Synthetic Time Series Generator For Anamoly Detection

Ms.Archana[1], Parv Jain[2]
[1]Head of Department, Department of Artificial Intelligence and Data Science
[1]Scholar B.Tech 4th Year, Department of Artificial Intelligence and Data Science
[1, 2]Dr. Akhilesh Das Gupta Institute of Professional Studies, New Delhi

Abstract- *Industries increasingly depend on continuous monitoring systems where fast and accurate anomaly detection is crucial for preventing failures and ensuring operational reliability. This project proposes a lightweight Python-based Synthetic Time Series Generator capable of producing realistic data with trends, seasonality, noise, and multiple anomaly types. The system supports controlled anomaly injection and automatic labeling, making it suitable for training and evaluating anomaly detection models. Experimental results show strong performance, achieving ROC-AUC values between 0.85–0.95 and high precision, recall, and F1-scores.*

Keywords- Synthetic Time Series, Anomaly Detection, Anomaly Injection, Time Series Generator, Machine Learning, Deep Learning, Trend Modeling, Seasonality, Noise Simulation, Python.

Abbreviations

TS – Time Series
AD – Anomaly Detection
ML – Machine Learning
DL – Deep Learning
GAN – Generative Adversarial Network
VAE – Variational Autoencoder
CSV – Comma-Separated Values
QPS – Queries Per Second
AUC – Area Under the Curve
ROC – Receiver Operating Characteristic
LSTM – Long Short-Term Memory
SVM – Support Vector Machine
AE – Autoencoder

I. INTRODUCTION

Time series data forms the backbone of many real-world systems where observations are collected sequentially over time. Detecting anomalies within these sequences is essential for identifying abnormal patterns that may indicate faults, fraud, or unexpected system behavior. However, the scarcity of labeled time series anomaly datasets significantly limits the development and evaluation of robust machine learning models.

This project focuses on developing a Synthetic Time Series Generation System capable of producing realistic, customizable, and reproducible datasets embedded with various anomaly types. The system simulates trends, seasonal patterns, noise, and multiple anomaly categories to mimic real-world temporal behavior. The generated datasets help researchers test anomaly detection algorithms under controlled and diverse conditions, addressing a major gap in the availability of high-quality benchmark datasets.

II. LITERATURE REVIEW

2.1 Synthetic Data Generation in Machine Learning

Prior research highlights the increasing importance of synthetic datasets for training and evaluating ML models, especially when real-world labeled data is scarce or difficult to collect. Studies confirm that synthetic time series can effectively simulate trends, noise, and structural variations, enabling researchers to test anomaly detection algorithms under controlled conditions.

2.2 Time Series Anomaly Detection Techniques

Existing literature covers statistical, machine learning, and deep learning approaches for anomaly detection. Classical algorithms such as ARIMA, Isolation Forest, and One-Class SVM work well for simpler patterns, whereas LSTM Autoencoders and GAN-based architectures handle complex temporal dependencies. These models require diverse datasets—an ongoing limitation in real-world environments.

2.3 Synthetic Anomaly Injection Methods

Researchers frequently introduce point, contextual, and collective anomalies artificially into datasets to simulate realistic abnormal events. These injection methods allow granular control over anomaly shape, duration, intensity, and frequency, helping benchmark detectors under variable difficulty levels.

2.4 Research Gaps Identified

Although several synthetic dataset generators exist, most lack modularity, reproducibility, or detailed control over anomaly types. Very few frameworks systematically integrate trends, seasonality, noise, and anomaly labeling together. Additionally, there is limited empirical benchmarking of how anomaly detection models behave under synthetic yet realistic conditions—this gap is addressed by the present study.

III. OBJECTIVES AND SCOPE

3.1 Objectives

1. Develop a flexible and realistic Synthetic Time Series Generator capable of simulating trends, seasonality, noise, and multiple anomaly types.
2. Implement a controlled anomaly injection module supporting point, contextual, and collective anomalies with configurable parameters.
3. Automatically label anomalies to support supervised ML and DL models.
4. Evaluate anomaly detection algorithms using ROC-AUC, Precision, Recall, and F1-score metrics.
5. Create a reproducible, scalable dataset generation framework suitable for research and industrial experimentation.

3.2 Scope

- The system focuses on univariate time series simulation, including realistic baseline patterns and injected anomalies.
- Multivariate time series, real-time streaming, and domain-specific signal modeling lie outside the current scope but are recommended for future enhancement.
- The study evaluates synthetic data on baseline detection models but does not attempt full system deployment or integration into production environments.

IV. METHODOLOGY

4.1 Baseline Time Series Construction

The baseline dataset is created using additive or multiplicative combinations of:

- Trend (linear, exponential, polynomial)
- Seasonality (single or multi-frequency sinusoidal patterns)
- Noise (Gaussian, uniform, or heavy-tailed distributions)

These components allow generation of realistic long-term patterns and short-term variations.

4.2 Anomaly Injection Module

Three major anomaly types are injected:

- Point anomalies – sudden spikes or drops
- Contextual anomalies – values normal globally but abnormal locally
- Collective anomalies – abnormal sequences representing pattern shifts or sustained deviations

Each anomaly is configurable through magnitude, duration, frequency, and location.

4.3 Preprocessing and Labeling

Anomalies are labeled automatically in a parallel vector.
Data is normalized and structured into windows for ML model training.

4.4 System Development & Tools Used

Python 3.x was used with NumPy, Pandas, Matplotlib, and Scikit-learn.
VS Code served as the IDE due to Git support and debugging tools.

4.5 Performance Evaluation

The generator was tested on:

- Baseline model performance: ROC-AUC, Precision, Recall, F1-score
- Scalability: generation speed under increasing dataset lengths
- Reproducibility: testing random seed-based regeneration

V. RESULTS AND PERFORMANCE ANALYSIS

5.1 Detection Accuracy

Across multiple experiments on synthetic datasets, anomaly detection models achieved:

- ROC-AUC: 0.85 – 0.95
- Precision: 0.75 – 0.90
- Recall: 0.80 – 0.92
- F1-Score: 0.78 – 0.91

Results demonstrate that the synthetic datasets effectively support the benchmarking of ML and DL anomaly detection models.

5.2 Behavioral Analysis of Generated Data

- Trend and seasonality components were clearly identifiable in visual inspections.
- Point anomalies displayed sharp deviations suitable for threshold-based detection.
- Contextual anomalies produced subtle pattern mismatches ideal for advanced models.
- Collective anomalies generated sustained irregular behavior resembling equipment failures or cyber-attacks.
- 5.3 Overall Performance Findings
- Datasets created with higher noise variance produced more realistic challenges.
- LSTM Autoencoders outperformed classical methods on collective anomalies.
- Isolation Forest excelled in detecting sharp spikes.
- The generator successfully produced datasets of varying complexity, aiding research reproducibility.

VI. CHALLENGES AND SOLUTIONS

6.1 Balancing Realism and Control

Fine-tuning trend, noise, and anomaly parameters to mimic realistic conditions is complex.
Solution: modular parameter groups with preset difficulty levels.

6.2 Avoiding Over-Simplified Anomalies

Simple rule-based anomalies may not reflect real-world irregularities.
Solution: introduce hybrid and variable-intensity anomaly shapes.

6.3 Computational Costs for Very Long Series

High-resolution datasets can increase processing time.
Solution: optimized vectorized operations using NumPy.

VII. CASE STUDY: INDUSTRIAL EQUIPMENT MONITORING

A synthetic dataset was created to simulate the temperature readings of industrial machinery operating under varying load conditions. Anomalies were injected to represent overheating, sensor malfunction, and gradual degradation. Machine learning models trained using this dataset identified early degradation patterns, demonstrating the generator’s utility for predictive maintenance use cases.

VIII. ENVIRONMENTAL AND ECONOMIC IMPACT

8.1 Environmental Impact

Because synthetic data eliminates the need for heavy AI training on large datasets, the approach reduces:

- Data collection overhead
- Sensor energy consumption
- Computational carbon footprint

8.2 Economic Impact

Organizations benefit from:

- Lower costs compared to collecting real-world data
- Faster model development cycles
- Ability to test systems under rare, dangerous, or expensive failure conditions
- Scalable dataset creation without specialized hardware

IX. FUTURE DIRECTIONS AND ADVANCEMENTS

- Multivariate Time Series Generation
- Real-Time Streaming Simulation
- GAN-based and VAE-based anomaly patterns
- Domain-specific modules (ECG, finance, IoT)
- Hybrid rule-based + ML anomaly generators

X. CONCLUSION AND FUTURE SCOPE

10.1 Conclusion

The Synthetic Time Series Generator successfully demonstrates a scalable method for producing high-quality datasets needed for anomaly detection research. Through configurable trend, seasonality, noise, and anomaly components, the system provides researchers with full control over dataset complexity. Performance evaluation shows strong detection accuracy across multiple ML models, confirming the generator’s effectiveness as a benchmarking and experimentation tool. The project establishes synthetic data as a cost-effective, reproducible alternative to real-world datasets, addressing the challenges of data scarcity, labeling effort, and privacy restrictions.

10.2 Future Scope

Future versions should support:

1. Multivariate time series with sensor correlations, enabling more realistic industrial simulations.

2. Real-time dataset streaming, allowing evaluation of online anomaly detection models.
3. Integration of AI-generated anomaly patterns, enhancing realism beyond rule-based limits.
4. Domain-specific presets for healthcare, finance, cyber security, and manufacturing.

REFERENCES

[1] K. Zhou, S. Gao, J. Cheng, Z. Gu, H. Fu, Z. Tu, J. Yang, Y. Zhao, J. Liu, Sparse-gan: Sparsity-constrained generative adversarial network for anomaly detection in retinal OCT image, CoRR abs/1911.12527 (2019). arXiv:1911.12527.

[2] C. Donahue, J. J. McAuley, M. S. Puckette, Synthesizing audio with generative adversarial networks, CoRR abs/1802.04208 (2018). arXiv: 1802.04208.

[3] P. Dhariwal, H. Jun, C. Payne, J. W. Kim, A. Radford, I. Sutskever, Jukebox: A generative model for music, CoRR abs/2005.00341 (2020). arXiv:2005.00341 .

[4] A. Radford, L. Metz, S. Chintala, Unsupervised representation learning with deep convolutional generative adversarial networks (2015).

[5] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, K. Kavukcuoglu, Wavenet: A generative model for raw audio, CoRR abs/1609.03499 (2016). arXiv:1609.03499.

[6] P. Isola, J. Zhu, T. Zhou, A. A. Efros, Image-to-image translation with conditional adversarial networks, CoRR abs/1611.07004 (2016). arXiv: 1611.07004.

[7] J. T. Zhou, K. Di, J. Du, X. Peng, H. Yang, S. J. Pan, I. W. Tsang, Y. Liu, Z. Qin, R. S. M. Goh, Sc2net: Sparse lstms for sparse coding, in: Proceedings of the 32th AAAI Conference on Artificial Intelligence, AAAI, New Orleans, Louisiana, 2018, pp. 4588–4595.