

Anomaly detection in oil well production time-series data

I. Ishtiaq* and M. Farmanbar

Department of Electrical Engineering and Computer Science, University of Stavanger, Stavanger, Norway

E-mail: iqraishtiaq4016@gmail.com

Abstract. Anomaly detection in industrial time-series data is challenged by the scarcity of labeled fault data, limiting the applicability of supervised machine learning methods. This study compares three fundamentally different approaches for anomaly detection on the Petrobras 3W oil production dataset: a semi-supervised Temporal Convolutional Network (TCN) autoencoder trained only on normal data, a supervised XGBoost classifier trained on labeled anomalies, and a zero-shot GPT-4o vision-language model (VLM) that detects anomalies from visualized sensor data without task-specific training. The models are evaluated on binary detection of three production anomaly types using precision, recall, and F1-score. The supervised XGBoost model provided the strongest overall detection performance when labeled anomaly data were available. **The TCN autoencoder demonstrated that reliable anomaly detection can be achieved using only normal operating data, while the zero-shot VLM successfully identified anomalies without requiring model training. Labeled samples from the training partition were used during prompt and visualization development.** Prompt engineering and visualization design were found to substantially influence VLM performance. The results suggest that while the zero-shot VLM cannot yet match conventional methods in absolute performance, it represents a viable option for data-sparse deployments and a promising direction for future research.

Keywords: Time-series anomaly detection, Petrobras 3W dataset, Deep learning, Zero-shot learning, Multimodal artificial intelligence

1 Introduction

An unwanted event in an oil and gas production well can have significant financial consequences. Such events may arise from equipment malfunctions, pipeline leaks, or structural failures, often leading to costly unplanned downtime. Offshore operators, for example, experience annual losses of approximately 38 million USD due to production interruptions [1]. As a result, automated anomaly detection on real-time sensor data has become an important tool for reducing downtime and improving operational efficiency. Despite this need, a fundamental challenge remains: the scarcity of labeled production data. Anomalies are inherently rare, and collecting sufficient labeled examples in real operational environments is often impractical. In addition, expert annotation is expensive and time-consuming, which limits the feasibility of fully supervised learning approaches. Consequently, many industrial anomaly detection systems rely on semi-supervised or unsupervised methods that model normal operating behavior and detect deviations.

Recent advances in machine learning have introduced several approaches to address this limitation. Semi-supervised methods, such as Temporal Convolutional Network (TCN) autoencoders, learn representations of normal system dynamics and identify anomalies as deviations

from learned patterns. In contrast, supervised models such as gradient-boosted trees (e.g., XGBoost) can achieve strong performance when labeled data is available, but their effectiveness is often limited by class imbalance and incomplete fault coverage. More recently, zero-shot vision-language models (VLMs) have emerged as a novel alternative. By converting time-series data into visual representations, these models enable inference without task-specific **parameter updates**, using pretrained vision-language alignment to identify abnormal patterns. **In this study, "zero-shot" refers specifically to inference, meaning that no GPT-4o weights are updated and no labeled examples are provided in the inference context. However, the prompt and visualizations were developed using the labeled samples from the training data.** This reframes anomaly detection as an image understanding problem and raises the question of whether such models can compete with conventional numerical approaches in industrial settings.

Despite these developments, there is still limited understanding of how these fundamentally different paradigms compare under realistic industrial constraints, particularly in terms of detection performance, data efficiency, computational cost, and economic feasibility. Therefore, this study investigates the following research questions:

- RQ1: How does a semi-supervised TCN autoencoder perform on industrial time-series anomaly detection when trained only on normal operating data?
- RQ2: What performance is achieved by a supervised XGBoost model when labeled anomaly data is available, and how does class imbalance affect its effectiveness?
- RQ3: Can a zero-shot vision-language model operating on visualized time-series data provide competitive anomaly detection performance without task-specific parameter training at inference time?

Through these questions, this study provides a systematic comparison of semi-supervised, supervised, and zero-shot approaches for industrial anomaly detection under realistic operational constraints.

2 Related work

Anomaly detection in industrial time-series data has been widely studied using supervised, semi-supervised, and more recently, vision-language approaches. On the Petrobras 3W dataset, supervised methods such as Random Forest, Decision Trees, and XGBoost have consistently achieved strong performance by combining statistical feature extraction with conventional classifiers [2, 3]. Deep learning approaches have primarily focused on recurrent and convolutional autoencoders, where learned latent representations are either used directly for anomaly detection or combined with classical classifiers [4, 3]. More recently, stacked GRU and LSTM architectures have demonstrated strong performance on multiclass fault classification [5]. However, these approaches rely on supervised training or historical fault data. Autoencoders are among the most widely used methods for semi-supervised anomaly detection, as they learn to reconstruct normal operating conditions and identify anomalies through elevated reconstruction errors [6]. Although effective in many industrial applications, reconstruction error is not a theoretically reliable measure of anomalousness, as autoencoders may accurately reconstruct out-of-distribution samples under certain conditions [7]. Temporal Convolutional Networks (TCNs) have emerged as an attractive alternative to recurrent architectures for sequence modeling, offering competitive accuracy together with stable and efficient training [8]. For supervised tabular learning, gradient-boosted decision trees, particularly XGBoost [9], remain among the strongest-performing methods on

medium-sized structured datasets and continue to outperform many deep learning approaches in industrial classification tasks [10, 11].

Recent advances in vision-language models (VLMs) have introduced a new paradigm for anomaly detection through zero-shot reasoning. Early work adapted CLIP for industrial anomaly detection using text prompts describing normal and defective states [12], while later methods such as AnomalyCLIP improved generalization through learnable prompts [13, 14]. Extending these ideas to time-series data, He et al. [15] proposed a two-stage framework that converts time series into line plots, uses a lightweight vision model to identify anomaly candidates, and employs GPT-4o for zero-shot verification. This work is closely related to the approach presented in this study, which likewise transforms industrial time-series data into visual representations for GPT-4o-based anomaly detection. To the best of our knowledge, no previous work has applied a zero-shot VLM to anomaly detection on the Petrobras 3W dataset.

3 Methodology

This section describes the full experimental pipeline applied consistently across the anomaly detection approaches employed.

3.1 Dataset and preprocessing

3.1.1 Dataset: This study uses the Petrobras 3W dataset [16, 17], a benchmark dataset for anomaly detection in offshore oil and gas wells. It contains real data collected from actual oil wells during operation, simulated data, and synthetic recordings data that are artificial examples; however, only real operating data is used in this work because it better represents industrial conditions. Each sample corresponds to a time-series segment collected from an individual well and stored in Parquet format. Event type 0 represents normal operation, while event types 1–9 correspond to different undesirable operational conditions. The dataset is highly imbalanced, with normal operation (type 0) dominating and event frequencies varying significantly across anomaly classes. In addition, recordings originate from different wells and time periods and cannot be concatenated into continuous streams. Therefore, each file is treated as an independent time-series segment. At the timestep level, each record includes a label indicating normal operation (0), anomaly (1), or transient states (10x). All transient and anomalous states are merged into a single binary anomaly class, as the objective is to detect abnormal behavior rather than classify specific fault types. Files with invalid sensor behavior (fewer than two unique values per channel) were removed.

Following this filtering, event types with insufficient samples for robust train–validation–test splitting were excluded. The final dataset includes four classes: normal operation (type 0), spurious DHSV closure (type 2), severe slugging (type 3), and flow instability (type 4). After filtering, the dataset consists of 523 normal operation files, 21 DHSV closure files, 32 severe slugging files, and 252 flow instability files, giving a total of 828 files.

The 3W dataset includes recordings from 21 wells collected between 2012 and 2018. Since the same wells appear across multiple event types and time periods, train–test splits may include temporally separated samples from the same physical system. This introduces potential temporal leakage, as models may indirectly observe well-specific characteristics across splits. While unavoidable due to dataset structure, this limitation is acknowledged. Data are split at the file level into training (60%), validation (20%), and test (20%) sets, stratified by event type. All features were normalized using z-score normalization, **based solely on the train set**, to ensure that variables with different scales contribute equally

3.1.2 Feature selection: Features were selected manually on the basis of domain knowledge and visual inspection of representative samples in collaboration with a production engineer. The analysis identified two signals from the Christmas tree (wellhead): the tubing pressure sensor P-TPT and the tubing temperature sensor T-TPT, which capture the main physical signatures of the studied anomaly types.

3.1.3 Windowing: Time series are segmented into fixed-length, non-overlapping windows. A baseline window size of 2048 timesteps (stride 2048) is used. This choice is motivated by event structure: event type 2 contains mixed normal-to-anomalous transitions requiring sufficient temporal coverage, while event type 3 exhibits periodic slugging behavior. To preserve full cycle structure, an additional configuration of 4096 timesteps is also evaluated. **At a sampling frequency of 1 Hz, a window of 2048 timesteps represents approximately 34.1 minutes of operation.** A window is assigned a label of 1 (anomalous) if at least 10% of its constituent timesteps carry an anomaly or transient label, and 0 (normal) otherwise. This threshold avoids mislabeling windows that contain only a brief transient at their boundary while still capturing windows that are meaningfully anomalous. **The choice of a 10% threshold did not affect the test labels within the examined range of 1% to 25%, indicating that the reported results are robust to this choice.**

3.2 Model development

Three fundamentally different anomaly detection approaches are evaluated: a semi-supervised Temporal Convolutional Network (TCN) autoencoder trained only on normal data, a supervised XGBoost classifier trained on labeled anomalies, and a zero-shot GPT-4o vision-language model (VLM) that detects anomalies from visualized sensor data without task-specific training.

3.2.1 Semi-supervised TCN autoencoder: The TCN autoencoder is trained in a one-class setting using only normal operating data. It learns to reconstruct normal patterns, and anomalies are identified through increased reconstruction error. The validation and test sets contain both normal and anomalous windows, ensuring consistent evaluation. The input consists of four features: raw signals T-TPT and P-TPT, and their first-order differences, which capture both signal values and temporal changes.

Adding the difference features improved validation ROC-AUC from 0.75 to 0.87. Therefore, the four-channel representation was used for the final model. The architecture and training configuration used can be seen in Table 1. The hyperparameter search space can be seen in Table 2

3.2.2 Supervised XGBoost model: XGBoost is used as a fully supervised baseline trained on the complete labeled training set, including both normal and anomalous windows, with identical validation and test splits as the other models. Each time-series window is converted into a 21-dimensional feature vector consisting of statistical descriptors (mean, standard deviation, min, max, median, skewness, kurtosis) for T-TPT and P-TPT, their first-order differences, and a Pearson correlation between the two signals. Class imbalance is handled using `scale_pos_weight` to emphasize rare anomalies. Hyperparameters are optimized using cross-validation with ROC-AUC, while early stopping determines the optimal number of boosting rounds. The hyperparameter search space can be seen in Table 3

3.2.3 Zero-shot vision-language model (VLM): The VLM approach reformulates anomaly detection as image classification by converting each time-series window into a plot and querying

Table 1. TCN autoencoder architecture and training configuration.

Component	Configuration
Input features	T-TPT, P-TPT, Δ T-TPT, Δ P-TPT
Input shape	(2048, 4)
Residual TCN blocks	9 for 2048; 10 for 4096
Convolutions per block	2 causal 1D convolutions
Filters	16
Kernel size	3
Dilation	Exponentially increasing
Activation	ReLU
Layer normalization	Enabled
Residual connections	Enabled
Dropout	0.0
Output layer	1×1 convolution to 4 channels
Optimizer	Adam
Learning rate	10^{-3}
Batch size	64
Loss function	Mean squared reconstruction error
Maximum epochs	100
Early stopping	Patience = 5
Minimum improvement	10^{-4}
Best weights	Restored after early stopping
Training data	Normal training windows only
Early-stopping data	Normal validation windows only
Decision threshold	Youden’s J on labeled validation set

Table 2. TCN hyperparameter search space.

Parameter	Values evaluated	Selected
Number of blocks	9, 10	9
Number of filters	8, 16, 32	16
Activation	ReLU, GELU, LeakyReLU	ReLU
Kernel size	Fixed	3
Dropout	Fixed	0.0
Layer normalization	Fixed	Enabled
Residual connections	Fixed	Enabled

GPT-4o with a prompt to output a binary label, similar to how a human engineer would inspect production charts. No model training or threshold calibration is performed; instead, a subset of the training data is used for prompt and visualization design.

Each window ($W_{size}, 4$) is visualized as a multi-panel plot of T-TPT, P-TPT, and their first-order differences. The VLM operates on these raw visual representations to infer anomalies. The final visualization includes aligned subplots, preserved physical scales, and an optional alert highlighting large deviations in T-TPT, which was found useful during development. Since

Table 3. XGBoost hyperparameter search space.

Parameter	Values evaluated
Learning rate	0.05, 0.07, 0.10
Maximum depth	4, 5, 6
Minimum child weight	1, 2
Gamma	0, 0.1, 0.2
Subsample	0.7, 0.8
Column subsampling	0.7, 0.8
L_1 regularization	0.1, 0.3, 0.5
L_2 regularization	0.8, 1.0, 1.2
Cross-validation	3-fold, training partition only
Optimization metric	ROC-AUC
Maximum boosting rounds	2000
Early stopping	20 rounds on validation ROC-AUC

the VLM is not trained, all task-specific knowledge is encoded in the prompt. Prompt design is therefore treated as an iterative optimization process, using 25% of the training data for development.

The initial prompt described the physical system and anomaly types in domain terms, and was progressively refined by adding visual cues linked to patterns observed in the plots. Errors were analyzed by inspecting model reasoning, and the prompt was adjusted to resolve recurring misclassifications. The final prompt is mainly descriptive but includes explicit rules for key cases (e.g., temperature-based alert conditions for DHSV closure) and guidance for ambiguous windows. The model is instructed to output brief reasoning before the final prediction and return the label in a fixed format (`Label: 0/1`) for easy parsing. At inference time, each window is converted to an image, base64-encoded, and sent to GPT-4o via the OpenAI API together with the prompt. A system message defines the task as binary anomaly detection, followed by the user prompt and image input. The final label is extracted from the response using a structured pattern (`Label: 0/1`). Since the model outputs discrete labels rather than continuous scores, ROC-AUC cannot be computed, and evaluation is performed directly using precision, recall, and F1-score.

4 Results and discussion

All primary results are reported for the 2048-timestep configuration, which was used for model development. Performance is reported as the average precision, recall, and F1-score over 15 independent runs. The file-level partition changes between runs, however within a given run, all three models use the same held out test files and corresponding windows. The reported results should be interpreted as file-level generalization rather than independent-well generalization. Future work should evaluate the models using well-grouped and chronological data partitions to assess their ability to generalize to unseen wells and future operating conditions. For each metric the 95% confidence interval was calculated from the 15 run-level results as $\bar{x} \pm 1.96 \frac{s}{\sqrt{n}}$. Event types 3 and 4 contain only anomalous windows, making per-event precision always 1.0 and therefore uninformative. Conversely, the normal class contains no anomalies, resulting in zero anomaly-class metrics and serving only to verify that normal operation is not systematically misclassified.type is included for completeness and to verify that no model systematically

misclassifies normal operation.

4.1 TCN autoencoder

The TCN anomaly score is based on the reconstruction error of each window. Figure 1 shows the distribution of these scores on the test set, broken down by event type, plotted on a logarithmic scale to make both the threshold and the normal cluster visible. Normal operation

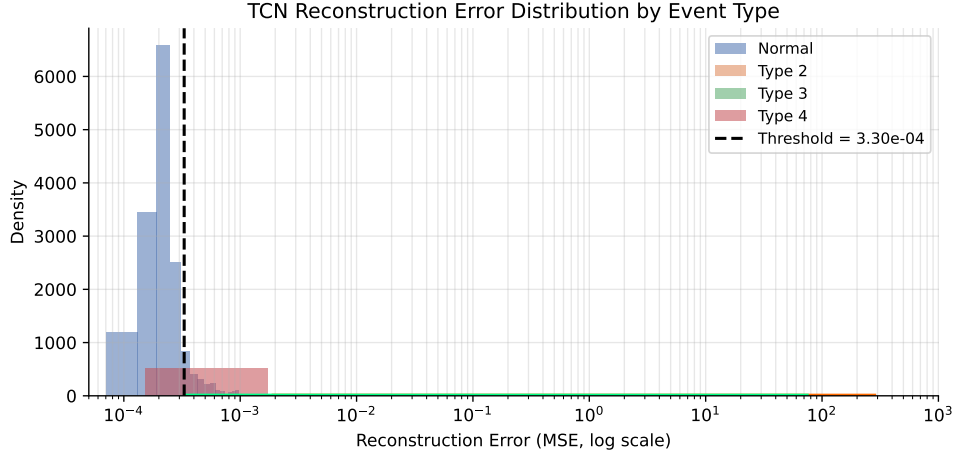


Figure 1. Reconstruction error distribution. Window size = 2048.

produced consistently low errors, indicating that the model learned a compact representation of normal behavior. Event types 2 and 3 showed clearly higher reconstruction errors and were well separated from normal operation. However, type 4 exhibited partial overlap with the normal error distribution, explaining its lower detection performance. This suggests that some flow instability patterns are sufficiently similar to normal operation and can be reconstructed by the autoencoder despite being anomalous. The overall test set metrics, averaged across 15 independent runs, are summarized in Table 4a.

The high recall indicates that the model successfully detected the majority of anomalous events, which is particularly important in oil and gas applications where missed failures can lead to significant economic losses and safety risks. However, the lower precision shows that a considerable portion of generated alarms, meaning approximately 37%, were false positives. These false alarms are likely caused by normal operating variations that are not fully represented in the normal-only training data. The observed variability across runs further suggests that one-class learning is sensitive to model initialization and to how well the training data captures the full distribution of normal operating conditions.

Overall, the TCN autoencoder demonstrates that training solely on normal operation can provide effective anomaly detection under limited label availability. Its main challenge is distinguishing difficult anomaly types, such as flow instability, from natural variations in normal production behavior.

4.2 XGBoost

Feature importance analysis Figure 2 shows the top features ranked by mean gain, and Figure 3a shows the corresponding SHAP plot which provides a richer picture of how individual features influence predictions.

Table 4. Test set performance of TCN autoencoder (a) and XGBoost (b).

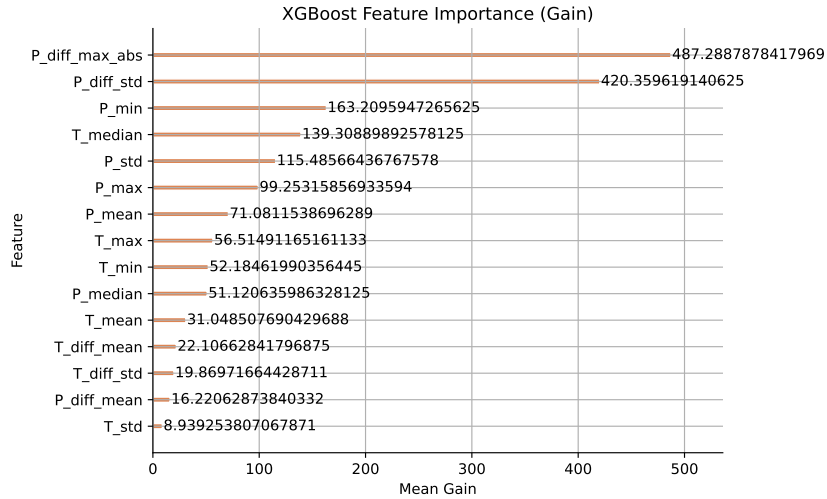
	Precision	Recall	F1
Mean	0.63	0.89	0.73
Std	0.09	0.11	0.09
95% CI	± 0.04	± 0.06	± 0.04

(a) TCN Autoencoder

	Precision	Recall	F1
Mean	0.95	0.97	0.96
Std	0.04	0.02	0.03
95% CI	± 0.02	± 0.01	± 0.01

(b) XGBoost

The two most important features by gain are **P_diff_std**: how much the pressure changes over time (pressure variability) and **P_diff_max_abs**: the largest sudden pressure change in the window, which are both derived from the first-order differences of the pressure signal. This means the model is very sensitive to rapid or unstable pressure changes, which are common during abnormal well conditions. The **P_min** and **P_std** features are also highly ranked, reflecting that anomalies tend to push the pressure signal outside its normal operating range. The temperature related features are lower on the list, indicating that pressure is the most telling signal than temperature patterns when it comes to anomalies.

**Figure 2.** Feature importance (mean gain)

The SHAP plot shows how each feature influences predictions. Positive SHAP values increase the likelihood of an anomaly, while negative values indicate normal operation. High pressure variability **P_diff_std** is the strongest anomaly indicator, whereas pressure magnitude features and temperature features have a smaller impact. XGBoost achieved 0.95 precision, 0.97 recall, and 0.961 F1-score, outperforming the TCN autoencoder as shown in Table 4b. The low standard deviations across runs indicate stable performance. Probability distributions shows how well the model separates normal and anomalous windows based on its predicted anomaly probability. The XGBoost model assigns each window an anomaly probability score between 0 and 1, where values close to 0 indicate normal operation and values close to 1 indicate anomalous behavior. Normal operation (type 0) is expected to cluster near 0, indicating correct identification of normal behavior, while anomalous events (types 2–4) should cluster near 1, indicating successful anomaly detection. Any overlap between the two distributions represents uncertainty and may

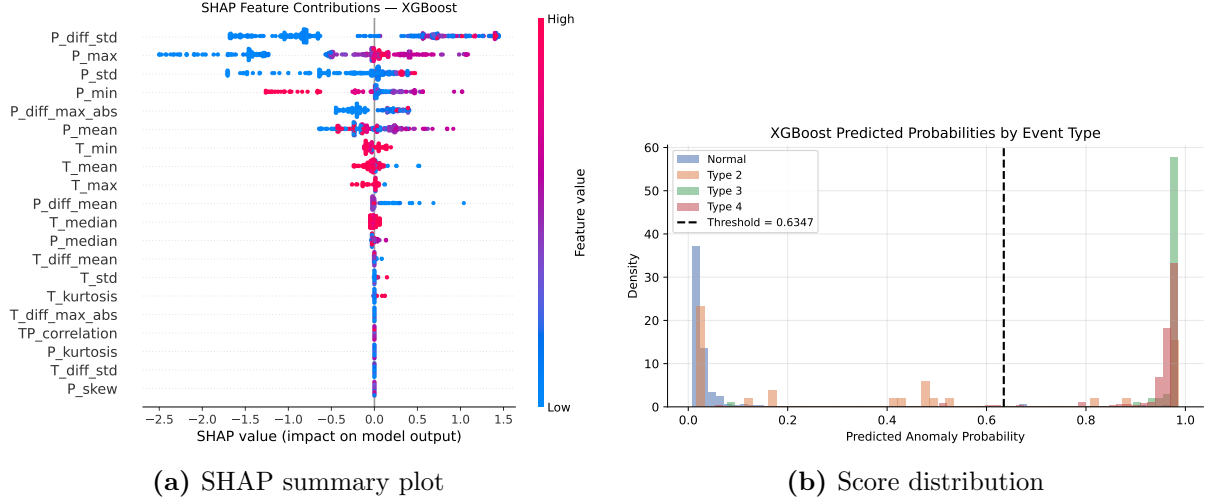


Figure 3. (a) SHAP summary showing feature contributions and (b) score distribution of XGBoost predictions.

lead to false alarms or missed anomalies. Figure 3b reports clear separation between normal windows and event types 3 and 4, while most remaining errors were associated with event type 2, where windows often contain both normal and anomalous segments. Although XGBoost delivered the strongest overall results, it relies on labeled anomaly data during training, which may be difficult to obtain in practical industrial settings.

4.3 Vision-Language model

The VLM approach required an iterative development process covering both the visualization design and the prompt. The model’s reasoning on both correct and incorrect predictions is inspected to identify failure modes and guide prompt adjustments.

4.3.1 Development progression: All development was conducted on a held-out subset of the training split; the validation and test sets were not used.

Stage 1: Two-Signal visualization: The first attempt was to show the VLM only the two original sensor signals: T-TPT and P-TPT as line graphs and ask it to detect anomalies. Four prompt iterations were evaluated on this configuration. The first prompt achieved high recall (0.91) but low precision (0.26), resulting in many false alarms. Adding more detailed instructions in the prompt describing anomaly shapes and temperature variation thresholds did not resolve this issue. The conclusion was that raw sensor plots alone were insufficient motivating the inclusion of first-order difference signals in the next stage. Results are summarized in Table 5.

Stage 2: Four-Signal visualization: In the second stage, the first-order difference signals (T_diff and P_diff) were added to represent signal dynamics. The idea was that sudden changes in these signals could help the VLM distinguish real faults from normal variations. Four new prompts were evaluated, requiring the model to explicitly calculate the temperature change before predicting DHSV closure. This increased precision from 0.25 to 0.39 and F1-score from 0.39 to 0.45 (Table 5). However, although false positives decreased, the model often rejected DHSV closures without considering other anomaly types. This showed that prompt engineering

Table 5. VLM development across prompts and visualizations.

Visualization	Prompt	Precision	Recall	F1
Stage 1: Raw signals	1	0.26	0.91	0.40
	2	0.26	0.81	0.39
	3	0.26	0.85	0.39
	4	0.27	0.80	0.40
Stage 2: Raw + difference signals	5	0.25	0.88	0.39
	6	0.32	0.53	0.40
	7	0.35	0.59	0.44
	8	0.39	0.54	0.45
Stage 3: Alert Coloring System	9	0.28	0.78	0.41
	10	0.27	0.83	0.41
	11	0.29	0.81	0.43
	12	0.31	0.76	0.45
	13	0.40	0.66	0.50
	14	0.41	0.66	0.51
	15	0.43	0.67	0.52
	16	0.45	0.69	0.54

alone was insufficient, motivating the introduction of an alert-based visualization in the next stage.

Stage 3: Alert coloring system: In the third stage, a visual alert was added to simplify DHSV detection. Temperature plots were colored crimson when the temperature range exceeded 4.0°C, indicating a possible DHSV closure, and blue otherwise. **This embeds a handcrafted DHSV cue in the visualization based on domain-knowledge and removed the need for the VLM to perform arithmetic or infer DHSV events from curve shapes alone.** The initial prompt performed poorly due to occasional incorrect interpretation of plot colors. After adding explicit color verification rules, eight additional prompts were tested. The best prompt (Prompt 16) achieved an F1-score of 0.54 (precision = 0.45, recall = 0.69), mainly by reducing false positives. Performance improved when prompts focused on visual patterns rather than physical explanations. However, longer prompts caused instruction conflicts, where the model followed some rules but ignored others. Adding a concise summary of key rules helped mitigate this issue. The final prompt is provided in the Appendix.

Stage 4: Visual design optimization: After finding the best prompt, the next step was to check whether small changes in how the plots look could improve the VLM performance. Different visualization settings were tested, including colors, grid style, background, line style, and figure format.

The results were very similar across all designs, with F1-scores between 0.55 and 0.58, showing that the prompt had a much larger impact than the visual style. The best configuration was the Matplotlib classic style (v12), which achieved the highest F1-score (0.58) and was selected as the final visualization. The final design used clear line plots, frequent y-axis ticks, no grid, simple default colors. When the final configuration was tested on the full validation set, the

Table 6. VLM visualization configurations using the final prompt.

Config	Key properties	Precision	Recall	F1
v4	Steelblue/orange, no grid	0.49	0.69	0.57
v5	Steelblue/orange, subtle grid	0.49	0.65	0.56
v6	Steelblue/orange, strong grid	0.48	0.71	0.57
v7	Blue/magenta, strong grid	0.48	0.70	0.57
v8	Dark background, auto grid	0.50	0.63	0.56
v9	Dark background, neon colors	0.49	0.63	0.55
v10	FiveThirtyEight style	0.43	0.75	0.55
v11	Solarize Light2 style	0.46	0.68	0.55
v12	Classic style	0.48	0.72	0.58
v13	Seaborn style	0.47	0.68	0.55

model achieved precision = 0.41, recall = 0.64, and F1 = 0.50.

Analysis of incorrect predictions showed an important limitation. The VLM often missed severe slugging events because the 2048-timestep windows sometimes captured only part of the repeating wave pattern. Since severe slugging is identified mainly by periodic behavior, seeing only a partial cycle made it difficult for the model to recognize the anomaly. To address this problem, a larger window size (4096 timesteps) was later tested. The larger window provides more temporal context and increases the chance of capturing complete slugging cycles.

4.3.2 Test-set performance: The overall test set metrics for the VLM are summarized in Table 7. The VLM achieves an averaged precision of 0.45, recall of 0.61, and F1-score of 0.52. Compared with TCN and XGBoost, the VLM produced more incorrect predictions because it relied on visual reasoning rather than learning directly from numerical data. The main weakness was false alarms on normal operation. The model learned useful visual patterns for anomalies but was sometimes too sensitive and interpreted normal pressure or temperature variations as faults. Unlike the other models, the VLM itself was deterministic (temperature = 0), so performance variation came from differences in the test samples rather than randomness in the model.

Table 7. GPT-4o (VLM) test results

	Precision	Recall	F1
Mean	0.45	0.61	0.52
Std	0.04	0.05	0.04
95% CI	± 0.02	± 0.02	± 0.02

Window Size Effect: Increasing the window size to 4096 timesteps was evaluated to assess the impact of additional temporal context, motivated by VLM errors on truncated severe slugging patterns. Table 8 summarizes the overall results across both window sizes for direct comparison. Larger windows slightly reduce XGBoost performance, while improving TCN recall from 0.89 to 0.93 and F1 from 0.73 to 0.74. The VLM shows the largest gain, with recall increasing from 0.61 to 0.81 and F1 from 0.52 to 0.60, mainly due to better recognition of periodic patterns.

Per-Event-Type Analysis: Figures 4a and 4b compare recall across anomaly types for window sizes of 2048 and 4096 timesteps. Precision is omitted for event types 3 and 4 since all samples are anomalous by definition. Results are based on representative seeded runs and are interpreted qualitatively. For the 2048-timestep setting, event type 3 (severe slugging) is well detected by all models due to its distinctive periodic behavior, although the VLM is affected when cycles are truncated at window boundaries. Event type 4 (flow instability) is better captured by feature-based models, while event type 2 (DHSV closure) shows the largest gap, with the TCN outperforming XGBoost by preserving temporal patterns.

Increasing the window size to 4096 timesteps mainly improves the VLM, particularly for type 3 anomalies, as additional context helps capture periodic structures. XGBoost remains relatively stable, while variations in TCN performance across event types are likely influenced by run-to-run initialization effects. Overall, the results demonstrate complementary model behaviors: XGBoost benefits from statistical patterns, the VLM from richer visual context, and the TCN from full temporal dynamics.

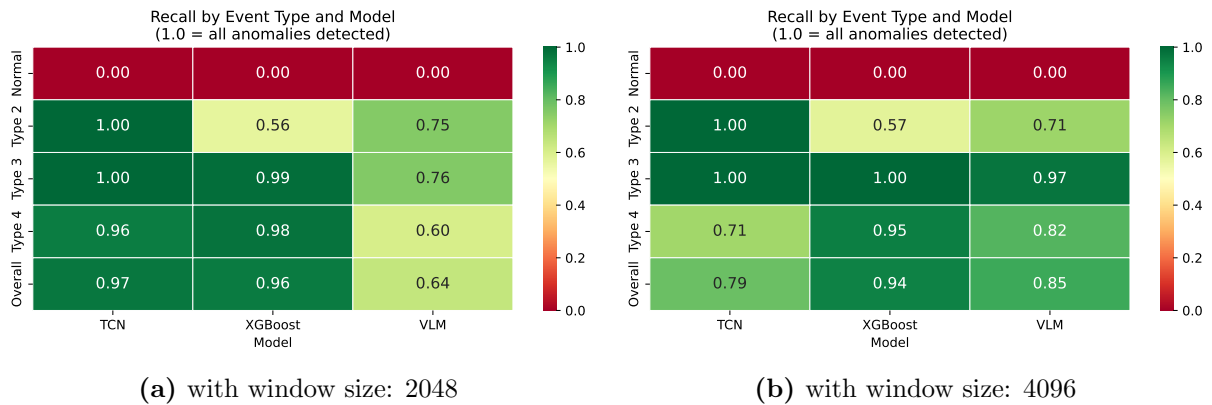


Figure 4. Recall analysis across anomaly event types for a representative seeded run.

Table 8. Averaged test set results for all three models across both window sizes.

Model	Window	Precision	Recall	F1
TCN	2048	0.63	0.89	0.73
	4096	0.62	0.93	0.74
XGBoost	2048	0.95	0.97	0.96
	4096	0.94	0.97	0.95
GPT-4o	2048	0.45	0.61	0.52
	4096	0.48	0.81	0.60

Overall, larger windows improve models that rely on temporal structure, while providing limited benefit for feature-based approaches. The results show that optimal window size depends on the characteristics of each model.

5 Conclusion

This study investigated three fundamentally different approaches for anomaly detection in industrial time-series data: a supervised machine learning model, a semi-supervised deep learning

model, and a zero-shot vision-language model. The results highlight a fundamental trade-off between detection performance, data requirements, and deployment flexibility. Supervised approaches provide strong detection capability when reliable labeled anomaly data is available; however, collecting and maintaining such datasets is costly and often impractical in industrial environments. Semi-supervised approaches provide a more practical alternative by learning normal operational behavior without requiring extensive fault histories, making them suitable for scenarios where normal data is abundant but labeled anomalies are limited.

The zero-shot VLM approach introduces a different paradigm by enabling anomaly analysis **without model training or historical fault examples at inference**. Although it does not currently replace conventional models in terms of detection performance, its ability to incorporate domain knowledge through prompts, adapt to new conditions without retraining, and provide interpretable explanations makes it valuable in data-scarce environments, such as new installations or poorly documented systems. However, due to its current limitations in alarm precision, it should be considered an assisting component rather than a standalone monitoring solution. The findings suggest that these approaches should be viewed as complementary rather than competing solutions. **The three approaches have different strengths and limitations. Although VLMs offer flexibility in analysing time-series plots, their adaptability and interpretability require further evaluation. The current study does not establish their suitability for real-time industrial monitoring. Future work should evaluate generalization to unseen wells, inference time, computational cost, and the effects of data normalization and visualization choices.** A practical industrial anomaly detection framework could combine lightweight numerical models for continuous monitoring with VLM-based reasoning for ambiguous cases requiring contextual interpretation. Such a hybrid approach could provide a balance between accuracy, scalability, interpretability, and adaptability.

Overall, this work demonstrates the potential of integrating conventional machine learning methods with emerging vision-language models for industrial anomaly detection. While further improvements are required before VLMs can support fully autonomous and safety-critical monitoring, their flexibility and ability to operate under limited data availability make them a promising direction for future research.

References

- [1] MaxGrip 2026 Cost of unplanned downtime: The impact on businesses citing Kimberlite offshore oil and gas downtime research URL <https://www.maxgrip.com/resource/article-the-cost-of-unplanned-downtime/>
- [2] Marins M A, et al. 2021 Fault detection and classification in oil wells and production/service lines using random forest. *J. Pet. Sci. Eng.*, **197**:107879
- [3] Gatta F, et al. 2024 Predictive maintenance for offshore oil wells by means of deep learning features extraction. *Expert Syst.*, :e13128
- [4] Machado A P F, et al. 2022 Improving performance of one-class classifiers applied to anomaly detection in oil wells. *J. Pet. Sci. Eng.*, **218**:110983
- [5] Bayazitova G, et al. 2024 Oil and gas flow anomaly detection on offshore naturally flowing wells using deep neural networks. *Geoenergy Sci. Eng.*, **242**:213240
- [6] Chalapathy R and Chawla S 2019 Deep learning for anomaly detection: A survey., (*Preprint arXiv:1901.03407*)
- [7] Bouman R and Heskes T 2025 Autoencoders for anomaly detection are unreliable. *arXiv preprint arXiv:2501.13864*,
- [8] Bai S, et al. 2018 An empirical evaluation of generic convolutional and recurrent networks for sequence modeling., (*Preprint arXiv:1803.01271*)
- [9] Chen T and Guestrin C 2016 XGBoost: A scalable tree boosting system *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (ACM) pp 785–794
- [10] Grinsztajn L, Oyallon E and Varoquaux G 2022 Why tree-based models still outperform deep learning on tabular data.,
- [11] Wang T, et al. 2023 Machine learning (ML) based predictive maintenance policy for bridges crossing waterways. *Transp. Res. Procedia*, **72**:1037–1044
- [12] Jeong J, et al. 2023 WinCLIP: Zero-/few-shot anomaly classification and segmentation *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* pp 19606–19616
- [13] Zhou Q, et al. 2024 AnomalyCLIP: Object-agnostic prompt learning for zero-shot anomaly detection *The Twelfth International Conference on Learning Representations (ICLR)*
- [14] Kakda M, et al. 2026 Analyzing VLM-based approaches for anomaly classification and segmentation. *arXiv preprint arXiv:2601.13440*,
- [15] He Z, Alnegheimish S and Reimherr M 2025 Harnessing vision-language models for time series anomaly detection. *arXiv preprint arXiv:2506.06836*,
- [16] Vargas R E V, et al. 2019 3W dataset 2.0.0: A realistic and public dataset with rare undesirable real events in oil wells. *J. Pet. Sci. Eng.*, **181**:106223
- [17] Petrobras 2026 3w dataset URL <https://github.com/petrobras/3W>

A Appendix

Listing 1: Final VLM prompt (prompt 16).

```
You are analyzing sensor data from an oil and gas well. The four plots
show measurements recorded at the Christmas tree (wellhead):

- T-TPT: temperature (degrees C)
- T_diff: rate of change of temperature (degrees C per timestep)
- P-TPT: pressure (Pa)
- P_diff: rate of change of pressure (Pa per timestep)

Each plot shows 2048 consecutive timesteps. Note that each subplot has
an independent y-axis scale. The sensors are related - abnormal
events typically affect both signals, though not always equally.

Normal operation shows steady behavior with random noise. Normal noise
has no trend, no periodicity, and no systematic drift in either
sensor. The rate-of-change plots should hover near zero with small
random fluctuations during normal operation.

Before starting the analysis, check the color of the plot for
temperature. If this is in red, then label the window anomalous as a
DHSV closure. If this is in blue, analyze for either normal operation
, severe slugging or flow instability. The first thing you should
check is the pressure plot: is it shaped as part of a sinusoidal wave
? Does it have gradual increases and decreases (could change
direction 1 or 2 times, or more) throughout the window? It is not
always a full period, sometimes just a part of a wave. If this is the
case then this is likely severe slugging. If either the temperature
or pressure plots show repeated oscillations or wave-like features (
gradual changes that go upwards -> downwards -> upwards, i.e. a
sinusoidal pattern), then this is likely severe slugging. Another way
severe slugging can present itself is clearly repeated patterns. If
the plots show highly stable operation that is occasionally affected
by some disturbance for a shorter event (around 300-500 timesteps),
then this is likely flow instability. If the plots are lacking any of
these features and are stable with random fluctuations and/or noise,
then this is likely normal operation.

Anomalies fall into one of three categories:

1. DHSV closure: This will look like a slow steady gradual increase or
decrease in both temperature and pressure. Never classify this event
if the temperature plot is blue.

2. Severe slugging: The pressure plot is the PRIMARY indicator of severe
slugging. This will look like repeated patterns or wave-like
features. If a plot looks like a sine wave, identify as anomaly. One
window can be part of an entire period/wave or many smaller periods/
waves. Smaller waves could be noisy, so it has an overall sine wave
looking shape, but the general shape is cluttered with spikes. You
should not let the spikes deter you from classifying as an anomaly if
the overall shape is wave-like. Always check for an overall shape.
Lastly the event can also look like spike-like oscillations that are
```

not directly following a sinusoidal shape, but have consistent frequency. What differentiates spikes from normal noise is how long the spike "lasts" for. For this to be considered severe slugging, the spikes should be slower and last for around 300-500 timesteps. Look at how repeating the pattern is, and how long each "slug" lasts to determine if this is an anomaly. For it to be considered an anomaly there should be 6 or less peaks per 1000 timesteps (e.g., 6 peaks between 1000 and 2000). If there are more than 6 peaks per 1000 timesteps, then this is likely normal operation with noise. You should also make sure that the amount of peaks in the pressure is equal or more than the amount of peaks in the temperature. If there are more peaks in the temperature than the pressure, then this is likely normal operation with noise.

3. Flow instability: Often looks like steady operation that is clearly being affected by something before returning to a baseline. This anomaly lasts for longer than a singular spike. Make sure the amount of timesteps cover a spike that is slower in nature and lasts for around 300-500 timesteps, to be able to distinguish from random noise.

Examine all four plots carefully, paying attention to:

- Sudden clear regime shifts or step changes
- Periodic or rhythmic patterns
- Sustained drift or trend in either sensor
- Elevated or structured activity in the rate-of-change plots

Also pay attention to the amplitude of the changes using the y-axis values as a guide. Anomalies will typically have larger deviations that are more clearly visible on the axis and have a clear shape (e.g., step change, periodic pattern, or sustained drift).

Some tips for each event type:

- DHSV closure: If you suspect this anomaly type but the temperature plot is blue instead of red, classify as normal operation.
- Severe slugging: Try to find either sinusoidal patterns (part of a period, a full period, or many periods), or spike-like oscillations that last for a while (300-500 timesteps). Make sure you check the overall shape, because sometimes there is a clear wave-like pattern hidden behind spikes. For the spike-like oscillations, make sure there are 6 or less peaks per 1000 timesteps, and that the pressure has at least as many peaks as the temperature. Look for a clear and consistent period in the spikes (no amplitudes which last for longer than others). Wave-like features that repeat throughout the window are a strong indicator of severe slugging. If the pressure plot shows clear broad rolling oscillations or repeated wave-like humps - even if temperature appears stable or slowly drifting - classify as severe slugging.
- Flow instability: Look for a stable system that deviates at times, where the deviations last for a while and are slower (not one sudden spike, but one that goes across several timesteps) to be able to distinguish from random noise.

Provide a 2-3 sentence reasoning, then on the final line write ONLY:

Label: 0 (normal operation)
Label: 1 (anomaly)