Model documentation and write-up

1. Who are you (mini-bio) and what do you do professionally?

McDonald is a graduate student in the Department of Aeronautics and Astronautics at MIT and a National Science Foundation Graduate Research Fellow. Before his work at MIT, McDonald worked in autonomy/controls at major aerospace companies including Boeing Commercial Airplanes and Aurora Flight Sciences. Presently, he is researching optimal scheduling and dispatching for future air transportation systems using machine learning and optimization methods. He received a Bachelor of Science in Aerospace Engineering and Applied Mathematics from the University of Tennessee.

Marek Travnik is also a graduate student in the Department of Aeronautics and Astronautics at MIT. He works as a Research Assistant in the International Center for Air Transportation laboratory, supervised by professor John Hansman. The broad focus of his research is on data-driven methods in air transport operations - currently automated runway surface condition assessment and data-driven approaches for assessing the fuel efficiency of the global aviation system. He received a Bachelor of Science in Aerospace Engineering from Delft University of Technology.

2. What motivated you to compete?

McDonald and Travnik are both graduate students in the Aerospace Engineering program at MIT. We both are passionate about the aerospace discipline and the application of machine learning technologies to solve some of the hardest problems in the aerospace field. The Runway Functions competition had both an exciting and relevant application in addition to providing an opportunity for us to explore innovative ML and data science methods to predict future airport configurations.

3. High-level summary of your approach: What did you do and why?

The overall approach presented in this paper involved three main components. The first was Data Processing - performing feature engineering on the raw data to create understandable features for prediction models. The second was Submodels - train prediction models that learn relevant subtasks that are related to the overarching task of predicting runway configurations. The third and last component was the Final Model - train dense neural network that leverages the outputs of the submodels to make the final probability predictions. The fundamental idea behind the approach presented was that the submodels would find a smaller (latent) feature space that encapsulates the information that is relevant to the problem at hand. From there, the final model could use the wisdom from all the submodels to make its final prediction. In making submissions, it was found that the model could not adequately account for configuration inertia (the tendency for configurations not to change). As such, a novel approach called Minimum Configuration Support was introduced to explicitly account for configuration inertia.

4. Do you have any charts, graphs, or visualizations of the process?

Refer to Attached SciTech Manuscript.

5. Copy and paste the 3 most impactful parts of your code and explain what each does and how it helped your model.

i) Minimum Configuration Support

```
## Miniumum Support and Minimum Config Support ##
y_pred = output_node
Num_Class = y_pred.shape[1]
configs = features[:,-Num_Class:]

y_pred = y_pred*(1-config_support)

# Min Config Support
y_pred = y_pred + config_support*configs

# Min Support
y_pred = y_pred*(1-min_support*y_pred.shape[1])
y_pred = y_pred + min_support

# Return Predictions
return y_pred
```

Minimum Configuration Support is our novel idea of specifically embedding configuration inertia into our final model design. The basic idea is that a parameter (config_support) above should be the minimum support provided to the current configuration for a given airport and lookahead combination. This parameter is learned in the validation procedure. We found that minimum configuration support proved to be one of the most important aspects of our final submission. More is discussed about this in the attached manuscript.

ii) Submodel Usage

Our overall approach to solving the problem involved the concept of ensembling. We found that by having submodels perform subtasks (such as predicting if a configuration will change) and using the predictions of those subtasks as inputs to the final model, we can do much better than just providing the raw features to the final model. The above submodel training is fitting Xgboost models to predict a binary prediction of whether or not a specific runway will be used.

iii) Autokeras

```
## Autokeras Training Routine ##
input_node = ak.Input()
output_node = ak.Normalization()(input_node)
output_node = ak.DenseBlock()(output_node)
output_node = final_layer()([output_node, input_node])
clf = ak.AutoModel(
    project_name=f"automodel/{airport}/{lookahead}", inputs=input_node,
    outputs=output_node, loss=binary_loss, overwrite=True, max_trials=50
)

# Train Model and Fit Best Hyperparemters
clf.fit(X_train, y_train, validation_data=(X_val, y_val), epochs=100, verbose=0, callbacks=cbs)
```

One of the most impactful parts of our code was the use of Autokeras to automatically train the best neural network architectures. In addition to finding parameters such as width and depth of NN, it would also tune parameters such as the minimum configuration support. This allowed us to focus on things at a higher level in the training process.

6. Please provide the machine specs and time you used to run your model.

- **CPU**: Intel i7-10750H CPU @2.60GHZ (6 cores)
- **GPU**: NVIDIA GeForce GTX 1650 Ti
- **Memory**: 32 GB
- **OS**: Windows 10
- **Train duration**:
  - ☐ Data Processing - 1 hr
  - ☐ Submodel Training - 5 hours
  - ☐ Final Model Training - 12 hours
- **Inference duration**: 2-3 min per prediction

7. Anything we should watch out for or be aware of in using your model (e.g. code quirks, memory requirements, numerical stability issues, etc.)?

No specific things come to mind as being issues. Just ensure that all requirements are met (see requirements.txt), such as installing AutoKeras. The important notes are made in the README file.

8. Did you use any tools for data preparation or exploratory data analysis that aren't listed in your code submission?

No major tools were used other than general data analysis with python. Many times submissions and losses were explored using Juypter notebooks and manual filtering with pandas. Everything that was relevant to training and inference is in the included submission.

9. How did you evaluate the performance of the model other than the provided metric, if at all?

Overall, we primarily used the metric given to us as the overall guide for our model designs. However, in an effort to carry out our approaches, some other metrics were used along the way. As an example, our submodels were trained to perform sub-tasks that relate to the final task of

predicting airport configurations, such as predicting whether an airport configuration will change in a given time period. Because of this, we used metrics such as accuracy and AUC values to train the submodels and hope that good performance with those metrics would lead to good performance on the final metric.

10. What are some other things you tried that didn't necessarily make it into the final workflow (quick overview)?

Before moving forward with our ensembling approach, we first wanted to consider a hidden Markov model type of approach to solving this problem. Specifically, we wanted to design models that would predict the probability of whether a configuration will change or remain stationary. From there we could use that as a prior to predicting the probability of it did change, and what configuration would it change to. While we still believe this approach has merits, it ultimately did not catch traction as it is hard to rigorously consider all possible options as the scenarios explode exponentially.

In regards to the model that we did make, almost everything we wanted to make was included. One thing we left out were the scheduled and estimated arrival and departure rates, but that was left out mostly due to runtime constraints. We also did not have time to explore more interesting submodel designs, such as predicting whether an airport is IFR or VFR.

11. If you were to continue working on this problem for the next year, what methods or techniques might you try in order to build on your work so far? Are there other fields or features you felt would have been very helpful to have?

Refer to the future work section of the Attached SciTech Manuscript.