Prediction of aircraft estimated time of arrival using a supervised learning approach

James Z. Wells * **Tejas G. Puranik** **
Krishna M. Kalyanam *** **Manish Kumar** ****

* *NASA OSTEM Intern, University of Cincinnati, Cincinnati, OH 45221 USA (e-mail: wells2jz@mail.uc.edu)*
** *USRA, NASA Ames Research Center, Moffett Field, CA 94035 USA (e-mail: tejas.puranik@nasa.gov)*
*** *NASA Ames Research Center, Moffett Field, CA 94035 USA (e-mail: krishna.m.kalyanam@nasa.gov)*
**** *University of Cincinnati, Cincinnati, OH 45221 USA, (e-mail: kumarmu@ucmail.uc.edu)*

Abstract:
We present a novel data-driven approach for prediction of the estimated time of arrival (ETA) of aircraft in the terminal area via the implementation of a Random Forest regression model. The model uses data fused from a number of sources (flight track, weather, flight plan information, etc.) and provides predictions for the remaining flight time for aircraft landing at Dallas/Fort Worth (DFW) International Airport. The predictions are made when the aircraft is at a distance of 200-miles from the airport. The results show that the model is able to predict estimated time of arrival to within ± 5 min for 90% of the flights in the test data with the mean absolute error being lower at 145 seconds. This paper covers the entire pipeline of data collection, pre-processing, setup and training of the ML model, and the results obtained for DFW.

Keywords: machine learning, random forest regression, estimated time of arrival, air traffic management

1. INTRODUCTION

Improving terminal descent and approach is a key element of the Federal Aviation Administration (FAA) Next Generation Air Transportation System (NextGen). According to FAA (2013), improvements in this phase of flight will result in a better utilization of the National Airspace System (NAS), improve efficiency of the runways, reduce fuel usage and costs, and most importantly increase safety. One improvement that aligns with the goals of the NextGen program is the development of more accurate arrival time predictions. Accurately predicting estimated time of arrival (ETA) for aircraft landing at an airport is a crucial enabler for efficient airspace operations that can benefit several stakeholders in the aviation ecosystem. Airlines can provide additional value to their customers by providing more accurate time of arrival, especially if passengers or crew need to make connecting flights. Airport operators can use such predictions to schedule any services and maintenance required for inbound aircraft. Traffic management personnel can utilize it to safely manage the airspace and maintain efficient flow of arriving aircraft by using it to determine the sequence and schedule of arrival flights. It can also improve runway efficiency by minimizing the amount of time runways have to be reserved for landing flights. This could, in turn, reduce flights having to go into holding patterns that requires more fuel to be burnt, and increases flight times.

Due to the various uses for accurate landing time predictions, several previous efforts have focused on improving these estimates using either physics-based models, machine learning (ML) models, or a combination of the two. Several existing ground-based decision-support tools provide predictions of landing times such as the FAA's Time Based Flow Management (TBFM) or Traffic Flow Management System (TFMS). These new systems improve FAA legacy systems that were composed purely on physics based models and static look-up tables based on limited historical data. Arrival times produced by various existing systems can be for different purposes and therefore, might not be identical or consistent across different systems. For physics-based models, while the physics of trajectory prediction is well-established for conventional fixed-wing aircraft, it requires knowledge of aircraft performance parameters (e.g., drag coefficients) and operating procedures (e.g., descent speed, flap schedule) for the flight being predicted. This information may not always be accurately available and could limit the applicability of these models. The work presented in this paper uses historical flight data from the Dallas/Fort Worth (DFW) International Airport and applies the Random Forest (RF) regression machine

⋆ This work was supported by NASA Office of STEM Engagement

learning algorithm to predict remaining flight time. For this paper, the RF predicts the remaining flight time once an aircraft is approximately 200-miles from the airport. The 200-mile radius was selected as this is approximately the top of descent (ToD) for flights landing at DFW.

ML algorithms have become more widely applied in air traffic management research domain because it has an abundance of data from many sources such as flight track data, flight plan information, meteorological data, traffic flow management data, etc. that provides a rich source for data driven methods. This data consists of numerical data (aircraft speed, position, etc.) and categorical data (desired approach vector, day of the week, etc.). The different types of data the ML algorithms can be trained on typically have relationships and patterns that are not easily detected by people. Examples of ML within the air traffic management domain, not related to ETA prediction, can be seen with flight deviation detection and prediction as seen in Bleu Laine et al. (2022), identification and mitigation of loss of separation events as seen in Hawley and Bharadwaj (2018) and finally airspace sector occupancy seen in Brito et al. (2021).

In the domain of ETA prediction, current methods can be broken down into pre-flight and in-flight methods. Pre-flight methods assign estimated arrival times when flights are initially scheduled. According to Jha et al. (2012), the scheduling process starts when airlines request flight plans to be developed via the Flight Operations Centers (FOCs). This process can start at midnight the day of the flight or a minimum of 45 minutes prior to flight. FOCs develop flight plan requests that are then passed to the Air Traffic Control System Command Center (ATCSCC) for approval. These flight plans take into consideration predicted NAS usage and weather and are subject to change after approval based on evolving conditions. Even with the pre-flight planning, arrival times can change, especially during flight, thus there is a need for accurate in-flight prediction methods to be developed.

Current in-flight prediction methods depend heavily on aircraft performance models (APMs). One such model developed and maintained by Eurocontrol and widely used in ATM research is the Base of Aircraft Data (BADA) models, see Nuic et al. (2010). BADA uses total energy models along with vehicle-specific parameters to model the aircraft during flight. These models are then used to estimate the performance of the aircraft, that can then be used to estimate landing times. Limitations to BADA include incomplete information about aircraft parameters such as drag coefficients and its inability to capture airspace specific procedures. One way to improve BADA is to develop new performance parameters based on historical data as seen in Fernandes et al. (2023).

Yan et al. (2012) used a Random Forest (RF) model for prediction and quantification of aircraft landing times at DFW. Their work focused on aircraft 60 nautical miles (nm) away from the airport and closer. With the RF they implemented, they were able to achieve a mean absolute error bar of 75.4 seconds when the aircraft was 60 nm away. This RF was trained on $4,011$ flights. The data on which the RF model was trained consisted of Euclidean distance from the aircraft to the airport, latitude, longitude, altitude, heading, speed, track start location, time of day, sample times, visual flight rules vs. instrument flight rules and assumed runway availability. The study does not report explicit hyperparameter tuning or usage of flight plan data in making predictions. This work aims to improve on these predictions by providing the estimated landing times much further out ($\approx 174nm$).

The work in Strottmann Kern et al. (2015) uses a RF model that consists of 100 trees and was trained on $24,787$ flights. This paper modeled the full flight instead of just the terminal descent portion. It also was not limited to aircraft landing at one airport. With the RF they implemented, they were able to reduce the mean absolute error of the FAA Enhanced Traffic Management System by 42.7%, however the actual prediction errors by the model were unclear.

Basturk and Cetek (2021) explore applications of RF and deep neural networks for predicting ETA. Both machine learning algorithms were able to predict ETA after an aircraft takes off and then again once it enters the terminal maneuvering area. The ML algorithms were trained with $63,460$ commercial flights and were able to achieve an Mean Absolute Error (MAE) of less than 3 minutes upon entering the Terminal Maneuvering area.

Silvestre et al. (2021) use Long-Short Term Memory (LSTM) Neural Networks (NN) to predict arrival times and focus on the last 2 hours of flight data. With their trained network, they were capable of getting a MAE of 340 seconds for the last 90 minutes of flight. The parameters that the LSTM-NN was trained on included latitude, longitude, altitude data, day of the week, the time of day and if it was a holiday.

The algorithm presented in this paper is purely ML-based and does not rely on physics. This provides an advantage of being broadly applicable at multiple airports as long as historical data is available. We use an RF to identify and model aircraft trends without the mathematical models that are dependent on variables such as drag coefficients, aircraft mass and thrust settings. We used an off-the-shelf implementation of Random Forest regression developed by SciKit-learn library in python Pedregosa et al. (2011). The main disadvantage with a purely ML approach is if the system changes drastically, the AI/ML approach will need to be retrained on new data. To address this disadvantage, we have developed robust data preparation methods to quickly prepare new training and testing sets if the system changes. The RF method we are developing is different from the literature because: it incorporates flight plan information via the standard terminal arrival routes (STAR), requires fewer flights than most of the works cited, and focuses on the approximate top-of-descent phase for aircraft (that is further out from the airport than most prior work) and thereby provides more time for stakeholders to make decisions. A STAR is an ATC coded IFR arrival route established for application to arriving IFR aircraft destined for certain airports. STAR routes simplify clearance delivery procedures and also facilitate transition between en-route and instrument approach procedures markers. [1] The STAR routes for DFW can be seen

[1] https://www.faa.gov/air_traffic/publications/atpubs/aim_html/chap5_section_4.html

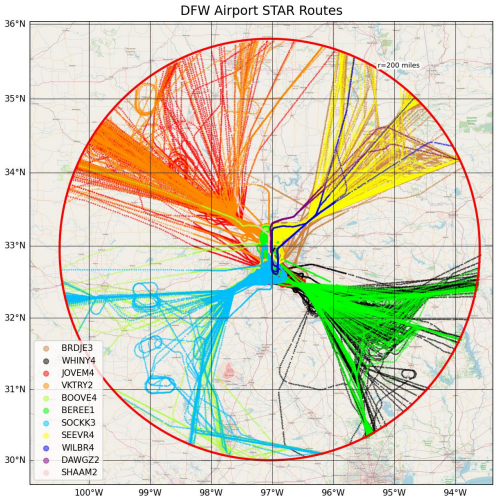

Fig. 1. Map showing flight trajectories with STAR routes

in figure 1. The introduction of flight plan information via STAR routes to the RF allows us to better capture the standard behavior of flights landing at DFW. Looking at the top of descent pushes our prediction distance out further that allows for more time to make crucial decisions. Requiring fewer flights is beneficial as it means this system can be retrained for another airport easier and more quickly. The literature survey shows that an RF is capable of generating ETA for commercial aircraft; however the results obtained by the RF presented here are unable to be directly compared with results from the literature survey as the RF was trained using a different data set, focusing on different airports and using a larger prediction distance.

2. PROBLEM FORMULATION

When developing an RF model for ETA prediction, we had to clearly define what problem we were solving, our desired outcome and the input parameters for our system based on the data we had available. We wanted to develop a machine learning algorithm that could fuse flight track, metadata and weather data to generate reliable and accurate predictions for aircraft landing at an airport. The desired outcome of this work is to develop a framework that could be applied at any airport, given the data.

One of the problem parameters we needed to decide was when to make the remaining time prediction. We selected a 200-mile radius around the airport because this was the approximate distance from the airport that aircraft landing at DFW started their descent (top of descent). In addition to determining where the ETA prediction is made, this work also had to define the point in the future for which the ETA is calculated. In an ideal scenario, this would be the touchdown point of the aircraft on the runway. However, due to noisiness in the track data near the runway surface, altitude differences between the runways, effect of deviations in atmospheric pressure on altitude sensors, etc. we chose the prediction point as the one when the aircraft goes below the average altitude of DFW airport plus 100 feet. A final parameter we had to choose was what inputs we could use for training the RF. The proposed system relies on historical flight data

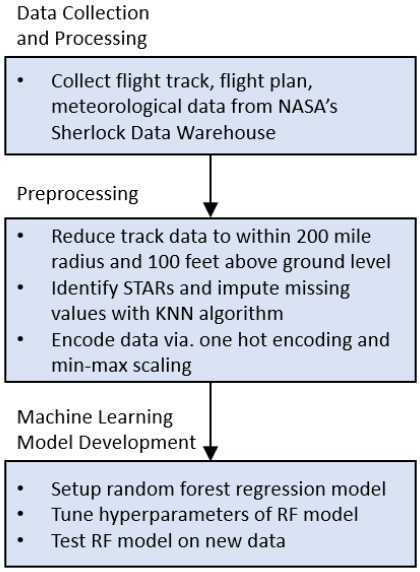

Fig. 2. Flow Chart for Data Preparation and RF Training

to train the model; however, the goal is to eventually implement this on real-time data, hence the RF we trained would need to use readily available, real-time data coming from weather facilities at the destination airport and information received by ADS-B and other airport facilities. In addition to only using available data, we also wanted to limit the data we used to avoid making our model overly complex which could take longer to train and more prone to overfitting.

3. METHODOLOGY

3.1 Data Preparation

In order to train the RF regression model, we had to first develop the training and testing set. To create the training and testing set, large amounts of flight data had to be collected and processed. Our data set had 13,302 flights pulled from flight data collected between March of 2019 through August of 2019. We followed the steps seen in figure 2 to process the data and create our training and testing set. Our training and testing sets consisted of the input and output parameters shown in table 1. The data we used were provided through NASA's Sherlock Data Warehouse that also has an open source version (Arneson et al. (2019)). The data was downloaded on a day-by-day basis and contained full-flight data for aircraft operating within the NAS. After obtaining the data, we had to identify and isolate flights landing at DFW.

After isolating the DFW flights, we applied the Haversine Algorithm(Robusto (1957)) to all of the flight data. Applying the Haversine algorithm to the data was done to identify where the 200-mile threshold value was for each flight. The Haversine algorithm can approximate the great circle distance between a pair of latitude and longitude points; going from degrees latitude and longitude to straight line distance measured in miles. The 200-mile threshold was used as the starting index for our data collection. Once we identified the 200-mile threshold, we then looked at each flight to identify when the aircraft

Table 1. Parameters used in RF Training

Parameter	Definition	Input/Output
lat	Aircraft Latitude	Input
lon	Aircraft Longitude	Input
alt	Aircraft Altitude	Input
CAS	Aircraft Calibrated Air Speed	Input
WS	Wind Speed	Input
WD	Wind Direction	Input
Pres	Atmospheric Pressure	Input
Temp	Atmospheric Temperature	Input
STAR	Standard Terminal Arrival Route	Input
dt	Elapsed Time	Output

reached below the average altitude of DFW plus 100 feet. The data between the 200-mile threshold and the final point was saved for each flight. The data we saved would further be reduced down to the first 10 latitude, longitude and altitude points. The average of the first 10 calibrated airspeeds was calculated and saved. Then the time elapsed between the 200-mile threshold and landing was calculated and used as our desired output label for the RF regression model.

After filtering all available data, we needed to fill in any missing data values. The main input variable that was missing data was the Standard Terminal Arrival Routes (STAR). To find the missing STAR values, we identified the flights that had either initial or final STAR values and developed a sub-training set. This sub-training set was different from the one we are developing for the RF and is used with a K-nearest neighbors (KNN) Classifier algorithm. The KNN used latitude and longitude inputs and grouped them based on the known STAR values. The flights with unknown stars were imputed via the neighborhood we had developed based on the flights with known STAR data. We used the SciKit KNN-Classifier for the development of our KNN (Pedregosa et al. (2011)). The KNN was only used for the assignment of missing STAR values and not directly for ETA predictions. The KNN we trained could be applied to real-time data to assign missing STAR values of aircraft in flight once they reach the 200-mile radius.

After accounting for missing values, we encoded the data using one hot encoding and min-max encoding. One hot encoding was used on categorical data while min-max encoding was used on numerical data. One hot encoding converts a single feature with n unique data points into n unique features. Each of the n features becomes either 0 or 1 values. The 0 indicates that the current feature does not apply to a flight while the 1 value indicates that the feature does apply. We used one hot encoding on the wind direction and STAR values. The additional categories generated by one hot encoding can be seen in table 2. Min-max encoding scales numerical data in the column between 0 and 1 based on the minimum and maximum values. Min-max encoding is applied to our data to reduce the effect of outliers. Min-max encoding was applied to all of the latitude, longitude and altitude data points, the calibrated airspeed, atmospheric pressure and temperature, and the wind speed. After encoding the data, we split the data into two new data sets to create the training and testing sets. We allocated 80% (10,647 flights) of the data for training and the remaining 20% (2,655 flights) for testing. The allocation was done through random assignment. The

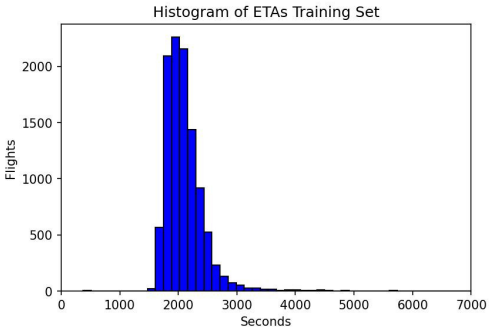

Fig. 3. Histogram showing number of training set flights based on remaining flight time

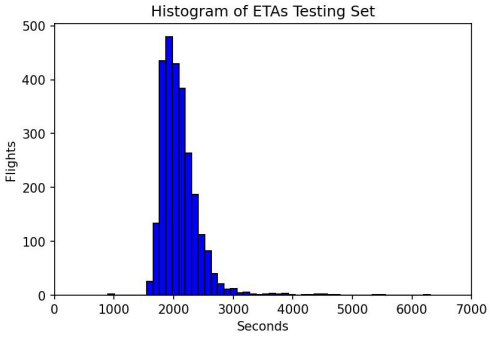

Fig. 4. Histogram showing number of testing set flights based on remaining flight time

time remaining (our output label) vs. number of flights for the training and testing sets breakdowns can be seen in figures 3 and 4 respectively. These two plots show that the testing set is representative of the training set and does not include a large number of outliers or a shift in distribution.

3.2 Random Forest Regressor

After developing the testing and training sets, we trained the Random Forest regression model. The RF model tries to learn the best mapping between the provided input and output values. The learning process stops when the individual decision trees reach a stopping criteria. The stopping criteria in our case is either the maximum depth or minimum number of samples at each node. After completing the training process, we applied the trained RF to the testing set data. The RF has not seen the testing set prior to this so it must generate the predicted ETA based on what it has previously learned. In order to systematically select the best RF model, we completed several initial iterations of the RF training process where we varied the input and hyperparameters. Initially when developing our RF, we started with a simpler set of input data consisting of only position and speed data. We then systematically included more parameters to achieve better results. For the hyperparameter tuning, we used a combination of manual and automatic parameter tuning to achieve the best fit the random forest could achieve on our data set. The three main hyperparameters we looked at when training the RF included the number of trees in the forest, the maximum tree depth, and the minimum samples at the leafs. The automatic hyperparameter tuning we used

Table 2. Encoded Parameter Catagories

Parameter	Members
Wind Direction	North (N), East (E), South (S), West (W), Northeast (NE), Northwest (NW), Southeast (SE), Southwest (SW)
STAR	BEREE1, BLOND5, BOOVE4, BRDJE3, CABBY2, CAINE2, COSTR3, CQY8, DAWGZ2, FINGR5, FORNY2, HOBTT2, JEN1, JOVEM4, PAWLZ3, SEEVR4, SHAAM2, SOCKK3, TILLA3, UKW5, VKTRY2, WHINY4, WILBR4

was based on a grid search algorithm to systematically vary the parameters. To determine the best model, the grid search parameter calculated the mean squared error (MSE). The RF with the lowest MSE on the validation set was recorded. In addition to looking at these metrics for accuracy, we also had to make sure we were not overfitting the RF. We made sure we weren't overfitting the RF by looking at the RF's performance on the test and training set. If the RF did extremely well for the training set, but performed poorly on the testing set, we knew overfitting was likely and readjusted the model parameters.

4. RESULTS AND DISCUSSION

The results for the RF regression model can be seen in figures 5 through 7 and in Table 3. The results we have presented in this paper include a feature importance plot (figure 5), 5^{th} through 95^{th} inter percentile plots for the training and testing set data (figures 6 and 7) and finally a table showing the MAE and Mean Absolute Percentage Error (MAPE) for the training and testing set (Table 3).

To develop the results shown, many iterations of the RF had to be trained and tested to find the best RF. We found the best RF for our data through the grid search and manual tuning methods mentioned in the previous section. The best RF had 10 estimators, 26 nodes deep and had a minimum of 2 samples per leaf. For our data set, training a single iteration of the RF took ≈ 0.63 seconds. The automatic hyperparameter tuning took longer and was dependent on the number of parameters being varied. For each combination of the hyperparameter tuning, a new RF had to be trained and evaluated. The training took place on a laptop computer with an Intel Core i5 1.6GHz processor and 8 gigabytes of RAM.

Figure 5 shows the importance of each input variable to the best RF we found when it is generating the predicted ETA. It can be seen from figure 5 that the position, STAR, and wind speed are some of the most important parameters.The importance data is useful when developing an RF because it shows how important inputs are in determining the final output. The importance can be useful to gauge the impact of new parameters or to trim down the RF to the most important inputs. The system presented did not need to be trimmed down as it is not constrained by complexity of the model; as adding additional inputs did not negatively affect performance the output.

Figure 6 and 7 show our error generated by applying the trained RF to the training set and to the testing set.

Table 3. Training and Test Set Statistics

Data Set	MAE (sec)	MAPE (%)
Training Set	127.31	5.93
Testing Set	145.16	9.34

The error data shown in figures 6 and 7 is the difference between each of the true values in the training/testing sets and the corresponding value generated by the trained RF for every flight. Figures 6 and 7 show that the trained RF is capable of identifying data within the 5^{th} through 95^{th} inter-percentile range with maximum error bounds from the true landing time being $\pm$ 120 seconds and $\pm$ 300 seconds, respectively. The Mean Absolute Error for the full training set is 127.31 seconds and the MAE for the full test set is 145.16 seconds. As expected, the errors observed in the training set are less than the errors observed in the testing set. The existence of the training and testing set errors and the relatively small difference between them shows that we did not over-fit the RF Regressor. Had the training set error been extremely low and the testing set error extremely high, then it would have been an indication that we had over-fitted the system. Our RF produced more accurate results when compared to work presented in the introduction Yan et al. (2012).

Table 3 shows typical statistics metrics for the errors generated by the RF model on the full data set. The statistics we used were the Mean Absolute Error (MAE) and Mean Absolute Percent Error (MAPE). MAE and MAPE are useful statistics as they are less sensitive to errors caused by outliers. MAE shows the average absolute value of the errors while MAPE is an average of the percent error. As expected, and as seen in figures 6 and 7, the results for the testing set had a larger error when compared to the training set.

The current results produced by the trained RF and presented in this paper are useful for less critical tasks such as aircraft maintenance scheduling or ensuring baggage handling services are at the gate when a flight arrives, however our results are currently not accurate enough for runway scheduling. The training and testing sets included aircraft that went into holding patterns resulting in longer than predicted flight times for several flights. One reason an aircraft can go into a holding patter is due to runway availability and other air traffic present around the airport. Incorporating runway availability would increase the accuracy of our system. Removing the flights that went into holding patterns would also increase accuracy of our system, however would be less representative of a real world application of our system.

5. CONCLUSION AND FUTURE WORK

We presented the development, training and testing of a Random Forest regressor model for prediction of estimated time of arrival for aircraft landing at the Dallas/Fort Worth International Airport. The RF model we developed predicted the ETA of 90% of the aircraft in the test set to within $\pm$ 5 minutes. Incorporating the STAR parameter in the RF has increased the accuracy of the system because the STAR parameter contains a large amount of information about the future flight track which helps improve the time estimate.

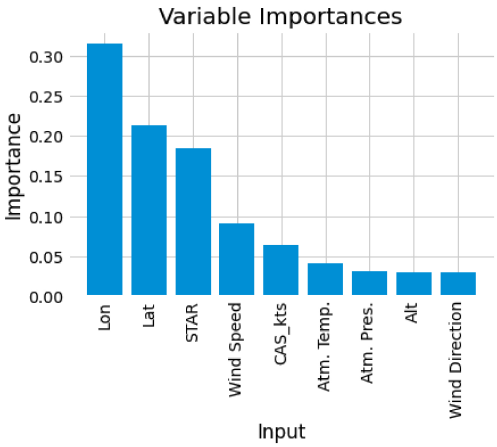

Fig. 5. Input Value (Feature) Relative Importance for RF model prediction

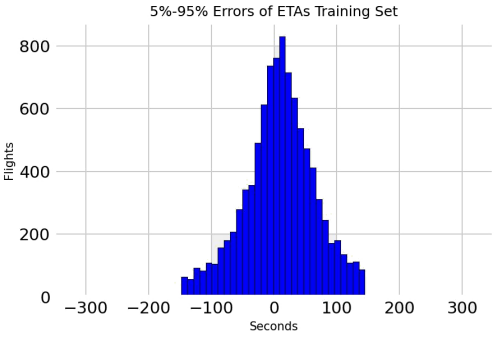

Fig. 6. 5th to 95th Percentile Training Set Errors

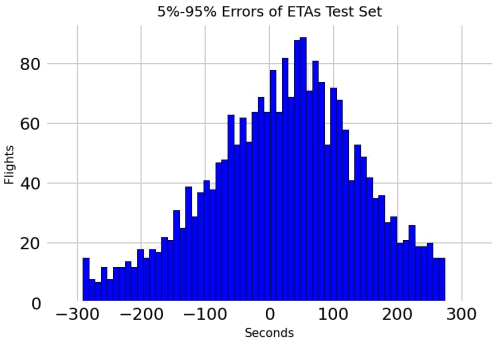

Fig. 7. 5th to 95th Percentile Testing Set Errors

Future work includes exploration of additional model parameters to see if we can further increase the prediction accuracy. Other parameters to explore include: flight rules (instrument vs. visual), aircraft type, runway availability, current traffic in the vicinity and current precipitation. Another avenue of future work is to extend the RF model to other airports and test its performance capabilities and limitations.

REFERENCES

Arneson, H.M., Hegde, P., La Scola, M.E., Evans, A.D., Keller, R.M., and Schade, J.E. (2019). Sherlock data warehouse. URL `https://ntrs.nasa.gov/citations/20190025090`.

Basturk, O. and Cetek, C. (2021). Prediction of aircraft estimated time of arrival using machine learning methods. *The Aeronautical Journal*, 125(1289), 1245–1259. doi:10.1017/aer.2021.13.

Bleu Laine, M.H., Puranik, T.G., Mavris, D.N., and Matthews, B. (2022). Multiclass multiple-instance learning for predicting precursors to aviation safety events. *Journal of Aerospace Information Systems*, 19(1), 22–36. doi:10.2514/1.I010971. URL `https://doi.org/10.2514/1.I010971`.

Brito, I.R., Murca, M.C.R., d. Oliveira, M., and Oliveira, A.V. (2021). *A Machine Learning-based Predictive Model of Airspace Sector Occupancy*. doi:10.2514/6.2021-2324. URL `https://arc.aiaa.org/doi/abs/10.2514/6.2021-2324`.

FAA (2013). *NextGen implementation plan*, 56–58.

Fernandes, A., Wesely, D., Puranik, T.G., Rohani, A.S., Kalyanam, K.M., and Morin, D. (2023). *Prediction of Critical Aircraft Performance Model Parameters from Historical Flight Data*. doi:10.2514/6.2023-2532. URL `https://arc.aiaa.org/doi/abs/10.2514/6.2023-2532`.

Hawley, M. and Bharadwaj, R. (2018). Application of reinforcement learning to detect and mitigate airspace loss of separation events. In *2018 Integrated Communications, Navigation, Surveillance Conference (ICNS)*, 4G1–1–4G1–10. doi:10.1109/ICNSURV.2018.8384897.

Jha, P., Suchkov, A., Crook, I., Tibitche, Z., Lizzi, J., and Subbu, R. (2012). *NextGen Collaborative Air Traffic Management Solutions*. doi:10.2514/6.2008-6329. URL `https://arc.aiaa.org/doi/abs/10.2514/6.2008-6329`.

Nuic, A., Poles, D., and Mouillet, V. (2010). BADA: An advanced aircraft performance model for present and future atm systems. *International Journal of Adaptive Control and Signal Processing*, 24(10), 850–866. doi: https://doi.org/10.1002/acs.1176.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.

Robusto, C.C. (1957). The cosine-haversine formula. *The American Mathematical Monthly*, 64(1), 38–40. URL `http://www.jstor.org/stable/2309088`.

Silvestre, J., de Santiago, M., Bregon, A., Martínez-Prieto, M.A., and Álvarez Esteban, P.C. (2021). On the use of deep neural networks to improve flights estimated time of arrival predictions. *Engineering Proceedings*, 13(1). doi:10.3390/engproc2021013003. URL `https://www.mdpi.com/2673-4591/13/1/3`.

Strottmann Kern, C., de Medeiros, I.P., and Yoneyama, T. (2015). Data-driven aircraft estimated time of arrival prediction. In *2015 Annual IEEE Systems Conference (SysCon) Proceedings*, 727–733. doi:10.1109/SYSCON.2015.7116837.

Yan, G., Jordan, R., and Ishutkina, M. (2012). A tree-based ensemble method for the prediction and uncertainty quantification of aircraft landing times. *10th Conference on Artificial Intelligence Applications to Environmental Science*, 10.