical decision support – flagging imminent complexity build-up to assist conflict detection. They additionally enable rapid what-if evaluation across candidate configurations in a DAC context, where the complexity of many overlapping sector options must be assessed near-instantaneously. The two regimes therefore call for different models and impose different requirements: the short-horizon task prioritises high update frequency and low latency, while the long-horizon task prioritises stability over an extended forecast window. As a concrete example, a forecast complexity spike in a given sector within the next 5–10 minutes could trigger an advisory in a controller's digital assistant, drawing attention to a sector where converging trajectories are about to elevate workload before the situation is visually apparent on the radar display. This earlier salience supports proactive rather than reactive conflict management.

The remainder of this paper is organized as follows:

This research was supported by the SESAR 3 Joint Undertaking (JU) under grant agreement No 101114686. UK participants in SMARTS receive funding from UK Research and Innovation (UKRI) under the UK government's Horizon Europe funding guarantee [grant numbers 10086651 (Lancaster University)] and 10091277 (NATS).

Section II reviews related work, and Section III introduces the complexity metric validated by ATCO practitioners. Section IV outlines the problem formulation and mathematical framework. In Section V, we present a case study of the Madrid Area Control Center (ACC) airspace and detail the experimental setup. Finally, Section VI discusses the results, followed by concluding remarks in Section VII.

II. Literature review

In recent years, research has pivoted from the traffic volume to the influence of traffic types. Consequently, the emphasis has shifted from simply measuring occupancy or entry counts, to evaluating the complexity of the traffic. Early efforts by Gianazza [6] utilized feed-forward neural networks to forecast workload based on historical sector operations, operating on the assumption that sector configuration changes serve as a proxy for ATCO stress levels. Djokic et al. [3, 7] discovered that, rather than relying solely on aircraft count, communication load, specifically frequency occupancy time and average radio duration, served as significant contributions of the controllers' subjective workload ratings.

Pérez Moreno et al. [8] have aimed to calculate the sector complexity based on the nature of its flows. The proposed methodology first performs a statistical analysis of individual flight plan data to assess air traffic flows and determine the complexity of the ATC sector. In addition, a machine learning approach is incorporated to create a dynamic framework. This led to a fairly accurate classification of the sectors based on their complexity.

Further advancements have introduced objective, flow-based indicators. Pérez Moreno et al. [9] address the limitations of subjective complexity metrics by developing a dynamic indicator that characterizes Air Traffic Control sectors based on the objective behavior of air traffic flows. Their methodology employs a chain of Random Forest algorithms to predict specific operational variables, such as flight trajectories, regulations, and delays, which are subsequently aggregated via a deterministic model to calculate the final sector complexity level. Using graph neural network method is an alternative approach for complexity metric calculation. Li et al. [10] proposed a multimodal adaptive spatio-temporal graph neural network (MAST-GNN) framework, which uses Graph Neural Networks (GNNs) to capture dependencies across large-scale sectors with a rigid configuration.

Research on new complexity metrics for both terminal airspace and en-route airspace is also emerging these years. Delahaye et al. [5] proposes a new air traffic complexity metric based on linear dynamical systems that quantifies the intrinsic disorder of four-dimension trajectories by fitting a local vector field to aircraft position and speed observations. By analyzing the eigenvalues of the resulting system matrix, the metric identifies traffic organization patterns such as convergence and divergence while accounting for future position uncertainties to accurately assess airspace congestion and controller workload. Lim et al. [11] introduces a data-driven framework for objectively quantifying air traffic complexity in the Terminal Maneuvering Area (TMA) by applying unsupervised machine learning to historical ADS-B data to analyze operational outcomes such as vectoring and holding patterns. Chen et al. [12] presents a new air traffic complexity assessment model that converts numerical radar data into images and employs a deep metric learning method based on Asymmetric distance, Aggregation loss, and Edge hard loss (AAEDM) to mitigate class imbalance. Yating et al. [13] developed a three-layer dynamic network framework that integrates spatiotemporal flight distributions, conflict interdependencies, and flow transmission patterns to characterize complexity in the TMA.

Despite this advancement on air traffic complexity, three primary limitations persist in the current studies. First, most research focuses on defining complexity metrics rather than the predictive task of estimating those metrics from raw historical data. Second, there is a lack of benchmarking between traditional tree-based models and complex neural networks, leaving the necessity of high-compute models unjustified. Third, though sophisticated graph neural network model [10] is already proposed to address the air traffic complexity prediction, it relies on stable configuration structure. While in the context of DAC, air traffic complexity need to be simulated for all possible sectors which might overlapped with each other, rigid network structure on configuration level is not a viable assumption for DAC. This work specifically addresses these gaps by performing a configuration-agnostic benchmarking study suitable for the DAC environment.

III. Complexity Metric

Within the framework of Air Traffic Complexity Management (ATCM), complexity is defined as a multidimensional construct that extends beyond simple aircraft counts. It characterizes the difficulty an Air Traffic Controller (ATCO) faces in maintaining safe and efficient operations under varying traffic conditions. In this study, we utilize the internal methodology developed by CRIDA to quantify the complexity of operational sectors, providing a more granular estimate of controller workload than traditional traffic counts metrics.

The complexity metric is implemented via the COMETA tool (Cognitive Model for the assessment of the workload of ATCO), a computerized model developed by CRIDA for ENAIRE and extensively applied in the SESAR program [4, 14]. Central to COMETA is the *eCOMMET* algorithm, which evaluates traffic demand by identifying and quantifying specific complexity factors for each flight.The algorithm operates on a 5-minute calculation window with a 1-minute shift, assigning a complexity value to each sector. A core innovation of *eCOMMET* is the concept of the **equivalent flight**, which modulates the contribution of each individual aircraft to the total complexity based on its operational characteristics and interaction with other flights. Formally, the total complexity ($CPLX_{total}$) is defined as:

$$CPLX_{total} = CPLX_{base} + a \cdot NSF + b \cdot EVOL + c \cdot SFI + d \cdot MIL + e \cdot OCE \quad (1)$$

Where the parameters and their respective weighting factors (a, b, c, d, e) include:

- **Base Complexity ($CPLX_{base}$):** A baseline value representing the minimum management effort required for any flight entering the sector.
- **Non-Standard Flow (NSF):** Complexity assigned to flights operating outside standard flows, which are less familiar to the ATCO.
- **Evolution ($EVOL$):** Differentiates between aircraft in level cruise and those ascending or descending, with higher weight attributed to the latter.
- **Standard Flow Interaction (SFI):** The core component of the model, accounting for interactions and potential conflicts within the sector's operational flow structure.
- **Military (MIL):** Additional complexity derived from the atypical nature of military operations.
- **Oceanic (OCE):** Complexity related to transoceanic flights, whose performance profiles in initial phases are often more constrained.

By calibrating these factors, a sector with an actual occupancy of 20 flights may result in a complexity value equivalent to 18 or 23 "equivalent flights" depending on the traffic mix. This yields a metric on the same scale as occupancy, facilitating an intuitive understanding of workload and enabling the use of standard capacity thresholds, such as the Peak and Sustained Capacity limits [2, 15].

This methodology has been consistently employed by CRIDA in recent years across various sectorizations, including real-world volumetries and experimental designs. As a result of these applications, a comprehensive dataset has been consolidated, mapping diverse traffic samples to their corresponding complexity values. This wealth of data provides a robust foundation for training machine learning models capable of learning these complex non-linear relationships. In the present work, we leverage this dataset to generate short-term complexity forecasts. Such predictions are vital for Trajectory Based Operations (TBO) and highly automated contexts, where dynamic traffic assignment and virtual controller assistants require proactive assessments of complexity to prevent sector overloads before they occur.

Among the diverse complexity formulations surveyed in Section II, we adopt the *eCOMMET* algorithm for three reasons. First, unlike many indicators that remain at the conceptual or single-study stage, *eCOMMET* has undergone experimental validation against controller-perceived workload [4], grounding the metric in observed ATCO behaviour rather than purely geometric or statistical proxies. Second, it has been operationally adopted by ENAIRE through CRIDA and applied across multiple SESAR exercises [4, 14], providing both institutional maturity and a consolidated multi-scenario dataset suitable for data-driven modelling. Third, the equivalent-flight construct expresses complexity on the same scale as occupancy count, which preserves interpretability for practitioners and allows direct reuse of established Peak and Sustained Capacity thresholds [2, 15]. This combination of validation, operational adoption, and interpretability makes it a pragmatic choice for a forecasting study oriented toward real-world ANSP integration.

IV. Methodology

A. Scalar regression problem

To investigate the nature of air traffic complexity prediction, we first formulate it as a baseline scalar regression problem. We develop a general model trained across all sectors in the catalog, where for any sector s at time t, the model takes as input the sector-specific features $\mathbf{X}_t^s \in \mathbb{R}^d$ and predicts the aggregated complexity $y_s(t+\Delta t) = \text{CPLX}_{\text{total}}(s, t+\Delta t)$ at future time horizons $\Delta t \in \{5, 10, 15, 30, 60\}$ minutes. This baseline approach treats each sector prediction independently without modeling spatial dependencies between sectors, enabling us to establish predictive accuracy benchmarks and conduct feature importance analysis to understand the fundamental characteristics of the prediction task. Let s denote a specific sector and $\mathcal{F}_s$ be the set of all flights passing through sector s during the analysis period. For each flight $f \in \mathcal{F}_s$, we define:

1) $t_{\text{en}}^f, t_{\text{ex}}^f$: The entry and exit times of flight f in sector s.
2) $\text{FL}_{\text{en}}^f, \text{FL}_{\text{ex}}^f$: The flight level (altitude) of flight f at entry and exit points.
3) V_s: The geometric volume of sector s (in km^3).

We classify the motion of each flight f based on its altitude change across the sector. A threshold of $\delta = 20$ FL (Flight Levels) is used to distinguish vertical phases from level flight, as agreed by ATCO. The flight type is defined as:

$$\text{type}(f) = \begin{cases} \text{climb}, & \text{if } \text{FL}_{\text{ex}}^f - \text{FL}_{\text{en}}^f > \delta \\ \text{descent}, & \text{if } \text{FL}_{\text{ex}}^f - \text{FL}_{\text{en}}^f < -\delta \\ \text{cruise}, & \text{otherwise} \end{cases} \tag{2}$$

The set of active flights in sector s at time t is:

$$\mathcal{A}_s(t) = \{f \in \mathcal{F}_s \mid t_{\text{en}}^f \le t \le t_{\text{ex}}^f\} \tag{3}$$

The total occupancy count at time t is:

$$\text{OCC}_s(t) = |\mathcal{A}_s(t)| \tag{4}$$

We decompose the occupancy into three flight phases:

$$\text{OCC}_s^{\text{climb}}(t) = |\{f \in \mathcal{A}_s(t) \mid \text{type}(f) = \text{climb}\}| \tag{5}$$

$$\text{OCC}_s^{\text{descent}}(t) = |\{f \in \mathcal{A}_s(t) \mid \text{type}(f) = \text{descent}\}| \tag{6}$$

$$\text{OCC}_s^{\text{cruise}}(t) = |\{f \in \mathcal{A}_s(t) \mid \text{type}(f) = \text{cruise}\}| \tag{7}$$

To normalize for varying sector sizes, we define traffic densities as:

$$\rho_s^i(t) = \frac{\text{OCC}_s^i(t)}{V_s}, \quad i \in \{\text{total}, \text{climb}, \text{descent}, \text{cruise}\} \tag{8}$$

where $\text{OCC}_s^{\text{total}}(t) = \text{OCC}_s(t)$.

To preserve the cyclical continuity of hourly patterns (e.g., 23:00 is close to 00:00), we apply sine-cosine encoding:

$$h_{\sin}(t) = \sin\left(\frac{2\pi \cdot \text{hour}(t)}{24}\right) \tag{9}$$

$$h_{\cos}(t) = \cos\left(\frac{2\pi \cdot \text{hour}(t)}{24}\right) \tag{10}$$

where $\text{hour}(t)$ is the hour of day (0-23) at time t. With the static feature *sector volume* V_s and the autoregressive feature $y_s(t-1)$, the input feature vector $\mathbf{X}_t^s \in$

TABLE I. Taxonomy of the 14 input features by category. Current-state features describe the instantaneous traffic picture at prediction time; autoregressive features encode short-term momentum from the preceding step.

Feature	Description	Type
$\mathrm{OCC}_s^{\mathrm{total}}(t)$	Total occupancy count	CS
$\mathrm{OCC}_s^{\mathrm{climb}}(t)$	Climbing occupancy	CS
$\mathrm{OCC}_s^{\mathrm{descent}}(t)$	Descending occupancy	CS
$\mathrm{OCC}_s^{\mathrm{cruise}}(t)$	Cruising occupancy	CS
$\rho_s^{\mathrm{total}}(t)$	Total traffic density	CS
$\rho_s^{\mathrm{climb}}(t)$	Climbing density	CS
$\rho_s^{\mathrm{descent}}(t)$	Descending density	CS
$\rho_s^{\mathrm{cruise}}(t)$	Cruising density	CS
$h_{\sin}(t)$	Hour-of-day (sine)	CS
$h_{\cos}(t)$	Hour-of-day (cosine)	CS
Day of Week	Categorical day index	CS
V_s	Sector volume	CS
$y_s(t-1)$	Previous complexity	AR
$\mathrm{OCC}_s(t-1)$	Previous occupancy	AR

CS = current-state, AR = autoregressive.

$\mathbb{R}^{14}$ comprises: (i) occupancy counts $\mathrm{OCC}_s^i(t)$ for $i \in \{\text{total}, \text{climb}, \text{descent}, \text{cruise}\}$; (ii) normalized traffic densities $\rho_s^i(t)$ for the same phases; (iii) temporal encodings $h_{\sin}(t)$ and $h_{\cos}(t)$ and Day of Week; (iv) sector volume V_s; and (v) the autoregressive term $y_s(t-1)$ and $\mathrm{OCC}_s(t-1)$. We intentionally include both raw counts and normalized densities despite their correlation to allow the model to learn their relative importance through feature analysis, which serves our investigative objective.

The feature set deliberately combines two categories of information:

Current-state features – the occupancy counts $\mathrm{OCC}_s^i(t)$, normalised densities $\rho_s^i(t)$, temporal encodings, and sector volume V_s – describe the instantaneous traffic picture available to the controller at the moment of prediction.

Autoregressive features – $y_s(t-1)$ and $\mathrm{OCC}_s(t-1)$ – supply short-term momentum by encoding the immediately preceding complexity and occupancy. This separation is intentional and central to our investigative aim: by exposing the model to both the static snapshot and the recent trajectory, the subsequent feature-importance analysis (Section VI-A) can reveal which category the tree-based models actually rely on. As the results show, the dominance of the autoregressive term over all current-state features is itself the key diagnostic finding, motivating the sequence-based TR-RNN.

Three tree-based models are investigated in this study, included Random Forest [16], XGBoost [17], and LightGBM [18]. All tree-based models use 100 estimators; gradient boosting methods use a learning rate of 0.1.

B. Temporal Residual Recurrent Neural Networks (TR-RNN)

In contrast to the scalar regression approach, which treats each time step as an independent prediction from a static feature vector, we propose a TR-RNN to explicitly model the temporal dynamics of air traffic complexity. Rather than relying on the hand-crafted feature set $\mathbf{X}_t^s \in \mathbb{R}^{14}$, this approach operates on the univariate complexity time series alone, allowing the model to learn relevant temporal patterns directly from the sequential structure of the data, and to test if temporal dynamics alone can outperform static state features. Unlike Residual Networks (ResNet) [19] that utilize skip connections to improve gradient flow across layers, our Temporal Residual approach reformulates the regression target to model the complexity change – essentially its derivative – over time.

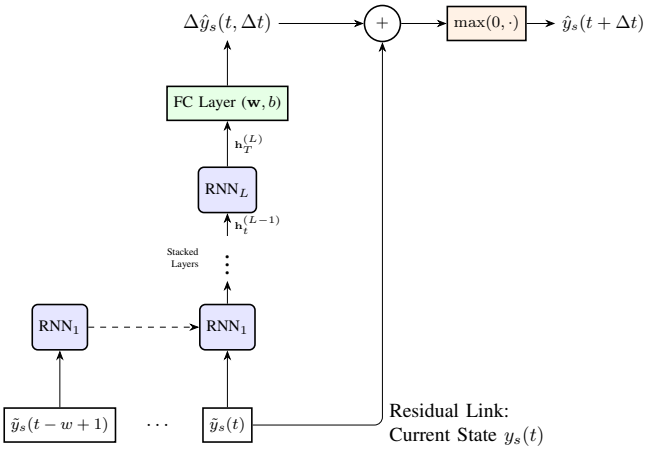


Figure 1: Architecture of the proposed TR-RNN.

1) Input Representation: For a given sector s, we construct a sliding window of w consecutive complexity observations. The input to the model at time t is:

$$\mathbf{Z}_t^s = [y_s(t-w+1),\ y_s(t-w+2),\ \ldots,\ y_s(t)]^\top \in \mathbb{R}^{w\times 1} \tag{11}$$

where $y_s(\tau) = \mathrm{CPLX}_{\text{total}}(s,\tau)$. Windows are constructed within each (sector, day) pair to prevent information leakage across sector or day boundaries.

2) Residual Formulation: Instead of predicting the absolute future complexity $y_s(t+\Delta t)$ directly, we formulate the prediction target as the residual change:

$$\Delta y_s(t,\Delta t) = y_s(t+\Delta t) - y_s(t) \tag{12}$$

The absolute prediction is then reconstructed as:

$$\hat{y}_s(t+\Delta t) = \max\Big(0,\ y_s(t) + \Delta\hat{y}_s(t,\Delta t)\Big) \tag{13}$$

where the non-negativity constraint ensures physically meaningful predictions. This residual formulation encourages the model to focus on predicting the *change* in complexity rather than its absolute level, which can improve learning stability particularly when the complexity signal exhibits varying baselines across sectors.

3) Architecture: We consider two recurrent architectures: Long Short-Term Memory (LSTM) [20] and Gated Recurrent Unit (GRU) [21]. Both models consist of a stacked RNN encoder with L layers and hidden dimension h, followed by a fully connected output layer. For the detailed network structure of LSTM/GRU modules, please refer to the references. Given the input sequence $\mathbf{Z}_t^s$, the model computes:

$$\mathbf{h}_t^{(l)} = \mathrm{RNN}^{(l)}\Big(\mathbf{h}_{t-1}^{(l)},\ \mathbf{h}_t^{(l-1)}\Big), \quad l = 1,\ldots,L \tag{14}$$

$$\Delta\hat{y}_s(t,\Delta t) = \mathbf{w}^\top \mathbf{h}_T^{(L)} + b \tag{15}$$

where $\mathbf{h}_T^{(L)}$ is the final hidden state of the last layer at the end of the sequence, and $\mathbf{w} \in \mathbb{R}^h$, $b \in \mathbb{R}$ are the parameters of the output layer. Dropout is applied between RNN layers for regularization when $L > 1$.

4) Scaling: Since the input (absolute complexity) and target (residual complexity) occupy different value ranges, we apply separate standardization. Let μ_X, σ_X and $\mu_{\Delta y}, \sigma_{\Delta y}$ denote the means and standard deviations estimated from the training set for the input and residual target, respectively. The input sequence is scaled as:

$$\tilde{y}_s(\tau) = \frac{y_s(\tau) - \mu_X}{\sigma_X} \tag{16}$$

and the residual target as:

$$\widetilde{\Delta y}_s(t, \Delta t) = \frac{\Delta y_s(t, \Delta t) - \mu_{\Delta y}}{\sigma_{\Delta y}} \tag{17}$$

At inference, predicted residuals are inverse-transformed before reconstruction of the absolute complexity.

5) Loss Function: We train the model using the Huber loss [22]:

$$\mathcal{L}_\delta\left(\widetilde{\Delta y},\ \widetilde{\Delta \hat{y}}\right) = \begin{cases} \frac{1}{2}\left(\widetilde{\Delta y} - \widetilde{\Delta \hat{y}}\right)^2, & \text{if } \left|\widetilde{\Delta y} - \widetilde{\Delta \hat{y}}\right| \le \delta \\ \delta\left(\left|\widetilde{\Delta y} - \widetilde{\Delta \hat{y}}\right| - \frac{1}{2}\delta\right), & \text{otherwise} \end{cases} \tag{18}$$

with $\delta = 1.0$. The Huber loss provides robustness to outliers compared to standard MSE, which is beneficial given the heavy-tailed distribution of complexity changes during unusual traffic situations. The model is optimized using Adam [23] with weight decay $\lambda = 10^{-4}$ for L_2 regularization, and gradient norms are clipped to 1.0 to stabilize training. A `ReduceLROnPlateau` scheduler halves the learning rate after 5 epochs without improvement in validation loss, and early stopping with a patience of 10 epochs is applied to prevent overfitting.

V. Case Study

A. Data Description

The dataset used in this study is sourced from the Madrid Area Control Center (ACC) as part of the SMARTS project validation exercise [2, 24]. The data spans 12 high-traffic days selected by operational practitioners between June 1st and July 8th, 2023, covering the primary operational window from 07:00 to 23:00 CET.

The dataset accounts for four distinct sector catalogs corresponding to varied operational scenarios, such as weekday/weekend and morning/afternoon configurations. Each scenario comprises approximately 230 sectors. To illustrate the scale of the data, the observation period for June 1st alone recorded more than 38,000 flights within the area of interest. In total, 2,132,904 rows of state-vector data were extracted for this analysis, providing a high-resolution foundation for complexity modeling.

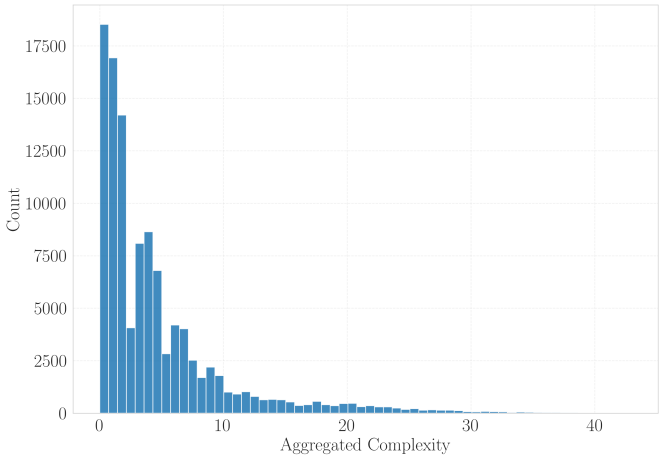


Figure 2: Distribution of aggregated complexity across all sectors and time intervals.

B. Exploratory Data Analysis (EDA)

Prior to model development, we performed an exploratory data analysis (EDA) to characterize the statistical properties and temporal dynamics of the complexity metric.

Figure 2 illustrates the distribution of aggregated complexity across all observed sectors. The histogram reveals a strongly right-skewed distribution (skewness $= 2.35$, kurtosis $= 6.66$) with a median value of 2.9 and a mean of 4.59 ($\sigma = 5.63$). Approximately 43.9% of all observations fall below a complexity value of 2, while only 11.4% exceed 10, and a mere 3.4% exceed 20. This heavy-tailed behavior underscores a significant challenge for forecasting: the majority of sector–time intervals experience low complexity, while the operationally critical high-complexity events, which necessitate tactical interventions, are relatively rare. Because high-complexity events are rare outliers, standard models tend to under-predict them. By using a residual formulation with Huber loss, the TR-RNN is less sensitive to the 'noise' of low-complexity routine traffic and can focus its gradient updates on the changes that lead to high-load situations.

Using the afternoon of June 1st as a representative case, Figure 3 presents the temporal evolution of the mean aggregated complexity. The mean complexity remains relatively stable between 15:00 and 22:00 CET (oscillating between 5 and 7), before declining sharply after 21:00 CET as traffic demand tapers off toward the end of the operational day. Notably, the wide $\pm 1\sigma$ band maintained throughout the peak hours reflects substantial inter-sector variability. This suggests that sector-level complexity is highly heterogeneous even within the same time window, motivating the inclusion of both sector-specific identifiers and cyclical time-of-day features in our forecasting models.

C. Evaluation metrics

To assess the performance of our proposed models, we employ two complementary regression metrics: the coefficient of determination (R^2) and Root Mean Squared Error (RMSE). These metrics provide both relative and absolute measures of prediction quality.

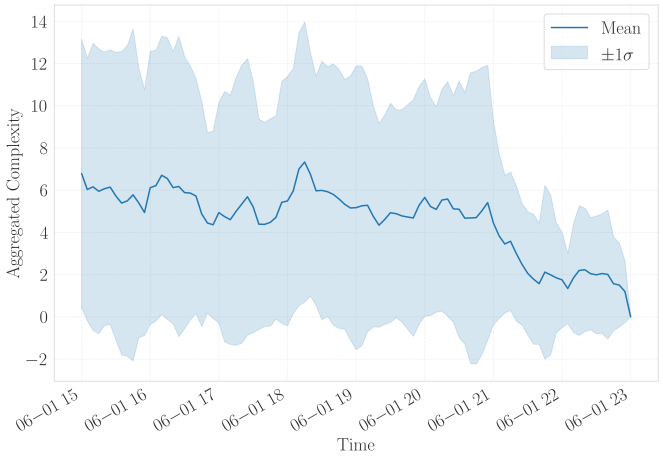


Figure 3: Temporal evolution of mean aggregated complexity (solid line) across all sectors, aggregated into 5-minute bins, with the shaded region denoting $\pm 1\sigma$.

Coefficient of Determination (R^2): The R^2 score quantifies the proportion of variance in the target variable explained by the model:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{19}$$

where y_i represents the true complexity value, $\hat{y}_i$ is the predicted value, $\bar{y}$ is the mean of true values, and n is the number of samples. An R^2 value of 1 indicates perfect prediction, while values closer to 0 suggest the model performs no better than a naive mean predictor. This metric is particularly useful for comparing relative performance across different model architectures.

Root Mean Squared Error (RMSE): RMSE provides an absolute measure of prediction error in the original units of aggregated complexity:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{20}$$

Unlike R^2, RMSE penalizes larger errors more heavily due to the quadratic term. In the context of Air Traffic Management, a large underestimation of complexity is significantly more hazardous than a series of minor fluctuations; therefore, RMSE serves as a critical proxy for the model's operational reliability during peak traffic periods.

For model selection during the cross-validation phase, we compute the average R^2 and RMSE across all $K = 5$ folds on $\mathcal{D}_{train}$. The final model performance is reported using both metrics evaluated on the held-out test set $\mathcal{D}_{test}$.

D. Experiment setup

Given the temporal dependencies inherent in air traffic flow management (ATFM) data, standard random shuffling is inappropriate as it would introduce data leakage (i.e., using future information to predict the past). Therefore, we employ a strictly chronological splitting strategy. The dataset is comprised of N unique days, sorted chronologically: $\mathcal{D} = \{d_1, d_2, \ldots, d_N\}$. We reserve the last two days of the dataset for the final model evaluation (Test Set), while the preceding $N-2$ days are used for model training and hyperparameter tuning (Training/CV Set).

$$\mathcal{D}_{test} = \{d_{N-1}, d_N\} \tag{21}$$

$$\mathcal{D}_{train} = \{d_1, d_2, \ldots, d_{N-2}\} \tag{22}$$

This approach simulates a real-world operational scenario where the model must predict future air traffic complexity based solely on historical data.

To assess model stability and selection during the training phase without overfitting, we utilize *Time Series Cross-Validation* (also known as Rolling-Origin forecasting [25]) with $K = 5$ splits on $\mathcal{D}_{train}$. Unlike standard K-Fold cross-validation, Time Series Split ensures that the training set always precedes the validation set in time. For the k-th split ($k \in \{1, \ldots, K\}$), the data is partitioned such that:

1) **Training indices** (T_k) and **Validation indices** (V_k) satisfy strict temporal ordering: $\forall t \in T_k, \forall v \in V_k : t < v$.
2) The training set size grows with each iteration, retaining historical context.

All computational experiments were performed on a workstation equipped with an AMD Ryzen Threadripper 3990X 64-Core Processor operating at 2.9 GHz base frequency (4.3 GHz boost) with 256 GB DDR4-3200 RAM. The implementation framework was developed in Python, utilizing `Torch` [26] for neural network architecture and `scikit-learn` for essential machine learning modules [27].

VI. Results and Discussion

A. Baseline Tree-Based Performance

Table II presents the predictive performance of three tree-based ensemble models across five prediction horizons. While all models achieve R^2 values exceeding 0.8, the absolute RMSE values reveal limitations of this baseline approach. Given that air traffic complexity typically ranges from 0 to 20, RMSE values of 2.0–2.8 indicating that current-state features alone provide limited predictive power.

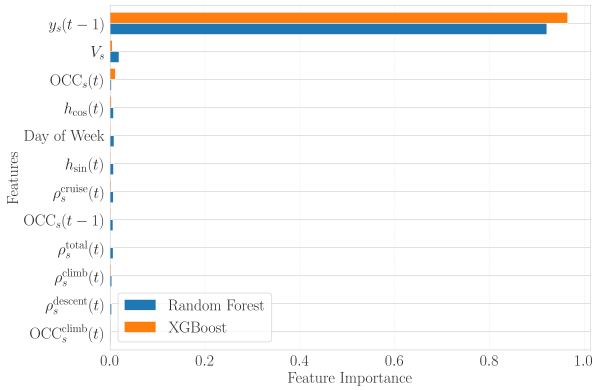
Feature importance analysis (Figures 4a and 4b) reveals the underlying cause of this limitation. Across both short-term (5 minutes) and long-term (60 minutes) horizons, the autoregressive term $y_s(t-1)$ dominates feature importance (0.7–1.0), while all other traffic and temporal features contribute marginally. This extreme dependence on the previous complexity value indicates that the model effectively learns a simple persistence forecast rather than capturing the underlying dynamics through current-state features.

The feature importance distribution remains remarkably stable across prediction horizons, with only minor increases in the relative contribution of current occupancy $\text{OCC}_s(t)$ and sector volume V_s at the 60-minute horizon. Notably, the decomposed occupancy by flight phase ($\text{OCC}_s^{\text{climb}}$, $\text{OCC}_s^{\text{descent}}$, $\text{OCC}_s^{\text{cruise}}$) and their normalized densities (ρ_s^i) show negligible importance, suggesting that static snapshots of traffic composition fail to capture the temporal evolution of complexity.

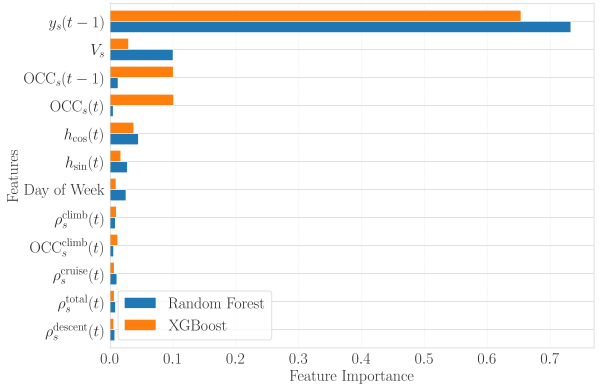
These findings strongly suggest that air traffic complexity prediction is inherently a time series forecasting problem

TABLE II. Baseline tree-based air traffic complexity predictive performance for different time horizon. Bold values indicate the best performance per horizon.

Model	**5 min**		**10 min**		**15 min**		**30 min**		**60 min**	
	RMSE	R^2	RMSE	R^2	RMSE	R^2	RMSE	R^2	RMSE	R^2
Random Forest	2.05	0.9	2.45	0.86	**2.60**	**0.85**	**2.63**	**0.84**	**2.68**	**0.83**
XGBoost	**1.99**	**0.91**	**2.43**	**0.87**	2.63	0.84	2.71	0.83	2.81	0.81
LightGBM	**1.99**	**0.91**	**2.43**	**0.87**	2.64	0.84	2.72	0.83	2.84	0.81



(a) 5-minute prediction horizon



(b) 60-minute prediction horizon

Figure 4: Feature importance comparison between Random Forest and XGBoost for short-term (5 min) and long-term (60 min) complexity predictions.

requiring historical sequential patterns rather than instantaneous state features. The dominance of the autoregressive term (y_{t-1}) in the tree-based models confirms that complexity is inherently path-dependent. Static snapshots fail to capture the rate of traffic accumulation, which justifies the necessity of the sequence modeling capability provided by the TR-RNN.

B. Temporal Residual RNN Performance

Table III summarizes the predictive performance of the univariate RNN models across all forecast horizons. We compare the baseline LSTM and GRU, which predict the absolute future complexity with Mean Square Error (MSE) loss directly, against their residual counterparts, which predict the change in complexity $\Delta y_s(t, \Delta t)$ with Huber Loss as described in Section IV-B.

Several observations emerge from these results. First, the baseline LSTM and GRU achieve nearly identical performance across all horizons, suggesting that the additional gating mechanism in LSTM provides no meaningful advantage over GRU for this prediction task. Both architectures achieve strong short-term accuracy ($R^2 = 0.91$ at 5 minutes) that degrades progressively with the forecast horizon, declining to $R^2 = 0.73$ at 60 minutes. This degradation is expected, as longer horizons require the model to extrapolate further beyond the observed sequence, and the univariate input provides no exogenous information about upcoming traffic events.

Second, the residual formulation yields a consistent improvement over the baseline at shorter horizons. At the 5-minute horizon, Residual-LSTM reduces RMSE from 1.98 to 1.84 and improves R^2 from 0.91 to 0.92. This improvement is attributable to the reformulation of the learning task: by predicting the change rather than the absolute level, the model avoids needing to reconstruct the baseline complexity from the input sequence, focusing its capacity on modeling the dynamics of change. Huber Loss's capability on providing robustness to outliers also shows in the accuracy improvement.

Figure 5 shows the learning curve for the Residual-LSTM model at the 5-minute horizon. Both training and test loss converge smoothly, with no evidence of overfitting. The test loss remains slightly below the training loss throughout, which is a known artifact of dropout regularization being active only during training.

Figure 5: Training and test loss (Huber) for the Residual-LSTM model at the 5-minute forecast horizon.

C. Comparative Analysis: Temporal Dynamics vs. Static Information

For **short-term prediction (5–10 minutes)**, the TR-RNN structure demonstrates superior performance. Remarkably, the

TABLE III. Air traffic complexity predictive performance for different time horizons with univariate RNN architectures. Bold values indicate the best performance per horizon.

Model	5 min		10 min		15 min		30 min		60 min	
	RMSE	R^2	RMSE	R^2	RMSE	R^2	RMSE	R^2	RMSE	R^2
Baseline-LSTM	1.98	0.91	2.43	0.86	2.68	0.83	2.99	0.79	3.37	0.73
Baseline-GRU	1.98	0.91	2.43	0.86	2.67	0.83	2.98	0.79	3.37	0.73
TR-LSTM	**1.84**	**0.92**	2.37	**0.87**	2.64	0.84	2.98	0.79	3.37	0.73
TR-GRU	**1.84**	**0.92**	**2.36**	**0.87**	2.64	0.84	2.98	0.79	3.37	0.73

univariate TR-GRU (RMSE 1.84) outperforms the multivariate XGBoost (RMSE 1.99) at the 5-minute horizon. This result is significant because the TR-RNN relies *solely* on the historical complexity sequence, whereas the tree-based models have access to the full suite of traffic composition and sector features. This suggests that in the short term, capturing the temporal dynamics and momentum of complexity is more valuable than accessing static snapshots of the sector state. The TR-RNN effectively models the derivative of the traffic flow, whereas the tree-based models struggle to look beyond the most recent observation (y_{t-1}).

However, for **medium to long-term prediction (15–60 minutes)**, the trend reverses. The tree-based models maintain better stability (RMSE 2.68 at 60 min) compared to the RNNs (RMSE 3.37). This crossover is driven by the information deficit in the univariate setting. As the forecast horizon expands, historical dynamics become less predictive of future states, and the model requires exogenous information (such as sector volume constraints) to bound the prediction. The tree-based models, despite their inability to model time, benefit from these "static" features which act as physical constraints on the prediction range.

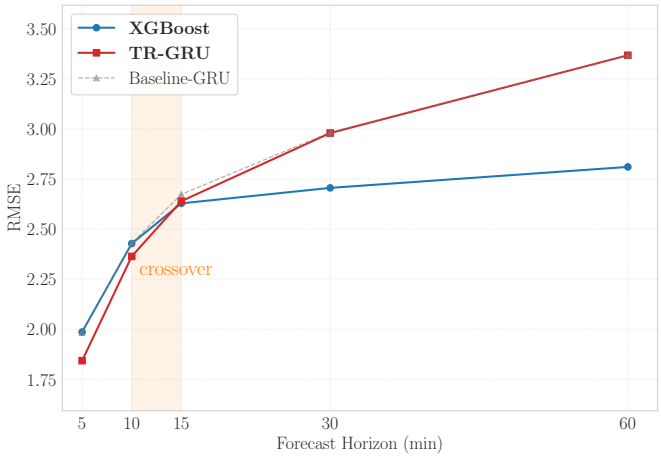


Figure 6: RMSE versus forecast horizon for the proposed TR-GRU, the best-performing tree model (XGBoost), and the Baseline-GRU.

Figure 6 visualises this regime change directly, plotting RMSE against forecast horizon for the best model of each family. The two curves intersect in the 10–15 minute region: the univariate TR-GRU holds a clear advantage at the shortest horizons, the models are effectively tied through the crossover band, and the tree-based model's access to exogenous static features yields progressively greater stability thereafter.

These findings suggest that the limitation of the current TR-RNN is not structural, but informational. Since the TR-RNN outperforms tree-based models in the short term using less information, it implies that the recurrent structure is fundamentally better suited for this problem. Integrating the additional traffic information (currently used by the tree models) into the TR-RNN would likely bridge the performance gap at longer horizons. By allowing the RNN to model the *temporal evolution* of these exogenous features—rather than treating them as static inputs—a multivariate TR-RNN could potentially combine the short-term dynamic accuracy of the residual structure with the long-term stability provided by traffic volume features.

D. Computational Efficiency and Real-Time Feasibility

To assess the practical applicability of the proposed models for real-time air traffic complexity forecasting, we benchmarked the inference latency of each trained model. The evaluation was conducted on a typical workstation environment (as introduced in Section V-D), measuring the average inference time per sample over 1,000 trials. We considered two scenarios: single-sample inference (batch size = 1), simulating a real-time streaming context, and batch inference (batch size = 256), representing high-throughput offline processing.

Table IV summarizes the computational performance of the five models.

TABLE IV. Comparison of Model Inference Latency (millisecond per Sample)

Model	Single (ms)	Batch (ms)
TR-LSTM	**0.28**	**0.005**
TR-GRU	**0.26**	**0.005**
XGBoost	2.14	0.010
LightGBM	10.82	0.035
Random Forest	35.21	0.164

The results demonstrate a significant disparity in computational efficiency between the deep learning models (LSTM, GRU) and the ensemble tree-based methods (Random Forest, LightGBM). Notably, the RNNs achieved sub-millisecond latency for single-sample predictions ($\approx 0.26 - 0.28$ ms), making them exceptionally well-suited for high-frequency, real-time decision support systems. This efficiency is likely attributable to the optimized matrix operations in the underlying PyTorch implementation, which effective leverage Single Instruction, Multiple Data (SIMD) instructions or GPU acceleration even at small batch sizes.

Among the tree-based models, XGBoost exhibited competitive performance with a latency of approximately 2.14 ms, which is sufficient for most air traffic management applications where updates occur on the order of minutes. However, Random Forest was considerably slower (≈ 35.21 ms per sample), likely due to the overhead of traversing a large number of deep decision trees sequentially or with limited parallelism during single-sample inference.

In the high-throughput scenario (batch size = 256), all models demonstrated excellent scalability. The TR-RNN models achieved remarkable efficiency, with per-sample inference times dropping to approximately 5 microseconds (0.005 ms). This represents a significant speedup over the tree-based ensembles, although XGBoost remained competitive at roughly 10 μs per sample. It is worth noting that the Random Forest latency (35.21 ms) is likely due to the specific implementation (CPU-bound tree traversal). While valid for the experiment, compiled tree implementations (like Treelite) could potentially narrow this gap.

VII. Conclusion

This study addressed the challenge of short-term air traffic complexity forecasting to support real-time operation task such as DAC and tactical conflict resolution. By benchmarking traditional tree-based ensembles against a novel Temporal Residual Recurrent Neural Network, we identified distinct regimes where temporal dynamics and static structural features respectively drive predictive performance.

Our results demonstrate that the proposed TR-RNN architecture, specifically the TR-GRU formulation, outperforms tree-based baselines (XGBoost, Random Forest) in short-term horizons (5 to 10 minutes). This superiority validates our hypothesis that short-term complexity is path-dependent; the recurrent architecture effectively captures the momentum and derivative of traffic evolution that static snapshots fail to represent. The residual learning formulation, coupled with Huber loss, proved robust against the heavy-tailed distribution of complexity data, allowing the model to focus on critical operational changes rather than dominant low-complexity baselines.

Conversely, for longer prediction horizons (15 to 60 minutes), tree-based models exhibited greater stability. Feature importance analysis revealed that as the temporal influence of the immediate past decays, the model must rely on exogenous physical constraints—such as sector volume and occupancy—to bound predictions. This highlights an informational deficit in the univariate RNN approach rather than a structural limitation.

From an operational perspective, the TR-RNN demonstrated exceptional computational efficiency, achieving sub-millisecond inference latencies suitable for real-time digital assistants and high-frequency loop operations. This scalability is essential for future DAC environments where complexity must be simulated across thousands of potential sector combinations instantaneously.

Future work will focus on bridging the gap between dynamic learning and static constraints. The logical next step is the development of a *multivariate* TR-RNN framework. By integrating exogenous state features (sector volume, flight phase distribution) into the recurrent structure, we aim to combine the short-term accuracy of residual dynamics with the long-term stability of physical traffic bounds. Additionally, extending this temporal framework to capture spatial information via GNN remains a promising avenue, provided the graph structures can accommodate the specific topology of dynamic airspace configuration.

References

[1] SESAR Joint Undertaking, "A proposal for the future architecture of the European airspace.," *Publications Office of the European Union*, 2019. DOI: https://doi.org/10.2829/309090.

[2] Horizon Europe, *Smart sectors: Optimising airspace design to meet air traffic demand efficiently*, 2023. DOI: 10.3030/101114686.

[3] J. Djokic, B. Lorenz, and H. Fricke, "Air traffic control complexity as workload driver," *Transportation research part C: emerging technologies*, vol. 18, no. 6, pp. 930–936, 2010.

[4] N. Suarez, P. López, E. Puntero, and S. Rodriguez, "Quantifying air traffic controller mental workload," in *Fourth SESAR Innovation Days*, vol. 220, 2014.

[5] D. Delahaye, A. García, J. Lavandier, S. Chaimatanan, and M. Soler, "Air traffic complexity map based on linear dynamical systems," *Aerospace*, vol. 9, no. 5, p. 230, 2022.

[6] D. Gianazza, "Forecasting workload and airspace configuration with neural networks and tree search methods," *Artificial intelligence*, vol. 174, no. 7-8, pp. 530–549, 2010.

[7] J. Djokic, "Investigation into air traffic complexity as a driver of a controller's workload," Ph.D. dissertation, Technische Universität Dresden, 2014.

[8] F. Pérez Moreno, V. Gómez Comendador, R. D.-A. Jurado, M. Z. Suárez, D. Janisch, and R. Arnaldo Valdés, "Dynamic sector characterisation model with the application of machine learning techniques," in *IOP Conference Series: Materials Science and Engineering*, IOP Publishing, vol. 1226, 2022, p. 012 018.

[9] F. P. Moreno, F. I. Rodríguez, V. F. G. Comendador, R. D.-A. Jurado, M. Z. Suárez, and R. M. A. Valdés, "Prediction of air traffic complexity through a dynamic complexity indicator and machine learning models," *Journal of Air Transport Management*, vol. 119, p. 102 632, 2024.

[10] B. Li, Z. Li, J. Chen, Y. Yan, Y. Lv, and W. Du, "MAST-GNN: A multimodal adaptive spatio-temporal graph neural network for airspace complexity prediction," *Transportation Research Part C: Emerging Technologies*, vol. 160, p. 104 521, 2024.

[11] Z. J. Lim, I. Dhief, D.-T. Pham, S. Alam, and D. Delahaye, "A data-driven framework for modelling complexity in terminal manoeuvring area," in *SESAR Innovation Days*, 2024.

[12] H. Chen, Z. Zhou, L. Wu, Y. Fu, and D. Xue, "Enhancing air traffic complexity assessment through deep metric learning: A cnn-based approach," *Aerospace Science and Technology*, vol. 160, p. 110 090, 2025.

[13] P. Yating, S. Di, W. Xiangxi, Y. Fuping, and J. Yuren, "Multi-layer dynamic network model of traffic flow in terminal area and complexity analysis," *Journal of Air Transport Management*, vol. 131, p. 102 881, 2026.

[14] J. Ibáñez-Gijón, D. Travieso, J. A. Navia, A. Montes, D. M. Jacobs, and P. L. Frutos, "Experimental validation of cometa model of mental workload in air traffic control," *Journal of Air Transport Management*, vol. 108, p. 102 378, 2023. DOI: 10.1016/j.jairtraman.2023.102378.

[15] EUROCONTROL, "Hourly entry count versus occupancy count relationship – robustness, calibration, application (ii)," Eurocontrol Experimental Centre, Brétigny-sur-Orge, France, Tech. Rep. EEC Note No. 16/07, 2007.

[16] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[17] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 785–794. DOI: https://dl.acm.org/doi/pdf/10.1145/2939672.2939785.

[18] G. Ke et al., "LightGBM: A highly efficient gradient boosting decision tree," *Advances in Neural Information Processing Systems*, vol. 30, 2017. DOI: https://dl.acm.org/doi/10.5555/3294996.3295074.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[20] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
[21] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.
[22] P. J. Huber, "Robust estimation of a location parameter," in *Breakthroughs in statistics: Methodology and distribution*, Springer, 1992, pp. 492–518.
[23] D. P. Kingma, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
[24] A. B. Romero et al., "Smarts solution for airspace design and configuration," in *SESAR Innovation Days*, 2025.
[25] L. J. Tashman, "Out-of-sample tests of forecasting accuracy: An analysis and review," *International journal of forecasting*, vol. 16, no. 4, pp. 437–450, 2000.
[26] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.
[27] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.