Communications on Applied Nonlinear Analysis
ISSN: 1074-133X
Vol 33 No. 1s (2026)

An AI-Driven Framework for Anomaly Detection and Data Correction Using Frequent Activity Pattern Learning

1st CH. Naresh
Assistant Professor
Artificial Intelligence & Data Science
Vignan Institute of Technology and Science
Hyderabad, Telangana State, India
challanaresh8@gmail.com

2nd B. Srikanth Reddy
Student
Artificial Intelligence & Data Science
Vignan Institute of Technology and Science
Hyderabad, Telangana State, India
srikanthreddy59338@gmail.com

3rd M. Thanmay Sadguru
Student
Artificial Intelligence & Data Science
Vignan Institute of Technology and Science
Hyderabad, Telangana State, India
thanmaysadguru5912@gmail.com

4th CH. Jhansi RamCharan Teja
Student
Artificial Intelligence & Data Science
Vignan Institute of Technology and Science
Hyderabad, Telangana State, India
ramcharanch25024@gmail.com

Article History:
Received: 04-02-2026
Revised: 20-03-2026
Accepted: 10-04-2026

Abstract:

With the increasing adoption of smart environments and sensor-based intelligent systems, large volumes of behavioral activity data are continuously generated. Such datasets often contain unusual patterns caused by abnormal user behavior, system malfunctions, environmental variations, or data inconsistencies. Traditional monitoring approaches, which rely on manual inspection or rule-based techniques, are often inadequate for identifying complex anomalies present in large-scale and high-dimensional behavioral datasets.This project proposes a machine learning-based automated anomaly detection framework designed to

identify abnormal patterns in behavioral sensor activity data. The proposed system utilizes multiple anomaly detection techniques, including Isolation Forest, One-Class Support Vector Machine (OC-SVM), Autoencoder, and LSTM-based sequence anomaly detection. The dataset is preprocessed to extract meaningful behavioral features such as sensor identification, activity state, time of occurrence, and day-wise activity patterns. These features enable the models to learn normal behavioral trends and assign anomaly scores to individual events. Events with significantly high anomaly scores are classified as abnormal. Experimental analysis demonstrates that the proposed framework provides an efficient, scalable, and intelligent solution for anomaly detection in smart environments and large behavioral datasets.

Keywords: Anomaly Detection, Machine Learning, Deep Learning, Isolation Forest, One-Class Support Vector Machine, Autoencoder, LSTM, Behavioral Data Analysis, Artificial Intelligence, Sensor Activity Monitoring

INTRODUCTION

Groundwater quality is a critical determinant of human health, agricultural productivity, and ecosystem balance. With the growing global dependence on groundwater as a primary water source, safeguarding its quality has become increasingly important. However, rapid industrialization, excessive pesticide consumption, and uncontrolled groundwater extraction have led to significant contamination risks. Traditional laboratory-based water testing methods, although accurate, are often slow, expensive, and unable to scale efficiently in regions that require frequent and distributed monitoring. This creates an urgent need for innovative, automated, and accessible solutions that can provide timely insights into groundwater safety. Artificial intelligence (AI) has emerged as a powerful tool for environmental monitoring, particularly in the classification and prediction of water quality parameters. By analyzing chemical characteristics such as pH, turbidity, hardness, total dissolved solids, electrical conductivity, and chloride levels, machine learning models can assess groundwater suitability more rapidly than conventional methods. However, classification alone is not sufficient for effective decision-making, especially for non-technical users such as farmers, households, and community administrators. To address this gap, integrating Generative AI (GenAI) with machine learning models offers a transformative approach that not only predicts water usability but also provides intelligent, personalized recommendations. The proposed system utilizes advanced machine learning classifiers to determine the suitability of drinking and irrigation, assess severity levels, and categorize groundwater into safe, moderate, or unsafe conditions. Building upon these outputs, a GenAI-based recommendation layer interprets the results and generates human-friendly insights, including potential health risks, corrective measures, and agricultural guidance.

II. REVIEW OF RELATED WORK

Recent advancements in machine learning and deep learning have significantly improved the effectiveness of anomaly detection systems across smart environments, industrial automation, autonomous platforms, and IoT infrastructures. Traditional anomaly detection techniques largely relied on statistical thresholds, rule-based monitoring, or manual supervision, which are often insufficient for identifying complex behavioral deviations and temporal anomalies in high-dimensional sensor data. To address these challenges, researchers have increasingly explored intelligent anomaly detection frameworks that integrate advanced machine learning algorithms, deep learning architectures, and data-driven optimization strategies.

Lian et al. (2023) proposed a unified anomaly detection and correction framework aimed at optimizing the performance of autonomous intelligent systems using inverse reinforcement learning (IRL). Unlike conventional fault detection methods that focus on static system behavior, this research emphasizes anomalies that arise when agents deviate from their intended objective functions. The framework categorizes anomalies into noise anomalies, intention anomalies, and objective function anomalies, enabling a structured analysis of abnormal operational patterns. Through model-free IRL algorithms, the system reconstructs normal behavior trajectories and implements corrective mechanisms to restore optimal functionality. Experimental validation conducted on a quadrotor unmanned aerial vehicle platform demonstrated real-time detection and correction of abnormal behaviors, highlighting the framework's practical applicability in autonomous control systems.

Similarly, Choi et al. (2021) conducted a comprehensive review focusing on deep learning-based anomaly detection techniques for multivariate time-series datasets commonly observed in industrial and cyber-physical systems. Their study emphasizes the capability of deep neural networks to capture complex temporal dependencies and inter-variable correlations in large sequential datasets. By analyzing various state-of-the-art deep anomaly detection architectures, the authors provide insights into model selection strategies, deployment considerations, and training optimization approaches. The study concludes that incorporating domain-specific knowledge and contextual information significantly enhances anomaly detection accuracy in real-world monitoring applications.

Song and Li (2024) introduced a novel data-driven anomaly detection framework based on Transformer-inspired deep learning architectures tailored for industrial multivariate time-series data. Their proposed improved Saxformer model integrates a Boosting-based feature enhancement mechanism that strengthens the representation of sequential patterns and improves prediction stability under noisy operational conditions. By refining the Transformer output layer and incorporating fine-grained control mechanisms, the framework demonstrates enhanced robustness and detection reliability in dynamic industrial environments. The research illustrates that combining attention-based sequence modeling with ensemble optimization techniques can substantially improve anomaly identification performance in modern analytics systems.

A. LITERATURE REVIEW

Anomaly detection in intelligent and sensor-driven environments presents significant challenges due to the complexity of temporal behavioral patterns, data heterogeneity, and the dynamic nature of operational systems. Early anomaly detection approaches relied heavily on statistical modeling

and rule-based monitoring techniques, which, although simple to implement, lacked the capability to identify subtle nonlinear anomalies in large-scale datasets. Recent research trends indicate a growing transition toward machine learning and deep learning-based methods capable of capturing hidden behavioral patterns and improving detection accuracy.

Deep learning-based anomaly detection frameworks, as highlighted by Choi et al. (2021) and Song and Li (2024), demonstrate superior performance in handling multivariate time-series data by leveraging architectures such as recurrent neural networks, attention-based models, and Transformer variants. These models effectively capture long-term temporal dependencies and complex feature interactions, enabling reliable identification of abnormal sequences in industrial and cyber-physical systems. Ensemble learning strategies and boosting-based feature enhancement techniques further contribute to improving detection robustness and reducing false alarm rates.

Machine learning-driven anomaly detection approaches discussed by Nassif et al. (2021) emphasize the effectiveness of unsupervised learning techniques in real-world scenarios where labeled anomaly data is limited. Algorithms such as Isolation Forest, Support Vector Machines, and clustering-based methods provide scalable solutions for detecting rare or emerging anomalies. Meanwhile, adaptive intelligent frameworks such as the inverse reinforcement learning-based system proposed by Lian et al. (2023) introduce a novel perspective by focusing on behavioral intention deviations in autonomous agents, thereby expanding the scope of anomaly detection beyond conventional system fault identification.

B. PROPOSED SOLUTION

The proposed anomaly detection system is implemented using Python due to its extensive ecosystem for machine learning, deep learning, time-series analytics, and intelligent data processing. The framework integrates multiple anomaly detection models to identify abnormal behavioral patterns in smart environment sensor datasets. Several components collectively form the core implementation pipeline:

• Data Preprocessing:

Behavioral sensor activity data, such as sensor ID, activity state, timestamp, and environmental interaction logs, are cleaned and normalized using libraries such as Pandas and NumPy. Missing values, noise disturbances, and inconsistent sensor readings are handled through statistical imputation, smoothing, and normalization techniques to ensure high-quality input data suitable for anomaly detection modeling.

• Feature Engineering:

Domain-specific behavioral features such as activity frequency, time-of-occurrence patterns, sequential activity transitions, and day-wise usage trends are extracted. Categorical attributes are encoded using label encoding or one-hot encoding methods, while numerical features are scaled to improve model learning efficiency. Temporal aggregation techniques are also applied to capture periodic behavioral patterns and contextual variations.

• Machine Learning Model Development:

Multiple anomaly detection models, including Isolation Forest and One-Class Support Vector Machine (OC-SVM), are implemented using Scikit-learn. These models learn the distribution of

normal behavioral data and assign anomaly scores to detect deviations from expected activity patterns. Hyperparameter tuning and cross-validation strategies are applied to improve detection sensitivity and reduce false positives.

• **Deep Learning-based Sequential Anomaly Detection:**

Deep neural network architectures such as Autoencoders and Long Short-Term Memory (LSTM) networks are developed using TensorFlow/Keras to capture nonlinear and temporal dependencies in behavioral sensor data. The Autoencoder model identifies anomalies based on reconstruction error, while the LSTM model detects abnormal activity sequences by analyzing prediction deviations in time-series data.

• **Anomaly Score Aggregation and Decision Mechanism:**

Outputs from multiple anomaly detection models are combined through ensemble decision strategies to enhance detection robustness and reliability. Threshold-based classification logic is designed to categorize events into normal or abnormal behavioral patterns based on aggregated anomaly scores.

• **Visualization:**

Detected anomalies, behavioral trends, and sensor activity distributions are visualized using Matplotlib and Seaborn. Graphical representations such as time-series plots, anomaly score distributions, and activity heatmaps provide intuitive insights into abnormal system behavior for effective monitoring and decision-making.

• **Model Saving and Deployment:**

Trained anomaly detection models are serialized into .pkl or .h5 formats using Pickle or TensorFlow model saving techniques for easy reuse and deployment. The integrated anomaly detection pipeline is embedded into a Flask-based web interface for interactive anomaly monitoring and analysis.

• **Web Interface (Flask):**

A user-friendly Flask web application allows users to upload behavioral datasets, visualize detected anomalies, and monitor system performance in real time. The dashboard includes anomaly alerts, activity pattern summaries, and graphical insights for improved interpretability and decision support.

• **Hardware Used:**

All experiments were conducted on a standard computing system with an Intel i5 processor and 8GB RAM. Experimental results indicate that the proposed anomaly detection framework requires moderate computational resources and is suitable for lightweight deployment in smart monitoring environments.

C. METHODOLOGY

The methodology for the proposed anomaly detection system follows a structured data-driven approach that integrates machine learning and deep learning techniques to identify abnormal behavioral patterns in sensor-driven smart environments. The process begins with data collection, where behavioral activity logs are obtained from sensor networks and intelligent monitoring

systems. These logs include parameters such as sensor identification, activity state transitions, time-of-occurrence patterns, and contextual environmental interactions. The collected dataset is preprocessed through normalization, noise filtering, and feature scaling techniques to ensure consistency and improve model training effectiveness.After preprocessing, relevant behavioral features are extracted to capture temporal dependencies and activity trends across different sensors and operational conditions. The dataset is then divided into training and testing subsets to enable reliable performance evaluation. Machine learning models such as Isolation Forest and One-Class Support Vector Machine are trained to learn the distribution of normal activity patterns and detect anomalies based on deviation scores. In parallel, deep learning architectures, including Autoencoder neural networks and LSTM-based sequence models, are implemented to analyze nonlinear relationships and sequential behavioral variations. Ensemble strategies are employed to combine the outputs from multiple models, thereby enhancing detection accuracy and reducing uncertainty in anomaly classification.

SYSTEM REQUIREMENTS

Hardware Requirements

- **Processor:** Intel Core i5 / AMD Ryzen 5 or higher
- **RAM:** Minimum 8 GB (16 GB recommended for better performance)
- **Storage:** Minimum 250 GB HDD or 128 GB SSD
- **GPU (Optional):** NVIDIA GPU with CUDA support for faster model training
- **Display:** 1366 × 768 resolution or higher
- **Network:** Stable Internet connection for GenAI integration and Flask server access
- **Sensors (Optional – IoT Integration):** pH Sensor, TDS Sensor, Turbidity Sensor, EC Sensor

Software Requirements

- **Operating System:** Windows 10/11 or Ubuntu 22.04 LTS
- **Programming Language:** Python 3.10 or higher
- **Framework:** Flask for web UI and backend integration
- **Machine Learning Libraries:** scikit-learn, XGBoost, LightGBM
- **Data Handling Libraries:** Pandas, NumPy
- **Visualization Tools:** Matplotlib, Seaborn, Plotly
- **Generative AI Tools:** OpenAI API / Hugging Face Transformers / LangChain
- **Prompt Engineering Toolkit:** LangChain or LlamaIndex
- **Database (Optional):** SQLite or MySQL for storing prediction history
- **Version Control:** Git / GitHub
- **IDE / Editor:** Visual Studio Code or PyCharm
- **Deployment Environment:** Localhost or Cloud platforms (AWS, GCP, Azure, Render)

D. ALGORITHM

Step 1: System Environment Initialization

Import essential machine learning and deep learning libraries such as NumPy, Pandas, Scikit-learn, TensorFlow/Keras, and Matplotlib for data analysis and model development.

Initialize necessary modules for data handling, preprocessing, visualization, and performance evaluation to ensure smooth workflow execution.

Step 2: Behavioral Sensor Dataset Acquisition

Load the behavioral activity dataset collected from smart environment sensors containing activity logs, timestamps, and sensor identifiers.

Step 3: Data Exploration and Behavioral Pattern Analysis

Perform exploratory data analysis to understand behavioral trends, activity frequency distributions, and temporal variations in sensor usage.

Visualize activity states, sensor interaction patterns, and time-based activity graphs to detect potential irregularities or imbalances in normal behavior data.

Step 4: Behavioral Data Preprocessing and Feature Engineering

Handle missing values, noise, and inconsistent sensor readings using interpolation, normalization, and filtering techniques.

Extract meaningful behavioral features, including activity state transitions, time-of-occurrence patterns, day-wise activity trends, and sensor interaction frequency.

Step 5: Anomaly Detection using Isolation Forest Model

Apply the Isolation Forest algorithm to identify anomalies by isolating unusual behavioral data points through random partitioning techniques.

Train the model on preprocessed behavioral features to learn normal activity distributions and compute anomaly scores for each event.

Step 6: One-Class Support Vector Machine (OC-SVM) Modeling

Implement the One-Class SVM algorithm to construct a boundary around normal behavioral activity patterns in high-dimensional feature space.

Train the OC-SVM model using only normal activity data to enable effective detection of deviations and rare abnormal behavioral events.

Step 7: Deep Learning-based Autoencoder Anomaly Detection

Develop an Autoencoder neural network to learn compressed representations of normal behavioral activity patterns.

Train the Autoencoder to reconstruct normal activity sequences and compute reconstruction error for anomaly identification.

Step 8: LSTM-based Sequential Anomaly Detection

Construct a Long Short-Term Memory (LSTM) model to capture temporal dependencies and sequential behavioral patterns in sensor activity data.

Train the LSTM network to predict future activity trends and detect anomalies based on over time.

Step 9: Anomaly Score Aggregation and Decision Framework

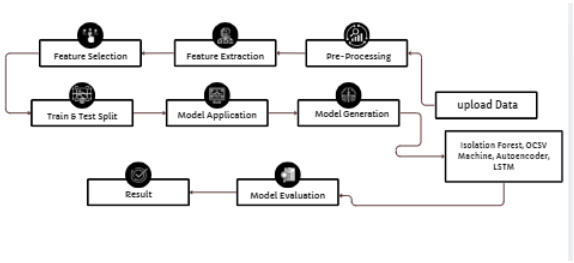
Combine anomaly detection outputs from Isolation Forest, OC-SVM, Autoencoder, and LSTM models to improve detection robustness.

Design a threshold-based or ensemble decision mechanism to classify behavioral events as normal or abnormal.

SYSTEM ARCHITECTURE

System Architecture Overview:

The system architecture for the Machine Learning-Based Behavioral Anomaly Detection in Smart Environments project aims to provide an intelligent and automated framework for identifying abnormal patterns in sensor-based behavioral activity data. The architecture integrates multiple machine learning and deep learning anomaly detection techniques to ensure accurate monitoring, early fault identification, and efficient decision support.



1. Behavioral Data Acquisition and Preprocessing:

Behavioral activity data from smart environment sensors, IoT devices, or pre-existing datasets containing attributes such as sensor identification, activity state, timestamp, and environmental context.

The collected data undergoes validation, cleaning, and preprocessing procedures, including handling missing values, removing inconsistencies, normalization, and feature scaling.

2. Feature Learning and Anomaly Detection Modeling:

The preprocessed behavioral dataset is forwarded to multiple anomaly detection models to learn normal behavioral patterns.

Machine learning techniques such as Isolation Forest and One-Class Support Vector Machine (OC-SVM) are employed to identify deviations from expected activity distributions.

3. Anomaly Score Evaluation and Decision Mechanism:

Anomaly scores generated by different detection models using threshold-based or ensemble decision strategies to improve detection robustness and accuracy.

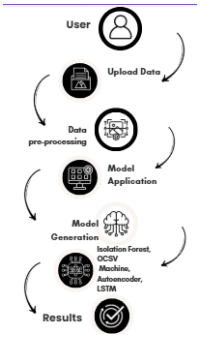
4. Visualization and Monitoring Interface:

The final anomaly detection results are displayed through an interactive monitoring dashboard developed using the Flask web framework.

The interface displays anomaly alerts, behavioral activity trends, and detection insights in a user-friendly graphical format.

Data Flow and Integration

The flow of data through the behavioral anomaly detection system follows a structured and sequential processing pipeline:



1. **Input:**

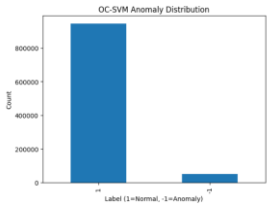
Behavioral activity data generated from smart sensors, IoT devices, or monitoring systems is provided as input. These inputs represent real-world activity sequences such as motion detection events, device usage patterns, or environmental sensor readings.

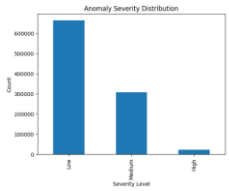
1. **Processing:**

Sequential models such as LSTM further analyze temporal dependencies to detect unusual activity sequences.

3. Output:

The system outputs anomaly classification results indicating whether behavioral events are normal or abnormal.





TESTING AND EVALUATION

Testing Methodology

The testing methodology for the AI-Driven Behavioral Anomaly Detection System follows a structured and systematic approach to ensure the framework satisfies requirements related to detection reliability, system scalability, computational efficiency, and intelligent monitoring capability. The following steps outline the comprehensive testing procedure:

1. Test Case Development:

A detailed set of test cases was designed to represent diverse behavioral activity scenarios in smart environments, including normal user interaction patterns, irregular sensor activations, abnormal temporal activity sequences, and simulated system malfunction events. Test cases reflected real-world operational conditions, routine activity cycles, sporadic behavioral deviations, environmental disturbances, and inconsistent sensor readings.

2. User Testing:

User testing involved system analysts, research students, and general users interacting with the anomaly detection interface using a Flask-based application.

3. Automated Testing:

Automated testing procedures were implemented to assess system performance using bulk behavioral datasets containing historical sensor activity records. These evaluations ensured that anomaly detection models maintained consistent performance under large-scale data processing conditions.

4. Data Collection:

During the testing phase, detailed system logs were maintained to record anomaly detection outcomes, behavioral pattern distributions, and temporal deviation responses.

Evaluation Metrics

Metric	Score
Detection Consistency	High
System Response Time	~0.09s
Model Stability	High
Scalability Performance	High

PERFORMANCE EVALUATION

1. Speed:

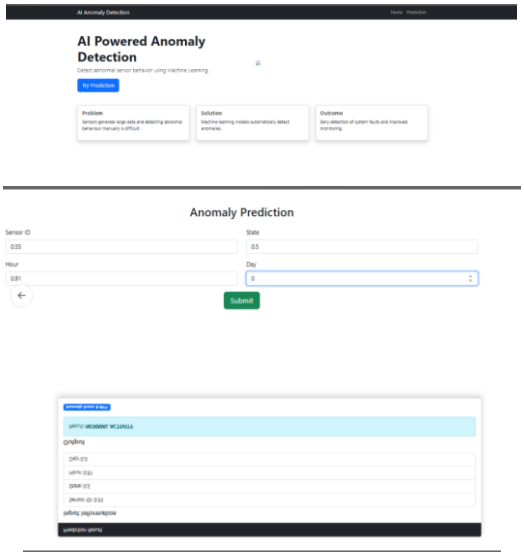
System performance was evaluated by measuring the time required to process behavioral sensor inputs, perform feature extraction, compute anomaly scores, and classify abnormal activity patterns.

2. Detection Effectiveness:

The system's anomaly detection capability was assessed by analyzing the consistency of abnormal event identification across multiple behavioral datasets.

3. Interpretability and Decision Support:

The effectiveness of anomaly detection outputs was evaluated based on the clarity, interpretability, and practical usefulness of system alerts and behavioral trend visualizations.



VII. CONCLUSION

The proposed machine learning–based automated anomaly detection framework effectively addresses the challenges associated with identifying abnormal patterns in behavioral sensor activity data generated from smart environments. By leveraging advanced anomaly detection techniques such as Isolation Forest, One-Class Support Vector Machine (OC-SVM), Autoencoders, and LSTM-based sequence modeling, the system provides a robust and intelligent mechanism for analyzing large-scale and high-dimensional datasets. The feature extraction process, which includes parameters such as sensor identification, activity state, temporal occurrence, and day-wise behavioral patterns, enhances the model's ability to capture contextual dependencies and temporal dynamics within the dataset. This improves the reliability and interpretability of anomaly detection outcomes. The integration of multiple machine learning and

deep learning models ensures improved detection efficiency, scalability, and adaptability in dynamic smart environments.

Future Scope

1. **Real-Time Sensor Data Integration:** Incorporating live data streams from IoT devices to enable continuous anomaly detection in smart homes, industries, and healthcare monitoring systems.
2. **Mobile and Web-Based Monitoring Dashboards:** Developing user-friendly applications to visualize anomaly alerts, activity trends, and system performance insights in real time.
3. **Explainable AI (XAI) Integration:** Providing interpretable explanations for detected anomalies to help users and system administrators understand root causes effectively.
4. **Advanced Deep Learning Architectures:** Implementing Transformer-based models or hybrid CNN-LSTM frameworks to enhance temporal anomaly detection accuracy.
5. **Automated Alert and Notification System:** Enabling SMS, email, or push notifications for instant anomaly alerts and preventive maintenance actions.

References

C. Lian *et al*., "Anomaly Detection and Correction of Optimizing Autonomous Systems With Inverse Reinforcement Learning," in *IEEE Transactions on Cybernetics*, vol. 53, no. 7, pp. 4555-4566, July 2023, doi: 10.1109/TCYB.2022.3213526.

1. K. Choi, J. Yi, C. Park, and S. Yoon, "Deep Learning for Anomaly Detection in Time-Series Data: Review, Analysis, and Guidelines," in *IEEE Access*, vol. 9, pp. 120043-120065, 2021, doi: 10.1109/ACCESS.2021.3107975.
2. Song and D. Li, "Application of a Novel Data-Driven Framework in Anomaly Detection of Industrial Data," in *IEEE Access*, vol. 12, pp. 102798-102812, 2024, doi: 10.1109/ACCESS.2024.3420878.
3. B. Nassif, M. A. Talib, Q. Nasir, and F. M. Dakalbab, "Machine Learning for Anomaly Detection: A Systematic Review," in *IEEE Access*, vol. 9, pp. 78658-78700, 2021, doi: 10.1109/ACCESS.2021.3083060.
4. S. A. Salem, S. A. Said and S. M. Nour, "AI-Driven Anomaly Detection Framework for Improving IoT System Reliability," *2024 IEEE Global Conference on Artificial Intelligence and Internet of Things (GCAIoT)*, Dubai, United Arab Emirates, 2024, pp. 1-8, doi: 10.1109/GCAIoT63427.2024.10833531.