SAD: Self-supervised Avionic Diagnostics

Attiano Purpura-Pontoniere*[a], Maksim Bobrov[b], Tarun Bhattacharya[a]
[a]Lockheed Martin Space, 1111 Lockheed Martin Way, Sunnyvale, CA 94089
[b]Lockheed Martin RMS, 300 Canal View Blvd, Rochester, NY 14620
*Attiano.purpura-pontoniere@lmco.com; phone 1 510 935-6602; fax 1 222 555-876; lockheedmartin.com, attiano.com

ABSTRACT

Oftentimes aircrafts will experience flight failures that produce a significant amount of complicated avionic diagnostic data, wherein many reports of faults will contain false positives across various subsystems, and true faults normally are identified by pouring through an entire flight's diagnostic logs. We investigated if avionic fault detection can be improved, or completely automated, through intelligent application of machine learning models to paired instrumentation and natural language time-series data from helicopters. We focused on using an unsupervised model, specifically an auto-encoder, as our data was unlabeled. We also focused on the natural language portion of the data, and we created a novel transformation from natural language time series data to image data for ease of model integration, taking inspiration from the spectrogram. This allowed us to leverage a linear & convolutional autoencoder for feature extraction, which we compared to a deterministic algorithm like Principal Component Analysis (PCA). We successfully trained convolutional autoencoders to reconstruct our avionic diagnostic images. We attempted to train linear autoencoders with our custom images and a bit transformation of our avionic diagnostic images. The linear model was unable to reconstruct the image and binary version. Finally we explored if the embeddings of the convolutional autoencoder and PCA could be used to automatically label our data by exploring the clusters produced by K-means. Our results were promising, where the PCA was a more accurate fault detector than the convolutional autoencoder, but further investigation is needed.

Keywords: Computer Vision, Classification, Avionic Diagnostics, Fault Detection, Auto-encoders, Unsupervised Learning, Self-Supervised Learning, Natural Language Data, Deep Learning, Convolutional Auto-encoders, Principal Component Analysis

1. INTRODUCTION

Rotorcraft avionic systems produce a considerable amount of instrumentation data as well as natural language tracking of fault detections. It is highly desirable to be able to accurately and quickly identify the root cause of a flight failure, but this can be very difficult given the large number of intricately connected subsystems that fault. Oftentimes each individual report of a subsystem's fault will be a false positive, and true faults normally are identified by pouring through an entire flight's diagnostic logs across all subsystems. To streamline & improve offline fault detection we explored using self-supervised learning as much of the data is unlabeled, specifically we investigated if autoencoders could autonomously extract features representative of a fault through reconstruction of flight logs. We accomplished this through a novel transformation from natural language time series data to image data for ease of model integration, taking inspiration from the spectrogram. The natural language to image transformation took a subset of the vocabulary that corresponds to faults, and enumerated all possible faults within that subset. We then turned that enumeration into a binary vector, where 1 corresponded to the presence of a fault, and 0 corresponded to no fault for a given subsystem at a given timestamp. We then plotted the binary vectors over time creating black and white tiled images, akin to the MNIST dataset. We successfully trained convolutional autoencoders to reconstruct our custom avionic diagnostic images. We also attempted to train linear autoencoders with the binary images, and a bit transformation of the binary images. The bit transformation turned each binary fault vector into an integer by interpreting each binary vector as a binary string, essentially representing a cumulative error code as an integer value. The linear model was unable to reconstruct both the binary image and bit encoded version. Finally we explored if the embeddings of the convolutional autoencoder could be used to automatically label our data using k-means with two means to cluster the data, one mean for faults, and the other for non-faults. We strived to see if naturally a fault is clearly separated from a non-fault in the autoencoder embedding space. We compared the machine learning results to conventional methods, such as PCA and DBSCAN. With DBSCAN the clusters obtained didn't seem to accurately capture the correct separation when visualized in two dimensions. Interestingly when we used PCA to reduce the dimensionality of the data to two dimensions, the resulting clusters

Big Data V: Learning, Analytics, and Applications, edited by Panos P. Markopoulos, Bing Ouyang, Vagelis Papalexakis, Proc. of SPIE Vol. 12522, 125220F
 · 0277-786X · doi: 10.1117/12.2664732

demonstrated more accurate results than the convolutional autoencoder's (CNN AE) clusters. We had roughly 1500 examples, all unlabeled, some of which were easy to determine if there was an actual fault or not through qualitative analysis. The CNN AE reported 500, or roughly 1/3 of those as faults, when in reality as a rule of thumb we could expect about 10% to be faults. The PCA only reported 170 faults which more closely matches what we expect. We also qualitatively evaluated batches of 16 random examples/reconstructions, and determined that PCA was better at resolving fault from non-fault, likely due to the simplicity of the data. We posit that the CNN AE was overparameterized for the problem, and so wasn't able to cleanly learn to distinguish fault from not fault, due to not having enough data, and the data being sparse, high-dimensional with little varying content (black and white 32x3000 images). We concluded by identifying other machine learning methods worth comparing our results to, such as obtaining feature vectors of the flight logs from a fine-tuned transformer based natural language processing model, as that would be able to view global context in a similar fashion to our CNN AE implementation.

2. RELATED WORK

The problem of avionic diagnostic fault detection can be seen as an analog to the problem of anomaly detection. There exist deterministic methods of evaluating flight data for anomalies like exceedance detection. Exceedance detection compares features to pre-configured thresholds, which require significant domain expertise to identify and tune, and is used and detailed by the FAA. [1] However oftentimes these methods are brittle and complicated to develop or use. In order to increase efficiency of avionic diagnostics other papers have explored using machine learning architectures, including variations of autoencoders. However this is the first paper the authors are aware of that applied unsupervised learning to helicopter data. Self-supervised and unsupervised learning have both been shown to be applicable to aerospace and defense contexts. [2] Autoencoders are often used to perform anomaly detection, they are a pair of deep feed-forward neural networks, one that encodes the data into some latent subspace, and one that decodes the data back from its latent subspace to the original data. [3] The model is trained through minimizing the reconstruction loss, or the difference between the input and the output. [3] Lee t al. created a visualization and processing framework for anomaly detection in FOQA data using supervised machine learning. [4] Janakiraman et al. produced a multi-instance supervised classification learning technique for FOQA data. [5] Reddy et al. used an autoencoder to process time stamped raw data collected from multiple disparate flight sensors. [6] Similarly Guo et al. applied autoencoders to space based sensor data. [7] Zhou et al. improved the robustness of auto encoders based on principal component analysis. [8] Another example of machine learning applied to avionic diagnostics was when Memarzadeh et al. used variational convolutional autoencoders on Yahoo’s benchmark data, and did a case study on FOQA data. [9]

We used convolutional autoencoders on our natural language transformed to image data. A convolutional autoencoder is like that of a standard autoencoder with linear layers, but it has convolutional layers instead of fully connected layers as the substrate that makes up the encoder and decoder networks. [10] Our image transformation took inspiration from a spectrogram where the Fourier transform of a signal is plotted over time, given a visualization of the frequency spectrum over time, which many image processing models are able to evaluate downstream for signal identification. [11] Autoencoders can be thought of as a non-linear generalization of PCA, which is why we chose it as a baseline method of comparison. [12] We chose to cluster our data using K-means as it is one of the most popular clustering methods, and our classification goal of fault vs non-fault naturally fit a binary k-means with two desired cluster centroids that were easily distinguishable in an autoencoder or PCA’s subspace. [13]

The novel contributions of this paper are as follows:

1. Applying unsupervised machine learning methods to helicopter data
2. Transforming natural language fault data into binary images
3. Comparing machine learning methods to their deterministic counter-parts

3. MODEL AND DATA

3.1 Model

We implemented two autoencoder models in PyTorch. One using linear layers, which we refer to as a linear autoencoder, and the other using convolutional layers, that we refer to as a convolutional autoencoder. We attempted to use the linear autoencoder to process the bit and image transformation of our natural language data, but it failed to reconstruct both forms of data with low reconstruction loss. We used the convolutional autoencoder to process the image transformation of our natural language fault data, and it was able to do so successfully with low reconstruction loss. We trained both autoencoder models using reconstruction loss, as it is the standard loss function for autoencoders. See equation (1) for a definition of reconstruction loss, where x represents the original input, and $\hat{x}$ refers to the output from the autoencoder. The model was trained using a P100 GPU and Python 3.8 on a remote Ubuntu 18 Linux server running PyTorch 1.9. The summary for the convolutional model's architecture in PyTorch is shown in Figure 1.

$$L = |x - \hat{x}| \tag{1}$$

Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 8, 16, 1500]	80
ReLU-2	[-1, 8, 16, 1500]	0
Conv2d-3	[-1, 16, 8, 750]	1,168
BatchNorm2d-4	[-1, 16, 8, 750]	32
ReLU-5	[-1, 16, 8, 750]	0
Conv2d-6	[-1, 32, 3, 374]	4,640
ReLU-7	[-1, 32, 3, 374]	0
Flatten-8	[-1, 35904]	0
Linear-9	[-1, 8976]	322,283,280
ReLU-10	[-1, 8976]	0
Linear-11	[-1, 1024]	9,192,448
ReLU-12	[-1, 1024]	0
Linear-13	[-1, 512]	524,800
Linear-14	[-1, 1024]	525,312
ReLU-15	[-1, 1024]	0
Linear-16	[-1, 8976]	9,200,400
ReLU-17	[-1, 8976]	0
Linear-18	[-1, 35904]	322,310,208
ReLU-19	[-1, 35904]	0
Unflatten-20	[-1, 32, 3, 374]	0
ConvTranspose2d-21	[-1, 16, 8, 750]	4,624
BatchNorm2d-22	[-1, 16, 8, 750]	32
ReLU-23	[-1, 16, 8, 750]	0
ConvTranspose2d-24	[-1, 8, 16, 1500]	1,160
BatchNorm2d-25	[-1, 8, 16, 1500]	16
ReLU-26	[-1, 8, 16, 1500]	0
ConvTranspose2d-27	[-1, 1, 32, 3000]	73

Total params: 664,048,273

Figure 1. Convolutional Autoencoder architecture

We compared the output from the convolutional autoencoder model to the PCA algorithm on the image transformation of our fault data, as the linear model was unable to reconstruct both the bit and image transformation. We configured our PCA embeddings to capture 95% of the variance of the data. We clustered embeddings from the PCA and convolutional autoencoder model using K-means with two means to determine if either embedding was able to clearly distinguish fault from non-fault. We visualized the embedding spaces in two dimensions to see if the clusters were well formed, as well as qualitatively inspecting representative subsets of the clusters to determine if they accurately captured fault and non-fault respectively.

3.2 Dataset and Data Curation

We had 1584 example flights. For each example flight we had large binary data files that corresponded to instrumentation for each component used in flight, as well as a master log file that listed every fault reported by each component at a 1Hz frequency with a natural language description. To simplify our problem space for an initial proof-of-concept we chose to focus on the just the master log file, and only focus on faults reported for one component. We then enumerated every possible fault for that subsystem, and transformed our trimmed master log file into a time series of

binary arrays. For the subsystem we chose there were 35 possible faults, effectively converting each log entry to a 35 element Boolean vector enumerating the different faults present at each second. We normalized the length of each flight to 5000 seconds, then down sampled to 3000 seconds. We interpreted the resulting array of size 35x3000 as black and white images for the PCA and convolutional autoencoder algorithms. We also interpreted those enumerations as binary integers, by dropping 3 of the faults and converting the array of 32 binary elements to a 32-bit integer. We attempted to train the linear autoencoder on an array of 3000 32-bit integers representing an error code over time, however the linear autoencoder was unable to reconstruct the sequence accurately. The results from the linear autoencoder bit transformation reconstructions showed higher bits were almost always on. We hypothesize that the reason it failed to capture the integer sequence was specific faults being on resulted in a much larger weight on the error code representation, as higher bits correspond to larger integer values. See Figure 2 for an example of the input images constructed for a faulty flight log, and non-faulting flight log, as well as the corresponding reconstructions from the convolutional autoencoder.

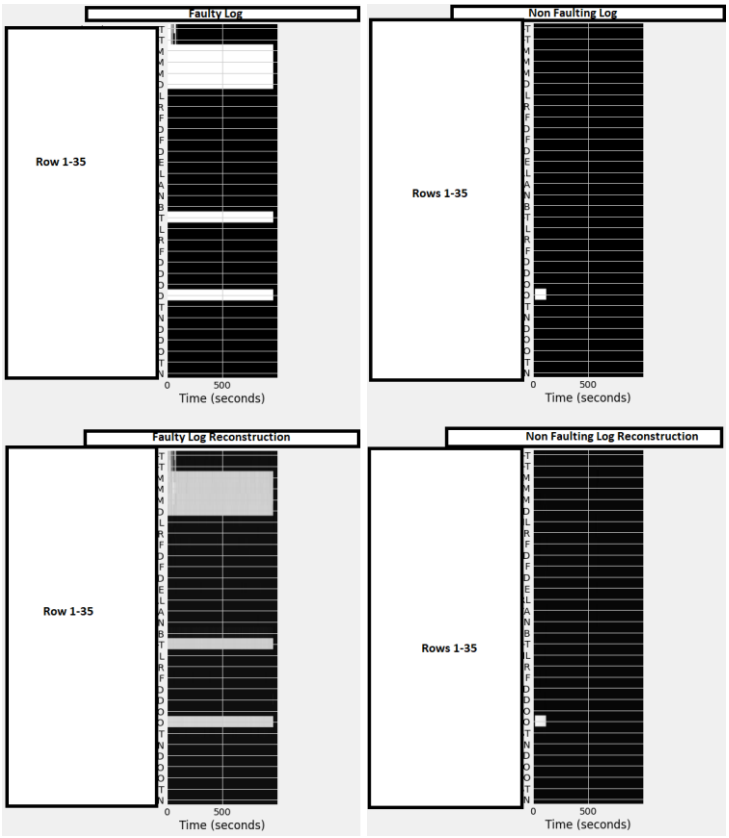

Figure 2. *Left:* Example of image transformation and convolutional autoencoder reconstruction for a faulty flight log. *Right:* Example of image transformation and convolutional autoencoder reconstruction for a non-faulting flight log.

4. RESULTS

Figure 3 shows the resulting clusters from K-means with two means on the embeddings obtained from the convolutional autoencoder. Figure 4 shows the resulting clusters from K-means with two means on the embeddings the PCA algorithm. Both are visualized in two dimensions. The convolutional autoencoder embeddings were passed through PCA down to two dimensions for visualization, and the first two elements in the PCA array were used for visualization of the PCA embeddings. Both sets of embeddings were obtained on the image transformation of the fault log data.

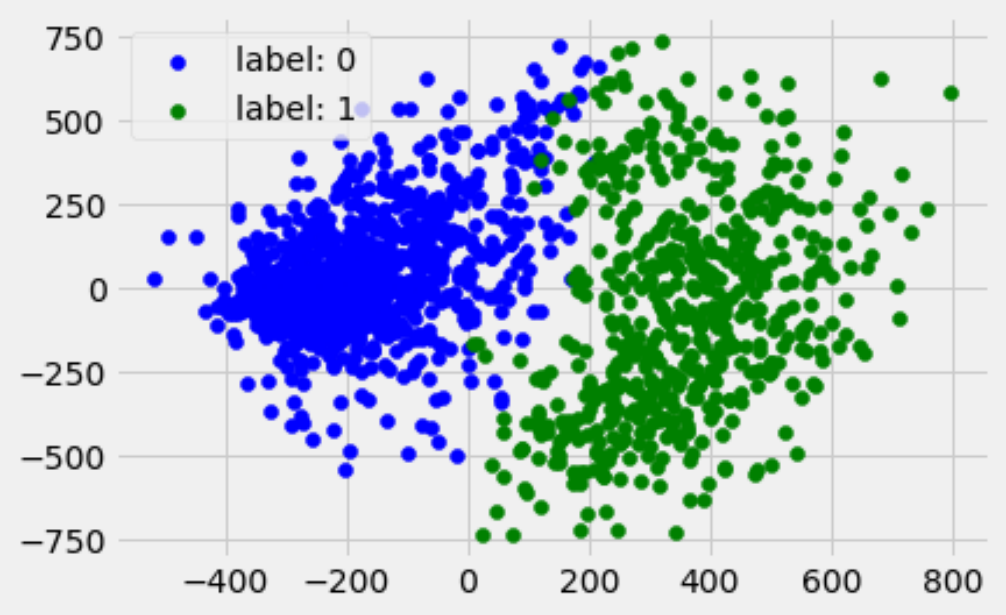

Figure 3. Clusters obtained from K-means on convolutional autoencoder embeddings visualized using 2-dimensional PCA

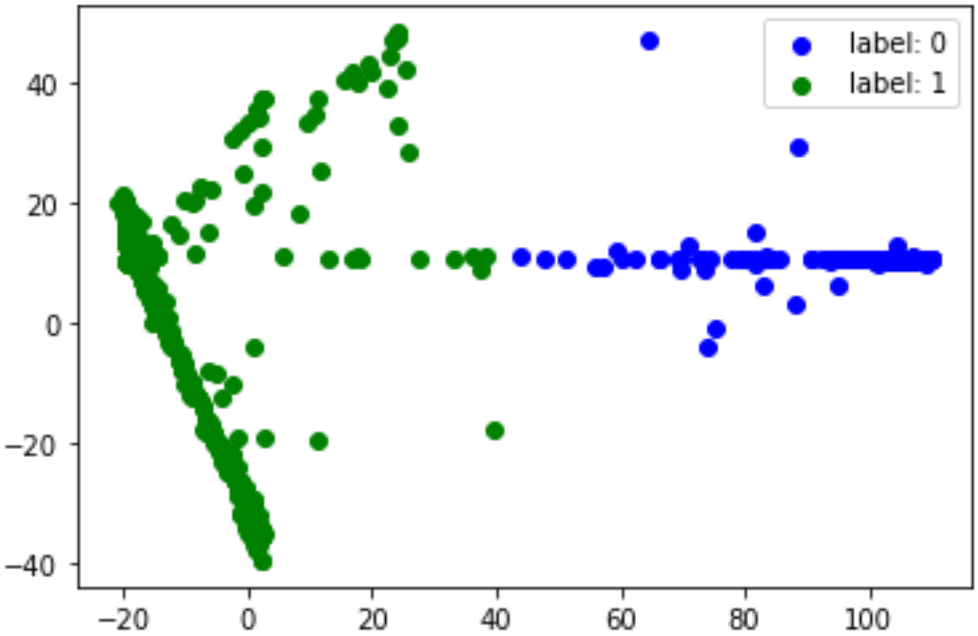

Figure 4. Clusters obtained from K-means on PCA embeddings visualized using first 2 elements in PCA embedding

Table 1 lists the number of members in each cluster for each algorithm compared to a rough number of expected number faults based on previous operational experience; approximately 10% of the flight logs should correspond to faults. For each cluster we determined which label (0 or 1) corresponded to fault vs non-fault by qualitatively inspecting representative elements of each cluster. See Figure 5 for an example faulty cluster member for both the PCA and convolutional autoencoder, with its corresponding cluster label.

Table 1. The number of faults reported by each algorithm's clusters compared to the expected number of faults from operational experience (roughly 10%). Determining which label corresponded to fault vs non-fault for each cluster was determined by qualitatively inspecting representative elements. The PCA cluster more closely matched expectation.

Model	**Faults *(cluster)***	**Non-Faults *(cluster)***
Principal Component Analysis	170 *(label: 0)*	1414 *(label: 1)*
Convolutional Autoencoder	545 *(label: 1)*	1039 *(label: 0)*
Expected	158	1426

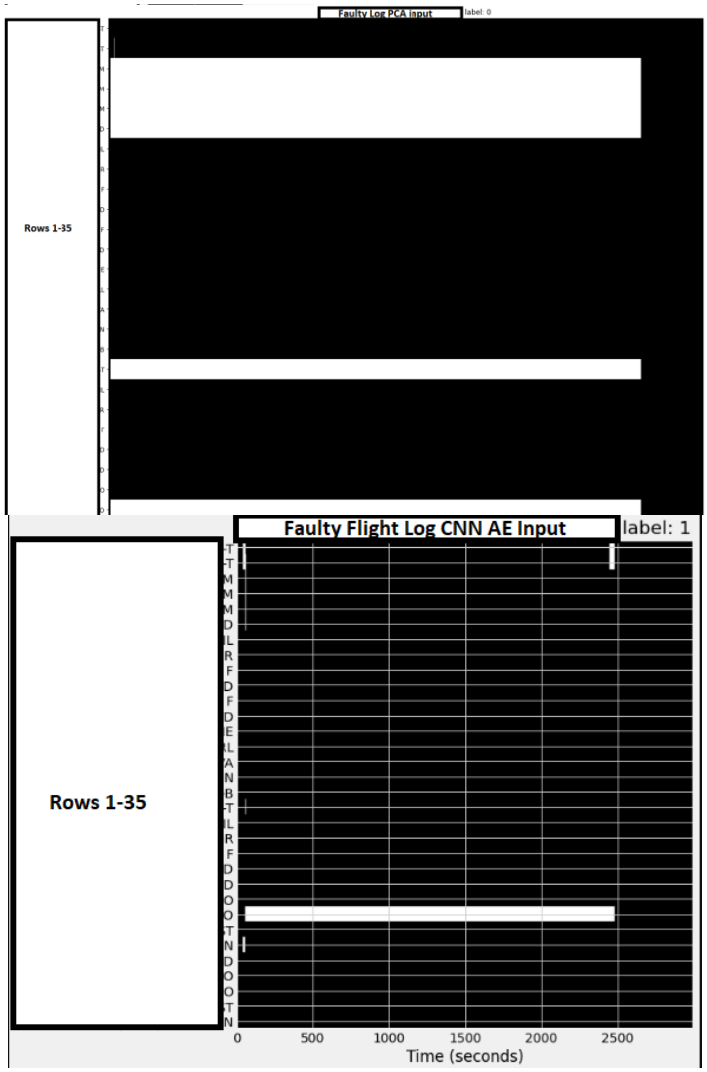

Figure 5. *Top:* Example faulty flight log image transformation and corresponding cluster label 0 from PCA embedding. *Bottom:* Example faulty flight log image transformation and corresponding cluster label 1 from convolutional autoencoder embedding

As can be seen from Table 1 the PCA seems to more accurately capture fault vs non-fault in its embedding space. This could be due to the large number of parameters in our convolutional autoencoder model not learning due to sparse data input, potentially leading to very slow learning gradients. However increasing the learning rate didn't seem to improve reconstruction results, but reducing the size of the data from 5000 to 3000 samples did. We looked at the cluster

members/labels for random subsets of our data, and determined qualitatively that the convolutional autoencoder is more prone to false positives and negatives when reporting faults. See Figure 6 for an example of a fault that the convolutional autoencoder falsely labeled as a non-fault, and the corresponding common example for non-fault (label 0) for the convolutional autoencoder.

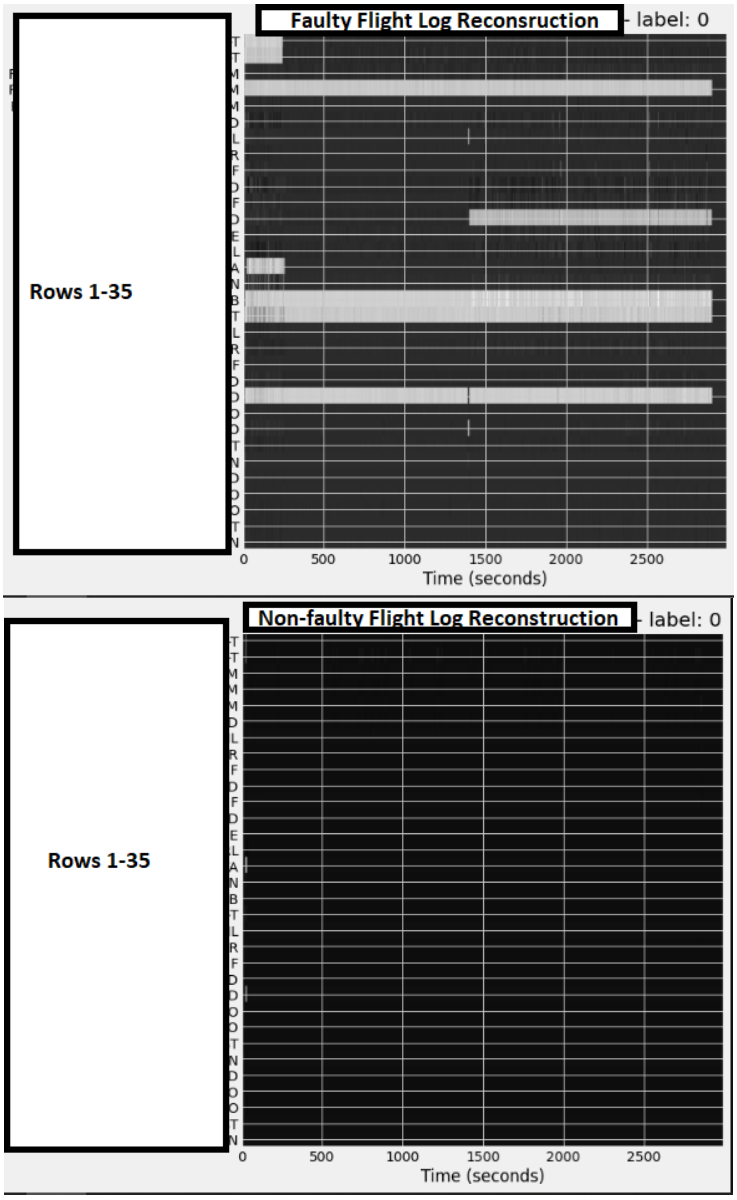

Figure 6: *Top:* Example faulty flight log image reconstruction from convolutional autoencoder embedding, with corresponding cluster label 0 as non-fault based on inspection of other cluster label 0 samples. *Bottom:* More frequent example of cluster label 0 corresponding to a non-faulty flight log data for convolutional autoencoder

5. CONCLUSION

Our results using unsupervised learning on a transformed version of avionic diagnostic data to a simple image processing problem indicate that intelligent data manipulation can be more important than advanced algorithms. The fact that PCA seems qualitatively to perform better at providing embeddings for clustering faults vs non-faults than a convolutional

autoencoder on the image transformation log data, and the fact that the linear autoencoder was not able to reconstruct the bit transformation log data both indicate that data curation is more important than model curation. If the data is curated properly for the model the model will perform well, and more advanced models might not out-perform simpler models if the data is well structured and abundant. However to further validate our findings and that claim there needs to be accurate quantification of fault clustering performance for both models, and further investigation into improving the vanilla convolutional autoencoder used.

A more intelligent AI model would leverage and correlate the instrumentation data we had provided for each flight with subsystem fault log data by parsing the timestamped binary data. One possible way of doing so is passing the different datatypes through parallel paths in a neural network, or perhaps by concatenating separate neural network outputs and using that for a final classification. Transformers are able to pay attention to an entire input sequence simultaneously, and transformer based autoencoders have been shown to perform well on anomaly detection for sequences directly. Transformers would effectively accomplish the translation of our natural language data to an image to encode the context of the entire input sequence into a single input, and would likely be the next best model to try.

Overall our results are promising in so far as validating the claim that the data contains enough intrinsic structure to distinguish fault from non-fault, which was our original goal. However further investigation and funding is needed to produce a robust algorithm for self-supervised avionic diagnostics.

REFERENCES

[1] Federal Aviation Administration. Flight Operational Quality Assurance. Technical Report; 2004.

[2] Purpura-Pontoniere, A., Imran, A.-A.-Z., and Bhattacharya, T., "Efficient ATR using contrastive learning", in *Automatic Target Recognition XXXII2022*, vol. 12096. doi:10.1117/12.2616822.

[3] Bank, Dor, Noam Koenigstein, and Raja Giryes. "Autoencoders." arXiv preprint arXiv:2003.05991 (2020).

[4] Lee, H.; Madar, S.; Sairam, S.; Puranik, T.G.; Payan, A.P.; Kirby, M.; Pinon, O.J.; Mavris, D.N. Critical Parameter Identification for Safety Events in Commercial Aviation Using Machine Learning. Aerospace 2020, 7, 73.

[5] Janakiraman, V.M. Explaining Aviation Safety Incidents Using Deep Temporal Multiple Instance Learning. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '18), London, UK, 19–23 August 2018; pp. 406–415.

[6] Reddy, K.K.; Sarkar, S.; Venugopalan, V.; Giering, M. Anomaly Detection and Fault Disambiguation in Large Flight Data: A Multi-modal Deep Auto-encoder Approach. Annu. Conf. Progn. Health Monit. Soc. 2016, 7.

[7] Guo, T.H.; Musgrave, J. Neural network based sensor validation for reusable rocket engines. In Proceedings of the 1995 American Control Conference-ACC'95, Seattle, WA, USA, 21–23 June 1995.

[8] Zhou, C.; Paffenroth, R.C. Anomaly Detection with Robust Deep Autoencoders. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '17), Halifax, NS, Canada, 13–17 August 2017; pp. 665–674.

[9] Memarzadeh, M.; Matthews, B.; Avrekh, I. Unsupervised Anomaly Detection in Flight Data Using Convolutional Variational Auto-Encoder. Aerospace 2020, 7, 115. https://doi.org/10.3390/aerospace7080115

[10] Zhang, Yifei. "A Better Autoencoder for Image: Convolutional Autoencoder." (2018).

[11] French, Mark & Handy, Rod. (2007). Spectrograms: Turning Signals into Pictures. Journal of Engineering Technology. 24. 32-35.

[12] Jolliffe Ian T. and Cadima Jorge. 2016 Principal component analysis: a review and recent developments. *Phil. Trans. R. Soc. A.* 374 : 20150202. 20150202.

[13] Jin, X., Han, J. (2011). K-Means Clustering. In: Sammut, C., Webb, G.I. (eds) Encyclopedia of Machine Learning. Springer, Boston, MA. https://doi.org/10.1007/978-0-387-30164-8_425