Machine Learning with Applications 13 (2023) 100484

Contents lists available at ScienceDirect

Machine Learning with Applications

journal homepage: www.elsevier.com/locate/mlwa

Multi-module-based CVAE to predict HVCM faults in the SNS accelerator

Yasir Alanazi [a,*], Malachi Schram [a], Kishansingh Rajput [a], Steven Goldenberg [a], Lasitha Vidyaratne [a], Chris Pappas [b], Majdi I. Radaideh [c], Dan Lu [b], Pradeep Ramuhalli [b], Sarah Cousineau [b]

[a] Thomas Jefferson National Accelerator Facility, Newport News, VA 23606, USA
[b] Oak Ridge National Laboratory, Oak Ridge, TN 37830, USA
[c] Department of Nuclear Engineering and Radiological Sciences, The University of Michigan, Ann Arbor, MI 48109, USA

ARTICLE INFO

Keywords:
Anomaly detection
Variational Autoencoder
Spallation Neutron Source
Accelerators
High Voltage Converter Modulator

ABSTRACT

We present a multi-module framework based on Conditional Variational Autoencoder (CVAE) to detect anomalies in the power signals coming from multiple High Voltage Converter Modulators (HVCMs). We condition the model with the specific modulator type to capture different representations of the *normal* waveforms and to improve the sensitivity of the model to identify a specific type of fault when we have limited samples for a given module type. We studied several Artificial Neural Network (ANN) architectures for our CVAE model and evaluated the model performance by looking at their loss landscape for stability and generalization. Our results for the Spallation Neutron Source (SNS) experimental data show that the trained model generalizes well to detecting multiple fault types for several HVCM module types. The results of this study can be used to improve the HVCM reliability and overall SNS uptime.

1. Introduction

Anomalies or outliers in an engineering system can be caused by multiple reasons including mechanical and human errors. Detecting anomalies and identifying their fault types is critical for the system's normal operation and guides the maintenance to achieve proper functioning of the system. Anomaly detection (AD) has been well-studied across a wide spectrum of scientific applications. There have been several non-parametric statistical and visualization methods used to detect outliers, including Decision Trees (John, 1995), K-Nearest Neighbor (Dang, Ngan, & Liu, 2015), and whisker plot (Potter, 2006). Recently, with the advances in Machine Learning (ML), specifically deep learning, much attention has been devoted to detect anomalies using ML techniques. ML algorithms have shown significant improvements in detecting outliers for structured and unstructured data. A comprehensive survey on ML methods to detect anomalies can be found in Chalapathy and Chawla (2019), Chandola, Banerjee, and Kumar (2009). In a typical AD problem, there is usually a large data set of *normal* observations, and only a few *abnormal* observations leading to significant sample bias. Additionally, the limited available *abnormal* observations might not include all types of potential anomalies. As such, a common technique is to develop a ML model that encodes the salient features of the *normal* data into a reduced representation then decodes this representation back to the original data using an Autoencoder (AE) (Hinton & Zemel, 1993). For a well-behaved model, the decoded data should match the input data and give a small reconstruction error (the difference between the input and decoded data). The assumption for anomaly detection, is that anomalous data will not have the same salient features resulting in a larger reconstructed error, which allows us to set a threshold based on the application requirements to identify *abnormal* samples. Variational Autoencoder (VAE) (Kingma & Welling, 2013) has also been proposed by An and Cho (2015) to detect anomalies using the reconstruction probability as opposed of using the reconstruction error as in AEs. The anomaly score in the proposed approach is calculated in terms of a probability that is extracted by sampling from the mean and variance parameters generated from the probabilistic encoder. Using the reconstruction probability, one can capture not only the difference between the reconstruction and the input data but also the variability of the reconstruction by using the variance parameter of the latent variable. Conditional Variational Autoencoder (CVAE) (Sohn, Lee, & Yan, 2015) has also been proposed to detect anomalies for structured data to make various predictions for different input sample. AEs and VAEs have been widely used to detect anomalies in multiple scientific applications with different types of data, such as fraud detection (Adewumi & Akinyelu, 2017), medical

* Corresponding author.
E-mail addresses: alanazi@jlab.org (Y. Alanazi), schram@jlab.org (M. Schram), kishan@jlab.org (K. Rajput), sgolden@jlab.org (S. Goldenberg), lasithav@jlab.org (L. Vidyaratne), pappasgc@ornl.gov (C. Pappas), radaideh@umich.edu (M.I. Radaideh), lud1@ornl.gov (D. Lu), ramuhallip@ornl.gov (P. Ramuhalli), scousine@ornl.gov (S. Cousineau).

https://doi.org/10.1016/j.mlwa.2023.100484
Received 26 April 2023; Received in revised form 13 July 2023; Accepted 15 July 2023
Available online 20 July 2023

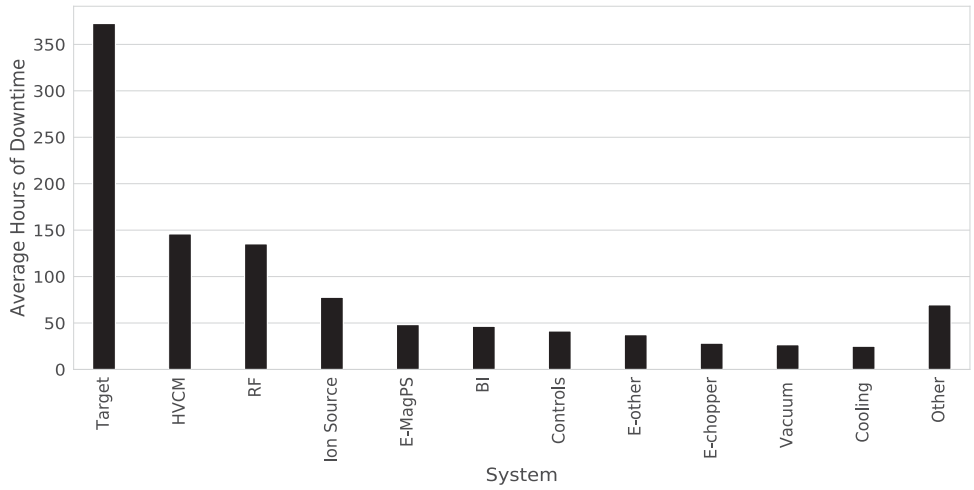

Fig. 1. SNS unscheduled downtime by system. On average, HVCM is the second leading source of downtime after Target. The average is calculated from fiscal year 2007 to 2021.

image analysis (Litjens et al., 2017), remote sensing (Ball, Anderson, & Chan, 2017), smart manufacturing (Alfeo, Cimino, Manco, Ritacco, & Vaglini, 2020), and Internet of Things (IoT) (Mohammadi, Al-Fuqaha, Sorour, & Guizani, 2017) where the data are images, and univariate or multivariate time series outlier detection (Kieu, Yang, Guo, & Jensen, 2019; Lu et al., 2017) where the data are sequences, and video anomaly detection (Xu, Ricci, Yan, Song, & Sebe, 2015) where the data are sequences of images. Based on the data types, the ANNs used in the AE and VAE architectures are different, e.g., Convolutional NNs (CNNs) are typically used for imaging data and Recurrent NNs (RNNs) are usually used for time series data.

In this paper, we present a study on how to detect anomalies in the High Voltage Converter Modulator (HVCM) (Reass et al., 2003) at the Spallation Neutron Source (SNS) facility (White, 2002) in order to reduce long downtimes. The HVCMs consist of multiple modules working cooperatively to produce high-quality neutron beams at the SNS. Therefore, it is critical to detect faults ahead of time to avoid long downtime. As shown in Fig. 1, on average the HVCM is the second largest source of downtime from fiscal year 2007 to 2021 at the SNS. While the scope of this paper is limited to the HVCM system, it is worth mentioning that there is concurrent work by other collaborators dedicated to reduce the SNS downtime caused by Target (Radaideh, Tran et al., 2022), which is the leading source of downtime at SNS. Previous studies using ML methods have been used to detect anomalies in the HVCMs, such as, (Pappas, Lu, Schram, & Vrabie, 2021; Radaideh, Pappas, Cousineau et al., 2022; Radaideh, Pappas, Walden et al., 2022). These studies showed promising results for detecting faults for a single module using a single waveform, however, the HVCMs consist of 15 modules with 14 sources of waveforms that were not considered in the previously developed ML model. In this study, we include all waveforms and HVCM module types. We found that different waveforms are sensitive to different fault types, and using a single waveform is only sufficient to detect anomalies coming from the corresponding source of waveform. For example, a fault in A-FLUX waveform (magnetic flux in the A-phase of the HVCM), might not be detected in MOD-V waveform (modulator voltage) and vice-versa. We also found that it is more efficient to incorporate all modules instead of using a single ML model for each module. This is because including all modules enforces the model to learn diverse representations of *normal* data, and hence, can generalize well to several anomalies.

In this study, we present our results using a multi-module Conditional Variational Autoencoder (CVAE) model to detect anomalies. The model was trained using all 14 source waveforms for each HVCM module and all 15 HVCM modules. By using a CVAE and all 15 HVCM modules we increase the overall number of samples which can improve the model performance for modules with limited samples, eliminate the need of retraining a model for each module individually, and allow us to develop a well-performed AD model that can generalize well to various anomalies.

We evaluate our trained models by visualizing their loss landscapes using filter normalization technique (Li, Xu, Taylor, Studer, & Goldstein, 2018) to assist in hyper-parameter optimization (HPO) and model selection to produce well-performed predictions. We present a side-by-side loss landscape comparisons between the proposed multi-module with a single-module that is trained individually for each HVCM module. The results of this analysis indicate that the multi-module approach can learn from multiple modules and produces a convex-like loss surface for all HVCM files, while the single-module model has a chaotic loss surface for some HVCMs using a fixed set of parameters and ANN architectures.

The rest of this document is organized as follows. In Section 2, we provide an overview of the HVCM. Section 3 reviews the previous work to detect anomalies in the HVCMs at the SNS. Section 4 gives a brief background of AE, VAE, and CVAE. The experimental data is presented in Section 5. Section 6 demonstrates the results and the loss landscape analysis. The conclusion is finally presented in Section 7.

2. High Voltage Converter Modulator (HVCM)

The SNS facility, which is the world's highest power pulsed spallation source, consists of a linear accelerator (LINAC), accumulator ring and transfer line to deliver protons to a target used to produce neutron beams. The SNS LINAC, as shown in Fig. 2, is comprised of several different types of radio frequency (RF) structures used to accelerate the beam from low energy to high energy, starting with the normal conducting (NL) cavities and ending with superconducting (SCL) ones. At the lowest energy, a radio frequency quadrupole (RFQ) is used followed by six drift tube LINAC (DTL) cavities. The remainder of the NL Section uses four coupled cavity (CCL) structures for acceleration, while the higher energy Sections use 81 superconducting RF cavities. Each of the cavities is powered by a high-power RF amplifier, or klystron (Caryotakis, 2004). Since different types of RF cavities at the SNS operate at different frequencies and require different RF power, they are powered by different types of klystrons which require different cathode voltages and appear as different impedances to the high voltage power supply. SNS uses pulsed klystrons, and the HVCMs are the pulsed power supplies used to drive the klystron's cathodes. The HVCMs were designed to operate at approximately 1 MW of average power, so each of the HVCM modules powers differing numbers of klystrons in order to use the minimum number of HVCMs. This results in slightly different designs of the HVCMs in order to accommodate the different voltages and load impedances for the klystrons. The types of HVCMs are those for the RFQ and first two DTL Sections, the other DTL klystrons, the CCL Section and the SCL Section.

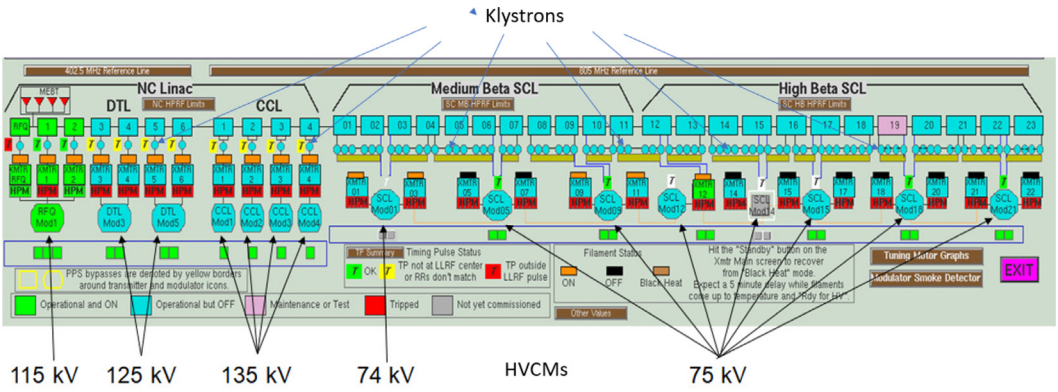

Fig. 2. Layout of the RF systems at the SNS showing the HVCMs.

3. Previous work

Although the anomaly detection efforts for the HVCMs are limited, there are a few recent studies that demonstrated the promise of using ML for anomaly detection in the HVCM systems. Pappas et al. (2021) conducted anomaly detection on one of the SNS HVCMs - SCL09 (superconducting LINAC) - using discrete cosine transform. Given the work was published in the early phase of the project, very limited data was available, as the authors have used 83 waveform samples (all are magnetic flux in the B-phase). Nevertheless, the authors were able to detect 11 of the 16 fault events. Radaideh, Pappas, Walden et al. (2022) extended the effort by applying deep learning RNNs on the RFQ module (radio-frequency quadrupole), the module that had a significant number of failures in the SNS. The authors developed long short-term memory (LSTM), gated recurrent unit (GRU), and convolutional LSTM (ConvLSTM) autoencoders, while using the C-FLUX waveform for anomaly detection. The authors demonstrated promising results by detecting 39 out of 50 fault events with a false positive rate of about 10%. It is worth highlighting that both of these efforts (Pappas et al., 2021; Radaideh, Pappas, Walden et al., 2022) highlighted a single HVCM module and a single waveform, whereas this study extends the approach to multiple waveforms (multivariate) and multiple HVCM modules (multi-system). Implementing this multi-system AD study is possible thanks to a larger dataset collected over two years, which includes many fault events in all 15 HVCM modules of the SNS (Radaideh, Pappas, Cousineau et al., 2022). Anomaly detection applications in particle accelerators for systems other than the HVCM have also been demonstrated. AEs are based on vanilla feed-forward ANNs were developed for AD of magnet faults in the Advanced Photon Source storage ring at Argonne National Laboratory (Edelen & Cook, 2021). While the developed models demonstrated a good ability to learn the binary output of whether a fault would occur or not, the authors concluded that their models were unsuccessful in accurately predicting the timing of the fault (Edelen & Cook, 2021). Unsupervised ML techniques with feature extraction support were applied to exploit the data from a RF tuning system in the ALPI accelerator at Legnaro National Laboratories in Italy (Marcato et al., 2021). The authors of Rescic, Seviour, and Blokland (2020) employed different ML binary classifiers (e.g. logistic regression, gradient boosting, random forests Breiman, 2001) to predict machine failures via beam current measurements before they actually occur. The models achieved failure prediction accuracy of up to 92%. The binary classifiers were then improved in a subsequent study by the team (Reščič, Seviour, & Blokland, 2022) for preemptive detection of machine trips in the SNS using differential beam current monitor (DCM); achieving a precision of 96% with 58% true positive and 0% false positive rates (Reščič et al., 2022). The work by Edelen et al. (2018) highlighted opportunities for ML for various applications in particle accelerators including but not limited to anomaly detection, machine protection, system modeling, diagnostics, tuning, and system control. Blokland et al. (2022) has proposed an uncertainty-aware ML method using Siamese model (Koch, Zemel, & Salakhutdinov, 2015) to predict upcoming errant beam pulses from a single monitoring device at the SNS. These results showed an approximately 2x improvement in identifying anomalies over the previously published results in the region of interest. To monitor the trigger of particle physics experiments at the Large Hadron Collider (LHC), Pol, Berger, Cerminara, Germain, and Pierini (2020) proposed to use a modified loss function that allows the CVAE to learn the optimal reconstruction resolution to satisfy stringent constraints, such as performance, simplicity and robustness. The authors reported that the proposed approach outperforms vanilla VAE and other baseline techniques using MNIST dataset (LeCun & Cortes, 2010) and CMS experimental dataset.

4. Background

4.1. Autoencoders

An autoencoder (AE) is a type of ANN for unsupervised learning that has two components: an *encoder* and a *decoder*. As shown in Fig. 3(a), the *encoder* denoted by Ψ learns to encode original input data x typically into a lower dimensional latent space (bottleneck) using a ANN. The encoded representation is then passed to the *decoder* denoted by Φ that learns to reconstruct the original input data using an ANN. The goal of training an AE is to select Ψ and Φ functions that have the minimal error to reconstruct the input data. The loss function used to train an AE is called *reconstruction loss*, that is a comparison of how well the output has been reconstructed from the original input. The *reconstruction loss* of a typical AE can be defined as,

$$\mathcal{L}_{\mathrm{AE}} = \|x - \Phi(\Psi(x))\|^2 \tag{1}$$

that takes the difference between the original input data and the reconstructed one, where x is the input data. There are different metrics proposed in the literature to be used as a *reconstruction loss*; one such example is Mean-Squared-Error as shown in Eq. (4), where x is the input data and $\hat{x}$ is the reconstructed data from the model.

4.2. Variational Autoencoders

While an AE learns a function to map each input to a fixed-size single point reduced representation in the latent space, a Variational Autoencoder (VAE) replaces the latent space with a probability distribution of the input data by replacing the (bottleneck) with two separate vectors μ and σ, representing the mean and the standard deviation of the distribution. As shown in Fig. 3(b), the *encoder* Ψ projects high-dimensional input data x into a lower latent variable z that is forced to follow a certain well-known distribution, such as a Gaussian. The latent variable z in VAE is not the output of the Ψ as in AE, instead, the Ψ estimates μ and σ parameters for each latent variable. Then, the

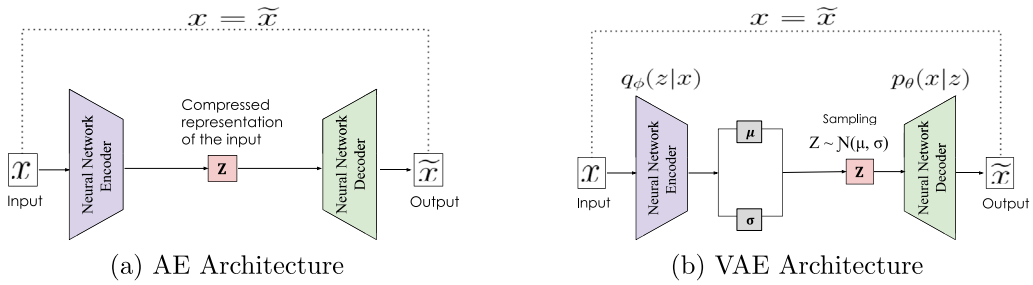

Fig. 3. Figure (a) shows a typical Autoencoder (AE) model that consists of an *encoder* that projects input data into smaller representation z, and a *decoder* that reconstructs input data given latent z. Figure (b) shows a typical Variational (VAE) model consists of an *encoder* that projects the input data into a probability distribution and estimates the μ and σ parameters of that distribution. The sampling layer z takes the estimated parameters to sample a distribution that needs to be as close as possible to the pre-defined distribution. The *decoder* takes the sampling layer z as inputs to reconstruct the input data and generate new samples.

latent z is sampled from the estimated parameters which are fed to the Φ to reconstruct the input data. A new parameter ϵ is also introduced to allow us to reparametrize the sampling layer z and allow the model to backpropagate the entire network. The latent z is now defined as, $z = \mu + \sigma \odot \epsilon$, where $\epsilon \sim \mathcal{N}(0, 1)$. A VAE generalizes the idea of an AE by not only learning the embeddings, but also being able to generate new data by sampling from the estimated latent distribution parameters μ and σ. The loss function for a VAE is motivated by variational inference (Blei, Kucukelbir, & McAuliffe, 2017) via minimizing the Kullback–Leibler divergence (KLD) (Kullback & Leibler, 1951) between the posterior $p(z|x)$ and the encoded prior distribution $q(z) = \mathcal{N}(0, 1)$:

$$\mathcal{L}_{\text{VAE}} = \|x - \Phi(\Psi(x))\|^2 + \eta \; KLD \; (q(z) \parallel p(z|x)) \tag{2}$$

where the first term is the reconstruction error, the second term computes KLD, and η is the harmonic parameter to balance the two.

4.3. Conditional Variational Autoencoders

For structured output predictions, Conditional VAEs (CVAE) were proposed by Sohn et al. (2015) to make diverse predictions for different input samples. The objective function of the VAE can be modified by adding the variable c,

$$\mathcal{L}_{\text{CVAE}} = \|x - \Phi(\Psi(x))\|^2 + \eta \; KLD \; (q(z, c) \parallel p(z|x, c)) \tag{3}$$

where we condition all of the distributions with c. CVAEs are an extension of VAEs by adding the conditional part in the *encoder* and *decoder* to associate the input samples with labels. Therefore at inference time, we have more control to generate samples that belong to specific labels in contrast to a VAE that does not have control over the generated samples.

5. Methods

In this section, we will first describe the experimental data used in this paper, then we will describe the architecture of the single-module-based VAE and the multi-module-based CVAE.

5.1. Data description

We train and test our developed models on experimental data extracted from the HVCM controller. Historically, the IGBT switches (Insulated-gate bipolar transistor) have been a significant source of unplanned downtime, and still occasionally have catastrophic failures which could result in several days of lost neutron production. Reliability has significantly improved over the years with upgrades to the HVCM. The upgrades provided a rich source of digitized waveforms from the HVCMs which will be used for ML. The control of the HVCMs is done with a PXI-based controller running LabView. Along with providing communications and control to the HVCMs, the controller digitizes up to 32 waveforms at a sample period of 20 ns. The waveforms are saved to an on-board computer with a record length of 3 ms. The controller also saves another file which is decimated to 2.5 MS/s but with a record length of 35.5 ms to capture "three" macro-pulses. These files are overwritten every macro-pulse with the exception of when the HVCM faults, and after maintenance has been performed or the system re-tuned in order to record *normal* waveforms. The controller saves a file containing all of the HVCM settings including configuration parameters. From the 32 waveforms saved by the controller, only 14 waveforms were used based on expert opinion on their importance to HVCM reliability. From the full record length of 35.5 ms, we extracted 1.8 ms macro-pulses according to the following strategy:

- For *normal* waveforms, all three macro-pulses are extracted since they are identical and can increase the number of data samples. Each 1.8 ms *normal* macro-pulse has 4500 time steps (i.e. sampling rate is 400 ns)
- For faulty waveforms, the pre-fault pulse was used to predict the anomalies (i.e. detecting the fault ahead of time to allow system trip). Typically in faulty waveform files, the first macro-pulse comes before the fault event (pre-fault), the second macro-pulse has the fault, while the third macro-pulse is usually not saved as the HVCM is down. The faulty 1.8 ms macro-pulse also has 4500 time steps (i.e. sampling rate is 400 ns).

These data pre-processing strategies were done by the team before in their previous work (Radaideh, Pappas, Walden et al., 2022), primarily to reduce data size and cut the irrelevant time steps when the HVCM is idle. See Figure 5, Section 3.2, and Section 4.1 of Radaideh, Pappas, Walden et al. (2022) for more information about data processing. The processed data used in this work is shared with the public (Radaideh, Pappas, Cousineau et al., 2022). The 14 waveforms (features) used in this work are:

- Six IGBT current waveforms, which express the current passing in the phases A+, A+*, B+, B+*, C+, and C+*.
- Three magnetic flux density in the phases A, B, and C of the resonant circuit.
- Two waveforms that represent the cap bank voltage and the cap bank current.
- Two waveforms that represent the modulator output voltage and the modulator current.
- One waveform that represents the time derivative change of the modulator output voltage (i.e. dV/dt).

After data processing, the number of *normal* and *abnormal* samples are saved in a 3D tensor of shape: (5240, 4500, 14) and (1270, 4500, 14) respectively, where the time steps are 4500 and the waveforms features are 14 as described above. The *abnormal* waveform types before and after grouping are shown in Fig. 4.

5.2. Anomaly detection approach

To detect anomalies using VAEs, we train the model using *normal* waveforms to capture the normal behavior. Afterwards we use the

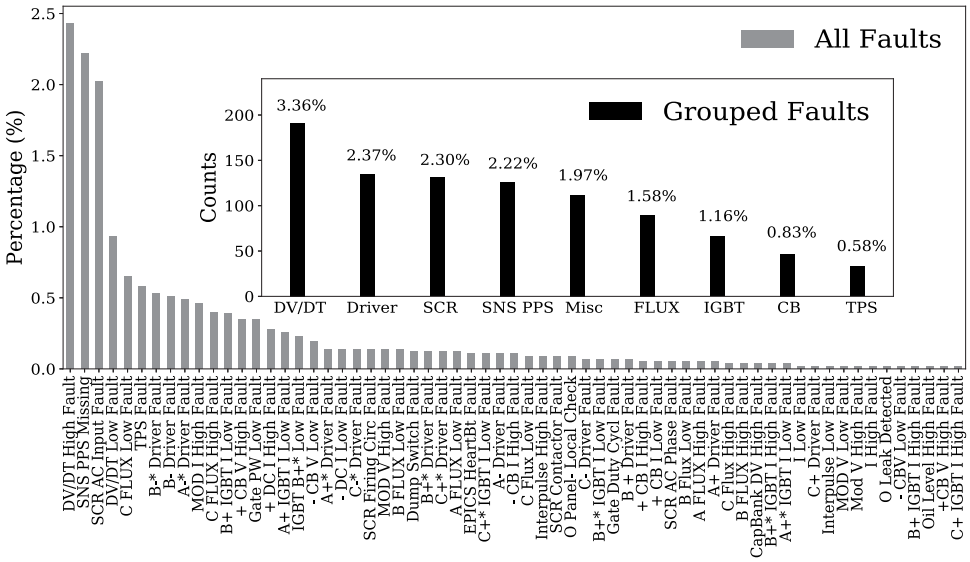

Fig. 4. The outer figure (gray bars) shows percentage of *abnormal* types with respect to all data including *normal*. The inner figure (black bars) shows the counts of *abnormal* data after regrouping, and the proportion of faults are shown on top of each bar.

trained model to detect anomalies by calculating the difference between the input waveform with the predicted waveform using Eq. (4), commonly known as the *reconstruction error*. The *normal* waveforms are expected to have smaller *reconstruction error* than *abnormal* waveforms, therefore, we can define a threshold to achieve the desired requirements. For this paper, we require less than 10% false positives in order to compare to previously published results. As an example, in Fig. 5 we have two different *abnormal* waveforms (blue color) and the reconstructed waveforms from our model (red color), with the *reconstruction error* (yellow band) measuring the difference between the input and reconstructed waveforms. We can see the input *abnormal* example on the left is flat and can be easily identified as faulty even without the need of any ML techniques; however, these easily identified faulty waveforms represent a small fraction of the *abnormal* data, whereas most of the other faulty examples are very similar to *normal* data, and cannot be identified easily, similar to the right plot of Fig. 5. The bottom row of Fig. 5 shows a subset from the top waveforms to zoom on a specific region of the waveforms for clearer visualizations. For faulty waveforms that are difficult to be distinguished from normal ones, ML techniques are needed for detection. The advantage of using VAEs over AEs in this application is that a VAE has a probabilistic model that provides a probability measure instead of a reconstruction error. By using the probabilistic encoder, we can sample from the mean and variance parameters to capture the variability of the reconstruction instead of having a single reconstruction error as an anomaly score. In the following, we use two ML methods for anomaly detection of HVCM data from SNS. One is a VAE for a single HVCM module, and the other is a CVAE for multiple HVCM modules.

5.3. Single-module-based Variational Autoencoder

A standard approach when using VAEs on data originating from multiple modules is to train an individual model for each module. This allows us to focus on each subsystem independently and detect anomalies specific to the module. Initially, we trained separate VAE for each subsystem. We designed a VAE based on one-dimensional convolution layers for both the *encoder* and the *decoder*. The *encoder* takes waveforms x of shape ($N_{samples}$ x $N_{time-steps}$ x $N_{features}$) as input that are processed through multiple 1-dimensional convolutional neural network (CNN) layers. The output of the last CNN block (Conv1D, Batch Normalization, MaxPooling1D) is flattened and fed to fully connected (dense) layers. The last dense layer in the encoder produces estimated parameters μ (mean) and σ (standard deviation) representing our prior distribution which is constrained to follow a Gaussian distribution. Using these two outputs, μ and σ, a customized layer (Lambda) is used to randomly sample a Gaussian distribution that is as close as possible to the prior $q(z)$ to generate the output of the encoder, latent z. The *decoder* receives latent z as an input which is then fed to two dense layers in a sequential manner. The resulting output of the Dense layer is reshaped and processed through three 1 dimensional CNN blocks that reconstruct the waveform x. Similar to a typical CVAE, we use Mean Squared Error (MSE) for the *reconstruction loss* and KLD between the posterior $p(z|x,c)$ and the encoded prior distribution $q(z,c) = \mathcal{N}(0,1)$ so the loss function will be identical to Eq. (2). The architecture of this model is identical to Fig. 6 excluding the conditional part shown in the dashed box. Our multi-module model architecture is inspired by CNNs (LeCun & Bengio, 1998) and fully CNNs (Krizhevsky, Sutskever, & Hinton, 2012). To train this model, we use Keras (Chollet et al., 2015) and TensorFlow (Abadi et al., 2015) backend with the parameters setup shown in Table 1. We create 15 different models, each model simulating one HVCM module. We train each model using all the available features discussed in Section 5.1. After leaving test-out samples, the training data shape becomes (N x 4500 x 14), where $N \approx 450$ samples for each subsystem. The results and evaluations of this methodology will be discussed in Section 6.

5.4. Multi-module-based Conditional Variational Autoencoder

In addition to training an individual model for each HVCM module, we also design a multi-module model that trains all the 15 HVCM systems (discussed in Section 5.1) together. Motivated by the architecture of CVAE, we extend the single-module-based VAE to include a conditional component c, which is the unique IDs for different modules. In Fig. 6, we show the architecture of our model where we have an *encoder* that takes *normal* waveforms x of shape ($N_{samples}$ x $N_{time-steps}$ x $N_{features}$) as inputs, that are passed through three 1 dimensional CNN blocks (Conv1D, Batch Normalization, MaxPooling1D). The output of the last block is flattened and fed to a fully connected layer concatenated with One-Hot-Encoding of the module's unique IDs, which then is passed to another fully connected layer that will generate the latent z which is constrained to follow a Gaussian distribution, providing an estimated parameters μ and σ. The *decoder* receives latent z and the same One-Hot-Encoding of the module's unique IDs as inputs which are concatenated and fed to two dense layers in sequential order. The output

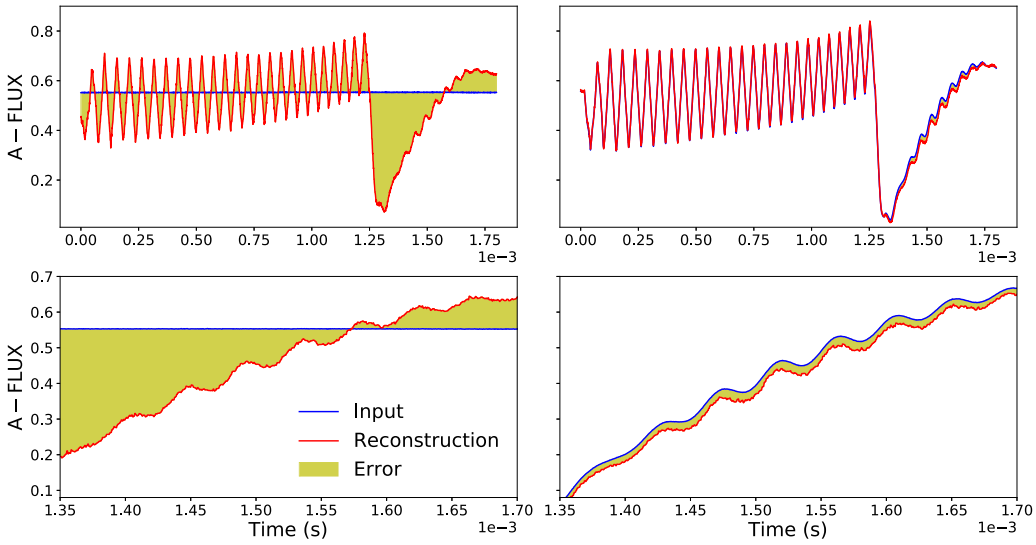

Fig. 5. Top row shows two different observed faulty waveforms (blue) and their corresponding reconstructed samples from multi-module CVAE (red); the difference between the observed and reconstructed waveform is highlighted in yellow. Bottom row shows a subset of the waveforms presented on the top row to zoom on a specific region. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 1
Model setup.

Parameter	Value
Input/output dimensions	(N,4500, 14)
encoder Conv1D blocks	3
decoder Conv1D blocks	3
Number of kernels	128
kernel size	12
Activation function	$ReLU$
Units per Dense layer	512
Latent z	512
Optimizer	Adam
Batch Size	16
Loss	$MSE + KLD$
Learning rate	$10^{-}5$

of the fully connected layers is reshaped and fed to three 1 dimensional convolutional blocks (UpSampling1D, Batch Normalization, Conv1D) that will reconstruct the waveforms x. A 20% Dropout (Srivastava, Hinton, Krizhevsky, Sutskever, & Salakhutdinov, 2014) rate is applied after each CNN block to prevent overfitting. Similar to the previous model, we use MSE for the *reconstruction loss* and KLD between the posterior $p(z|x,c)$ and the encoded prior distribution $q(z,c) = \mathcal{N}(0,1)$ so the loss function will be identical to Eq. (3) after adding the conditional component. The idea of combining all 15 HVCM modules together is to allow the model to learn different representations of *normal* waveforms, improve the model performance for modules with limited samples, eliminate the need for individually retraining each module model and to provide a more generic solution than using a single-module-based model. By adding the conditional component c, the model can learn the association between the waveforms and their modules. To train this model, we use all available 15 modules and features discussed in Section 5.1 which have 5240 waveforms for each feature. After leaving test-out samples, the training data shape becomes (524 x 4500 x 14). We use the same parameters configuration as in the previous model which is shown in Table 1.

6. Results

For this work, we use several evaluation techniques to quantify the performance of two methodologies. The first is the multi-module model that combines all systems together, while the second is the single-module model that is trained individually for multiple systems. We first show the results for the multi-module model and discuss the accuracy of the detecting anomalies, then we compare the two models and have side-by-side plots. In Fig. 7, we show the MSE *reconstruction error* distributions between *normal* and *abnormal* data of multi-module model using box plots. Fig. 7 shows the ability of the model to produce very small errors when reconstructing *normal* waveforms, while the model tends to generate larger errors for *abnormal* waveforms. We can see in almost all waveform sources, such as MOD-V and MOD-I, only using the mean of the box plot we can have a threshold to separate *normal* from *abnormal* waveforms. Next, in Fig. 8, we show a kernel density estimate (KDE) plot, and the corresponding Receiver Operating Characteristic (ROC) curve with the Area Under The Curve (AUC) values using the *reconstruction error* from the multi-module model. For this, we use six faults: (DV/DT, FLUX, IGBT, Driver, SCR, and SNS PPS faults) and use the *reconstruction error* from MOD-V waveforms that produced the highest AUC values. We can see that for certain fault types, such as DV/DT and SNS PPS, the model is capable of separating *normal* waveforms from *abnormal* waveforms, where the density estimation for *normal* is centered between 10^{-5} and 10^{-6}, while the faulty waveforms between 10^{-1} and 1, and this clear separation is being reflected in the corresponding ROC that has AUC = 0.98, and 0.96 respectively. The other faults show reasonable separation with an AUC values ranging from 0.83 to 0.93. It is important to mention that the overlapping between *normal* and *abnormal* KDE are because those waveforms are $normal-like$ samples and do not carry any implication that an anomaly is going to occur in the system, knowing that we are using pre-fault pulses as discussed in Section 5.1. For some faults, it may not have an early enough pre-cursor/fault-indicator in the pre-fault pulse to forecasting the coming fault, thus we are not able to identify them here. Having evaluated the multi-module results, now we compare it with the single-module approach. In Fig. 9, we show normalized KDE between the two methods. As expected, the multi-module is learning from the multiple modules and produce lower MSE values than single-module when reconstructing *normal* waveforms. The Figure shows the overall performance between the two approaches and have a side-by-side comparison between the *reconstruction error* values for all modules. In Fig. 10, we show the AUC values for six fault types that have the highest number of statistics shown in Fig. 4 across multiple modules. We can see for almost all scenarios that multi-module approach has higher AUC values over the single-modules. The error bar is generated by using the probabilistic *encoder* model that provides a probability distribution given a mean and variance parameters. By sampling from

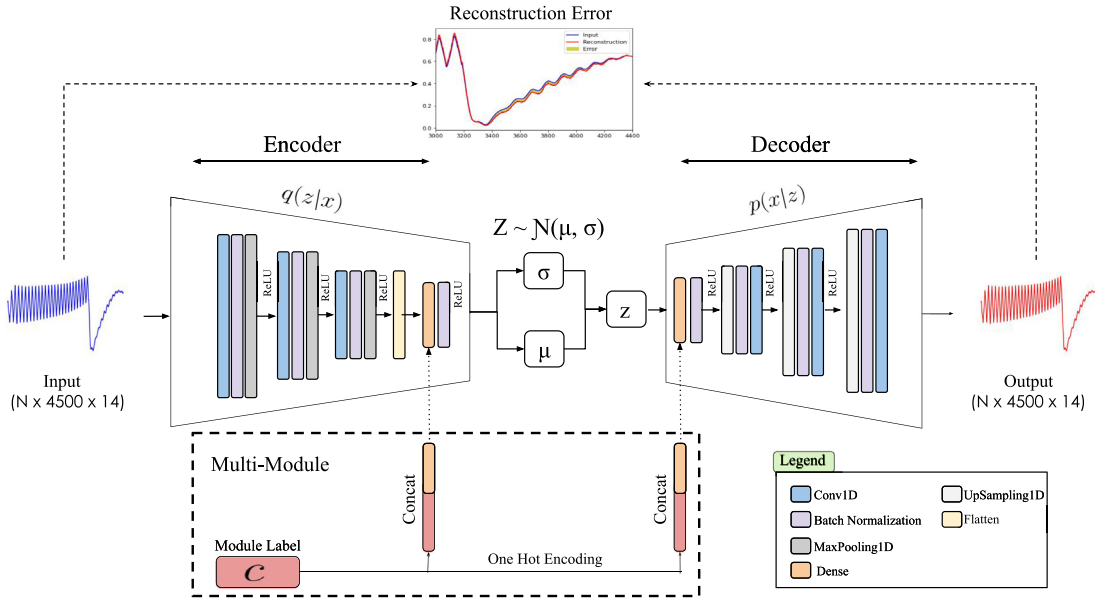

Fig. 6. Anomaly detection method overview. single-module and multi-module models share the same architecture, but the later has the conditional part shown in the dashed box on the bottom of this diagram.

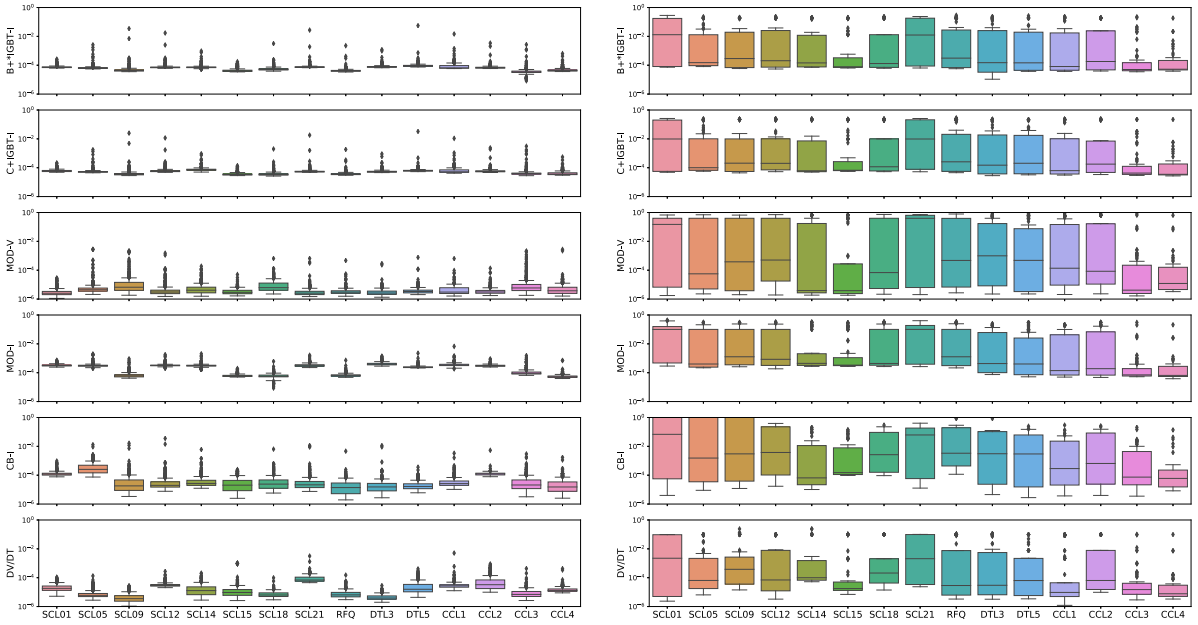

Fig. 7. Box plot shows the MSE *reconstruction error* distributions of *normal* (left plot) and *abnormal* (right plot) using multi-module. The x-axis shows all trained modules and the y-axis shows six source of waveforms.

the estimated parameters at inference time, we generated multiple replicas of reconstructed *normal* and *abnormal* waveforms, then use the mean and SD of the replicas.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(x_i - \hat{x}_i)^2 \tag{4}$$

6.1. Loss landscape

In a simple ML problem, ANNs are trained on feature vectors x_i, and corresponding labels y_i, and number of data samples m, with model parameters θ, and the objective function can be defined as,

$$\mathcal{L}_\theta = \frac{1}{m}\sum_{i=1}^{m} l(x_i, y_i, \theta) \tag{5}$$

that measures how well the ANN can predict y_i given x_i with the model weights θ. The weights of ANNs can be impacted by several factors, such as variable initialization, optimizer, network architecture, batch size, and other hyper-parameters. Studying the effects of various hyper-parameters is challenging because their loss values live in a high-dimensional space. Several scientific applications rely on a simple (1D line) loss curve which is computing the mean or sum of the loss value for each epoch. This produces a scalar for each iteration, and then plot the loss values as a function of epochs. While this method is beneficial to give an overview of the model performance, it only shows a small range of gradients of the parameters, and it does not show the convexity of the function, and why certain NNs architectures generalize better than others. Recently, Li et al. (2018) devolved *filter normalization* technique to visualize the loss landscape of CNNs that can show how convex/non-convex an ANN function is, and explain why certain NN

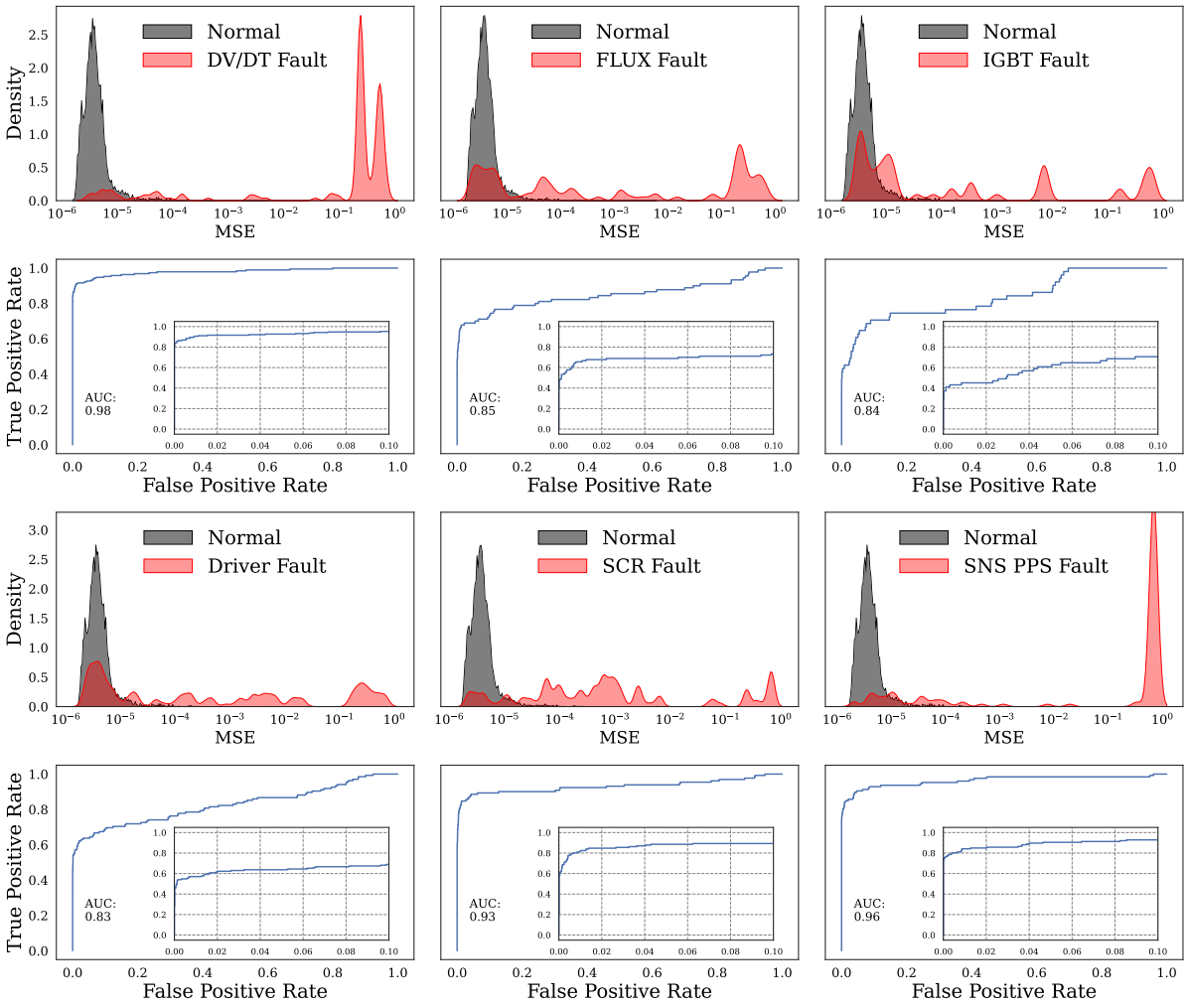

Fig. 8. KDE distributions of the MSE from reconstructing normal (gray color) and faulty waveforms (red color) for six fault types, with the corresponding ROC curve for each fault. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

architectures generalize well while others suffer from high generalization errors. To use this approach, one can choose a center point in the loss surface θ and choose two random direction vectors γ and ν and plot 2D surface of the form $f(\alpha, \beta) = L(\theta + \alpha\gamma + \beta\nu)$, where α and β are the grids of the loss surface and a typical value is between -1 and 1. Increasing the grid values, will increase the space to compute the loss values, while shrinking the grids will zoom in and focus on the area around the minimizer of the trained model. Because the plots are sensitive to the scale of the model weights, the author suggests to rescale the random directions using the proposed *filter normalization* method to have the same Frobenius norm of the corresponding filter in θ. This technique has been used to study the effects of different ANNs architectures (e.g skip connections) and to investigate sharp versus flat minimizers and how they correlate with generalization error.

6.1.1. Single-module vs multi-module

In this Section, we evaluate the performance of the proposed approach multi-module CVAE and compare the model loss surface with the single-module VAE. To have a side-by-side comparison, for single-module we trained 15 individual VAEs and produced the corresponding loss surface, while for the multi-module we trained the model once and generated 15 loss surfaces by feeding different modules at inference time to generate the corresponding loss landscapes. For this exercise, we fixed the random weights initialization and used the parameters in 1. We can see for the single-module shown in Fig. 11 some VAE models show convex behavior such as RFQ and CCL3, while others have chaotic loss surfaces such as CCL4. It is important to mention that there might exist an optimal model architecture for each module and this will require an extensive Neural Architecture Search (NAS) that we leave for future studies. For this analysis, we have instead tested the stability of the results by running different trials with different weights initialization using the same ANN architecture and found that the results are consistent and produce similar loss surface. In Fig. 12, the results obtained from the multi-module show convex-like loss surface regardless to the module we use to compute the loss landscape. This consistency in all modules agrees with the obtained results discussed in Section 6. Evaluating each model using the loss surface was an important step to select our models in addition to analyzing the anomaly detection classification accuracy.

6.2. Model selection

Model selection in ML refers to the process of selecting the final model that addresses a certain problem. There are several criteria that determine the choice of the selected model, such as accuracy, generalization, complexity, scalability, interpretability and explainability. To select a model, one needs to find a set of optimal configuration values which requires exhaustive HPO and NAS. This is because ANNs are typically difficult to configure as they contain millions of trainable parameters and can be impacted by various hyper-parameters choices, such as variable initialization, batch size, and optimizers. In this work, we have investigated different ANNs by implementing a Grid Search

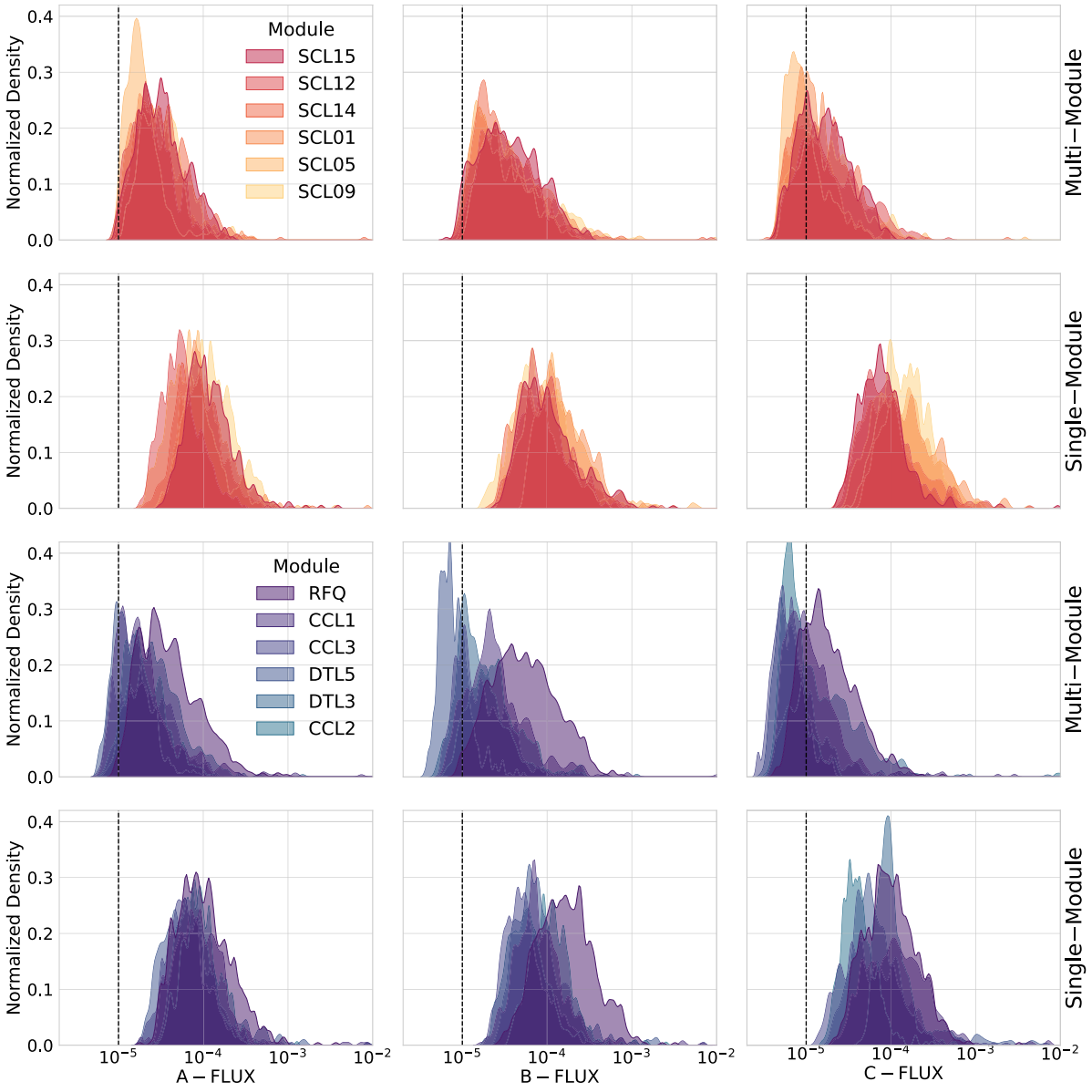

Fig. 9. Normalized density estimation plot shows the reconstruction error (MSE) of normal waveforms using multi-module and single-module. The multi-module model shows smaller reconstruction error, where the distributions of most of the individual systems are more shifted to the dashed black line at $MSE = 10^{-5}$.

(GS) using Weights & Biases Sweeps tool (Biewald, 2020) to find a set of optimal parameters that produce the lowest validation error and smooth loss surface. The GS process builds and evaluates one model for each combination of hyper-parameters. In this example, we trained 150 trials of our multi-module model with a fixed 100 training epochs. All other hyper-parameters that are not part of the GS process are held constants. As shown in Fig. 13, overall we have smaller validation error with Adam optimizer (Kingma & Ba, 2017), and kernel size equals 9 and 12, with smaller batch size. Recent efforts have shown that Adam optimizer tends to converge faster to a sharp minima than Stochastic Gradient Descent (SGD) (Sun, Cao, Zhu, & Zhao, 2019). It has also been debatable that using Adam optimizer leads to poor generalization capability while training complex ANNs (Tang, Huang, Yuan, Wang, & Peng, 2019). This motivates us to further explore the generalization capability by looking at the loss landscape of different ANNs architectures. Visualizing the loss landscape provides some visual understanding of the internal behavior of ANNs and show that the model is likely to produce smaller generalization error, and larger error when the loss surface is chaotic (Li et al., 2018). We use the filter normalization technique discussed in Section 6.1 to explore the effects of deep and shallow ANNs. When deep layers are introduced, it is suggested to implement some tricks to avoid vanishing gradients (Bengio, Simard, & Frasconi, 1994), such as Residual Neural Network (ResNet) (He, Zhang, Ren, & Sun, 2015). To find an optimal number of hidden layers that produces stable results and better generalization, we trained our multi-module model six times using 3, 5, 10, 20, 30, and 40 CNN layers in the *encoder* and the *decoder*. All the other hyper-parameters are held constant to see only the effect of deep vs shallow *encoders* and *decoders*. The hyper-parameter choice is adopted from the model discussed in 5.3 and motivated by the GS process. As shown in Fig. 14, there is a transition from nearly convex to being chaotic loss surface associated with increasing the number of layers. While this behavior is expected because the model starts to suffer from several problems (e.g. vanishing gradient descents) with very deep ANN layers, it can help to select the appropriate number of layers that have convex-like loss surface. In this case, we can see 3, 5 and 10 deep layers are likely to provide stable results with lower generalization error given the optimized hyper-parameters in Table 1.

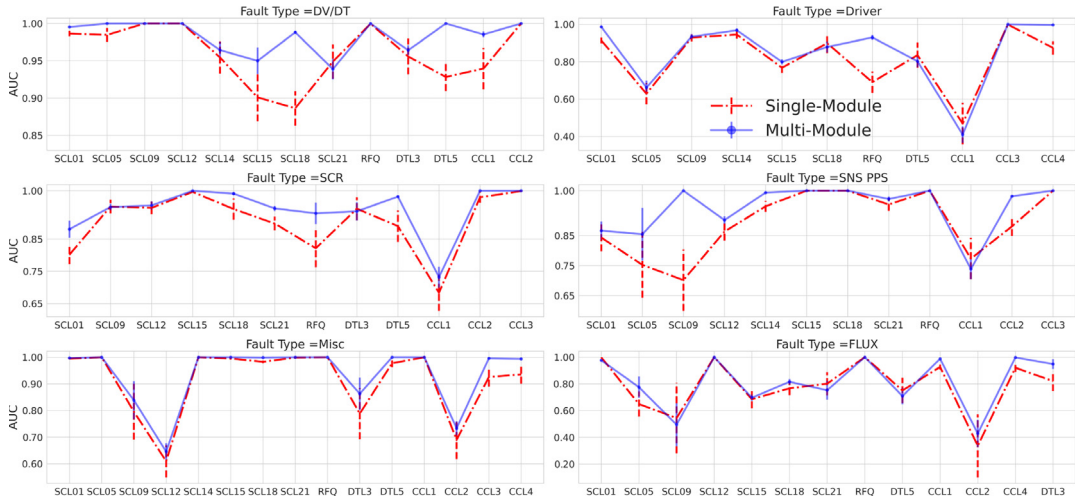

Fig. 10. Compare the AUC values between single-module and multi-module using six types of faults across several modules. The error bar is ± 1 Standard Deviation (SD) error generated by sampling the latent Z of each Method.

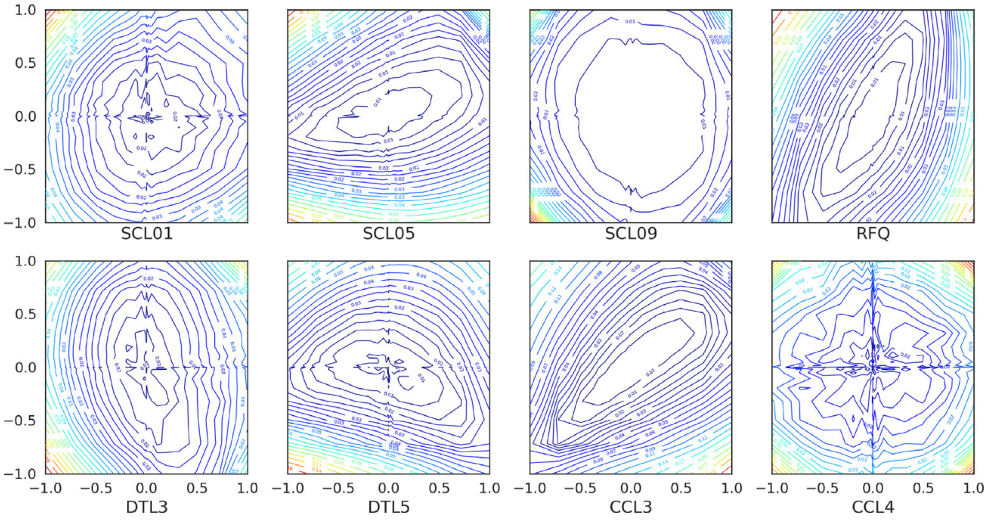

Fig. 11. 2D contour plots of the loss surface of the single-module-based VAEs using the architecture in Fig. 6. The x- and y-axes represent the two random directions in the weight space, and the center corresponds to the model minimizer. Several loss landscapes are dominated by a region with convex contours, while others have less convexity (e.g CCL4).

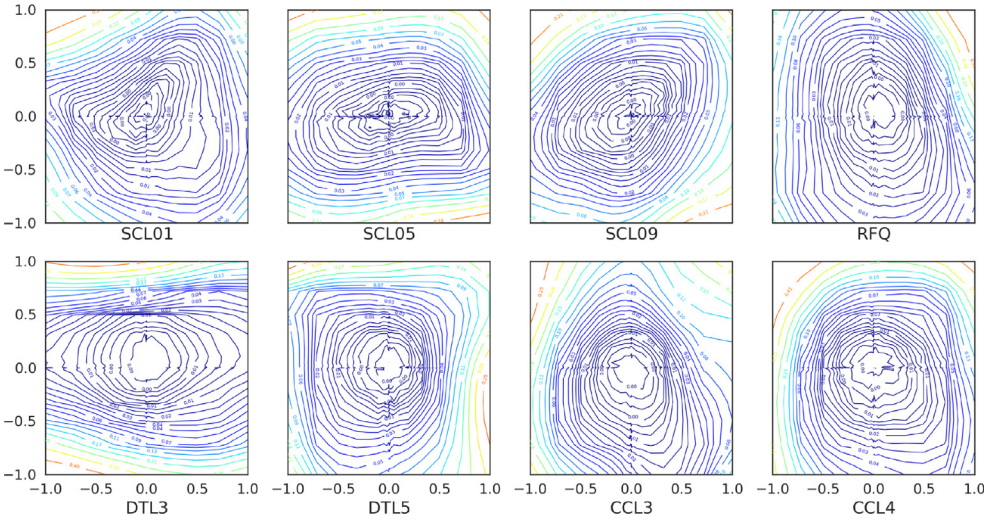

Fig. 12. 2D contour plots of the loss surface of the multi-module CVAE based using the architecture in Fig. 6. The x- and y-axes represent the two random directions in the weight space, and the center corresponds to the model minimizer. All examples have smooth loss surface with convex-like loss landscape.

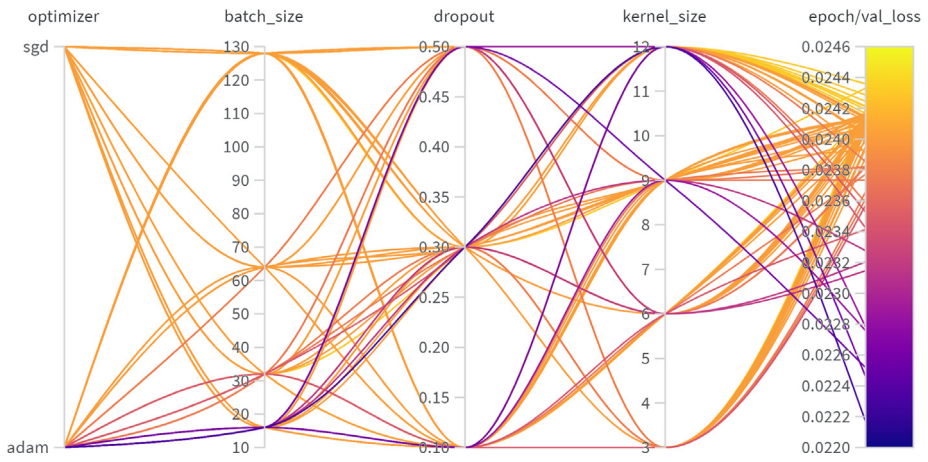

Fig. 13. Parallel Coordinate Plot (PCP) shows different combinations of hyper-parameters, where darker colors show the paths with smaller validation error. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

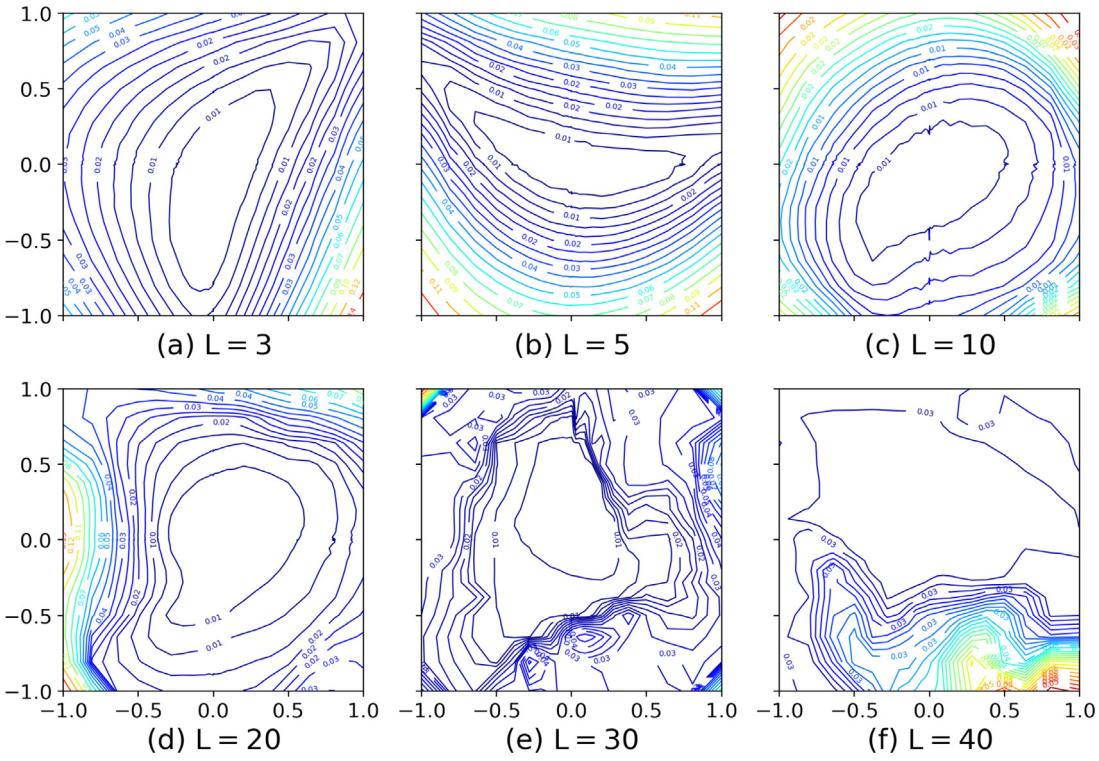

Fig. 14. 2D visualization of the loss surface of the multi-module model trained using different number of Conv1D layers, where L is the number of layers in the *encoder* and *decoder*. The transits from smooth to very chaotic loss surface.

7. Conclusion and outlook

In this paper, we have presented multi-module-based CVAE to detect HVCMs anomalies coming from different modules. The HVCMs consists of 15 modules that produce high quality neutron beams at the SNS. A fault in one or more of the modules can cause a major loss in operational time. To reduce the downtime caused by the HVCMs, our developed multi-module CVAE learns different representation of *normal* waveforms to detect several types of anomalies and produce well-performed AD prediction ahead of time. Through our analysis, we found the multi-module CVAE model to be more sensitive to detecting anomalies than single-module VAE approach. The multi-module produces lower *reconstruction error* for *normal* waveforms with an average of 10^{-4} as compared to single-module that has an average of 10^{-3}. This increases the separation between the *reconstruction* − *errors* of *normal* and *abnormal* samples and produces higher accuracy in detecting abnormalities. For example, for DV/DT fault, multi-module achieves AUC values ranging from 0.94 to 1.00 for all modules, while single-module VAE produces lower than 0.89 AUC as in SCL18. Similarly, for SNS PPS fault, there is more than 20% improvement on SCL09 module, increasing from 0.73 to 1.0 AUC value. The multi-module can be trained once which eliminates the need of retraining a model for each module separately, and allow us to perform NAS and HPO efficiently. The results of the multi-module loss landscape analysis shows convex-like function for all modules, while single-module has a chaotic loss surface for some cases as in CCL module. This indicates that with the given model configurations, multi-module CVAE can produce more staple results and have lower generalization error. The proposed multi-module CVAE approach can improve the HVCM reliability and allows us to schedule preventative maintenance before a catastrophic failure occurs.

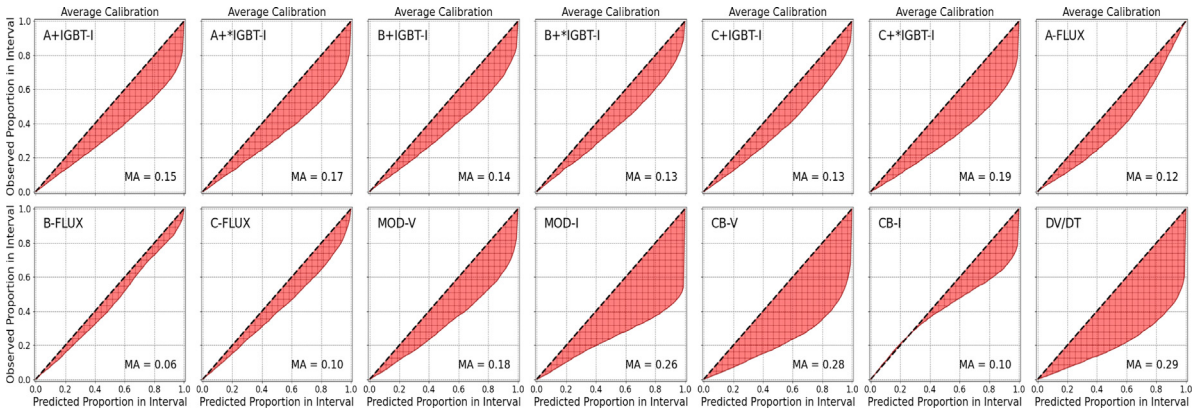

Fig. A.15. Average Miscalibration Area (MA) for each waveform using random examples from SCL01 module. The MA ranges from 6% to 29%.

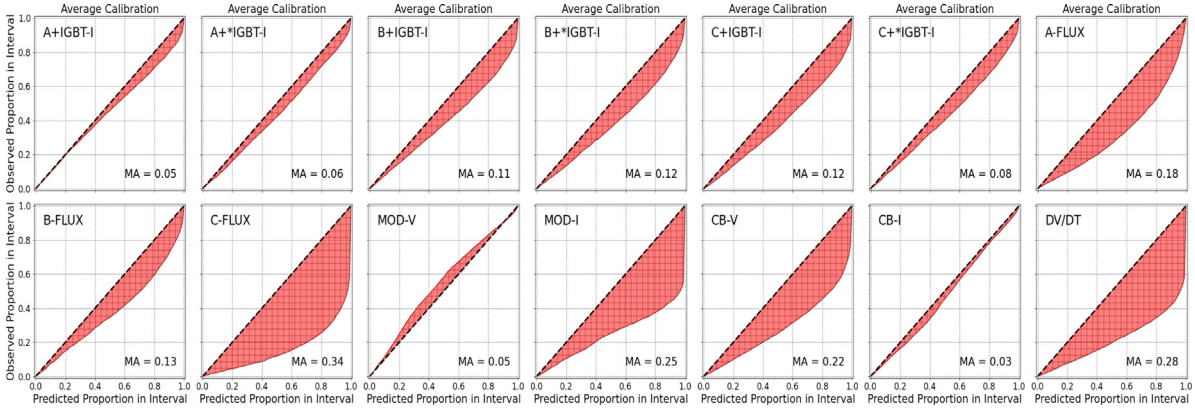

Fig. A.16. As in Fig. A.15, but using CCL4 module.

While the multi-module can generalize to 15 different modules, we expect the model will have limitations to detect anomalies when there is a voltage change in any module that causes some data drift. The voltage change is a *normal* behavior in the modulator, and our proposed model is likely to identify the relative waveforms as *abnormal*. This limitation is seen with any changes in HVCM settings or tuning such as changing the frequency modulation to flatten the pulse, or the start pulses to the IGBTs to ensure the fluxes reset each cycle. Therefore, we plan to incorporate configuration values that provide information in which a data drift is going to happen to accommodate these changes over time. Another potential improvement is to apply attention mechanism (Vaswani et al., 2017) to combine all 14 features that can eliminate the need to do feature engineering to extract the important waveform features that are more sensitive to detecting anomalies.

CRediT authorship contribution statement

Yasir Alanazi: Investigation, Methodology, Software, Visualization, Writing – original draft. **Malachi Schram:** Conceptualization, Funding acquisition, Methodology, Project administration, Supervision, Writing – review & editing. **Kishansingh Rajput:** Software, Supervision, Validation, Writing – review & editing. **Steven Goldenberg:** Writing – review & editing. **Lasitha Vidyaratne:** Software, Writing – review & editing. **Chris Pappas:** Conceptualization, Data curation, Resources, Validation, Review & editing. **Majdi I. Radaideh:** Data curation, Resources, Validation, Writing – review & editing. **Dan Lu:** Investigation, Writing – review & editing. **Pradeep Ramuhalli:** Writing – review & editing. **Sarah Cousineau:** Funding acquisition, Project administration, Resources.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The dataset was published. The link is provided in the manuscript.

Acknowledgments

The authors acknowledge the help from David Brown in evaluating Operations requirements, Frank Liu, for his assistance on the Machine Learning techniques, and Sarah Cousineau for making this grant work possible. This manuscript has been authored by UT-Battelle, LLC, under contract DE-AC05-00OR22725 with the US Department of Energy (DOE). The Jefferson Science Associates (JSA) operates the Thomas Jefferson National Accelerator Facility for the U.S. Department of Energy under Contract No. DE-AC05-06OR23177. This research used resources at the Spallation Neutron Source, a DOE Office of Science User Facility operated by the Oak Ridge National Laboratory, grant No. DE-SC0009915. The US government retains and the publisher, by accepting the article for publication, acknowledges that the US government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this manuscript, or allow others to do so, for US government purposes. DOE will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).

Appendix. Uncertainty quantification

There are several Uncertainty Quantification (UQ) methods used in the ML community to quantify the model and data uncertainty. A comprehensive survey of UQ methods in ML can be found in Abdar et al. (2021). In this work, we generated the uncertainty for the predicted results by sampling the latent z given a mean and variance parameters generated from the probabilistic *encoder* model. Given the generated replicas of reconstructed *normal* and *abnormal* waveforms, we use the mean and SD to estimate the uncertainty using an uncertainty toolbox (Chung, Char, Guo, Schneider, & Neiswanger, 2021). The toolbox provides a miscalibration area (MA) by plotting the observed proportion versus prediction proportion of outputs falling into a range of intervals, and given a number of bins. Figs. A.15 and A.16 show the Average Miscalibration Area (MA) from 10 random examples selected from SCL01 and CCL4 modules respectively. The results show that some of the waveforms, such as the IGBTs have small MA, while MOD-I and CB-V have larger MA for both modules. It is important to mention that the results are generated using few random examples and they might not represent all other examples or modules. For our future work, we plan to investigate model calibration and have an extensive study to quantify the uncertainty of the predicted results.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., et al. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. URL https://www.tensorflow.org/, Software available from tensorflow.org.

Abdar, M., Pourpanah, F., Hussain, S., Rezazadegan, D., Liu, L., Ghavamzadeh, M., et al. (2021). A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, *76*, 243–297. http://dx.doi.org/10.1016/j.inffus.2021.05.008.

Adewumi, A. O., & Akinyelu, A. A. (2017). A survey of machine-learning and nature-inspired based credit card fraud detection techniques. *International Journal of Systems Assurance Engineering and Management*, *8*, 937–953.

Alfeo, A. L., Cimino, M. G., Manco, G., Ritacco, E., & Vaglini, G. (2020). Using an autoencoder in the design of an anomaly detector for smart manufacturing. *Pattern Recognition Letters*, *136*, 272–278. http://dx.doi.org/10.1016/j.patrec.2020.06.008, URL https://www.sciencedirect.com/science/article/pii/S0167865520302269.

An, J., & Cho, S. (2015). Variational autoencoder based anomaly detection using reconstruction probability. *Special Lecture on IE*, *2*(1).

Ball, J. E., Anderson, D. T., & Chan, C. S. (2017). Comprehensive survey of deep learning in remote sensing: theories, tools, and challenges for the community. *Journal of Applied Remote Sensing*, *11*(04), 1. http://dx.doi.org/10.1117/1.jrs.11.042609.

Bengio, Y., Simard, P., & Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, *5*(2), 157–166. http://dx.doi.org/10.1109/72.279181.

Biewald, L. (2020). Experiment tracking with weights and biases. URL https://www.wandb.com/, Software available from wandb.com.

Blei, D., Kucukelbir, A., & McAuliffe, J. (2017). Variational inference: A review for statisticians. *Journal of the American Statistical Association*, *112*(518), 859–877.

Blokland, W., Rajput, K., Schram, M., Jeske, T., Ramuhalli, P., Peters, C., et al. (2022). Uncertainty aware anomaly detection to predict errant beam pulses in the oak ridge spallation neutron source accelerator. *Physical Review Accelerators and Beams*, *25*, Article 122802. http://dx.doi.org/10.1103/PhysRevAccelBeams.25.122802, URL https://link.aps.org/doi/10.1103/PhysRevAccelBeams.25.122802.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32. http://dx.doi.org/10.1023/A:1010933404324.

Caryotakis, G. (2004). High power klystrons: Theory and practice at the stanford linear accelerator CenterPart I.

Chalapathy, R., & Chawla, S. (2019). Deep learning for anomaly detection: A survey. http://dx.doi.org/10.48550/ARXIV.1901.03407, URL https://arxiv.org/abs/1901.03407.

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys*, *41*(3), http://dx.doi.org/10.1145/1541880.1541882.

Chollet, F., et al. (2015). Keras. https://github.com/fchollet/keras.

Chung, Y., Char, I., Guo, H., Schneider, J., & Neiswanger, W. (2021). Uncertainty toolbox: an open-source library for assessing, visualizing, and improving uncertainty quantification. arXiv:2109.10254.

Dang, T. T., Ngan, H. Y., & Liu, W. (2015). Distance-based k-nearest neighbors outlier detection method in large-scale traffic data. In *2015 IEEE international conference on digital signal processing* (pp. 507–510). http://dx.doi.org/10.1109/ICDSP.2015.7251924.

Edelen, J. P., & Cook, N. M. (2021). Anomaly detection in particle accelerators using autoencoders. arXiv preprint arXiv:2112.07793.

Edelen, A., Mayes, C., Bowring, D., Ratner, D., Adelmann, A., Ischebeck, R., et al. (2018). Opportunities in machine learning for particle accelerators. arXiv preprint arXiv:1811.03172.

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep residual learning for image recognition. arXiv:1512.03385.

Hinton, G. E., & Zemel, R. S. (1993). Autoencoders, minimum description length and Helmholtz free energy. In *Proceedings of the 6th international conference on neural information processing systems* (pp. 3–10). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc..

John, G. H. (1995). Robust decision trees: Removing outliers from databases. In *Proceedings of the first international conference on knowledge discovery and data mining* (pp. 174–179). AAAI Press.

Kieu, T., Yang, B., Guo, C., & Jensen, C. S. (2019). Outlier detection for time series with recurrent autoencoder ensembles. In *Proceedings of the twenty-eighth international joint conference on artificial intelligence* (pp. 2725–2732). International Joint Conferences on Artificial Intelligence Organization, http://dx.doi.org/10.24963/ijcai.2019/378.

Kingma, D. P., & Ba, J. (2017). Adam: A method for stochastic optimization. arXiv:1412.6980.

Kingma, D. P., & Welling, M. (2013). Auto-encoding variational Bayes. http://dx.doi.org/10.48550/ARXIV.1312.6114, URL https://arxiv.org/abs/1312.6114.

Koch, G., Zemel, R., & Salakhutdinov, R. (2015). Siamese neural networks for one-shot image recognition.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In F. Pereira, C. Burges, L. Bottou, & K. Weinberger (Eds.), *Advances in neural information processing systems, Vol. 25*. Curran Associates, Inc., URL https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

Kullback, S., & Leibler, R. A. (1951). On information and sufficiency. *The Annals of Mathematical Statistics*, *22*(1), 79–86.

LeCun, Y., & Bengio, Y. (1998). Convolutional networks for images, speech, and time series. In *The handbook of brain theory and neural networks* (pp. 255–258). Cambridge, MA, USA: MIT Press.

LeCun, Y., & Cortes, C. (2010). MNIST handwritten digit database. URL http://yann.lecun.com/exdb/mnist/.

Li, H., Xu, Z., Taylor, G., Studer, C., & Goldstein, T. (2018). Visualizing the loss landscape of neural nets. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, & R. Garnett (Eds.), *Advances in neural information processing systems, Vol. 31*. Curran Associates, Inc., URL https://proceedings.neurips.cc/paper/2018/file/a41b3bb3e6b050b6c9067c67f663b915-Paper.pdf.

Litjens, G., Kooi, T., Bejnordi, B. E., Setio, A. A. A., Ciompi, F., Ghafoorian, M., et al. (2017). A survey on deep learning in medical image analysis. *Medical Image Analysis*, *42*, 60–88. http://dx.doi.org/10.1016/j.media.2017.07.005, URL https://www.sciencedirect.com/science/article/pii/S1361841517301135.

Lu, W., Cheng, Y., Xiao, C., Chang, S., Huang, S., Liang, B., et al. (2017). Unsupervised sequential outlier detection with deep architectures. *Transactions on Image Processing*, *26*(9), 4321–4330. http://dx.doi.org/10.1109/TIP.2017.2713048.

Marcato, D., Arena, G., Bortolato, D., Gelain, F., Martinelli, V., Munaron, E., et al. (2021). Machine learning-based anomaly detection for particle accelerators. In *2021 IEEE conference on control technology and applications* (pp. 240–246). IEEE.

Mohammadi, M., Al-Fuqaha, A., Sorour, S., & Guizani, M. (2017). Deep learning for IoT big data and streaming analytics: A survey. http://dx.doi.org/10.48550/ARXIV.1712.04301, URL https://arxiv.org/abs/1712.04301.

Pappas, G., Lu, D., Schram, M., & Vrabie, D. (2021). Machine learning for improved availability of the SNS klystron high voltage converter modulators. In *International particle accelerator conference*, *Proc. IPAC'21* (12), (pp. 4303–4306). JACoW Publishing, Geneva, Switzerland, http://dx.doi.org/10.18429/JACoW-IPAC2021-THPAB252.

Pol, A. A., Berger, V., Cerminara, G., Germain, C., & Pierini, M. (2020). Anomaly detection with conditional variational autoencoders. arXiv:2010.05531.

Potter, K. C. (2006). Methods for presenting statistical information: The box plot. In *Visualization of large and unstructured data sets*.

Radaideh, M. I., Pappas, C., & Cousineau, S. (2022). Real electronic signal data from particle accelerator power systems for machine learning anomaly detection. *Data in Brief*, *43*, Article 108473.

Radaideh, M. I., Pappas, C., Walden, J., Lu, D., Vidyaratne, L., Britton, T., et al. (2022). Time series anomaly detection in power electronics signals with recurrent and ConvLSTM autoencoders. *Digital Signal Processing*, *130*, Article 103704.

Radaideh, M. I., Tran, H., Lin, L., Jiang, H., Winder, D., Gorti, S., et al. (2022). Model calibration of the liquid mercury spallation target using evolutionary neural networks and sparse polynomial expansions. *Nuclear Instruments & Methods in Physics Research, Section B (Beam Interactions with Materials and Atoms)*, *525*, 41–54.

Reass, W., Apgar, S., Baca, D., Borovina, D., Bradle, J., Doss, J., et al. (2003). Design, status, and first operations of the spallation neutron source polyphase resonant converter modulator system. In *Proceedings of the 2003 particle accelerator conference, Vol. 1* (pp. 553–557). http://dx.doi.org/10.1109/PAC.2003.1288975.

Rescic, M., Seviour, R., & Blokland, W. (2020). Predicting particle accelerator failures using binary classifiers. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, *955*, Article 163240.

Reščič, M., Seviour, R., & Blokland, W. (2022). Improvements of pre-emptive identification of particle accelerator failures using binary classifiers and dimensionality reduction. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, *1025*, Article 166064.

Sohn, K., Lee, H., & Yan, X. (2015). Learning structured output representation using deep conditional generative models. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, & R. Garnett (Eds.), *Advances in neural information processing systems, Vol. 28*. Curran Associates, Inc., URL https://proceedings.neurips.cc/paper/2015/file/8d55a249e6baa5c06772297520da2051-Paper.pdf.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, *15*(56), 1929–1958, URL http://jmlr.org/papers/v15/srivastava14a.html.

Sun, S., Cao, Z., Zhu, H., & Zhao, J. (2019). A survey of optimization methods from a machine learning perspective. arXiv:1906.06821.

Tang, M., Huang, Z., Yuan, Y., Wang, C., & Peng, Y. (2019). A bounded scheduling method for adaptive gradient methods. *Applied Sciences*, *9*(17), http://dx.doi.org/10.3390/app9173569, URL https://www.mdpi.com/2076-3417/9/17/3569.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., et al. (2017). Attention is all you need. arXiv:1706.03762.

White, M. (2002). The spallation neutron source (SNS). In *Proceedings of LINAC 2002, Gyeongju, Korea*.

Xu, D., Ricci, E., Yan, Y., Song, J., & Sebe, N. (2015). Learning deep representations of appearance and motion for anomalous event detection. http://dx.doi.org/10.48550/ARXIV.1510.01553, URL https://arxiv.org/abs/1510.01553.