Group 7 Names: Ishan Dutta, Harvir Ghuman

Improved Model and Trade-offs

Github Link:
https://github.com/Hghn02/Inverse-Design-of-Optimal-Airfoil-Geometry/blob/main/model_build.ipynb

Project Description/Contributions:

We chose project 6 where the objective is to inversely design an airfoil with the ultimate objective of optimizing the lift to drag ratio using the chosen boundary conditions such as Mach Number, Reynold's number, and Angle of Attack. Ishan created the document.

Improved Implementation:

One choice we made to improve the model was to use a different Loss function. Before we were using Mean Squared Error Loss and the training and testing losses were a bit higher and capping out on the order 10^-2. We were also seeing that some of the reconstructed airfoils had weird geometries compared to the ground truth airfoils. We think what was happening was that MSE was penalizing outliers too severely since it was squaring the differences. We opted to go for the Huber Loss function instead which provides a good balance between Mean Squared Error and Mean Absolute Error. It penalized large errors less severely compared to MSE by transitioning to linearly penalizing for large errors but still penalizing small errors enough.

Increasing the number of components in our PCA pipeline also helped to improve the model performance and accuracy. We noticed that when we increased the number of principal components for reconstruction, more geometry was preserved and there were less anomalies in the reconstructed geometry.

We also implemented some k-fold cross validation that help reduce our loss and ensure the model was seeing variety of data instead of the previous 80 – 20 train test split.

For our proposed final model using an Autoencoder into a deep neural network we can continue to use Pytorch and parts of scikit learn libraries

Preliminary Results and Plan:

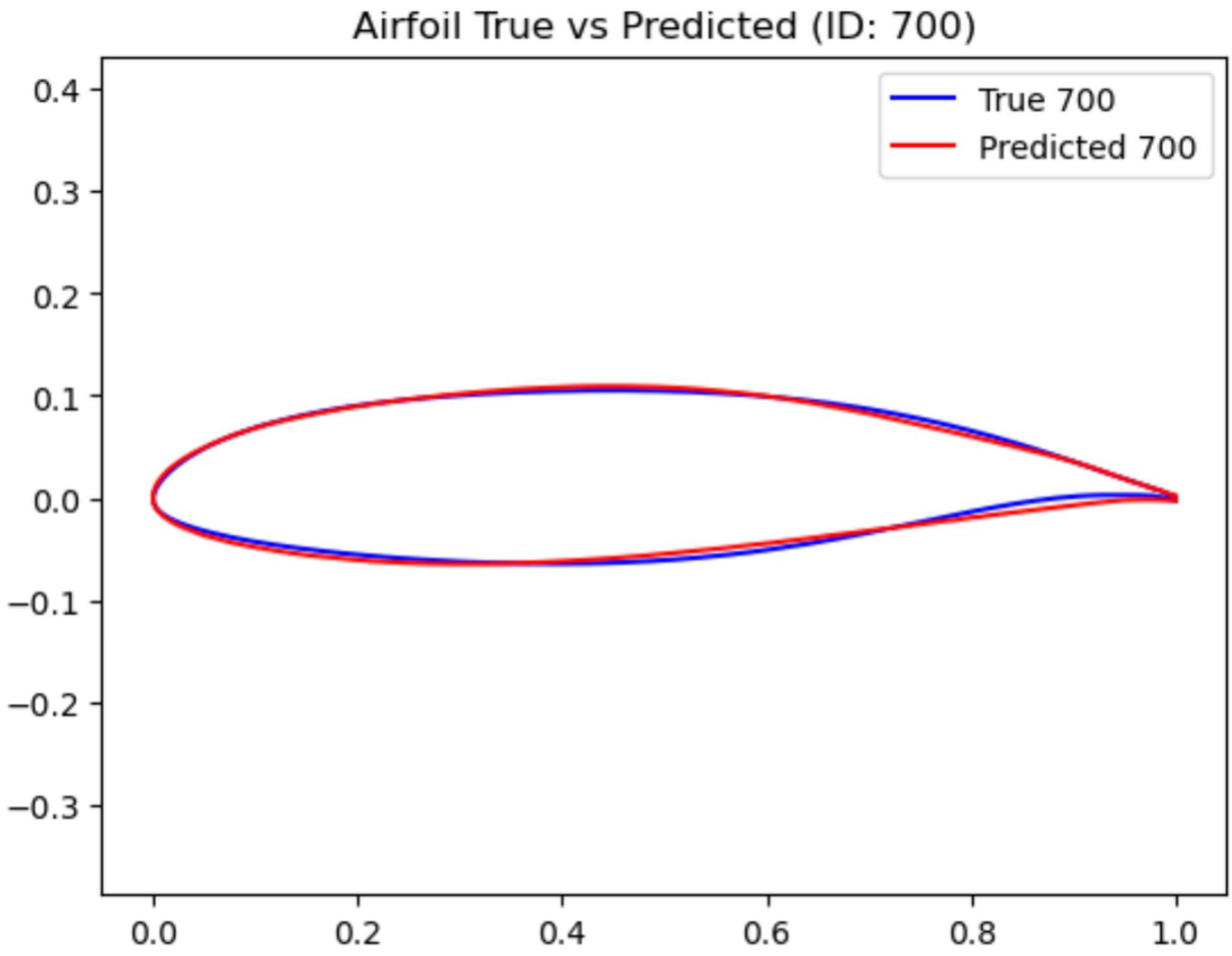
So far we have tried PCA dimension reduction and using the latent space as the output of a deep neural network and input as Mach number, Reynolds number, and lift coefficient. Our cross validation strategy was k fold with 6 splits. The deep neural network architecture consists of 2 hidden linear layers with ReLU activation that is sufficient enough to capture some nonlinearity in geometry.

Compared to our baseline model, our training and test loss has improved significantly along with our reconstructed results. Previously with the baseline model with no cross validation, MSE loss, and smaller latent space we achieved loss on order of 10^-3. With the new and improved

model we are down to test losses on the order of 10^-6. However, there are still some strange geometry airfoils that do not match very closely with their predicted counterparts. It is only a small handful but we think the nonlinearity of these airfoils are not able to be captured by PCA. For our final steps we will replace PCA with an autoencoder so our dimensionality reduction can learn all the complex nonlinearities of the airfoils. The latent space fed into a deep neural network architecture will likely remain the same as before. Finally, we will craft some final visualizations of our reconstructed airfoils as well as our losses per iteration.

We anticipate struggling with building the Autoencoder architecture in terms of code as we have not had much experience with this kind of model before. However, we will learn how to build one by example and tutorials as well as by doing. After this is complete, we will be mostly done except for a few visualizations for the results.

An example of a good prediction below.



Group 7 Names: Ishan Dutta, Harvir Ghuman

An example of a bad prediction below.