Title
Sequential Prediction of Go-Around Occurrence

Journal
SSRN Electronic Journal

ISSN
1556-5068

Authors
Dai, Lu
Liu, Yulin
Hansen, Mark

Publication Date
2022

DOI
10.2139/ssrn.4019517

Sequential Prediction of Go-Around Occurrence

Lu Dai*, Yulin Liu, Mark Hansen
Department of Civil and Environmental Engineering
University of California, Berkeley
Berkeley, CA 94720, USA
* *Corresponding author at: 107D McLaughlin Hall, University of California, Berkeley, CA 94720-1720, USA.*
E-mail address: dailu@berkeley.edu (L. Dai).

Abstract

A go-around is an aborted landing event of an aircraft that is on final approach. Go-arounds are costly and detrimental to safety. Building upon our previous work in go-around detection and analysis of feature contributions, we investigate different learning models and prediction regimes for making sequential predictions of go-around probabilities based on realized trajectory data and environment factors as the aircraft proceeds on its approach. This paper develops and compares the performance of different learning algorithms and prediction strategies for the sequential go-around prediction problem. Applying these methods to a data set consisting of more than 100,000 flight approaches into JFK airport, we find that the Input Output Hidden Markov Model with multi-step prediction strategy, in general, outperforms other models due to its capability of capturing the inherent temporal structure of the entire flight sequence.

Keywords:
Go-around
Hidden Markov model
Machine learning
Sequence Classification
Multi-step prediction

I. INTRODUCTION

A go-around is an aborted landing of an aircraft due to adverse conditions such as wind shear, runway incursion, or unstabilized approach. While a go-around often reduces risk compared to proceeding with the landing, it is an emergence maneuver that also significantly degrades the efficiency of the airspace system with respect to airport throughput [1], flight on-time performance [2], air traffic controller workload [3], and noise [4]. However, if the probability of a go-around can be predicted in advance and communicated to operators (e.g., air traffic controllers and pilots) in time, mitigations may be taken to obviate the need for a go-around, and awareness that a go-around may be needed is increased. Therefore, this paper aims to provide a go-around predictive framework that can incorporate the operational conditions and environmental measures for the real-time prediction of go-arounds. This work enables a real-time system-wide safety assurance that is applicable to NASA's In-Time System-Wide Safety Assurance (ISSA) strategic trust initiative and builds a practical safety-enhancing risk detection tool for air navigation service providers (ANSPs), airports, airlines, and other ISSA stakeholders.

Most of the research related to go-arounds has focused on the contributing factors of go-around initiation. Zaal et al. [5] evaluated the effects of environmental conditions on go-around decision-making using statistical tests and decision tree analysis. They found that the wind speed, visibility, and localizer deviation strongly affect the perception of risk. Dai et al. [6, 7] developed a trajectory-based go-around detection algorithm to label go-around flights for JFK arrivals. They identified factors that affect go-around occurrence using principal component logistic regression and conducted a counterfactual analysis to quantify the contribution of different features. It is found that visibility, ceiling, and flying the glideslope path are the most salient determinants in reducing go-arounds [7]. These works suggest what situational conditions and operational environments might lead to go-around initiation, help the decision-making at the strategic level, and provide a crucial indication for predictive modeling.

There is little work on predicting go-arounds in the NAS system. John Bro [8] investigated the utility of neural networks in predicting go-arounds using 2000 hours of general aviation training flight data. In their work, the features are extracted to create a snapshot of an aircraft's parameters at 200 feet above ground level on approach. Figuet et al. [9] predicted the probability of go-arounds for each landing aircraft at a cutoff point located 10 km in front of the runway threshold with supervised learning classifiers. However, a flight approach procedure is a time-series sequence. A predictive model for go-arounds should cover the entire approach and leverage a time series of relevant data to capture the inherent temporal structure of the flight approach. Existing models fall short of this, as they are based on only one snapshot of features in the time series process.

As for sequential predictions, there are two common approaches. One is to use point features extracted from the sequence to build multiple static models at every timestamp. Martinez et al. [10] apply gradient boosting methods to predict the actual runway occupancy times at Vienna airport. For each flight sequence, the instant in which the prediction is made has been set to [10, 9.5, 9, …, 2.5, 2] nautical miles before the landing runway threshold – every 0.5 nm between the landing runway threshold and 10 nm from that. Dai and Hansen [11] apply machine learning models to predict the Runway Occupancy Buffer (ROB), which is the time difference between the runway threshold crossing time of the trailing aircraft and the runway exit time of the leading aircraft, when the trailing aircraft is at [10, 9, …, 3, 2] nm before the landing runway threshold. The other method is to use time series of relevant data for temporal models that can learn the inherent temporal structures of the entire sequence. These temporal models, such as hidden Markov models (HMMs) and Recurrent Neural Networks (RNNs), capture the temporal dynamics in a more flexible way and allow a "memory" of the previous inputs to persist in the internal hidden units, which then influences

the prediction result. Both methods display promising results in the aviation field, especially in dealing with flight trajectory data. Dai et al. [12] made the first attempt to predict go-around occurrence sequentially using both methods. Though the down-sampling technique was applied to address the class imbalance issue, the model performance is not good as it only provides the single-step prediction of the go-around occurrence prior to the next one track point.

In this work, we seek to explore different strategies to predict go-around occurrence at each timestamp of a landing aircraft sequence, given the realized trajectory data and its surrounding environment available at different times of the approach. The go-around prediction is formulated as a sequence-to-sequence (seq2seq) prediction problem, where we train models to convert multivariate sequences in the feature space to a sequence consisting of go-around probabilities at each timestamp. The main contributions of this paper are as follows: First, we develop sequential supervised learning classifiers and deep generative models to make sequential predictions of the go-around probabilities, under both single-step and multi-step prediction regimes. Second, our predictive framework establishes a baseline for predictive performance that may be improved upon in subsequent studies. Finally, to support these models, we develop a unique data set containing a rich set of features derived from realized airborne and airport surface trajectories, airport conditions, and contextual variables.

The methodological framework, including data processing, modeling, and performance evaluation, is presented in Figure 1. In the rest of the paper, we will briefly describe the data preprocessing in Section II, focusing on constructing the multivariate subsampled flight sequences, detecting/labeling go-around flights, and time-varying features derivation. In Section III, we discuss two modeling approaches – multiple supervised learning models trained at different timestamps independently, and a deep generative model trained on sequential data. We describe the performance evaluation of the single-step prediction and the multi-step prediction in Section IV. Section V shows the experimental results of the go-around prediction with a real-world dataset. Conclusions and future work are proposed in Section VI.

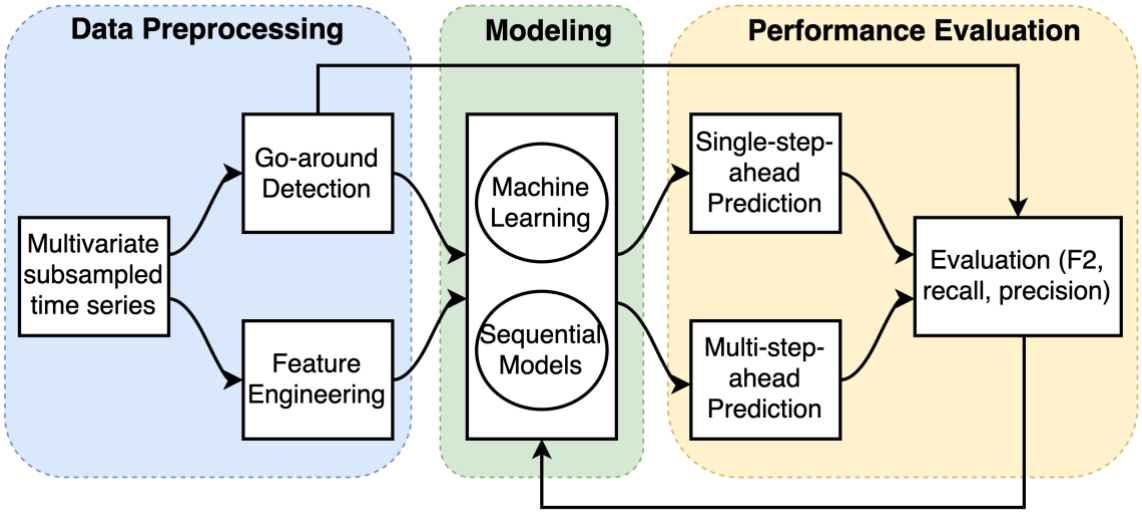

Fig. 1. Methodological framework.

II. DATA PREPROCESSING

A. Subsampled Flight Sequences

The flight trajectory data are time-series of aircraft position recorded every few seconds. In order to reduce computational complexity maintaining an informative representation of trajectories, we applied linear extrapolation to sample the flight trajectories at every nautical mile away from the landing runways, and only kept the trajectories within ten nautical miles of all the landing runway thresholds at the analyzed airport (Flights beginning apparent go-arounds outside the 10 nm range will not be considered). We hereafter define those points at every nautical mile away from the landing runway thresholds as *information cutoff gates*, and we will only derive features at those gates. The extrapolation technique ensures that the sequential prediction will be only based on the information available when the subject flight passes a certain information cutoff gate. As a result, each flight sequence can be represented by at most nine information cutoff gates – located at 10, 9, …, 2 nm to the runway threshold. Some of the flight sequences may have a length of fewer than nine timestamps if the aircraft aborted the landing and flew away in the middle of the approach procedure.

B. Identifying Go-Arounds

The raw flight trajectory data does not explicitly label go-around flights that can serve as ground truth for the prediction task. Therefore, we first apply the trajectory-based go-around detection algorithm introduced by Dai et al. [6, 7] to identify go-around from the JFK arrival flights. Each flight trajectory will be processed and must meet the criteria described in the paper [7] to be considered as a go-around. Notice that the algorithm can not only detect the go-around flights but also identify when and where the go-around occurred. For each labeled go-around flight, we treated the timestamp/trackpoint at the start of ascent as the initial time/location for the go-around procedure, and further truncated the trajectory after that point. Therefore, each flight trajectory considered to be a go-around can be represented by at most nine track points, which end before the flight altitude increases. Before a go-around is initiated, all of the go-around labels $G_{i,d}$ for flight i at distance d will be 0. Afterward, the $G_{i,d}$ values will be 1. For example, a flight that initiated a go-around at 5.3 nautical miles from the landing runway threshold has a flight sequence spanning from 10 nm to 6 nm, which means that the 6-nautical-mile to 10-nautical-mile information gates will be considered for feature engineering. Information for the 1-nautical-mile to 5-nautical-mile gates will not be considered.

C. Data Sources and Features

We limit the scope of the study to JFK airport, and our datasets come from three sources covering the period from July 1st to December 24th in 2018, except for four days with incomplete data. According to the literature, theoretical studies, and data availability, data mining was adopted to derive six categories of features at each information cutoff gate. All the features are summarized in TABLE I; details can be found in our previous publications [7, 11, 13].

The first dataset is retrieved from the Integrated Flight Format (IFF) and Reduced Data (RD) summary of the NASA Sherlock Data Warehouse. Fields of interest include flight summary (e.g., aircraft type, origin, destination), and track points (timestamp, latitude, longitude, altitude, ground speed, etc.). The RD summary and the IFF data have been further processed and merged on a daily basis for each flight arriving at JFK. This dataset has also been used to derive flight approach features, and in-trail separation features described in TABLE I.

TABLE I
FEATURE DESCRIPTION

Category	Variable Code	Description
Flight specific characteristics	Airline	1 if flight i is operated by an international airline, 0 otherwise
	Body	1 if flight i is wide-body aircraft, 0 otherwise
	WC	Dummy variable for the weight class of flight i (heavy, B757, large, small)
	LeadWC	Dummy variable for the weight class of flight i's leading aircraft (no leading, heavy, B757, large, small)
	Runway	Dummy variable for landing runway of flight i
	Daytime	1 if the observed time is between 6 am and 6 pm in local time, 0 otherwise
Go-around clustering effect features	GaGap	The minimal time interval between the approaching time of flight i and the initiation time of the latest go-around occurred in the past 300 minutes (in minutes)
	GaGnt	The number of go-arounds that occurred in the past 30 minutes
Flight approach features	Angle	Angle with the extended runway centerline (in degree
	AltDev	Absolute altitude deviation from 3-degree glideslope (in feet)
	Speed	Flight groundspeed (in knots)
	Energy	Kinetic energy height (in feet)
Flight in-trail separation features	LOS	The loss of separation between leading and the trailing flight i (nautical miles)
	SpeedDiff	Groundspeed difference between leading and the trailing flight i (knots)
	AltDiff	The altitude difference between leading and the trailing flight i (feet)
Airport and weather condition features	ArrQue	Difference between airport supplied arrival rate (AAR) and the number of intended landing aircraft (counts)
	DepQue	Difference between airport supplied departure rate (ADR) and the number of intended depart aircraft (counts)
	RwyChange	1 if the used runway configuration is changed from the previous quarter hour, otherwise 0
	Headwind	Headwind speed (knots)
	Tailwind	Tailwind speed (knots)
	Crosswind	Crosswind speed (knots)
	Visibility_k	Discretized visibility (k = 1, 2, 3, 4); intervals are [0, 1], (1, 3], (3, 5], (5, 10] (statute miles)
	Ceiling_k	Discretized ceiling (k = 1, 2, 3, 4); intervals are [0, 5], (5, 10], (10, 30], (30, 100] (100 feet)
	VMC	1 for visual meteorological condition, 0 for instrument meteorological condition

Category	Variable Code	Description
Surface operation features	$\widehat{ROB}$	The predicted time difference between the runway threshold crossing time of the trailing aircraft and the runway exit time of the leading aircraft (seconds)
	RwyCnt	The number of aircraft and vehicles appearing on the landing runway (counts)

The second dataset, Airport Surface Detection Equipment Model X (ASDE-X) data, allows us to determine the position of aircraft and ground support vehicles in the airport surface area. Each record of raw surface track data contains the timestamp, latitude, longitude, altitude, and groundspeed. This dataset will be used to derive surface operation features – runway occupancy time and the counts of objects on the runway.

The third dataset, which comes from FAA aviation system performance metrics (ASPM), provides airport level configuration and weather information every quarter hour. We use this dataset to derive the airport and weather features by matching flight-level operation with the 15-minute interval weather information according to the time at which the aircraft reaches a certain distance from the landing runway threshold.

After data cleaning and matching, there were 100,032 arrivals in the JFK airport during the analysis period. We identified and validated 371 go-arounds within the period, accounting for 0.371% of total JFK arrivals. Features described in TABLE I are derived at each information cutoff gate and are of two types – static and dynamic features. The static features, which contain aircraft type, operated airline, and landing runways, will not change during the final approach; the dynamic features, such as flight altitude and speed, vary along with the approach. In addition, continuous features are standardized to alleviate variation in magnitudes of the feature values and improve the convergence speed of the models.

III. PREDICTIVE MODELING

As the flight approaches the airport, our goal is to predict whether the flight will initiate a go-around or not, given the information we can gather from the realized datasets at a certain point in time. The underlying question we wish to answer is that, by solely observing the current flight status and its surrounding traffic/weather environment, how well do the predictive models learn to predict the probability of go-around initiation, after the flight passes a certain information cutoff gate?

A. Sequential Supervised Learning

One issue with the go-around sequential prediction task is that the training data consist of sequences of feature-response pairs, which exhibit significant sequential correlation and do not quite fit the classical supervised learning framework. Therefore, we employ the divide-and-conquer strategy and sliding window method to break the overall sequential learning problem of predicting go-around labeling sequence $\boldsymbol{G_i}$ of flight i given feature sequences $\boldsymbol{F_i}$ into subproblems of predicting individual output labels $G_{i,d}$ of flight i at the information cutoff gate $d$given subsets of information from $\boldsymbol{F_i}$. The sliding window method enables any classical supervised learning algorithms to be applied to sequential prediction problems since it retains the information from the previous timestamps by stacking the feature spaces with a sliding window. It is based on the assumption that (1) the training samples are drawn independently and identically from joint distributions $P(\boldsymbol{F}, G)$ specific to each information cutoff gate, and (2) only a fixed-sized window of features

is relevant for predicting output values at each gate. In this paper, we assume a window size of *one*, in which the lag-one features from the previous information cutoff gate will be included to predict $G_{i,d}$ given $\boldsymbol{F_{i,d}, F_{i,d-1}}$.

We train and compare four types of supervised learning algorithms: logistic regression (LR), kernelized support vector machine (SVM), random forest (RF), and extreme gradient boosting (XGBoost) following a similar modeling regime and experimental steps in [14]. TABLE II shows the concatenated input features that are mapped to an individual output value $G_{i,d}$. Specifically, for each type of supervised learning algorithm, we have trained eight standard supervised learning models to predict $G_{i,d}$ at each information cutoff gate, given different feature vectors, and then concatenating them to form the predicted go-around label sequence $\boldsymbol{G_i} = [G_{i,9}, G_{i,8}, G_{i,7}, G_{i,6}, G_{i,5}, G_{i,4}, G_{i,3}, G_{i,2}]$ for flight i. A longer-range interaction might not be necessary as the effects yielded long ago are mostly manifested in the flight status in the previous one nautical mile already. In future work, we will investigate the sliding window size by subsampling the flight sequences with a different frequency to have more cutoff gates or expanding the prediction horizons.

TABLE II
FEATURE-RESPONSE MAPPING

Feature vectors for flight i at distance d	**Predicted labels for flight i at distance d**	
$\boldsymbol{F_{i,10}}$	$G_{i,9}$	
$\boldsymbol{F_{i,9}}$		$G_{i,8}$
$\boldsymbol{F_{i,8}}$	$G_{i,7}$	
$\boldsymbol{F_{i,7}}$		$G_{i,6}$
$\boldsymbol{F_{i,6}}$	$G_{i,5}$	
$\boldsymbol{F_{i,5}}$		$G_{i,4}$
$\boldsymbol{F_{i,4}}$	$G_{i,3}$	
$\boldsymbol{F_{i,3}}$		$G_{i,2}$
$\boldsymbol{F_{i,2}}$		

Lastly, to balance bias and variance, we have fine-tuned the hyper-parameters for each model using five-fold cross-validation. TABLE III summarizes the descriptions of hyper-parameters and their tuning ranges.

B. Hidden Markov Models

Although sequential supervised learning is straightforward and elegant, it is not well-suited for the present setting because the training data actually consists of sequences of $(\boldsymbol{F}, G)$ pairs with temporal correlations that might not be well captured by simple stacking. The sequential supervised learning method is also computationally expensive because we need to train and fine-tune each type of supervised learning algorithm for each distance-variant dataset. The temporal correlations of the sequence and the conditional dependence structure between random variables are usually expressed in probabilistic graphical models, which use a graph-based representation as to the foundation for encoding the distribution over a multi-dimensional space. The Hidden Markov Model is a probabilistic graphical model of the representation of the joint probability distribution $P(\boldsymbol{F}, G)$. A graph of a standard HMM is shown in Figure 2, where nodes represent variables, and edges represent transitions and dependence relationships. Specifically, the white nodes in the figure suggest hidden states that are typically not observed, and the blue nodes represent the

visible output emitted by those hidden states. The underlying probabilistic transitions between hidden states are termed transition probabilities and denoted as edges between the white nodes. The transitions between hidden states and output states are called emission probabilities and are represented by the edges between white and blue nodes. In this paper, we propose an input-output hidden Markov model (IO-HMM), an extension to the HMM that can better capture the sequential structure inherent in our problem to model and predict the go-around occurrence for an approaching flight. The IOHMM is a conditional model that represent $P(\boldsymbol{F} \mid G)$ rather than $P(\boldsymbol{F}, G)$. It can be applied to achieve our goal of predicting any particular G values given the particular feature set $\boldsymbol{F}$, instead of trying to explain how the $\boldsymbol{F}$'s are generated.

TABLE III
TUNING HYPERPARAMETERS

Model	Hyperparameter	Search Range
Logistic Regression	Penalty term for the l_2 regularization	[10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 100]
SVM	Penalty term for the l_2 regularization	[10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 100]
Random Forest	The maximal depth of the tree	[5, 10, 15, 20]
	The minimal number of samples required to split an internal node	[2, 5, 10, 15, 20]
	The number of features to consider when looking for the best split	[sqrt, log2]
XGBoost	Learning rate	[10^{-6},10^{-5},10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 0.3, 0.8]
	Minimum loss reduction required to make a further partition on a leaf node of the tree	[1, 3, 5, 10, 15, 20]
	Maximum depth of a tree	[6, 10, 15, 20]
	The maximum number of steps we allow each leaf output to be. It might help when class is extremely imbalanced	[2, 6, 10, 15, 20]
	l_2 regularization term on weights	[10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 100]
	Subsample ratio of the training instances	[0.2, 0.4, 0.6, 0.8, 0.9, 1]
IO-HMM	The l2 regularized term in linear regression	[10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 100]
	The l2 regularized term in logistic regression	[10^{-3}, 10^{-2}, 10^{-1}, 1, 10, 100]
	The number of hidden states	[2, 3, 4, 5, 6]

The IO-HMM is a special graphical model proposed by Bengio and Frasconi [15] for learning problems involving sequentially structured data. While it is originated from HMM, it has a much more flexible architecture, which can be interpreted as a statistical model for target propagation [16] based on the Expectation-Maximization (EM) algorithms. HMMs adjust their parameters using unsupervised learning, whereas the IO-HMM uses EM in a supervised fashion. It is a conditional model that predicts the labels

given the features, rather than looking at the joint probability, and explains how the features are generated. Experiments on artificial tasks [17] have shown that IO-HMM, which uses EM recurrent learning, can deal with time dependencies more effectively than backpropagation through time and other alternative algorithms. In this paper, we implement a variant of the IO-HMM that allows us to fully exploit both input and output portions of the flight sequence data, as required by the go-around prediction task.

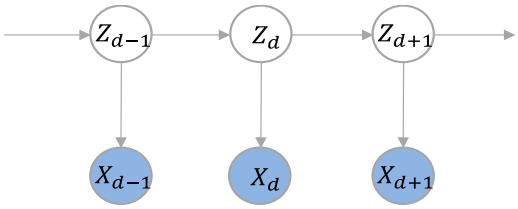

Fig. 2. HMM architecture

1) Model architecture

In order to deal with the go-around prediction problem, we regard the flight approach procedure as a discrete state dynamical process based on the following state-space description:

$$z_d = f(z_{d-1}, \boldsymbol{u_d}, \boldsymbol{x_{d-1}}) \tag{1}$$

$$[\boldsymbol{x_d}, G_d] = g(z_d, \boldsymbol{u_d}, \boldsymbol{x_{d-1}}) \tag{2}$$

Equation (1) describes the transition function between different states, where $z_d \in \{1, 2, \dots, s\}$ is a discrete hidden state variable that encodes symbols representing flight approach status (locations, speeds, etc.), s is the total number of hidden states, which is set a priori for IO-HMM. Researchers typically associate those hidden states with semantics. In our specific context, those hidden states may be representations of how stable the flight approaches are, but the semantic interpretation is not critical; rather they provide a means of capturing patterns of evolution of the output variables.

$\boldsymbol{u_d}$ is the input variable at the information cutoff gate d, including contextual information that can be known before the aircraft reaches the information gate d, such as flight-specific characteristics (e.g., operated airline, aircraft type, landing runway), weather conditions (e.g., visibility, ceiling, wind speed), and airport information (e.g., airport arrival rate, runway configuration change). The unique advantage of the IO-HMM is that it incorporates the input vector, allowing contextual variables to affect transition probabilities and emission variables. In other words, flight sequences with similar input vectors—i.e. having similar contextual variables--will have similar estimated parameters, and thus a high probability of being "clustered" in the same latent state.

Equation (2) describes the emission function relating the output variables $[\boldsymbol{x_d}, G_d]$ to the state variables z_d. To be more specific, in the equation, $[\boldsymbol{x_d}, G_d]$ the output variables at the information cutoff gate d. $\boldsymbol{x_d}$ contains dynamic features described in TABLE I, such as flight altitude and loss of separation. G_d is the go-around label obtained from the go-around detection algorithm in Section II.A. According to equation (2), the G_d is determined by the current state of the system z_d, input features at the current information cutoff gate $\boldsymbol{u_d}$ and the lag-one output features in the previous information cutoff gate $\boldsymbol{x_{d-1}}$. The output variables are available only after the aircraft passes the information cutoff gate d.

In contrast to the input variables, the output variables contain information that is not available at the transition to a new approach state. In other words, output variables can be observed when training the models but must be inferred when predicting the go-around probability. As dynamic features from the

previous information cutoff gates also contribute to part of the context information for predicting what will happen to the aircraft in the following information cutoff gate, we link the lag-one output variables $\boldsymbol{x_{d-1}}$ to the following information gate by incorporating $\boldsymbol{x_{d-1}}$ in the input vector layer.

Such discrete state dynamical system defined by equation (1) (2) can be modeled by the graph depicted in Figure 3, in which the white nodes represent hidden state variables z_d, the green nodes represent observed input variables $\boldsymbol{u_d}$, and the blue nodes contain output variables $\boldsymbol{x_d}$ and G_d.

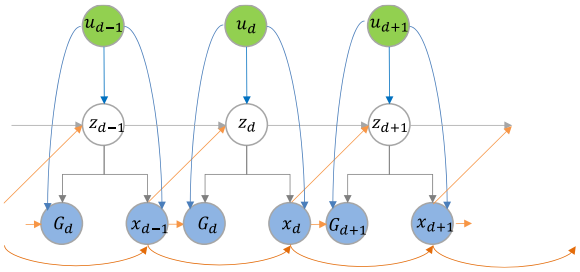

Fig. 3. IO-HMM architecture

2) Model specification

In this paper, we assume a multinomial distribution for the state variable z_d and use a Bayesian network to characterize the probabilistic dependencies among these state variables, input variables, and output variables. The IO-HMM captures the dynamics of the Markovian chain by using the current inputs and current state distribution to estimate the state variable and the output variable distributions for the next information cutoff gate. According to the connections shown in the IO-HMM graph shown in Figure 3, three probability models need to be specified: an initial probability model which outputs a vector of initial state probabilities at the start of the sequence, a transition probability model conditioned on the input sequence which outputs a square matrix of state transition probabilities at each information cutoff gate, and an emission probability (a.k.a. output probability) model governing the distribution of the output variables – including go-around probability – at a particular time given the hidden state and input features. Model details and formulas are described in the following sections.

a) Initial model

We develop a multinomial logistic regression model to estimate the initial probability parameters $\widehat{\boldsymbol{\theta}}_{\boldsymbol{initial}}$.

$$P(z_1 = i \mid \boldsymbol{u_1}; \boldsymbol{\theta_{initial}}) = \frac{\exp\left(\theta_{initial_i} \cdot \boldsymbol{u_1}\right)}{\sum_k^s \exp\left(\theta_{initial_k} \cdot \boldsymbol{u_1}\right)} \tag{3}$$

where i is the state label at the initial ($d = 1$ represents the first information cutoff gate of the sequence, which is 9nm before the landing threshold) information cutoff gate; s is the total number of hidden states.

b) Transition model

At the information gate d, the hidden state z_d is related to the input features u_d, lag-one output features x_{d-1}, and the previous hidden state z_{d-1}, subject to some Gaussian noise. The multinomial logistic regression model is used to estimate the transition probability parameters $\widehat{\boldsymbol{\theta}}_{\boldsymbol{trans}}$.

$$P(z_d = j \mid z_{d-1} = i, \boldsymbol{u_d}, \boldsymbol{x_{d-1}}; \boldsymbol{\theta_{trans}}) = \frac{\exp(\theta_{trans}{}_i^j \cdot)}{\sum_k^s \exp(\theta_{trans}{}_i^k \cdot \begin{bmatrix} \boldsymbol{u_d} \\ \boldsymbol{x_{d-1}} \end{bmatrix})} \tag{4}$$

where i is the state label at the previous information cutoff gate $d-1$, j is the state label at the current information cutoff gate d; s is the total number of hidden states. The estimated $\widehat{\boldsymbol{\theta}}_{\boldsymbol{trans}} = \begin{bmatrix} \theta_{trans}{}_1^1 & \cdots & \theta_{trans}{}_1^s \\ \vdots & \ddots & \vdots \\ \theta_{trans}{}_s^1 & \cdots & \theta_{trans}{}_s^s \end{bmatrix}$, $\widehat{\boldsymbol{\theta}}_{\boldsymbol{trans}} \in \mathbb{R}^{s \times s}$ is the transition probability matrix for the approach state transited from the previous state. The transition probabilities are heterogeneous and depend on contextual information u_d. This can improve the accuracy of state inference.

a) Emission model

To gain interpretability, we choose linear models for continuous outputs represented as Gaussian random variables ($\boldsymbol{x_d}$), thus:

$$P(\boldsymbol{x_d} \mid z_d = j, \boldsymbol{u_d}, \boldsymbol{x_{d-1}}; \boldsymbol{\theta^c_{emis}}) = \frac{1}{\sqrt{2\pi}\sigma_j} \exp\left(- \frac{\left(\boldsymbol{x_d} - \boldsymbol{\theta^c_{emis_j}} \cdot \begin{bmatrix} \boldsymbol{u_d} \\ \boldsymbol{x_{d-1}} \end{bmatrix} \right)^2}{2\sigma_j^2} \right) \tag{5}$$

where j is the state label at the current information cutoff gate d. The estimated $\widehat{\boldsymbol{\theta}}^{\boldsymbol{c}}_{\boldsymbol{emis}} = \begin{bmatrix} \widehat{\boldsymbol{\theta}}^{\boldsymbol{c}}_{\boldsymbol{emis_1}} & \cdots & \widehat{\boldsymbol{\theta}}^{\boldsymbol{c}}_{\boldsymbol{emis_s}} \end{bmatrix}$, $\widehat{\boldsymbol{\theta}}^{\boldsymbol{c}}_{\boldsymbol{emis}} \in \mathbb{R}^{m \times s}$ is the emission coefficient matrix where its column $\boldsymbol{\theta^c_{emis_j}}$ denotes the coefficients of m output variables in the linear model when the hidden state is j. σ_j represents the standard deviation of the linear model when the hidden state is j. Therefore, the number of coefficients to be estimated in the emission probability model is equal to the product of the number of hidden states s and the number of output variables m.

For binary random variable G_d, the logistic regression model is used as the output emission model. The probability is as follows:

$$P(G_d = 1 \mid z_d = j, \boldsymbol{u_d}, \boldsymbol{x_{d-1}}; \boldsymbol{\theta^G_{emis}}) = \frac{1}{1 + \exp\left(-\boldsymbol{\theta^G_{emis_j}} \cdot \begin{bmatrix} \boldsymbol{u_d} \\ \boldsymbol{x_{d-1}} \end{bmatrix}\right)} \tag{6}$$

When implementing the maximum likelihood method, a Gaussian may be fit onto a single data point and lead to a singular covariance matrix. Whenever the covariance matrix is singular, the log-likelihood function will go to infinity. Thus, the maximization of the log-likelihood function is ill-posed. Ridge regularization [18] is employed to objective functions of both linear and logistic regression. The regularized term C_{ols}, C_{logit} will be fine-tuned from the range specified in TABLE III using five-fold cross-validation.

Based on the model architecture and model specifications, the likelihood of a sequence in our IO-HMM can be written as:

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{u}) = \sum_{s} [\, P(z_1 \mid \boldsymbol{u_1}; \boldsymbol{\theta_{initial}}) \cdot \prod_{d=2}^{D} P(z_d \mid z_{d-1}, \boldsymbol{u_d}, \boldsymbol{x_{d-1}}; \boldsymbol{\theta_{trans}}) \cdot \prod_{d=1}^{D} P(\boldsymbol{x_d} \mid z_d, \boldsymbol{u_d}, \boldsymbol{x_{d-1}}; \boldsymbol{\theta^c_{emis}}) \cdot P(G_d = 1 \mid z_d, \boldsymbol{u_d}, \boldsymbol{x_{d-1}}; \boldsymbol{\theta^G_{emis}})] \quad (7)$$

The temporal dependency is captured by the transitions between the hidden approach states $\boldsymbol{z}$. The direct dependency between input features and output features can capture relationships that are not fully mediated by the hidden state learned for the current timestamp.

3) Model estimation

As illustrated in the previous section, the IO-HMM includes three groups of unknown parameters to be estimated: initial probability parameters $\boldsymbol{\theta_{initial}}$, transition probability parameters $\boldsymbol{\theta_{trans}}$, and emission probability parameters $\boldsymbol{\theta_{emis}}$. In this study, we implement the Expectation-Maximization (EM) algorithm to optimize the parameter set. Explicitly, in the E-step, we compute the expected value of the complete log-likelihood as in Equation (8), given the observed data and parameters estimated (or initialized) at the previous (or initialization) step. In the M-step, parameters are updated to maximize the expected data log-likelihood.

$$\mathcal{Q}(\boldsymbol{\theta}, \boldsymbol{\theta^r}) = \sum_{i=1}^{s} \gamma_{i,1} ln\, P(z_1 = i \mid \boldsymbol{u_1}; \boldsymbol{\theta_{initial}}) + \sum_{d=2}^{D} \sum_{i=1}^{s} \sum_{j=1}^{s} \xi_{ij,d} \ln \mathrm{P}(z_d = j \mid z_{d-1} = i, \boldsymbol{u_d}, \boldsymbol{x_{d-1}}; \boldsymbol{\theta_{trans}}) + \sum_{d=1}^{D} \sum_{i=1}^{s} \gamma_{i,d} [\ln P(\boldsymbol{x_d} \mid z_d = j, \boldsymbol{u_d}, \boldsymbol{x_{d-1}}; \boldsymbol{\theta^c_{emis}}) + \ln P(G_d = 1 \mid z_d = j, \boldsymbol{u_d}, \boldsymbol{x_{d-1}}; \boldsymbol{\theta^G_{emis}})] \quad (8)$$

where r represents the iteration; s is the total number of hidden states; D is the total number of information cutoff gates in each flight sequence; $\boldsymbol{u_d}, \boldsymbol{x_{d-1}}, \boldsymbol{x_d}$, G_d, and z_d are the input variables, lag-one output variables, output variables, go-around binary labels, and hidden state variables at information cutoff gate d, which were introduced in Section III.A.a; $\boldsymbol{\theta_{initial}}, \boldsymbol{\theta_{trans}}, \boldsymbol{\theta^c_{emis}}, \boldsymbol{\theta^G_{emis}}$ are parameters to be estimated in initial probability model, transition probability model, and emission probability model which were discussed in Section III.A.b.

$\xi_{ij,d}$ is the posterior transition probability, which defines the probability of being in state i at the information cutoff gate d and state j at the information cutoff gate $d+1$. $\gamma_{i,d}$ is the posterior state probability for state i at the information cutoff gate d. $\xi_{ij,d}$ and $\gamma_{i,d}$ are computed from forward probability α and backward probability β under the forward-backward algorithm [19], which involves three steps:

- Computing the forward probability, which provides the probability of ending up in any particular state i given the first d observations in the sequence;

- Computing the backward probability, which provides the probability of seeing the observations from information cutoff gate $d+1$ to the end given we are in a particular state at the time;
- These two sets of probabilistic distributions can then be combined to obtain the distribution over states at any specific point $d \in \{1, \cdots, D\}$ given the entire observations sequence.

$$\xi_{ij,d} = P(z_d = i \mid z_{d-1} = j, \boldsymbol{u_d}, \boldsymbol{x_{d-1}}; \boldsymbol{\theta_{trans}}) \cdot \alpha_{i,d} \cdot \beta_{j,d} \cdot P(\boldsymbol{x_d} \mid z_d = j, \boldsymbol{u_d}, \boldsymbol{x_{d-1}}; \boldsymbol{\theta^c_{emis}}) \cdot P(G_d = 1 \mid z_d = k, \boldsymbol{u_d}, \boldsymbol{x_{d-1}}; \boldsymbol{\theta^G_{emis}}) / \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{u}) \quad (9)$$

$$\gamma_{i,d} = \frac{\alpha_{i,d}\beta_{i,d}}{\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{x}, \boldsymbol{u})} \quad (10)$$

In summary, the forward-backward algorithm starts with some initial estimates of the IO-HMM parameters $\boldsymbol{\theta_0}$. We then iteratively run the Expectation-step and the Maximization-step. In the E-step, we compute the expected value of the complete data log-likelihood, the posterior state probability $\gamma_{i,d}$, and the posterior transition probability $\xi_{ij,d}$ for each training sequence, given the entire observed data sequence and parameters estimated at the previous (or initialized) step. In the M-step, we use the computed distribution over states to update the transition probability matrix and the emission likelihood matrix for maximizing the expected data likelihood.

C. Performance Evaluation

There are usually two regimes for sequential prediction – single-step prediction and multi-step prediction. Figure 4 and Figure 5 show the simplified process of single-step prediction and multi-step prediction of the IO-HMM. $\boldsymbol{F_d}$ represents the feature vectors available/known after the flight passes gate d, including both the input variables $\boldsymbol{u_d}$ and output variables $\boldsymbol{x_d}$, G_d. Sequential interactions are modeled by the state variables $\boldsymbol{z_d}$. For the sequential supervised learning models, only the single-step prediction is available based on TABLE II.

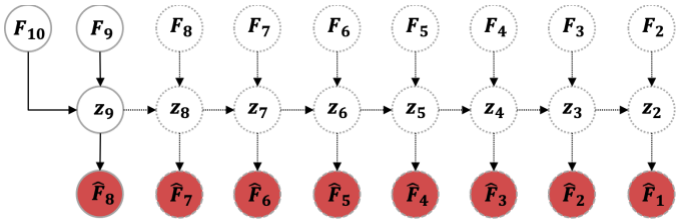

Fig. 4. Single-step prediction

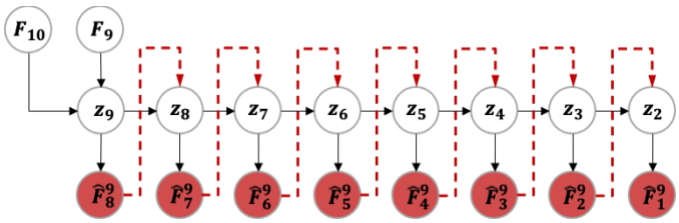

Fig. 5. Multi-step prediction

The single-step prediction only predicts one time step ahead. Feature space must be updated to include the latest information in order to predict the values in the next step. For example, without knowing the

features at 8 nm $\boldsymbol{F_8}$, we cannot produce predictions at 7 nm $\widehat{\boldsymbol{F}}_{\boldsymbol{7}}$, and we fail to predict the probability of go-arounds before the flight reaches the 7-nm gate, so on for the subsequent timestamps. Thus, the go-around probability at a certain gate d is denoted below, with $f(\cdot)$ representing the method or algorithm that performs the prediction.

$$\hat{G}^{d_{single}} = f(\boldsymbol{F_{d-1}}, \boldsymbol{F_{d-2}}) \tag{11}$$

The multi-step prediction provides the results of the rest of the steps or multiple steps ahead by recursively feeding through the prediction results from the previous timestamp until the end of the sequence. Every time the flight reaches a certain gate d, we will obtain a sequence of predictions $\widehat{\boldsymbol{F}}^{\boldsymbol{d}} = [\widehat{\boldsymbol{F}}^{\boldsymbol{d}}_{\boldsymbol{d-1}}, \widehat{\boldsymbol{F}}^{\boldsymbol{d}}_{\boldsymbol{d-2}}, \ldots, \widehat{\boldsymbol{F}}^{\boldsymbol{d}}_{\boldsymbol{1}}]$. Figure 5 shows the multi-step prediction after the flight passes the 9-nm gate, at which the feature vectors at 10 nm and 9 nm are available ($\boldsymbol{F_{10}}$, $\boldsymbol{F_9}$), and a sequence of go-around predictions $\widehat{\boldsymbol{G}}^{\boldsymbol{9}} = [\hat{G}^9_8, \hat{G}^9_7, \ldots, \hat{G}^9_1]$ can be extracted from the $\widehat{\boldsymbol{F}}^{\boldsymbol{9}} = [\widehat{\boldsymbol{F}}^{\boldsymbol{9}}_{\boldsymbol{8}}, \widehat{\boldsymbol{F}}^{\boldsymbol{8}}_{\boldsymbol{7}}, \ldots, \widehat{\boldsymbol{F}}^{\boldsymbol{9}}_{\boldsymbol{1}}]$. Note that we will obtain a new sequence of go-around predictions once the flight passes the next gate, and more information becomes available. For example, after the flight passes the 5-nm gate, the sequence of go-around predictions is $\widehat{\boldsymbol{G}}^{\boldsymbol{5}} = [\hat{G}^5_4, \hat{G}^5_3, \hat{G}^5_2, \hat{G}^5_1]$. For comparison, we adopt the complement rule of probability to convert a sequence of go-around probabilities at a certain gate d, $\widehat{\boldsymbol{G}}^{\boldsymbol{d}}$, to a single probability G^d:

$$\hat{G}^{d_{multi}} = 1 - \prod\nolimits_{r=1}^{d-1} \left(1 - \hat{G}^d_r\right) \tag{12}$$

Therefore, every time the flight passes a certain information cutoff gate d, we obtain a single-step prediction $\hat{G}^{d\_single}$, and an updated "single-ste" prediction $\hat{G}^{d\_multi}$ converted from a sequence of predicted go-around probabilities using Equation (12). The single-step prediction $\hat{G}^{d\_single}$ represents the probability of go-around prior to the next nautical mile, while the multi-step prediction $\hat{G}^{d\_multi}$ can represent the probability of go-around prior to landing. Assuming that we are able to constantly update the observed information at the previous *1 nm* in real time, these two go-around probabilities will be updated accordingly and can be transmitted to pilots or controllers as a signal or alert for go-around occurrence during the approach and landing phase.

IV. Experimental Results

A. *Experimental Steps*

In our go-around dataset, we have 371 go-arounds out of 100,032 arrival flights. For each distance-variant dataset, we first split the data into a training set (80%) and a testing set (20%). Then, we train, validate, and test the aforementioned models on these datasets. In addition, we observe that the dataset is extremely imbalanced with go-around flights significantly less than those non-go-around flights - only 0.3% of arrivals are go-around flights. With so few go-arounds relative to non-go-arounds, the learning algorithm is often unable to generalize the behavior of the go-around flights well, and tends to classify all observations as majority class; hence the algorithm performs poorly on the minority class. Therefore, we apply a special treatment to deal with the class imbalance issues – penalizing the misclassification of the minority class by an amount proportional to how under-represented it is.

The experimental steps and model selection process follow a similar regime in [14], including data processing, parameter tuning, and performance evaluation with statistical hypothesis tests. Specifically, after preprocessing the imbalanced data and splitting it into 80% of the training set and 20% of the testing set, we then use five-fold cross-validation on the training set to fine-tune the model parameters as shown in

TABLE III based on the F-score. F-score, the weighted harmonic mean of the precision and recall, is chosen to be the evaluation criteria through the experiments. Based on the accuracy measures in the confusion matrix [20], $precision = \frac{TP}{TP+FP}$ and $recall = \frac{TP}{TP+FN}$ are defined. We want to have high confidence that observations predicted as go-arounds are actually correct (high precision), as well as a high detection rate of the go-arounds (high recall). However, high precision comes at the cost of low recall and vice versa. Referring to [1] that the cost of miss detecting a go-around highly outweighs the cost of getting a false alarm warning in the system, we set the parameter, which determines the weight of recall in the F-score, as ***2***.

$$F_2 = \frac{(1+2^2)\cdot precision \cdot recall}{2^2 \cdot precision + recall} \tag{13}$$

Lastly, we fit models using selected parameters on the entire training set and make predictions on the testing set. For the conventional classification, each kind of particular supervised learning model is trained eight times separately over different distance-specific datasets, using only features up to the current nautical miles. For the sequential classification, the IO-HMM model is trained once for all the 80,025 flight sequences. In order to compare the model performance, five metrics will be reported: F2 score, the area under the receiver operating characteristic curve (AUC), precision, recall, and accuracy.

B. Model Performance

TABLE IV reports the model performance on the testing set of four supervised learning models and the single-step prediction and the multi-step prediction of the IO-HMM. The corresponding metrics scores are comparable across different models, for a specific information cutoff gate, in the same dataset.

According to TABLE IV, the IO-HMM consistently outperforms the other models except for the early stage of the flight approach. This suggests that incorporating temporal structures in the model can improve go-around prediction performance. As for the supervised learning models, the random forest model is comparable to the IO-HMM, yet it still slightly falls behind the IO-HMM, especially during the later stage of a flight approach process. The bootstrapping and oversampling techniques applied in RF help overcome the imbalanced class issue and avoid overfitting, thus the model has better predictive performance.

We also notice that the IO-HMM gives a consistent and monotonically increasing performance, in terms of F2 score, as the approach progresses, while other models do not. The main reason may be that most go-arounds happen closer to the airport. The supervised learning models are trained separately using independent input features, and are incapable of learning a whole flight sequence that inherently captures the temporal dependency. For the IO-HMM, the most confident prediction is obtained when the flight is at 2 nm before the runway threshold, with a 0.702 recall score and a 0.359 precision score, meaning that 70.2% of go-arounds are correctly classified, and 35.9% of the go-arounds predicted by IO-HMM are correct.

With multi-step predictions, the IO-HMM slightly improves over the single-step predictions for every distance-variant dataset in terms of F2 score, precision, recall, and AUC. In addition, the prediction performance improves as the flight gets closer to the airport. This is because more information in the temporal sequences is preserved, and the feature space used for multi-step predictions becomes more accurate.

TABLE IV
MODEL PERFORMANCE

After the flight passes gate d	Model	F2	Precision	Recall	AUC	Accuracy
9	Logistic	0.151	0.064	0.227	0.848	0.985
	SVM	0.143	0.062	0.213	0.829	0.985
	XGBoost	0.164	0.085	0.213	0.860	0.988
	RF	0.165	0.073	0.240	0.841	0.986
	IO-HMM (single)	0.148	0.067	0.213	0.777	0.978
	IO-HMM (multi)	0.413	0.228	0.520	0.708	0.794
8	Logistic	0.152	0.167	0.149	0.843	0.994
	SVM	0.146	0.056	0.243	0.814	0.982
	XGBoost	0.165	0.047	0.446	0.846	0.964
	RF	0.156	0.054	0.297	0.846	0.978
	IO-HMM (single)	0.149	0.062	0.230	0.711	0.984
	IO-HMM (multi)	0.452	0.266	0.547	0.726	0.820
7	Logistic	0.156	0.071	0.222	0.857	0.987
	SVM	0.185	0.066	0.333	0.828	0.981
	XGBoost	0.180	0.063	0.333	0.856	0.980
	RF	0.180	0.052	0.472	0.864	0.967
	IO-HMM (single)	0.153	0.077	0.203	0.728	0.988
	IO-HMM (multi)	0.432	0.216	0.575	0.731	0.776
6	Logistic	0.184	0.063	0.352	0.862	0.979
	SVM	0.228	0.091	0.366	0.854	0.985
	XGBoost	0.177	0.068	0.296	0.851	0.983
	RF	0.210	0.091	0.310	0.860	0.987
	IO-HMM (single)	0.230	0.092	0.368	0.808	0.985
	IO-HMM (multi)	0.494	0.313	0.577	0.756	0.852
5	Logistic	0.195	0.065	0.391	0.862	0.978

After the flight passes gate d	Model	F2	Precision	Recall	AUC	Accuracy
	SVM	0.226	0.146	0.261	0.855	0.992
	XGBoost	0.195	0.073	0.333	0.863	0.983
	RF	0.247	0.122	0.333	0.856	0.989
	IO-HMM (single)	0.255	0.100	0.417	0.828	0.986
	IO-HMM (multi)	0.505	0.275	0.638	0.773	0.857
4	Logistic	0.226	0.080	0.413	0.891	0.983
	SVM	0.270	0.153	0.333	0.884	0.992
	XGBoost	0.220	0.088	0.349	0.874	0.987
	RF	0.282	0.131	0.397	0.854	0.990
	IO-HMM (single)	0.291	0.117	0.463	0.836	0.989
	IO-HMM (multi)	0.510	0.322	0.597	0.794	0.871
3	Logistic	0.262	0.127	0.357	0.861	0.991
	SVM	0.299	0.129	0.446	0.862	0.990
	XGBoost	0.265	0.205	0.286	0.874	0.995
	RF	0.304	0.410	0.286	0.877	0.997
	IO-HMM (single)	0.307	0.141	0.435	0.839	0.990
	IO-HMM (multi)	0.529	0.274	0.690	0.820	0.845
2	Logistic	0.278	0.115	0.429	0.873	0.989
	SVM	0.270	0.102	0.457	0.878	0.989
	XGBoost	0.253	0.151	0.304	0.867	0.994
	RF	0.284	0.126	0.413	0.848	0.992
	IO-HMM (single)	0.312	0.155	0.417	0.861	0.990
	IO-HMM (multi)	0.589	0.359	0.702	0.884	0.907

V. CONCLUSION AND FUTURE WORK

In this paper, we have shown how go-around prediction can be addressed as a multivariate sequential prediction problem. We view it as a sequence classification problem and compare different learning

algorithms with two prediction strategies. A set of supervised learning models serve as the benchmarks of the go-around prediction task. The experiments were also carried out with the IO-HMM designed to evolve model estimates (learned weights) through discrete state space. It was found that the IO-HMM can capture the dependency structure in the flight sequence, and the multi-step prediction strategy achieves better prediction performance. Further research performed with more real-world datasets and additional sequential models is required to generalize and improve the results.

A major challenge of go-around prediction is the imbalanced class distribution of the data. The go-around rate is (thankfully) very low – FAA reports that the average go-around occurrence across the core 30 airports in the US from 2012 to 2018 is at a rate of 0.4% [21]—but the resulting class imbalance in go-around data sets makes predictive modelling difficult. Future work will be envisaged investigating different techniques to address the class imbalance issue. Our ongoing efforts of applying Generative Adversarial Networks (GANs) to generate synthetic go-around sequences achieve promising results compared to the model trained on the original data.

Aside from prediction, another product of the IO-HMM model is the pattern recognition capability. The hidden state variables $\boldsymbol{z}$ inferred from data are categorical labels corresponding to unobserved activity patterns, or flight approach types, or other semantic meanings that can be associated to it following an in-depth analysis. First, a set of decision rules based on the spatial-temporal representations of features (e.g., speed/altitude profile) need to be designed to identify the latent semantics of each of the hidden state variables. Second, we would need to compare the annotated activity patterns with additional data sources (e.g., ground truth or simulated approach patterns identified by domain experts via manual inspection), that are independent of the in-use dataset, to validate the pattern recognition results and the activity transition chains. Although it may prove to be difficult in light of the highly specialized nature of the aviation domain (as compared, for example, with urban activity modeling), the pattern recognition capability can help "annotate" flight activity types (in the approach phase, for this work), which may simplify scenario evaluation for aviation operators, and thus improve the safety and efficiency of air traffic control.

ACKNOWLEDGMENT

This work was sponsored by the National Aeronautics and Space Administration (NASA) Ames Research Center (grant number: NNA16BE49C). We thank John Shade from ATAC Corporation for providing the data and project advisory.

REFERENCES

[1] Shortle, J. and L. Sherry. *A Model for Investigating the Interaction Between Go-Arounds and Runway Throughput*, in *2013 Aviation Technology, Integration, and Operations Conference*. 2013, American Institute of Aeronautics and Astronautics.

[2] Blajev, T. and C.W. Curtis. *Go-Around Decision-Making and Execution Project*. 2017, Flight Safety Foundation.

[3] Jou, R.-C., C.-W. Kuo, and M.-L. Tang. *A study of job stress and turnover tendency among air traffic controllers: The mediating effects of job satisfaction.* Transportation Research Part E: Logistics and Transportation Review, 2013. **57**: p. 95-104.

[4] Prats, X., V. Puig, J. Quevedo, and F. Nejjari. *Multi-objective optimisation for aircraft departure trajectories minimising noise annoyance.* Transportation Research Part C: Emerging Technologies, 2010. **18**(6): p. 975-989.

[5] Zaal, P., A. Campbell, J.A. Schroeder, and S. Shah. *Validation of Proposed Go-Around Criteria Under Various Environmental Conditions*, in *AIAA Aviation 2019 Forum*. 2019, American Institute of Aeronautics and Astronautics.

[6] Dai, L., Y. Liu, and M. Hansen. *Modeling Go-around Occurrence*. in *Thirteenth USA/Europe Air Traffic Management Research and Development Seminar (ATM2019)*. 2019.

[7] Dai, L., Y. Liu, and M. Hansen. *Modeling go-around occurrence using principal component logistic regression.* Transportation Research Part C: Emerging Technologies, 2021. **129**: p. 103262.

[8] Bro, J. *FDM Machine Learning: An investigation into the utility of neural networks as a predictive analytic tool for go around decision making.* Journal of Applied Sciences and Arts, 2017. **1**(3): p. 3.

[9] Figuet, B., R. Monstein, M. Waltert, and S. Barry. *Predicting airplane go-arounds using machine learning and open-source data*. in *Multidisciplinary Digital Publishing Institute Proceedings*. 2020.

[10] Martinez, D., S. Belkoura, S. Cristobal, F. Herrema, and P. Wachter. *A boosted tree framework for runway occupancy and exit prediction.* Eighth SESAR Innovation Days, 2018.

[11] Dai, L. and M. Hansen. *Real-Time Prediction of Runway Occupancy Buffers*. in *2020 International Conference on Artificial Intelligence and Data Analytics for Air Transportation (AIDA-AT)*. 2020. IEEE.

[12] Dai, L., Y. Liu, and M. Hansen. *Predicting Go-around Occurrence with Input-Output Hidden Markov Model*. 2020, ICRAT.

[13] Dai, L., Y. Liu, and M. Hansen. *In Search of the Upper Limit to Air Traffic Control Communication*, in *International Conference for Research in Air Transportation*. 2018: Barcelona, Spain.

[14] Dai, L., M. Hansen, M.O. Ball, and D.J. Lovell. *Having a Bad Day? Predicting High Delay Days in the National Airspace System.*

[15] Bengio, Y. and P. Frasconi. *An Input Output HMM Architecture* 1995.

[16] Lee, D.-H., S. Zhang, A. Fischer, and Y. Bengio. *Difference target propagation*. in *Joint european conference on machine learning and knowledge discovery in databases*. 2015. Springer.

[17] Bengio, Y. and P. Frasconi. *Credit assignment through time: Alternatives to backpropagation*. in *Advances in Neural Information Processing Systems*. 1994.

[18] Thisted, R.A. *Ridge regression, minimax estimation, and empirical Bayes methods*. 1976: Department of Statistics, Stanford University.

[19] Yu, S.-Z. and H. Kobayashi. *An efficient forward-backward algorithm for an explicit-duration hidden Markov model.* IEEE signal processing letters, 2003. **10**(1): p. 11-14.

[20] Townsend, J.T. *Theoretical analysis of an alphabetic confusion matrix.* Perception & Psychophysics, 1971. **9**(1): p. 40-50.

[21] FAA. *Air traffic by the numbers*, in *FAA Report*. 2019.

Declaration of Interest

None.