OSTBAYERISCHE TECHNISCHE HOCHSCHULE REGENSBURG

A Transformer-Based Framework for Anomaly Detection in Multivariate Time Series

Fabian Folger[1], Murad Hachani[1], Philipp Fuxen[1], Julian Graf[1], Sebastian Fischer[1] , Rudolf Hackenberg[1]

[1]**Department of Computer Science and Mathematics**
Ostbayerische Technische Hochschule Regensburg, Germany

e-mail: fabian1.folger@st.oth-regensburg.de
e-mail: {murad.hachani | philipp.fuxen | julian.graf |sebastian.fischer | rudolf.hackenberg}@oth-regensburg.de

IARIA

Folger Fabian

Fabian Folger received his master's degree with a specialization in Embedded and Automotive Security in March 2025. He is currently working as a Software Developer at Krones AG. His research interests lie in the intersection of embedded systems and cybersecurity, with a particular focus on secure communication protocols, AI and Reverse Engineering

Aim of the paper

Developing a framework with a comprehensive Transformer-based architecture [20] for anomaly detection in multivariate timeseries [2] .

Key points for the Transformer:

- Works with high-dimensional sensor data without extensive feature engineering.
- Detects anomalies in long-term dependencies due to self-attention [20] [8].
- Enables early detection of unusual patterns to prevent critical system failures.
- In a subsequent laboratory setup, the framework will be applied using fuzzing techniques [4] to induce anomalies in an Electronic Control Unit, while monitoring side channels, such as temperature, voltage, and Controller Area Network messages [5].

CATS-Dataset

• **Realistic Dataset Selection:**
The dataset was chosen to enable an approach that is as realistic as possible and applicable in a laboratory environment.

• **CATS Dataset by Solenix Engineering GmbH:**
This dataset is ideal because it encompasses various control, environmental, and sensor data, and it includes specifically controlled, labeled anomalies [19].

• **Laboratory Implementation:**
In the lab, a fuzzer is used to generate synthetic disturbances and errors [4].

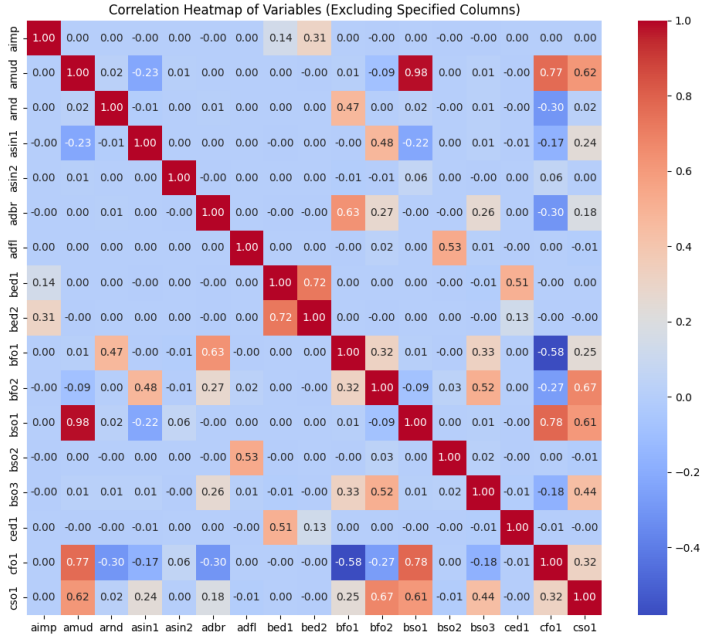
• **Precise Modeling with Rich Data:**
With 17 different variables and a total of five million timestamps, both the normal behavior of the system and the occurrence of rare deviations can be modeled accurately [19].

Korrelation - Heatmap

Variable Correlations:
The dataset displays correlations among variables, with some showing relatively strong correlations.

Anomaly Root Cause:
It is important to note that the root cause of the anomalies originates in other variables.

Preserving Dependencies:
The dataset should not be truncated, as doing so would lead to the loss of these critical dependencies.



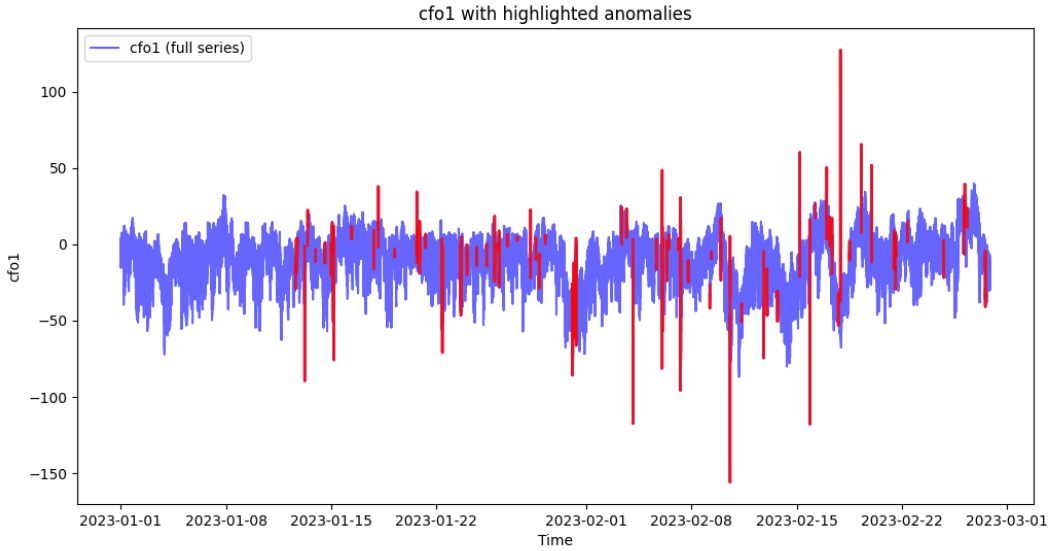
Injected anomalies in variables

Example showing injected anomalies in the variable CFO1, with anomalies highlighted in red.

Pre-Processing Data:
The plot was generated before any data cleaning or scaling.

Data Visualization Details:
- Y-axis: Raw measurement values
- X-axis: Temporal progression (dates)

Data Collection Context:
Data were collected over several days.



Additions to the framework

Scaling with MinMaxScaler:
All values are scaled to the range [0, 1] using a MinMaxScaler this reduces numerical issues.

Sliding Window Segmentation:
Time series data is segmented into overlapping windows which serve as the input to the Transformer

Consistency Check:
A consistency check verifies that the number of detected anomalies matches the expected count (e.g., 200); if not, the script halts to ensure data integrity.

RAM Logger:
A RAM Logger has been implemented with a GUI using tkinter.ttk to monitor memory usage.

Data Preparation for Optimization and Training:
Tensors are labeled with .zeros or .ones this is crucial because, for F1 optimization in Optuna [24].

Transformer Architecture

Transformer Encoder:

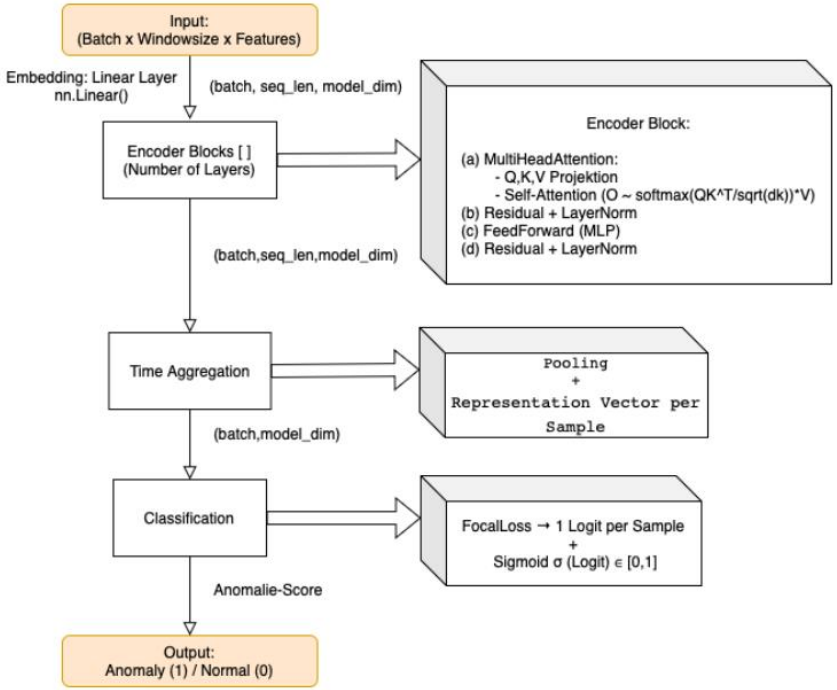
- The encoder is structured in blocks to leverage Multi-Head Attention [20].
- These blocks capture complex temporal relationships within the data [9].

Temporal Aggregation:

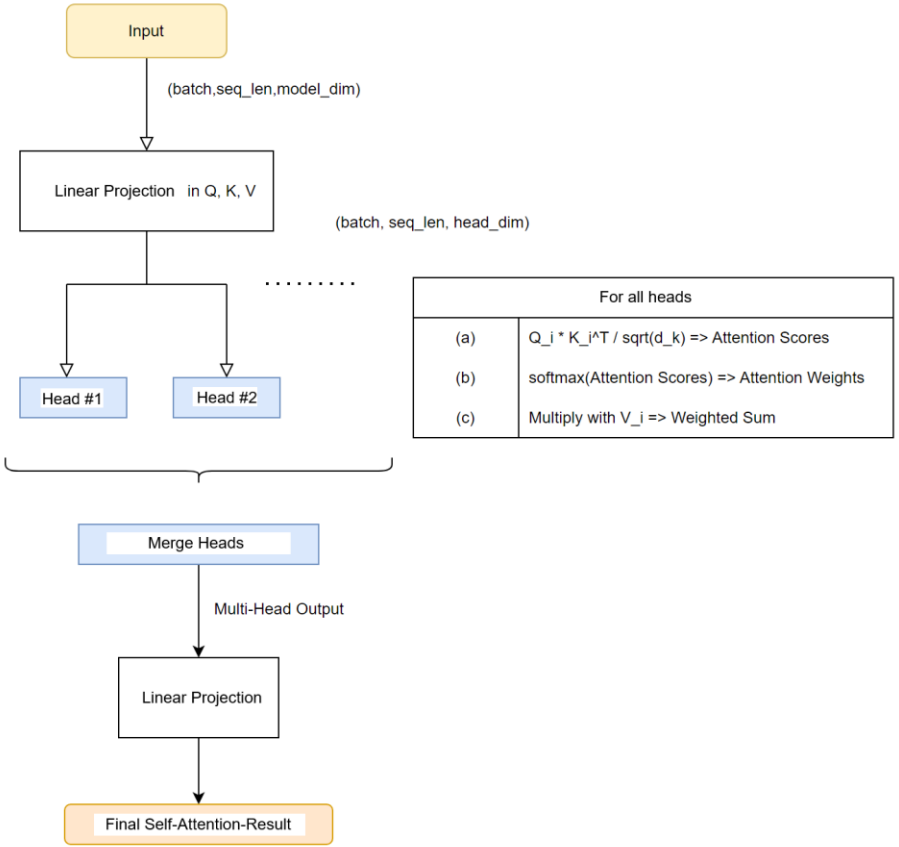
- Pooling is performed to aggregate the time dimension.
- A single representation vector is generated for each sample.

Classification Module:

- Focal Loss Scheduling sets a binary logit per sample (more details to follow) [23] .
- A Sigmoid function, combined with weights and a threshold, produces the final classification.



Self-Attention [20]



Input Layer and Batch Normalization

Time Series Segments:
The model processes time series segments containing raw sensor data (17 features) [19].

Linear Embedding Transformation:
The input is transformed using the formula:

$$y = xA^T + b$$

where x is the input, represents the weight matrix (initialized uniformly with

$$k = \frac{1}{in\_features}$$ and b the bias.

Purpose of the Transformation:
This step is essential for converting raw sensor data into a representation that is optimal for the self-attention mechanism.

Batch Normalization:
Batch normalization is applied to each sliding window to ensure a stable feature distribution, enhancing the model's robustness during training.

Details of the Self-Attention

Each time step in the input sequence is first projected into three matrices: Q (Query), K (Key), and V (Value).

The dot product QK^T is computed and then scaled by $\frac{1}{\sqrt{d_k}}$ (with d_k equal to model dimension) to prevent overly large values [20].

The scaled result is passed through a softmax function, converting it into probabilities that indicate how much attention each element pays to all others.

This process produces an Attention Map where each row represents the importance of other time steps for the current one (Attention Scores) [20].

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right) V$$ [20]

Multiple attention heads operate in parallel, with each head learning different aspects of the data.

The outputs from these heads are then concatenated, combining different perspectives to enable effective anomaly detection.

Classification Pipeline

Transformer Encoder Output:
After the sequence passes through the Transformer encoder layers, each time step x_t produces an output vector in $\mathbb{R}^{d_{model}}$

Temporal Aggregation:
A mean pooling operation condenses these time-dependent representations into a single vector z per segment [21][22]:

$$\mathbf{z} = \frac{1}{T}\sum_{t=1}^{T}\mathbf{x}_t,$$

Classification Head & Focal Loss for Anomaly Detection

Classification Head:

•The aggregated vector z is fed into a linear layer to produce a scalar output [21].

•A sigmoid activation converts this scalar into a probability between 0 (normal) and 1 (anomalous.

$$o = \text{Linear}(\mathbf{z}) \in \mathbb{R}.$$

$$\sigma(\text{Logit}) = \frac{1}{1+\exp(-\text{Logit})}.$$

Addressing Class Imbalance:

To better handle the extreme imbalance between normal and anomalous samples, Focal Loss is employed instead of standard Binary Cross Entropy (BCE).

$$\text{BCE}(\hat{y}, y) = -\Big[y \log(\hat{y}) + (1-y) \log(1-\hat{y})\Big],$$

$$\text{FL}(\hat{y}, y) = \alpha \Big(1-\hat{y}\Big)^{\gamma} \text{BCE}(\hat{y}, y),$$

$$\alpha = 0.5, \quad \gamma = 2.5.$$

Key Benefit:

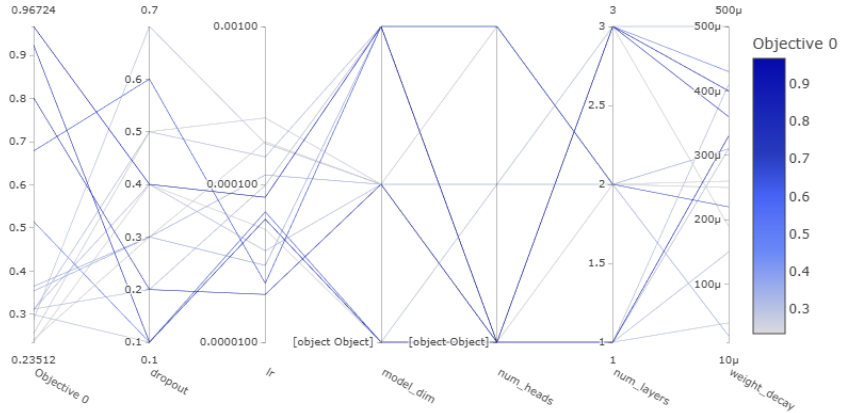
Focal Loss down-weights the loss contribution of well-classified (majority) samples and emphasizes hard-to-classify (minority) anomalies, thereby enhancing detection performance [23].

Hyperparameter Tuning with Optuna [24]

Semi-Supervised Approach:
• Training is conducted primarily on normal data.
• The validation set includes a few anomalies, enabling the tuning process to optimize for metrics like the F1 score.

Unsupervised Retraining:
• Once optimal hyperparameters are determined, the model is retrained solely on normal data (unsupervised), without explicit anomaly examples.

Final Evaluation:
• The model is evaluated on a test set containing both real and synthetic anomalies to assess its true detection capability.



Model Performance & Hyperparameter Tuning

Summary:
- The high ROC-AUC and F1-Score indicate excellent discrimination between normal and anomalous instances.
- Balanced precision and recall demonstrate effective anomaly detection while minimizing false positives.
- The confusion matrix confirms that 98.52% of true anomaly labels are correctly identified.

Optimized Hyperparameters (via Optuna):
- **Dropout:** 0.2
- **Learning Rate:** 2.0075e-05
- **Model Dimension:** 128
- **Attention Heads:** 8
- **Encoder Layers:** 3
- **Weight Decay:** 0.00036
- **Batch Size:** 128

Transformer Metrics	Values
Optimal threshold	0.0117
ROC-AUC	0.9993
F1-Score	0.9717
Precision	0.9585
Recall	0.9853
Accuracy	0.9921
Anomalies detected (all Labels)	11731/83558 (14.04%)
Anomalies detected (Anomaly-Labels)	11244/11413 (98.52%)

Comparative Analysis

Preliminary Comparison:

• Our Transformer-based model is compared to established methods from *Anomaly Detection in Time Series: A Comprehensive Evaluation*.

• Evaluation is currently limited to the CATS dataset, which closely resembles our laboratory setup.

Benchmarking Metrics:

Acknowledges limitations due to different datasets across studies.

The CATS dataset's integration on Timeeval GitHub paves the way for future evaluations [25] [27].

Custom Framework Features:

1. **Focal Loss with Self-Attention:**
   Emphasizes misclassified anomaly cases to address class imbalance.
2. **Flexible Time Window Segmentation:**
   Optimizes sequence length and attention heads to capture diverse temporal characteristics.
3. **Tailored Binary Anomaly Classification:**
   Focuses on detecting rare anomalies with a specialized loss function.

Discussion Points

Generalizability:
Uncertainty remains on transferring results to more complex, real-world scenarios. Further experiments in domains like industrial processes or medical applications are needed.

Sensitivity to Preprocessing:
Parameters such as sliding window size, handling of missing data, and hyperparameter tuning (e.g., via Optuna) significantly impact performance.

Comparison with Traditional Methods:
Transformer encoder-only configuration offers flexibility and superior performance for complex, high-dimensional data, whereas methods like autoencoders or Isolation Forest may struggle with nonlinear dependencies.

Conclusion & Future Work

Transformer-Based Anomaly Detection:
- Leverages self-attention to capture complex, high-dimensional relationships in multivariate time series.
- Minimal feature engineering required thanks to a flexible data pipeline.

Key Advantages:
- Reliable detection of anomalies even in noisy environments or under changing data distributions.
- Particularly relevant for industrial, safety-critical, automotive, and IoT security applications, with potential in fields like medicine.

Future Directions & Challenges:
- Further validation on diverse, real-world datasets to assess generalizability.
- Standardized benchmarking using integrations like the CATS dataset on Timeeval.
- Sensitivity analyses on preprocessing parameters (e.g., window size, missing-data strategies) to refine model robustness.
- Continued development in Laboratory environments, including enhancing anomaly induction techniques (e.g., ECU and Fuzzing-Engine integration on CAN bus).

Overall Impact:
- Provides a strong foundation for future research and practical deployments in anomaly detection.

References

[1] NVIDIA, “Dlss 4: Multi-frame generation and ai innovations”, Accessed: 2025.03.19, NVIDIA Corporation, 2024, [Online]. Available: https://www.nvidia.com/en-us/geforce/news/dlss4-multi-frame-generation-ai-innovations/.
[2] J. Xu, H. Wu, J. Wang, and M. Long, “Anomaly transformer: Time series anomaly detection with association discrepancy”, CoRR, vol. abs/2110.02642, 2021. arXiv: 2110.02642.
[3] M. Böhme, C. Cadar, and A. Roychoudhury, “Fuzzing: Challenges and reflections”, IEEE Software, vol. 38, no. 3, pp. 79–86, 2021.
[4] L. McDonald, M. I. U. Haq, and A. Barkworth, “Survey of software fuzzing techniques”, Dec. 2021.
[5] P. Sperl and K. Böttinger, “Side-channel aware fuzzing”, in Computer Security–ESORICS 2019: European Symposium on Research in Computer Security, Springer, vol. 24, Sep. 2019, pp. 259–278, ISBN: 978-3-030-29958-3. DOI: 10.1007/978-3-030-29959-0_13.
[6] M. Muench, J. Stijohann, F. Kargl, A. Francillon, and D. Balzarotti, “What you corrupt is not what you crash: Challenges in fuzzing embedded devices”, NDSS Symposium 2018, 2018. DOI: 10.14722/ndss.2018.23176.
[7] Q. Wen et al., Transformers in time series: A survey, 2023. arXiv: 2202.07125.

References

[8] S. Li et al., “Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting”, Advances in neural information processing systems, vol. 32, 2019.
[9] T. Zhou et al., “Fedformer: Frequency enhanced decomposed transformer for long-term series forecasting”, in International conference on machine learning, PMLR, 2022, pp. 27 268–27 286.
[10] A. Zeng, M. Chen, L. Zhang, and Q. Xu, “Are transformers effective for time series forecasting?”, in Proceedings of the AAAI conference on artificial intelligence, vol. 37, 2023, pp. 11 121–11 128.
[11] Y. Nie, N. H. Nguyen, P. Sinthong, and J. Kalagnanam, A time series is worth 64 words: Long-term forecasting with transformers, 2023. arXiv: 2211.14730 [cs.LG].
[12] Z. Wu, Z. Liu, J. Lin, Y. Lin, and S. Han, “Lite transformer with long-short range attention”, CoRR, vol. abs/2004.11886, 2020. arXiv: 2004.11886.
[13] S. Mehta, M. Ghazvininejad, S. Iyer, L. Zettlemoyer, and H. Hajishirzi, “Delight: Very deep and light-weight transformer”, CoRR, vol. abs/2008.00623, 2020. arXiv: 2008.00623.
[14] A. Bapna, M. X. Chen, O. Firat, Y. Cao, and Y. Wu, “Training deeper neural machine translation models with transparent attention”, CoRR, vol. abs/1808.07561, 2018. arXiv: 1808 . 07561.

References

[15] M. Dehghani, S. Gouws, O. Vinyals, J. Uszkoreit, and L. Kaiser, “Universal transformers”, CoRR, vol. abs/1807.03819, 2018. arXiv: 1807.03819.
[16] G. Zerveas, S. Jayaraman, D. Patel, A. Bhamidipaty, and C. Eickhoff, “A transformer-based framework for multivariate time series representation learning”, in Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining, 2021, pp. 2114–2124.
[17] J. Graf, K. Neubauer, S. Fischer, and R. Hackenberg, “Architecture of an intelligent intrusion detection system for smart home”, in 2020 IEEE international conference on pervasive computing and communications workshops (PerCom Workshops), IEEE, 2020, pp. 1–6.
[18] P. Fuxen, M. Hachani, J. Schmidt, P. Zaumseil, and R. Hackenberg, “Side channel monitoring for fuzz testing of future mobility systems”, CLOUD COMPUTING 2023, 2023.
[19] Patrick Fleith, Controlled anomalies time series (CATS) dataset, version 2, Sep. 14, 2023. DOI: doi.org/10.5281/zenodo.7646896.
[20] A. Vaswani et al., “Attention is all you need”, CoRR, vol. abs/1706.03762, 2017. arXiv: 1706.03762.
[21] S. Song, N. -M. Cheung, V. Chandrasekhar, and B. Mandal, “Deep adaptive temporal pooling for activity recognition”, in Proceedings of the 26th ACM international conference on Multimedia, Oct. 15, 2018, pp. 1829–1837. DOI: 10 . 1145 / 3240508.3240713. arXiv: 1808.07272[cs].

References

[22] G. Zerveas, S. Jayaraman, D. Patel, A. Bhamidipaty, and C. Eickhoff, A transformer-based framework for multivariate time series representation learning, Dec. 8, 2020. DOI: 10.48550/arXiv.2010.02803. arXiv: 2010.02803[cs].
[23] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, Focal loss for dense object detection, Feb. 7, 2018. DOI: 10.48550/arXiv.1708.02002. arXiv: 1708.02002[cs].
[24] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework", in Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, ser. KDD '19, New York, NY, USA: Association for Computing Machinery, Jul. 25, 2019, pp. 2623–2631, ISBN: 978-1-4503-6201-6. DOI: 10.1145/3292500.3330701.
[25] P. Wenig, S. Schmidl, and T. Papenbrock, "TimeEval: A benchmarking toolkit for time series anomaly detection algorithms", Proc. VLDB Endow., vol. 15, no. 12, pp. 3678–3681, Aug. 1, 2022, ISSN: 2150-8097. DOI: 10.14778/3554821.3554873.
[26] S. Schmidl, P. Wenig, and T. Papenbrock, "Anomaly detection in time series: A comprehensive evaluation", Proceedings of the VLDB Endowment, vol. 15, no. 9, pp. 1779–1797, May 2022, ISSN: 2150-8097. DOI: 10.14778/3538598.3538602.
[27] T. Project, "Timeeval - datasets overview", Accessed: 2025.03.19, 2022, [Online]. Available: https://timeeval.github.io/evaluation-paper/notebooks/Datasets.html.