22-07-2026

Anomaly Detection in perfSONAR Data Using Autoencoders

Grant Agreement No.:	101194278
Work Package:	WP6
Task Item:	T3
Document Type:	White Paper
Dissemination Level:	PU
Lead Partner:	CARNET
Document ID:	GN5-2-26-FF794249

Authors:	Ljubomir Hrboka (CARNET); Krešimir Šodan (CARNET)

Abstract

This paper presents an approach to anomaly detection in network performance data collected through the perfSONAR-based Performance Measurement Platform (PMP), in GN5-2 Work Package 6, Task 3. Using unsupervised machine learning techniques, specifically autoencoders, the study focuses on identifying deviations in latency distribution without relying on labelled datasets. Two approaches are evaluated: direct histogram-based learning and statistical feature extraction. Results demonstrate that the first proposed method successfully detects both transient anomalies and sustained distribution shifts, highlighting its potential for real-time network monitoring.

© GÉANT Association on behalf of the GN5-2 project. The research leading to these results has received funding from the European Union's Horizon Europe research and innovation programme under Grant Agreement No. 101194278 (GN5-2).

Co-funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union. The European Union cannot be held responsible for them.

Contents

Figures

Executive Summary

This document, developed within Work Package 6 (WP6), Task 3 (T3) of the GN5-2 project, presents an approach for detecting anomalies in perfSONAR network performance data using autoencoders. The goal is to improve upon traditional threshold-based monitoring, which often misses subtle or gradually developing issues.

The proposed method learns normal latency patterns and identifies deviations by analysing the full distribution of measurements rather than relying on simple metrics. Two data representations were evaluated, with histogram-based input proving more effective, as it preserves the structure of latency distribution. Results on real and artificially generated synthetic data show that this model can detect both sudden spikes and longer-term changes in network behaviour, including cases where traditional metrics remain stable. This method achieves high precision with few false positives, although smaller anomalies may sometimes be missed.

Overall, the results demonstrate that the proposed approach is effective in detecting complex changes in network performance and provides a solid foundation for enhancing anomaly detection in monitoring systems.

1 Introduction

Modern research and education networks rely heavily on continuous performance monitoring to ensure reliability and efficiency. perfSONAR provides a distributed framework for network measurement and monitoring, collecting metrics such as latency, jitter and throughput. These measurements are aggregated within the Performance Measurement Platform (PMP) in WP6 T3 of the GN5-2 project.

Traditional monitoring systems rely on static thresholds, which are often not reliable for detecting subtle anomalies or gradual performance degradation. Machine learning, particularly unsupervised models, offers a promising alternative by learning normal behaviour directly from collected data.

This research explores the application of autoencoder-based anomaly detection on perfSONAR one-way delay data, with the goal of identifying both transient anomalies and long-term deviations in network performance.

2 Background and Motivation

The Performance Measurement Platform is intended for network engineers, NOC operators, researchers and participants from National Research and Education Networks (NRENs) to monitor, analyse, and better understand network performance across multi-domain environments. The platform consists of approximately 50 distributed measurement points equipped with perfSONAR that regularly perform measurements towards selected measurement points located within the core of the GÉANT network. The collected performance data is stored in a central archive, creating a comprehensive database of historical measurements.

Network latency is a critical factor in many applications, particularly in video conferencing and other real-time communication systems, as it directly influences service quality and responsiveness. Ideally, latency between two nodes should remain stable over time. However, in practice, this varies due to factors such as network congestion, hardware issues or changes in routing paths. Minor variations are expected and are typically handled by modern applications, but larger fluctuations can significantly degrade overall performance.

Under normal conditions, distribution remains relatively stable over time. However, network issues such as congestion, routing changes, or hardware problems can introduce:

- Sudden spikes in latency
- Sustained shifts in baseline latency
- Increased variance or distribution skewness

Detecting such anomalies early is beneficial for maintaining network performance and supporting latency-sensitive applications. A major challenge in this respect is the lack of labelled anomaly data, which prevents the use of supervised machine learning methods. Therefore, this work focuses on unsupervised learning approaches that can detect deviations from established patterns of normal behaviour, such as the baseline latency shift shown in Figure 2.1.

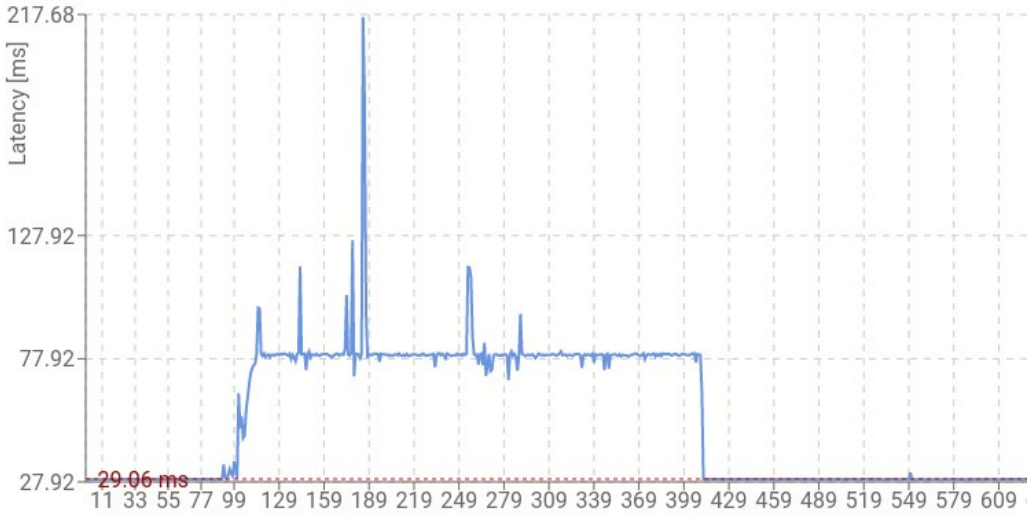

Figure 2.1 Example of shift in baseline latency

3 Data Exploration and Preprocessing

The analysis carried out began with an exploratory phase aimed at understanding the relationships between different network performance metrics and identifying suitable data sources for anomaly detection. Based on these findings, the focus shifted toward latency measurements and the preparation of a consistent, high-quality dataset for machine learning.

3.1 Exploratory Data Analysis

During the exploratory data analysis phase, the T3 team first examined whether latency and jitter measurements could be correlated with anomalies observed in throughput. Selected throughput events were analysed by extracting jitter values within a defined time window around each anomaly, enabling a temporal comparison between the metrics to identify potential relationships.

This analysis showed that, while some events did not exhibit any clear correlation, others revealed noticeable spikes in jitter occurring immediately after throughput anomalies. These findings suggest that in certain cases different network performance metrics may reflect related underlying events, highlighting the value of cross-metric analysis during the initial data exploration stage. This also indicates the potential benefit of further investigating how such

correlations could support NOC operations, for example by introducing features within perfSONAR that enable correlation across different measurement types.

However, due to the limited availability and low frequency of throughput measurements, this approach could not be applied consistently across the dataset. As a result, the focus was shifted to latency-based measurements, which provided a significantly larger and more continuous dataset for further analysis.

Latency, specifically one-way delay (OWD), is a key network performance indicator, reflecting the time it takes for packets to travel from source to destination and providing insight into overall network conditions. Variations in delay can indicate congestion, routing changes, queuing effects, or transient faults, while a certain level of delay is expected under normal network operation.

3.2 Data Source

The dataset used in this research consists of perfSONAR one-way delay measurements which were chosen because of the large volume of available data. Those measurements in PMP are collected as histograms containing 600 latency values every minute, representing the distribution of observed delay values by grouping them into bins and showing how frequently values fall within each bin.

An example of such a histogram is shown in Figure 3.1., illustrating the typical distribution of latency values within a single measurement interval.

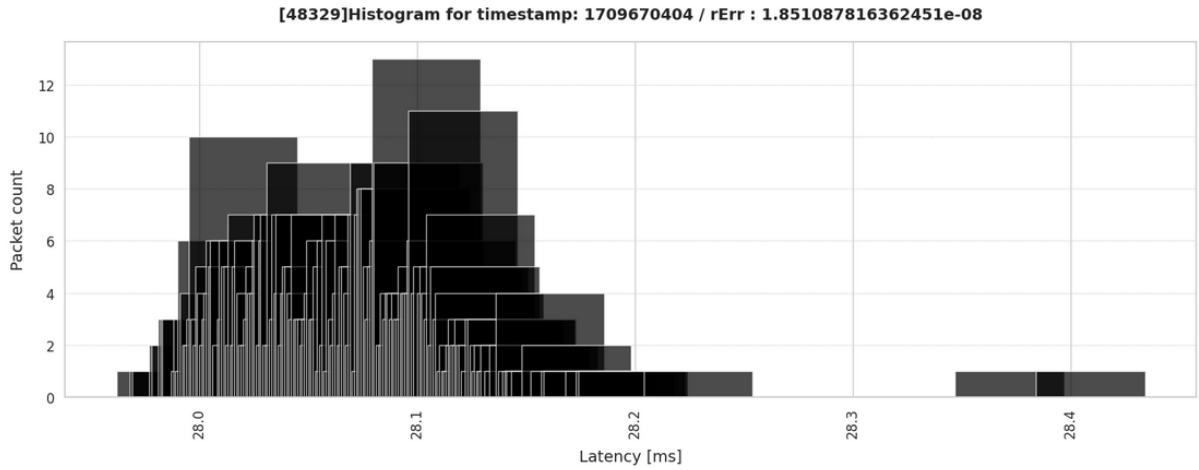

Figure 3.1 Histogram example from measurement data

3.3 Preprocessing Steps

To ensure that the collected data could be effectively utilised by a machine learning model, a series of preprocessing steps were performed. These steps were designed to standardise the input format, improve data quality and enhance the model's ability to learn meaningful patterns.

- Histogram Resampling: The original histogram data, which varies in number of bins, was transformed into uniform representation. Each histogram was resampled into a

fixed-size vector of 256 bins. This allows the model to process all inputs consistently, eliminating variability caused by differing histogram resolutions.

- Normalisation: After resampling, all histogram values were scaled to a common numerical range of [0,1]. This normalisation step is crucial for stabilising the training process, as it prevents features with larger magnitudes from dominating the learning process. It also improves convergence speed and overall model performance by ensuring that all input features contribute proportionally.
- Noise Handling: Measurement data often contains noise or inconsistencies caused by measurement errors. To address this, basic noise reduction techniques were applied, primarily through normalisation and aggregation. These methods help smooth irregularities and reduce the impact of anomalous values resulting in cleaner and more reliable input data.

Overall, this preprocessing pipeline ensures that the input data is consistent, comparable, and well-structured. By reducing variability and noise while standardising feature representation, the model is better equipped to learn robust and generalisable patterns across different measurement links.

Figure 3.2 shows an example of a resampled and normalised histogram represented as a fixed-size tensor. Although displayed in multiple rows for readability, the values correspond to a single histogram encoded as a 256-dimensional vector.

```
dataTensor=torch.tensor(df.values)
print(dataTensor[0])
```

```
tensor([0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.4750, 0.5200, 0.0017, 0.0017, 0.0000, 0.0000, 0.0017,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000, 0.0000,
        0.0000, 0.0000, 0.0000, 0.0000])
```

Figure 3.2 Example of resampled and normalised histogram

4 Methodology

An unsupervised learning approach and an autoencoder-based model were used for anomaly detection, including the selection and evaluation of different input data representations and the rationale behind the final model configuration.

4.1 Unsupervised Learning Approach

Due to the lack of labelled data, an unsupervised learning approach was adopted. The model is trained exclusively on data representing normal network behaviour. Including all data in the training process, particularly anomalous samples, would reduce the model's ability to distinguish between normal and abnormal patterns, leading to less reliable anomaly detection results.

4.2 Autoencoder Model

An autoencoder neural network was used for anomaly detection. The model consists of:

- Encoder: Compresses input data into a lower-dimensional representation
- Decoder: Reconstructs the original input

The model is trained to minimise reconstruction errors. During inference, anomalies are identified as inputs with high reconstruction error, indicating deviation from learned normal patterns.

4.3 Input Data Representation and Feature Selection

Two input representations were considered for the autoencoder-based anomaly detection model. The first approach used normalised 256-bin histograms as input to the model. This representation preserves the latency distribution and retains information about spread, asymmetry and structural properties of the measurements Such properties are particularly relevant in perfSONAR measurements, where anomalies may appear not only as simple shifts in central tendency, but also as changes in the overall distributional shape.

The second approach used a set of statistical descriptors extracted from each histogram, including mean, variance, skewness, kurtosis, entropy, and selected percentiles. This representation has the advantage of lower dimensionality and improved interpretability, since each feature has a clear statistical meaning.

However, feature extraction may discard fine-grained structural information of data. In the evaluated data, this reduction was found to weaken the model’s sensitivity to certain distributional changes, particularly in cases where anomalies were reflected in subtle changes in histogram shape rather than in large changes of summary statistics alone.

For this reason, the final model configuration was based on histogram input. Although the histogram-based representation is higher-dimensional and potentially more sensitive to noise, it preserved the complete latency distribution and, in practice, produced more discriminative reconstruction-error patterns in the analysed examples. This made it more suitable for detecting both sustained distribution shifts and short-lived irregularities in perfSONAR latency measurements.

5 Analysis and Results

The setup used in this study consists of server-running scripts that continuously retrieve perfSONAR one-way delay data from the perfSONAR archive at one-minute intervals. The data is then processed and passed on to a trained autoencoder model, which calculates reconstruction error as an indicator of anomalous behaviour. These reconstruction error values are stored in an InfluxDB time-series database and visualised through Grafana dashboards within the nmaas infrastructure, enabling near real-time monitoring and analysis of network performance.

5.1 Detection of Transient Anomalies

The model successfully detects short-term anomalies, such as sudden latency spikes. These events are reflected in sharp increases in reconstruction error.

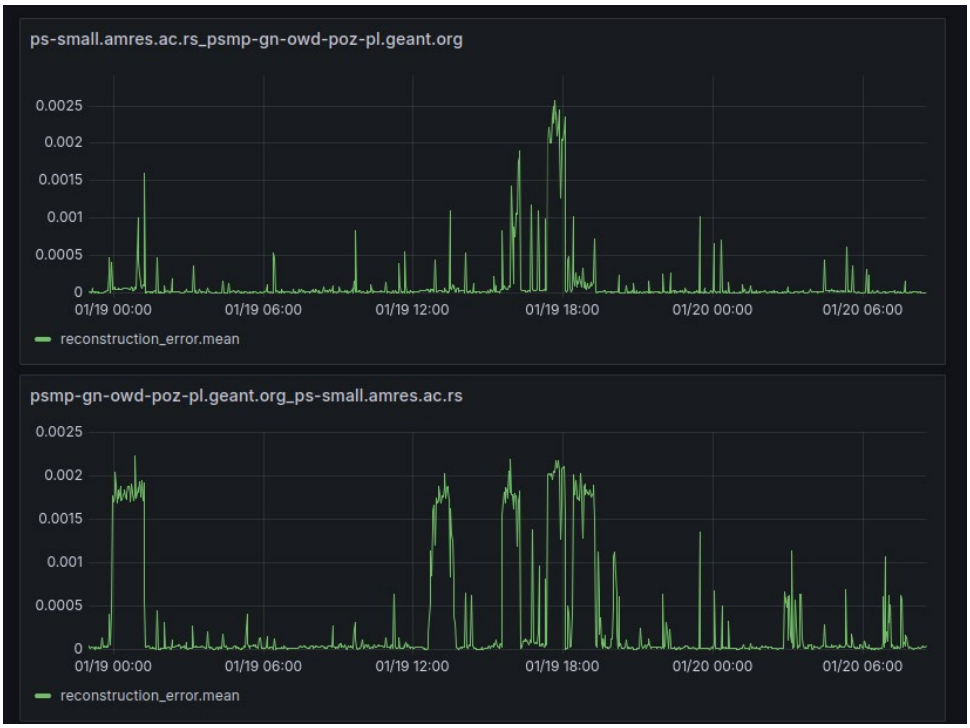

Figure 5.1 Reconstruction error for both measurement directions showing multiple short detected anomaly spikes

Figure 5.2 Reconstruction OWD latency measurements, where only minor fluctuations are visible

Figures 5.1 and 5.2 present an example of transient anomalies detected by the autoencoder for two opposite measurement directions. In Figure 5.1, the upper plot shows the reconstruction error for the direction from ps-small.amres.ac.rs to psmp-gn-owd-poz-pl.geant.org, while the lower plot shows the reverse direction. Figure 5.2 presents the corresponding perfSONAR latency measurements in the same order, with the upper latency plot matching the upper reconstruction-error plot and the lower latency plot matching the lower reconstruction-error plot.

For the upper direction, several short-lived increases in reconstruction error are visible, with the most pronounced activity occurring later in the observed interval. These peaks correspond closely to visible changes in the latency plot, including a clear latency spike and preceding smaller variations. A short latency drop is visible before the large spike, indicating a rapid change in network behaviour. Additional operational evidence supports this interpretation as traceroute measurements recorded during the same interval show a change in hop count, as illustrated in Figure 5.3. This temporal correlation suggests that the detected anomaly was associated with a path-level network event, such as a route change, rather than with ordinary latency fluctuation alone.

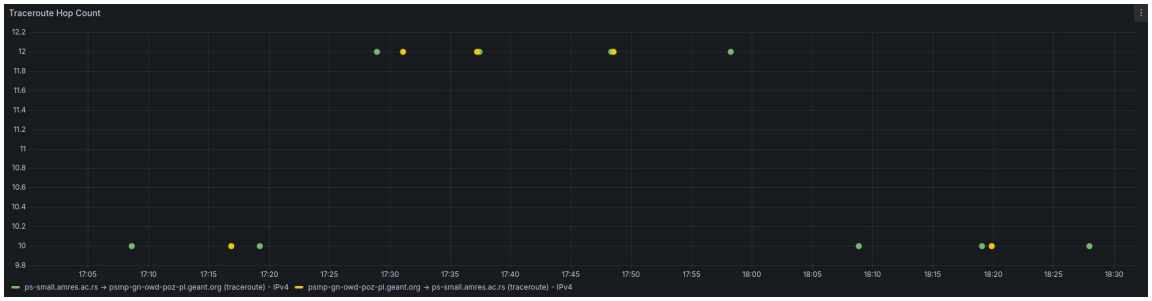

Figure 5.3 Traceroute hop-count variation observed during the same interval as the detected transient anomaly

For the lower direction, the reconstruction-error pattern is different: in this case, longer periods of elevated reconstruction error are observed, indicating that the deviation is not limited to isolated spikes but persists over a broader time interval. The corresponding lower latency plot shows a more sustained change in delay behaviour, including a pronounced latency increase and additional variations before and after the main event. This suggests that the model captures a more prolonged disturbance in the reverse direction.

At the same time, not every increase in reconstruction error has an immediately obvious counterpart in the median or percentile latency curves. Some anomaly responses occur during intervals where the standard latency metrics appear relatively stable. This suggests that the model is not limited to identifying only visually obvious spikes. Instead, it is also sensitive to more subtle changes in the statistical structure of the measurements, such as temporary increases in variability, less stable timing, or changes in the overall distribution of delay values.

The observed directional difference also reinforces the importance of analysing forward and reverse measurements separately. Although both directions show anomaly responses, the amplitude, timing and persistence of reconstruction-error peaks differ, indicating that the same operational event may affect the two directions differently.

5.2 Detection of Sustained Anomalies

In addition to transient spikes, the model identifies prolonged deviations in latency, including step changes in baseline latency and extended periods of increased delay. These events correspond to increases in reconstruction error, demonstrating the model's ability to capture distribution shifts.

Figure 5.4 Reconstruction error for both measurement directions between two perfSONAR nodes. Prolonged anomalous event is observed

Figure 5.5 perfSONAR OWD measurements for the same link and time period, showing a step increase in base latency

Figures 5.4 and 5.5 present a real-world example of prolonged anomalous behaviour observed between two perfSONAR nodes. Figure 5.4 shows the reconstruction error generated by the autoencoder for both measurement directions, while Figure 5.5 shows the corresponding one-way delay measurements for the upper direction.

For the upper direction, a clear and sustained increase in reconstruction error is visible over an extended time interval. This period corresponds closely to a step change in the OWD

measurements, where the latency level rises from approximately 31 ms to around 36-37 ms and remains elevated for several hours before returning to its earlier baseline. The prolonged elevation of reconstruction error indicates that the model identifies this shifted operating state as significantly different from the normal behaviour learned during training.

Although only the upper-direction OWD plot is shown in Figure 5.5, the reconstruction-error plot in Figure 5.4 indicates that anomalous behaviour is also present in the reverse direction during a broadly similar time interval. This suggests that the observed event was not limited to a single isolated measurement path, even though the available latency visualisation in this example is shown only for one direction.

This example highlights an important strength of the proposed approach: its ability to detect sustained shifts in network behaviour rather than only short-lived transient spikes. In this case, the dominant anomaly is a prolonged change in the normal operating level of latency, which is consistently reflected in the reconstruction-error signal.

From an operational perspective, such behaviour is consistent with a sustained change in end-to-end network conditions rather than a short-lived isolated disturbance. Possible explanations include a temporary path change, altered queuing conditions, or another network-state transition affecting latency over an extended period. Although the available measurements do not permit definitive root-cause identification, the results show that the model can reliably detect these shifts as deviations from previously learned normal behaviour.

A second example of sustained anomalous behaviour is shown in Figures 5.6 and 5.7. In this case, the anomaly is characterised by a sequence of step increases in latency rather than a single shift in latency measurements.

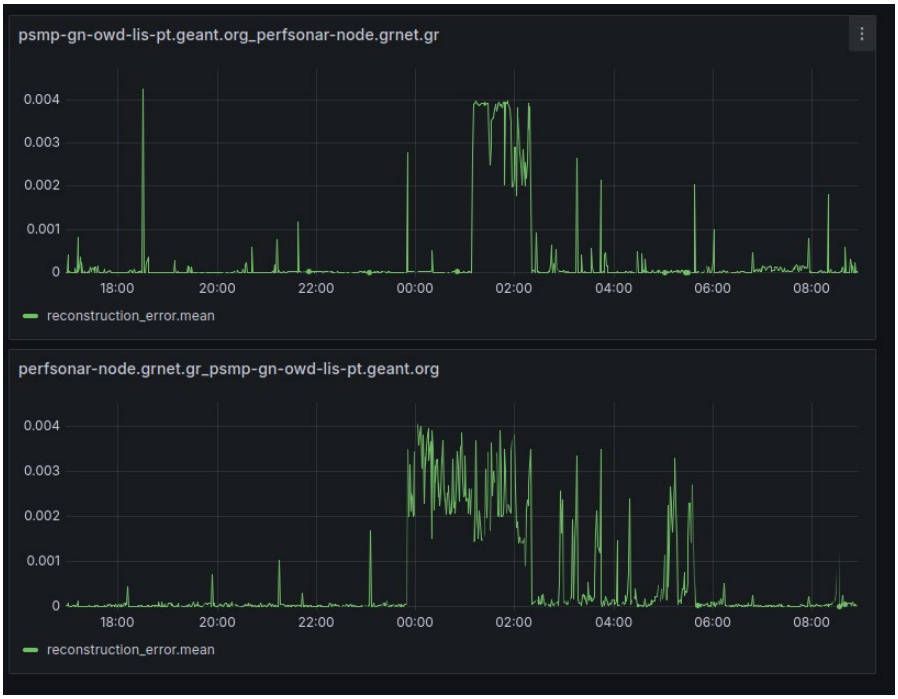

Figure 5.6 Reconstruction error for both directions showing sustained and structured anomalous behaviour with multiple elevated phases

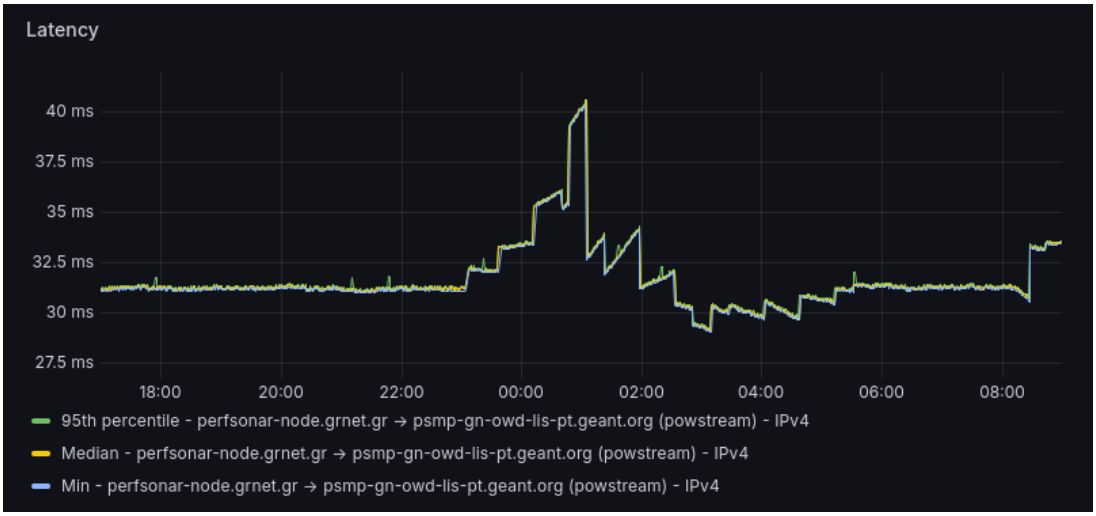

Figure 5.7 OWD latency showing stepwise increases and gradual transitions corresponding to the detected anomaly

As shown in Figure 5.7, latency gradually changes, rising from approximately 31 ms to over 40 ms before decreasing again in stages. This pattern shows dynamic changes in network conditions. The corresponding reconstruction error in Figure 5.6 follows this behaviour. An elevation in error is observed during the entire anomalous period, with higher error values corresponding to the most significant latency changes. This example displays the model's ability to detect not only static distribution shifts but also anomalies with multiple intermediate states.

Additional operational evidence supports this interpretation. As shown in Figure 5.8, a route change is also visible during the same interval through a reduction in the traceroute hop count. This temporal correlation suggests that the detected anomaly was associated with a path-level network change rather than with ordinary latency fluctuation alone.

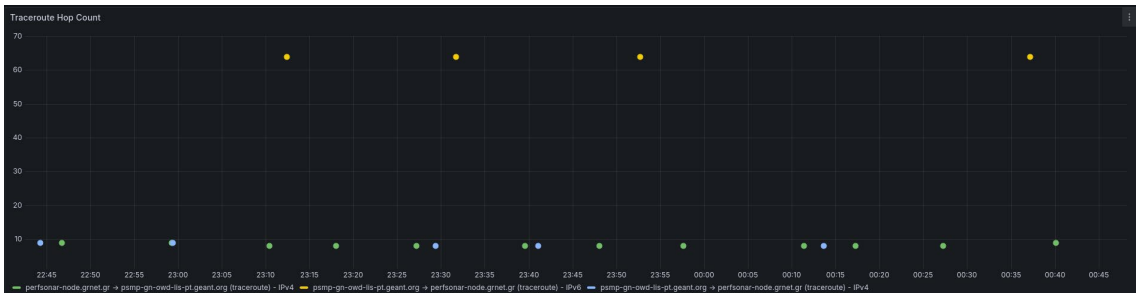

Figure 5.8 Traceroute hop-count variation during the same interval as the sustained anomaly, showing a reduction in the number of hops

These sustained anomalies also provide insight into directional behaviour. In Figures 5.4 and 5.5, both measurement directions exhibit a prolonged increase in reconstruction error corresponding to a step change in latency, but the magnitude and temporal characteristics of the anomaly are not identical. Similarly, in Figures 5.6 and 5.7, both directions reflect the same overall multi-phase pattern, although the transitions occur at different times and with different intensities.

This indicates that while anomalies affect the full path, they may manifest differently in each direction. Such behaviour is consistent with real network conditions, where forward and reverse

paths can have different properties. As a result, analysing each direction independently is essential, while partial correlation between directions provides additional confidence that the detected event reflects a network anomaly.

5.3 Directional Differences

The results presented in the previous sections show that anomaly patterns are often present in both measurement directions, but not necessarily in identical manner. While both directions may reflect the same underlying network event, differences can be observed in the magnitude, and temporal properties of reconstruction error.

This behaviour reflects real network conditions, where forward and reverse paths may traverse different routes with different network properties. As a result, anomalies may manifest asymmetrically even when they originate from the same underlying cause.

For this reason, each direction should be analysed independently to ensure accurate detection. At the same time, partial correlation between directions can provide additional confidence that the detected anomaly corresponds to a genuine network event rather than random variation or measurement noise.

5.4 Validation on Synthetic Data

The synthetic dataset was introduced primarily to provide a controlled evaluation setting in which both normal and anomalous cases were explicitly defined. This made it possible to compute a confusion matrix and related classification outcomes against a known reference. In real perfSONAR measurements, such verification is considerably more difficult, because anomalous events are often not labelled in advance and the boundary between normal and abnormal behaviour may remain ambiguous.

For this reason, artificial normal and anomalous samples were generated to reflect the general characteristics of perfSONAR measurements while still allowing an unambiguous interpretation of the model output. In this context, the confusion matrix should therefore be understood as a methodological validation tool rather than as a direct representation of operational performance on real-world data.

The role of the synthetic data in this document is to complement, not replace, the analysis of real measurements. Its main value is that it enables quantitative evaluation in a setting where the true class of each sample is known, which is not generally the case for real monitoring data without additional annotation effort. This allows the proposed approach to be assessed in a transparent and reproducible way and helps demonstrate whether the model can distinguish expected behaviour from deliberately introduced deviations under controlled conditions.

The comparison with observations from real data is intended to show that the model behaviour seen on synthetic examples is not disconnected from practical measurement patterns, but the purpose of the synthetic analysis is specifically to provide a verifiable evaluation framework for classification-oriented metrics such as the confusion matrix.

The synthetic normal data was designed to resemble typical perfSONAR measurements under non-anomalous conditions, where latency values are concentrated within a narrow range of bins, forming a compact distribution. In contrast, two types of anomalous distributions were generated. The first type consisted of distributions with multiple peaks, representing situations where latency values are split clustered around two bins. The second type was based on a Gaussian-like distribution with some values scattered across a wider range of bins, simulating increased variability and anomalous network behaviour.

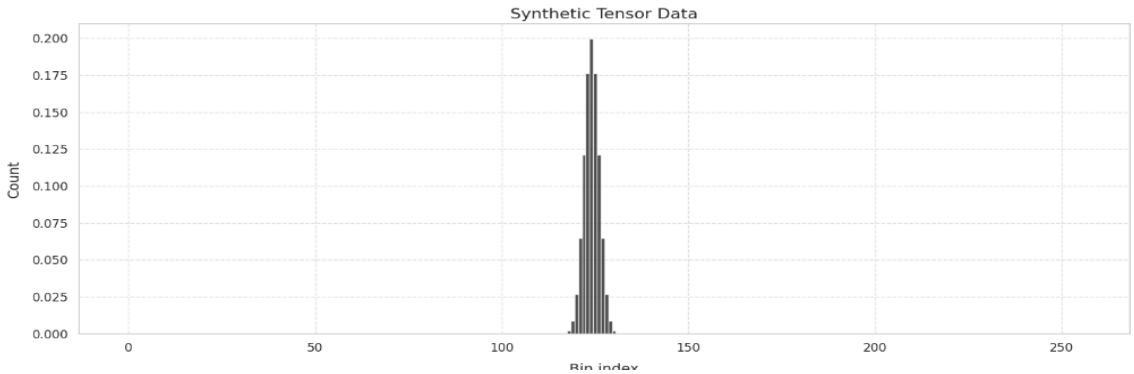

Figure 5.9 Normal synthetic data histogram

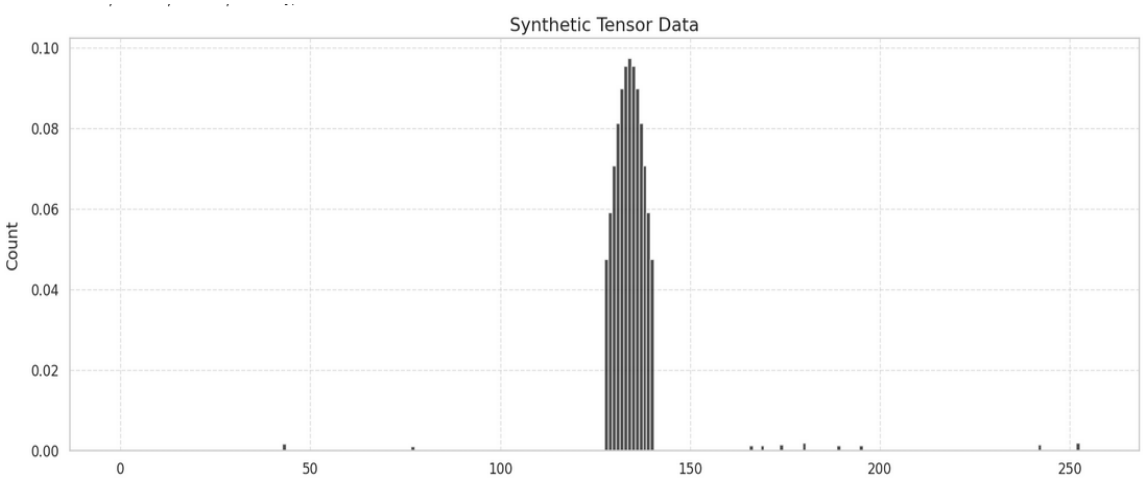

Figure 5.10 Anomalous synthetic data histogram

Figures 5.9 and 5.10 show examples of the generated normal and anomalous synthetic histograms. The normal distribution is clearly concentrated within a limited number of bins, while the anomalous example exhibits a wider spread of values across the histogram.

The model was evaluated on this synthetic dataset to assess its ability to distinguish between normal and anomalous patterns. The resulting confusion matrix is shown in Figure 5.11. The results show that the model correctly classified all normal samples, indicating high precision and no false positives in a controlled setting. However, some of the anomalous samples were misclassified as normal, reflecting moderate sensitivity.

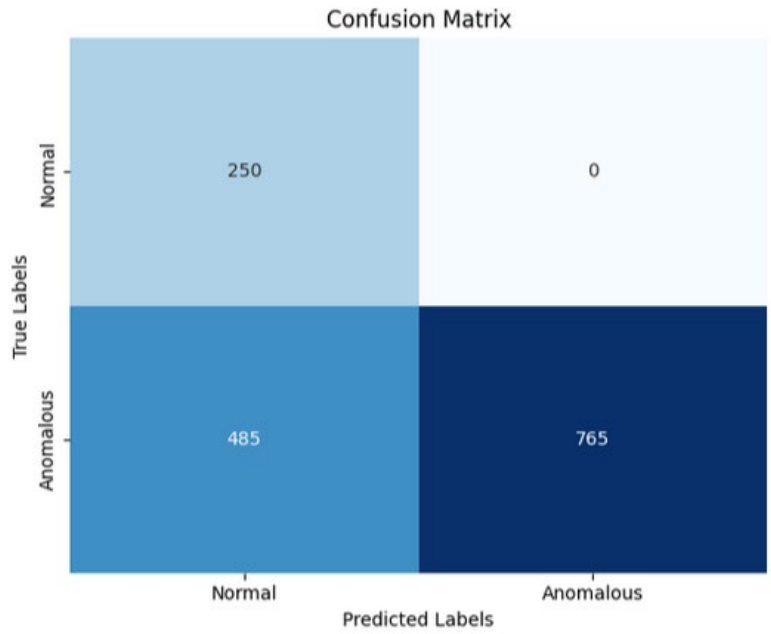

Figure 5.11 Synthetic data confusion matrix

These results are consistent with observations from real data. The model is highly reliable when identifying anomalies but may miss less pronounced or ambiguous cases. This further confirms that the histogram-based autoencoder is effective at detecting clear structural deviations in latency distributions, while smaller or more subtle anomalies may require complementary detection methods.

5.5 Performance Characteristics

The histogram-based approach achieved high precision, meaning it produced relatively few false positives. When the model detected an anomaly, it was typically associated with a clear change in the latency distribution, either visible in the OWD measurements or reflected in structural changes of the histogram.

However, the sensitivity of the model was more moderate. Some smaller or less distinct anomalies were not always detected, particularly when changes in the distribution were subtle or did not significantly affect its overall structure.

This behaviour is further supported by the synthetic validation results (Section 5.4), where all normal samples were correctly classified, while a portion of anomalous samples was misclassified as normal. This confirms that the model is very precise, but may miss some anomalies, especially when they are subtle.

5.6 Comparison with Classical Statistical Thresholds

Classical statistical methods are widely used in network monitoring because they are simple, fast, and easy to explain. Fixed thresholds or rules based on summary statistics can detect clear

latency spikes with very low computational cost, which makes them useful for real-time monitoring and baseline alerting.

However, these methods usually look at one metric at a time and may miss more subtle changes in network behaviour. Gradual shifts in baseline latency, increased variability, or changes in histogram shape may not be detected if the median or average remains relatively close to normal values. In the examples presented in this study, the autoencoder was more sensitive to such changes because it learned the overall distribution of normal behaviour rather than relying on a single fixed rule.

In addition, the autoencoder-based approach has its own drawbacks: its outputs are less intuitive than a simple threshold, and it requires model training, threshold selection, and maintenance. For this reason, the two approaches should be seen as complementary rather than mutually exclusive. Classical thresholds can provide level-one alerting, while the autoencoder can help detect more complex and subtle anomalies.

While reconstruction error plots may not be intended as a primary visualisation for day-to-day NOC monitoring, they provide a useful signal for automated anomaly detection and for identifying subtle performance shifts that may not be evident in standard perfSONAR latency metrics. Therefore, reconstruction error should not be viewed as purely a theoretical artefact, but rather as a practical feature for alerting, anomaly scoring, and deeper network performance analytics.

6 Conclusions

This study has shown that histogram-based autoencoder models can be applied to perfSONAR latency measurements for anomaly detection. The main contribution of the outlined approach is its ability to capture not only obvious latency disturbances but also more subtle changes in the structure of latency distribution. These results suggest that the method can provide additional analytical value in network monitoring environments where standard metrics alone may not fully reveal anomalous behaviour.

The main takeaway from the approach outlined here is not that the proposed model can detect obvious latency spikes, since such events are typically already visible in standard perfSONAR metrics and dashboards. The added value it provides is rather in detecting changes in network behaviour that are much less apparent in conventional monitoring views. In several real-world examples, the reconstruction error increased even when commonly used latency indicators, such as median and percentile values, remained relatively stable.

This suggests that the model is capturing deviations that would not be easily identified through direct inspection of standard latency plots or through simple threshold-based monitoring. More specifically, the results indicate that the model is sensitive to changes in the overall structure of the latency distribution, including increased variability, irregular behaviour, or the emergence of multiple modes, rather than only to large shifts in average latency. The synthetic experiments

support this interpretation by showing that the reconstruction error responds to such distribution-level changes even when aggregate latency statistics do not change substantially.

Therefore, in practice the proposed method provides an additional layer of analysis that can reveal anomalies that are not clearly visible in standard perfSONAR summaries and would be more likely to remain unnoticed in routine monitoring. There is still room for further improvement, especially in the integration of the proposed method into real-time monitoring systems, but it can be concluded that the current implementation achieved its intended goals and provided a solid foundation for eventual future development in this area.

The given examples show that anomaly behaviour can differ between measurement directions. Because of this, each direction should be analysed separately, even when both directions are part of the same link. At the same time, similar patterns in both directions provide additional confidence that the detected event reflects a real network condition rather than random measurement noise.

The method depends on choosing a suitable threshold value, and different threshold values can change the outcome between false alarms and missed anomalies. The behaviour observed in real data is also supported by results on artificially generated synthetic datasets, where the model showed high precision and limited sensitivity. In a controlled environment, the model consistently identified clear structural changes in latency distribution, while some anomalies were not detected.

Overall, the results indicate that autoencoders can provide useful additional insight for the purposes of network monitoring, especially in cases where anomalies develop gradually or appear as changes in distribution rather than as simple threshold violations. For this reason, the proposed method is best used as a complement to classical monitoring techniques, adding another layer of analysis, rather than as a standalone approach.

Acronyms

OWD	One-way delay
NOC	Network operations centre
PMP	Performance measurement platform
LTSM	Long short-term memory
nmaas	Network Monitoring as a Service