Enhancing Runway Configuration Assistant Model: The Role of Explainable AI for Model Interpretability

Nina Ebensperger*
George Washington University, Washington, DC, 20052, USA

Pouria Razzaghi† and Milad Memarzadeh‡
NASA Ames Research Center, Moffett Field, CA, 94035, USA

Peng Wei§
George Washington University, Washington, DC, 20052, USA

Krishna M. Kalyanam¶
NASA Ames Research Center, Moffett Field, CA, 94035, USA

In the Air Traffic Management (ATM) domain, machine learning-based systems are poised to be deployed as a decision-making aid to controllers. However, the inherent complexity of these models makes it difficult to understand the machine recommendations, which is critical for acceptance in a safety-critical domain such as ATM. This research explores the use of explainable Artificial Intelligence (XAI) techniques, such as SHapley Additive exPlanations (SHAP) and permutation of features, to enhance the interpretability of the models being exercised in ATM. As our use case, we chose runway configuration assistance (RCA) [1], a specific tool that predicts the optimal runway configuration to be deployed on an airport surface. We used data from three major US airports: Charlotte Douglas International Airport (CLT), Denver International Airport (DEN), and Dallas/Fort Worth International Airport (DFW). As expected, the data shows that wind components, cloud ceiling, and visibility are the most critical factors determining the tool output. In addition, at DEN, the hour of the day became the other contributor to the decision made by the RCA tool. This work reinforces the importance of interpretability for the acceptance of AI models, specifically in safety-critical domains such as ATM.

I. Introduction

As air traffic volume increases, efficient runway configuration management is a critical challenge in airport operations. Runway configurations, which are the decisions about which runways are available for arrivals and departures, have a direct impact on airport capacity, safety, and operational efficiency. Traditionally, these decisions are made based on historical traffic patterns, weather conditions, and the experience of air traffic controllers (ATCs). However, recent developments in machine learning (ML) have demonstrated the potential to dynamically help ATCs make decisions more efficiently by responding to real-world conditions and reducing delays. Although ML-based models offer performance benefits, they typically function as "black-box" systems, which hinders their reliability and application in safety-critical fields such as air traffic management (ATM) systems. To bridge this gap, this study explores the interpretability of an reinforcement learning (RL)-based Runway Configuration Assistant (RCA) model [1] using SHapley Additive Explanations (SHAP).

SHAP is a prominent technique in the domain of explainable artificial intelligence (XAI) that calculates the contribution of each input feature to the model's recommandations. The application of SHAP is designed to enhance the transparency, reliability, and usability of artificial intelligence/machine learning (AI/ML) models by providing actionable insights into the impact of various factors on model predictions. This study presents two major contributions:

*Machine Learning Intern, NASA Ames Research Center
†Principal Systems Research Engineer, Metis Technology Solutions, Inc., NASA Ames Research Center
‡AI technical lead, NASA Ames Research Center
§Associate Professor Department of Mechanical and Aerospace Engineering, AIAA Associate Fellow
¶NARI Deputy Director, Aviation Systems Division, NASA Ames Research Center, AIAA Associate Fellow

(1) the integration of a SHAP-based interpretive approach and feature permutation to measure the decision-making of the model and identify the most influential factors, and (2) illustrating the real-world usefulness of explainable artificial intelligence in the air traffic management to ensure that the model's suggestions are consistent with expert intuition and real-world feasibility. The findings of our study offer a more comprehensible decision support system, thus fostering trust and facilitating the adoption of AI-based solutions for airport management.

II. Literature Review

Applying ML tools for runway configuration decision support, introduces the application of novel ML and XAI methods for the optimization of runway usage within the scope of air traffic management (ATM) [1]. As airports face increasingly complex problems with dynamic traffic demand and weather conditions, optimizing runway configuration is crucial to facilitating operational efficiency and safety. The RCA model supports air traffic controllers by providing empirical-based decision support in real time, enabling them to respond more effectively to dynamic conditions and ultimately contributing to safer air transportation. Incorporating forecast data from the Localized Aviation Model Output Statistics Program (LAMP) and Terminal Area Forecasts (TAF) [2] has improved the accuracy of the RCA model in predicting the most suitable runway configurations [3]. The advanced modeling presented in the paper, based on discrete choice algorithms that enable predictions on a 15-minute basis, has a far-reaching impact on airport operations, particularly in major airports where effective decision-making is crucial for minimizing delays and maximizing safety [4–6]. Furthermore, the inclusion of state-of-the-art data enhancement methods helps create more varied training sets, which are essential for improving the robustness of runway detection systems under various operating conditions [7]. A key feature of the RCA model is its adherence to transparency, as outlined in the principles of Explainable AI. By providing explanations of the AI reasoning process, XAI enhances stakeholder trust and accountability, which are critical in high-stakes situations such as aviation. Interpretability enables air traffic controllers and airport managers to make accurate decisions, while also ensuring compliance with safety regulations and promoting harmonization between human operators and automation [8]. Despite significant progress, the application of XAI and machine learning in runway configuration remains limited by challenges in human-agent interaction and the ongoing need for refinement and validation of these technologies. Ethical concerns are also raised, including accountability and the reliability of AI-generated explanations in decision-making. All these challenges must be addressed to achieve the optimal effectiveness and reliability of ATM tools [9–12].

XAI methods are generally divided into two types: intrinsic interpretability, achieved through inherently transparent models, and post hoc interpretability, where external techniques explain complex, often non-linear models. In our study, we utilize post-hoc interpretability techniques, such as SHAP and feature permutation. SHAP values, based on game theory, assign importance scores to features, considering all possible combinations of features. This method has been recognized for its ability to offer global, local, and consistent explanations of feature importance. Due to its robust explanability, it is often utilized in other safety-critical fields such as medicine [13]. Although SHAP provides comprehensive information, it is computationally intensive, particularly in high-dimensional data spaces. Methods like LIME (Local Interpretable Model-Agnostic Explanations) create interpretable surrogate models by approximating the decision boundary around a specific instance of interest. By generating estimate explanations in local neighborhoods, LIME bypasses the challenges associated with Shapley values [14].

As a baseline, we utilized feature permutation. The concept of feature permutation has been a tried-and-true evaluation method for measuring feature importance across machine learning models since the early 2000s. Its implementation was popularized from Leo Breiman's Random Forests work in 2001 [15]. In which Breiman demonstrated that by randomly shuffling the values of a feature and measuring the reduction in model accuracy, one can determine the importance of each feature. Feature permutation is a stable evaluation metric that is both simple and intuitive. Although permutation is straightforward to implement and interpret, it can be computationally expensive for large datasets, as each feature must be permuted independently. Additionally, this method is sensitive to data noise and inter-feature correlations, which may obscure the true contribution of individual features. To address these limitations, we employ the SHAP method, which provides more robust and consistent explanations by accounting for feature interactions and reducing sensitivity to noise.

III. Methodology

A. Data Structure

The data used for developing and validating the runway configuration model were collected from multiple sources to represent the various conditions that affect airport operations comprehensively. It consists of historical data from major U.S. airports, specifically Charlotte Douglas International Airport (CLT) and Denver International Airport (DEN), which were gathered from databases including NASA's Sherlock Data Warehouse and the Federal Aviation Administration's (FAA) Aviation System Performance Metrics (ASPM) reports. Additional weather data, essential for predicting runway configurations, was obtained from METAR (Meteorological Aerodrome Reports), LAMP, and TAF systems [2]. The data span hourly intervals, covering critical variables such as wind direction, wind speed, visibility, and time of day, alongside airport-specific traffic demand and runway usage patterns. For each hour, a complete "state" was formed, integrating both real-time and forecasted weather indicators. The dataset was curated through meticulous preprocessing steps, including the discretization of wind and traffic variables, to create a structured, high-dimensional state space that enabled accurate and efficient learning within the model's offline RL framework [2]. The order and details of the final features are shown in Table 1.

Table 1 Summary Statistics for Each Dataset by Feature Group

Feature/Group	**DEN**					**DFW**					**CLT**				
	Min	Max	Median	Mean	Unique Value Counts	Min	Max	Median	Mean	Unique Value Counts	Min	Max	Median	Mean	Unique Value Counts
Hours 1 -24	···	···	···	···	0 or 1	···	···	···	···	0 or 1	···	···	···	···	0 or 1
Departures	0.0	0.978	0.1153	0.177	93	0.0	0.963	0.119	0.220	82	0.0	1.0	0.088	0.220	35
Arrivals	0.0	0.963	0.124	0.184	85	0.0	1.0	0.150	0.227	76	0.0	0.971	0.088	0.225	34
u (eastward)	-0.470	0.7484	0.0	0.006	653	-0.414	0.843	-0.021	-0.011	514	-0.374	0.395	0.0	0.004	540
v (northward)	-0.902	0.620	0.038	0.025	659	-0.6	0.707	0.077	0.050	556	-0.420	0.571	0.0	0.005	557
Cloud Ceiling	0.0	1.0	0.999	0.580	151	0.0	0.999	0.300	0.534	77	0.0	1.0	0.250	0.509	78
Visibility	0.0	1.0	1.0	0.926	20	0.0	1.0	1.0	0.948	20	0.0	1.0	1.0	0.908	20
IMC	···	···	0.0	0.097	0 or 1	···	···	0.0	0.196	0 or 1	···	···	1.0	0.20	0 or 1
VMC	···	···	1.0	0.90	0 or 1	···	···	1.0	0.804	0 or 1	···	···	1.0	0.7	0 or 1

The initial 24 attributes correspond to the 24 hours in a day, and each hour is encoded as a binary attribute (0 or 1). The 24 attributes are then divided into four separate intervals of 6 hours each, which will be discussed in the following section. Features 24 and 25 are the predetermined arrival and departure times, respectively. Interestingly, most of their values are centered around 0, as indicated by their mean and median statistical values, reflecting a lack of uniform distribution. Features 26 and 27 are associated with the u and v components of eastward and northward wind, respectively. For SHAP value analysis purposes, these two features are combined into a single variable that indicates both wind strength and direction. Like the departure and arrival features, the values of these variables are concentrated around 0. Features 28 and 29 are visibility and cloud ceiling, and both have distributions with values that are more centered near 1. Finally, features 30 and 31 are binary indicators of meteorological conditions, with the first being Instrument Meteorological Conditions (IMC) and the second Visual Meteorological Conditions (VMC). We merge these into a single variable to represent the overall meteorological conditions that affect runway configuration decisions.

B. SHAP Method

Building on the general idea of feature permutation, SHAP enhances it by being more rigorous and theoretically grounded. One way it enhances feature permutation is by presenting both local and global explanations by calculating the marginal contribution of each feature across all possible combinations of features. Unlike permutation, SHAP considers interactions between features and ensures consistency [16]. Based on game theory, each characteristic is treated as a "player" in a coalition game [17]. The goal of which is to assign each feature its fair contribution to the model's output. It does so by calculating the Shapley values. To calculate the Shapley values, the algorithm computes the average marginal contribution of each feature in all possible combinations of features [17]. The SHAP Eq. (1) is as follows.

$$g(z') = \phi_0 + \sum_{j=1}^{M} \phi_j z'_j \tag{1}$$

where,

- $g(z')$ represents the SHAP explanation model for an instance where z' indicates the presence or absence of features.

- ϕ_0 is the baseline value or the expected output when all features are absent.
- $\sum_{j=1}^{M} \phi_j z'_j$ is the summation of SHAP values, where M is the total number of features, ϕ_j is the contribution of the feature j, and z'_j indicates whether the feature j is present (1) or absent (0).

In the calculation of the SHAP value, it is simulated that a feature is playing (present) or not (absent) [16]. Equation (1) can be simplified since all features are present in x', each z'_j is equal to 1, so the summation term simplifies to the sum of all the feature contributions ϕ_j. The formula simplifies because x' represents an instance with all the features present, so each indicator variable z'_j is 1, making the summation term simply $\sum_{j=1}^{M} \phi_j$ [16].

SHAP has three notable properties:

1. Local Accuracy

$$\hat{f}(x) = g(x') = \phi_0 + \sum_{j=1}^{M} \phi_j x'_j \tag{2}$$

where $\hat{f}(x)$ is the model prediction for the input x, and $g(x')$ is a function that approximates this prediction using SHAP values. Here, ϕ_j represents the SHAP value for feature j in the model $\hat{f}$ for the given instance x, and x'_j is an indicator that specifies whether feature j is present or absent in the coalition [16]. Having set all $x'_j = 1$, the formula distributes the total prediction across all features. This gives the final model prediction as seen in Eq. (3)

$$\hat{f}(x) = \phi_0 + \sum_{j=1}^{M} \phi_j \tag{3}$$

Therefore, the sum of all SHAP values ϕ_j, when all features are present, is equal to the model prediction. This aligns with the efficiency property of Shapley values, where the total model prediction is exactly distributed among all features [16].

2. Missingness

The missingness property states that if a feature is missing (i.e., $x'_j = 0$), then the corresponding SHAP value is zero [16].

$$x'_j = 0 \Rightarrow \phi_j = 0 \tag{4}$$

where x'_j represents the elements of the coalition vector, where a value of 0 indicates the absence of a feature. Ideally, for a complete explanation of an instance, all x'_j values should be 1, meaning all features are present. If $x'_j = 0$ for any feature j, the feature is then considered missing for this instance. Unlike standard Shapley, SHAP enforces the missingness property. Lundberg refers to it as a "minor bookkeeping property", meaning that while missing features could theoretically have arbitrary SHAP values without affecting local accuracy. This is due to the values being multiplied by $x'_j = 0$. Missingness ensures that missing features receive a Shapley value of 0 [16].

3. Consistency

Let $\hat{f}_x(z') = \hat{f}(h_x(x'))$ and let z'_{-j} indicate that $z'_j = 0$. For any two models f and f' that satisfy:

$$\hat{f}'_x(z') - \hat{f}'_x(z'_{-j}) \geq \hat{f}_x(z') - \hat{f}_x(z'_{-j}) \tag{5}$$

for all inputs $z' \in \{0,1\}^M$, then:

$$\phi_j(\hat{f}', x) \geq \phi_j(\hat{f}, x) \tag{6}$$

Where,

- f and f' are the two models being compared,
- $\hat{f}_x(z')$ and $\hat{f}'_x(z')$ represent the prediciton outputs for the models f and f' with input z',
- $\phi_j(\hat{f}, x)$ and $\phi_j(\hat{f}', x)$ are the SHAP values for feature j in models f and f' respectively, and
- M is the total number of features in the dataset.

The consistency property states that if a model changes so that the marginal contribution of a feature increases or remains the same (regardless of other features), then the SHAP values for the feature should also increase or stay the same [16]. From this consistency property, the Shapley properties of Linearity, Dummy, and Symmetry naturally follow, as detailed in the appendix of Lundberg and Lee [17].

In sum, Shapley values quantify the importance of individual features by showing how the prediction changes when a feature is included or excluded from the model, ensuring a fair distribution of credit for the prediction. It does so by ensuring that each feature's contribution is distributed based on how much it changes the model's output when included with other features. It examines how adding or removing a feature affects the prediction for all possible combinations of features. Furthermore, SHAP ensures that the sum of individual feature contributions equals the model's prediction, providing a clear explanation of how the input features influence the output. SHAP can work with any type of model and, as such, can be used on black-box models like neural networks and SVMs [17].

C. SHAP-Based Feature Attribution for the RL Model

To interpret the decisions made by our reinforcement learning (RL) model, the `shap.Explainer()` interface from the SHAP library [17, 18] was utilized. This high-level interface automatically selects an appropriate explanation method based on the type of model and input data. Since our RL model is a black-box neural network and not tree-based, SHAP defaults to the *Kernel-SHAP* method.

Kernel-SHAP is a model-agnostic approach to estimating SHAP values by simulating the removal and inclusion of features through the use of feature coalitions. Each coalition is represented as a binary vector $\mathbf{z}' \in \{0, 1\}^M$, where $z'_i = 1$ indicates that the i-th feature is included, and $z'_i = 0$ indicates that the feature is excluded [18].

To simulate feature exclusion, Kernel-SHAP replaces the excluded features with values drawn from a background dataset (typically the training data). This substitution removes the predictive power of the feature while preserving the input dimensionality. The SHAP value estimation proceeds by [17]:

1) Sampling multiple binary coalition vectors $\mathbf{z}'$.
2) Using a mapping function $h_x(\mathbf{z}')$ to reconstruct a full input vector:

$$h_x(\mathbf{z}')_i = \begin{cases} x_i & \text{if } z'_i = 1 \\ \tilde{x}_i & \text{if } z'_i = 0 \quad (\tilde{x}_i \text{ sampled from background data}) \end{cases} \tag{7}$$

3) Querying the original model $f(h_x(\mathbf{z}'))$ to get outputs for the sampled inputs.
4) Weighting each coalition using the SHAP kernel:

$$\pi_x(\mathbf{z}') = \frac{M - 1}{\binom{M}{|\mathbf{z}'|} \cdot |\mathbf{z}'| \cdot (M - |\mathbf{z}'|)} \tag{8}$$

5) Fitting a weighted linear model $g(\mathbf{z}') = \phi_0 + \sum_{j=1}^{M} \phi_j z'_j$ to approximate $f(h_x(\mathbf{z}'))$.

The estimated coefficients ϕ_j from this regression represent the SHAP values, which quantify the local contribution of each feature to the model's prediction.

In terms of visualizations, SHAP can be clearly explained using scatter plots, beeswarm plots, water-fall plots, and heatmaps [13]. Such methods were utilized in the paper *Explainable Machine Learning for the Prediction and Assessment of Complex Drought Impacts*, where XGBoost (an ML algorithm based on gradient boosting that builds an ensemble of decision trees to improve predictive performance) was used for predictions and SHAP for explainability, applied to predict drought impacts on agriculture, the economy, public health, and other related areas [13]. Further visualizations include dependence plots and force plots. Here, dependence plots show the SHAP value of one feature against the feature's actual value, illustrating the relationship between the feature's contribution and its actual value. Additionally, force plots visualize the impact of each feature on a single prediction, showing how features push the prediction towards higher or lower values. For our paper, we utilized waterfall plots, beeswarm plots, and bar plots.

Post-hoc Grouping of SHAP Values

After computing the SHAP values, a post-processing step is applied to regroup individual feature attributions into interpretable domain-level groups (e.g., "visibility", "meteorology"). This regrouping is performed using the following Alg. 1. This regrouping step improves interpretability by aligning SHAP attributions with domain-relevant feature clusters used throughout the rest of the analysis.

D. Feature Permutation

As mentioned in an earlier section, the baseline explanation will be provided by feature permutation. Feature permutation involves shuffling the values of a feature within a dataset and measuring the change in model performance to assess the feature's importance. The standard pipeline is that the baseline accuracy must be calculated after the model has been run. Then, each feature is permuted across the batch. After which, the error on the permuted batch is calculated. This will be used to compute the importance of the feature by finding the difference between the baseline error and the permuted batch error.

IV. Model Architecture

The utilized model architecture to optimize runway configurations is structured around an offline RL framework, employing a Conservative Q-Learning (CQL) algorithm [2]. This approach was chosen to leverage historical data and avoid the practical challenges of gathering real-time data in air traffic management, particularly under rapidly changing environmental conditions. By using CQL, the model effectively addresses the distributional shift problem, where the policy learned from historical data may differ significantly from operational policies used by air traffic controllers (ATCo). The CQL-based framework has been shown to reduce overestimation errors associated with out-of-distribution actions, enabling the model to operate in realistic, actionable configurations that align with safe and efficient airport operations [1].

Algorithm 1: RegroupSHAPValuesWithHours

```
Input: shap_values: SHAP value array of shape (num_samples, num_features)
test_states: Input feature matrix (e.g., test set)
name_mapping: Dictionary mapping group names to feature indices
Output: grouped_shap_values, grouped_data

Initialize variables ;
num_samples ← shap_values.shape[0] ;
grouped_shap_values ← zeros(num_samples, |name_mapping|) ;
grouped_data ← zeros(num_samples, |name_mapping|) ;
group_idx ← 0 ;
foreach (group_name, indices) in name_mapping.items() do
    Convert indices to NumPy array ;
    indices ← array(indices, dtype=int) ;
    if 'hours' in group_name then
        Compute mean SHAP values over hourly indices ;
        grouped_shap_values[:, group_idx] ← mean(shap_values[:, indices], axis=1) ;
        for (i, hour) in enumerate(indices) do
            Mark which hour is active in each instance ;
            present_hours ← (test_states[num_samples, hour] == 1) ;
            Assign active hour index to grouped data ;
            grouped_data[present_hours, group_idx] ← hour ;
    else if group_name == 'visibility' then
        Handle visibility as continuous input ;
        grouped_shap_values[:, group_idx] ← mean(shap_values[:, indices], axis=1) ;
        grouped_data[:, group_idx] ← mean(test_states[:, indices], axis=1) ;
    else if group_name == 'wind UV' then
        Handle wind components ;
        grouped_shap_values[:, group_idx] ← mean(shap_values[:, indices], axis=1) ;
        grouped_data[:, group_idx] ← mean(test_states[:, indices], axis=1) ;
    else
        if group_name == 'meteorology' then
            Handle binary meteorological indicators ;
            grouped_shap_values[:, group_idx] ← mean(shap_values[:, indices], axis=1) ;
            grouped_data[:, group_idx] ← mean(test_states[:, indices], axis=1) ;
        else
            Default aggregation for other continuous features ;
            grouped_shap_values[:, group_idx] ← mean(shap_values[:, indices], axis=1) ;
            grouped_data[:, group_idx] ← mean(test_states[:, indices], axis=1) ;
    Move to the next group ;
    group_idx ← group_idx + 1 ;
return grouped_shap_values, grouped_data ;
```

To enhance adaptability, the model incorporates forecast data from LAMP and TAF, which augments the state-space and utility function to accommodate rapid changes in meteorological conditions [2]. Specifically, the model's state space includes both real-time and forecasted wind speed, wind direction, and relevant meteorological variables, ensuring a comprehensive view of the environmental factors that influence runway configurations. The forecast integration is central to mitigating one of the challenges observed in previous models: handling quick wind changes that would otherwise cause frequent and operationally costly configuration shifts [6]. The model also uses a continuous state-space representation, which enables scalability and allows the integration of additional, longer forecast periods without incurring prohibitive computational costs.

Further refinements in the model's utility function help address operational requirements such as throughput, go-arounds, and configuration stability [2]. In particular, a penalty is imposed for rapid configuration changes, reflecting ATCo's preference for maintaining stable configurations in the face of short-lived environmental fluctuations. The inclusion of forecast-driven elements in the utility function aligns the model's recommendation with ATCo practices, enhancing the model's relevance in operational settings. Continuous state-space also facilitates the future expansion of features, including traffic load and convective weather data, providing a robust foundation for adapting the model to more complex air traffic management scenarios [1].

V. Results

A. Model Performance

This section presents an analysis of the SHAP values calculated on the results of our model, as applied to the three datasets. The SHAP values explain the model's prediction of an airport's configuration given a specific state. The model's performance, as measured by the F1-score and accuracy, demonstrates its high predictive power across the datasets. The following analysis also explores how SHAP values reveal the contribution of each feature to the model's output globally and locally. The model's accuracy and F1 score are highest when applied to the CLT airport data, in which the F1-score is 0.814, and the accuracy is 81.78%. It remains high when used on the DFW airport data, with an F1-score of 0.777 and an accuracy of 80.22%. The model performs its lowest with the DEN airport data with an F1-score of 0.512 and an accuracy of 52.97%. The model demonstrates strong performance in two airports; however, its comparatively lower effectiveness with DEN airport data necessitates a deeper investigation into feature importance and inter-feature interactions. In this analysis, SHAP values are utilized to ensure that the model's predictions are grounded in industry expectations.

B. Model Explainability

1. Global Explanation

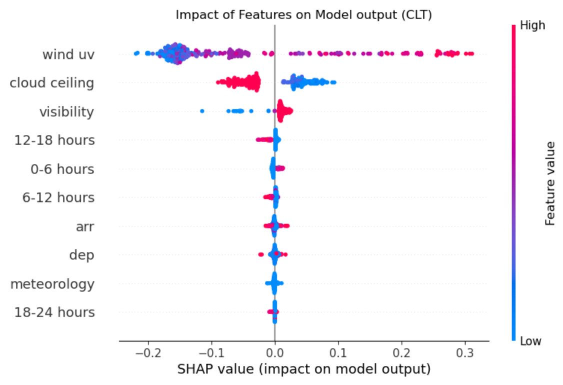
The SHAP value computation was run on a test set of 300 instances. To visualize the feature importance on a global scale, SHAP's beeswarm plot is utilized for all 300 instances and is demonstrated in Fig. 1. The boxplots for the SHAP values are included as well in Fig. 2 to further visualize the density of instances that had similar or equal SHAP values. In Fig. 1, the beeswarm plots for each airport dataset are depicted. In which the features are ranked from the highest to lowest contribution based on the SHAP value, which is denoted on the x-axis. The x-axis indicates the magnitude and direction of the feature's influence on the model prediction. Each point corresponds to a mean SHAP value for a single instance. Furthermore, the color gradient represents the value of the feature for each instance. Conditioning on a specific feature value enables analysis of the model's local behavior and its sensitivity to that feature.

Fig. 1a illustrates the model's predictive dynamics when applied to CLT runway configuration prediction. Notably, the feature wind UV is the most significant, followed by cloud ceiling and visibility. All other features have a minor impact on the model's output. Analyzing the model's sensitivity to wind UV reveals a clear pattern. As the wind UV value decreases, its attribution increasingly drives the model away from selecting the configuration given. Meanwhile, the higher the value of wind UV, the greater its influence on the model's selection of a given configuration.

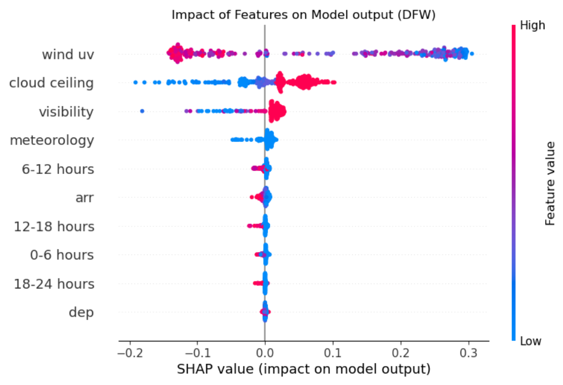
The second most significant feature, cloud ceiling, exhibits a clear pattern. The value of the cloud ceiling affects how the model is influenced, given a specific configuration. Instances with higher cloud ceiling values have feature attributions that shift the model's prediction away from the corresponding configuration. Having a lower value creates attributions that change the model's predictions towards the given configuration.

The feature with the subsequent significant effect on the model's output is visibility; however, this effect is minimal, with outliers showing a higher impact on the model's decision. According to industry practices, it is expected that wind UV will be the most influential feature, along with cloud ceiling and visibility. Therefore, the model is performing as expected on the CLT airport data.

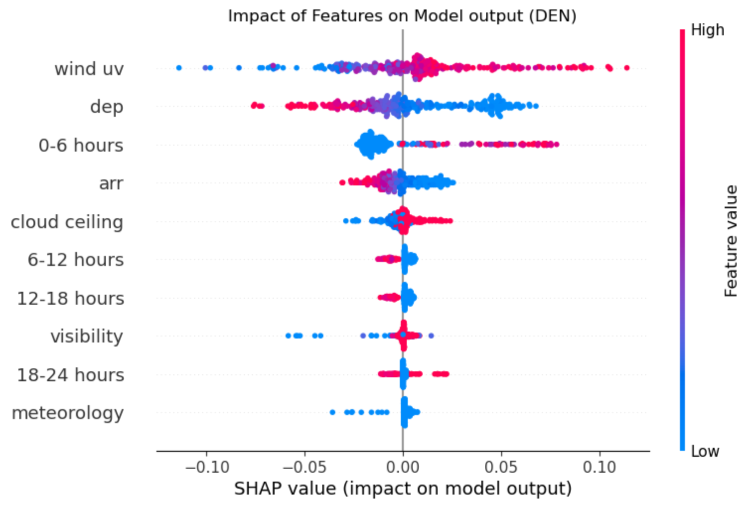
The model's internal reasoning, as reflected by the SHAP values, was closely aligned when executed on DFW and CLT airports. Specifically, wind, UV, cloud ceiling, and visibility remained the three most influential features, aligning with industry standards. However, when the model was executed on DFW airport data, a consistent trend emerged in the model's response to wind UV values. wind UV displayed a directional effect, where lower values consistently reduced the likelihood of the model selecting a given configuration. The attribution trajectory also revealed an incline as its values increase. In Fig. 1b, wind UV exhibits high variability in its contribution to the model's output when evaluated



(a) CLT



(b) DFW



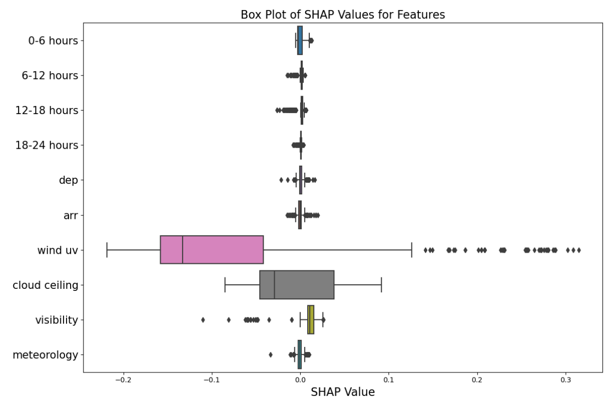
(c) DEN

Fig. 1 Visualization of the SHAP values for different airports

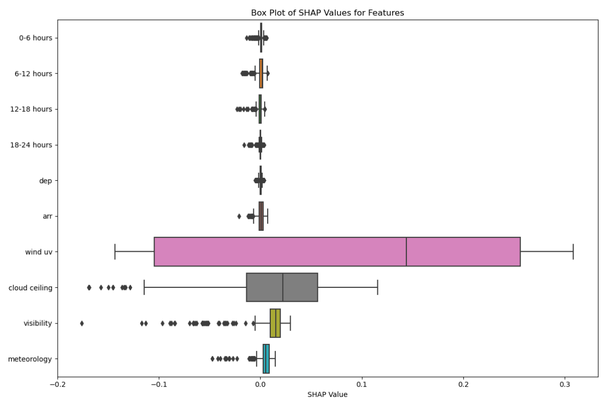
on DFW airport data. Additionally, the second most salient feature, cloud ceiling, has a clear pattern in the data. The higher the value, the more likely it is to influence the model's output. Higher values of the feature exhibit a positive attribution, increasing the model's likelihood of selecting the given configuration. Conversely, as cloud ceiling's value decreases, its influence shifts the model's output against the configuration. The results from the model deployed on DEN airport data (Fig. 1c) deviate from the current trend established by its preceding executions. While wind UV remains the dominant predictive feature, this is the only airport where the model output is significantly affected by the hour of the day. This suggests an underlying factor in model decision-making in the ATCo of the Denver airport, where unique features are considered in their operations.

2. *Local Explanation*

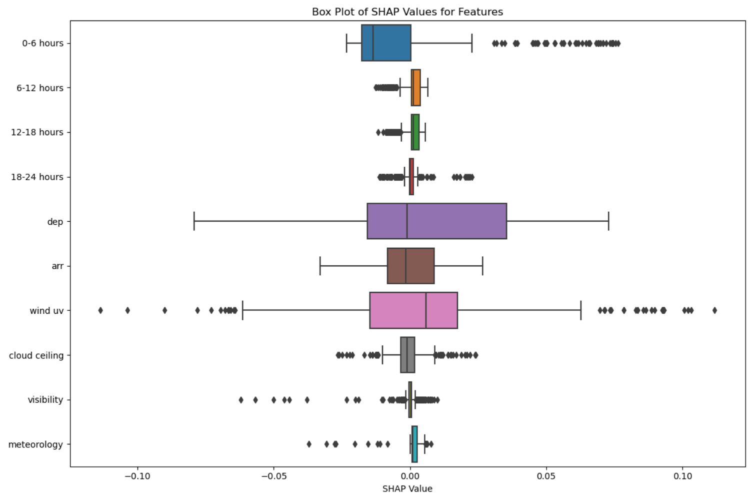
One of the advantages of SHAP is its ability to provide local explanations, which offer an interpretation of individual predictions. In the context of a SHAP waterfall plot, the explanation begins at $E[f(x)]$, which is the expected value of the model output across the entire dataset. This expected value serves as a baseline, representing the model's prediction in the absence of any knowledge about the specific instance. Each step in the waterfall plot represents a feature-level attribution, quantifying how much a particular feature contributes to shifting the prediction away from the baseline. These attributions can be positive or negative, as is also evident in SHAP's global explanation, depending on the feature's influence on the model's decision given a particular configuration. The plot culminates in $f(x)$, which displays the actual model output after all feature attributions have been taken into account. $F(x)$ corresponds to the model's output



(a) CLT



(b) DFW



(c) DEN

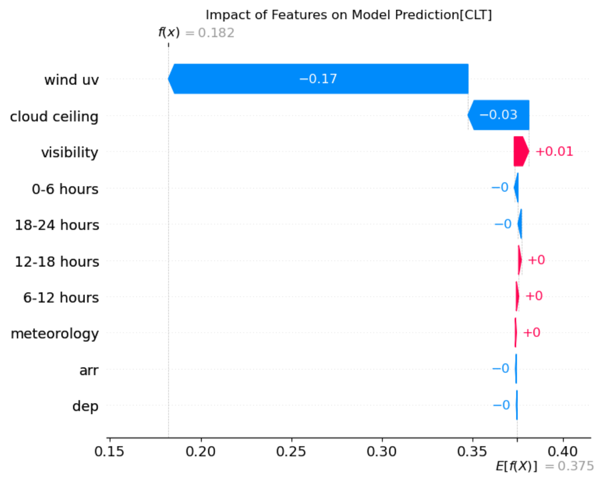
Fig. 2 Boxplot visualization of the SHAP values for different airports

given the specific input features of the instance. Starting at the baseline $E[f(x)]$, each feature's SHAP value acts as an additive adjustment that pushes the prediction toward the final decision $f(x)$ based on the instance's state.

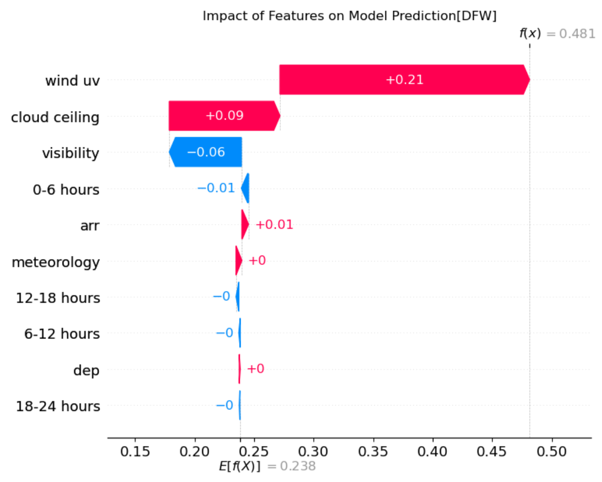
Figure 3 provides valuable insights into how various meteorological and operational factors influence the model's predictions for runway configuration selections. The analysis highlights the relative importance of features such as wind UV components, visibility, and cloud ceiling, among others, in shaping the model's decisions. In Fig. 3a, the value $f(x) = 0.182$ suggests that the model is not likely to choose the current configuration in this instance. While the specific configuration being decided against is unclear, as per the configuration of SHAP's library, the features influencing this decision are evident. The wind UV is the most significant factor that negatively influences the model output against the given configuration, reducing its selection probability. The cloud ceiling also has a negative impact, while visibility exerts a negligible positive influence on the decision. All other features, including time periods ("0-6 hours," "12-18 hours," and "18-24 hours"), are neutral and have an insignificant effect on the model's decision for CLT airport data.

In Fig. 3b, the model is moderately likely to predict a configuration with $f(x) = 0.481$, heavily influenced by wind UV, which positively shifts the model output to a higher probability. Visibility positively contributes to the decision, while the cloud ceiling acts as a counterforce, moving the model slightly away from the given configuration. Interestingly, meteorological conditions also support the model's recommendation, reinforcing the importance of environmental factors at DFW airport. Other features exhibit minimal or neutral impact, suggesting that wind and visibility are the primary factors influencing the decision-making process for DFW airport.

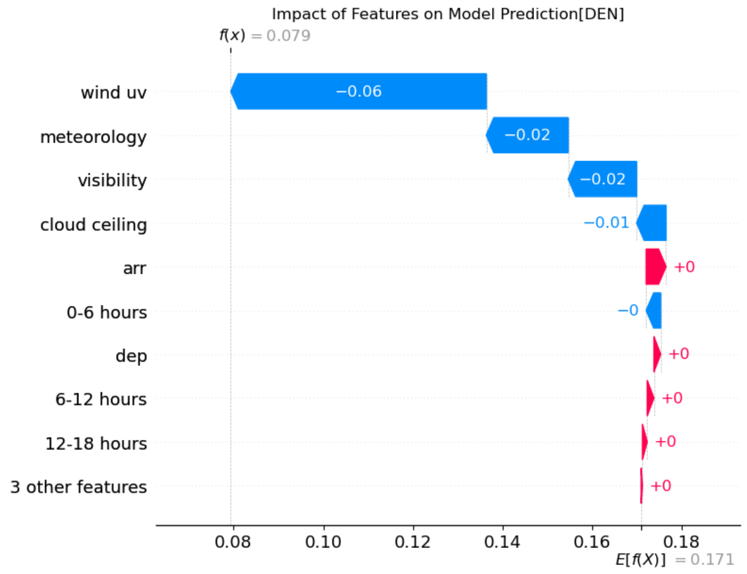
While the results from DFW and CLT airport data align with expected industry practices, and as noted in the previous



(a) CLT



(b) DFW



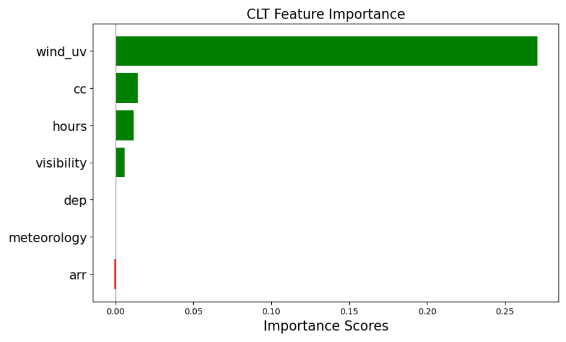
(c) DEN

Fig. 3 Waterfall plot of the SHAP values for single instance on different airports

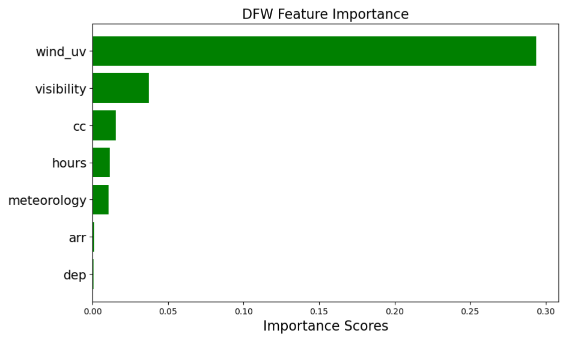
section, when the model was applied to DEN airport data (Fig. 3c), the results veer from industry norms. Specific, unique operational patterns influence the model. The value $f(x)$ indicates a shift in feature impact compared to CLT and DFW. Wind UV remains influential. However, hours become less critical, likely due to the previously discussed possible variation in operational practices at the Denver airport. Short-term changes, represented by operational nuances and wind conditions, play a significant role in determining the outcome. The airport's unique practices may necessitate customized models to capture these dynamics.

3. Feature Permutation

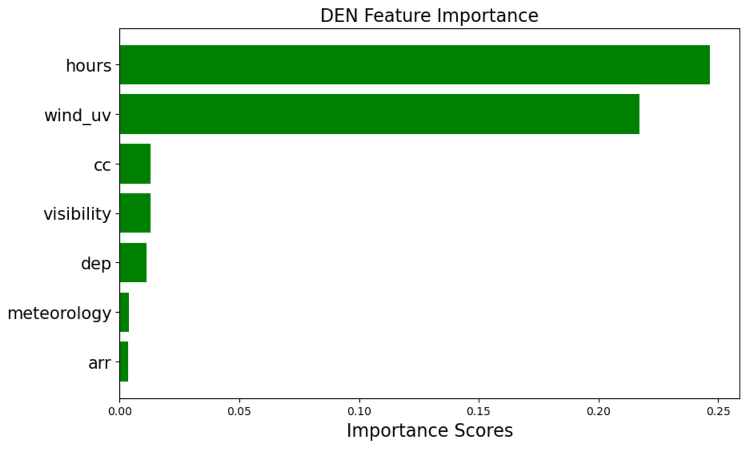
Feature permutation is a well-known method for explainability; therefore, by showing similar results between SHAP and feature permutation, it can be concluded that our SHAP results are accurate. Thus, SHAP explainability can improve upon feature permutation's explainability by analyzing at a finer granularity. In a feature permutation bar graph, the x-axis represents the change in the model performance metric, which in our case is the accuracy. Importance scores, therefore, represent the magnitude of performance degradation for accuracy when the feature is permuted. A larger value indicates a higher impact of the feature on model performance, suggesting that the model relies heavily on this feature to make accurate predictions.



(a) CLT



(b) DFW



(c) DEN

Fig. 4 Feature Importance calculation for different airports

As seen from the feature importance plots (Fig. 4), the wind UV, when shuffled, has the most significant impact on the model's accuracy. The wind UV being the most considerable feature aligns with the expected results in CLT data. Similarly, with DFW data, wind UV is the most important feature when utilizing feature permutation methods. The other features have little to no effect on the model's accuracy. Arrival and departure showed minimal impact on the model's performance in both DFW and CLT, indicating that they could potentially be removed to streamline the model without loss of predictive power.

When the model is run on the DEN airport data, the results differ from those of DFW and CLT data. Unlike DFW and CLT, the hour of the day is the most important feature that contributes the most to a decrease in accuracy when permuted through the dataset. As discussed earlier, this is likely due to the preferences of ATCos and operational constraints at DEN.

VI. Conclusion

This research demonstrates the usability of explainable AI techniques in explaining runway configuration recommendations made by the RCA tool. Using SHAP analysis and feature permutation methods, we identified the crucial features that affect model predictions, such as wind components, cloud ceiling, and visibility, which align with the operational expectations of CLT and DFW airports. In contrast, the DEN airport dataset revealed the presence of unique factors, such as the importance of the hour of the day, which influenced specific operations in terms of environmental factors. The observation that weather conditions were consistently the most crucial feature suggests that the controllers were evaluating options for the runway configurations based on the prevailing weather conditions. Considering that we used historical data from these airports, it follows that the RCA model would follow industry standards. Our most significant result is that for the DEN airport, the most important feature was the hour.

The findings confirm that SHAP and feature permutation techniques are the appropriate tools for exploring model behavior and ensuring that predictions are based on industry practices, thereby ensuring model transparency and trust among stakeholders. Further work that could be done includes developing the model to encompass additional operational constraints and exploring other XAI methods to enhance real-time explainability in critical environments. This work highlights the potential of XAI as a means of ensuring safe, efficient, and secure AI model implementations in ATM and other safety-critical domains.

References

[1] Kalyanam, K. M., Memarzadeh, M., Crissman, J., Yang, R., and Tejasen, K., "Applying Machine Learning Tools for Runway Configuration Decision Support," *11th International Conference on Research in Air Transportation(ICRAT)*, 2024.

[2] Nethi, S., Memarzadeh, M., and Kalyanam, K., "Optimization of Airport Runway Configuration with Forecast-Augmented Offline Reinforcement Learning," *AIAA SciTech Forum and Exposition*, 2024.

[3] Memarzadeh, M., and Kalyanam, K., "Runway Configuration Assistance: Offline Reinforcement Learning Method for Air Traffic Management," *Journal of Aerospace Information Systems*, Vol. 22, No. 4, 2025, pp. 275–287.

[4] Nethi, S., Memarzadeh, M., and Kalyanam, K., "Optimization of Airport Runway Configuration with Forecast-Augmented Offline Reinforcement Learning," *AIAA SciTech Forum and Exposition*, 2024.

[5] Avery, J., and Balakrishnan, H., "Predicting airport runway configuration: A discrete-choice modeling approach," *Proceedings of the 11th USA/Europe Air Traffic Management Research and Development Seminar, ATM2015*, 2015.

[6] Agrawal, M., Memarzadeh, M., Kalyanam, K. M., Mulholland, K., and Tan, R., "Predicting Air Traffic Management Initiative Using Supervised Learning," *AIAA SciTech Forum and Exposition*, 2024.

[7] Li, Y., Xia, Y., Zheng, G., Guo, X., and Li, Q., "YOLO-RWY: A Novel Runway Detection Model for Vision-Based Autonomous Landing of Fixed-Wing Unmanned Aerial Vehicles," *Drones*, Vol. 8, No. 10, 2024, p. 571.

[8] Shamsuddin, R., Tabrizi, H. B., and Gottimukkula, P. R., "Towards responsible AI: an implementable blueprint for integrating explainability and social-cognitive frameworks in AI systems," *AI Perspectives & Advances*, Vol. 7, No. 1, 2025, pp. 1–23.

[9] Hernandez, C. S., Ayo, S., and Panagiotakopoulos, D., "An explainable artificial intelligence (xAI) framework for improving trust in automated ATM tools," *2021 IEEE/AIAA 40th Digital Avionics Systems Conference (DASC)*, IEEE, 2021, pp. 1–10.

[10] Degas, A., Islam, M. R., Hurter, C., Barua, S., Rahman, H., Poudel, M., Ruscio, D., Ahmed, M. U., Begum, S., Rahman, M. A., et al., "A survey on artificial intelligence (ai) and explainable ai in air traffic management: Current trends and development with future research trajectory," *Applied Sciences*, Vol. 12, No. 3, 2022, p. 1295.

[11] Guo, W., Zhou, Y., and Wei, P., "Exploring online and offline explainability in deep reinforcement learning for aircraft separation assurance," *Frontiers in Aerospace Engineering*, Vol. 1, 2022, p. 1071793.

[12] Razzaghi, P., Tabrizian, A., Guo, W., Chen, S., Taye, A., Thompson, E., Bregeon, A., Baheri, A., and Wei, P., "A survey on reinforcement learning in aviation applications," *Engineering Applications of Artificial Intelligence*, Vol. 136, 2024, p. 108911.

[13] Zhang, B., Abu Salem, F. K., Hayes, M. J., Smith, K. H., Tadesse, T., and Wardlow, B. D., "Explainable machine learning for the prediction and assessment of complex drought impacts," *Environmental Modelling & Software*, Vol. 135, 2023, p. 104933.

[14] Samek, W., Montavon, G., Lapuschkin, S., Anders, C. J., and Müller, K.-R., "Explaining Deep Neural Networks and Beyond: A Review of Methods and Applications," *Proceedings of the IEEE*, Vol. 109, No. 3, 2021, pp. 247–278.

[15] Brieman, L., "Random Forests," *Machine Learning*, Vol. 45, No. 1, 2001, pp. 5–32.

[16] Molnar, C., "Interpretable Machine Learning: A Guide for Making Black Box Models Explainable," , 2022. URL `https://christophm.github.io/interpretable-ml-book/shap.html`.

[17] Lundberg, S. M., and Lee, S.-I., "A Unified Approach to Interpreting Model Predicitons," *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS)*, 2017, pp. 4765–4774.

[18] SHAP Contributors, "shap.Explainer — SHAP Documentation," `https://shap.readthedocs.io/en/latest/generated/shap.Explainer.html`, 2023.