Characterizing the current state of a propulsion engine: A comparison of machine learning frameworks

CASI AERO 2019 – 22nd CASI Aerospace Propulsion Symposium

Srishti Sehgal [1,3]
srishti.sehgal@nrc-cnrc.gc.ca
OR
srishti.sehgal@mail.utoronto.ca
(613) 991 9646

Catherine Cheung [2]
catherine.cheung@nrc-cnrc.gc.ca

Julio J Valdes [2]
julio.valdes@nrc-cnrc.gc.ca

[1] *Full-time undergraduate at University of Toronto – National Research Council Canada Co-op Student*
[2] *National Research Council Canada*
[3] *Corresponding author*

Extended Abstract

Introduction

Given the ability to install sensors on equipment, like a propulsion diesel engine (PDE), equipment health can be monitored and studied to improve its safety, economy and dependability. Typically, a machine learning model can examine relationships between equipment steady state physical characteristics and other indicators of equipment health to systematically generate failure predictions when these parameters deviate from pre-defined settings. The challenge is that during training, many models assume an equal proportion of healthy and failed state examples. However, for many complex systems, the distribution of failed to healthy steady state examples is skewed as failure events are not common. This poses a difficulty for these algorithms, as they will be biased towards the majority healthy class.

To combat the imbalanced dataset, NRC Aerospace studied the impact of *data reduction* on a classification task for sample PDE data [1]. Although some classification and anomaly detection methods proved fruitful, there is interest to extend the study into the effects of *dimension reduction* on classification performance. In this process, a dataset with a large feature set is transformed into a dataset with a smaller feature set while ensuring to convey the same information concisely.

NRC Aerospace is investigating the use of a multi-task framework (MTF) to extract features that may share a complex non-linear relationship. If learning tasks are related, training them jointly can lead to a greater performance improvement than if they were trained separately. Multi-task learning aims to improve the generalization performance of several related tasks and share knowledge to develop a common feature set among tasks. Similar to human learning, the knowledge contained in one task can be leveraged by other tasks [2, 3].

This paper describes the structure of two frameworks. The first framework uses a Neural Network for direct classification from an input dataset. The second framework is the MTF where dimension reduction will be used prior to classification. The objective of this effort is to see if the MTF can improve training time, improve classification accuracy and set a direction towards the future development of efficient frameworks. To quantify the success of these frameworks, classification results are evaluated with binary classification performance metrics such as True Positive Rate and Positive Predictive Value.

Preliminary Results

Only 7% of training (171836 points) and 7% of testing (58323 points) are failed states. In the first framework, a Neural Network is used to classify each sample in the dataset as a healthy operating condition or as a deteriorated one.

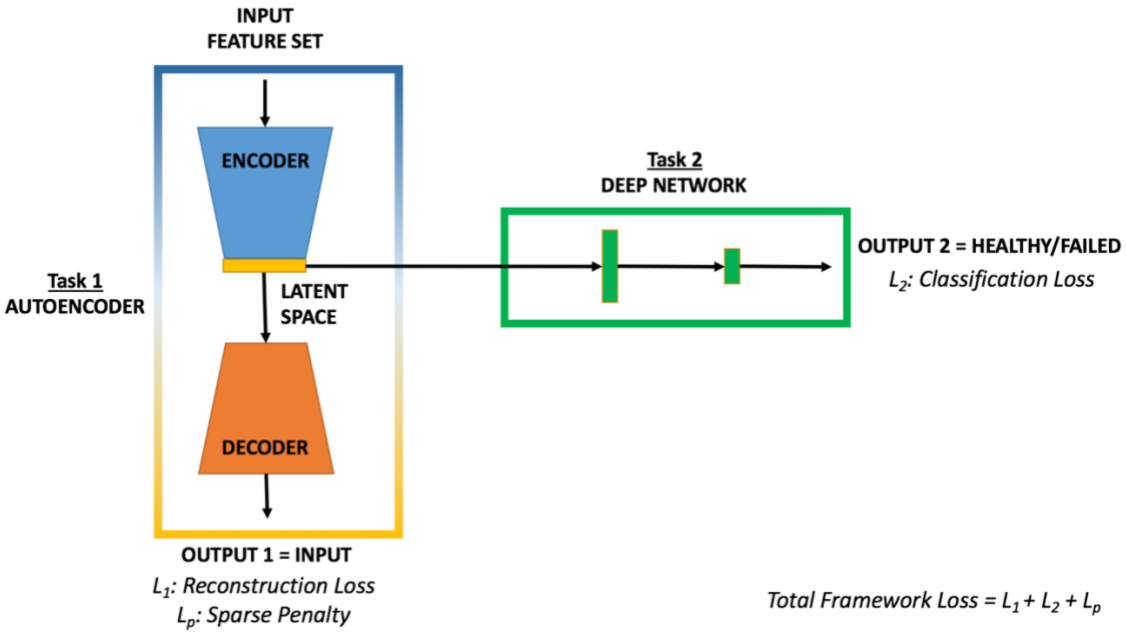


Figure 1 MTF Structure showing two learning tasks and total framework optimization function. Note that the sparse penalty that only affects the weights of the Autoencoder.

As shown in Figure 1, there are two tasks in the MTF. Using an Autoencoder, the first task is to reconstruct the input dataset. At the bottleneck layer, the network is compelled to learn a compressed representation of the input. If some of the input features are related, then this algorithm will be able to discover some of those non-linear relationships. At the same time, a second learning task performs a classification of the samples as either healthy or failed states. The primary difference between the MTF and the other framework is that the second learning task will attempt classification on the latent space produced by the first task.

Listed in Equations 1-4 and in Table 1, receiver operating characteristic metrics are employed to evaluate the performance of this classification methodology. A perfect classifier will only have elements along its main diagonal. In this study, a true positive refers to a failed steady state predicted as a failed state, while a true negative refers to a healthy steady state correctly predicted as a healthy state. False positives, or false alarms, should be minimized to reduce complacency in operator response. False negatives also should be minimized, if not avoided, to ensure that a system failure is not overlooked. Since the data is imbalanced, metrics such as PPV and FNR should be considered. An ideal classifier is one that has a PPV equal to 1 and the FNR equal to 0.

Table 1 Confusion matrix as a basis for Receiver Operating Characteristic Metrics

		Ground Truth/Observed	
		Failed	Healthy
Model Prediction	Failed	True Positive (TP)	False Positive (FP)
	Healthy	False Negative (FN)	True Negative (TN)

$$TPR = \frac{TP}{(TP + FN)}$$

Equation 1 True Positive Rate (TPR)

$$FPR = \frac{FP}{(TN + FP)}$$

Equation 2 False Positive Rate (FPR)

$$FNR = \frac{FN}{(TP + FN)}$$

Equation 3 False Negative Rate (FNR)

$$PPV = \frac{TP}{(TP + FP)}$$

Equation 4 Positive Predictive Value (PPV)

Framework 1

As seen in Figures 2 and 3, the Neural Network classifier can differentiate failed steady states from healthy steady states very well in both training and testing sets. PPV is also very close to 1 and FNR is very close to 0. This suggests that the model is not being biased towards the high number of healthy steady state samples.

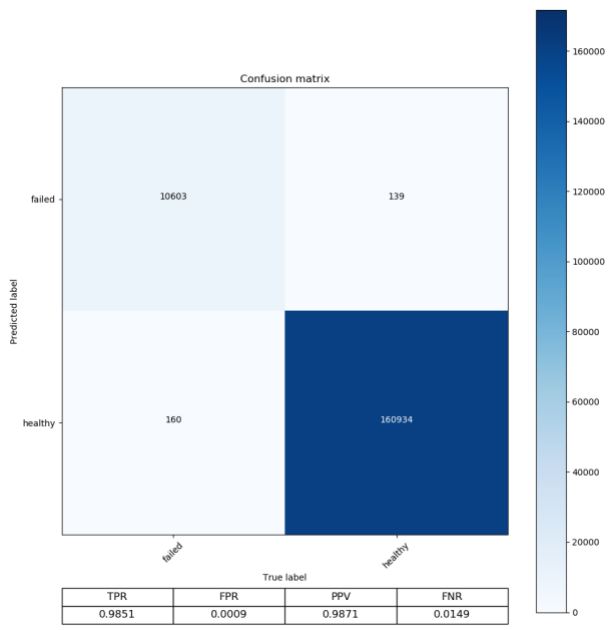


Figure 2 Confusion matrix for training set using Neural Network

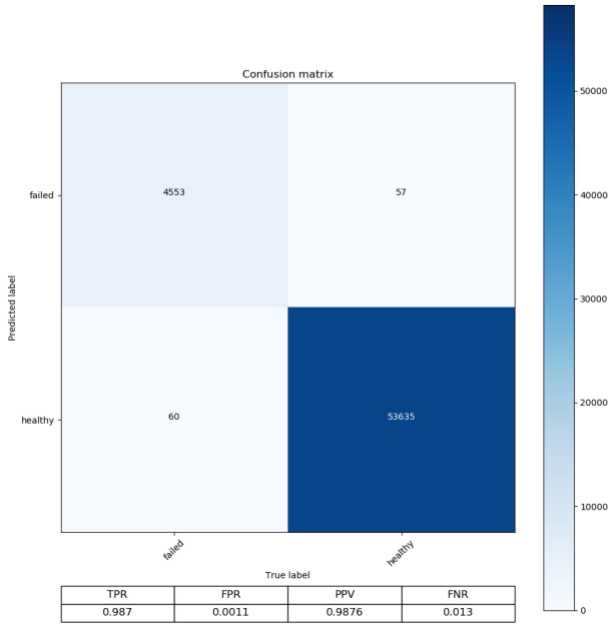


Figure 3 Confusion matrix for testing set using Neural Network

Framework 2

As shown in Figure 4 and 5, the model was able to recreate the dataset very well in the first task. The final reconstruction loss was only 0.87% of the initial loss. Since the model did not learn the trivial identity function to reconstruct the input dataset, the latent representation retained information from the original input dataset with only 30 features instead of the original 51.

The second learning task used this latent space to train a Neural Network architecture similar to Framework 1. Its classification results are shown in Figures 6 and 7. Other hyperparameters such as choice of optimizer or choice of learning rate were kept constant to ensure that the only difference between the two frameworks would be architecture and the use of a latent dataspace. Although the TPR is slightly less than that of the Neural Network classifier from the first framework, PPV is slightly higher and the FPR is slightly lower in the testing set. The classification results of the MTF are similar to the results of Neural Network in Framework 1. A remarkable observation, however, is that the efficiency of the MTF was significantly better. For the same datasets, the MTF only required 25 epochs to converge while the other framework required 75 epochs.

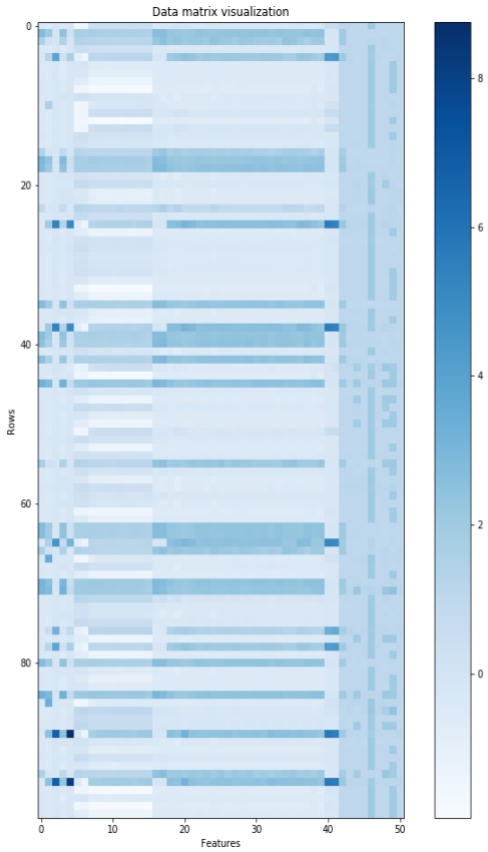


Figure 4 Visualization of last 100 rows of original data matrix

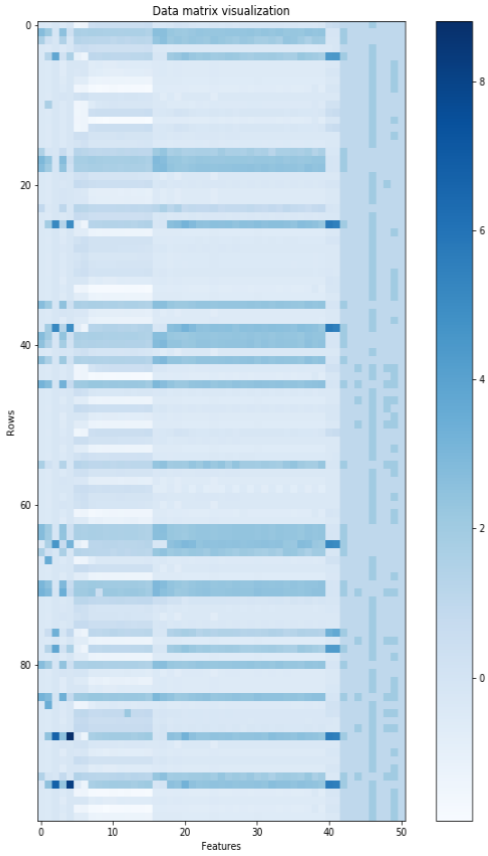
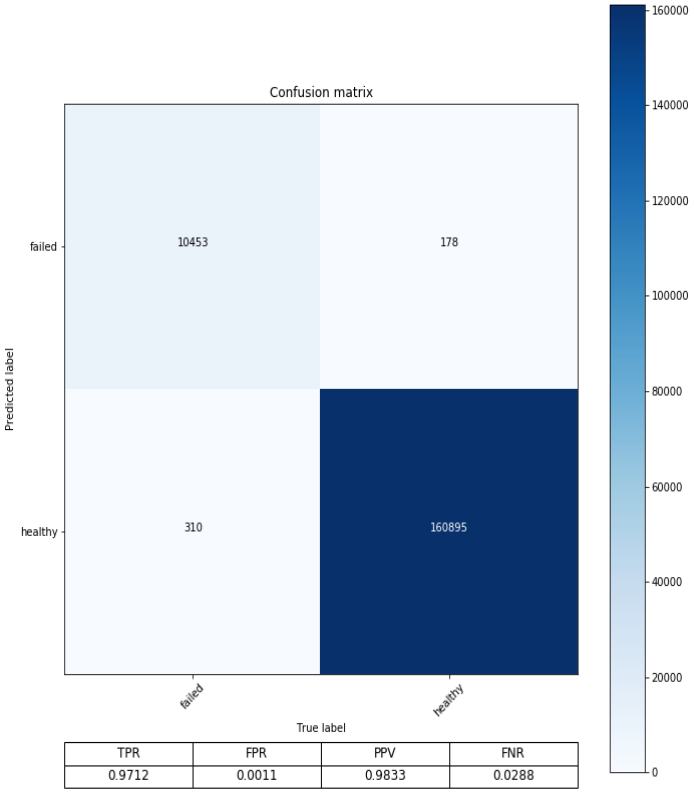
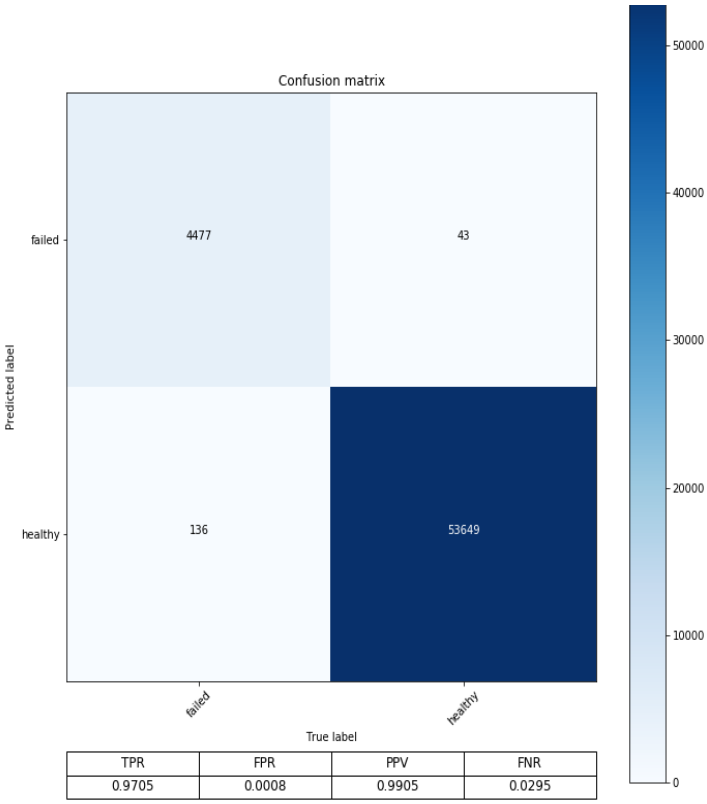


Figure 5 Visualization of last 100 rows of model generated data matrix



TPR	FPR	PPV	FNR
0.9712	0.0011	0.9833	0.0288

Figure 6 Confusion matrix for training set using MTF



TPR	FPR	PPV	FNR
0.9705	0.0008	0.9905	0.0295

Figure 7 Confusion matrix for testing set using MTF

Conclusion

The main objective is to correctly identify healthy and failed states of a propulsion diesel engine. Results of an MTF that generated a lower-dimensional latent data space for classification were compared against a framework that did not use dimension reduction for classification. The study shows considerable promise for using an MTF to generate a latent data space for a classification problem with an imbalanced dataset. Though the prediction rates are similar to the previous framework, the time elapsed decreased by threefold. This is a very significant outcome of the study because as datasets get larger, the time required to train a model will increase too. Furthermore, future real-time tools that use faster-to-train frameworks, like the MTF, can be developed to analyze sensor information and to provide more up-to-date and adaptive predictions. Thus, choosing a framework that will potentially decrease the amount of training time is important. Ultimately, being able to detect failures near-real-time during critical operation is significantly useful to: operators who will plan their next set of actions to avoid failure, maintainers who will understand where their efforts should be concentrated and engineers who can initiate improvements in new engine designs based on areas of recurring anomalies in historical data.

References

[1] Cheung, C., Valdes, J., Chavez, R., and Sehgal, S. (2019). Failure modelling of a propulsion subsystem: unsupervised and semi-supervised approaches to anomaly detection. *International Journal of Pattern Recognition and Artificial Intelligence.*

[2] Zhang, Y., Wei, Y., and Yang, Q. (2018). Learning to multitask. *Conference on Neural Information Processing Systems*.

[3] Zhang, Y., and Yang, Q. (2018). A survey on multi-task learning.