Deep-learning based Time Series Forecasting of Go-around Incidents in the National Airspace System

Conference Paper · January 2018

DOI: 10.2514/6.2018-0424

2 authors, including:

Arjun H Rao
The Ohio State University
12 PUBLICATIONS **26** CITATIONS

SEE PROFILE

Deep-learning based Time Series Forecasting of Go-around Incidents in the National Airspace System

Shreyas Vathul Subramanian*,
Robust Analytics, Inc., Crofton, MD 21112
Arjun H. Rao†
The Ohio State University, Columbus, Ohio, 43210

This research uses Long Short-Term Memory (LSTM) networks in the forecasting of incident data derived from the National Aeronautics and Space Administration's (NASA's) Aviation Safety Reporting System (ASRS) incident database. Specifically, we analyze Go-around (GAR) and Missed-approach (MA) data from the NASA ASRS database. First, we query the ASRS database to obtain all reported incidents pertinent to GARs or MAs between 1993 and 2017. Second, we identify 20 causal mechanisms, and classify the incidents based on their causes, where each incident could have more than one cause. Using this data, we train and validate a LSTM network, and generated a forecast of the number of incidents. The forecast generated help identify factors that contribute significantly to the the trends seen in multiple categories of incidents, and also provide insight into which categories of incidents are more (or less) likely to occur in the forecast period.

Nomenclature

N	Sample size
d	Input dimensionality
v	Variable
t	Time step
t_m	T minus, or look back time

Subscript

i	Number of hidden layers
j	Output dimensionality
k	Hidden layer dimensionality
l	Number of LSTM units
m	Number of LSTM layers

I. Introduction

To safely increase National Airspace System (NAS) operations through NextGen, there is a need to modernize Air Transportation Management (ATM) system. NextGen technologies such as ADS-B, Data Comm, En Route Automation Modernization (ERAM) and System Wide Information Management (SWIM) aim to increase situational awareness, and thereby increase safety and efficiency of the NAS. While efficiency improvements are tracked using metrics like throughput and number of operations, safety is treated as a constraint—proposed NAS-wide operational improvements are accepted only if the level of safety in the NAS remains the same or is improved. As such, lessons learned from accident and incident analysis have played an important role in improving aviation safety and analyzing trends in operations.

*Director of Research, Robust Analytics, Inc., AIAA Member
†Research Specialist, Center for Aviation Studies, The Ohio State University, AIAA Member

Several safety researchers have analyzed historical accident and incident data to understand the causes for accidents involving commercial[1–5] (14 CFR Part 121) and General Aviation (GA) (14 CFR Part 91) aircraft [6–10].Researchers have analyzed data from historical accident and incident databases (e.g., NTSB accident database) as well as data that can be downloaded form on-board flight data recorders (FDRs) [6–8, 11–14]. Retrospective analyses have focused on a variety of areas including the role of maintenance, high-risk accident sequences, energy management, and human factors.

Retrospective analyses help identify the most frequent causes for fatal and non-fatal accidents (and incidents), providing recommendations for improving safety and operations in the NAS. These retrospective approaches, while providing valuable insights into the mechanisms of accident/incident causation, provide little information to predict the trends in future accidents.

Researchers have used several predictive techniques to forecast occurrence of accidents in various industries. Zheng and Liu [15] identified seven major accident forecasting techniques—time-series based forecasting techniques (time-series method, Markov chain method, Grey model, and neural networks) and causality forecasting methods (scenario analysis, regression method and Bayesian networks). Further, they compared the different prediction techniques on a 14-year sample of chemical industry accident data (1989–2002).They highlighted the challenges associated with using a single technique to forecast accidents. They suggest combining existing techniques with non-linear approaches to obtain better results. Luxhøj [16] developed an integrated predictive model that combined a risk analysis framework with human performance in the aviation maintenance domain. His approach used a combination of Bayesian Belief Networks with Reason's Swiss Cheese Model to understand the interplay between technical content and human performance in the context of aviation maintenance. Most methods presented in literature focus on forecasting gross cumulative quantities (such as sum of accidents of incidents in a year); in this paper we wish to predict changes in the frequency of occurrence of various categories of incidents that contribute to the overall sum.

As a first step to using deep learning techniques to potentially improve safety in the NAS, this research analyzes Go-around (GAR) and Missed-approach (MA) data from the National Aeronautics and Space Administration's (NASA's) Aviation Safety Reporting System (ASRS) database. First, we query the ASRS database to obtain all reported incidents pertinent to GARs or MAs between 1989 and 2017. Second, we identified 20 causal mechanisms, and classified the incidents based on their causes, where each incident could have more than one cause. Next, using deep learning, specifically, Long Short-Term Memory (LSTM) algorithm, we performed a short forecast of the number of incidents and accidents, and identified features relevant to NAS operations that affect this forecast. The procedure presented herein may be used to further inform stakeholders about mitigatory steps that can be taken to maintain or improve NAS safety.

II. NASA ASRS Data for Training

This section presents our approach to the analysis of accident and incident data.We begin by providing a brief overview of the ASRS program, followed by a description of the query used to identify *Go-around/Missed approach* incidents.

In 1976, the FAA and NASA established a confidential, self-reporting system, which could be used by any person in the NAS. Some of the key objectives of the program are: (1) to record self-reported, confidential incident reports from individuals in the NAS; (2) to maintain a computer-based database of all reported incidents; (3) to implement an interactive analytical system for storage and retrieval; and, (4) to design and implement a responsive system for communication of data and analyses.The system enables anonymous (and voluntary) reporting of aviation incidents from pilots, air traffic controllers, flight attendants, maintenance staff, and eye witnesses. To identify *Go-arounds/missed approaches*, we used the following query:

Event Assessment → Results → Flight Crew→ Executed Go-around/Missed approach

Using this query, we identified 3835 incidents relating to unstabilized approaches in GA between 1993 and 2017.The downloaded dataset contained information that included geographical location of the incident, airport identifiers, types of aircraft, flight plan, operating certificate, flight phase, maintenance history of the aircraft, type of event, primary cause, contributing factors, and narratives provided by reporters.

While each incident was associated with an anomaly (e.g.,"ATC issue"), it was not easy to discern the cause for the GAR. For instance in some cases, the real reason for the GAR was mentioned in the narrative, but not in the anomaly column or synopsis. Consider for example a commercial carrier arriving at Miami

International Airport (ASRS ID: 285338). The synopsis stated the following:

> *ALT BUST DURING GAR*

The anomaly column indicated that there was some for mechanical anomaly, altitude deviation, and procedure deviation. The contents of the anomaly column were:

> *Aircraft Equipment Problem Critical; Deviation - Altitude Overshoot; Deviation - Procedural Published Material / Policy; Deviation - Procedural Clearance*

A quick review of the anomaly column and synopsis does not provide any insight into the real reason for the GAR. However, a closer look at the narrative indicates that the aircraft experienced some for landing gear malfunction, which led the crew to perform the a fly-by (a GAR for the tower to provide visual of the gear). The more detailed narrative said:

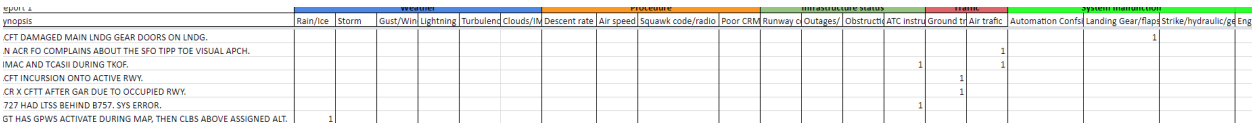
> *ON FINAL TO MIA RWY 12 ILS, GEAR DOWN GOT RED NOSE GEAR UNSAFE AND NO GREEN LIGHTS FOR GEAR. DID FLY-BY AND TWR INFORMED US GEAR APPEARED DOWN AND LOCKED. TOLD TO CLB TO 3000 FT, SWITCH TO DEP AND TURN R 180 DEGS. DURING ALL OF THIS TRYING TO FIGURE OUT WHETHER TO TRY TO RAISE GEAR, SETTING SPD BUG, TRYING TO GET HDG SELECT, FO TALKING TO DEPT, HAND FLYING AND LOOKING AT EFIS DISPLAY (WE GET TO SEE THIS ONCE EVERY WK OR 2), I CLBED TO 3600 FT OR 3700 FT BEFORE CATCHING IT AND DSNDING BACK TO 3000 FT. ATC DIDN'T SAY ANYTHING AND WE KNOW OF NO TFC CONFLICT*

After reading the narratives for each incident we identified five top-level cause categories, and 20 sub-categories as shown in Table 1. For example, in the aforementioned example, the cause for the GAR would be placed under the "*Landing gear/flaps/spolier*" category. While the number of categories and sub-categories are certainly not exhaustive, we deemed the resolution practical for the purposes of this research.

Table 1. Categories and Sub-Categories for the Classification of Incident Reports

Cause Categories	**Sub-Categories**
Weather	Rain/Ice
	Storm
	Gust/Windshear
	Turbulence
	Clouds/IMC
Procedure	Descent rate
	Airspeed
	Squawk code/Radio
	Poor CRM
Infrastructure Status	Runway condition
	Outages/infrastructure issues
	Obstruction/Terrain
	ATC instruction
Traffic	Ground traffic
	Air traffic
System Malfunction	Automation confusion
	Landing gear/flaps/spoiler
	Strike/hydraulics/generic cause
	Engine/propulsion

After reading each of the reports, we placed the causes for each incident under at least one of the five "cause categories" shown in the first column of Table 1. Next, we placed the causes under the sub-categories shown in the second column of Table 1. Figure 1 provides a snapshot of the binary recoding system that we used in this paper.



	Weather						Procedure				Infrastructure Status			Flight		System Malfunction		
ynopsis	Rain/Ice	Storm	Gust/Win	Lightning	Turbulend	Clouds/IM	Descent rate	Air speed	Squawk code/radio	Poor CRM	Runway c	Outages/	Obstructi	ATC instru	Ground tr	Air traffic	Automation Confs	Landing Gear/flaps	Strike/hydraulic/ge	Eng
/CFT DAMAGED MAIN LNDG GEAR DOORS ON LNDG.																		1		
/N ACR FO COMPLAINS ABOUT THE SFO TIPP TOE VISUAL APCH.																1				
IMAC AND TCASII DURING TKOF.														1		1				
/CFT INCURSION ONTO ACTIVE RWY.															1					
/CR X CFTT AFTER GAR DUE TO OCCUPIED RWY.															1					
727 HAD LTSS BEHIND B757. SYS ERROR.														1						
GT HAS GPWS ACTIVATE DURING MAP, THEN CLBS ABOVE ASSIGNED ALT.	1																			

Figure 1. The causes for the incidents were recorded by placing a "1" in the cells corresponding to the pertinent cause sub-category.

III. Learning and Prediction using LSTM

Recent research suggests that Deep Neural Networks (DNNs) have achieved tremendous success in the prediction accuracy of problems involving spatio-temporal datasets. Convolutional Neural Networks (CNNs), which are a flavor of DNNs exploit local dependencies and have demonstrated record setting results on various applications [17]. However, these methods are not applicable for data that are arranged as a sequence, or where input data are of varying lengths. Markov approaches (like Hidden Markov Models or HMMs) tackle the temporal aspect of the input structure by modeling a sequence of observed states as dependent upon a sequence of hidden states. Here, states can only depend on the immediately previous state. HMMs are also limited by the size of the constructed transition and emission probability matrices, which grow exponentially with the number of states. Tradional Recurrent Neural Networks (RNNs) also allow states to only be dependent on the current and previous time step, however, the hidden state at any time step can contain an arbitrarily long context window (as will be demonstrated in our application problem here).

One state of the art RNN variant is the Long Short-Term Memory (LSTM) network (others being Bidirectional RNN and Gated Recurrent Units of GRUs [18]). Of course, CNNs, DNNs and RNNs are limited in their modeling capabilities, and there have been attempts to combine or "stack" one or more models to obtain superior predictions. [19] In this work however, we will use LSTM, which unlike the multi-layer perception (typically used for predicting time series data), does not have neurons – instead, it uses memory blocks that are connected through layers. A block contains additional components such as gates that manage the the block's state and output. Each gate within a block uses sigmoid activation units to allow or disallow flow of information. LSTMs were introduced by Hochreiter's work to solve two problems: 1) recurrent back propagation takes a very long time for inputs that were characterized by long time intervals, and 2) vanishing gradient problem, when the problem had long input sequences. LSTM is an end-to-end differential cell with an analytical gradient, a computational complexity of $O(1)$. Several variants of LSTM have been explored for various applications, but we use a *vanilla* LSTM setup as described in Greff *et al.* [20], i.e., we use LSTM cells with input, output and forget gates, and also use input and output activation functions. Readers are encouraged to review sources cited herein for a detailed introduction to RNNs, LSTMs and variants [17–22]. What we do intend to clarify here are our reasons to choose LSTM for the problem at hand - forecasting trends of accident and incident *classes* and *counts*:

1. The ASRS data used in our application is sparse, and includes long time sequences - a few rows of data may traverse through months or years of time. The LSTM gating mechanism (similar to GRUs) allows us to explicitly model long-term dependencies, if any exist in the data. By learning the parameters that define LSTM gates, the network can express which states it must "store" in memory, and how [23].

2. Factors such as weather, that have a strong influence on the variety and number of incidents reported, are seasonal in nature. LSTMs excel at modeling seasonal trends using local temporal dependence, unlike GRUs, which are computationally easier, but exposes the entire memory rather than having dedicated memory units (accident and incident trends relevant to the year 2017 should not depend entirely on incidents that occurred in the 1980s)

3. LSTMs can capture multiple time scales. In our application, while local trends and repetitive factors may be related to one another, they are also connected to the long term trend of increasing or decreasing number of reports, overall with respect to time.

4. LSTM networks can be tuned to learn complicated, multivariate time series datasets, including chaotic or noisy signals [24]. While there may be a trend to be discovered in our dataset, the occurrence of incidents and accidents, and the classification of these into various categories introduce subjectivity and randomness into the process.

5. LSTMs lend themselves easily to recursive prediction (which is useful for multi-step forecasting as in this case) of highly non-linear, multivariate outputs (in our case, with a maximum of 34 dimensions of variables), especially where accumulation of errors may be an important problem.

A. Transforming and Preparing the Raw Data

We used *Python 3.0* to perform data transformation, learning and prediction steps outlined in the section below. The code and data used will be provided to interested readers upon request. We consider 2338 reported incidents of the total available 3838 cases related to "go around". We include various classifications of incidents, manually coded by us, as well as some factors included in the ASRS database. As indicated by the discussion on data transformation steps below, categorical variables (like 'IMC or VMC?') were converted into individual columns corresponding to the number of categories. We call the the 2338 rows "samples", with a sample size $N = 2338$, and the the number of columns or "factors", $d = 34$ (after data transformation). We call this $N \times d$ matrix containing raw data the input matrix. The columns or factors we considered were 'date', 'time of day' (three eight hour intervals), 'IMC or VMC conditions', 'day or night', 'Rain/Ice Storm', 'Gust/Wind Shear', 'Lightning', 'Turbulence/Wake turbulence', 'Cloud cover', 'Descent rate', 'Air speed', 'Squawk code/radio', 'Poor CRM/Pilot', 'Dispatch error', 'Runway conditions', 'Ceiling/ Visibility', 'Outages/ Infrastructure issues', 'Obstruction/Terrain', 'ATC instruction', 'Ground traffic', 'Air traffic/TCAS', 'Automation Confusion/Autopilot malfunction', 'Landing Gear/flaps/slats/spoiler', 'Strike/hydraulic' and 'Engine/propulsion related'. Steps involved in transforming and preparing the raw data are described below:

1. We recognize that due to data entry and corruption, a few cells in the $N \times d$ input matrix may not be numbers ("NaN"), and thus we find and replace all "NaN" cells in the input matrix with zeros.

2. The 'Ceiling' column, that records ceiling values in feet from ground elevation contained words such as "Ceiling" and "CLR" (or "clear") which we interpreted as being an incident during clear conditions, or a ceiling of 35000 feet (the maximum reported ceiling in these columns was 28000 feet, which may not have been a useful factor in that particular case).

3. We then sorted the input matrix using the 'date' column. sorting by the date column changed the indices of the input matrix, and so, the matrix was re-indexed.

4. Categorical variables such as 'time of day' (three categories including 1200-0800, 0900-1600, 1700-2400), 'IMC or VMC' (two categories, Instrument or Visual Meteorological Conditions), and 'Day or Night' (two categories) were converted into separate indicator columns for each category using a process called one-hot encoding.

5. Converted the 'date' column into a column with zeros when the date remained the same across multiple rows, and the difference of time in number of days when the date changed from one row to the next. This is important for predicting trends of when certain incidents are reported, as well as whether certain types of incidents are reported more (or less) frequently as time progresses.

6. We used a min-max scaler to normalize all variables to have values between 0 and 1. Inverse scaling using the same scaler obtained in this step before presenting the final forecast columns so that values of quantities in each column are representative of the original units of the factors.

7. For binary factors, for example 'turbulence/wake turbulence', values were encoded as "float" numbers, since output predictions correspond to "probability of an event being classified as turbulence/wake turbulence", i.e., a value ranging from 0 to 1, rather than "classification as turbulence/wake turbulence or not", i.e,. a binary 1 or 0 label.

8. An important parameter in the preparation of data is the look-back window, t_m. This parameter defines the local temporal dependency of input data, and answers the question, how many timesteps (in this case, rows that can span days or months) of data directly affects the next timestep. The $N \times d$

input matrix is reshaped to a $N \times (d \cdot t_m)$ matrix. The value of t_m used in our study is 15, which is close to the median number of observations in the original raw data for each unique date (multiple incidents were reported on each date, each as a separate row of information).

9. We then calculated the probability thresholds for each binary factor under question using a frequentist approach, by counting the number of times accidents were classified using that particular factor, and dividing the count by total number of incidents considered, N.

10. We finally divided the dataset into a training set (first 70% of N rows) and testing set (last 30% of N rows).[a]

B. LSTM Hyper-parameter tuning

Given that literature recommending the right choice of LSTM hyper-parameters to use to solve for the LSTM network weights used in applications similar to *our* application did not exist at the time of writing this paper, we performed a preliminary search across the optimization space relevant to weight optimization, where the actual "training" of the network happens. We considered a standard LSTM structure used in the forecasting of multivariate time series problems, that typically includes one or more LSTM layers, one or more dense and dropout layers (along with a globally set dropout rate), and a choice of a scoring function to be used in conjunction with an optimization algorithm. Figure 2 shows some of the hyper-parameters we varied in search for a suitable network including LSTM layers in blue, hidden dense layers in orange and the output layer in green - i, the number of hidden layers, j, the output dimension, k the hidden layer dimension (not controlled in this work), l the number of LSTM units, and m the number of LSTM layers. Nodes with dashed outlines (x_0 and h_1) are dropped out using the drop out rate parameter. Readers interested in a detailed introduction may refer to the references cited herein. [17–22]

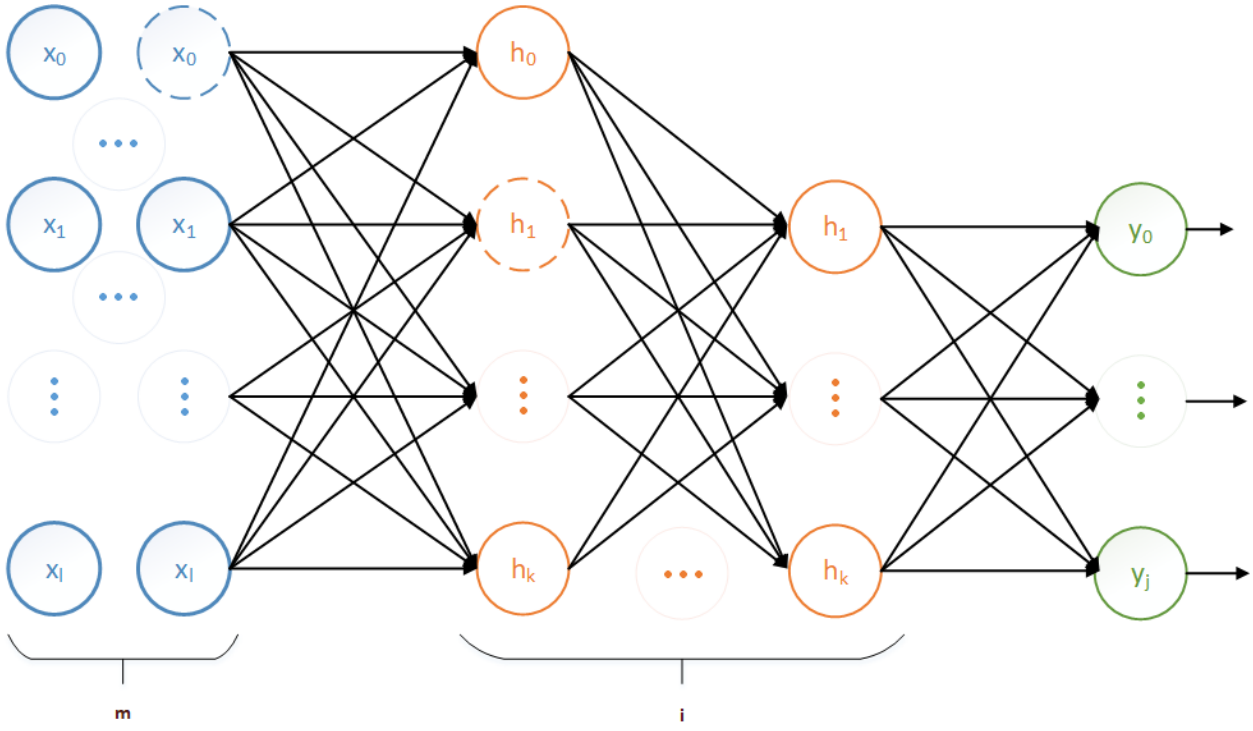


Figure 2. Generic LSTM network showing input, hidden, and output layers, as well as drop-out nodes.

Since a large scale optimization including various folds of the input matrix was out of the scope of this work, we performed a preliminary grid search involving the following parameter choices to guide us to make a final decision on the parameters that defined the network that would be finally used for the forecasting

[a]Apart from the steps outlined above, the Python deep learning package 'keras' required a few tweaks that are not relevant to the discussion here.

example. A grid search involves building an LSTM network for each combination of the multiple parameter sets defined below, and comparing them using a single metric (here, the accuracy score after training). Table 2 lists these hyper-parameter values:

Table 2. LSTM Hyper-parameter choices and values explored via grid search

Hyper-parameter (*number of choices*)	**Values/choices Explored**
Optimizers (7)	Stochastic Gradient Descent [25]
	RMS Propagation [26]
	Adaptive Gradient [27]
	Adaptive Delta [28]
	Adam Optimizer [29]
	Adam Max Optimizer [29]
	Nesterov Adam Optimizer [30]
Loss Functions (9)	Mean squared error
	Mean absolute error
	Mean absolute percentage error
	Mean squared logarithmic error
	Squared hinge
	Hinge
	Log Cosh
	Kullback Leibler Divergence
	Cosine Proximity
Number of LSTM layers or m (6)	1,2,4,6,8,10
Number of LSTM units per layer or l (4)	10,30,50,100
Number of dense unit layers or i (5)	2,4,8,16,32
Drop-out rates (3)	0, 5 and 10 %
Batch Sizes (5)	1,50,100,200,500

The gird search performed involved comparing mean squared loss values for combinations of parameters defining the network in Table 2. The total number of combinations possible equals 113400 ($= 7 \times 9 \times 6 \times 4 \times 5 \times 3 \times 5$). Insights obtained for this study may be reported in detail in a future publication, but here, we discuss the salient outcomes and recommendations for performing the actual training in Section C. We found that the Nadam optimizer (Nesterov Momentum Adam Optimizer) showed superior performance compared to other optimizers for our dataset, all other factors remaining constant. A large difference was seen in the function values reported at the end of the optimization procedure while using multiple loss functions - squared error functions (like the basic mean squared error and mean squared logarithmic error) performed well and showed high levels of validation accuracy when test data was used with the trained model. Fit accuracy improved with increasing number of LSTM layers, but a significant improvement was seen when the number of LSTM units per layer increased. The same was true in the case of dense layers. This suggests that for our problem, the number of LSTM and dense units per layer was more important, however, marginal improvement could be obtained by stacking more layers as well. Non-zero drop-out rates tended to add fluctuations to our training history, and caused the training loss history curve to diverge from the testing loss curve with respect to epochs, a typical example of overfitting the data. Increasing the batch size improved the time taken to train the network, with no change in accuracy scores.

C. Trend Forecasting Using LSTM

Based on the outcome of our grid search, we used the following parameters to define and train our LSTM network (refer to Table 2 for all choices considered for preliminary study): Nesterov Adam Optimizer, mean squared logarithmic error loss function, 1 layer containing 1000 LSTM units, and 6 layers containing dense units with a zero dropout rate. The forecast included 20 subcategories manually coded by us, and 14 other columns obtained either from the raw data or results of one-hot encoding. The network is trained to predict not only what incidents may occur, but also *when* they occur. Thus, temporal patterns are captured along

with other spatial patterns.

As a result of incorporating Step 8 (regarding look-back times) from the previous section, each variable is shifted in time and included in the same row as a factor. For example, with a look-back time of 15 time steps[b], a particular variable or factor, say *variable 1* or $v1(t)$, represents the value of that factor in the current time step, t; we add, as is standard in practice, 15 new variables including $v1(t-1), v1(t-2), \ldots, v1(t-15)$ that contribute to the prediction of output variables at the current time-step including $v1, v2$ etc. While we recognize 34 factors or columns in our input set, the LSTM network considers 510 factors (34×15).

In our experience, a prohibitive number of factors may make the training problem intractable (at least on a local computer). Therefore, we sought out study the relative importance of factors. We performed this by fitting a Random Forest model with 500 trees. Relative importance scores were calculated according to the method outlined in Louppe *et al.* [31] Instead of looking for the best split among all variables, the process outlines randomly selecting a variable and measuring the loss in accuracy. This *impurity* or perturbation introduced to the process can be used to evaluate the importance of a factor in predicting one, or multiple outputs. Specifically, the score used to judge relative importance is the sum of weighted impurity decreases for all nodes in the forest where that particular node is used, averaged over all trees in the random forest. For more details, please refer to [31].

Figure 3 shows a plot of relative importance scores for all 510 variables considered. The periodicity in the plot is not surprising, given how we reframed our data using the look-back time. This means that the same features at previous time steps have similar importance values. The typical process of using this plot to determine features to be selected for actual training is by selecting a threshold relative importance value. As an example, using half of the maximum value as a threshold (about 0.004) yields 62 relatively important factors, including 'ceiling', 'time of day' (especially 1201 to 1800, and 1801 to 2400), 'IMC or VMC' label and daylight) at various look back times. A more realistic threshold, the median of all importance scores (0.0012), was used in our final feature set. This resulted in 255 factors in total, including the 62 factors above, and factors such as 'Gust', 'Descent rate', 'Turbulence', 'Crew Resource Management', 'ATC instruction', 'Ground Traffic', 'Air traffic', 'Landing gear' at various previous time steps. and 'Epoch time difference'. Inclusion of the actual date (through the 'Epoch time difference' factor) of the event only at a lower, more inclusive threshold of 0.0012 (and not at the higher threshold of 0.004) implies that the data does not contain long term relationships, at least relative to relationships that are more local.

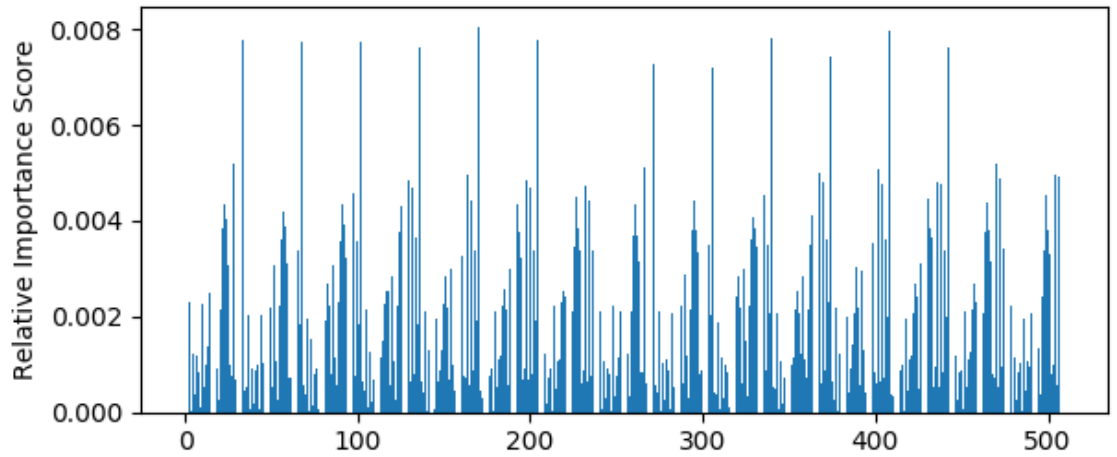


Figure 3. Relative importance scores for all 510 time-dependent variables considered in the study

We generated forecast data corresponding to one year; it is important to note here that unlike fixed time interval forecasting (most other literature that exist fall into this category), time itself is a forecasted, non-linear variable. In our actual implementation, we used trial and error to find an example forecast containing 1687 forecasted samples corresponding to one years worth of forecasted factors. Given the non-deterministic

[b]does not correlate directly with any fixed unit of time, since the data does not contain incidents that are reported in a regular time interval

Table 3. Comparisons of counts and probabilities of various incident types in appearing in the raw ASRS data versus the generated 1 year forecast.

Factor	Original Count	Original Probability	Forecasted Count	Forecasted Probability
Rain/ Ice	34	0.0227	0	0.0000
Storm	23	0.0153	8	0.0047
Gust/ Wind Shear	179	0.1193	8	0.0047
Lightning	2	0.0013	0	0.0000
Turbulence	97	0.0647	22	0.0130
Clouds/ IMC	3	0.0020	3	0.0018
Descent rate	64	0.0427	2	0.0012
Air speed	61	0.0407	0	0.0000
Squawk code/ radio	1	0.0007	2	0.0012
Poor CRM/ Pilot error/ DISP error	156	0.1040	32	0.0190
Runway conditions/ Ceiling/ Visibility	24	0.0160	8	0.0047
Outages/ Infrastructure issues	83	0.0553	0	0.0000
Obstruction/ Terrain	93	0.0620	238	0.1411
ATC instruction	119	0.0793	262	0.1553
Ground traffic	122	0.0813	298	0.1766
Air traffic	177	0.1180	219	0.1298
Automation Confusion/ Autopilot malfunction	39	0.0260	8	0.0047
Landing Gear/ flaps/ slats/ spoiler	289	0.1927	133	0.0788
Strike/ hydraulic/ generic	86	0.0573	99	0.0587
Engine/ propulsion	20	0.0133	0	0.0000

nature of many steps of the process used, one years worth of forecast may correspond to varying number of output rows or samples. Training was performed using the Python package Keras, on computer with 8 Intel Xeon cores and 32 GB of RAM. The training resulted in a loss value (mean squared logarithmic error) of $8.24e-04$, indicating that a good fit was found for the training data, and validated with a loss of 0.0059 for the test data, indicating that the increased number of LSTM cells did not cause overfitting. Note that ideally, the validation loss would also be of the order of the training loss; we continue to work on bridging this gap by further model exploration. At this juncture, the model has been trained, and validated with test data (30% sample), and we may proceed with discussing the actual forecasting of incidents.

Forecasting was generated by re-training the model with the entire raw data (rather than using the 70% sample training data), and using windows of 15 samples to generated the sample. From the sample output we generated, a comparison of incidents we found in the raw data versus incidents forecasted is made in Table 3. A detailed discussion of the patterns of incidents generated, local time variations and factor relationships are out of the scope of this paper, but are equally, if not more interesting by themselves.

Note once again that this is sample forecast data, and only an example set that we would like to present. Multiple folds of forecast data may be generated even with the same trained network, and of course, with networks defined using other hyper-parameters. Forecasts are highly sensitive to the chosen lookahead time, as well as the tuned network weights and batch sizes (not discussed in detail here).

It is interesting to see that the network more-or-less maintains the order of counts and probabilities for each incident type. Four of the six most commonly reported incident types (Obstruction/ Terrain, ATC instruction, Ground Traffic and Air traffic) show increases in counts and probabilities; Gust/ Wind and Landing gear/ flaps/ slats/ spoiler related incidents showed an overall decrease. Incident types containing

low to moderate counts such as Strike/ hydraulic, Turbulence, Squawk code, and Clouds/ IMC retained similar low values of count and probability in the forecast. Not surprisingly, low count factors that have a forecast count of zero (Rain/ Ice, Lightning, Air speed, Outages/ Infrastructure issues, and Engine/ propulsion related) and a probability of 0.0 are artifacts of the relatively decreasing number of incidents characterized by these factors in the original data.

Forecasts such as the one discussed above, and in general, generated by this method may be used where datasets are sparse, multi-variate in nature, and erratically reported. The forecasted and summarized data only intends to inform the reader of another flexible method available to study real-world data with complex structures, and highlight, growing or diminishing trends.

IV. Conclusion

In this paper, we explored the use of LSTM networks in the forecasting of incident data derived from the NASA's ASRS incident database. First, we filtered out data relevant to Go-arounds and Missed-approaches. Then, we queried the ASRS database to obtain all reported incidents pertinent to GARs or MAs between 1989 and 2017. We identified 20 causal mechanisms, and classified the incidents based on their causes, where each incident could have more than one cause. While the ASRS database can provide valuable insight into situations that did not result in accidents, it has several well-documented shortcomings. Common problems with any voluntary reporting system include under-reporting and bias associated with the reporting. Further, while several reports might be submitted, resource constraints result only in a small proportion of reports being transcribed. Data from the ASRS database are also not suitable for frequentist or statistical analysis. The LSTM forecast was tested using 30% of the data and had a low validation error. The sample one-year forecast generated using the trained network provided insights into incident types with increased probability of occurrence in the following year, and also factors in this dataset that contributed significantly to the multiple outputs being predicted.

Acknowledgments

The Authors would like to thank the Center for Aviation Studies and Robust Analytics, Inc. for permitting this independent research project.

References

[1] Moriarty, D. and Jarvis, S., "A systems perspective on the unstable approach in commercial aviation," *Reliability Engineering & System Safety*, Vol. 131, No. Supplement C, 2014, pp. 197–202.

[2] Matthews, B., Das, S., Bhaduri, K., Das, K., Martin, R., and Oza, N., "Discovering Anomalous Aviation Safety Events Using Scalable Data Mining Algorithms," *Journal of Aerospace Information Systems*, Vol. 10, No. 10, 2013, pp. 467–475.

[3] Lee, Y.-H. and Liu, B.-S., "Inflight workload assessment: Comparison of subjective and physiological measurements," *Aviation, Space, and Environmental Medicine*, Vol. 74, No. 10, 2003, pp. 1078–1084.

[4] Wang, Z., Sherry, L., and Shortle, J., "Airspace risk management using surveillance track data: Stabilized approaches," *2015 Integrated Communication, Navigation and Surveillance Conference (ICNS)*, April 2015, pp. W3–1–W3–14.

[5] Sherry, L., Wang, Z., Kourdali, H. K., and Shortle, J., "Big data analysis of irregular operations: Aborted approaches and their underlying factors," *2013 Integrated Communications, Navigation and Surveillance Conference (ICNS)*, April 2013, pp. 1–10.

[6] Puranik, T., Harrison, E., Min, S., Jimenez, H., and Mavris, D., "General Aviation Approach and Landing Analysis Using Flight Data Records," *16th AIAA Aviation Technology, Integration, and Operations Conference*, 2016, Paper No. AIAA 2016-3913, doi:10.2514/6.2016-3913.

[7] Puranik, T. G., Jimenez, H., and Mavris, D. N., "Utilizing Energy Metrics and Clustering Techniques to

Identify Anomalous General Aviation Operations," *AIAA SciTech Forum*, Jan. 2017, Paper No. AIAA 2017-0789, doi:10.2514/6.2017-0789.

[8] Puranik, T. G. and Mavris, D. N., "Identifying Instantaneous Anomalies in General Aviation Operations," *17th AIAA Aviation Technology, Integration, and Operations Conference*, 2017, Paper No: AIAA-2017-3779. doi:10.2514/6.2017-3779 .

[9] Rao, A. H. and Marais, K., "Comparing Hazardous States and Trigger Events in Fatal and Non-Fatal Helicopter Accidents," *16th AIAA Aviation Technology, Integration, and Operations Conference*, American Institute of Aeronautics and Astronautics, June 2016, Paper No. AIAA-2016-3916, doi:10.2514/6.2016-3916.

[10] Rao, A. H. and Marais, K., "High Risk Occurrence Chains in Helicopter Accidents," *Reliability Engineering and System Safety*, Vol. 170, 2018, pp. 83–98, doi:10.1016/j.ress.2017.10.014.

[11] Shappell, S., Hackworth, C., Holcomb, K., Lanicci, J., Bazargan, M., Baron, J., Iden, R., and Halperin, D., "Developing Proactive Methods for General Aviation Data Collection," Technical Report DOT/FAA/AM-10/16, Federal Aviation Administration, Washington, DC, Nov. 2010.

[12] Fultz, A. J. and Ashley, W. S., "Fatal weather-related general aviation accidents in the United States," *Physical Geography*, Vol. 37, No. 5, 2016, pp. 291–312.

[13] Rao, A. H. and Marais, K., "Identifying High-Risk Occurrence Chains in Helicopter Operations from Accident Data," *15th AIAA Aviation Technology, Integration, and Operations Conference*, 2015, Paper No. AIAA 2015-2848, doi:10.2514/6.2015-2848.

[14] Gavrilovski, A., Jimenez, H., Mavris, D. N., Rao, A. H., Shin, S., Hwang, I., and Marais, K., "Challenges and Opportunities in Flight Data Mining: A Review of the State of the Art," *AIAA SciTech Forum*, American Institute of Aeronautics and Astronautics, Jan. 2016, Paper No: AIAA 2016-0923 doi:10.2514/6.2016-0923 .

[15] Zheng, X. and Liu, M., "An overview of accident forecasting methodologies," *Journal of Loss Prevention in the Process Industries*, Vol. 22, No. 4, 2009, pp. 484 – 491.

[16] Luxhøj, J. T., "Risk Analysis of Human Performance in Aviation Maintenance," *16th Human Factors in Aviation Maintenance Symposium*, Human Factors in Aviation Maintenance, April 2002.

[17] Lipton, Z. C., Berkowitz, J., and Elkan, C., "A critical review of recurrent neural networks for sequence learning," *arXiv preprint arXiv:1506.00019*, 2015.

[18] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y., "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014.

[19] Sainath, T. N., Vinyals, O., Senior, A., and Sak, H., "Convolutional, long short-term memory, fully connected deep neural networks," *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, IEEE, 2015, pp. 4580–4584.

[20] Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., and Schmidhuber, J., "LSTM: A search space odyssey," *IEEE transactions on neural networks and learning systems*, 2017.

[21] Hochreiter, S. and Schmidhuber, J., "Long short-term memory," *Neural computation*, Vol. 9, No. 8, 1997, pp. 1735–1780.

[22] Tai, K. S., Socher, R., and Manning, C. D., "Improved semantic representations from tree-structured long short-term memory networks," *arXiv preprint arXiv:1503.00075*, 2015.

[23] Gers, F. A., Schmidhuber, J., and Cummins, F., "Learning to Forget: Continual Prediction with LSTM," *Neural Computation*, Vol. 12, No. 10, 2000, pp. 2451–2471.

[24] Gers, F. A., Eck, D., and Schmidhuber, J., "Applying LSTM to time series predictable through time-window approaches," *Neural Nets WIRN Vietri-01*, Springer, 2002, pp. 193–200.

[25] Hardt, M., Recht, B., and Singer, Y., "Train faster, generalize better: Stability of stochastic gradient descent," *arXiv preprint arXiv:1509.01240*, 2015.

[26] Dauphin, Y., de Vries, H., and Bengio, Y., "Equilibrated adaptive learning rates for non-convex optimization," *Advances in neural information processing systems*, 2015, pp. 1504–1512.

[27] Duchi, J., Hazan, E., and Singer, Y., "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, Vol. 12, No. Jul, 2011, pp. 2121–2159.

[28] Zeiler, M. D., "ADADELTA: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[29] Kingma, D. and Ba, J., "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[30] Dozat, T., "Incorporating nesterov momentum into adam," *Stanford University Press*, 2016.

[31] Louppe, G., Wehenkel, L., Sutera, A., and Geurts, P., "Understanding variable importances in forests of randomized trees," *Advances in neural information processing systems*, 2013, pp. 431–439.