The Journal of Open Source Software

Sentinel: Signal Validation and Anomaly Detection for Enterprise Log Data

José Manuel Vergara Álvarez [1], **Nicolás Laverde Manotas** [1], **Juan Pablo Aguilar Calle** [1], **Jeisson Vicente Niño Castillo** [1], **Julián David Muñoz Pertuz** [1], **Daniel Monsalve Muñoz** [1], **and Sebastián Osorio Agudelo** [1]

1 Bancolombia, Medellín, Colombia

DOI: 10.21105/joss.11275

Software

- Review
- Repository
- Archive

Editor: Neea Rusch

Reviewers:

- @kishanraj41
- @khushboo2243

Submitted: 27 March 2026
Published: 07 November 2026

License
Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License (CC BY 4.0).

Summary

Modern enterprise systems generate large volumes of operational logs that contain information about system behavior, performance, and failure conditions. These logs are frequently used for troubleshooting, monitoring, and anomaly detection. However, raw logs are typically unstructured and heterogeneous, making it difficult to determine whether they contain meaningful signals for analytical methods. Applying anomaly detection algorithms without first validating the quality and informational content of the data can lead to misleading results or unnecessary computational costs.

Sentinel is a Python library designed to address this challenge by introducing **early signal validation for enterprise logs prior to anomaly detection**. The library provides a modular pipeline that transforms unstructured logs into structured data, evaluates the presence of meaningful statistical signals, and applies anomaly detection algorithms only when the data meets predefined quality criteria. This fail-fast validation approach helps practitioners determine whether log datasets are suitable for anomaly analysis before investing effort in complex modeling pipelines.

The library supports common enterprise log formats and provides tools for ingestion, transformation, signal validation, anomaly detection, visualization, and streaming simulation. Sentinel is implemented in Python and builds upon widely used scientific computing libraries including pandas (McKinney, 2010) and scikit-learn (Pedregosa et al., 2011), with optional deep learning detectors implemented using PyTorch (Paszke et al., 2019).

Statement of Need

Enterprise infrastructures such as application servers, message brokers, security modules, and networking systems continuously produce operational logs. These logs often contain indicators of system failures, abnormal activity, or performance degradation. Detecting anomalies in log data is therefore an important task in operational monitoring, cybersecurity, and reliability engineering.

Despite the extensive literature on anomaly detection methods (Chandola et al., 2009), a practical challenge often arises in real-world environments: **not all log data contains analyzable signals**. Logs may contain excessive missing values, constant fields, inconsistent timestamps, or insufficient variability to support meaningful detection algorithms. In such situations, applying machine learning models can produce unstable or misleading outputs.

Most anomaly detection libraries focus on the modeling stage and assume that the input data already contains useful patterns. Libraries such as PyOD (Zhao et al., 2019), ADTK (Arundo Analytics, 2019), pySAD (Yilmaz & Kozat, 2021), TODS (Lai et al., 2021), and

Anomalib (Akcay et al., 2022) provide extensive implementations of detection algorithms, but they generally rely on pre-processed, signal-rich datasets.

Sentinel addresses the earlier stage of the workflow: **determining whether a log dataset is suitable for anomaly detection in the first place**. The library evaluates signal quality metrics before applying detectors, enabling practitioners to avoid unnecessary modeling steps when the underlying data lacks informative structure. This approach is particularly valuable for enterprise logs, where large datasets may still contain limited analytical value.

State of the Field

The ecosystem of anomaly detection software has expanded significantly in recent years. PyOD (Zhao et al., 2019) provides a unified interface for a wide range of outlier detection algorithms. ADTK (Arundo Analytics, 2019) focuses on time-series anomaly detection using rule-based and statistical techniques. pySAD (Yilmaz & Kozat, 2021) addresses streaming anomaly detection scenarios, while TODS (Lai et al., 2021) introduces automated machine learning pipelines for time-series outlier detection. Anomalib (Akcay et al., 2022) provides deep learning architectures primarily targeting computer vision applications.

In the context of log analysis, research has also explored specialized anomaly detection techniques such as DeepLog, which applies recurrent neural networks to learn sequential patterns in system logs (Du et al., 2017). Public datasets such as LogHub provide standardized benchmarks for evaluating log anomaly detection approaches (Zhu et al., 2023).

These tools and datasets have significantly advanced anomaly detection research. However, they typically assume that the dataset has already undergone preprocessing and validation. In many operational environments, the primary difficulty lies in determining whether the available logs contain sufficient information to justify anomaly detection.

Sentinel complements existing tools by introducing a **signal validation stage before detection**. Rather than replacing existing anomaly detection libraries, Sentinel can be used alongside them to assess data suitability and prepare structured log datasets for downstream analysis.

Software Design

Sentinel implements a modular architecture designed to support the complete lifecycle of enterprise log analysis. The system is organized into six core modules that can be used independently or combined into a unified pipeline.

The Journal of Open Source Software

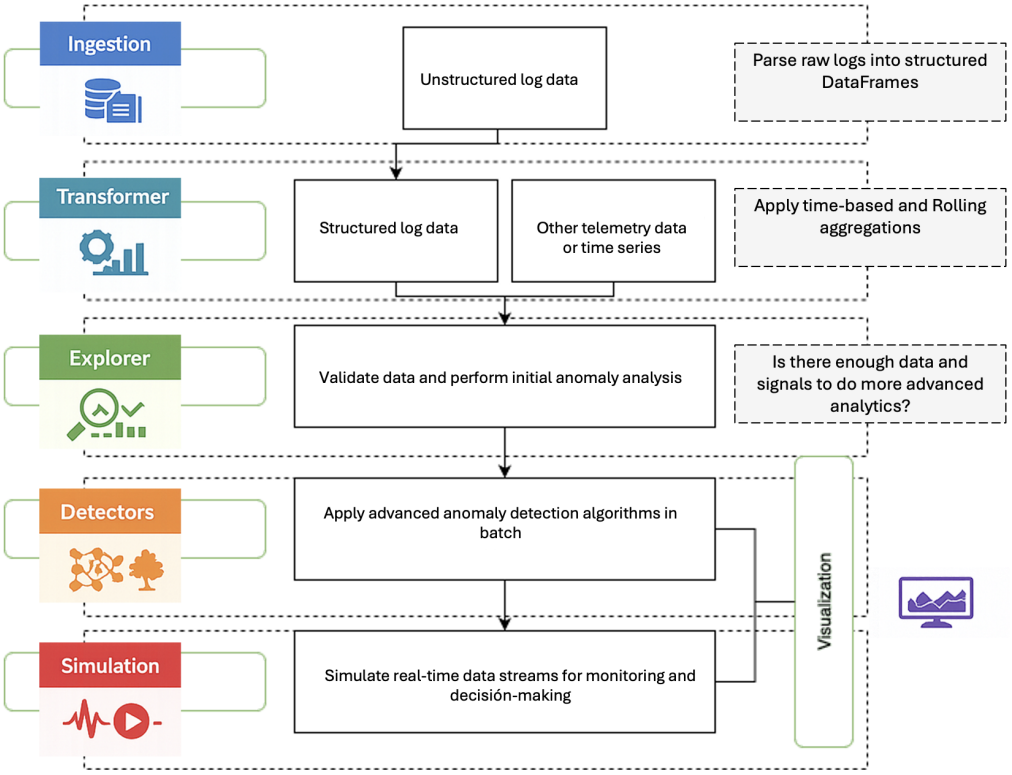


Figure 1: Sentinel modular architecture for log analysis and anomaly detection.

Module	Purpose	Techniques and Components
Ingestion	Convert raw log files into structured tabular data	Parsers for WAS, HSM, HDC, IBMMQ and ZTNA logs; extensible parser interface
Transformer	Prepare logs for analysis	Time aggregation, rolling statistics, feature generation
Explorer	Validate signal presence before anomaly detection	IQR anomaly checks, variance thresholds, label validation, correlation analysis
Detectors	Apply anomaly detection algorithms	Isolation Forest (Liu et al., 2008), Robust Random Cut Forest (Guha et al., 2016), LSTM Autoencoder, Liquid Neural Networks (Hasani et al., 2022)
Visualization	Interpret anomaly detection outputs	Time-series anomaly visualization and SHAP-based explainability (Lundberg & Lee, 2017)
Simulation	Test streaming anomaly detection workflows	Real-time chunk processing and adaptive thresholds

The **Ingestion module** converts unstructured log files into structured tabular representations using pandas. Sentinel includes built-in parsers for enterprise systems such as WebSphere

Application Server (WAS), Hardware Security Modules (HSM), High-Density Computing (HDC), IBM Message Queue (IBMMQ), and ZTNA logs. A base parser interface allows users to implement custom parsers for additional log formats.

The **Transformer module** prepares structured logs for analysis by aggregating events into time windows and computing rolling statistics. These transformations allow event-based logs to be converted into feature matrices suitable for statistical or machine learning methods.

The **Explorer module** is the central innovation of the library. It evaluates signal quality using statistical metrics including interquartile range (IQR) anomaly checks, variance thresholds, label presence validation, and correlation analysis. The module also evaluates simple predictive models on individual features to estimate signal relevance. Based on these metrics, the module provides a fail-fast decision indicating whether further analysis is justified.

The **Detectors module** provides a unified interface for multiple anomaly detection algorithms. Implementations include Isolation Forest (Liu et al., 2008), Robust Random Cut Forest (Guha et al., 2016), an LSTM-based autoencoder implemented with PyTorch (Paszke et al., 2019), and Liquid Neural Networks (Hasani et al., 2022). The unified interface enables practitioners to compare different algorithms within the same workflow.

The **Visualization module** provides tools for analyzing anomaly detection results. Anomaly scores can be visualized as time-series plots with anomaly markers, and SHAP explanations can be used to interpret feature contributions to anomaly predictions.

The **Simulation module** enables testing of anomaly detection pipelines in streaming scenarios. Data can be processed in configurable chunks to emulate real-time ingestion pipelines, allowing practitioners to evaluate detection strategies under operational conditions.

Research Impact Statement

Sentinel has demonstrated research and operational impact through its adoption in production and research activities within the organization. The software has been used to assess the analytical potential of operational data sources for service availability and anomaly detection studies, helping teams identify log datasets that contain sufficient statistical signal before investing effort in advanced analytics.

To date, Sentinel has been applied to seven operational data sources, including Cortex XDR logs from three endpoint environments, as well as logs from HSM, HDC, WebSphere Application Server (WAS), and IBM MQ platforms. These evaluations have supported the selection of data sources suitable for observability and anomaly detection research.

The software has been operational in production for approximately one year and currently supports analytical activities associated with five services. During this period, Sentinel has enabled the validation of log quality and analytical readiness in real-world environments, providing practical evidence of the applicability of its signal-validation approach.

Sentinel has been utilized by three different teams across research and operational contexts, supporting investigations related to observability, service availability, and anomaly detection. By standardizing preprocessing, signal validation, and detector integration workflows, the software has facilitated reproducible experimentation while simultaneously supporting production monitoring activities.

The combination of sustained production use, adoption across multiple teams, and application to diverse enterprise log sources demonstrates Sentinel's value as both a research-enabling platform and a practical operational analytics tool.

AI Usage Disclosure

Generative AI tools were used to assist with code scaffolding, documentation drafting, and language refinement during the development of this software and manuscript. All generated content was reviewed, validated, and modified as necessary by the authors, who assume full responsibility for the accuracy and integrity of the work.

Acknowledgements

The authors thank the Technology Innovation team (ARQUITECTURA INNOVACION TI) at Bancolombia for supporting the development of Sentinel and for providing feedback during the design of the library.

References

Akcay, S., Ameln, D., Vaidya, A., Lakshmanan, B., Ahuja, N., & Genc, U. (2022). Anomalib: A deep learning library for anomaly detection. *2022 IEEE International Conference on Image Processing (ICIP)*, 1706–1710. https://doi.org/10.1109/ICIP46576.2022.9897283

Arundo Analytics. (2019). *ADTK: Anomaly detection toolkit* (Version 0.6.2). GitHub. https://github.com/arundo/adtk

Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys*, *41*(3), 1–58. https://doi.org/10.1145/1541880.1541882

Du, M., Li, F., Zheng, G., & Srikumar, V. (2017). DeepLog: Anomaly detection and diagnosis from system logs through deep learning. *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, 1285–1298. https://doi.org/10.1145/3133956.3134015

Guha, S., Mishra, N., Roy, G., & Schrijvers, O. (2016). Robust random cut forest based anomaly detection on streams. *Proceedings of the 33rd International Conference on Machine Learning (ICML 2016)*, 2712–2721. https://proceedings.mlr.press/v48/guha16.html

Hasani, R., Lechner, M., Amini, A., Liebenwein, L., Ray, A., Tschaikowski, M., Teschl, G., & Rus, D. (2022). Closed-form continuous-time neural networks. *Nature Machine Intelligence*, *4*, 992–1003. https://doi.org/10.1038/s42256-022-00556-7

Lai, K.-H., Zha, D., Wang, G., Xu, J., Zhao, Y., Kumar, D., Chen, Y., Zumkhawaka, P., Wan, M., Martinez, D., & Hu, X. (2021). TODS: An automated time series outlier detection system. *Proceedings of the AAAI Conference on Artificial Intelligence*, *35*, 16060–16062. https://doi.org/10.1609/aaai.v35i18.18012

Liu, F. T., Ting, K. M., & Zhou, Z.-H. (2008). Isolation forest. *2008 Eighth IEEE International Conference on Data Mining*, 413–422. https://doi.org/10.1109/ICDM.2008.17

Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems*, *30*.

McKinney, W. (2010). Data structures for statistical computing in Python. *Proceedings of the 9th Python in Science Conference (SciPy 2010)*, 56–61. https://doi.org/10.25080/Majora-92bf1922-00a

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., … Chintala, S. (2019). PyTorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems 32 (NeurIPS 2019)*, 8024–8035. https://proceedings.neurips.cc/paper/

2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830. http://jmlr.org/papers/v12/pedregosa11a.html

Yilmaz, S. F., & Kozat, S. S. (2021). *pySAD: A streaming anomaly detection framework in Python* (Version 0.7.0). GitHub. https://github.com/selimfirat/pysad

Zhao, Y., Nasrullah, Z., & Li, Z. (2019). PyOD: A Python toolbox for scalable outlier detection. *Journal of Machine Learning Research*, *20*(96), 1–7. http://jmlr.org/papers/v20/19-011.html

Zhu, J., He, S., He, P., Liu, J., & Lyu, M. R. (2023). Loghub: A large collection of system log datasets for AI-driven log analytics. *2023 IEEE 34th International Symposium on Software Reliability Engineering (ISSRE)*, 355–366. https://doi.org/10.1109/ISSRE59848.2023.00071