Time Series Anomaly Detection for IoT Sensors – Summary Report

# 1. Problem Understanding and Approach

Objective: Detect anomalies in multivariate time series data from IoT sensors monitoring manufacturing equipment to identify early signs of failure or maintenance needs.  
Dataset: NASA IMS Bearing Dataset with multiple test folders containing time series snapshots (~20,480 points each). Data completeness and consistency were verified—no missing values were found.  
Approach:  
1. Exploratory Data Analysis (EDA): Examine sensor distributions, correlations, and outliers.  
2. Feature Engineering: Extract meaningful time-domain and frequency-domain features to capture normal and abnormal patterns.  
3. Labeling: Heuristic labeling marking the last 10% of snapshots as anomalous to approximate bearing degradation.  
4. Modeling: Use both statistical and deep learning models to detect anomalies.  
5. Evaluation: Assess model performance using precision, recall, F1-score, and ROC-AUC.

# 2. Feature Engineering Rationale

Feature engineering combined domain knowledge and statistical methods to capture key characteristics of sensor vibrations.  
Time-Domain Features:  
- Mean, RMS, standard deviation, skewness, kurtosis, median, rolling statistics  
- Capture signal amplitude, variability, and trends over time  
  
Frequency-Domain Features:  
- Spectral centroid, peak frequency, spectral entropy, energy ratios via Welch's method  
- Capture frequency shifts and periodic patterns indicative of mechanical degradation  
  
Additional Steps:  
- Outlier treatment using IQR and z-score methods  
- Robust scaling to mitigate noise and enhance model stability  
  
Rationale: Combining both domains improves interpretability and allows models to detect both subtle temporal changes and sudden spikes.

# 3. Model Selection and Comparison

Two models were used for anomaly detection:  
1. Isolation Forest: Unsupervised model detecting statistical outliers, effective for simple statistical anomalies but limited for temporal dependencies.  
2. LSTM Autoencoder: Recurrent neural network trained to reconstruct normal sequences; high reconstruction error indicates anomalies; captures complex temporal dependencies and nonlinear relationships.

Comparison of Results (Sample Metrics):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Test Folder | Model | Precision | Recall | F1-score | ROC AUC |
| 1sttest | Isolation Forest | 0.34 | 0.17 | 0.23 | 0.57 |
| 1sttest | LSTM Autoencoder | 0.39 | 0.19 | 0.26 | 0.58 |
| 2ndtest | Isolation Forest | 0.98 | 0.50 | 0.66 | 0.75 |
| 2ndtest | LSTM Autoencoder | 0.92 | 0.47 | 0.62 | 0.73 |
| 3rdtest | Isolation Forest | 0.23 | 0.11 | 0.15 | 0.54 |
| 3rdtest | LSTM Autoencoder | 0.00 | 0.00 | 0.00 | 0.47 |

# 4. Key Findings and Business Insights

- Feature engineering is critical; combining time and frequency domain features enhances model effectiveness.  
- Deep learning models (LSTM Autoencoder) capture temporal trends that statistical models miss.  
- Heuristic labels approximate degradation patterns, but true labels or expert annotations would improve reliability.  
- Potential for cross-dataset training to enhance robustness for different operating conditions.  
- Early detection of anomalies can reduce downtime, prevent equipment failures, and lower maintenance costs.

# 5. Limitations and Future Improvements

Limitations:  
- Heuristic labeling may not perfectly align with actual failure events.  
- Models may struggle with high-noise datasets or unseen operating conditions.  
- LSTM Autoencoder requires careful tuning and more data for generalization.  
  
Future Improvements:  
- Incorporate true labels or expert annotations for supervised learning.  
- Extend models with attention mechanisms or ConvLSTM for improved temporal modeling.  
- Implement real-time anomaly detection pipelines with automated alerts.  
- Explore multimodal sensor fusion to improve fault diagnosis.