Variational Recurrent Autoencoder for Anomaly Detection on satellite telemetry data.

Francesco Danese[1]

[1]Università di Roma La Sapienza, Via Aldo Moro 5, Rome, Italy

Abstract

This study introduces an anomaly detection framework for satellite telemetry data released by the European Space Agency (ESA). The data come as a multivariate time-series where multiple channels represent measurements from several satellite sensors, with the data preprocessed according to ESA's standard methodology. Our approach employs a Variational Recurrent Autoencoder consisting of a Bi-directional Long Short-Term Memory (Bi-LSTM) network, replacing traditional positional encoding, followed by a block of multihead self-attention with residual connections that encodes each timestep into a latent representation. A corresponding specular decoder reconstructs the input, and the Mean Absolute Error (MAE) between the reconstructed output and the original input serves as the anomaly score. We test and compare several thresholding techniques, with the Global STD error thresholding method resulting in greater performance. Evaluation metrics are computed both pointwise and through an overlap analysis between predicted and labeled anomaly sequences, according to the fact that real anomalies usually last several consecutive time steps. The training strategy segments the input data into consecutive windows with a defined overlap percentage, which is more efficient than the classical sliding window approach and also reduces redundancy among training windows while still ensuring enough temporal coverage. The presented approach outperforms both tested baselines and known methods whose results are acquired from the related literature.

Keywords

Anomaly Detection, Time Series, Variational Autoencoder, ESA

1. HIGHLIGHTS

- Input series are created with a **strided window** approach instead of the direct sliding window method, reducing training set size and lowering computational burden and training time, while achieving similar performance.
- A novel Variational Recurrent Autoencoder (VRAE) architecture integrating Bi-directional LSTM and multi-head self-attention layers for effective anomaly representation
- Extensive comparison of thresholding techniques, highlighting Global Error STD (N=5) as a robust and precise method suitable for operational anomaly detection
- Point-wise and segment-wise metrics are utilized to thoroughly evaluate model performance, providing insights into both instantaneous and prolonged anomalies.
- The proposed method significantly outperforms state-of-the-art baseline models, showcasing enhanced precision and balanced detection performance in real-world telemetry scenarios.

2. Introduction

Satellite telemetry denotes the continuous stream of measurements and information transmitted from satellites to ground stations. The term "telemetry" originates from the Greek roots "tele," meaning "far off," and "metron," meaning "measure," signifying the process of remotely collecting and transmitting data from inaccessible locations. The concept of telemetry dates back to the late 19th and early 20th centuries, coinciding with the advent of radio communication, with NASA's Deep Space Network (DSN), established in the late 1950s, designed to track and communicate with spacecraft venturing beyond Earth's orbit, highlighting the critical role of telemetry in space missions. Spacecraft are intricate and costly machines equipped with hundreds of sensors that monitor various parameters such as temperature, radiation levels, power consumption, instrumentation status, and computational activities. Given their complexity and the significant investments involved, monitoring these channels is essential to ensure spacecraft health and functionality while notifying whatever unexpected phenomenon is happening in the vast outer space. As satellites generate vast amounts of multivariate time-series data, traditional manual monitoring methods requiring human experts labor, are increasingly stressful and time-consuming. Missing the detection of potential issues or unexpected space events can lead to partial or complete mission failures, making anomaly detection a crucial task for alerting operations engineers and the earth base.

Acknowledging this necessity, the European Space Agency (ESA) has taken a pioneering step by releasing the first open, large-scale, real-life satellite telemetry anomaly dataset, including measurements coming from three ESA missions. This dataset serves as a benchmark for developing and evaluating machine learning algorithms aimed at automating anomaly detection in satellite telemetry. The experiments presented in this work are conducted on the first 3 consecutive months of acquisition from mission 1.

According to [1], the most widely adopted method for automatically detecting anomalous behavior in space operations is the use of out-of-limits (OOL) alarms. This approach involves establishing predefined upper and lower thresholds for each telemetry channel of interest. When a data point exceeds the upper limit or falls below the lower limit, an alarm is triggered. Engineers then inspect the parameter that was reported to determine whether it constitutes an anomaly and decide what is the appropriate response or fix. This approach may overlook collective anomalies (sequences of data points that are anomalous when considered together) and contextual anomalies (data points that are anomalous within a specific context but may appear normal otherwise). Additionally, the OOL method requires significant domain expertise to set appropriate thresholds, which can be challenging in complex systems with a great num-

francescodanese01@gmail.com (F. Danese)
https://github.com/Paco-Danes (F. Danese)

ber of heterogeneous measurements, and lacks any kind of dynamic adaptation in case of evolving environments.

Other traditional and simple anomaly detection methods, such as the Global-STD approach, works by pointing out samples that deviate from the mean of a channel by more than a specified number of standard deviations. Common thresholds include 3 (STD3) and 5 (STD5) standard deviations, according to the normality assumption of the data and its statistical property of the majority of samples falling in that predefined interval. While straightforward, this method has known limitations: it is insensitive to local anomalies, does not account for inter-channel dependencies, and its performance degrades with non-stationary signals or sudden changes in the input distibutions caused by long-lasting anomalies.

Another studied approach is Telemanom [2], which employs a semi-supervised algorithm based on Long Short-Term Memory (LSTM) recurrent neural networks. The model predicts a short range of future data points for a single channel based on the larger window of preceding samples from multiple input channels. The mean absolute difference between the forecasted and actual signals serves as the anomaly score, which is then thresholded using a non-parametric dynamic algorithm (NDT) created ad-hoc. However, Telemanom faces challenges [3] such as memory inefficiency, reliance on data-specific thresholding conditions, inadequate handling of anomalies in training data, and scalability issues when dealing with numerous channels.

In response to these challenges, the Dilated Convolutional Variational Autoencoder (DC-VAE) [4] has been introduced as a novel approach to anomaly detection in multivariate time-series data. DC-VAE exploits dilated convolutions to capture both long and short-term dependencies therefore being efficient even with large datasets. The model outputs two values for each input timestep and channel, representing the predicted mean and standard deviation of the normal distribution from which that point presumably comes from, and uses this statistics to threshold anomalous inputs.

Building upon the known literature, our study proposes Variational Recurrent Autoencoder architecture that integrates a Bi-directional Long Short-Term Memory (Bi-LSTM) network with a multi-head self-attention mechanism, employed to learn and replicate the non-anomalous distribution of the data. The Bi-LSTM captures temporal dependencies in the telemetry data, removing the need of positional encoding, while the self-attention mechanism enables the model to focus on relevant features across different time steps, resulting in a fine-grained latent space that keeps the time-dimension instead of being a single vector. The decoder part of the VAE than aims to reconstruct the input window, failing whenever an anomaly occurs, if properly trained on only nominal data points. The reconstruction error at each point in time for each particular channel is consequently used as anomaly score, and several thresholding schemes are tested. The system therefore permits a channel-aware time-specific detection procedure, which is the most granular anomaly identification outcome, as opposite to other methods reporting just the timestamp of the anomalous occurrence with no clue on the specific channel(s) involved.

3. Related work

Telemannon-ESA [2] A semi-supervised algorithm created by NASA engineers, that employs a simple LSTM-based RNN trained to forecast the incoming 10 future samples of a single channel using as features the 250 preceding samples from multiple channels. The absolute difference between the actual and predicted points is taken as anomaly score and passed to their Non-Parametric Dynamic Thresholding (NDT) algorithm to finally flag anomalies. The NDT algorithm works by first smoothing the error sequence E_s with an Exponentially Weighted Moving Average (EWMA), then computes the mean and standard deviation of the smoothed error across time. Afterwards, for each candidate z (from 2 to 10):
-For all $e_s \in E_s$ flag anomalies where:

$$e_s > t$$

$$t = \mu(E_s) + z \cdot \sigma(E_s)$$

- Compute the score function:

$$S(z) = \frac{\frac{\Delta\mu(E_s)}{\mu(E_s)} + \frac{\Delta\sigma(E_s)}{\sigma(E_s)}}{|E_a| + |E_{aseq}|^2}$$

where:
$\Delta\mu(E_s) = \mu(E_s) - \mu(\{e_s \in E_s \mid e_s < t\})$;
$\Delta\sigma(E_s) = \sigma(E_s) - \sigma(\{e_s \in E_s \mid e_s < t\})$;
$|E_a| = \{e_s \in E_s \mid e_s > t\}$
$|E_{aseq}|$ is the number of contiguous anomalous sequences.
- Choose z that maximizes $S(z)$:

$$z^* = \arg\max_z S(z)$$

- Use z^* to flag final anomalies.
In simple terms, among all evaluated thresholds, we choose the one that if all values above are removed, would cause the greatest percent decrease in the mean and standard deviation of the smoothed errors E_s. The score also penalizes for having larger numbers of anomalous values ($|E_a|$) and sequences ($|E_{aseq}|$) to prevent an extremely greedy behavior that would otherwise result in all points be classified as anomalous. This thresholding algorithm is one of the methods tested also with our model, but we noticed that within our framework, even with the terms in the denominator of the score function, an excessive number of false positives is still detected.

DC-VAE [4] A CNN-based VAE that uses dilated convolution to increase the receptive field of the hidden neurons in a causal manner, such that the prediction of each sample depends only on the preceding ones. A sliding window select a section of the input time series and encodes it in a latent space of small hidden size and same number of time-steps. The decoder then uses the values sampled from the latent space to produce μ_x and σ_x, two matrices of shape (*number of channels* $\times$ *number of time steps*). From these two output matrices, the anomaly detection only considers their values at time t, corresponding to two vectors $\mu(t)$ and $\sigma(t)$. The value $x_c(t)$ of a particular channel c at time t is flagged as an anomaly if:

$$|x_c(t) - \mu_c(t)| > z_c \cdot \sigma_c(t)$$

where z_c can be set independently for each channel allowing a sensor-dependent detection of the whole telemetry time series. The original paper does not specify what is the loss function used to train such a model, but it will likely have the negative log likelihood of the data under the probability

distribution parameterized by the predicted parameters in place of the reconstruction loss, namely:

$$Recon.\,Loss = -\log\, P(x|\mu_x, \sigma_x)$$

where for Gaussian distributions, this can be expressed as:

$$Recon.\,Loss = -\sum_{i}^{c}\Big[\log(2\pi\sigma_i^2) + \frac{(x_i-\mu_i)^2}{\sigma_i^2}\Big]$$

eventually accumulated over all timesteps.

4. Dataset

The dataset employed in this work is a segment of the large-scale *ESA-AD* dataset, released to the public on June 2024. We considered the first 3 months of mission 1, encompassing more than 260000 consecutive observations from multiple input channels. Out of 76 channels, only 58 have to be scanned for abnormal values (target channels) while the remaining 18 (non-target channels) are not expected to contain any anomaly according to ESA experts, but they can still be used as features to help the detection process on the target channels. An overview of the characteristics of the used data is shown in Table 1. There are many different types of channels, including monotonic, categorical, and binary ones, so a consistent preprocessing procedure is needed to run and compare the majority of algorithms. Moreover, due to changing in policies by the space operators happening over the long course of the mission, the sampling rate is not constant, hence a resampling process is necessary to work with uniformly-sampled data. Fortunately, ESA released the code of their standard preprocessing [3], that takes care of resampling, standardization of continuous channels and discretization of categorical channels. After running this procedure on the raw measurements, the resulting dataset have the value of each channel ranging between 0 to 1 and a constant sampling rate according to the new timestamps. In

Mission 1	-
Duration	3 months
Target-Ch.	58
Non-Target Ch.	18
Total timesteps	262903
Anomalies	38264
Telecommands	unused

Table 1
Summary of dataset information

addition to the channel values, the original dataset contains also telecommands, i.e. different kinds of inputs sent from earth to run activities or procedures on board. The exact timestamp of execution is reported along the specific code of the telecommand, which could be useful to consider for reducing false positive detections caused by commanded maneuvers that arises (apparently) anomalous sensor values. However, in the considered section of the dataset, the number of issued telecommands is limited and counting the ones that happen together with an anomaly results in a very small negligible number. As a consequence, we discard all telecommand information and focus on working with just the target e non-target channels.

5. Preliminaries

LSTM Long Short-Term Memory (LSTM) [5] is a special type of recurrent neural network architecture specifically designed to address the vanishing gradient problem and capture long-term dependencies in sequential data. An LSTM cell incorporates gating mechanisms to selectively remember or forget information across timesteps. It processes input x_t at time t, along with the previous hidden state h_{t-1} and cell state c_{t-1}, updating as follows:

$$\begin{aligned}
f_t &= \sigma(W_f x_t + U_f h_{t-1} + b_f)\\
i_t &= \sigma(W_i x_t + U_i h_{t-1} + b_i)\\
o_t &= \sigma(W_o x_t + U_o h_{t-1} + b_o)\\
\tilde{c}_t &= \tanh(W_c x_t + U_c h_{t-1} + b_c)\\
c_t &= f_t \odot c_{t-1} + i_t \odot \tilde{c}_t\\
h_t &= o_t \odot \tanh(c_t)
\end{aligned}$$

Here, σ denotes the sigmoid function that outputs values between 0 and 1, allowing the gates to control the flow of information. The element-wise multiplication, denoted by $\odot$, is used to selectively pass information forward. A Bidirectional LSTM processes the input sequence in both forward and backward directions simultaneously, eventually concatenating the outputs.

Multi-Head Self-Attention Attention mechanisms [6] are powerful tools that enables a model to weigh the importance of different positions in the input sequence, effectively capturing relationships between data points regardless of their positions. Multi-Head Self-Attention makes use of multiple attention heads, allowing the model to focus simultaneously on different kinds of information. Given an input sequence $X \in \mathbb{R}^{n\times d}$, each attention head computes attention scores and aggregates information as follows:

$$\text{Attention}(Q,K,V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V,$$

where queries (Q), keys (K), and values (V) are obtained by linear transformations of the input X:

$$Q = XW^Q, \quad K = XW^K, \quad V = XW^V.$$

The Multi-head attention mechanism concatenates the outputs from all attention heads and applies a final linear transformation, enabling the model to leverage diverse patterns learned by individual heads:

$$\text{MultiHead}(X) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W_O$$

$$\text{head}_i = \text{Attention}(XW_i^Q, XW_i^K, XW_i^V)$$

In these equations, $W_i^Q, W_i^K, W_i^V \in \mathbb{R}^{d\times d_k}$, and $W^O \in \mathbb{R}^{hd_k\times out}$, with h denoting the number of attention heads, $d_k = d/h$ representing the dimensionality of each head and out denoting the final output dimensionality.

VAE Variational Autoencoder (VAE) [7] is a generative model composed of an encoder-decoder structure that learns to represent input data as a latent distribution rather than deterministic values. The encoder maps the input x to a latent representation z, modeled by a Gaussian distribution with mean μ and variance σ^2, computed as:

$$\mu = f_\mu(x),$$

$$\sigma = f_\sigma(x).$$

A latent vector z is sampled from this distribution using the reparameterization trick:

$$z = \mu + \sigma \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I).$$

The VAE optimizes the Evidence Lower Bound (ELBO), which combines the reconstruction loss and a Kullback-Leibler (KL) divergence term that regularizes the latent space to match a prior distribution, typically a unit Gaussian:

$$\mathcal{L}(\theta, \phi; x) = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - D_{KL}\left(q(z|x) \| \| p(z)\right)$$

Here, the first term encourages accurate reconstruction of input data, while the second term regularizes the latent distribution with $q_\phi(z|x)$ and $p_\theta(x|z)$ being respectively the encoder and decoder parts of the neural network. With regards to the anomaly detection task, VAEs are widely employed to learn the distribution of nominal data by training exclusively on non-anomalous examples. At inference time, samples that significantly deviate from the learned distribution, indicated by high reconstruction errors, are classified as anomalies.

6. Methods

6.1. Dataset preparation

One common strategy employed to work with extensive time series data, is the *sliding* window approach. A window size w is fixed or decided during validation, and the model takes the samples falling in the window as inputs. This window then moves forward by 1 sample, and the process repeats, explicitly representing sequential patterns and temporal dependencies among consecutive observations. A major issue encountered with this method, is the enormous amount of new input windows created, that can slow down learning on big dataset with low hardware capabilities. For a time series of length L a total of $L - w + 1$ input frames are created, with lot of redundancy caused by samples that are shared between consecutive windows. To address so, we relied on *strided overlapping* windows, where each segments slides more than 1 single sample when building the next input section. An overlapping percentage, or a stride parameter, is prefixed to decide the amount of shared points between successive windows. At the end, with a stride s, a total of $\frac{L-w}{s} + 1$ input frames are obtained, still achieving data augmentation and capturing temporal relationships but with less magnitude burden and reduced redundancy. Both the *sliding* and *strided* window approach were tested, finding zero performance degradation when using the latter, even if it produces less data, therefore being more efficient and preferable for our case.

In addition to the standard preprocessing procedure released by ESA (explained in sec. 4), we performed a quick analysis on the channels, finding few of them, usually belonging to the non-target group, being almost constant and equal to 0 with rare peaks to 1. We decided to filter the preprocessed data and remove those channels since we believe they carry minimal information. The training data is successively prepared by taking slices of the entire time series that *do not* contain anomalies, of a minimal length equal to the window size, so that at least 1 input frame can be created by the means written above. This is important since we don't want the model to learn a distribution poisoned with anomalous points.

6.2. Variational Recurrent Autoencoder

Encoder The Encoder Block combines a Bidirectional LSTM (Bi-LSTM) and multiple Multi-Head Self-Attention (MHSA) layers (both explained in details in Section 5) to encode input time series data. The input to the encoder is a window of data, namely:

$$X \in \mathbb{R}^{w \times (tc+nc)},$$

where w is the window size, tc denotes the number of target channels and nc the number of non-target channels. These channels are concatenated along the feature dimension. The encoder first passes the input window X through a Bidirectional LSTM with two stacked layers:

$$H_{\text{BiLSTM}} = \text{BiLSTM}(X), \quad H_{\text{BiLSTM}} \in \mathbb{R}^{w \times 2h},$$

where h is the hidden dimension of a single-directional LSTM. Due to the bidirectionality, the hidden dimension doubles to $2h$. A linear layer then projects these outputs back down to a single hidden dimension:

$$H_{\text{proj}} = \text{Linear}(H_{\text{BiLSTM}}), \quad H_{\text{proj}} \in \mathbb{R}^{w \times h}.$$

Next, the projected output H_{proj} is processed by a sequence of Multi-Head Self-Attention layers. Each MHSA layer is followed by a residual connection and layer normalization (Add & Norm):

$$\hat{H}^{(l)}_{\text{attn}} = \text{LayerNorm}(H^{(l-1)} + \text{MHSA}(H^{(l-1)})),$$

$$\hat{H}^{(l)}_{\text{attn}} \in \mathbb{R}^{w \times h}.$$

where l denotes the index of the attention layer, with the initial $H^{(0)} = H_{\text{proj}}$. Following the first normalization, the output passes through a position-wise feedforward layer, consisting of two linear transformations separated by a ReLU activation and dropout:

$$H^{(l)}_{\text{FF}} = \text{Linear}\left(\text{Dropout}\left(\text{ReLU}\left(\text{Linear}(\hat{H}^{(l)}_{\text{attn}})\right)\right)\right),$$

$$H^{(l)}_{\text{FF}} \in \mathbb{R}^{w \times h}.$$

Another residual connection and a second layer normalization are applied after the feedforward network:

$$H^{(l)} = \text{LayerNorm}(\hat{H}^{(l)}_{\text{attn}} + H^{(l)}_{\text{FF}}), \quad H^{(l)} \in \mathbb{R}^{w \times h}.$$

This process repeats for each MHSA layer. In our implementation, we use three MHSA layers, with the final MHSA layer's output dimension being out instead of h. Finally, the output of the Encoder Block is transformed into latent-space parameters using two independent linear layers:

$$\mu = \text{Linear}(H^{(L)}), \quad \mu \in \mathbb{R}^{w \times J}, \tag{1}$$

$$\log \sigma^2 = \text{Linear}(H^{(L)}), \quad \log \sigma^2 \in \mathbb{R}^{w \times J}, \tag{2}$$

where J is the dimension of the latent representation, and $H^{(L)}$ is the output from the final MHSA layer. A general view of the proposed encoder design is depicted in Fig. 1

Decoder Mirroring the Encoder structure, the Decoder Block reverses its operations, starting from the latent representation. The decoder reconstructs the original input sequence from a latent vector Z, which is sampled from the

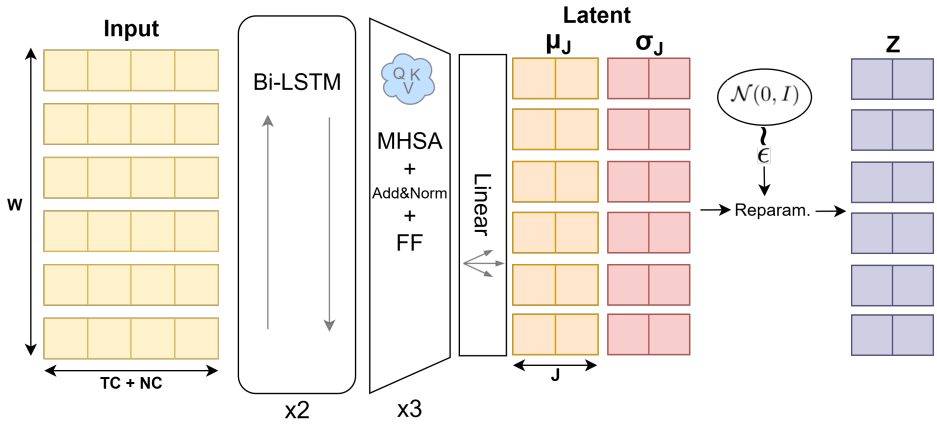

Figure 1: High-level view of the encoder architecture of the proposed VRAE

encoded latent distribution using the reparameterization trick:

$$Z = \mu + \sigma \odot \epsilon, \quad \epsilon \sim \mathcal{N}(0, I),$$

where μ and σ are the outputs of the encoder block representing the latent distribution, and ϵ is a random noise vector drawn from a standard Gaussian. Thanks to this process, there is no issue in back-propagating the gradient, which would be impossible with a normal non-differentiable sampling step. Following that, the latent vector Z is projected to match the hidden dimension h:

$$H_{\text{dec, proj}} = \text{Linear}(Z), \quad H_{\text{dec, proj}} \in \mathbb{R}^{w \times h}.$$

This projected vector then undergoes multiple Multi-Head Self-Attention layers. Each MHSA layer includes residual connections and layer normalization similar to the encoder:

$$H^{(l)}_{\text{dec, attn}} = \text{MHSA}(H^{(l-1)}_{\text{dec, attn}}), \quad l = 1, \ldots, L.$$

Following the MHSA sequence, a two-layer stacked Bidirectional LSTM decodes the resulting sequence:

$$H_{\text{dec, lstm}} = \text{BiLSTM}(H^{(L)}_{\text{dec, attn}}), \quad H_{\text{dec, lstm}} \in \mathbb{R}^{w \times 2h}.$$

Due to bidirectionality, the hidden dimension is doubled and a linear layer projects it down to the dimension corresponding to the target channels:

$$\hat{X}_{\text{target}} = \text{Linear}(H_{\text{dec, lstm}}), \quad \hat{X}_{\text{target}} \in \mathbb{R}^{w \times tc},$$

This represents the reconstructed target-channels of the input time-series, against which we can compute the reconstruction error.

Loss Function The model is trained using a loss function that combines reconstruction and regularization terms. Given a batch consisting of target channels X_{tc} and non-target channels X_{nc}, the input to the model is constructed by concatenating these along the feature dimension:

$$X_{\text{input}} = \text{Concat}(X_{tc}, X_{nc}), \quad X_{\text{input}} \in \mathbb{R}^{w \times (tc+nc)}.$$

The Variational Autoencoder (VAE)-based model outputs the reconstruction $\hat{X}_{tc}$ and parameters μ, σ of the latent distribution $q(z|X)$. The loss is computed as follows:

$$\mathcal{L} = \underbrace{\text{MAE}(X_{tc}, \hat{X}_{tc})}_{\text{Reconstruction Loss}} + \beta \cdot \underbrace{D_{\text{KL}}\left(q(z|X) \| p(z)\right)}_{\text{Regularization Loss}},$$

where the reconstruction loss is the Mean Absolute Error (MAE), defined as:

$$\text{MAE}(X_{\text{target}}, \hat{X}_{\text{target}}) = \frac{1}{N} \sum_{i=1}^{N} |X_{\text{target},i} - \hat{X}_{\text{target},i}|,$$

and the regularization term is the KL divergence between the latent distribution $q(z|X) \sim \mathcal{N}(\mu, \sigma^2)$ and the prior $p(z) \sim \mathcal{N}(0, I)$:

$$D_{\text{KL}}\left(q(z|X) \| p(z)\right) = -\frac{1}{2} \sum_{j=1}^{J} \left(1 + \log \sigma_j^2 - \mu_j^2 - \sigma_j^2\right).$$

The hyperparameter β controls the trade-off between reconstruction accuracy and latent distribution regularization. It was introduced because the KL divergence tended to decrease rapidly toward zero, while the reconstruction loss decreased more gradually. During training, both losses are logged separately for monitoring purposes.

6.3. Thresholding strategies

During inference in a real world scenario, the VRAE takes as input the last w consecutive data points of the incoming flow, encodes it in the lower-dimensional latent space, samples from it and the decoder tries to reconstruct the input time series as close as possible. Assuming that it has being trained on nominal data, the reconstruction fails when dealing with anomalous inputs, hence we expect a great difference between the input and the output whenever an anomaly occurs. In this work, we measure the Absolute Error between each channel at each time-steps, in agreement with the chosen reconstruction loss during training (l1-loss), giving $e_c(t) = |x_c(t) - \hat{x}_c(t)|$ To decide whether an error indicates an anomaly, we implemented and tested various thresholding methods:

- **NDT**: Non-Parametric Dynamic Thresholding, discussed in details in Sec. 3. We first smooth out the error sequence through EWMA then we calculate the score function for each candidate z. The optimal z values are found freshly at each input window during testing, in order to capture local evolutions.
- **Local Error STD**: We compute the error mean $\mu(t) \in \mathbb{R}^c$ and standard deviation $\sigma(t) \in \mathbb{R}^c$ of

each channel c of the current window. These statistics represents the errror distribution at the current point in time. Every sample in the input window whose error exceeds the mean by N STDs is flagged as an anomaly. We tested both $N = 3$ and $N = 5$.

- **Global Error STD**: We run the model over the entire training/validation set, and compute the reconstruction error means and standard deviations considering all points of all processed windows. Those parameters are saved and remains static during inference, since they represent the overall nominal error distribution, i.e how the model naturally perform on expected data. In testing, we use those statistics to threshold the errors, flagging channel c at time t as an anomaly if $e_c(t) > \mu_c + N \cdot \sigma_c$. As before, we tried both values of 3 and 5 for N.
- **Fixed Thresholds**: As above, we make a full run over the entire validation set keeping all reconstruction errors. Now, we examine 30 threshold values separately for each channel, linearly spaced between the minimum and maximum error for that channel in the validation set. We then pick the threshold that maximize the F-0.5 score among all others. At the end of the procedure, we retain a unique threshold value for each channel. If a channel in the validation set does not contain anomalies, the F-0.5 score cannot be computed, hence we set the threshold to be slightly over the maximum error of that channel.

Each thresholding strategy explores different trade-offs between adaptability and simplicity: dynamic approaches like NDT and Local Error STD adapt thresholds to local or recent data, potentially capturing subtle anomaly patterns, while global methods such as Global Error STD and Fixed Thresholds leverage comprehensive offline analysis, favoring stability and ease of deployment. Lastly, according to the fact that anomalies are usually segments lasting more than 1 time step, we tested a simple *expansion* method on the output labels: every time a negative is predicted, but it is between two predicted anomalous points, we convert its prediction to an anomaly. For instance, the output sequence 01010010 becomes 01110010, where we indicate with 1 and 0 the output of the model (1 = anomaly, 0 = nominal). Empirically, we found that there is a tiny decrease on the evaluation metrics when using this approach with respect to simply taking the output as it is, therefore we discarded it at the end.

7. Metrics

Since anomalies are very few with respect to the totality of input points, *accuracy* does not represent a meaningful measure, indeed *precision*, *recall*, *F-1* and *F-0.5* scores are the most chosen ones in the literature. We measured these metrics on the test set, but of 2 different kinds: point-wise and sequence-aware. For point-wise metrics we simply consider each timestep independently and compare it with the ground truth, counting True Positives (TP), False Positive (FP) and False Negatives (FN) separately for each channel. Then the cited values are computed as:

$$\text{Precision} = \frac{TP}{TP + FP}, \qquad \text{Recall} = \frac{TP}{TP + FN},$$

$$F\text{-}\beta = \frac{(1 + \beta)^2 \cdot (\text{precision} \cdot \text{recall})}{(\beta^2 \cdot \text{precision}) + \text{recall}}$$

We end up with independent metrics for each channel, that is extremely useful for pointing out particular channels on which the system struggles or fails, while understanding what features are instead more predictable. To have a general overview of the model capabilities, we reported the mean across all channels of these measures. For sequence-aware metrics, we take into consideration that usually anomalies are segments including several consecutive points, that reflects unexpected events happening in space lasting many minutes. As a consequence, according to ESA [3], it's generally not important to point out exactly the precise moment an anomaly occurs, but it's enough to flag an anomalous period that has at least some overlapping with a real anomaly. Then, it doesn't take much for trained engineers to analyze the surrounding values in time and check where does the anomaly actually start. For this reason, the literature proposed a new way of counting positives and negatives to give a better representation of the system capabilities in a real-world setting, where the elements to be compared are *sequences* rather than single points:

- For each labeled anomaly sequence (i.e. a contiguous range of points labeled with 1s), a **true positive** is counted if there is an overlap of at least 1 time-step with a predicted anomaly sequence.
- Similary, a **false negative** is found if in the anomaly sequence there are no overlaps with any predicted sequence, i.e. each point is flagged as nominal (0) by the system.
- Lastly, a **false positive** is counted if a predicted anomalous sequence have no overlap with any real labeled positive sequence.

The metrics are then computed with the same formulas shown before.

8. Training

The model was deployed in PyTorch and trained on a single GPU using Adam Optimizer with beta values of 0.9 and 0.999. The learning rate was set to an initial value of 0.001, and it was subjected to a Cosine Annealing scheduler reaching a minimum value of $5 \cdot 10^{-5}$. The hidden dimension was set to $h = 64$ and the latent dimension to $J = 28$, the number of Bi-LSTM stacked layers to 2 and we used 4 heads in the Multi-Head attention layers. The KLD weight in the loss was set to $\beta = 0.003$ and remained constant over the all training course. The model was trained for 5 epochs, with a batch size = 128, and the dropout probability fixed to 0.2 for the affected layers. A summary of the principal training parameters is presented in Table 3. The first 75% of the dataset was taken as training set while the remaining 25% as test set. For the thresholding schemes that required a validation set, we used the training set as such, but without eliminating anomalies from it. At the end of the training loop, the model had a final reconstruction loss around 0.002 and a KLD loss around 10^{-5}, showing that is highly capable of reconstructing the input time series. Each loss was monitored and logged during training to figure out eventual problems or model collapses.

9. Results

The comparative performance of the different thresholding methods is summarized in Table 2. We can find also the

Table 2
Performance comparison of thresholding strategies and final benchmark on the test set.

Thresholding Method	pw-precision	pw-recall	pw-F1	pw-F0.5	sa-precision	sa-recall	sa-F1	sa-F0.5
Global Error STD(3)	0.716	**0.674**	**0.685**	0.673	0.367	**0.754**	0.450	0.390
Global Error STD(5)	**0.775**	0.582	0.682	**0.677**	**0.631**	0.669	**0.593**	**0.590**
Local Error STD(3)	0.145	0.007	0.012	0.026	0.142	0.340	0.178	0.152
Local Error STD(5)	0.132	0.003	0.117	0.116	0.131	0.173	0.116	0.117
NDT	0.008	0.65	0.015	0.012	0.009	0.55	0.018	0.011
Fixed	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
DC-VAE	0.342	0.551	0.461	0.399	0.380	0.449	0.420	0.401

F-0.5 score since it is noted by ESA experts that is preferable to have less false positives than false negatives, since every time a positive is flagged, it triggers an alarm that requires space experts to analyze the situation and manually find the problem. Wince this manual effort is costly, we care more about the precision score instead of recall. Among the tested approaches, Global Error STD (with N=3) achieved the highest point-wise (pw) recall (0.674) and F1-score (0.685), indicating its effectiveness in capturing anomalies at individual time points. However, Global Error STD (N=5) exhibited superior precision (0.775) and an optimal balance reflected in the highest F0.5-score (0.677), demonstrating greater reliability in reducing false positives. This is an expected outcome since a higher value of N enlarges the "acceptable" interval of nominal points, therefore reducing false claims.

Regarding sequence-aware (*sa*) metrics, Global Error STD (N=3) showed the highest recall (0.754) but notably lower precision (0.367), leading to a moderate *sa* F1-score (0.450). On the other hand, Global Error STD (N=5) provided significantly better *sa* precision (0.631), F1-score (0.593), and F0.5-score (0.595), proving its balanced performance in identifying anomalous events lasting several minutes.

Local Error STD methods (N=3 and N=5) performed poorly across all metrics, showing low precision, recall, and F1 scores, with Local Error STD (N=5) displaying slightly better F1 and F0.5 scores compared to N=3. This is probably due to the fact that the statistics used to threshold the data points are based on the errors of the current window, that very small length with respect to the whole dataset, hence producing a worse estimates of the true error distribution of the model. The Non-parametric Dynamic Thresholding (NDT) method achieved relatively high recall but was severely limited by its low precision, leading to poor F1 and F0.5 scores. As noted in section 6.3 this is due to a great amount of false positive detections, since the method must choose some data points as anomalous even if the input window is completely normal. It may work better with a larger window size. The Fixed Thresholding method failed entirely, resulting in zero across all metrics. This is because the validation set utilized to find the optimal threshold values probably have different anomalous channels with respect to the one in the test set, therefore resulting in meaningless threshold numbers. It may work better with the whole dataset, having a validation set and test set large enough to have most of the channelsin both with at least 1 anomaly. Overall, Global Error STD (N=5) emerged as the most balanced and practical approach for anomaly detection in this setting, providing an advantageous trade-off between precision and recall, both at point-wise and sequence-aware evaluation levels. You can see also the reported result by DC-VAE taken from the original paper, that the proposed system outperforms.

10. Conclusion

This work introduced a Variational Recurrent Autoencoder framework designed specifically for anomaly detection in satellite telemetry data. Through systematic evaluation, the Global Error STD thresholding approach with N=5 emerged as the most effective method, providing an optimal balance between precision and recall. By leveraging advanced recurrent architectures and effective thresholding strategies, the presented model successfully demonstrated improved detection performance over baseline approaches, highlighting its suitability and efficiency for real-world satellite telemetry monitoring tasks.

Parameter	Value
Epochs	5
Batch Size	128
w	64
h	64
J	28
β	0.003
Dropout	0.2
Initial l.r	0.001
LSTM layers	2
MHSA layers	3
Dataset split	75-25

Table 3
Summary of training hyperparameters

References

[1] J. M. Heras, A. Donati, Enhanced telemetry monitoring with novelty detection, AI Magazine 35 (2014) 37–46. doi:https://doi.org/10.1609/aimag.v35i4.2553.

[2] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, T. Soderstrom, Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery amp; Data Mining, KDD '18, ACM, 2018, p. 387–395. URL: http://dx.doi.org/10.1145/3219819.3219845. doi:10.1145/3219819.3219845.

[3] K. Kotowski, C. Haskamp, J. Andrzejewski, B. Ruszczak, J. Nalepa, D. Lakey, P. Collins, A. Kolmas, M. Bartesaghi, J. Martinez-Heras, G. D. Canio, European space agency benchmark for anomaly detection in satellite telemetry, 2024. URL: https://arxiv.org/abs/2406.17826. arXiv:2406.17826.

[4] G. G. González, S. M. Tagliafico, A. F. Iie-Fing, G. Gómez,

J. Acuña, P. Casas, Dc-vae, fine-grained anomaly detection in multivariate time-series with dilated convolutions and variational auto encoders, in: 2022 IEEE European Symposium on Security and Privacy Workshops (EuroSPW), 2022, pp. 287–293. doi:10.1109/EuroSPW55150.2022.00035.

[5] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (1997) 1735–1780. URL: https://doi.org/10.1162/neco.1997.9.8.1735. doi:10.1162/neco.1997.9.8.1735.

[6] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin, Attention is all you need, 2023. URL: https://arxiv.org/abs/1706.03762. arXiv:1706.03762.

[7] D. P. Kingma, M. Welling, Auto-encoding variational bayes, 2022. URL: https://arxiv.org/abs/1312.6114. arXiv:1312.6114.