Bibliography

[1] Aldair E Gongora, Kelsey L Snapp, Richard Pang, Thomas M Tiano, Kristofer G Reyes, Emily Whiting, Timothy J Lawton, Elise F Morgan, and Keith A Brown. Designing lattices for impact protection using transfer learning. matter 5, 9 (2022), 2829–2846, 2022.

Predicting Maximum acceleration of impact Testing for lattices using Transfer learning

Tejas Badakere Gopalakrishna

March 2024

0.1 Abstract

The data [1] consists of experiments conducted for Quasi-static testing and Impact testing. We aim to find the maximum acceleration incurred by the lattices during the impact testing. However, these experiments are expensive to conduct as they are destructive. Hence, the goal is to relate the results from the Quasi-static testing to the Impact testing. The caveat here is that not all Quasi-static testing is mapped to Impact testing. Some Quasi-static testing has multiple maps for the same Impact testing. Hence, we have to incorporate this into our model. We are given a total of 817 Quasi-static tests and 153 Impact tests. However, not all Impact tests have a mapping to Quasi-static, and some of the Impact tests are mapped to the same Quasi-static testing. Therefore, in the end, we have 146 data points, and we would be training with the subset of these data points, with the remaining data points being used as the test case. In this work, the primary aim is to reduce the dimension of the Force-Displacement curve further from 6, as reported in this paper [1], to 2 using an RNN-based model reduction, and predict the maximum acceleration using these two features.

0.2 Methodology

The methodology used will be as follows.

- **Data preprocessing:** Data given from Quasi-static testing are not continuous and hence suitable ID's from the quasi-static testing must be matched to Impact testing. Also, we will have to convert the type of lattices in impact testing to numerics so that we can later work with that as an additional feature. Data from the impact testing corresponding to the none quasi-static testing are removed. Data from Impact testing are arranged in the ascending order of Quasi-static testing Id.
- **Dimensionality reduction:** The force-displacement curve from Quasi-Static testing has about 148 data points. Using these 148 features with less data in hand can cause overfitting issues. Hence different dimensionality reduction techniques such as PCA and GRU autoencoder are explored.
- **Choosing the final features:** The data given has multiple Quasi-static testing mapped to the same Impact testing. This can be difficult for a model to find a unique map between input and Output. Hence, additional features can be added to get a unique map between input and output along with the reduced features from the dimensionality reduction. These additional features include lattice type, xvert, xstretch and also time in case of RNN based autoencoder.
- **Model selection:** This corresponding problem has less training data and hence careful selection of the model has to be done. A few of the candidates for the model chosen here are Gaussian regression, Linear regression, Ridge regression, etc. All the models are carefully tested by tuning the hyperparameter based on k-fold cross-validation.
- **Training:** Training is carried out to predict maximum acceleration for two impact tests. Features obtained from Dimensionality reduction are used as the input data. The output corresponds to the maximum acceleration during the impact testing.
- **Testing:** Testing is carried out and evaluated based on various metrics such as r2 score and mean squared error.

0.3 Dimensionality Reduction

0.3.1 Principal Component Analysis

Principal Component Analysis(PCA) is carried over the Quasi-Static data sets. Data points from the F-D curve are drawn and the values corresponding to the forcing are used as the features of the PCA algorithm. Once the PCA is performed the new coordinates z is treated as the new set of features. Not all values of z are necessary as the data usually lie in the low-dimensional subspace. This is calculated based on the total variance explained by each dimension. Figure 1 shows the plot of the total explained variance for each component. The first 6 features explain approximately 99% of the total variance and hence the initial number of features chosen is 6. Later sequential feature selection is carried out a posteriori to

define the final model for the prediction. To check the efficacy of the learned low-dimensional features, the data is projected back into the high-dimensional space. Figure 2 shows the comparison

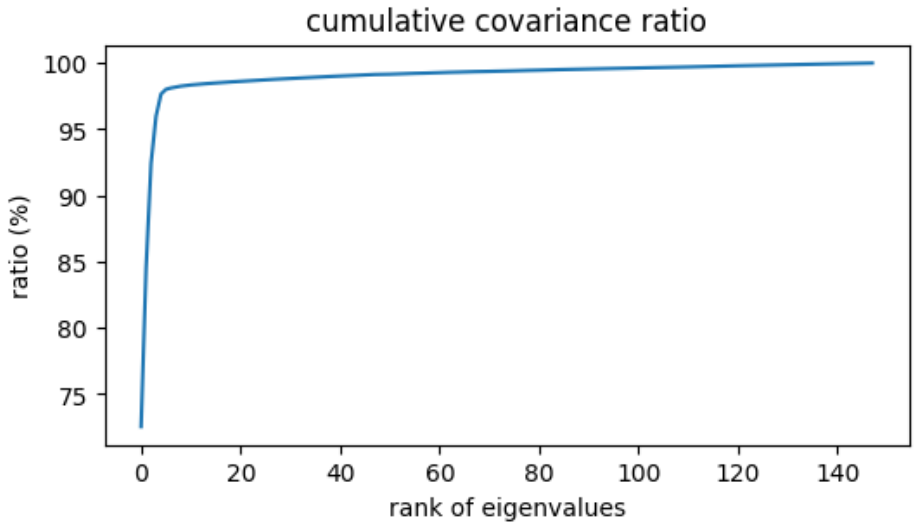


Figure 1: Cumulative covariance for the Force data.

between the original curve and the reconstructed curve of the F-D diagram. The reconstructed graph matches the original graph indicating that the low dimensional representation of the data was accurate. The reconstruction msq loss was also 1e-09 which is very low.

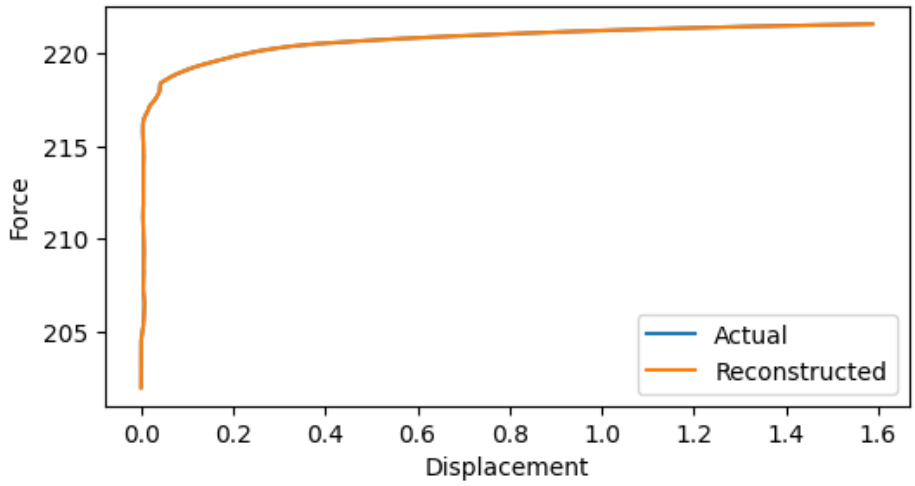


Figure 2: Comparison between PCA reconstruction and actual F-D curve.

0.3.2 GRU based autoencoder

We have seen that PCA reconstructs with great accuracy. However, we will try GRU based autoencoder and evaluate its efficacy. The input data to the GRU-based autoencoder is different from that of PCA. Here we input the time series of the force and displacement curve. Hence input will be of 3 dimensional viz time, force and displacement where force displacement and time are treated as input features which are fed to the autoencoder as time-series. The output of the GRU encoder is the low-dimensional representation of the high-dimensional data. The GRU decoder is constructed by inputting the encoded feature at each timestep along with the physical time as the input feature so that the problem of memory loss due to long sequences can be avoided. Figure 3 shows the loss curve for the training and testing loss of the GRU autoencoder for latent size=2. The error has converged to the order 1e-04. Figure 4 shows

the comparison between the actual and reconstruction from the GRU autoencoder. The reconstruction is not as accurate as PCA though the error is still the order 1e-4. This discrepancy might be attributed to the fact that the data might indeed lie in the linear subspace rather than the nonlinear manifold.

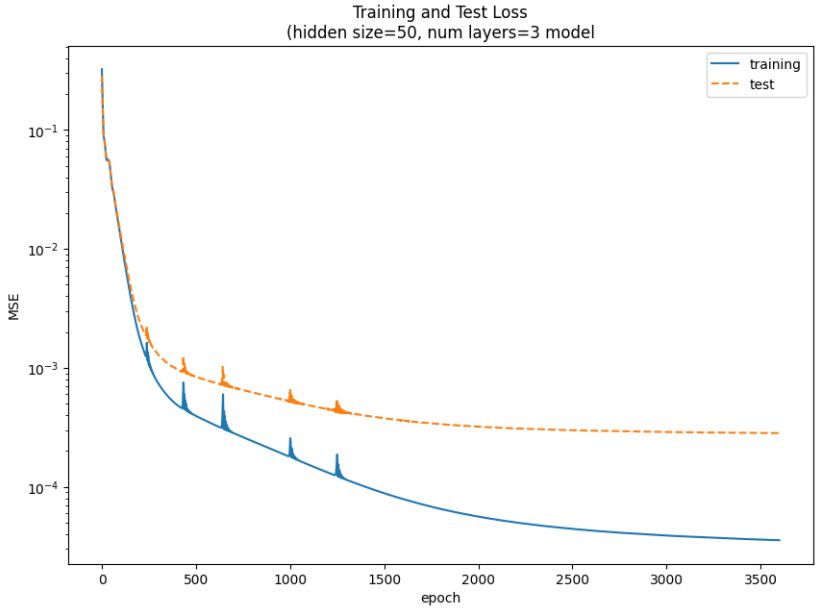


Figure 3: Plot of train and test loss for GRU autoencoder.

Table 1 shows 2 other hidden states and their corresponding r2 score and mean squared error. one can see that for latent size=2 yields the lowest reconstruction error and performed equally gooD as PCA with 6 features.

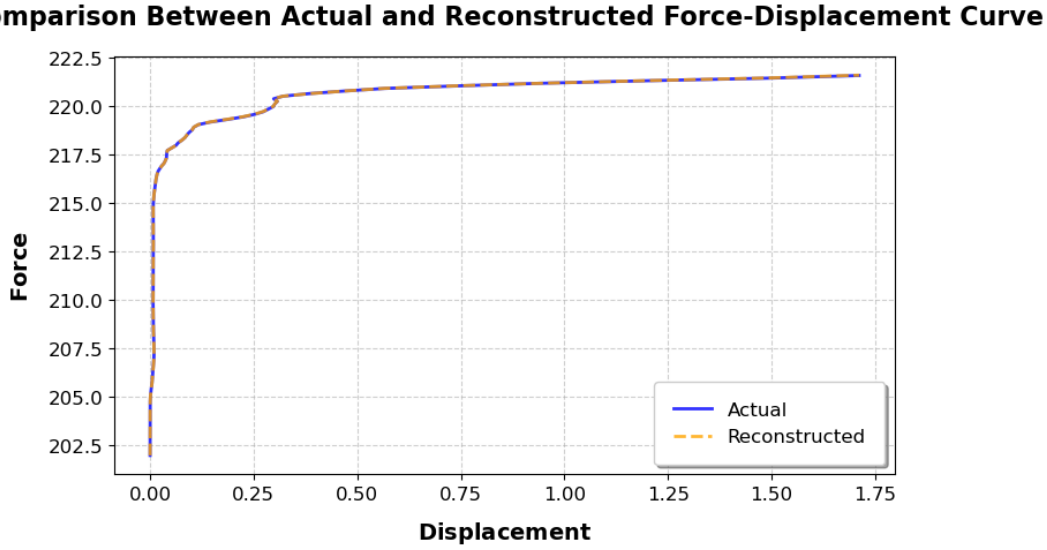


Figure 4: Comparison between GRU autoencoder reconstruction and actual F-D curve.

0.4 Sequential Feature selection and model selection

This section is only necessary for the PCA based models to reduce the dimension further. Some of the impact tests map to the same Quasi-static tests. Hence additional features such as lattice features and the type of lattice are added to this low dimensional learned feature. These additional features along

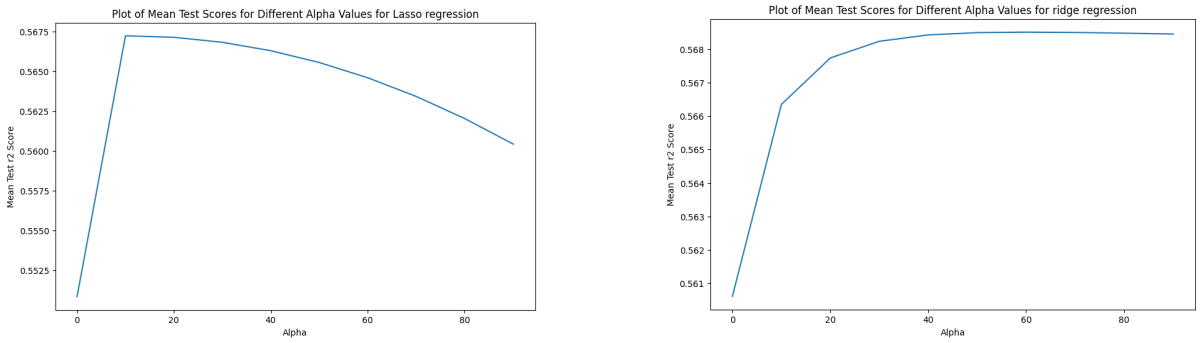
Table 1: Comparison between different latent sizes.

Encoded feature	r2 score	Mean Squared Error
6	0.9996	0.000671
3	0.9997	0.000671
2	0.9999	0.0007484

with the low dimensional representation are used to build the final model. We have already constructed low-dimensional construction using PCA and GRU. Therefore we now have 10 features for PCA-based models and 6 features for GRU based models to work on. Now, in this section, we'll choose the final model and number of features. The candidates for different models are Gaussian, Linear, Ridge, and Lasso regression. Implementing a DNN to this type of problem is of the least use since we might overfit the data as we have fewer examples in hand. This section will be divided as follows, first, we will work with PCA and the four models, next, we'll work with the GRU encoder and the four models.

0.4.1 PCA based models

This section is only necessary for the PCA-based models to reduce the dimension further. Some of the Impact tests map to the same Quasi-static tests. Hence, additional features such as lattice features and the type of lattice are added to this low-dimensional learned feature. These additional features, along with the low-dimensional representation, are used to build the final model. We have already constructed low-dimensional representations using PCA and GRU. Therefore, we now have 10 features for PCA-based models and 6 features for GRU-based models to work on. Now, in this section, we will choose the final model and number of features. The candidates for different models are Gaussian, Linear, Ridge, and Lasso regression. Implementing a DNN for this type of problem is of the least use since we might overfit the data as we have fewer examples at hand. This section will be divided as follows: first, we will work with PCA and the four models; next, we will work with the GRU encoder and the four models.



(a) (b)

Figure 5: Plot of alpha vs r2 score for (a) Lasso (b) Ridge

Table 2: Comparison between different PCA models.

Model	Mean RMSE	Number of feature
Linear	64.21	4,7,8,9
Gaussian	64.21	4,7,8,9
Ridge	62.6138	6,7,9
Lasso	62.8867	0,7,9

0.4.2 GRU based models

A similar procedure to that of PCA-based models is conducted. As found earlier, the reconstruction from the GRU-based autoencoder was not as efficient as the PCA-based autoencoder. However, we will

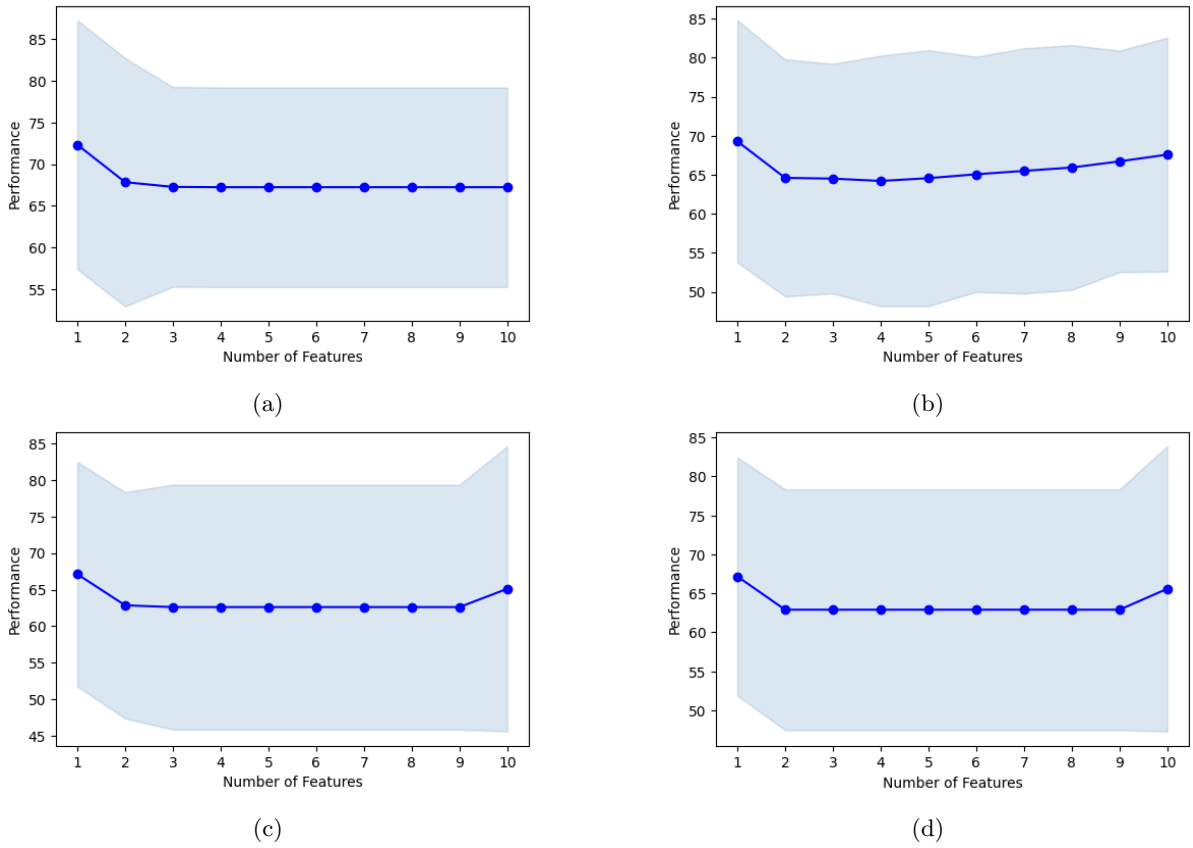


Figure 6: Box plot for various PCA based models. (a) Gaussian (b) Linear (c) Ridge (d) Lasso

still check the predictive capacity of the features. Similar to PCA-based models, the encoded features are combined with the lattice properties and lattice type to predict the maximum acceleration. This time, we will only have 5 features to begin with, as we had only chosen 2 encoded features initially as opposed to 6 in PCA. Again, four different models are used to predict maximum acceleration. Sequential Feature Selection is carried out for these four different models. Figure 8 shows the Grid Search for optimal alpha for both Ridge and Lasso Regression. Figure 9 shows the box plot for RMSE for various subsets of the features for various models. Again, Ridge regression performed the best among all, with a mean RMSE of 49.6138 (g), which is much better than the PCA-based linear regression. The rest of the models were also comparable to the PCA-based models. Table 3 shows the mean RMSE and optimal features for various models.

Table 3: Comparison between different GRU models.

Model	Mean RMSE	Number of feature
Linear	64.69	0,3,5
Gaussian	64.52	2,3,5
Ridge	63.728	2,3,5
Lasso	65.40	3,4,5

0.5 Training and testing

Based on the preceding sections, the conclusion suggests that the PCA-based ridge regression model performed optimally with only three features. We will utilize this model to predict two different accelerations from distinct impact tests. The predictions are depicted against actual values in Figure 10a for acceleration 1 and Figure **??** for acceleration 2. Root Mean Squared Error(RMSE) was calculated using leave-one-out cross-validation (CV). Despite achieving a root mean square error (RMSE) of 49.6138 (g)

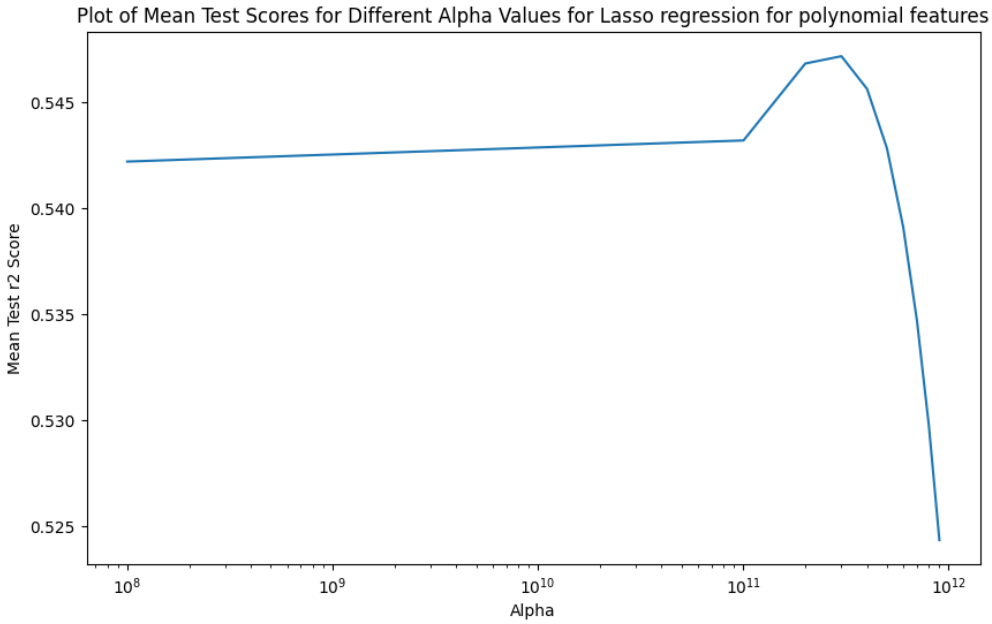
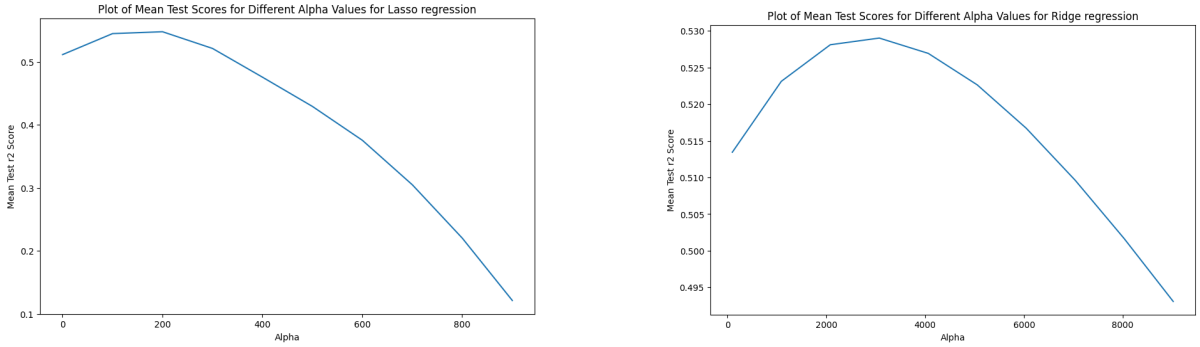


Figure 7: Plot of alpha vs r2 score for Ridge regression using polynomial features.



(a) (b)

Figure 8: Plot of alpha vs r2 score for (a) Lasso (b) Ridge

for acceleration 1, the accuracy of predicting acceleration from the same model appears less reliable for acceleration 2. While the model captures the general trend, its coefficient of determination (r2 score) remains notably low

0.6 Conclusion

In this project, we utilized transfer learning from Quasi-Static data to predict impact acceleration. Principal Component Analysis (PCA) was specifically applied to the Forcing dataset to reduce features. We experimented with a GRU-based autoencoder to learn the latent space but found PCA to be more effective. Additionally, we incorporated extra features such as Unit type, xvert, xbend, and xstrech, which proved crucial as multiple Quasi-Static forces were mapped to the same impact acceleration. Sequential Feature Selection further refined our input features.

Various models, including Linear Regression, Quasi-Static Regression, Ridge, Lasso, and polynomial features, were evaluated. Surprisingly, Gaussian Regression with an RBF kernel did not yield satisfactory results, but introducing additional white noise enhanced its predictive ability. Although polynomial features were explored, high regularization posed memory allocation challenges. Ultimately, Ridge Regression with an alpha value of 40 emerged as the optimal model for predicting acceleration 1. However, its performance for acceleration 2 fell short of expectations, indicating a limitation.

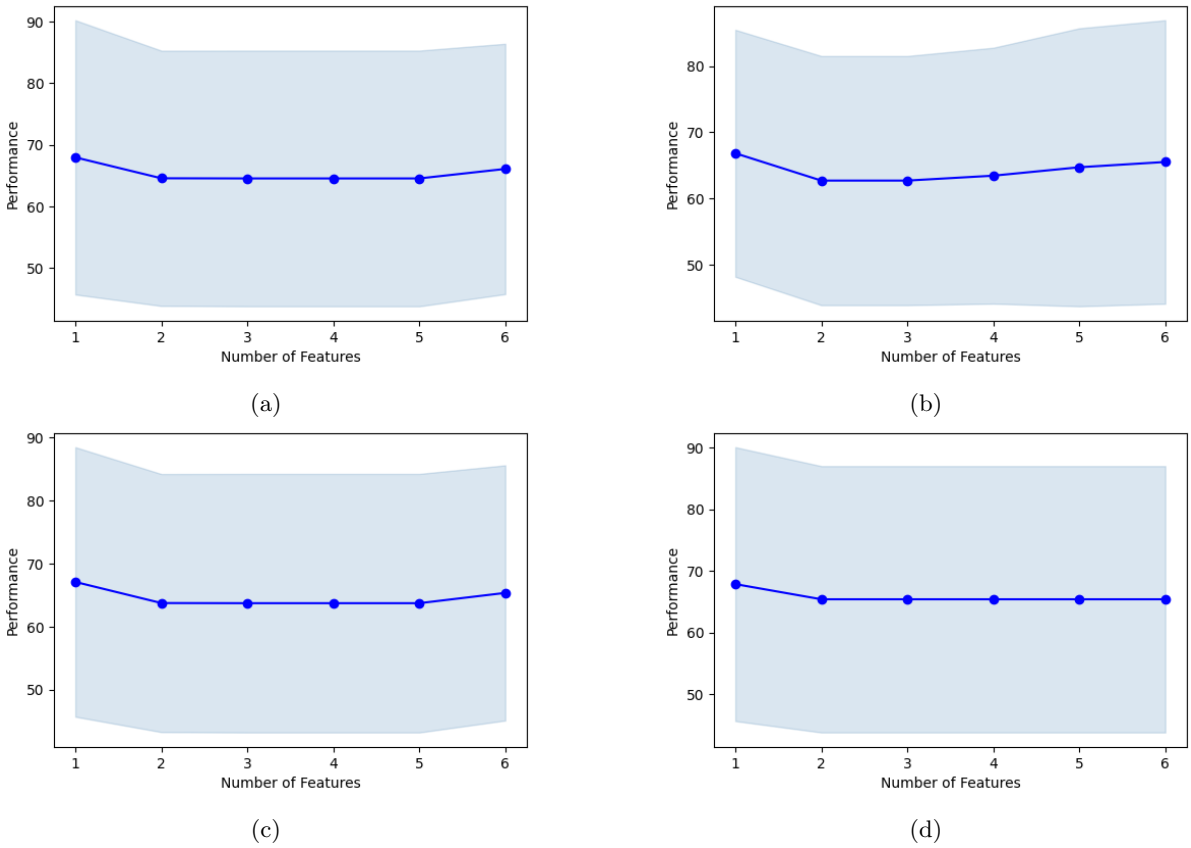
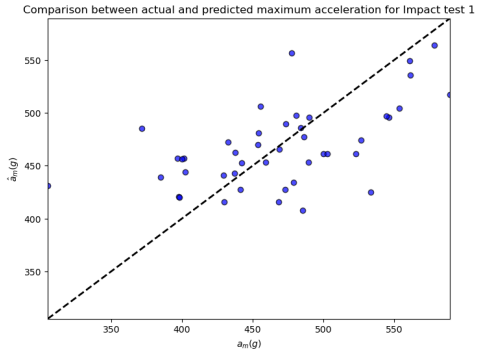
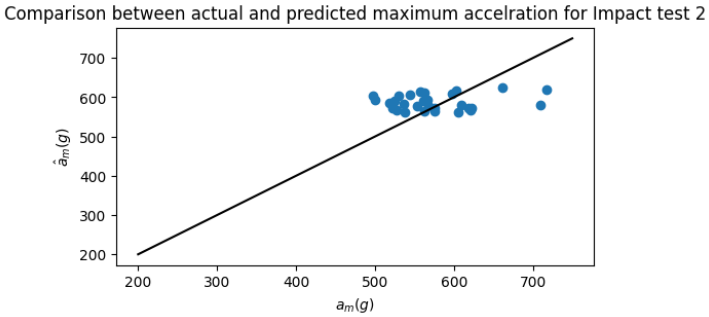


Figure 9: Box plot for various GRU based models. (a) Gaussian (b) Linear (c) Ridge (d) Lasso

To enhance the model's predictive capacity, collecting more data is essential. Furthermore, considering Deep Neural Networks (DNN) could be beneficial if additional data becomes available.



(a)



(b)

Figure 10: Impact predictions for two acceleration

Bibliography

[1] Aldair E Gongora, Kelsey L Snapp, Richard Pang, Thomas M Tiano, Kristofer G Reyes, Emily Whiting, Timothy J Lawton, Elise F Morgan, and Keith A Brown. Designing lattices for impact protection using transfer learning. matter 5, 9 (2022), 2829–2846, 2022.