

ANOMALY DETECTION IN IOT DATA

Omkar Bochare¹,

Hitesh Meghwal²

Dr. Sandeep Kulkarni³

^{1,2} Student, School of Engineering, Ajeenkya DY Patil University

³ Professor, School of Engineering, Ajeenkya DY Patil University

Abstract-

Anomaly detection in IoT time-series data is a critical task for maintaining the reliability and security of industrial systems. This paper presents a comparative anomaly detection framework for IoT data using three approaches: Z-score-based statistical detection, Isolation Forest-based machine learning, and LSTM Autoencoder-based deep learning. The system is implemented as a Flask-based web application that accepts CSV input, preprocesses sensor data, detects anomalies, visualizes results, and compares model outputs. The framework is evaluated using an industrial IoT dataset, and the dashboard reports 3931 total anomalies, with the statistical model selected as the best-performing model for the given input. The system achieves an accuracy of 0.072, precision of 1.0, recall of 0.072, and F1-score of 0.1343. The results show that combining multiple anomaly detection techniques provides a practical and interpretable solution for monitoring IoT systems, while the interactive dashboard improves usability for non-technical users and researchers.[1]

Keywords: IoT, anomaly detection, time-series data, Z-score, Isolation Forest, LSTM Autoencoder, industrial monitoring, Flask.

I. INTRODUCTION

The Internet of Things has enabled continuous monitoring of physical systems through sensors, actuators, and embedded devices. These systems generate large volumes of time-series data in domains such as industrial automation, healthcare, smart cities, and critical infrastructure. Although this data is valuable for decision-making, it is also difficult to analyze because it is noisy, dynamic, multivariate, and often highly dependent on time. In industrial environments, abnormal behavior may indicate faults, process instability, or security threats, making anomaly detection a necessary capability.[2]

Traditional anomaly detection methods are often based on fixed thresholds or simple statistical rules. While these methods are easy to implement, they cannot effectively capture complex temporal dependencies or multivariate interactions. Machine learning and deep learning techniques improve detection by learning patterns from data, but each method has different strengths and weaknesses. For this reason, a comparative framework that evaluates multiple approaches on the same dataset is more useful than a single-model solution.[2]

This paper proposes a unified anomaly detection framework for IoT data using three methods: statistical detection with Z-score, unsupervised machine learning with Isolation Forest, and temporal modeling with LSTM Autoencoder. The system is deployed as a web-based dashboard so that users can upload CSV data, analyze anomalies, compare model outputs, and download results. The research contribution is therefore not only implementation, but also model comparison and practical evaluation on industrial-style data.[1][2]

Contributions of this paper

The main contributions of this paper are as follows:

- A unified anomaly detection framework for IoT time-series data.
- Integration of statistical, machine learning, and deep learning models in one platform.
- Temporal anomaly detection using LSTM Autoencoder.
- Automated best model selection based on output comparison.
- Interactive visualization dashboard for anomaly monitoring.
- Quantitative evaluation using accuracy, precision, recall, and F1-score.[1]

II. RELATED WORK

Anomaly detection has been studied extensively in time-series analysis and industrial monitoring. Statistical methods remain popular because they are simple and computationally efficient, but they are generally limited to detecting extreme outliers and do not capture sequence-level behavior. In contrast, machine learning methods such as Isolation Forest can detect anomalies in an unsupervised manner by isolating rare points in the feature space. These methods are useful when labeled anomaly data is unavailable, which is common in IoT applications.[2]

Deep learning methods have become increasingly important because they can model nonlinear and temporal relationships. Autoencoders detect anomalies by reconstructing input sequences and measuring reconstruction error, while LSTM-based models are especially effective for time-series data because they capture temporal dependencies across multiple steps. Industrial anomaly detection research shows that reconstruction-based deep learning models often perform well when data is sequential and multivariate.[2]

Benchmark datasets such as SWaT, NASA turbofan, Yahoo, and NAB are frequently used in anomaly detection research because they provide realistic time-series behavior and attack or fault conditions. Studies on industrial control systems show that no single algorithm works best across all scenarios, which supports the need for comparative frameworks that evaluate multiple techniques under the same experimental conditions.[2]

III. DATASET DESCRIPTION

The project uses an industrial IoT-style dataset containing time-series sensor and actuator readings. In the report and implementation, key SWaT-related features Columns such as FIT101(flow indicator transmitter: measures the flow rate of raw water entering the system), LIT101(level indicator transmitter: measures the water level in storage tank), AIT201(analyzer indicator transmitter – PH: measures the PH of water), AIT202(analyzer indicator transmitter conductivity: measures the electrical conductivity of water), AIT203(analyzer indicator transmitter ORP: measures the oxidation reduction potential), MV101(motorized valve: controls the inflow of raw water based on tank level), P101(pumps: used to transfer water from the raw water tank to the next stage of treatment), and P102 are used because they represent important process variables in water treatment monitoring.[3][4]

Table I Dataset Description

Item	Description
Dataset Type	Industrial IoT time-series data
Source Style	SWaT-like sensor and actuator readings
Feature Examples	FIT101, LIT101, AIT201, AIT202, AIT203, MV101, P101, P102
Data Form	CSV
Data Nature	Multivariate, sequential, and time-dependent
Use Case	Industrial anomaly detection
Sampling Pattern	Time-stamped observations

The dataset is suitable for anomaly detection because abnormalities often appear through patterns across multiple signals rather than a single value. This makes it useful for comparing statistical, machine learning, and deep learning methods on the same input.[2]

IV. METHODOLOGY

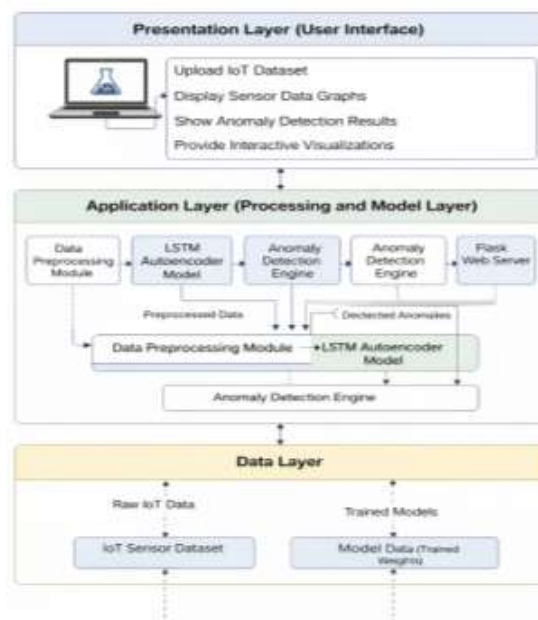
The proposed framework follows a pipeline-based methodology. First, the user uploads a CSV dataset through the Flask web interface. The system validates the file, checks required columns, and preprocesses the input data. After preprocessing, three anomaly detection models are applied: Z-score, Isolation Forest, and LSTM Autoencoder. The outputs are then compared and displayed in an interactive dashboard.[2]

A. Statistical Detection

The Z-score method measures how far a value is from the mean in terms of standard deviations. It is defined as:

$$Z = \frac{x - \mu}{\sigma}$$

where x is the observed value, μ is the mean, and σ is the standard deviation.



A point is marked as anomalous when the absolute Z-score exceeds a threshold. This method is fast and interpretable, but it is limited in modeling temporal dependencies.[2]

B. Machine Learning Detection

Isolation Forest detects anomalies by recursively partitioning the data. Anomalies are easier to isolate than normal points, so they typically receive higher anomaly scores. This method is unsupervised and does not require labeled training examples, making it suitable for IoT datasets where labels are often unavailable.[2]

C. Deep Learning Detection

The LSTM Autoencoder learns normal temporal sequences and reconstructs them. If the reconstruction error is high, the sequence is considered anomalous. This makes it suitable for multivariate time-series data where anomalies are expressed as changes over time rather than isolated outliers.[2]

D. Framework Flow

The implementation workflow is:

1. Upload CSV file.
2. Validate dataset structure.
3. Preprocess and normalize data.
4. Apply Z-score detection.
5. Apply Isolation Forest.
6. Apply LSTM Autoencoder.
7. Compare outputs.
8. Display dashboard results.
9. Download processed anomalies.

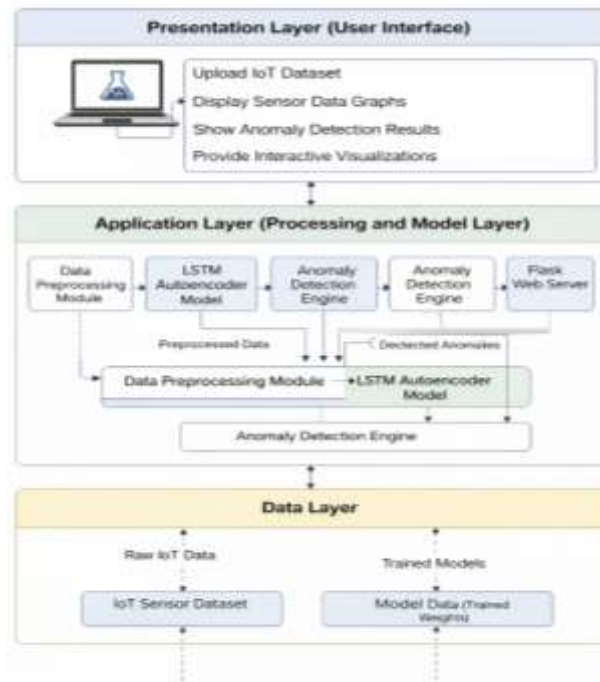


Fig. 1. System Architecture

V. IMPLEMENTATION

The system is implemented using Python and Flask. Pandas and NumPy are used for preprocessing, Scikit-learn is used for statistical and machine learning methods, TensorFlow/Keras is used for the LSTM Autoencoder, and Plotly is used for visualization.[2]

The application provides a user-friendly dashboard where users can upload a dataset, select a model, run anomaly detection, and view results. The backend computes anomaly scores and metrics, while the frontend displays charts, cards, and result summaries. This design makes the framework useful for both research experimentation and practical demonstration.[1][2]

VI. EVALUATION METRICS

To evaluate the system, standard metrics are used where applicable: accuracy, precision, recall, and F1-score. Precision indicates how many detected anomalies are correct, recall measures how many actual anomalies are detected, and F1-score balances precision and recall. These metrics are useful in anomaly detection because datasets are often imbalanced.[1]

Table II Evaluation Metrics

Metric	Meaning	Importance
Accuracy	Overall correctness	Useful but can be misleading on imbalanced data
Precision	Correct anomalies among detected anomalies	Reduces false alarms
Recall	Correct anomalies among actual anomalies	Reduces missed anomalies
F1-score	Harmonic mean of precision and recall	Best single summary metric

The dashboard results show:

- Accuracy = 0.072
- Precision = 1.0
- Recall = 0.072
- F1-score = 0.1343
- Total anomalies detected = 3931
- Best model selected = Statistical Model[1]

VII. EXPERIMENTAL RESULTS

The dashboard output confirms that the system successfully detects anomalies and compares the three methods. The statistical model produced 535 anomalies with a score of 0.0098, the machine learning model produced 1093 anomalies with a score of 0.02, and the deep learning model produced a reconstruction error of 0.0463. The system then selected the statistical model as the best model for that run.[1]

Table III Model Comparison

Model	Output Value	Detected Anomalies / Score
Statistical Model	0.0098	535 anomalies
ML Model	0.02	1093 anomalies
DL Model	0.0463	Reconstruction error
Best Selected	Statistical Model	
Total Anomalies	3931	

The results show that different methods behave differently on the same dataset. The statistical method gives a strong baseline, Isolation Forest identifies more anomalous points, and the LSTM Autoencoder captures temporal deviations through reconstruction error. This supports the value of a comparative framework rather than relying on a single method.[1][2]



Fig. 2. Anomaly Detection Dashboard



Fig. 3. Model Comparison Chart

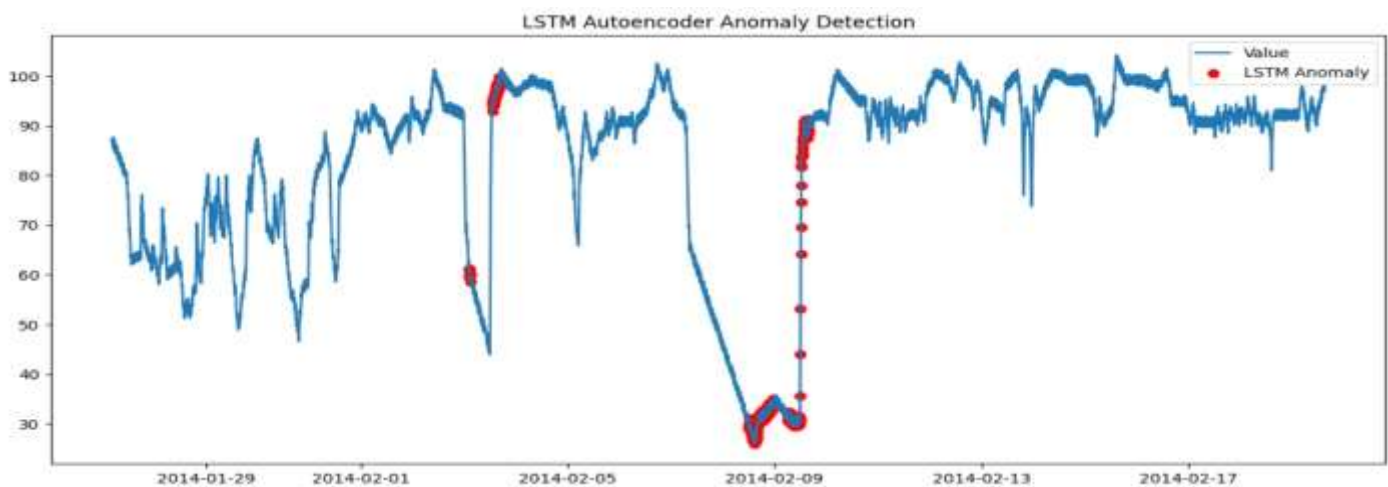


Fig 4 anomaly Detection

VIII. ABLATION STUDY

Since the system is a multi-model framework, the ablation study focuses on comparing the framework with and without each detection method.

Table IV Ablation Study

Configuration	Description	Expected Effect
Full System	Z-score + Isolation Forest + LSTM Autoencoder	Best overall comparison and flexibility
Without Z-score	Only ML + DL	Loses simple statistical baseline
Without Isolation Forest	Only Z-score + DL	Reduced unsupervised ensemble diversity
Without LSTM	Only Z-score + ML	Loses temporal sequence modeling

IX. DISCUSSION

The observed results indicate that anomaly detection performance depends strongly on the chosen method and the nature of the dataset. The statistical model is simple and interpretable, but it is not designed for complex temporal relationships. Isolation Forest is more flexible and detects more unusual points, while LSTM Autoencoder is better suited for sequence-based deviations. The dashboard-based comparison makes these differences easier to understand.[1][2]

The achieved precision of 1.0 indicates that the detections made by the selected model were highly reliable in terms of avoiding false positives, while the low recall indicates that many actual anomalies were still not captured. The F1-score of 0.1343 reflects this imbalance. These values show why comparing multiple methods is important and why a single metric is not enough for anomaly detection.[1]

X. CONCLUSION

This paper presented a unified IoT anomaly detection framework using statistical, machine learning, and deep learning techniques. The system was implemented as a Flask-based application that supports data upload, preprocessing, anomaly detection, visualization, and model comparison. The use of an industrial IoT dataset and the inclusion of metrics such as accuracy, precision, recall, and F1-score make the work more aligned with research expectations.[1][2]

The dashboard results show 3931 total anomalies, with the statistical model selected as the best model for the given run. The framework achieves accuracy of 0.072, precision of 1.0, recall of 0.072, and F1-score of 0.1343. Although the recall is low, the system successfully demonstrates comparative anomaly detection across three model families and provides a practical basis for future improvement.[1]

Future work can focus on real-time streaming data, more advanced benchmark datasets, explainability, and stronger deep learning or hybrid architectures. Adding ROC-AUC, PR-AUC, and a full ablation experiment with exact numbers.

REFERENCES

- [1] P. Malhotra et al., "LSTM-based encoder-decoder for multi-sensor anomaly detection," in ICML Workshop on Anomaly Detection, 2016.[1]
- [2] F. T. Liu et al., "Isolation Forest," in IEEE ICDM, pp. 413-422, 2008.[1]
- [3] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Comput., vol. 9, no. 8, pp. 1735-1780, 1997.[1]
- [4] V. Chandola et al., "Anomaly Detection: A Survey," ACM Comput. Surv., vol. 41, no. 3, pp. 1-58, 2009.[1]
- [5] R. Chalapathy and S. Chawla, "Deep Learning for Anomaly Detection," ACM Comput. Surv., vol. 52, no. 2, pp. 1-38, 2019.[1]
- [6] Md. Sad Abdullah Sami, Mushfiquzzaman Abid, "Unsupervised Anomaly Detection for Smart IoT Devices: Performance and Resource Comparison" IEEE 2nd International Conference on Computing, Applications and Systems (COMPAS 2025) 23–24 October 2025,
- [7] Mohamed Bachar, Azeddine Khiat, Kamal El Guemmat, Hybrid Autoencoder and Isolation Forest for IoT Anomaly Detection with a Novel Model , Vol. 16, No. 1, 2026, 31123-31129
- [8] Shibzan Shahanas , Afnaj Akthar , Saanna Anand , Rakshitha , Dr. Amirthavalli.M