A Tree-based Machine Learning Model for Go-around Detection and Prediction

Imen Dhief, Sameer Alam, Chan Chea Mean and Nimrod Lilith
Saab-NTU Joint Lab,
School of Mechnacial and Aerosapce Engineering,
Nanyang Technological University, SINGAPORE
Email : {imen.dhief | sameeralam | chan0840 | nimrod.lilith}@ntu.edu.sg

***Abstract*—The approach phase of a flight represents a safety-critical flight operation that requires close and timely cooperation between pilots and tower controllers leading to a smooth landing operation. Go-around or missed-approach procedures are in place to discontinue an unsafe landing. These procedures may generate further safety concerns due to their complex manoeuvre and time constraints. Due to the availability of high-fidelity air traffic data, such as ADS-B, new data-driven metrics can be derived in order to enhance the situational awareness of tower controllers and thus, increase the safety level of landing operations. This paper proposes a novel safety metric based on machine learning techniques that may assist tower controllers in detecting and predicting go-around events. First, a data-driven model is developed for labeling go-around events. Then, features are engineered for a tree-based learning model to predict go-around events. The model is trained, validated, and tested using ten months' of ADS-B data for flights arriving at Philadelphia International Airport (PHL), comprising 132,118 flights with 662 go-around events. Results demonstrate that the best prediction results are found at 2 NM away from the runway threshold. For the down-sampling data, the model is able to predict 56% of the go-around with only a 10% false-positive alert rate, while for the full data set the model is able to detect 33% of the go-around with a 25% false alert rate. The proposed model outperforms state-of-the-art methods in terms of decreasing the false-positive alerts in the system by 60%. The proposed model achieves a generic solution that is not specific for a single runway, and therefore can be deployed at other runways without a need for extensive model training.**

I. Introduction

The approach/landing flight phase records the highest operational risk occurrence and accounts for half of the global aviation accidents and incidents, such as runway excursions/incursions, undershooting/overshooting and unstabilized approaches. In order to mitigate these risks, go-around (or missed-approach) procedures are put in place to discontinue an unsafe landing. While these procedures are designated to prevent hazardous landings, they may generate further safety concerns due to their complex manoeuvre and time constraints. In fact, go-arounds are usually performed close to the ground, at low altitude and low speed. Thus, several actions should be taken in a short period of time including changing the altitude, thrust, flight path, and aircraft configuration while ensuring no conflict with surrounding traffic. As a result, the role of air traffic controllers (ATCs) is vital in ensuring safe and efficient go-around operations. The transition towards the digitisation of airport control towers brings more challenges in safely conducting go-around manoeuvres.

Situational awareness is the key to engage tower controllers as early as possible in order to establish effective communication between controllers and pilot, and to help controllers advise safe, efficient and optimal actions. Therefore, the current paper proposes to augment the tower control environment with predictive tools that increase the situational awareness of tower controllers and enhance the safety of airport operations.

Recent developments in the Air Traffic Management (ATM) have demonstrated the potential of Data Analytics and Machine Learning (ML) in addressing complex problems in ATM [1], [2], for which traditional techniques, based on mathematical models and hand-crafted rules, fail to deal with.

This paper proposes a data-driven and machine-learned safety metric to assist tower ATC with increased situational awareness. In particular, this paper develops a machine learning prediction model for go-around events when an aircraft is in its final approach phase.

The concept diagram of the proposed approach is illustrated in Figure 1. First, the data sets are collected from different sources, specifically surveillance, airspace and meteorological data. Then, an ML model is trained to predict whether the flight is going to perform a go-around in its approach/landing phase. When the aircraft reaches 10 NM from the runway threshold, the model starts predicting the go-around event. The prediction is subsequently updated every 2 NM in order to consider the change in the flight profile during the approach.

The paper is organized as follows. First, Section II includes a background of the considered problem, the main previous works that belong to the scope of the topic, and illustrates the current research gaps. Then, Section III highlights the proposed approach. Section IV discusses the computational results. Finally, conclusions are drawn in Section V.

II. Background

A. Go-around Problem Motivation

Go-around is a manoeuvre designated to discontinue a landing deemed to be unsafe. It consists of a set of instructions for a pilot to abandon his/her approach to landing. A pilot on an instrument flight rules (IFR) flight plan making an instrument approach should execute the published go-around procedure or proceed as instructed by the ATC.

When conducting a go-around procedure, the ATC has to instruct the pilot tactically, and this manoeuvre is considered challenging for both ATC and pilots due primarily to the time pressure. In fact, the causes leading to a go-around can be

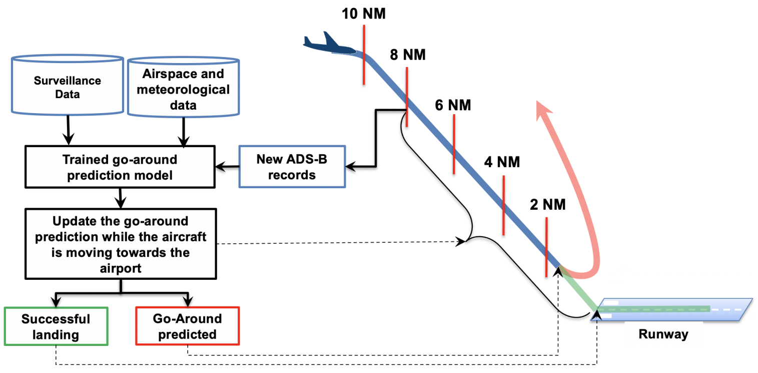
10 NM
8 NM
6 NM
4 NM
2 NM
Surveillance Data
Airspace and meteorological data
Trained go-around prediction model
New ADS-B records
Update the go-around prediction while the aircraft is moving towards the airport
Successful landing
Go-Around predicted
Runway

Figure 1. A data-driven prediction model of Go-Around event for an aircraft in its final approach phase.

unanticipated and recognized after the decision height, or even immediately prior to touchdown. Thus, the timely cooperation between the pilot and ATC is hard to establish. The study in [3] demonstrates that the startle effect included in the go-around situation reduces the pilot's ability to cope with the complexity of the manoeuvre. Their findings also show that, if the ATC gives last-minute instructions which differ from the published procedure, the pilot may be overloaded while performing crucial actions during the go-around. This may lead to critical trajectory deviations during the go-around, which has been found to be a precursor of accidents [4]. As a result, many pilots do not comply with the airline company's go-around policy, and they tend to continue the landing even when a go-around is required [5]. The flight safety foundation [5] claims that 83% of runway excursions could have been avoided with a decision to go-around, thus preventing 54% of all accidents.

When a go-around is initiated, the ATC may need to vector the aircraft at a low altitude. In this case, ATC needs to consider several flight parameters such as the aircraft's position, heading and speed, along with a clear representation of the aircraft's surrounding traffic. Vectoring a go-around may also include holding the aircraft for queuing purposes. This may increase the complexity of the manoeuvre. Thus, managing a go-around may increase the cognitive workload on controllers, especially when dealing with tightly spaced arrivals on a final approach path.

These challenges related to go-around manoeuvres provide an opportunity to research and investigate new metrics that help in improving safety of air-side airport operations. In fact, there may be a need to couple the tower control environment with alert systems that increase ATC situation awareness for more optimal, efficient, and safer operations.

B. Research Question

The objective of this research is to develop a data-driven machine learned safety metric to assist tower ATC in the prediction of go-around events. The main research question is as follows: How can a machine learning model be trained to predict go-around events, with sufficient look ahead time, for flights in their final approach phase?
The motivation of this research is to enhance the situational awareness of tower ATC, but also to implement an accurate and trustworthy prediction model for an airport with multiple runway operations. Firstly, with an effective prediction of a go-around event, the tower ATC can prepare for a re-sequencing prior to the actual initiation of the go-around. Secondly, accurate prediction (low rate of false alarms) of go-around events may reduce the workload of ATC.

C. Literature Review of Go-around Prediction and Detection

Several research works have investigated the criteria and factors that lead to go-around manoeuvres. The main factors are unstable approach [6], weather conditions [7], [8], and runway configuration change [8].

However, few works have been proposed to predict and detect go-around in the final approach phase; and thus far, effective approaches to deal with this research problem are very limited. Previous works use three main approaches, namely a data-analytic approach, statistical Markov model and decision tree-based model, which will be elaborated further in the following section. As per imbalanced data-set problems, the evaluation of different prediction models is performed via two metrics: the number of go-arounds that are correctly detected (Recall) and the number of correct go-arounds in the prediction results (Precision).

1) Data-Analytic Model

The work in [7] is among the first studies that attempts to predict a go-around manoeuvre. It investigates the operational factors that lead to a go-around event. The authors apply a statistical approach to explore features such as airborne, ground operations or weather that are most likely to affect nominal operations immediately preceding a go-around. Then the authors present an alert system for go-around prediction. A linear discriminant analysis (LDA) mechanism is applied to classify go-around from nominal trajectories in each time sample. Experiments are conducted with four years of air traffic departing or arriving at one of the three airports in the Bay Area, namely San-Francisco International Airport (SFO), Oakland International Airport (OAK) and San Jose International Airport (SJC), from January 2006 to December 2009. The data used includes flight tracks recorded by the secondary radar located at OAK and ground data collected from the Aviation System Performance Metric (ASPM) flight database. Findings show that the primary causes of go-arounds are operational errors such as runway incursion or late runway departure. However, the authors state that the prediction results were not successful. For example, when

15% of the trajectory samples were in threat alert (trajectory segments that differ from nominal trajectories), only 39% of go-arounds could be identified. The authors justify the limitation of their finding is due to the imbalanced nature of the dataset.

2) *Statistical Markov Model*

The work presented in [8] proposes a causal effect statistical study on the factors that contribute to a go-around. Also, an approach to label go-around samples from nominal samples is proposed. This approach is similar to the labelling method presented in [7], and it relies on flight altitude changes. For instance, if a flight is in the descent phase and climbs more than 400 feet, then a go-around event is captured. The same author then proposes in [9] a probabilistic graphical model based on Input-Output Hidden Markov Model (IO-HMM) to make sequential predictions of go-around probabilities for a flight approaching its destination airport. When an aircraft reaches 10NM from the airport, a short-term prediction is performed every 1NM. Experiments are conducted with six months of data for flights arriving at JFK airport, from 1^{st} July 2018 to 24^{th} December 2018. The flight records dataset is collected from the Integrated Flight Format (IFF) and Reduced Data (RD) summary of the NASA Sherlock Data Warehouse, and the ground data is collected from the airport surface detection equipment Model X (ASDE-X) data. The number of go-around events during the considered period was 371 out of 100,032 flight instances. The dataset was down-sampled to correct for biasing during the training of the model. For detection, findings show that only 41.7% of the go-arounds are correctly classified. The final evaluation of the prediction model shows that only 15.5% of the predicted go-around are correct. This is due to the fact that the implemented objective function prioritized detecting a go-around over false alarms in the system.

3) *Decision Tree-based Model*

To mitigate the data imbalance problem, the work in [10] presents a data-driven online prediction model to predict the aircraft landing speed. The authors claim it is the most important factor in evaluating the landing performance and detecting critical landing. The main objective of the paper is to aid decision-making and risk assessment during the approach and landing phases of flight, particularly during a go-around. Their proposed methodology includes two steps. First, an offline model component is used to process, analyze, and train the prediction model. Then, the second component consists of an online process in which pre-trained models are deployed to provide a real-time prediction of critical landing metrics for new flight data. A Random Forest Regression algorithm is applied for building the prediction model. The data used are retrieved from the Flight Operations Quality Assurance (FOQA), which is a process that analyses data collected from Flight Data Recorders (FDR). It includes 18,000 flight from 70 airports. Findings show that their model outperforms the state-of-the-art methods in predicting landing speed when the aircraft is at an altitude of 300 feet ($\approx$ 18 sec before touchdown). However, a go-around is a very complex manoeuvre that may be due to several factors (unstable approach, weather, visibility, or presence of obstacles in the runway) other than the aircraft landing speed. Furthermore, based on our initial experiments on the data, most of the go-arounds are initiated before reaching 300 feet, the altitude from which the prediction is performed.

In [11], the authors propose a data-driven model to predict the probability of go-around events using different classification machine learning model based on decision-tree algorithms. Their methodology includes two approaches. The first is a macroscopic model that predicts the occurrence of a go-around within the next hour, while the second predicts for each single flight the likelihood to perform a go-around when it reaches 10 Km ($\approx$5.4 NM) from the runway threshold. Experiments were conducted with 3.5 years of arrival flight to Zurich airport on runway 14. The data collected consists of ADS-B flight records provided by the OpenSky Network. It includes almost 250,000 landing flights with 715 go-arounds. The authors apply a down-sampling technique to mitigate to the data imbalance problem. The results show that their model is able to detect 50% of the go-arounds, however, only 2% of the predicted go-arounds are actual go-around events.

D. *Research Gaps*

The main gaps identified the presented state-of-the-art research works are as follows:

- Specificity of the solution: all the previously discussed works propose a specific solution for a single runway. By using features associated with a specific runway, the solution may be highly dependent to the application case, and may not provide similar results for airports with multiple runways. Particularly, in the case of DTC, the solution should be adaptable to different runway systems. In fact, DTC, by its construction, is dedicated to provide services to multiple runways.
- Prediction accuracy: a go-around prediction system is primarily dedicated to the ATC in order to increase their situation awareness and provide them with the possibility to better manage the runway usage in the case of a potential go-around. However, the go-around prediction models proposed in the literature possess a very low accuracy. For example, in [11] only 2% of the predicted go-around are actual go-around events observed in the data. A slightly better result is proposed by [9], where 15.5% of the predicted go-around are correct. This low accuracy increases the rate of false alerts in the system. As a result, having many false alerts reduces the system's trustworthiness and disrupts the ATC attention, which increases their workload.
- Data imbalance: this is a common issue when dealing with anomaly prediction such as predicting go-around occurrence which is a rare event. In a typical anomaly prediction problem, data imbalance is solved by over-sampling or under-sampling the dataset. However, these methods fail to address the more inherent issue that go-arounds occur during standard phases of operation and therefore, go-around samples will generally have features closely aligned with nominal samples. In other words, nominal and go-around samples are very poorly

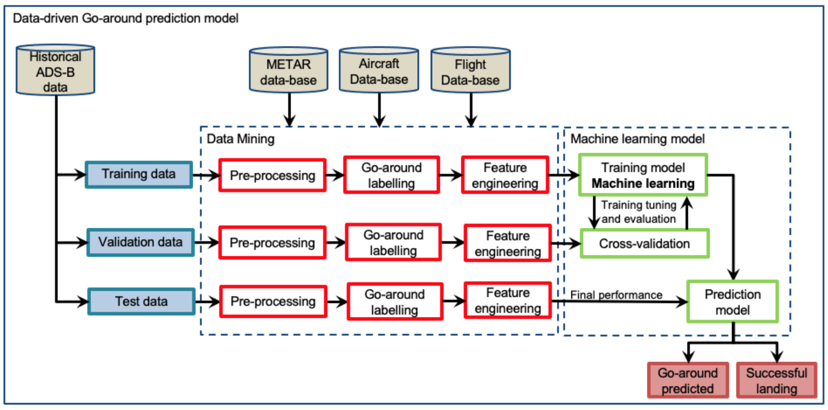

Figure 2. System diagram for the proposed go-around prediction model.

separated in the state space which may cause interference with causal relationships [7], [9].

- Go-around trajectory labelling: labelling go-around trajectories from the nominal samples is a very important step for a machine learning-based approach. Both works in [7], [8] use the altitude increase in the flight profile as the only criterion to identify a go-around. However, flights in the approach phase may increase altitude due to other factors such as to correct the approach profile or instructed by the ATC. Furthermore, if the go-around is initiated close to the missed approach altitude, the increase in altitude may not be significant, less than 400 feet. Therefore, the change in altitude should not be the only criterion to recognize a go-around, and other criteria should be investigated.

III. Proposed Approach

The current research work proposes a go-around prediction model when the aircraft is at its final approach phase. First, a go-around prediction model is build using historical surveillance, airspace and meteorological data. Then, the trained prediction model is used for the prediction of go-around when the aircraft reaches 10 NM from the runway. Subsequently, using the new records of flight data while the aircraft is moving towards the airport, the proposed algorithm updates the go-around prediction every 2 NM. The model eventually either detects an actual go-around or confirms a successful landing (figure 1).

The flow chart of the proposed approach is presented in Figure 2. The different components of the flow chart are detailed in the following sections.

A. Data set

This work considers Philadelphia International Airport (PHL) as the case study to perform our experiments. PHL is a hub airport, and is one of the airports that records the highest rate of go-arounds in the United-States [12]. The current research deploys several data sets that are collected from a range of different data sources:

- Air traffic data: the 4D flight trajectories data of this study correspond to the Automatic Dependent Surveillance-Broadcast (ADS-B) flight data extracted from the OpenSky Network [13]. For each flight trajectory, ADS-B data are recorded with unequal frequency. Thus, interpolation is performed on trajectory points in order to fix the time difference between 2 adjacent records as 1 second;
- Meteorological data: the meteorological data of Philadelphia International Airport (PHL) station are extracted from historical Meteorological Aerodrome Reports (METAR) of Iowa Environmental Mesonet from Iowa State University [14]. A single METAR reports the weather conditions around the specified aerodrome and a new METAR is generated every 30 minutes; and
- Flight and aircraft data sets: these are data collected from open-access resources that provides information such as the aircraft type, Wake Turbulence Category (WTC), and the operator for flights defined by ICAO code. They also provide flight timings (arrival time/departure time/ touchdown time, etc.) for US airlines. For flights that do not belong to US airlines, flight timings are extracted from the tracking data points.

B. Data Processing

1) Noise Filtering and Outlier Removal

The data used in the current study includes arrival and departure flights to/from PHL from March 01, 2019 to December 31, 2019. The first step of the data processing involves noise filtering and outlier removal. This consists of removing duplicated flights and flights with incomplete trajectory points. Furthermore, Visual Flight Rules (VFR) aircraft are also removed. Those flights are detected through the squawk code which is provided with ADS-B records. After this there remain $132,118$ arrival flights and $125,126$ departure flights. The inconsistency between arrival and departure counts is probably due to aircraft that are not equipped with ADS-B, thus not provided in the data.

2) Runway Assignment to Flight Trajectories

The second step of the data processing includes the assignment of flight trajectories to different runways. PHL includes four runways namely: 09R/27L, 09L/27R, 17/35, and 26/08. In order to identify the departure/arrival runway for each set of trajectory tracking data points, we use two metrics. First, the flight heading is used to identify the direction of the departure or arrival. The flight heading is the only measurement used to identify flight departing/arriving from/to runway 17/35. However, for the rest of runways, the

TABLE I. SUMMARY OF ALL CONSIDERED FEATURES. THE GREEN HIGHLIGHTED FEATURES ARE CONSTRUCTED FEATURES.

Type	Name	Description
Flight Information	Icao24	A unique 24-bit identifier of the aircraft.
	WTC	The wake turbulence category of the aircraft.
	Typecode	The aircraft type.
	Callsign	The flight call-sign.
	Operator	The aircraft operator.
	First_runway	The flight assigned runway. If the flight changes its runway after a go-around, the first assigned runway is considered.
Flight parameter	Latitude	The latitude of 3-D position of the aircraft.
	Longitude	The longitude of 3-D position of the aircraft.
	Altitude	The altitude of 3-D position of the aircraft.
	Heading	The aircraft heading.
	Ground speed	The aircraft ground speed.
	Vertical rate	The descent vertical rate.
	HOD	The hour of the day.
	DOW	The day of the week.
	DOM	The day of the month.
	Time period	Early morning, morning, noon, evening, night, or late night.
Flight landing performance metrics	Angle_runway	The angle with the runway centreline.
	Glide_slope	The glideslope.
	Energy	The aircraft energy.
Airport performance metrics	NB_DEP_FLT_BEFORE_(5MIN/10MIN/15MIN)	Number of departure flights 5min, 10min, and 15 min before the aircraft reaches the considered radius.
	NB_ARR_FLT_BEFORE_(5MIN/10MIN/15MIN)	Number of arrival flights 5min, 10min, and 15 min before the aircraft reaches the considered radius.
	NB_DEP_FLT_AFTER_(5MIN/10MIN/15MIN)	Number of departure flights 5min, 10min, and 15 min after the aircraft reaches the considered radius.
	NB_ARR_FLT_AFTER_(5MIN/10MIN/15MIN)	Number of arrival flights 5min, 10min, and 15 min after the aircraft reaches the considered radius.
	dep_(time/type/category)_before	The time difference with the preceding departure flight on the same runway, its aircraft type and category.
	dep_(time/type/category)_after	The time difference with the following departure flight on the same runway, its aircraft type and category.
	arr_(time/energy/type/category)_before	The time difference with the preceding arrival flight on the same runway, its energy, aircraft type and category.
	arr_(time/energy/type/category)_after	The time difference with the following arrival flight on the same runway, its energy, aircraft type and category.
Meteorological data	direction	The wind direction, gust, and temperature.
	gust	
	temp	
	vsby	The visibility.
	sky_cover	The sky cover (clear, few, scatter, broken, or overcast).
	sky_cover_altitude	The sky cover altitude.
	sky_cover_index	A ratio that indicates the severity of the sky cover condition.
	wind_runway	The wind direction compared to the runway heading.

flight heading results in three possible runways. Thus, the second metric used consists of framing a polygon around each runway and computing the overlap between each flight trajectory and the different polygons.

C. Go-around Labelling

As discussed in section II-D, previous works on go-around label identification apply the flight altitude as the only criterion to determine whether a flight trajectory includes a go-around manoeuvre. Based on data analysis, it is found that considering only the altitude can be misleading for the identification a go-around. In fact, flights may increase their altitude due to other factors such as to align with the glide-slope or to avoid clouds. To avoid this issue, the proposed go-around identification process includes two steps. In the first step, the flight altitude increase is captured. Then, if the altitude is increased by more than 300 feet, the second step of the algorithm checks if the flight 2D trajectory (latitude, longitude) intersects with itself. Figure 3 depicts the go-around detection algorithm. For each flight trajectory, the algorithm iterates among all the trajectory segments (which are defined by two consecutive data points as highlighted in figure 3), and for each iteration checks the intersection between the active segment, which is outlined in red in figure 3, and the preceding trajectory path, which is outlined in green in figure 3. If an intersection is found, a go-around is identified.

As this method is computationally expensive, we use the Ramer Douglas Peucker (RDP) algorithm [15] to reduce the number of trajectory points in each flight. RDP is a trajectory simplification algorithm that decimates a curve composed of line segments to a similar curve with fewer points. Data exploration shows that a go-around may occur even when the flight trajectory does not intersect with itself. This can occur when the aircraft changes its landing runway following a go-around. An example of such a trajectory is shown in

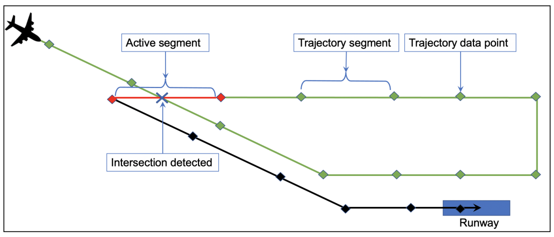

Figure 3. An example of a flight trajectory intersecting with itself in the case of a go-around flight.

figure 4. Here it is seen that the aircraft intends to land at runway 35, but it performs a go-around and lands at runway 27R without a trajectory intersection. To identify this pattern,

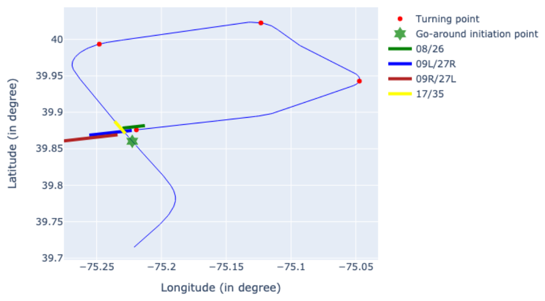

Figure 4. A trajectory of a flight that was intends to land at runway 35, but performs a go-around and lands at runway 27R

we compare the aircraft heading when the altitude increase is initiated, with the aircraft heading at the final landing. If those two headings are different, then a go-around with runway change is identified. Unfortunately, this method is not able to determine flights that change the landing runway to an adjacent runway having the same heading. Thus a supplementary process is added by checking the number of turning points after initiating the altitude increase as illustrated in figure 4. In fact, after initiating the go-around (point labeled by the green star in figure 4), the RDP algorithm proceeds to count the number of turning points of the remaining trajectory, which are the red points highlighted in figure 4. If the number of turning points exceeds two, than a go-around with runway change is detected. The number tow of turning point is chosen in order to avoid miss-classification with runway exit turn.

D. Feature engineering

Table I summarises the set of features that are included in our model. It can be divided into two groups. The first group includes the primary features that are directly extracted from the data-sets. The second group, which is highlighted in green in table I, contains features that are constructed and deduced from the primary features in order to help the prediction model in the training process. The description of the considered features is presented in table I. However, some features require further explanation, specifically the angle with the runway center-line, aircraft kinetic energy, and the sky cover index. Details on these three features is provided in the following paragraphs.

The flight angle with the runway center-line defines the angle between the aircraft heading and the runway heading at the horizontal plane, as shown in figure 5.

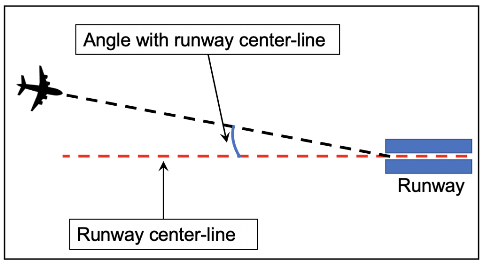

Figure 5. Top view of flight angle with the runway center-line.

The traditional method to compute the aircraft kinetic energy requires the knowledge of the aircraft mass which is not provided in our data. Therefore, we use the Specific Total Energy (STE) metric which defines the total mechanical energy per unit weight of the aircraft [16]. STE is defined by the following equation:

$$STE = h + \frac{V^2}{2g} \tag{1}$$

Where h is the altitude of the aircraft, V is the ground speed of the aircraft, and g is the gravitational acceleration.

The METAR data provides separate information about the sky cover and its altitude. Thus, the system has to learn that the combination of these two features defines the sky condition at the runway proximity. For instance, an overcast condition at $10,000$ feet should be less critical than an overcast condition at 200 feet. For this reason we include a new feature named "sky cover index (SCI)" to help the system better learn the overall sky condition. SCI is defined as follows :

$$SCI = \frac{SC}{altitude} \tag{2}$$

Where SC is a numerical value of 0, 1, 2, 3, or 4 if the sky cover is clear, few, scatter, broken, or overcast, respectively, and altitude is the sky cover altitude. Thus, when SCI increases, the probability of a successful landing decreases.

E. Prediction model

In this work, the eXtreme Gradient Boosting (XGBoost) classifier is adopted for the prediction model. XGBoost is one of the most popular algorithms in the data science community, and is able to handle large imbalanced data-sets [17]. It is based on the Gradient Tree Boosting technique, and is able to handle large-scale machine learning tasks [18]. The XGBoost algorithm has shown success in dealing with several classification and regression problems in numerous applications [19], [20]. The XGBoost algorithm uses boosting techniques to create and combine a large number of trees. The algorithm creates new models that predict previous models' residuals (errors) and learns from them to converge to the final prediction. It employs a gradient descent algorithm in order to minimize the loss when adding new models. XGBoost has recently proven its effectiveness in handling binary label-imbalanced classification tasks by implementing weighted cross-entropy on the boosting machine [17]. Weighted cross-entropy loss is a cost-sensitive method for learning imbalanced data [21]. It consists of increasing the penalization of miss-classifying the minority classes.

The model is implemented using the XGBoost python package. Tuning the model parameters has a significant impact on the overall model's performance. In order to avoid over-fitting and configure the best model parameters, we use the grid search capability from the scikit-learn python machine learning library. As our problem is a classification problem, gbtree is used as the booster type. It consists of creating a set of trees sequentially to reduce the miss-classification rate in each iteration. The grid search cross validation is implemented to tune the algorithm parameters.

IV. Results & Discussions

A. Go-around labeling results

The developed algorithm for go-around labeling is able to correctly identify 731 go-arounds, among which 93 flight change their runway following a go-around. This shows that the runway change is relatively frequent when performing a go-around, as it represents around 13% of the total number of go-arounds. Results on go-around labeling showed that most of the go-arounds are initiated within 2 NM from the runway threshold (206 go-arounds). In this paper we focus on the flight final approach phase, namely when the aircraft is aligned with the runway. Therefore, only the go-arounds that are initiated within 10 NM from the runway threshold are considered, which results in a count of 662.

B. Data preparation for the prediction model

Table II shows the number of flights and go-arounds recorded for each runway. Flights assigned to runway '09R/09L/08' or '27L/27R/26' refer to the flights that have inaccurate trajectory tracking data points, but include the aircraft heading. Thus, we only know from which direction the aircraft is approaching the runway. For instance, flights that are approaching from the west are assigned to '09R/09L/08',

TABLE II. Number of flights and go-around for each runway.

Runway	09L	09R	27L	27R	08	26	17	35	09R/09L/08	27L/27R/26	Not assigned	Total
Number of arrivals	1550	23274	9955	65834	40	708	2927	16249	1158	491	82	132,118
Number of go-around	12	124	62	281	1	20	15	139	5	0	3	662

flights that are approaching from the east are assigned to '27L/27R/26', flights that are approaching from the north are assigned to runway 17, and flights that are approaching from the south are assigned to runway 35. For some flights, even the heading is not provided. Those cases are not assigned to any runway, but are still considered in the prediction model. The runway information represents one of the features in the prediction model feature vector. Therefore, the proposed model is generic and is not dedicated to a single runway.

The number of go-around accounts for only 0.5% of the total number of arrivals. This is a highly imbalanced data set, and requires techniques to help the prediction model learn from the minority class. Down-sampling the majority class is one of the most popular techniques adopted for this type of data imbalance problem. Down-sampling consists of randomly removing instances from the majority class. As mentioned in section II-D, these down-sampling methods can fail to address the current problem as previous flight manoeuvres might contribute to a subsequent go-around event. Therefore, in this work, we propose a novel method of down-sampling. It consists of considering flights that precede or follow a go-around instance, as shown in figure 6. Let us consider a go-around flight f that reaches a radius R NM from the runway threshold. First, a Δt time interval is fixed. Then, flights that arrive within Δt time from flight f are retained in the data, and are not eligible for down-sampling removal. After this, random flight instances are also included to avoid over-fitting and to reach a percentage of go-around ranging from 5% to 15%. After experimentation, the value of Δt is set to 10 minutes.

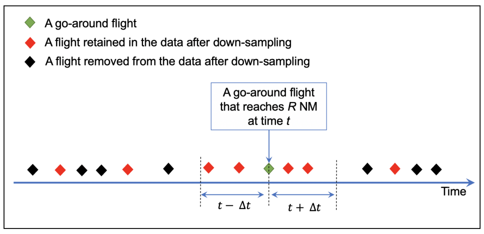
A go-around flight
A flight retained in the data after down-sampling
A flight removed from the data after down-sampling
A go-around flight that reaches R NM at time t
Time
t − Δt
t + Δt

Figure 6. An example of the proposed down-sampling technique.

Experiments on both the full and down-sampled data-sets are performed. The number of data instances in each data-set is presented in table III.

TABLE III. Data demography

Radius	Full data-set		Down-sampled data-set	
	Nb. arrivals	Nb. go-around	Nb. arrivals	Nb. go-around
10 NM	132,118	662	5663	662
8 NM	132,073	617	5516	617
6 NM	131,965	509	5344	509
4 NM	131,825	369	4825	369
2 NM	131,662	206	3427	206

The data is split into 80% training data and 20% test data for all prediction models at different radii. Furthermore, during the training three-fold cross-validation is performed to determine the optimal parameters for each prediction model. Five metrics are considered to evaluate our model: F1-score, the AUC (Area Under The Curve)-ROC (Receiver Operating Characteristics) curve, Precision, Recall, and accuracy.

C. Go-around prediction results

The prediction performance results of the down-sampling model are presented in figure 7. The algorithm performance increases as the aircraft approaches the landing. Thus, the best results are recorded at 2 NM, where 56% of the total number of go-arounds are detected with only a 10% rate of false positive alarm. Even though only half of the go-around are detected, 90% of the system go-around alarms are actual go-arounds. In comparison with other work, [9] proposes a down-sampled model that detects 74% of the go-around where only 32% of the predicted go-around are actual go-around. This means in [9] the controller will expect a false go-around in almost 70% of the system go-around alerts. Similarly, in [11], half of the go-around are detected with only 2% of the predicted go-arounds being correct. We believe this will increase the controller workload, as well as potentially perturb his/her attention unnecessarily. Therefore, having higher accuracy in terms of correct alerts results in higher trust from the controllers to use the system.

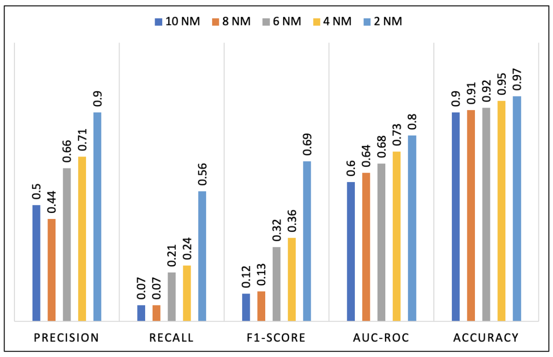

Figure 7. Histogram of the down-sampling performance results for different radii. Results show a significant improvement of the model performance while approaching the runway threshold.

The prediction model performance in full data-set is presented in figure 8. Results show that the system performance with the full data-set decreases slightly compared to the model with down-sampling data. However, the system characteristics remain similar. The performance of the model increases as the distance to the runway decreases. Also, the system places higher priority to predicting a correct go-around (high precision) than to detecting a go-around (high recall). We perform a comparison with the hidden Markov method proposed in [9] for the full data-set model at 2 NM. Our model detects 33% of go-around where 75% of the detected go-around are actually correct. However, in [9] 41% of the go-around are detected

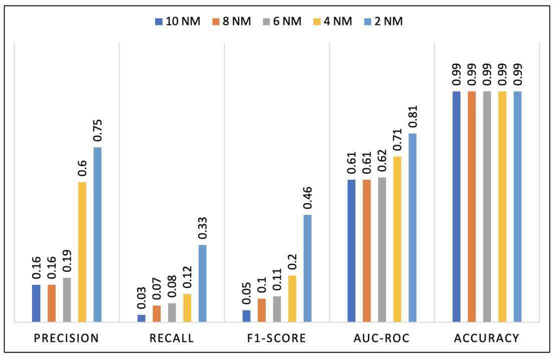

Figure 8. Histogram of the full data performance results in different radii. The results shows that the model performance increases while approaching the runway.

but only 15.5% of the detected go-arounds are actual go-arounds. Thus, we achieve a reduction in the false positive alerts by almost 60% while the detection rate decreases by approximately 8%. This means that an ATC can rely on the go-around prediction as being accurate the vast majority of the time, increasing trust in the system.

To sum up, the achieved results are more accurate when predicting a go-around compared to state-of-the-art models [7], [9], [11]. Furthermore, our findings confirm that predicting go-around occurrence is a very challenging problem. In fact, the number of detected go-around before 4 NM is very low (results in 7 and 8). The failure in accurately predicting a go-around can be justified through three main factors. First, the most important factor is that go-around events are subjective to the pilot decisions. Thus, the decision to go-around may be due to several factors that are hard to or even impossible be measured and/or collected, such as the pilot experience, personality, or his/her level of fatigue, etc. The second factor is the visibility. Even though this data is provided in METAR, it is not accurate and it does not provide the actual visibility level at the considered decision height. Finally, as we are only considering ADS-B data, our model lacks the airport surface movement data which we believe can affect a go-around decision. All these factors together mean that go-around and non go-around instances are closely aligned and similar.

V. Conclusion

The current work proposes a method to predict flight go-around events at the final approach phase for the purpose of increasing the situational awareness of tower controllers to safely perform their control tasks. It includes an innovative go-around trajectory labeling technique to detect go-around flights from historical data. Furthermore, it presents a data-driven model based on a binary classification prediction methods in order to predict flight go-around occurrence at different radii away from the runway threshold. In order to evaluate the performance of proposed method, computational experiments are conducted using ten months' of air traffic data for flights arriving at Philadelphia International Airport (PHL). Two types of experiments are conducted; the first includes down-sampling techniques and the second includes the full data set. The best prediction results are found at 2 NM away from the runway threshold. For the down-sampling data, the model is able to predict 56% of the go-arounds with only a 10% false positive alert rate, while for the full data set the model is able to detect 33% of the go-arounds with a 25% false alert rate. Our model outperforms state-of-the-art methods in terms of decreasing the false positive alerts in the system by 60%.

In future work we plan to address the go-around prediction problem by predicting the safety level of an approach profile. This will reduce the impact of the pilot subjectivity on the prediction, as well as alleviate the data imbalance problem.

Acknowledgement

This work was conducted under the Saab-NTU Joint Lab under its Machine Learning For Airport Management And Tower Control project with support from Saab AB (publ).

References

[1] Z. J. Lim *et al.*, "Causal effects of landing parameters on runway occupancy time using causal machine learning models," in *2020 SSCI*. IEEE.

[2] I. Dhief *et al.*, "Speed control strategies for e-aman using holding detection-delay prediction model," 2021.

[3] F. Dehais *et al.*, "Pilot flying and pilot monitoring's aircraft state awareness during go-around execution in aviation: A behavioral and eye tracking study," *The International Journal of Aerospace Psychology*, 2017.

[4] G. Adam and J. Condette, "Study on aeroplane state awareness during go-around," *Bureau d'Enquetes et d'Analyses pour la securite de l'aviation civile: Bourget, France*, 2013.

[5] T. Blajev and W. Curtis, "Go-around decision-making and execution project: Final report to flight safety foundation," *Flight Safety Foundation, March*, 2017.

[6] N. P. Singh *et al.*, "Real-time unstable approach detection using sparse variational gaussian process," in *2020 AIDA-AT*. IEEE.

[7] M. Gariel *et al.*, "On the statistics and predictability of go-arounds," *Computing Research Repository - CORR*, 2011.

[8] L. Dai *et al.*, "Modeling go-around occurrence," *13th USA/Europe ATM R& D Seminar, Vienna, Austria*, 2019.

[9] ——, "Predicting go-around occurrence with input-output hidden markov model," *ICRAT*, 2020.

[10] T. G. Puranik *et al.*, "Towards online prediction of safety-critical landing metrics in aviation using supervised machine learning," *Transportation Research Part C: Emerging Technologies*, 2020.

[11] B. Figuet *et al.*, "Predicting airplane go-arounds using machine learning and open-source data," *Proceedings*, 2020.

[12] FAA, "Air traffic by numbers," FAA report, Tech. Rep., 2019.

[13] M. Schfer *et al.*, "Bringing up opensky: A large-scale ads-b sensor network for research," in *Proceedings of ACM/IEEE IPSN*, 2014.

[14] I. E. Mesonet, "Asos-awos-metar data download."

[15] J. Hershberger and J. Snoeyink, "An o (n log n) implementation of the douglas-peucker algorithm for line simplification," in *Proceedings of the tenth annual symposium on Computational geometry*, 1994.

[16] T. Puranik *et al.*, "Energy-based metrics for safety analysis of general aviation operations," *Journal of Aircraft*, 2017.

[17] C. Wang *et al.*, "Imbalance-xgboost: leveraging weighted and focal losses for binary label-imbalanced classification with xgboost," *Pattern Recognition Letters*, 2020.

[18] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in 22^{nd} *International Conference on Knowledge Discovery and Data Mining*, 2016.

[19] D. Pham *et al.*, "An air traffic controller action extraction-prediction model using machine learning approach," *Complexity*, 2020.

[20] Y. Chang *et al.*, "Application of extreme gradient boosting trees in the construction of credit risk assessment models for financial institutions," *Applied Soft Computing*, 2018.

[21] Y. Sun *et al.*, "Classification of imbalanced data: A review," *International journal of pattern recognition and artificial intelligence*, 2009.