Use of Design of Experiments in Determining Neural Network Architectures for Loss of Control Detection

Newton H. Campbell Jr.
NASA Goddard Space Flight Center
LaRC OCIO Data Science Team
Greenbelt, MD 20771
newton.h.campbell@nasa.gov

Jared A. Grauer
NASA Langley Research Center
Dynamic Systems and Control Branch
Hampton, VA 23681
jared.a.grauer@nasa.gov

Irene M. Gregory
NASA Langley Research Center
Dynamic Systems and Control Branch
Hampton, VA 23681
irene.m.gregory@nasa.gov

***Abstract*—We describe empirical methods for selecting a neural network architecture to implement belief state inference on generic commercial transport aircraft. We highlight a case study on the planning, execution, and analysis of a set of experiments to determine the configurations of a conditional variational autoencoder (CVAE). Our main contribution is the application of a structured method that can be used for machine learning in many aerospace applications. This method optimizes the structure and training parameters of a neural network for belief state inference, using Design of Experiments (DOE) statistical methodologies. The motivation for this specific DOE analysis was to identify the appropriate hyperparameters for measuring the CVAE reconstruction probability and latent space, such that the measurements can be used to infer qualitative state changes for the aircraft. We demonstrate that this process yields information about a trained neural network's utility for this specific application, along with a quantifiable range of certainty. We execute 84 experiments using loss-of-control flight maneuver data from the NASA T-2 aircraft, demonstrating that this empirical process allows us to construct cheap and simple models with specific attributes amenable to belief state inference in aerospace applications.**

TABLE OF CONTENTS

1. INTRODUCTION 1
2. BACKGROUND 2
3. METHODOLOGY 2
4. EXPERIMENTS 4
5. ANALYSIS 6
6. CONCLUSIONS 12
APPENDIX 13
ACKNOWLEDGMENTS 13
REFERENCES 18
BIOGRAPHY 20

1. INTRODUCTION

Numerous warning systems have been developed and studied to predict when a flight vehicle approaches a loss-of-control (LOC) state. Current research focuses on developing anomaly detection models that inform intelligent agents, such that manned and unmanned aircraft alike may avoid abnormal situations that may lead to LOC [1–5]. This research is the natural progression of decades of identifying performance indicators that characterize the precursors for a flight to suffer a LOC event through empirical studies [6–10].

These empirical precursor studies have primarily been used to perform anomaly detection through either unsupervised or supervised methods. In unsupervised applications, models for anomaly detection are vetted against ground truth as specified by a particular study to demonstrate a capacity to identify when significant shifts in behavior are occurring. In supervised applications, studies tend to label and classify "normal" states, without an adequate representation of outlier behavior. Both approaches focus on performing anomaly detection by creating high-quality representations of inlier behavior. However, for anomaly detection in physical applications that can have multiple failure states, it has been shown that semi-supervised methods, informed by physics labelings, can produce more conservative representations of these states and permit a better understanding of system transition. For example, an aircraft may experience sequences of failure state transitions prior that may or may not reach a LOC state. Recently, the conditional variational autoencoder (CVAE) [11–13] has been demonstrated to be a viable probabilistic inference model for interpreting these kinds of transitions and detecting anomalies in physics [14–18].

Contribution

In this paper, we demonstrate the effectiveness of the CVAE model in detecting the transition of the NASA Generic Transport Model (GTM) aircraft to a LOC state. An in-depth characterization of its capacity for this application is highlighted in our previous work [19]. However, the main contribution described by this paper is the empirical process by which we develop a CVAE architecture that is amenable to the process of detecting aircraft state changes, with a particular focus on LOC. We model our analysis from the Design of Experiments (DOE) methodologies found in the *NIST/SEMATECH e-Handbook of Statistical Methods*, a widely used handbook for statistical analysis [20]. While our work provides insight into the use of CVAE in predicting LOC, our contribution is for this paper to serve as a template for designing the architecture of a CVAE (or other neural network models) in similar aerospace applications.

We report an analysis of design optimization, based on DOE methodology, to study the influence of architecture design parameters and training performance of a CVAE with recurrent neural network (RNN) layers to an aerospace application. While design optimizations to optimize neural network accuracy and training have previously been explored, our study focuses on the ability to use measurements of the

Figure 1. T-2 subscale jet transport aircraft (credit: NASA Langley Research Center)

CVAE to indicate state transition for aircraft. We develop and analyze a response surface model to show how we can obtain information about the utility of a trained neural network for a specific application, along with a quantifiable range of certainty about those values. We determine to what extent, if any, the selection of the type of recurrent neural network, activation function, optimization function, and dropout value have on the application. We assess not only the standard performance measurements (i.e. loss value, accuracy) but its actual applicability to detecting LOC and other state changes. Aerospace engineers often use DOE methodology for experimental design and analysis. This paper presents methods for using the same techniques and tools to construct cheap and simple models for architecting a neural network.

2. BACKGROUND

Loss-of-Control

The NASA Aviation Safety Program was established in response to a national goal to reduce the fatal aircraft accident rate by 80% within 10 years [21, 22]. The program involved all four NASA aeronautics centers and promoted coordination with the other government agencies, including the Federal Aviation Administration, industry, and academia. The Boeing Company and NASA Langley Research Center jointly developed loss-of-control (LOC) metrics for the NASA T-2 test aircraft (Figure 1). The T-2 is a 5.5% dynamically scaled version of a generic transport-type model [8], with retractable tricycle landing gear and twin jet engines mounted under the wings. These studies define LOC as any motion of the vehicle that is:

- outside of the normal operating flight envelopes
- not predictably altered by pilot control inputs
- characterized by nonlinear aerodynamic effects
- probable to result in high angular rates and displacements
- characterized by the inability to maintain heading, altitude, and wings-level flight

Design of Experiments for Neural Networks

DOE describes a domain of study in which systematic changes to the input variables of a system, or process under observation, are monitored in the context of system outputs [23]. These changes, the system, and outputs are studied for any combination of the following four motivations:

- Comparative — Determining to what extent, if any, variation in one or more input variables impacts the system or its outputs.
- Screening/Characterizing — Ranking the inputs of the system from most effective in explaining the output variation to least.
- Modeling — Developing a model relating the input variables to the system outputs.
- Optimization — Determining the ranges or set of values for input variables that optimize system outputs.

While formal DOE processes are often executed and improved by practitioners of the aerospace engineering domain [24–26], the field of machine learning has experienced significantly less practical use cases. As researchers in these fields continue to cross over, understanding how to apply DOE methodologies to determine machine learning model architectures has become a growing problem. The performance and results of machine learning models, particularly neural networks, are often reported with little-to-no rationale for the architectural design. DOE methodology presents an opportunity to ensure that statistical hypothesis testing methods are employed to assess the impacts of factors on outcomes correctly.

The direct use of DOE models and methods for neural network architecture design has been previously explored for a variety of applications [27–33]. However, most studies focus on the optimization of loss values, convergence speeds, and robustness measures of the neural network. While using these experimental outputs are appropriate for optimizing the accuracy of a neural network performing regression and classification in a specific domain, they do not necessarily account for the applicability of the neural network to a problem within the domain. The dynamics between a regression or classification inference and its utility in a certain application are rarely addressed.

Significantly less work has been done in applying DOE methods to the architectural design of autoencoders of any kind [34,35]. Hyperparameter optimization tools for machine learning software libraries like Keras and Tensorflow are often used to modify architectural choices for their autoencoder implementations [36–38]. Efficient methods of grid search, random search, and Bayesian optimization are the most commonly explored hyperparameter tuning algorithms behind such tools [39–43]. Like the previous DOE studies on neural networks, these tools only account for accuracy unless the application is incorporated into the loss function for optimizing the neural network. Given the recent popularity of these neural networks, our study serves as a guide for using the autoencoder that can be followed or automated in other environments.

3. METHODOLOGY

In this work, we apply DOE methodologies to optimize the neural network architectural design parameters for the application of detecting that an aircraft is transitioning between emergency states, with a focus on LOC. The factors that we choose are optimized to that application, not simply the accuracy of the neural network in modeling the data that we give it.

Computing Flight Envelopes for NASA T-2

As in the Wilborn study [8], we define the flight dynamics characteristics that play an important role in the causes of LOC, as shown in Table 1. In addition, we use subscripts to define the following measurements:

- α_{norm} and β_{norm} - Normalized α and β, respectively
- α_{sw} - α (deg) for stall warning activation

Table 1. **Key Flight Measurements for Determining LOC**

angle of attack	α	sideslip angle	β
bank angle	ϕ	pitch attitude	θ
equivalent airspeed	V_e	normal load factor	n
pitch control	δ_c or δ_e	pitch rate	q
roll control	δ_w or $\frac{\delta_a}{\delta_{sp}}$	roll rate	p

- β_{mdxw} - sideslip (deg) for non-crabbed approach in the max demonstrated crosswind for takeoff and landing
- V_e - equivalent airspeed, kts
- V_{fe} - max flaps extended operating equivalent airspeed (flaps down), kts
- V_{mo} - max operating equivalent airspeed (flaps up), kts
- V_{norm} - normalized airspeed
- V_{sw} - stall warning equivalent airspeed in 1-g flight, kts

The number of flight envelopes that are violated by the aircraft correlate heavily with the vehicle being in an LOC state [8]. The envelopes are:

Adverse Aerodynamics (AA)—The boundaries of this envelope represent the maximum limits of α and β a line pilot should expect to encounter in normal flight operations, including all emergency procedures covered by checklist. The AA flight envelope is defined by:

$$\alpha_{norm} = 0 \text{ at } \alpha = 0^\circ \tag{1}$$

$$\alpha_{norm} = 1 \text{ at } \alpha = \alpha_{sw} \tag{2}$$

$$\beta_{norm} = -1 \text{ at } \beta = -\beta_{mdxw} \tag{3}$$

$$\beta_{norm} = +1 \text{ at } \beta = +\beta_{mdxw} \tag{4}$$

Unusual Attitude (UA)— This envelope relates information about the flight path parameters that pilots rely on most in recovering from upsets and in maintaining control for continued safe flight and landing. The UA envelope is defined by:

$$-45^\circ \leq \phi \leq +45^\circ \tag{5}$$

$$-10^\circ \leq \theta \leq +25^\circ \tag{6}$$

Structural Integrity (SI)— This envelope bounds the normalized airspeed facilitates comparisons between different airplanes and configurations. The SI envelope is defined by:

$$V_{norm} = \frac{V_e - V_{sw}}{V_{mo} - V_{sw}} \text{ (Flaps-Up Configuration)} \tag{7}$$

$$V_{norm} = \frac{V_e - V_{sw}}{V_{fe} - V_{sw}} \text{ (Flaps-Down Configuration)} \tag{8}$$

Dynamic Pitch Control (DPC)—The limits of this envelope reflect whether the trend in θ' is consistent with pitch control commands, or whether the control is opposing the aircraft motion. It is computed as the sum of the current pitch angle and its time derivative:

$$\theta' = \theta + \dot{\theta} \tag{9}$$

Dynamic Roll Control (DRC)—The limits of this envelope are analogous to those for the DPC envelope, i.e., reflecting whether the trend in roll attitude is consistent with roll control commands or whether the control is opposing the aircraft motion.

$$\phi' = \phi + \dot{\phi} \tag{10}$$

Additional Flight Envelopes— In addition, we add 4 more envelopes based on real-world LOC constraints that have been added since the original study:

1. Weight:
$$47.16 \leq W \leq 58.16 \text{ lbs} \tag{11}$$
2. Altitude:
$$0 \leq \text{alt} \leq 2000 \text{ ft} \tag{12}$$
3. Flap Deflection:
$$-5^\circ \leq \text{flap deflection } \leq 25^\circ \tag{13}$$
4. Center of Mass:
$$50 \leq \text{x}_{cg} \leq 60 \text{ in} \tag{14}$$

Conditional Variational Autoencoders for Detecting LOC

Our CVAE methods for interpreting that the T-2 aircraft is approaching a LOC were described and characterized in our prior work [19]. Here, we report the DOE methods that guided our design decisions for this CVAE architecture used in that characterization. We train a CVAE by labeling the measurements in Table 1 with a binary vector that describes whether or not the vehicle is in each envelope (1 if the vehicle is in an envelope, 0 otherwise). The CVAE learns the behavior of flight dynamics for each envelope configuration and uses that as a model to detect unusual behavior. During flight, the CVAE attempts to reconstruct flight measurements from encodings. If reconstructed measurements veer too far away from the actual flight measurements, we know that something anomalous is occurring. Our exploration of the capacity of a CVAE to model complex relationships in data from the NASA T-2 resulted in the model structure depicted in Figure 2.

We construct the encoder portion of the CVAE using variants of bidirectional recurrent neural network (RNN) layers and use unidirectional layers to construct the decoder portion. Intuitively, only the encoder side of the CVAE needs to be bidirectional. The latent space distributions that represent flight measurement encodings will capture information about future timesteps, as well as past timesteps. The data are encoded in such a way that only the sequence of the distributions matter. We describe the details of the encoder and decoder in Section 4, in the context of our DOE analysis.

The reconstruction probability is calculated by the stochastic latent variables that derive the parameters of the original input variable distribution [14]. What is being reconstructed are the parameters of the input distribution, not the input variable itself. This allows us to compute the probability of the data being generated from a given latent variable drawn from the approximate posterior distribution.

In this study, we predicted LOC of a commercial transport by assessing two measurements of the neural network illustrated above: **Reconstruction Probability** and **Gaussian Shift**.

The reconstruction of commercial transport flight data from the low-dimensional representation of the CVAE typically

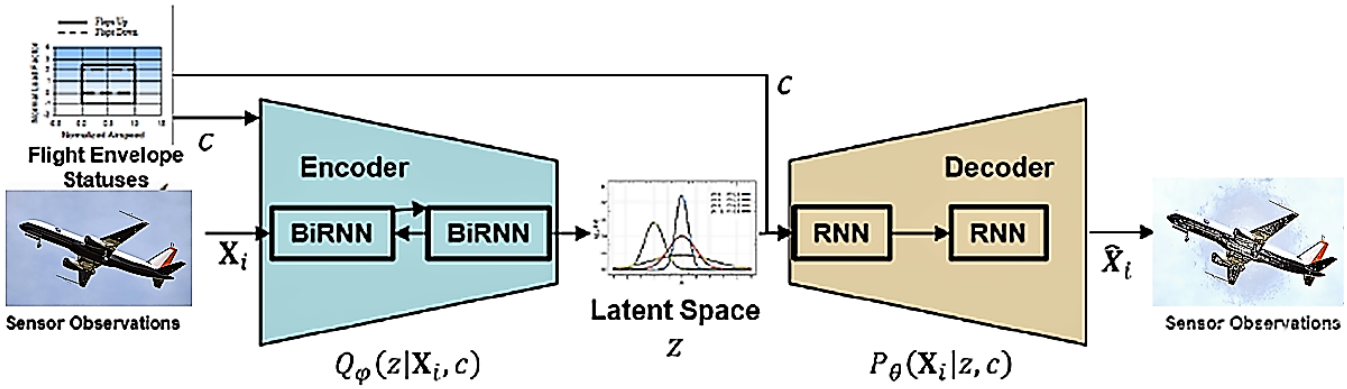


Figure 2. Schematic visualization of a conditional variational autoencoder that encodes observations from NASA T-2 flight data, based on the observation's envelope status, into a probabilistic latent space representation. The CVAE samples from this representation for decoding.

represents the true nature of the measurements, without any uninteresting features and noise. At time t, the reconstruction probability $R(\mathbf{X}_t|\hat{\mathbf{X}}_t)$ is computed by estimating the reconstruction $\hat{\mathbf{X}}_t$ of input $\mathbf{X}_t$ with respect to a binary encoding of the flight envelope status c:

$$R(\mathbf{X}_t|\hat{\mathbf{X}}_t) = \frac{1}{L}\sum_{l=1}^{L} logP_\theta(\mathbf{X}_t|z,c) \tag{15}$$

for L latent vectors of the input data. The CVAE is trained to minimize the following loss function:

$$L_{CVAE}(\mathbf{X}_t, c, \hat{\mathbf{X}}_t; \theta, \phi) = D_{KL}(Q_\phi(z|\mathbf{X}_t, c)||P_\theta(\hat{\mathbf{X}}_t|z,c)) + R(\mathbf{X}_t|\hat{\mathbf{X}}_t) \tag{16}$$

We establish a threshold α such that if $R(\mathbf{X}_t|\hat{\mathbf{X}}_t) < \alpha$, we declare that the observation at that timestep is anomalous. This indicates that LOC is occurring at that specific timestep.

To identify shifts in flight dynamics for a particular flight envelope configuration, at each timestep t and subsequent timestep $t+1$, we analyze a shift from the Gaussian distribution $P(z_t)$ required to reconstruct an input $\mathbf{X}_t$ to the Gaussian distribution $P(z_{t+1})$ required to reconstruct an input $\mathbf{X}_{t+1}$.

As our latent space has more than one dimension (more than one Gaussian sampling distribution), we represent the latent variables of our model as a multivariate distribution. We use the multivariate KL-Divergence [44] to measure the shift in latent space representation over subsequent timesteps. The KL-Divergence between a multivariate Gaussian distribution P with means $\mu_1 \in \mathbb{R}^n$ and covariances $\Sigma_1^2 \in \mathbb{R}^n \times \mathbb{R}^n$ and multivariate Gaussian distribution Q with means $\mu_2 \in \mathbb{R}^n$ and covariances $\Sigma_2^2 \in \mathbb{R}^n \times \mathbb{R}^n$ is given by:

$$D_{KL} = \frac{1}{2}[\log\frac{|\Sigma_2|}{|\Sigma_1|} - n + \text{tr}\{\Sigma_2^{-1}\Sigma_1\} + (\mu_2 - \mu_1)^T\Sigma_2^{-1}(\mu_2 - \mu_1)] \tag{17}$$

For two arbitrary probability distributions, P and Q, the KL-Divergence describes how much information is lost if Q is an approximation of P. This is not a symmetric measurement. The Jensen-Shannon (JS) divergence is the symmetric version of the KL Divergence, defined as:

$$D_{JS}(P||Q) = \frac{D_{KL}(P||M) + D_{KL}(Q||M)}{2} \tag{18}$$

where $M = \frac{1}{2}(P+Q)$, $M \sim N(\mu_M, \Sigma_M)$, where $\mu_M = \frac{1}{2}(\mu_P + \mu_Q)$ and $\Sigma_M = \frac{1}{2}(\Sigma_P + \Sigma_Q)$.

To obtain a metric that can be analyzed over time, we simply use $\sqrt{D_{JS}(P \,||\, Q)}$ as the JS Distance.

Since there are no constraints on the μ and Σ distribution parameters represented by the latent space, the encoder can learn to generate very different means for different classes, some of which cluster together. Furthermore, the variances of the models are typically small. This allows the RNN decoder to efficiently reconstruct the training data and signifies that large shifts in the means of the latent space have some correlation to an anomalous shift in behavior for a particular flight envelope configuration or towards LOC. We track these larger shifts by continuously measuring the gradient of the JS-Distance. Formally, let τ_z be an experimentally determined threshold for the latent space. We propose a detector such that if $\sqrt{D_{JS}(P \,||\, Q)} > \tau_z$, an alert is raised indicating that the vehicle is pushing envelope limits.

4. EXPERIMENTS

The DOE setup and analysis in this study followed formats described in the *NIST/SEMATECH e-Handbook of Statistical Methods* [20]. We modeled our analysis after Section 5.4 — *Analysis of DOE data*. This section attempts to convey the fundamental idea behind empirical model-building, which allowed us to construct cheap and simple models that characterize the impact of different CVAE architectures. The purpose of this design optimization was to determine the effect of training hyperparameters and performance for a conditional variational autoencoder on the inference of T-2 loss-of-control (LOC) and envelope change.

The AirSTAR program measured the data used for neural network training and evaluation of these experiments during flights of the T-2 [45]. During these flights, the aircraft was piloted by a research pilot in a mobile control room using synthetic vision displays. During each flight, the pilot

executed several slow stall maneuvers, where they slowly provided a nose-up elevator command and injected multi-axis excitation to the control surfaces. As this happened, the aircraft climbed, increased in pitch angle, and decreased in airspeed. After each stall, the aircraft nose dropped and the aircraft rolled. The pilot then regained airspeed and leveled the wings by applying normal stall recovery controls. Each of these events happened at varied times throughout the flight and constituted an LOC. We labeled each observation of the data according to envelopes violated during the observation and whether the aircraft was in an LOC state. We conducted our DOE using 58 flight missions and captured recordings of specific maneuvers.

Response Variables and Factors

The following are the response variables for our set of experiments:

- y_1: Average KL loss [11] over full flights
- y_2: Average reconstruction loss over full flights
- y_3: Average KL loss over specific maneuvers
- y_4: Average reconstruction loss over specific maneuvers
- y_5: Area of intersection for reconstruction probabilities of LOC and normal flight observations
- y_6: Area of intersection for Gaussian shift of state change and normal flight observations
- y_7: Balanced accuracy for reconstruction probability-based anomaly detection
- y_8: Difference in the average reconstruction probability between normal and LOC observations (# violated envelopes ≥ 3)
- y_9: Difference in the average value of the gradient of Jenson-Shannon distance when entering/exiting an envelope and no envelope change

The following are discrete factors for our sequence of experiments:

- x_1: Layer Type — Gated Recurrent Unit (GRU) [46], Long Short-Term Memory (LSTM) [47]
- x_2: Activation Function — Exponential Linear Unit (ELU) [48], Linear [49], ReLU [50], Scaled Exponential Linear Units (SELU) [51], Softplus [52], Softsign [53], tanh [54]
- x_3: Optimization Function — Adam [55], Stochastic Gradient Descent (SGD) [56], Adadelta [57]
- x_4: Dropout — (0,0.1)

We include the training results of the neural network as additional factors in our experiments. Each neural network was trained on observations from 46 T-2 aircraft flight missions and validated on 12 flight missions. On average, missions were 17 minutes in length, and data was resampled to 1s intervals. The following training results serve as continuous factors in our DOE:

- x_5: Number of epochs used to train the CVAE
- x_6: Final CVAE training loss measure
- x_7: Final CVAE validation loss measure

We did not use the number of layers or nodes as factors, as the effect is simply a more accurate fit to training data and capturing distinct input features. Our goal was to explore how the actual equations used for inference modify the network's utility in envelope state inference and LOC detection.

We specified these factors and responses as such in Design-Expert. We then performed 84 experiments, using the observations from the T-2, which yielded all combinations of our discrete factors. In future studies, due to the random initialization of CVAE weights and variance in training performance, multiple experiments should occur for each configuration of factors to account for randomization effects.

Measurement

Each CVAE was trained with a conditioning vector computed based on the flight envelope status during an observation. The encoder portion of the network was comprised of the following layers, from input to output:

- A single feed-forward layer with tanh activation
- 1 bidirectional RNN layers (type x_1) with x_2 output activation
- 7 bidirectional RNN layers (type x_1, dropout x_4) of decreasing size (from input to latent space) with x_2 output activation

The tanh activation at the beginning of the encoder outputs values between -1.0 and 1.0. Extreme values typically saturate to -1.0 and 1.0, which is reasonable for our application. Whether an extreme shift or a moderate shift in the input values occurs, most neurons at the beginning layer should fire to their maximum potential. The architecture of the encoder follows with a single recurrent neural network layer that captures all information and several recurrent neural network layers with dropout, which prevents co-adaptation of features across subsequent layers [58]. We set the number of dimensions of the latent space to 11, which is half of the number of input variables (22). Using at least half the input size for the latent space has empirically shown significant performance benefits across VAE studies [59]. The decoder is comprised of the following layers, from input to output:

- A single feed-forward layer with x_2 activation
- 7 unidirectional RNN layers (type x_1, dropout x_4) of increasing size (from latent space to output) with x_2 output activation
- 1 unidirectional RNN layers (type x_1) with x_2 output activation
- A single feed-forward layer with tanh activation
- A single feed-forward layer with linear activation

The decoder effectively mirrors the encoder, starting with a single feed-forward layer to capture and learn the full encoding representation before it is sent to layers that may have dropout.

We used the Keras deep learning framework [60] to implement each experimental CVAE. The Keras Functional API allowed us to combine layers very easily. We trained the T-2 flight data on CVAE architectures, each with architectures specified by the factors of the 84 combinations. We compare and contrast models with varying hyperparameters and architectures to develop a response surface for the ability of the CVAE model to detect loss-of-control and describe belief state change. We trained and vetted each CVAE in parallel on the NASA Langley Research Center K-cluster, which used one NVIDIA Tesla K40 GPU per experiment. Each network was trained for a maximum of 10,000 epochs, with early stopping if the validation loss of the network was non-decreasing for 1,000 epochs. Most networks converged at about 2,200 epochs. The average neural network training phase was approximately 5 hours, with all of the networks completing after approximately 21 hours. After this, experimental outcomes were measured and captured for analysis.

The first four outcomes measured are direct, standard mea-

surements of the performance and robustness of the CVAE on modeling our data. For each network, y_1 and y_3 were measured by encoding observations of all full flight missions and specific maneuvers, respectively. We computed the KL-Divergence between the encoded input data and the standard Gaussian and averaged this measure across all observations. This helped us understand if movement in the latent space impacts the outcomes of our application. y_2 and y_4 were measured by calculating the Monte Carlo estimate of $E_{q_\phi(z|x)}[logp_\theta(x|z)$ for all full flight missions and specific maneuvers, respectively.

To measure y_5, we denote the density estimates of reconstruction probability over all full flight missions as δ_n for reconstruction probabilities of normal observations and δ_λ for reconstruction probabilities of LOC observations. Also, let min_n represent the lowest reconstruction probability observed for normal observations and let max_λ represent the largest reconstruction probability observed for a LOC observation. We measured y_5 using the estimate of the area under the curve of these two densities:

$$y_5 \approx \int_{min_n}^{max_\lambda} |\delta_n(t) - \delta_\lambda(t)|dt \tag{19}$$

In the same way, we denote δ_{sc} as the Gaussian shift (Jensen-Shannon Distance) in the latent space from previous timestep $t-1$ to the current timestep t for observations in which a state change occurred at time t and δ_{nc} for observations in which no state change occurred. And max_{nc} is the largest Gaussian shift observed when no state change has occurred at time t and min_{nc} is the smallest Gaussian shift observed when a state change has occurred. We measured y_6 using an estimate of the area under the curve:

$$y_6 \approx \int_{min_{sc}}^{max_{nc}} |\delta_{nc}(t) - \delta_{sc}(t)|dt \tag{20}$$

The measurement y_7 describes how well the neural network can predict that the aircraft is **in** a LOC state. We set the anomaly threshold α_r by identifying the percentile P_i of reconstruction probabilities that yielded the closest overestimate of the actual number of LOC observations. If, at timestep t, $R(\mathbf{X}_t|\hat{\mathbf{X}}_t) < \alpha_r$, we infer that the aircraft is under LOC. Using this to determine true positives (TP), true negatives (TN), false positives (FP), and false negatives (FP), we calculate y_7 as:

$$y_7 = \frac{\frac{TP}{TP+FN} + \frac{TN}{TN+TP}}{2} \tag{21}$$

Finally, we examined the differences in average values for LOC inference and (envelope) state change reconstructions. For y_8, we computed the difference between the average reconstruction probability for operational control observations (ones in which the vehicle is not in an LOC state) and the average reconstruction probability for LOC observations. For y_9, we examined the average gradient of the Jenson-Shannon distance for observation sequences that do not result in envelope changes and calculated the difference from the average gradient of those that do.

Results

A summary description of our results, described as the average responses for each experiment factor, is shown in Table 4 of the Appendix. After our experiments, we followed the steps defined in *NIST/SEMATECH e-Handbook of Statistical Methods* Section 5.4.7.3 [20] for response surface modeling for each of our responses $y_i, i = 1...9$. The following is a summary description of the steps of this analysis process:

1. Fit the full model to response y_i.
2. Use stepwise regression, forward selection, or backward elimination to identify important variables.
3. When selecting variables for inclusion in the model, apply the hierarchy principle. Keep all the main effects of significant higher-order terms or interactions, even if the main effect p-value is larger.
4. Generate diagnostic residual plots for the model selected.
5. Examine the fitted model plot, interaction plots, and ANOVA statistics to determine if the model fit is satisfactory.
6. Use contour plots of the response surface to explore the effect of changing factor levels on the response.
7. Repeat all the above steps for another response variable.
8. After satisfactory models have been fit to both responses, you can overlay the surface contours for both responses.
9. Find optimal factor settings.

Table 2. Summary of Adjusted R^2 scores for a full model of each response.

Outcome	Model	Adjusted R^2
y_1	Quadratic	0.9868
y_2	Quadratic	0.9251
y_3	Quadratic	0.9854
y_4	Quadratic	0.9262
y_5	Linear	0.657
$\log(y_6 + 1)$	Linear	0.846
$\log(y_7 + 1)$	Linear	0.2853
$\frac{1}{y_8+0.001}$	Quadratic	1.0
$\ln(y_9 + 0.001)$	Linear	0.8252

For each response, y_i, the equation for the full quadratic model is:

$$y_i = \beta_0 + \sum_{j=1}^{7} \beta_j x_j + \sum_{1 \le j < k \le 7} \beta_{jk} x_j x_k + \sum_{j=1}^{7} \beta_{jj} x_j^2 + \epsilon \tag{22}$$

where β_0 is a constant (intercept), β_1 is a linear effect parameter, β_2 is a quadratic effect parameter, and ϵ is error.

Table 2 shows the adjusted R^2 value when we fit a full model (all main effects and interaction terms) to each response. By default, we attempted full quadratic models as the first step for each outcome. However, some of these full quadratic models resulted in negative predicted R^2 values. In such cases, we applied a transform to each outcome (based on analysis of the average main effect values in Table 4) and then fit a full model.

5. ANALYSIS

We leverage the analysis tools in Design Expert, as well as Python libraries (NumPy [61], SciPy [62], Statsmodels [63], seaborn [64]) to analyze and visualize our DOE results. The following is a summary of our analysis for each of the aforementioned experimental responses.

Response y_1: Average KL Loss over all Missions

We start by fitting a full quadratic model for Average KL Divergence of the latent space sampling distribution from the standard Gaussian, using ordinary least squares. The R^2 and adjusted R^2 were fairly high for the y_1 full quadratic model. We then perform stepwise regression for the Average KL Loss, with a focus on minimizing the Akaike information criterion (AIC) [65]. By using AIC as stopping criteria for stepwise regression, our analysis penalizes the model for the number of coefficients. After 14 steps, this resulted in the following model with an AIC of 131.78:

$$\begin{aligned} y_1 \sim x_1^2 + x_2^2 + x_6^2 \\ +x_1x_3 + x_1x_4 + x_1x_6 + x_1x_7 + x_2x_7 + x_3x_5 \\ +x_3x_7 + x_4x_5 + x_4x_6 + x_5x_7 + x_6x_7 \\ +x_1 + x2 + x3 + x5 + x6 + x7 \end{aligned} \tag{23}$$

For visual clarity of the relationship between factors and responses, we remove the coefficients from this and subsequent equations, with the implication of a constant added to the model as a final term. Stepwise regression tells us which of the main factors and interactions are needed to reliably infer a quadratic relationship for the Average KL Loss. Next, we follow the principle followed by most statisticians of keeping all main effects that are part of significant higher-order terms and interactions, known as the *effect hierarchy principle* [66]. We do not include an interaction term in a model unless both main effects are included[2]. The dropout, x_4, does not appear as a main effect in Equation 23. Therefore, we use an estimate from a previous regression step that contains x_4 as a main effect, granting us a final reduced model for Average KL Loss as:

$$\begin{aligned} y_1 \sim x_1^2 + x_2^2 + x_4^2 + x_6^2 \\ +x_1x_3 + x_1x_4 + x_1x_6 + x_1x_7 + x_2x_7 + x_3x_5 \\ +x_3x_7 + x_4x_5 + x_4x_6 + x_5x_7 + x_6x_7 \\ +x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 \end{aligned} \tag{24}$$

The AIC for this model was only (10^{-12}) larger than the model described in Equation 23, so the model maintained the same goodness of fit and simplicity.

Plots from our analysis of residuals are visualized in Figure 3. This visual representation of the normal plot of the residuals,

[2]Note that this process is done automatically by Design-Expert.

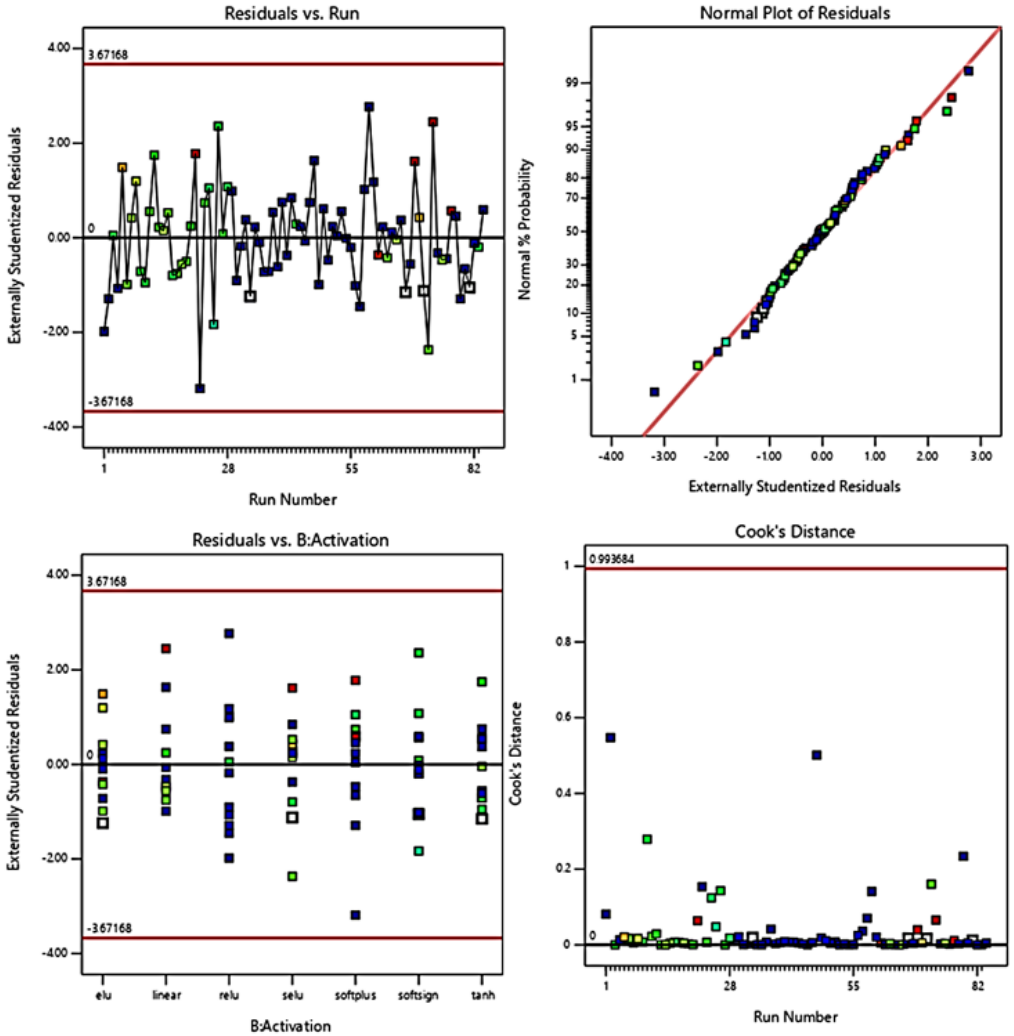


Figure 3. The residuals plots for the Average KL-Divergence loss (y_1) model does not indicate any problems with our underlying assumptions.

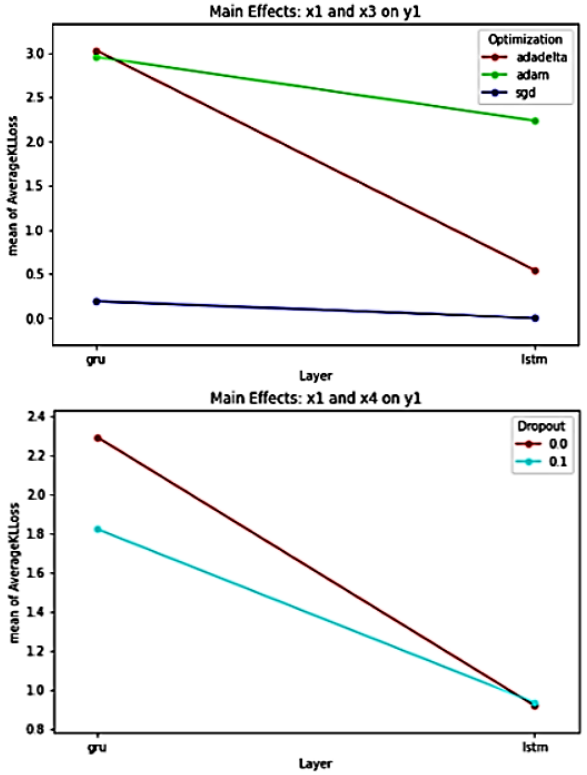


Figure 4. Two Interaction Plots for average KL loss response over all mission flight data(y_1).

a comparison of the residuals to the actual experimental runs, a plot of the residuals versus each main effect, and a Cook's distance regression [67], indicates that we can have confidence in the underlying assumptions of the model described by Equation 24. The adjusted R^2 for the model generated from stepwise regression is 0.918, slightly lower than the full quadratic model, but not significantly lower. And the Predicted R^2 of 0.8069 is in agreement with the Adjusted R^2 of 0.9180.

Figure 4 shows two of the interaction plots, Layer-Optimization(x_1x_3) and Layer-Dropout(x_1x_4), for this model of average KL-Divergence Loss (y_1). And the contour plots in Figure 5 show how the training performance for each CVAE impact the average KL-Divergence from the standard Gaussian over full flights. Both plots confirm the need for these interaction term in the model (otherwise, the lines would be parallel). This analysis, combined with the performance data in Table 4, allow us to make the following claims:

- Latent space learning is stable: Figure 5 shows that as the final training loss improves, so does the final validation loss.
- The relationship between number of training epochs and the final validation loss is parabolic. If too many epochs are used for training, the network will be overfit.
- GRUs with an 0.0 dropout tend to have larger average KL losses. Because of the defined loss function for the CVAE, this will result in longer training times for neural networks with this architecture.

Response y_2: Average Reconstruction Loss over all Missions

The stepwise regression to identify the most influential factors for the average reconstruction loss over all missions (y_2) resulted in the following model after 18 steps with an AIC of 576.95:

$$\begin{aligned} y_2 \sim\ & x_2^2 + x_5^2 + x_7^2 \\ & + x_1x_4 + x_1x_7 + x_2x_5 + x_3x_5 \\ & + x_3x_7 + x_4x_5 + x_4x_7 + x_5x_7 \\ & + x_1 + x_2 + x_3 + x_4 + x_5 + x_7 \end{aligned} \tag{25}$$

All main factors except x_6 (final training loss) are necessary to appropriately fit our model. It is already excluded from any interaction terms in this model. This model already adheres to the effect hierarchy principle. The adjusted R^2 for this model is 0.7464, much lower than that of the full quadratic model. The predicted R^2 is actually negative (-0.3736), which implies that it is unreliable (the mean would be a better estimate of y_2). The residuals plots, one of which is shown in Figure 6, indicate that there is a significant outlier in this model. The following are the discrete factors from this outlier experiment:

- Layer — GRU
- Activation — Softplus
- Optimization — SGD
- Dropout — 0

Further inspection of the Predicted vs. Actual chart in Figure 7 shows that the largest outliers are from experiments in which the softplus activation function were used for the RNN and the SGD optimization function was used. The epochs, as well as the final training and validation losses for experiments with these properties were fairly normal. The activation function (x_2) has a quadratic relationship with the y_2 outcome, whereas the optimization's (x_3) relationship is linear with interaction effects. The interaction effect for x_2x_3 was removed late during stepwise regression. When added back into Equation 25, the adjusted R^2 is 0.9252 with high adequate precision.

Response y_3: Average KL Loss over Specific Maneuvers

The stepwise regression model, computed for y_3 (KL Loss for specific maneuvers), is not similar to that of y_1 (KL Loss for full flight missions). The model is described as follows, with no changes due to the effect hierarchy principle:

$$\begin{aligned} y_3 \sim\ & x_1^2 + x_4^2 + x_5^2 + x_6^2 \\ & + x_1x_3 + x_1x_4 + x_1x_6 + x_1x_7 + x_3x_5 \\ & + x_3x_7 + x_4x_5 + x_4x_6 + x_5x_7 + x_6x_7 \\ & + x_1 + x_3 + x_4 + x_5 + x_6 + x_7 \end{aligned} \tag{26}$$

The adjusted R^2 for the y_3 model, 0.8856, is slightly less than that of y_1. There were no major outliers shown by our analysis of residuals. We investigated y_3 further by performing an adjusted means squared assessment [68] on a model with higher-order interaction terms. The results of this assessment revealed the top 5 interaction effects of y_3, shown in Table 3. These 5 effects are the outliers on the higher-end of the quantile-quantile plot for y_3, shown in Figure 8. The activation function factor, x_2, is not a part of the model derived by stepwise regression. However, the most significant positive effect shown by the quantile-quantile plot includes x_2. Generally, strong interaction between the amount to which the network is trained (x_5—x_7) and the discrete factors (x_1—x_4) significantly impacts the latent space encodings of the CVAE when observing individual maneuvers.

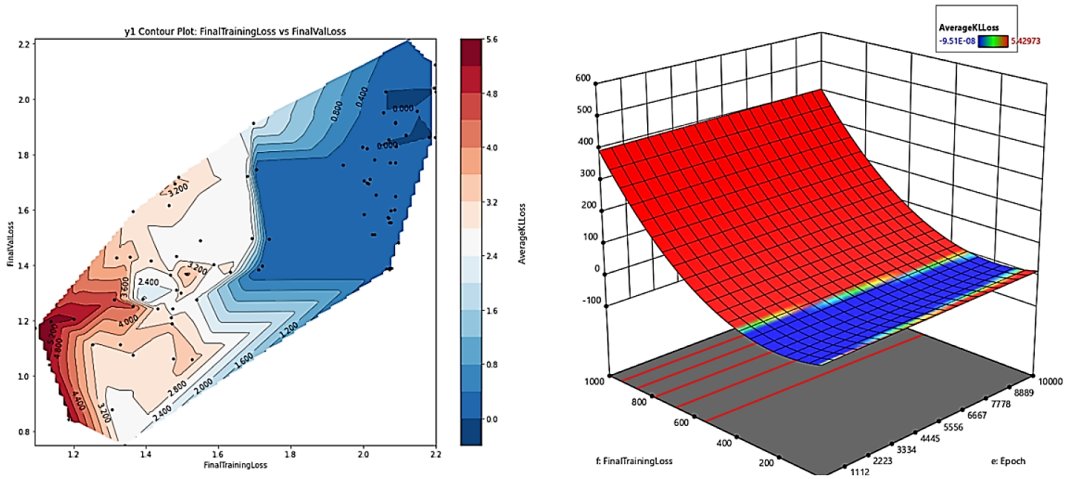


Figure 5. The contour and perspective plots that show the response surface for average KL loss over all mission flight data.y_1.

Table 3. Top Interaction Effects of average KL loss with respect to specific flight maneuvers (y_3)

Interaction Effect	Adjusted Mean Squares
1. Activation, Epochs, FinalTrainingLoss, FinalValLoss	8.011232E7
2. Epochs, FinalTrainingLoss, FinalValLoss	5.005503E7
3. Activation, Optimization, Epochs, FinalTrainingLoss, FinalValLoss	2.944778E7
4. Layer, Activation, Epochs, FinalTrainingLoss, FinalValLoss	1.673118E7
5. Optimization, Epochs, FinalTrainingLoss, FinalValLoss	1.654618E7

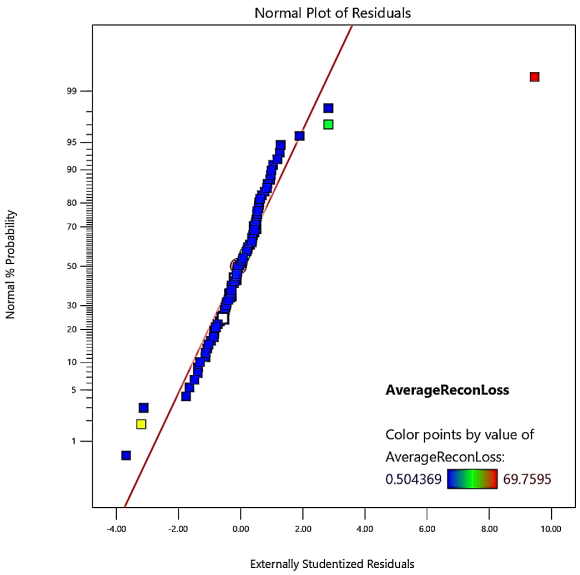


Figure 6. The residuals plots for the average reconstruction loss of all mission observations (y_2) model are largely normal, with one outlier.

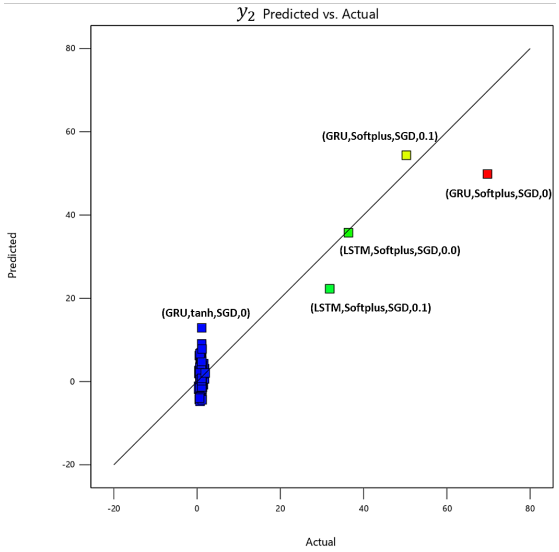


Figure 7. Predicted model for average reconstruction loss of all mission observations (y_2) from Equation 25.

Response y_4: Average Reconstruction Loss for Specific Maneuvers

The final direct measurement of our CVAE to replicate the T-2 data is y_4. The following model, resulting from stepwise regression, required no correction due to the effect hierarchy principal:

$$\begin{aligned} y_4 \sim & x_1^2 + x_2^2 + x_4^2 + x_5^2 + x_7^2 \\ & + x_1x_4 + x_1x_7 + x_2x_5 + x_3x_5 \\ & + x_3x_7 + x_4x_5 + x_4x_7 + x_5x_7 \\ & + x_1 + x_2 + x_3 + x_4 + x_5 + x_7 \end{aligned} \tag{27}$$

The AIC (578.245) was only slightly higher than that of the model for y_2. Like y_2, x_6 (the final CVAE training loss) is

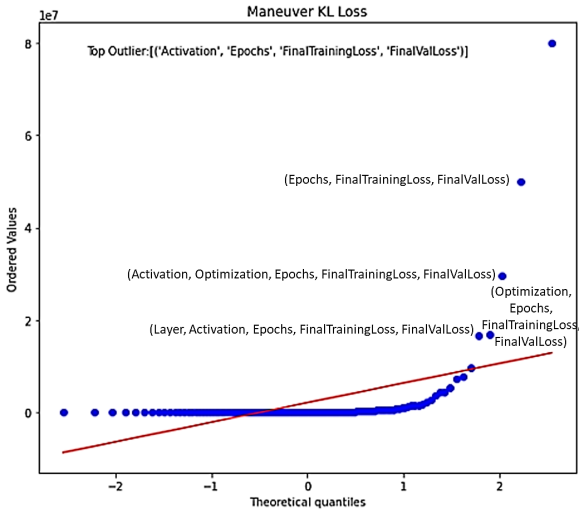


Figure 8. Quantile-Quantile plot for average KL loss with respect to specific flight maneuvers (y_3) with top 5 interaction effects highlighted.

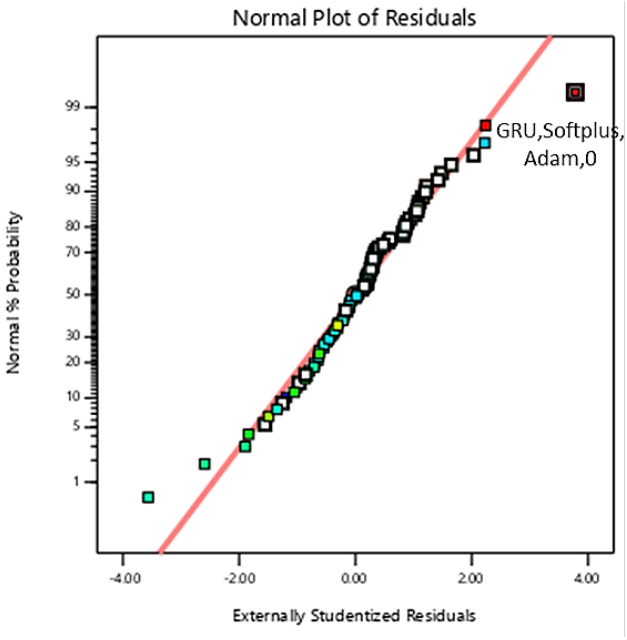


Figure 9. Normal vs Residuals plot for y_5; Outlier experiments from this model used GRU layers with softsign and softplus activation functions.

also excluded as a factor from our final model. Layer (x_1) and Dropout (x_4) have additional quadratic effects that were not seen for y_2. The same data point that was an outlier in the residual plot for y_2 was also an outlier for y_4, causing all of the same effects. That includes the y_4 stepwise regression model yielding a much lower adjusted R^2 (0.7422) and a negative predicted R^2 (-0.397) compare to its full quadratic model. Likewise, adding the (x_2x_3) interaction term back in had the same effect, increasing the adjusted R^2 to 0.924 with high adequate precision.

Response y_5: LOC/Operational Control Area of Intersection

The response variable y_5 describes the extent to which our CVAE models can be used to distinguish between LOC and operational control observations. The less overlap that the reconstruction probabilities for LOC observations have with operational control observations on our test data, the more reliable the CVAE should be for other missions. The default full model that produced the highest adjusted R^2 (with positive predicted R^2) for this outcome was linear. However, stepwise regression from a quadratic model produced a model with a lower AICc [69] and higher adjusted R^2:

$$\begin{aligned} y_5 \sim x_6^2 + x_3x_5 + x_3x_6 + x_4x_5 + x_4x_6 \\ +x_3 + x_4 + x_5 + x_6 \end{aligned} \tag{28}$$

This resulted in a higher adjusted R^2 than the original model (0.7777). Our residuals (shown in Figure 9) indicate a significant error for the y_5 model in predicting the area of intersection. Further analysis indicated that the model was thrown off by experiments that used a combination of GRU layers and softsign/softplus activation functions. However, adding the x_1x_2 interaction effect to the model did not improve its performance. Higher-order interactions would likely be necessary to improve the model even further.

Recall that y_5 should be low for the overall application of detecting LOC. The box plots in Figure 10 visualize the effects of our factors on the y_5 measurement. It appears that CVAEs with LSTM layers have a narrower distribution. Those implemented with softsign and softplus activation functions actually have lower y_5 measurements than other activations, while those with SELU activation functions are higher.

Response y_6: Envelope Change/No Change Area of Intersection

The full quadratic model for y_6 has significantly lower R^2 compared to the other outcomes. After a log transform, we applied stepwise regression from a full quadratic model to select the following model:

$$\begin{aligned} \log(y_6 + 1) \sim x_6^2 + x_7^2 \\ +x_1x_3 + x_1x_5 + x_1x_6 + x_1x_7 \\ +x_2^6 + x_4x_5 + x_4x_6 + x_5x_6 + x_5x_7 + x_6x_7 \\ +x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 \end{aligned} \tag{29}$$

This new model is a much better fit:

- R^2: 0.9692
- Adjusted R^2: 0.9498
- Predicted R^2: 0.8233

By investigating the plots for the interaction terms, the Layer of the CVAE (x_1) influences the outcome to a large extent. Figure 11 shows the layers plotted with respect to y_6. LSTM CVAE data points appear to be more widely distributed. But the intersection of latent space representations when the vehicle experiences a state change and when no state change occurs is generally smaller than that of GRUs. However, the variance of y_6 for LSTMs are an order of magnitude larger than that of GRU, which makes GRUs more reliable in this context.

Response y_7: Balanced Accuracy for LOC Detection

Next, we model the balanced accuracy in determining LOC. The full linear model has a very poor adjusted R^2. The following reduced quadratic model was produced through

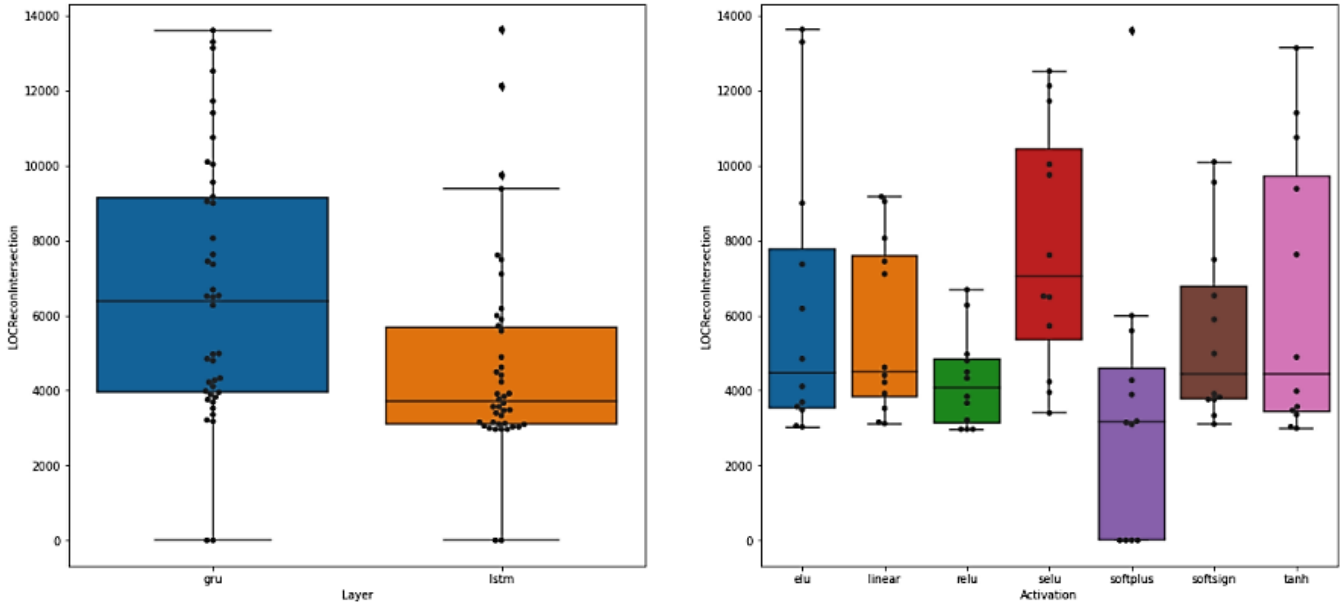


Figure 10. **Box plots of the distribution of y_5 with respect to Layer and Activation Functions**

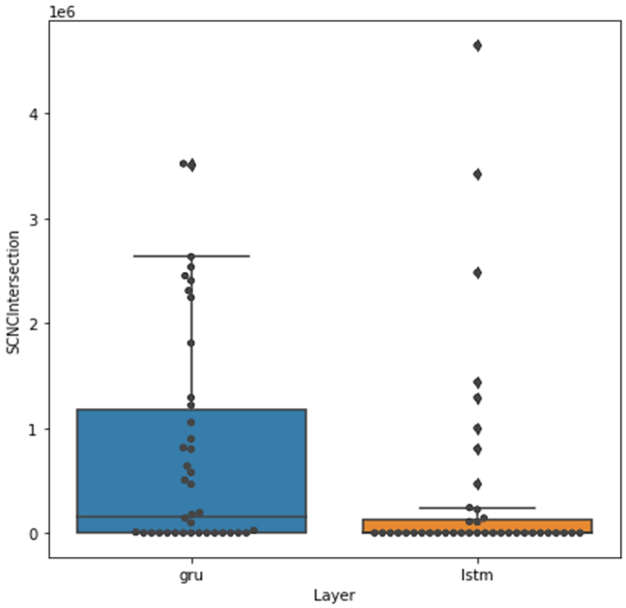


Figure 11. **Box plot of the CVAE layer (x_1) with respect to y_6**

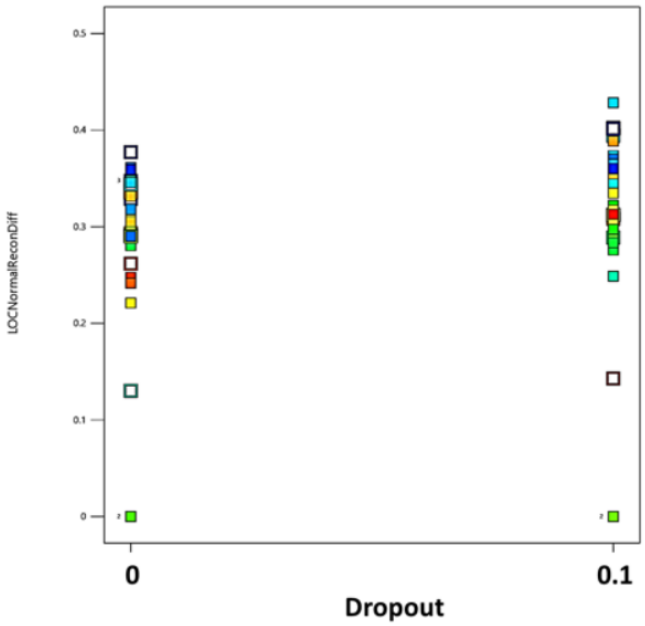


Figure 12. **Scatter plot showing the balanced accuracy (y_7) of predicting LOC/Operational Control compared to the dropout of intermediate layers.**

stepwise regression:

$$y_7 \sim x_4x_5 + x_5x_7 + x_6x_7 + x_4 + x_5 + x_6 + x_7 \quad (30)$$

The Adjusted R^2 was 0.4217 and was one of the few that we found with a positive Predicted R^2. This model indicates that the dropout (Figure 12) and loss values of CVAE training (Figures 13,18,19) had the most significant effects on balanced accuracy. A dropout of 0.1 appears to be more reliable in having a balanced accuracy above 0.7. Most CVAE experiments require between 2000 and 3000 training epochs (with these specific datasets) to achieve a balanced accuracy above 0.7. More experiments need to be done to build a more robust model.

Response y_8: Difference in Average Reconstruction Probability (Normal-LOC) observations

The full quadratic model of the average reconstruction probability during the LOC has an extremely high R^2 and adjusted R^2. Stepwise regression results in the following model, with AICc of 210.13 and adjusted R^2 of 1.0:

$$\frac{1}{y_8 + 0.001} \sim x_2x_3 + x_3x_7 + x_2 + x_3 + x_7 \quad (31)$$

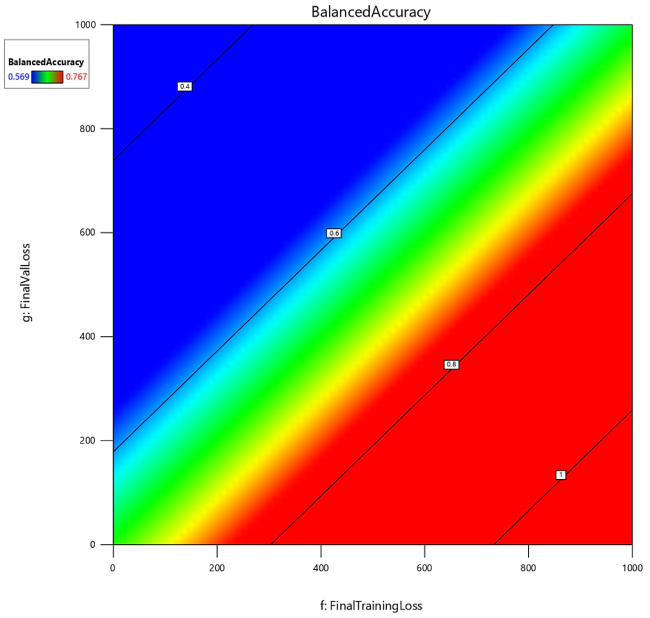


Figure 13. Contour plot of balanced accuracy (y_7) of predicting LOC/Operational Control: Training Loss vs Validation Loss

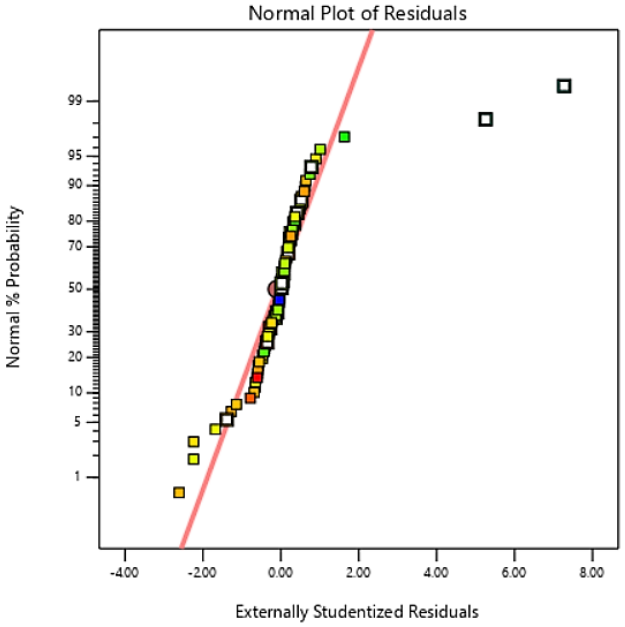


Figure 14. Residuals plot for y_8 model described by Equation 31.

The model is thrown off by two points, as shown in the residuals plot of Figure 14. These two experiments that are on the extremes of the residuals both were experiments in which the neural network hidden layers were constructed using softplus activation functions. The 3D surface plot in Figure 15 shows that CVAEs with softsign and softplus activation yield lower than average difference between LOC observations for Adam and SGD optimization, while those trained with Adadelta optimization yield the largest responses.

Response y_9: Difference in Average Jenson-Shannon distance (Enter/Exiting Envelope - Operating in Envelope)

Finally, for y_9, stepwise regression found the following reduced quadratic relationship with an adjusted R^2 of 0.8961:

$$\begin{aligned} \ln\left(y_9 + 0.001\right) \sim x_6^2 + x_7^2 \\ +x_1x_3 + x_1x_6 + x_3x_6 + x_4x_6 + x_6x_7 \\ +x_3x_6 + x_3x_7 + x_1 + x_3 + x_4 + x_6 + x_7 \end{aligned} \tag{32}$$

Again, our analysis of residuals plot (Figure 16) showed that experiments with softsign and softplus activation at extreme ends of the distribution. We examined the interactions highlighted in this model and discovered the dramatic differences in y_9 for different RNN layer types (x_1) and optimization algorithms (x_3). These are highlighted in the box plots in Figure 17.

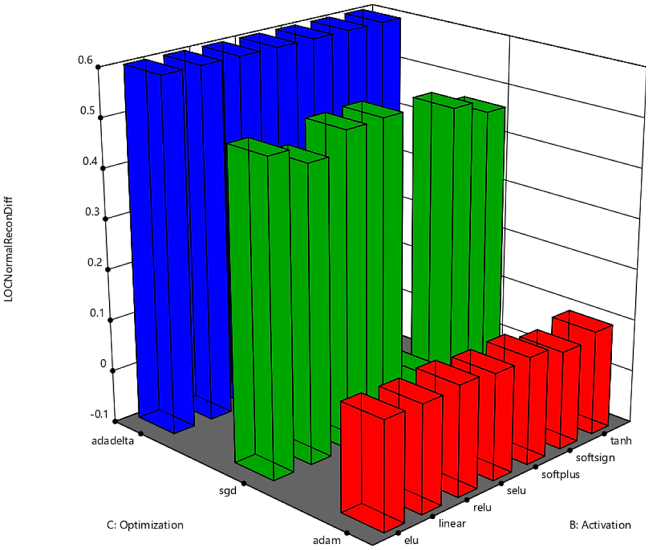


Figure 15. Surface plot of y_8: x_2x_3 Interaction

6. Conclusions

We used a DOE approach to characterize the effects of architectural choices, parameters, and training performance on applying conditional variational autoencoder models for inferring belief states in the flight environment. The conditioning vector for the CVAE is a vector of 1s and 0s, indicating whether the vehicle is operating inside (1) or outside (0) of a particular envelope. We monitored changes in reconstruction probability to detect loss-of-control observations and changes in the probability distributions encoded in the latent space to detect shifts in or out of a flight envelope. We performed a DOE analysis to serve as a template for conducting this approach when applying learning models to sensors that measure the dynamics of a flight vehicle. This analysis showed how we could vet different neural network architectures and their utility (responses) for a specific application, along with a quantifiable range of certainty about that utility. For details about this specific application, detecting of loss-of-control and envelope limits using CVAEs, please see our previous work [19].

This methodology proved successful in helping to analyze key factors in constructing the neural network. We will take more factors, such as layer depth and width, weight initialization, learning rates, data augmentation (such as adding Gaussian noise), and batch sizes into account in future

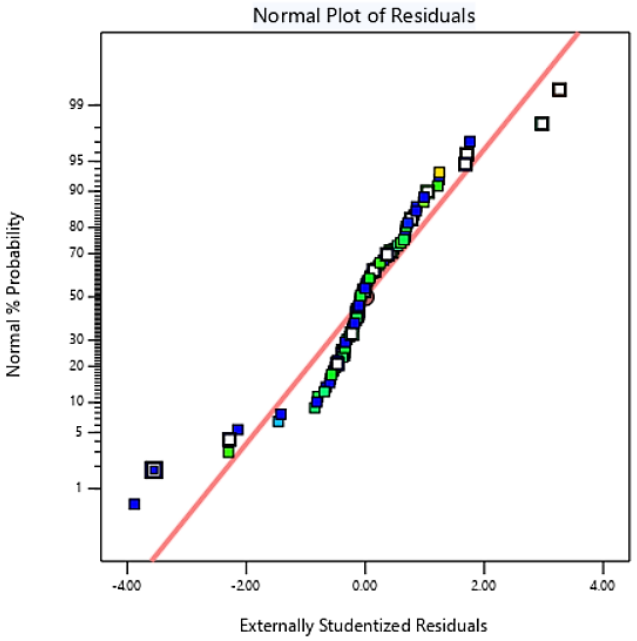


Figure 16. Residuals plot for difference between average JS-distance when pushing an envelope and when operating in an envelope(y_9).

studies. While we could not determine the most effective combination of factors (with high balanced accuracy) to perform LOC detection, our empirical data indicates that the training performance (length of epochs, training loss, and validation loss) has significant impacts on this application, as expected. Training for the CVAE is necessary only for 2000-3000 epochs to achieve sufficient balanced accuracy in detecting that the aircraft is in a LOC state. Based on our analysis, the following are recommendations for using a CVAE for distinguishing LOC from operational control observations:

- Activation Function: elu, relu, selu, tanh
- Optimization Function: Adadelta
- Layer Type: LSTM (GRU is not significantly worse)
- Dropout: 0.1 performed better for this application, but not significantly

However, the following are recommendations for using a CVAE to determine that the vehicle is approaching the boundaries of an envelope (and quantifying probability of approach):

- Activation Function: elu, selu
- Optimization Function: Adadelta
- Layer Type: GRU
- Dropout: 0.1 performed better for this application, but not significantly

In the choice between GRUs and LSTMs, GRUs have been more suitable for our experiments. Dropout does a good job of capturing features without co-adaptation. Future studies should conduct more experiments than were conducted in this paper to account for randomization in the initialization of neural network weights during training. In future LOC studies, we plan to use the recommended configurations for the CVAE to train multiple neural networks, with significantly smaller input spaces, based on statistical relationships that we identify among input data. The dataset that we used here provide 140 inputs, and we selected 22 specific percepts based on subject-matter expertise. We hope to use an algorithmic process to produce fast LOC and envelope change detectors by reasoning about the results of a set of smaller CVAE models.

APPENDIX

Figure 18 summarizes the training performance of all CVAEs from our experimental studies. Table 4 shows a chart of average responses for each main effect of the experiments in this study. Figure 20 shows a histogram for all experimental outcomes in this study.

ACKNOWLEDGMENTS

This research was supported by the NASA Aeronautics Research Mission Directorate (ARMD), Transformative Tools and Technologies (TTT) project, under the Autonomous Systems / Intelligent Contingency Management subproject. The T-2 flight data used was generated by the NASA Langley Research Center AirSTAR team under the NASA Aviation Safety Program, Vehicle Systems Safety Technologies (VSST) project. Additional thanks to the NASA Langley Research Center MidRange Computer User's Group for supporting experimental processing and analysis on the K-cluster.

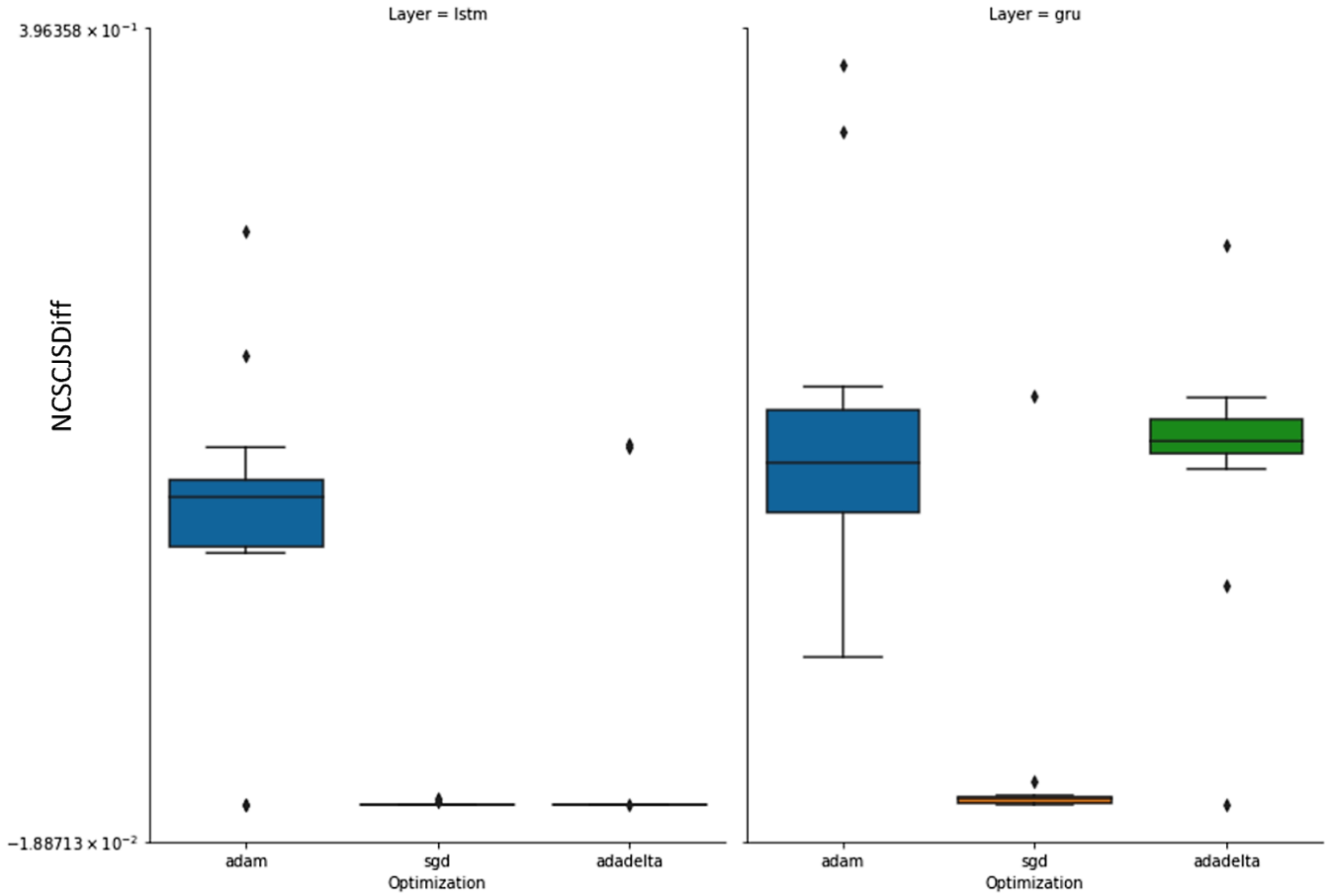


Figure 17. Box plots for y_9: Layer (x_1) vs Optimization (x_3) Function

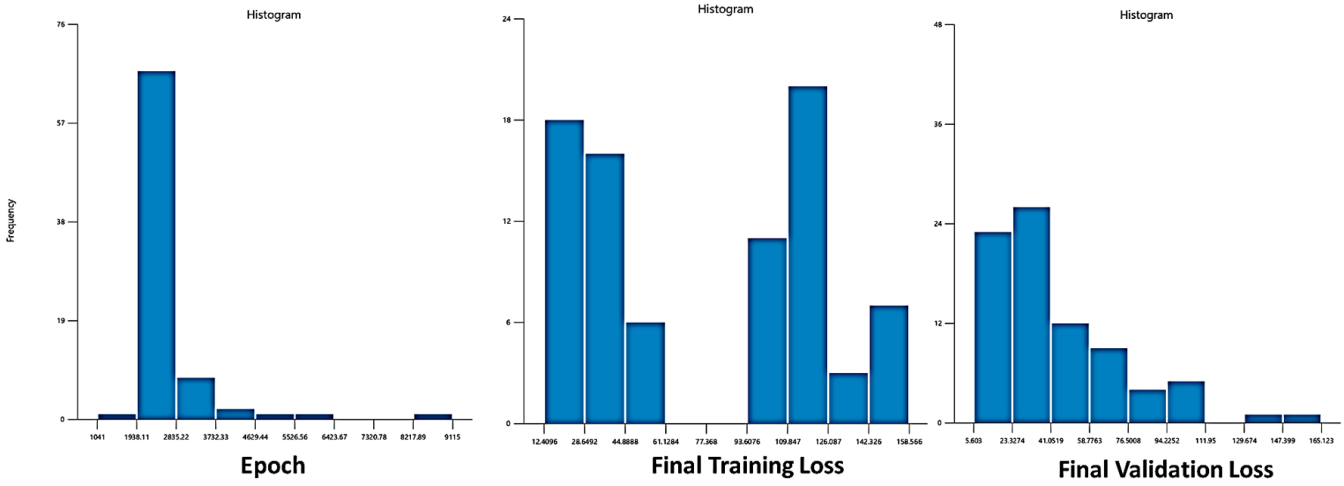


Figure 18. Histograms for # Epochs, Training Loss, and Validation Loss from our experimental runs.

Table 4. Average outcomes for the main effects of 84 experiments.

Layer	AverageKLLoss	AverageReconLoss	ManeuverKLLoss	ManeuverReconLoss	LOCReconIntersection	SCNCIntersection	BalancedAccuracy	ReconProbDiff	JSDiff
gru	2.057126	3.755259	2.159451	4.407554	6646.594659	769482.188413	0.712023	0.302079	0.115365
lstm	0.925567	2.630835	0.968229	3.196932	4656.622465	389900.829617	0.713771	0.302537	0.054143

Activation	AverageKLLoss	AverageReconLoss	ManeuverKLLoss	ManeuverReconLoss	LOCReconIntersection	SCNCIntersection	BalancedAccuracy	ReconProbDiff	JSDiff
elu	1.943108	0.906901	2.064702	1.499071	6269.562809	9.648408e+05	0.731546	0.330900	0.091513
linear	1.754307	0.935008	1.754860	1.526798	5647.370276	8.806088e+05	0.719780	0.329961	0.086577
relu	0.211634	1.034071	0.237273	1.565826	4260.717151	2.003483e+03	0.730877	0.330454	0.031503
selu	2.552423	0.790521	2.635373	1.444083	7838.254286	1.260535e+06	0.722171	0.331534	0.135704
softplus	1.314638	16.370333	1.444055	17.011838	3563.482221	4.269162e+05	0.681875	0.213505	0.082940
softsign	1.253442	1.264398	1.317819	1.894943	5518.256986	1.043452e+05	0.690402	0.271768	0.085164
tanh	1.409874	1.050095	1.492798	1.673141	6463.616205	4.185916e+05	0.713627	0.308033	0.079876

Optimization	AverageKLLoss	AverageReconLoss	ManeuverKLLoss	ManeuverReconLoss	LOCReconIntersection	SCNCIntersection	BalancedAccuracy	ReconProbDiff	JSDiff
adadelta	1.785809	0.988274	1.857127	1.585921	6042.346115	821040.663078	0.711858	0.306989	0.088327
adam	2.592785	0.912426	2.737745	1.603903	7397.170390	914486.677797	0.702996	0.346108	0.156810
sgd	0.095447	7.678440	0.096646	8.216904	3515.309181	3547.186169	0.723836	0.253826	0.009125

Dropout	AverageKLLoss	AverageReconLoss	ManeuverKLLoss	ManeuverReconLoss	LOCReconIntersection	SCNCIntersection	BalancedAccuracy	ReconProbDiff	JSDiff
0.0	1.605063	3.518387	1.662741	4.153910	6103.692949	650885.76721	0.705411	0.290767	0.074374
0.1	1.377631	2.867707	1.464939	3.450575	5199.524175	508497.25082	0.720383	0.313848	0.095134

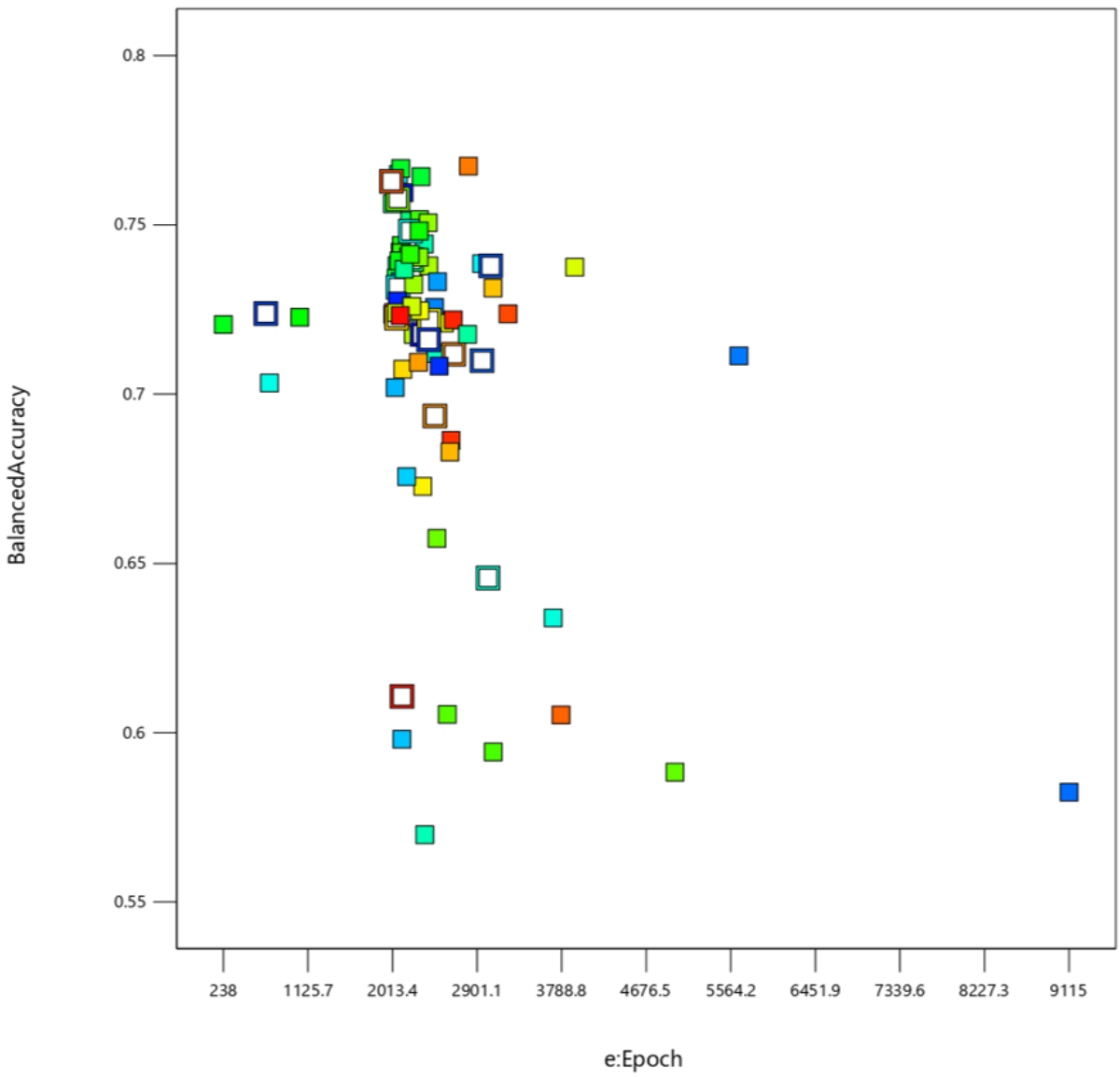


Figure 19. **Scatter plot showing the balanced accuracy of predicting LOC/Operational Control compared to the number of epochs completed in training.**

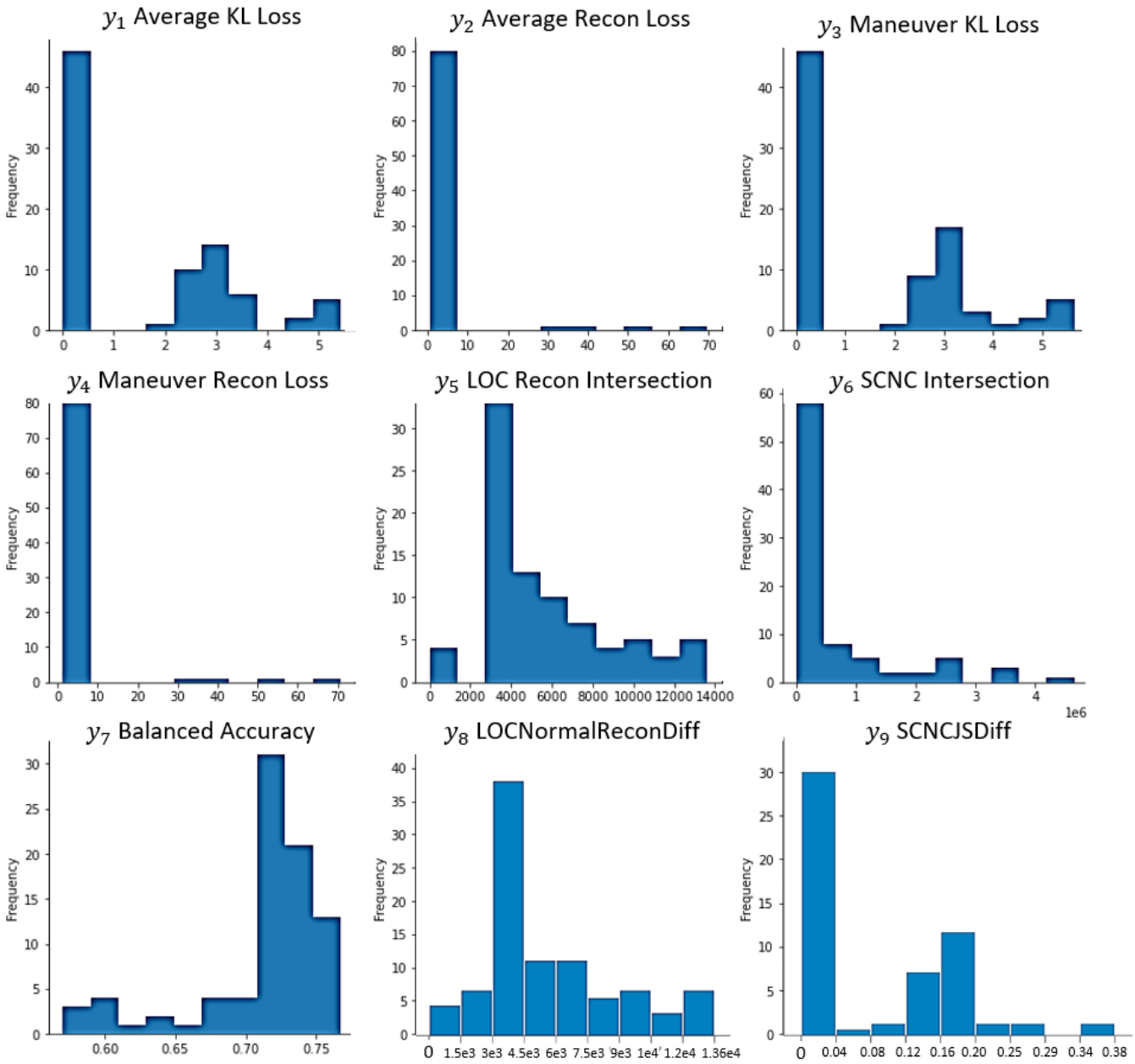


Figure 20. Histograms for each of the responses from the DOE results. These histograms illustrate the range and distribution of outcomes from each experiment.

REFERENCES

[1] H. Lee, G. Li, A. Rai, and A. Chattopadhyay, "Real-time anomaly detection framework using a support vector regression for the safety monitoring of commercial aircraft," *Advanced Engineering Informatics*, vol. 44, p. 101071, 2020.

[2] H. Lee, H. J. Lim, P. Parker, and A. Chattopadhyay, "Precursor Detection of Aircraft Loss of Control Inflight (LOC-I) and Prediction of Future Trajectory," in *AIAA AVIATION 2020 FORUM*, 2020, p. 2879.

[3] H. Lee, H. J. Lim, and A. Chattopadhyay, "Data-driven system health monitoring technique using autoencoder for the safety management of commercial aircraft," *Neural Computing and Applications*, pp. 1–16, 2020.

[4] H. Moncayo, M. G. Perhinschi, and J. Davis, "Artificial-immune-system-based aircraft failure evaluation over extended flight envelope," *Journal of guidance, control, and dynamics*, vol. 34, no. 4, pp. 989–1001, 2011.

[5] S. Das, B. L. Matthews, A. N. Srivastava, and N. C. Oza, "Multiple Kernel Learning for Heterogeneous Anomaly Detection: Algorithm and Aviation Safety Case Study," in *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '10. New York, NY, USA: Association for Computing Machinery, 2010, p. 47–56. [Online]. Available: https://doi.org/10.1145/1835804.1835813

[6] C. Belcastro and J. Foster, "Aircraft loss-of-control accident analysis," in *AIAA Guidance, Navigation, and Control Conference*, 2010, p. 8004.

[7] C. M. Belcastro, "Aircraft loss of control: Analysis and requirements for future safety-critical systems and their validation," in *2011 8th Asian Control Conference (ASCC)*. IEEE, 2011, pp. 399–406.

[8] J. Wilborn and J. Foster, "Defining commercial transport loss-of-control: A quantitative approach," in *AIAA atmospheric flight mechanics conference and exhibit*, 2004, p. 4811.

[9] G. Rohith, "An investigation into aircraft loss of control and recovery solutions," *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, vol. 233, no. 12, pp. 4509–4522, 2019.

[10] J. P. C. Macedo, J. H. Bidinotto, and M. Bromfield, "Loss of Control in Flight: comparing qualitative pilot opinion with quantitative flight data," in *AIAA AVIATION 2020 FORUM*, 2020, p. 2911.

[11] K. Sohn, H. Lee, and X. Yan, "Learning structured output representation using deep conditional generative models," in *Advances in neural information processing systems*, 2015, pp. 3483–3491.

[12] S. Suh, D. H. Chae, H.-G. Kang, and S. Choi, "Echo-state conditional variational autoencoder for anomaly detection," in *2016 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2016, pp. 1015–1022.

[13] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, "Conditional variational autoencoder for prediction and feature recovery applied to intrusion detection in IOT," *Sensors*, vol. 17, no. 9, p. 1967, 2017.

[14] A. A. Pol, V. Berger, C. Germain, G. Cerminara, and M. Pierini, "Anomaly Detection With Conditional Variational Autoencoders," in *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*. IEEE, 2019, pp. 1651–1657.

[15] M. Hwasser, D. Kragic, and R. Antonova, "Variational Auto-Regularized Alignment for Sim-to-Real Control," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 2732–2738.

[16] X. Cheng and K. Jiang, "Crustal model in eastern Qinghai-Tibet plateau and western Yangtze craton based on conditional variational autoencoder," *Physics of the Earth and Planetary Interiors*, p. 106584, 2020.

[17] R. Jiao, K. Peng, and J. Dong, "Remaining Useful Life Prediction of Lithium-Ion Batteries Based on Conditional Variational Autoencoders-Particle Filter," *IEEE Transactions on Instrumentation and Measurement*, 2020.

[18] Y. Tang, K. Kojima, T. Koike-Akino, Y. Wang, P. Wu, M. Tahersima, D. Jha, K. Parsons, and M. Qi, "Generative deep learning model for a multi-level nano-optic broadband power splitter," in *2020 Optical Fiber Communications Conference and Exhibition (OFC)*. IEEE, 2020, pp. 1–3.

[19] N. H. Campbell, J. A. Grauer, and I. M. Gregory, *Loss of Control Detection for Commercial Transport Aircraft Using Conditional Variational Autoencoders*. AIAA SciTech 2021 Forum, January 2021. [Online]. Available: https://arc.aiaa.org/doi/abs/10.2514/6.2021-0778

[20] NIST/SEMATECH, "NIST/SEMATECH e-Handbook of Statistical Methods," 2020. [Online]. Available: http://www.itl.nist.gov/div898/handbook/

[21] J. Shin, "The NASA Aviation Safety Program: Overview," in *Turbo Expo: Power for Land, Sea, and Air*, vol. 78545. American Society of Mechanical Engineers, 2000, p. V001T01A024.

[22] S. Jacobson, "Aircraft loss of control causal factors and mitigation challenges," in *AIAA Guidance, navigation, and control conference*, p. 8007.

[23] P. W. John, *Statistical design and analysis of experiments*. SIAM, 1998.

[24] P. Murphy and A. Sabharwal, "Design, implementation, and characterization of a cooperative communications system," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 6, pp. 2534–2544, 2011.

[25] P. M. Rothhaar, P. C. Murphy, B. J. Bacon, I. M. Gregory, J. A. Grauer, R. C. Busan, and M. A. Croom, "NASA Langley distributed propulsion VTOL tiltwing aircraft testing, modeling, simulation, control, and flight test development," in *14th AIAA aviation technology, integration, and operations conference*, 2014, p. 2999.

[26] P. C. Murphy and D. Landman, "Experiment design for complex VTOL aircraft with distributed propulsion and tilt wing," in *AIAA Atmospheric Flight Mechanics Conference*, 2015, p. 0017.

[27] J. F. Khaw, B. Lim, and L. E. Lim, "Optimal design of neural networks using the Taguchi method," *Neurocomputing*, vol. 7, no. 3, pp. 225–245, 1995.

[28] M. Packianather, P. Drake, and H. Rowlands, "Optimizing the parameters of multilayered feedforward neural networks through Taguchi design of experiments," *Quality and reliability engineering international*, vol. 16, no. 6, pp. 461–473, 2000.

[29] Y.-S. Kim and B.-J. Yum, "Robust design of multi-

layer feedforward neural networks: an experimental approach," *Engineering Applications of Artificial Intelligence*, vol. 17, no. 3, pp. 249–263, 2004.

[30] W. Laosiritaworn and N. Chotchaithanakorn, "Artificial neural networks parameters optimization with design of experiments: An application in ferromagnetic materials modeling," *Chiang Mai J. Sci*, vol. 36, no. 1, pp. 83–91, 2009.

[31] F. S. Lasheras, J. V. Vilán, P. G. Nieto, and J. del Coz Díaz, "The use of design of experiments to improve a neural network model in order to predict the thickness of the chromium layer in a hard chromium plating process," *Mathematical and Computer Modelling*, vol. 52, no. 7-8, pp. 1169–1176, 2010.

[32] T. Nazghelichi, M. Aghbashlo, and M. H. Kianmehr, "Optimization of an artificial neural network topology using coupled response surface methodology and genetic algorithm for fluidized bed drying," *Computers and electronics in agriculture*, vol. 75, no. 1, pp. 84–91, 2011.

[33] F. J. Pontes, G. Amorim, P. P. Balestrassi, A. Paiva, and J. R. Ferreira, "Design of experiments and focused grid search for neural network parameter optimization," *Neurocomputing*, vol. 186, pp. 22–34, 2016.

[34] A. M. Karim, M. S. Güzel, M. R. Tolun, H. Kaya, and F. V. Çelebi, "A New Generalized Deep Learning Framework Combining Sparse Autoencoder and Taguchi Method for Novel Data Classification and Processing," *Mathematical Problems in Engineering*, vol. 2018, 2018.

[35] A. Glushkovsky, "AI Giving Back to Statistics? Discovery of the Coordinate System of Univariate Distributions by Beta Variational Autoencoder," *arXiv preprint arXiv:2004.02687*, 2020.

[36] H. Jin, Q. Song, and X. Hu, "Auto-keras: An efficient neural architecture search system," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 1946–1956.

[37] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, 2016, pp. 265–283.

[38] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 2623–2631.

[39] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *The Journal of Machine Learning Research*, vol. 13, no. 1, pp. 281–305, 2012.

[40] J. Bergstra, D. Yamins, and D. Cox, "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures," in *International conference on machine learning*, 2013, pp. 115–123.

[41] J. Snoek, H. Larochelle, and R. P. Adams, "Practical Bayesian optimization of machine learning algorithms," in *Advances in neural information processing systems*, 2012, pp. 2951–2959.

[42] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, "Hyperband: A novel bandit-based approach to hyperparameter optimization," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 6765–6816, 2017.

[43] S. Paul, V. Kurin, and S. Whiteson, "Fast Efficient Hyperparameter Tuning for Policy Gradient Methods," in *Advances in Neural Information Processing Systems*, 2019, pp. 4616–4626.

[44] K. B. Petersen and M. S. Pedersen, "The matrix cookbook (version: November 15, 2012)," 2012.

[45] A. Murch, "A flight control system architecture for the NASA AirSTAR flight test infrastructure," in *AIAA Guidance, Navigation and Control Conference and Exhibit*, p. 6990.

[46] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555*, 2014.

[47] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to Forget: Continual Prediction with LSTM," *Neural Computation*, vol. 12, no. 10, pp. 2451–2471, 2000.

[48] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *arXiv preprint arXiv:1511.07289*, 2015.

[49] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain." *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[50] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *ICML*, 2010.

[51] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-normalizing neural networks," in *Advances in neural information processing systems*, 2017, pp. 971–980.

[52] Hao Zheng, Zhanlei Yang, Wenju Liu, Jizhong Liang, and Yanpeng Li, "Improving deep neural networks using softplus units," in *2015 International Joint Conference on Neural Networks (IJCNN)*, 2015, pp. 1–4.

[53] J. Bergstra, G. Desjardins, P. Lamblin, and Y. Bengio, "Quadratic polynomials learn better image features (Technical Report 1337)," *Département d'Informatique et de Recherche Opérationnelle, Université de Montréal*, 2009.

[54] E. W. Weisstein, "Hyperbolic Functions," *https://mathworld. wolfram. com/*, 2003.

[55] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[56] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning."

[57] M. D. Zeiler, "Adadelta: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[58] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[59] C. Jernbäcker, "Unsupervised real-time anomaly detection on streaming data for large-scale application deployments," 2019.

[60] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[61] S. v. d. Walt, S. C. Colbert, and G. Varoquaux, "The NumPy array: a structure for efficient numerical computation," *Computing in science & engineering*, vol. 13, no. 2, pp. 22–30, 2011.

[62] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright *et al.*, "SciPy 1.0: fundamental algorithms for scientific computing in Python," *Nature methods*, vol. 17, no. 3, pp. 261–272, 2020.

[63] S. Seabold and J. Perktold, "Statsmodels: Econometric and statistical modeling with python," in *Proceedings of the 9th Python in Science Conference*, vol. 57. Austin, TX, 2010, p. 61.

[64] M. Waskom and the seaborn development team, "mwaskom/seaborn," Sep. 2020. [Online]. Available: https://doi.org/10.5281/zenodo.592845

[65] Y. Sakamoto, M. Ishiguro, and G. Kitagawa, "Akaike information criterion statistics," *Dordrecht, The Netherlands: D. Reidel*, vol. 81, 1986.

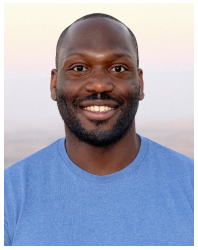
[66] X. Li, N. Sudarsanam, and D. D. Frey, "Regularities in data from factorial experiments," *Complexity*, vol. 11, no. 5, pp. 32–45, 2006.

[67] R. D. Cook, "Detection of influential observation in linear regression," *Technometrics*, vol. 19, no. 1, pp. 15–18, 1977.

[68] R. Lenth and M. R. Lenth, "Package 'lsmeans'," *The American Statistician*, vol. 34, no. 4, pp. 216–221, 2018.

[69] J. E. Cavanaugh, "Unifying the derivations for the Akaike and corrected Akaike information criteria," *Statistics & Probability Letters*, vol. 33, no. 2, pp. 201–208, 1997.

BIOGRAPHY



Newton Campbell *is a Computer Scientist specializing in artificial intelligence. Through SAIC, he currently serves as an Artificial Intelligence subject matter expert on the NASA Langley Research Center OCIO Data Science Team. There, he leads the development of several programs in urban air mobility, geomagnetism, virtual reality, and high-performance computing for Earth Sciences. Dr. Campbell completed his Ph.D. in Computer Science at Nova Southeastern University. He is a member of the Schusterman Foundation REALITY network, a Technology Fellow at American University Washington College of Law, and a US Young Leadership Delegate for the Australian-American Leadership Dialogue and the French-American Foundation.*

Jared Grauer *received his Ph.D. in Aerospace Engineering from the University of Maryland at College Park. For the past 10 years he has served as a research engineer with NASA Langley Research Center, where his research interests have focused on system identification, feedback control, and flight dynamics. He is an associate fellow of the AIAA and serves on the AIAA Atmospheric Flight Mechanics technical committee and the SAE/IEEE Aerospace Control and Guidance Systems Committee*

Irene Gregory *received the S.B. and M.S. in Aeronautics and Astronautics from the Massachusetts Institute of Technology and her Ph.D. in Control and Dynamic Systems from the California Institute of Technology. She is an Associate Fellow of the AIAA, a senior member of IEEE, serves on the AIAA GNC Technical Committee, Future Technology Directions Committee, IEEE Control Systems Society Technical Committee on Aerospace Control and on IFAC Aerospace Control Technical Committee. Dr. Gregory currently serves as a Senior Research Engineer with NASA Langley Research Center.*