Contents

[Case Study: Multi-Modal Anomaly Detection in Oil Rig Operations Using Time Series and Text Logs 3](#_Toc203732407)

[Objective: 3](#_Toc203732408)

[Step:1 Simulate Data 3](#_Toc203732409)

[**1. Mud Pump** 3](#_Toc203732410)

[**2. Centrifugal Pump** 3](#_Toc203732411)

[**3. Blowout Preventer (BOP)** 4](#_Toc203732412)

[**4. Shale Shaker** 4](#_Toc203732413)

[**5. Diesel Generator** 5](#_Toc203732414)

[Step:2 Preprocess Data 5](#_Toc203732415)

[**1.** **Timestamp Parsing & Feature Engineering** 5](#_Toc203732416)

[**2.** **Missing Value Handling** 5](#_Toc203732417)

[**3.** **Preprocessing Pipeline** 5](#_Toc203732418)

[**4.** **Batch Processing – All Equipment** 6](#_Toc203732419)

[Step:3 Anomaly Detection 6](#_Toc203732420)

[**1.** **Configuration & Setup** 6](#_Toc203732421)

[**2.** **Load Processed Sensor Files** 6](#_Toc203732422)

[**3.** **Isolation Forest (Tree-Based)** 6](#_Toc203732423)

[4. **One-Class SVM (Kernel-Based)** 6](#_Toc203732424)

[**5.** **LSTM Autoencoder (Deep Learning)** 6](#_Toc203732425)

[Step:4 Text embedding and correlation 7](#_Toc203732426)

[**1.** **Load Anomaly Outputs** 7](#_Toc203732427)

[**2.** **Parse Unstructured Operator Logs** 7](#_Toc203732428)

[**3.** **Generate Semantic Embeddings** 7](#_Toc203732429)

[**4.** **Anomaly–Log Matching (Semantic Correlation)** 7](#_Toc203732430)

[**5.** **Result Compilation** 7](#_Toc203732431)

[**6.** **Save Correlation Output** 7](#_Toc203732432)

[Step:5 Insight generation 8](#_Toc203732433)

[**1.** **Load API Key** 8](#_Toc203732434)

[**2.** **Prompt Construction Per Equipment** 8](#_Toc203732435)

[**3.** **Language Model Interaction** 8](#_Toc203732436)

[**4.** **Postprocessing and Storage** 8](#_Toc203732437)

[Step:6 Streamlit-Based Real-Time Monitoring Dashboard 8](#_Toc203732438)

[**1.** **App Setup & Styling** 8](#_Toc203732439)

[**2.** **File Mapping & Directory Initialization** 9](#_Toc203732440)

[**3.** **Sidebar Navigation** 9](#_Toc203732441)

[**4.** **Equipment Logs View** 9](#_Toc203732442)

[**5.** **Anomaly Detection View** 9](#_Toc203732443)

[**6.** **Correlation View** 9](#_Toc203732444)

[**7.** **Anomaly Summary View** 9](#_Toc203732445)

[Critical Analysis of the Approach 10](#_Toc203732446)

[**1.** **Potential Points of Failure** 10](#_Toc203732447)

[**2.** **Key Assumptions** 10](#_Toc203732448)

[**3.** **Improvements** 10](#_Toc203732449)

[Future Work 10](#_Toc203732450)

# Case Study: Multi-Modal Anomaly Detection in Oil Rig Operations Using Time Series and Text Logs

## Objective:

Design and demonstrate a prototype system that detects anomalies in oil rig operations by combining sensor time series data and operator text logs. The focus is on the approach, reasoning, and deployment-readiness, not just accuracy.

## Step:1 Simulate Data

Equipment’s selected:

### **1. Mud Pump**

* **Description:** Circulates drilling mud into the wellbore to cool the bit and transport cuttings.
* **Normal Ranges:**
  + Pressure: 110–130 psi
  + Flow Rate: 80–100 L/min
  + Temperature: 50–65°C
* **Anomalies:**
  + Pressure < 90 psi → Leak
  + Temp > 75°C → Overheating
* Flow < 60 L/min → Obstruction
* **Data Simulation Approach:**
  + Use sinusoidal + random walk for pressure and flow.
  + Inject sudden drops or gradual drifts for anomalies.
  + **Columns:** timestamp, equipment\_id, pressure\_psi, flow\_rate\_lpm, pump\_temp\_C
* **Operator Logs Generation:**
  + For each anomaly injected, a synthetic operator log is created.
  + Logs include:
    - Timestamp
    - Equipment name
    - Operator (random from fixed list)
    - Shift (A/B/C)
    - Observation/alert description (from pre-defined templates)
    - Sensor snapshot at the anomaly timestamp

### **2. Centrifugal Pump**

* **Description**: Transfers fluids using centrifugal force; relies on consistent vibration and current draw.
* **Normal Ranges:**
  + Vibration: < 5 mm/s
  + Bearing Temp: 45–60°C
  + Motor Current: 50–70 A
* **Anomalies:**
  + Vibration > 7 mm/s → Cavitation
  + Temp > 70°C → Bearing failure
  + Current > 80 A → Impeller blockage
* **Data Simulation Approach:**
  + Simulate stable vibration with noise; inject random spikes.
  + Gradual rise in temp/current to simulate component wear.
  + Columns: timestamp, equipment\_id, vibration\_mmps, bearing\_temp\_C, motor\_current\_A
* **Operator Logs Generation:**
  + One log per anomaly event, capturing time, sensor state, and context.
  + Helps in identifying early signs of wear or cavitation from operator feedback.

### **3. Blowout Preventer (BOP)**

* **Description:** High-pressure safety system that seals and controls the well during drilling.
* **Normal Ranges:**
  + Hydraulic Pressure: 1500–3000 psi
  + Accumulator Pressure: ~3000 psi
  + Valve Position: OPEN/CLOSED
* **Anomalies:**
  + Hydraulic Pressure < 1400 psi → Leak
  + Stuck valve → No status change
  + Sudden pressure drop → Accumulator issue
* **Data Simulation Approach:**
  + Create status toggles with pressure response.
  + Inject faults with frozen valve state or sudden drops.
  + **Columns:** timestamp, equipment\_id, hydraulic\_pressure\_psi, accumulator\_pressure\_psi, valve\_status, command\_signal
* **Operator Logs Generation:**
  + One log per anomaly event, capturing time, sensor state, and context.
  + Helps in identifying early signs of wear or cavitation from operator feedback.

### **4. Shale Shaker**

* **Description:** Separates solid drill cuttings from mud using vibrating screens.
* **Normal Ranges:**
  + Vibration Speed: 1800–2200 RPM
  + Motor Load: 60–75%
  + Throughput: 300–400 L/min
* **Anomalies:**
  + Vibration > 2500 RPM → Imbalance
  + Motor Load > 85% → Blockage
  + Throughput < 250 L/min → Clog/tear
* **Data Simulation Approach:**
  + Sinusoidal + noise for vibration; throughput linked to load.
  + Inject sudden spikes or drops for screen faults.
  + **Columns:** timestamp, equipment\_id, vibration\_rpm, motor\_load\_pct, throughput\_lpm
* **Operator Logs Generation:**
  + Simulated operators note issues like excessive shaking or low fluid output.
  + Helps correlate sensor spikes with on-field observations.

### **5. Diesel Generator**

* **Description:** Provides continuous power to the rig; must run efficiently across varying loads.
* **Normal Ranges:**
  + Load: 60–80%
  + Oil Pressure: 35–50 psi
  + Fuel Level: 40–100%
  + Engine Temp: 75–95°C
* **Anomalies:**
  + Load > 90% → Overload
  + Oil Pressure < 30 psi → Damage
  + Engine Temp > 100°C → Overheat
  + Fuel Drop > 10% in 15 min → Leak
* **Data Simulation Approach:**
  + Load as smoothed random walk; fuel decreases slowly.
  + Inject temperature surges and oil pressure dips.
* **Columns:** timestamp, equipment\_id, load\_pct, engine\_temp\_C, fuel\_level\_pct, oil\_pressure\_psi
* **Operator Logs Generation:**
  + Includes notes like fuel leak suspicion, overheating alerts, or load spikes.
  + Logs provide human context to correlate unexpected changes.

## Step:2 Preprocess Data

This step involves cleaning, transforming, and standardizing raw sensor data files using a modular and reusable preprocessing pipeline.

### **Timestamp Parsing & Feature Engineering**

Parses timestamps, sets them as the index, and enriches the data with time-based features such as hour, day, weekday, and weekend flags. It also supports optional resampling and interpolation of numeric data to create uniform time intervals.

### **Missing Value Handling**

Handles missing values using configurable strategies like forward fill, backward fill, mean, or median imputation. Also provides an option to drop rows with any missing values entirely.

### **Preprocessing Pipeline**

Combines timestamp handling and missing value treatment into a unified pipeline. This ensures that sensor data is cleaned, resampled, and standardized in a consistent manner.

### **Batch Processing – All Equipment**

Processes multiple CSV files representing different equipment by applying the preprocessing pipeline to each. Saves the cleaned and transformed datasets into a specified output directory with structured logging.

## Step:3 Anomaly Detection

### **Configuration & Setup**

* Sets directories for reading processed data and saving anomaly detection outputs.
* Creates output folder if it doesn't exist.
* Defines hyperparameters for each detection model:
  + Isolation Forest contamination rate
  + One-Class SVM nu values
  + LSTM Autoencoder time window, epochs, and contamination threshold

### **Load Processed Sensor Files**

* Loads all preprocessed CSV files from the specified input directory.
* Each file represents sensor readings from a different equipment unit.
* Stores them in a dictionary format for batch processing.

### **Isolation Forest (Tree-Based)**

* Runs a grid search over various contamination values to evaluate anomaly counts per file.
* Creates a pivot table summarizing anomaly counts per contamination level across all files.
* Executes Isolation Forest on each dataset using a selected contamination value.
* Saves annotated datasets with detected anomalies to the output directory.

### **One-Class SVM (Kernel-Based)**

* Runs a grid search over different nu values to assess anomaly counts.
* Generates a pivot table showing how the anomaly counts vary across values and files.
* Applies One-Class SVM to all equipment datasets using a chosen nu value.
* Drops conflicting anomaly columns if they exist (e.g., from Isolation Forest).
* Saves results with One-Class SVM anomaly annotations.

### **LSTM Autoencoder (Deep Learning)**

* Applies LSTM Autoencoder model to each file.
* Detects anomalies based on reconstruction error.
* Dynamically extracts numeric sensor columns for model input.
* Drops previous anomaly flags before saving.
* Saves anomaly-detected files and logs the reconstruction threshold used.

## Step:4 Text embedding and correlation

### **Load Anomaly Outputs**

* Loads all previously saved CSV files that contain anomaly detection results.
* Each file corresponds to a specific equipment type and contains timestamped sensor data with anomaly flags.

### **Parse Unstructured Operator Logs**

* Reads raw text logs written by operators during rig operations.
* Extracts structured information including:
  1. Timestamps
  2. Equipment types (e.g., blowout preventer, drill pipe)
  3. Freeform log text messages

### **Generate Semantic Embeddings**

* Loads a pretrained sentence embedding model (all-MiniLM-L6-v2).
* Converts each operator log message into a high-dimensional embedding vector.
* Similarly, generates embeddings for anomaly descriptions to enable semantic comparison.

### **Anomaly–Log Matching (Semantic Correlation)**

* For each anomaly in each dataset:
  1. Matches log entries within a specified time window (±240 minutes).
  2. Computes cosine similarity between the anomaly text and log message embeddings.
  3. Filters matches above a defined similarity threshold.
  4. Extracts sensor readings and metadata associated with each matched anomaly.

### **Result Compilation**

* For each correlated anomaly-log pair:
  1. Records anomaly time, log time, equipment type, texts, and similarity score.
  2. Includes all associated sensor readings for context.
* Sorts and keeps the top 20 most similar correlations per anomaly file.

### **Save Correlation Output**

* Merges all correlation results across equipment.
* Saves the final structured correlation table as a CSV file.
* Creates output directory if it doesn’t exist.

## Step:5 Insight generation

### **Load API Key**

* Loads the OpenAI API key from a YAML configuration file.
* Ensures secure and modular access to the model authentication credentials.

### **Prompt Construction Per Equipment**

* Iterates over anomaly-log correlation entries grouped by equipment\_type.
* For each group, constructs a detailed prompt containing:
  1. The anomaly and operator log context.
  2. Clear instructions to output a **tabular summary** with standardized fields:
     1. Equipment name
     2. Total anomalies
     3. Most recent anomaly details
     4. Diagnosis and likely root cause
     5. Relevant operator log
     6. Actionable maintenance review

### **Language Model Interaction**

* Sends each equipment-specific prompt to the OpenAI API (e.g., GPT-3.5-turbo).
* Controls output format using:
  1. Moderate temperature for balanced creativity and accuracy.
  2. Token limit to manage response length.
* Extracts and stores the LLM-generated summary text.

### **Postprocessing and Storage**

* Collects all generated summaries into a structured DataFrame.
* Saves final output as a CSV file, where each row corresponds to one equipment type and its diagnosis summary.

## Step:6 Streamlit-Based Real-Time Monitoring Dashboard

### **App Setup & Styling**

* Configures the Streamlit app with a custom title, icon, and wide layout.
* Applies custom CSS to:
  1. Hide default Streamlit UI elements (e.g., footer/menu).
  2. Enforce consistent typography.
  3. Implement a sticky header for persistent visibility.
  4. Disable vertical scrolling to keep views contained on one screen.

### **File Mapping & Directory Initialization**

* Defines paths for key data categories:
  1. Raw equipment sensor logs
  2. Detected anomaly results
  3. Anomaly-log correlation outputs
  4. Final anomaly summary table
* Dynamically maps file names into clean, user-friendly dropdown labels.

### **Sidebar Navigation**

Adds a sidebar radio menu for switching between the following dashboard views:

* **Equipment Logs**
* **Anomaly Detection**
* **Correlations**
* **Anomaly Summary**

### **Equipment Logs View**

* Loads and displays raw sensor log files per selected equipment.
  1. Enables filtering by:
  2. Date range (via detected time columns)
  3. Numeric parameter ranges (via sliders)
* Shows filtered logs as a responsive table.

### **Anomaly Detection View**

* Displays anomaly detection output from prior model runs.
  1. Filters by:
  2. Equipment type
  3. Time range
  4. Numeric anomaly indicators (e.g., scores or sensor values)
* Allows inspection of time-localized anomaly events.

### **Correlation View**

* Loads and displays log-anomaly correlation entries generated earlier.
* Allows filtering by equipment name.
* Useful for verifying alignment between detected anomalies and operator observations.

### **Anomaly Summary View**

* Loads the final, **GPT-generated diagnosis summaries**.
* Users can view summaries for:
  1. All equipment combined
  2. A specific equipment type
* Each card-style section includes:
  1. Total anomaly count
  2. Latest anomaly timestamp and description
  3. Root cause diagnosis
  4. Relevant operator log snippet
  5. Actionable maintenance recommendations

## Critical Analysis of the Approach

### **Potential Points of Failure**

* **Noisy Logs**: Unstructured or irrelevant operator logs may confuse embeddings and reduce context accuracy.
* **False Positives**: Time-window-based correlation can wrongly link unrelated anomalies and logs.
* **Prompt Sensitivity**: GPT summaries depend heavily on the quality of prompt formatting and may hallucinate details.

### **Key Assumptions**

* **Logs and anomalies are timestamped accurately and meaningfully.**
* **Similarity between log-anomaly pairs is sufficient for meaningful grouping.**
* **GPT can interpret domain-specific anomalies with general instruction prompts.**

### **Improvements**

1. Add **sensor metadata**, severity scores, or anomaly types for richer context.
2. Use **domain-tuned models** (e.g., finetuned GPT or SciBERT).
3. Include **human-in-the-loop validation** or expert feedback to refine root cause accuracy.
4. Integrate **log classification models** to filter noise before correlation.

## Future Work

**1. Domain-Specific Fine-Tuning**  
Train or fine-tune LLMs (e.g., GPT, SciBERT) on oil rig logs and failure reports for more accurate diagnostics.

**2. Advanced Log Preprocessing**  
Use NLP techniques to denoise logs, extract structured entities (equipment names, failure types), and improve relevance.

**3. Anomaly Categorization**  
Auto-label anomalies by severity/type to prioritize analysis and enable dashboard filtering.

**4. Real-Time Streaming Support**  
Extend pipeline to support real-time anomaly detection and contextual logging using Kafka or similar frameworks.

**5. Feedback Loop Integration**  
Collect expert feedback on generated summaries to iteratively improve model prompts, validation, and performance.

**6. Sensor Fusion Analytics**  
Combine logs with structured sensor signals for multi-modal anomaly reasoning.

**7. Confidence Scoring**  
Add trust metrics or confidence levels to each generated diagnosis using ensemble methods or uncertainty modeling.

**8. Root Cause Explanation Graphs**  
Visualize anomaly-log-sensor connections via explainable AI frameworks (e.g., SHAP, LIME for sensor-driven models).