LLM-ASSISTED LOGIC RULE LEARNING: SCALING HUMAN EXPERTISE FOR TIME SERIES ANOMALY DETECTION

Haoting Zhang & Shekhar Jain
Supply Chain Optimization Technologies, Amazon.com, Inc.
Bellevue, WA 98004, USA
{htizhang,shekhajt}@amazon.com

ABSTRACT

Time series anomaly detection is critical for supply chain management to take proactive operations, but faces challenges: classical unsupervised anomaly detection based on exploiting data patterns often yields results misaligned with business requirements and domain knowledge, while manual expert analysis cannot scale. We propose a framework that leverages large language models (LLMs) to systematically encode human expertise into interpretable, logic-based rules for detecting anomalies in supply chain time series data. Distinct from direct LLM deployment, our approach utilizes LLMs to generate and iteratively optimize deterministic rules offline. Deployed at Amazon for monitoring millions of individual products, our approach achieves detection accuracy comparable to state-of-the-art multimodal LLMs while delivering 1) orders of magnitude faster execution with 2) zero inference API costs in production. This establishes a scalable paradigm for bridging expert-driven decision-making with automated production systems, eliminating the latency and stochasticity barriers of direct LLM deployment.

Track: Industry & Applications

1 INTRODUCTION

Time series anomaly detection is critical for different applications, including cloud infrastructure health, manufacturing quality control, and supply chain management (Ansari et al., 2024; Das et al., 2024a; Goswami et al., 2024; Williams et al., 2024; Huang et al., 2025; Cai et al., 2025). At Amazon, continuous monitoring of weekly product-level metrics is essential for maintaining optimal inventory levels. However, classical unsupervised methods (Munir et al., 2018; Tedjopurnomo et al., 2020) struggle with the extreme volatility of individual item data. Lacking business context, these pattern-based methods often misclassify beneficial deviations (e.g., sharp drops in out-of-stock ratios) as anomalies, while their black-box nature hinders validation. On the other hand, human domain experts effectively identify anomalies, yet manual analysis cannot scale in production.

While recent work utilizes Large Language Models (LLMs) to replicate this reasoning (Jin et al., 2023; Zhang et al., 2024; Lan et al., 2025), direct LLM deployment suffers from high computational latency, prohibitive costs, and non-deterministic outputs that undermine reliability in production. To bridge this gap, we propose an *LLM-assisted logic rule learning framework* (Fig. 1) that treats LLMs not as real-time detectors, but as offline "compilers" to distill implicit domain expertise into explicit, deterministic symbolic rules, thereby exhibiting high interpretability and execution efficiency.

2 METHODOLOGY & RESULTS

1. Labeling Stage: Human-AI Collaboration. To encode domain expertise, we prompt LLMs with *visualized* time series plots to label anomalies (Liu et al., 2025b), which mitigates tokenization artifacts regarding numerical precision and aligns with human cognitive pattern recognition .

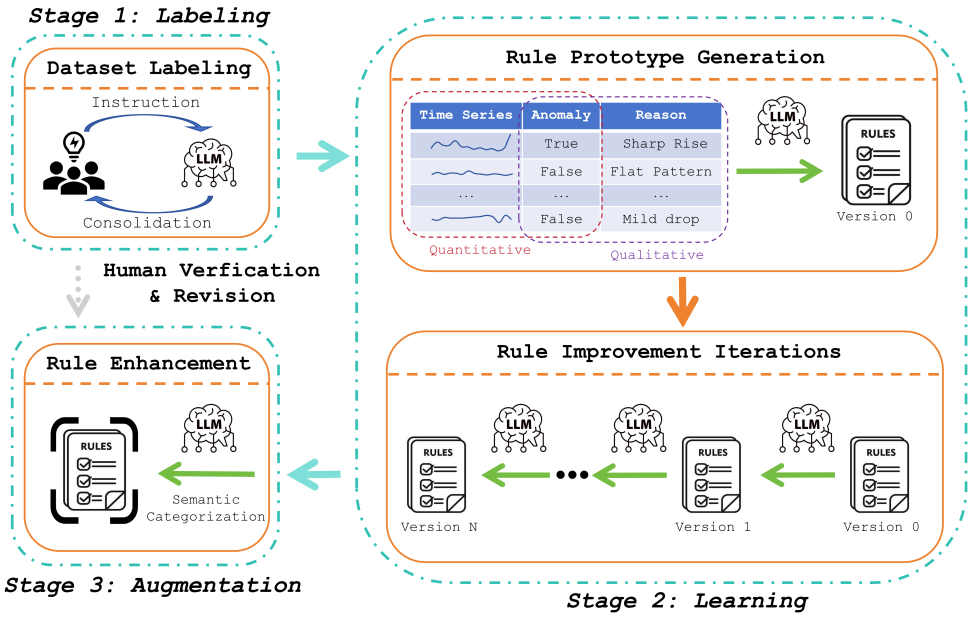

Figure 1: Our three-stage framework: 1) Multimodal LLM-based data labeling, 2) Iterative LLM-driven logic rule learning, and 3) Semantic categorization for interpretability. This process distills domain expertise into efficient, executable rules.

Table 1: Performance comparison of anomaly detection methods on 10K products (892 anomalies)

Method	Recall (%)	Precision (%)	F1 Score (%)	Execution Time (s)	Business Interpretability
iForest-Based Method	83.95	80.03	81.94	1,250.33	Medium
LSTM-VAE Hybrid	88.00	66.98	76.04	348.50	Low
Anomaly Transformer	90.02	72.02	79.93	425.80	Low
Claude Sonnet 4	94.96	97.02	95.98	18,234.17	High
Amazon Nova Pro	93.05	94.97	94.00	5,267.28	High
Meta Llama 3.2	92.04	93.94	92.98	4,838.37	High
Logic-Based Rules	**91.03**	**93.01**	**92.01**	**4.27**	**High**

2. Learning Stage: Iterative Rule Optimization. This stage, our core contribution, functions as an offline "compiler" that distills the labeled dataset into deterministic logic. First, an LLM synthesizes a rule prototype by combining *qualitative reasoning* (textual rationales) with *quantitative statistics* (e.g., z-scores). We then treat rule optimization as a learning problem: in an iterative loop, the system generates a *behavioral analysis* (e.g., "over-conservative") that functions as a *semantic gradient* (Yang et al., 2023), guiding the LLM to apply targeted modifications (e.g., "decrease threshold").

3. Augmentation Stage: Semantic Categorization. To enhance business utility, we use LLMs to map specific rule conditions to business-relevant categories (e.g., mapping `value > 80` to "Critical Stock-Out Crisis"). This ensures the system outputs not just a binary alert, but a context-aware diagnosis without altering the underlying detection logic, enabling targeted remediation strategies.

In-production evaluation (Table 1) demonstrates that our framework outperforms iForest (previous in-production method) and deep learning baselines with business context encoded. Our method achieves a 92.01% F1 score comparable to direct LLM inference. Crucially, our logic-based rules cut execution time from the LLMs' 5+ hours to just **4.27 seconds**, delivering a $\sim$1,000-4,000$\times$ speedup and the deterministic reliability essential for scalable deployment. Additionally, our method incurs minimal cost while making **no API inference calls** in production. Detailed methodology and results are included in the Appendix.

REFERENCES

Virginia Aglietti, Ira Ktena, Jessica Schrouff, Eleni Sgouritsa, Francisco Ruiz, Alan Malek, Alexis Bellot, and Silvia Chiappa. Funbo: Discovering acquisition functions for bayesian optimization with funsearch. In *Forty-second International Conference on Machine Learning*, 2024.

Sarah Alnegheimish, Linh Nguyen, Laure Berti-Equille, and Kalyan Veeramachaneni. Can large language models be anomaly detectors for time series? In *2024 IEEE 11th International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 1–10. IEEE, 2024.

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, et al. Chronos: Learning the language of time series. *arXiv preprint arXiv:2403.07815*, 2024.

Yifu Cai, Xinyu Li, Mononito Goswami, Michał Wiliński, Gus Welter, and Artur Dubrawski. Timeseriesgym: A scalable benchmark for (time series) machine learning engineering agents. *arXiv preprint arXiv:2505.13291*, 2025.

Abhimanyu Das, Matthew Faw, Rajat Sen, and Yichen Zhou. In-context fine-tuning for time-series foundation models. *arXiv preprint arXiv:2410.24087*, 2024a.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. *arXiv*, 2024b. URL https://arxiv.org/abs/2310.10688. arXiv preprint.

Manqing Dong, Hao Huang, and Longbing Cao. Can llms serve as time series anomaly detectors? *arXiv preprint arXiv:2408.03475*, 2024.

Azul Garza, Cristian Challu, and Max Mergenthaler-Canseco. Timegpt-1. *arXiv*, 2024. URL https://arxiv.org/abs/2310.03589. arXiv preprint.

Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. Moment: A family of open time-series foundation models. *arXiv preprint arXiv:2402.03885*, 2024.

Shi Bin Hoo, Samuel Müller, David Salinas, and Frank Hutter. From tables to time: Extending tabpfn-v2 to time series forecasting. *arXiv*, 2026. URL https://arxiv.org/abs/2501.02945. arXiv preprint.

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. Ruler: What's the real context size of your long-context language models? *arXiv preprint arXiv:2404.06654*, 2024.

Bosong Huang, Ming Jin, Yuxuan Liang, Johan Barthelemy, Debo Cheng, Qingsong Wen, Chenghao Liu, and Shirui Pan. Shapex: Shapelet-driven post hoc explanations for time series classification models. *arXiv preprint arXiv:2510.20084*, 2025.

Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, et al. Time-llm: Time series forecasting by reprogramming large language models. *arXiv preprint arXiv:2310.01728*, 2023.

Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y. Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, and Qingsong Wen. Time-llm: Time series forecasting by reprogramming large language models. *arXiv*, 2024. URL https://arxiv.org/abs/2310.01728. arXiv preprint.

Tian Lan, Hao Duong Le, Jinbo Li, Wenjun He, Meng Wang, Chenghao Liu, and Chen Zhang. Axis: Explainable time series anomaly detection with large language models. *arXiv preprint arXiv:2509.24378*, 2025.

Shuyu Lin, Ronald Clark, Robert Birke, Sandro Schönborn, Niki Trigoni, and Stephen Roberts. Anomaly detection for time series using vae-lstm hybrid model. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4322–4326. Ieee, 2020.

Chen Liu, Shibo He, Qihang Zhou, Shizhong Li, and Wenchao Meng. Large language model guided knowledge distillation for time series anomaly detection. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence (IJCAI-24)*, pp. 2162–2170. International Joint Conferences on Artificial Intelligence Organization, 2024a. doi: 10.24963/ijcai.2024/239.

Chen Liu, Shibo He, Qihang Zhou, Shizhong Li, and Wenchao Meng. Large language model guided knowledge distillation for time series anomaly detection. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pp. 2162–2170, 2024b.

Fei Liu, Tong Xialiang, Mingxuan Yuan, Xi Lin, Fu Luo, Zhenkun Wang, Zhichao Lu, and Qingfu Zhang. Evolution of heuristics: Towards efficient automatic algorithm design using large language model. In *Forty-first International Conference on Machine Learning*, 2024c.

Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pp. 413–422, 2008. doi: 10.1109/ICDM.2008.17.

Jun Liu, Chaoyun Zhang, Jiaxu Qian, Minghua Ma, Si Qin, Chetan Bansal, Qingwei Lin, Saravan Rajmohan, and Dongmei Zhang. Large language models can deliver accurate and interpretable time series anomaly detection. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pp. 4623–4634, 2025a.

Penghang Liu, Elizabeth Fons, Svitlana Vyetrenko, Daniel Borrajo, Vamsi Potluru, and Manuela Veloso. Ts-agent: A time series reasoning agent with iterative statistical insight gathering. *arXiv preprint arXiv:2510.07432*, 2025b.

Pingchuan Ma, Tsun-Hsuan Wang, Minghao Guo, Zhiqing Sun, Joshua B Tenenbaum, Daniela Rus, Chuang Gan, and Wojciech Matusik. Llm and simulation as bilevel optimizers: A new paradigm to advance physical scientific discovery. In *International Conference on Machine Learning*, pp. 33940–33962. PMLR, 2024.

Mohsin Munir, Shoaib Ahmed Siddiqui, Andreas Dengel, and Sheraz Ahmed. Deepant: A deep learning approach for unsupervised anomaly detection in time series. *Ieee Access*, 7:1991–2005, 2018.

Kashif Rasul, Arjun Ashok, Andrew Robert Williams, Hena Ghonia, Rishika Bhagwatkar, Arian Khorasani, Mohammad Javad Darvishi Bayazi, George Adamopoulos, Roland Riachi, Nadhir Hassen, Marin Biloš, Sahil Garg, Anderson Schneider, Nicolas Chapados, Alexandre Drouin, Valentina Zantedeschi, Yuriy Nevmyvaka, and Irina Rish. Lag-llama: Towards foundation models for probabilistic time series forecasting. *arXiv*, 2024. URL https://arxiv.org/abs/2310.08278. arXiv preprint.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M Pawan Kumar, Emilien Dupont, Francisco JR Ruiz, Jordan S Ellenberg, Pengming Wang, Omar Fawzi, et al. Mathematical discoveries from program search with large language models. *Nature*, 625(7995):468–475, 2024.

Alicia Russell-Gilbert, Alexander Sommers, Andrew Thompson, Logan Cummins, Sudip Mittal, Shahram Rahimi, Maria Seale, Joseph Jaboure, Thomas Arnold, and Joshua Church. Aad-llm: Adaptive anomaly detection using large language models. In *2024 IEEE International Conference on Big Data (BigData)*, pp. 4194–4203. IEEE, 2024.

Chathurangi Shyalika, Harleen Kaur Bagga, Ahan Bhatt, Renjith Prasad, Alaa Al Ghazo, and Amit Sheth. Time series foundational models: Their role in anomaly detection and prediction. *arXiv*, 2024. URL https://arxiv.org/abs/2412.19286. arXiv preprint.

Dimitris Spathis and Fahim Kawsar. The first step is the hardest: Pitfalls of representing and tokenizing temporal data for large language models. *Journal of the American Medical Informatics Association*, 31(9):2151–2158, 2024.

Mingtian Tan, Mike A. Merrill, Vinayak Gupta, Tim Althoff, and Thomas Hartvigsen. Are language models actually useful for time series forecasting? *arXiv*, 2024. URL https://arxiv.org/abs/2406.16964. arXiv preprint.

David Alexander Tedjopurnomo, Zhifeng Bao, Baihua Zheng, Farhana Murtaza Choudhury, and Alex Kai Qin. A survey on modern deep neural network for traffic prediction: Trends, methods and challenges. *IEEE Transactions on Knowledge and Data Engineering*, 34(4):1544–1561, 2020.

Avijit Thawani, Jay Pujara, Pedro A Szekely, and Filip Ilievski. Representing numbers in nlp: a survey and a vision. *arXiv preprint arXiv:2103.13136*, 2021.

Jiaqi Wang, Hanqi Jiang, Yiheng Liu, Chong Ma, Xu Zhang, Yi Pan, Mengyuan Liu, Peiran Gu, Sichen Xia, Wenjun Li, et al. A comprehensive review of multimodal large language models: Performance and challenges across different tasks. *arXiv preprint arXiv:2408.01319*, 2024.

Andrew Robert Williams, Arjun Ashok, Étienne Marcotte, Valentina Zantedeschi, Jithendaraa Subramanian, Roland Riachi, James Requeima, Alexandre Lacoste, Irina Rish, Nicolas Chapados, et al. Context is key: A benchmark for forecasting with essential textual information. *arXiv preprint arXiv:2410.18959*, 2024.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers. *arXiv*, 2024. URL `https://arxiv.org/abs/2402.02592`. arXiv preprint.

Jiehui Xu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Anomaly transformer: Time series anomaly detection with association discrepancy. In *International Conference on Learning Representations*, 2022.

Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V Le, Denny Zhou, and Xinyun Chen. Large language models as optimizers. In *The Twelfth International Conference on Learning Representations*, 2023.

Xiyuan Zhang, Ranak Roy Chowdhury, Rajesh K Gupta, and Jingbo Shang. Large language models for time series: A survey. *arXiv preprint arXiv:2402.01801*, 2024.

Zihao Zhou and Rose Yu. Can llms understand time series anomalies? *arXiv preprint arXiv:2410.05440*, 2024.

APPENDIX

We include in Appendix the literature review (Sec. A), the detailed description of the methodology (Sec. B), and the experimental settings and result analysis for the in-production evaluation (Sec. C).

A RELATED WORK

Recent efforts in time series forecasting and anomaly detection have embraced the foundation model paradigm. Instead of building bespoke models per dataset, researchers pretrain large transformers on heterogeneous time series data and reuse them in zero-shot or few-shot settings. Zhang et al. (2024) provide a comprehensive survey of these approaches, noting that cross-domain generalization and unified modelling interfaces motivate the use of large pretrained models for time series.

A number of time series foundation models adapt language-model training recipes to numerical sequences. Chronos (Ansari et al., 2024) quantizes real-valued series into discrete tokens and trains transformer-based architectures via cross-entropy loss; its pretrained models achieve strong zero-shot performance on unseen datasets. Google's TimesFM, formalized as a decoder-only foundation model for time series (Das et al., 2024b), trains on billions of time points and demonstrates competitive zero-shot accuracy. Nixtla's TimeGPT similarly targets universal forecasting using generative pretraining and shows robust performance across domains (Garza et al., 2024). Lag-Llama proposes a probabilistic foundation model that uses lagged covariates and demonstrates cross-domain transfer (Rasul et al., 2024), while Moirai introduces a masked encoder-based universal transformer trained on a massive multi-domain corpus (Woo et al., 2024).

Parallel work focuses on reprogramming or adapting general-purpose LLMs for time series rather than pretraining models from scratch. Time-LLM frames forecasting as a modality-alignment problem: it reprograms time-series windows into patch sequences, adds prefix prompts, and feeds the resulting tokens into a frozen language model (Jin et al., 2024). The authors report that careful prompting and projection allow large language models to capture time-series patterns in few-shot and zero-shot regimes.

Compared with forecasting, the application of foundation models to time-series anomaly detection is also emerging. Liu et al. (2024a) propose AnomalyLLM, which uses a pretrained LLM as a teacher and distills its representations into a smaller student network for anomaly detection. However, a systematic study by Shyalika et al. (2024) concludes that existing time-series foundation models often match or underperform relative to classical methods on anomaly detection tasks and require substantial computation.

Finally, several critical evaluations question the true benefit of incorporating language models into time series pipelines. Tan et al. (2024) conduct an extensive ablation study and show that removing the LLM component from several popular LLM-based forecasters does not degrade (and sometimes improves) forecast accuracy, suggesting that specialized attention and patching architectures may be sufficient. Complementary evidence comes from TabPFN-TS (Hoo et al., 2026), which treats forecasting as a tabular regression task and achieves strong zero-shot results without any time-series-specific pretraining. These findings underscore the importance of principled evaluations and the need for efficient representations that support both accurate forecasting and reliable anomaly detection.

B DETAILED METHODOLOGY

The proposed large language model (LLM)-assisted rule learning for time series anomaly detection operates in three stages: 1) a labeling stage (Sec. B.1), 2) a learning stage (Sec. B.2), and 3) an augmentation stage (Sec. B.3).

B.1 LABELING STAGE

To incorporate human domain knowledge into a scalable and interpretable anomaly detection, we leverage the power of multi-modal (vision-language) LLMs and input the visualized time series plots as a part of the prompt (Wang et al., 2024; Russell-Gilbert et al., 2024; Alnegheimish et al., 2024; Liu et al., 2024b; 2025a). The choice of visual representation over numerical input is motivated by three

reasons. First, literature demonstrates that LLMs often struggle with numerical precision due to tokenization artifacts (Thawani et al., 2021; Spathis & Kawsar, 2024), where numerical values may be split across tokens or mapped to similar embeddings, introducing potential errors. Second, visual anomaly detection aligns with human cognitive process, as domain experts can more easily identify anomalies through pattern recognition from visualized plots than numerical series. Third, the image-based approach provides greater generalizability across diverse time series metrics without requiring metric-specific pre-processing.

In addition to the visualized time series data, the business context and domain knowledge are included in the prompt input to the LLMs to ensure that the detection results are aware of the application context. For the output detection results, we specifically require in the prompt that the LLMs should provide with both 1) the binary detection classification (anomalous/normal) and 2) the reason for this detection decision. In this manner, human-level domain knowledge is encoded into the dataset labeled by multimodal LLMs.

To ensure reliable labeling, we implement a two-tier consensus mechanism. We employ multiple multimodal LLMs from different providers to detect anomalies. Each LLM independently analyzes each data point multiple times, and we adopt the majority vote as that model's decision. We only accept a data point's label when all LLMs' majority votes agree, thereby achieving both intra- and inter-model consistency (Hsieh et al., 2024). This multi-model approach with internal voting mitigates potential systematic biases inherent to any single LLM while reducing the impact of stochastic variations in LLM responses[1].

Although LLMs showcase promising capability in anomaly detection (Zhou & Yu, 2024; Dong et al., 2024; Liu et al., 2025b), our evaluation reveals that certain data patterns consistently produce disagreement between different models or different trials for an identical model, particularly in boundary cases where the distinction between normal and anomalous behaviors depends on subtle contextual factors. To ensure the quality of the labeled dataset, we implement an iterative refinement process where human experts manually review samples of the labeled data, identify systematic inconsistencies, refine the prompts accordingly to align detection results with business context, and decide those baseline filtering rules to detect anomalies. On the other hand, this does not indicate that human experts can accomplish the labeling task without the help of multimodal LLMs.

Specifically, employing LLMs to automate and standardize the detection procedure helps human experts consolidate the domain expertise: by checking both agreements and disagreements provided by LLMs on large-scale datasets, humans have a more targeted direction to solidify the understandings of anomalies. For example, in initial rounds, LLMs provided inconsistent results when the value of time series moderately rise following long-period zero values. This finding helped humans to set up a deterministic threshold for time series data to be considered as an anomaly. Without LLMs, human experts could have difficulty manually identifying the necessity of specifying such a threshold. In other words, LLMs act as *operators*, who continuously label data points at large scales under instructions, while human experts serve as *managers* to evaluate the results and modify the instructions. This *human-AI collaboration* is essential for generating 1) explicit filtration rules excluding scenarios that LLMs can hardly detect and 2) a high-quality labeled dataset for the following stage.

B.2 Learning Stage

In the learning stage, we first employ the LLM to generate an initial rule prototype (Sec. B.2.1). The input to the LLM includes 1) LLM-extracted/summarized reasoning patterns and 2) data-driven statistics, both from the labeled dataset. Then, we implement a systematic iterative improvement pipeline, which improves rules performance through behavioral analysis and targeted modifications, driven by the LLM (Sec. B.2.2). Additionally, the entire learning procedure is consistent with the well-established machine learning pipeline, and therefore ensures generalizability and prevents overfitting (Sec. B.2.3). Compared to the first labeling stage involving human-AI collaboration, this learning stage is data-driven and fully automated *without* any human intervention.

[1]The prompt input to different multimodal LLMs remains the same and the temperature is set to 0.

B.2.1 RULE PROTOTYPE GENERATION

The learning stage starts from the labeled dataset that contains (i) time series data, (ii) binary anomaly label, and (iii) the reason for the detected label in text form. Both (ii) and (iii) are provided by multimodal LLMs in the previous labeling stage. Our approach first extracts both *qualitative* and *quantitative* information for logic-based symbolic rules generation:

- **Qualitative Reasoning:** We use the LLM to extract the textual reasons from the dataset on why each time series data is detected as an anomaly or not. Extracted textual information includes "current/recent weeks," "significantly higher than," "breaking established trends," etc. Note that we do not include any specific information about the metric nor anomaly descriptions in the prompt during this step, and these qualitative information is inherited from the "reasons" in the dataset provided by the multimodal LLMs in the labeling stage.
- **Quantitative Thresholds:** We implement a standard feature engineering procedure to summarize the statistics of normal points and anomalies, including historical statistics, z-scores, trend detection, and zero-rate patterns. This step quantifies the features of normal points/anomalies, where we use standardized Python code instead of LLMs.

With both the qualitative and quantitative information provided as the input, a reasoning LLM is invoked to generate the logic-based rules to detect anomalies, with a specified requirement on the output format. Thus we get a logic-based rules prototype for further improvement. This strategy integrates the semantic understanding of LLMs with standard statistical analysis to automatically generate interpretable, executable logic rules without requiring domain expertise or manual feature engineering.

B.2.2 ITERATIVE RULE IMPROVEMENT

After obtaining the initial rules prototype, we implement a systematic and automated improvement pipeline to iteratively improve the rules' performance (F1 score for anomaly detection). The improvement process consists of iterative 1) performance evaluation & behavioral analysis and 2) targeted modification:

1. **Performance Evaluation & Behavioral Analysis:** In each iteration, the rule is first evaluated by metrics including precision, recall, and F1-score, based on the labeled dataset, with robust handling of edge cases such as rules execution failures. Based on the evaluation, the rules behavioral patterns are categorized into distinct types including *over-conservative*, *over-aggressive*, etc. This behavioral categorization is derived from confusion matrix analysis rather than relying solely on F1-score, providing more granular insights into the rules specific detection deficiencies.
2. **Targeted Modification:** For each identified behavioral pattern, we leverage the LLM's reasoning capabilities to generate targeted rules modifications. The LLM receives 1) the current rules, and 2) behavioral analysis results. Therefore, LLM identifies specific improvement directions corresponding to the detected pattern (e.g., decreasing thresholds for over-conservative rules). The LLM reasons about the logical structure of the rules and generates contextually appropriate rule modifications that address the specific performance deficiencies identified through behavioral analysis.

Additionally, the improvement employs a *trajectory-aware* learning procedure. That is, the pipeline maintains a complete history of all improvement attempts (as well as failures), including the rule versions, performance metrics, and modification outcomes. This trajectory information is fed back to the LLM in subsequent iterations, enabling it to learn from previous failures and avoid repeating ineffective modification patterns. The process continues iteratively until the maximum number of iterations is reached. *We also provides in prompt the hierarchal templates that align with the business contexts to encourage LLMs to propose layers of rules.* The procedure is formalized and summarized in Algorithm 1, as well as a *simplified* illustration in Fig. 2.

This automated improvement approach enables systematic rule optimization through iterative behavioral analysis and targeted modifications, eliminating the need for manual rule tuning while ensuring consistent improvement toward the desired performance target. The trajectory-aware mechanism

Algorithm 1 Iterative Rule Refinement Pipeline

Require: Initial rule prototype R_0, labeled dataset D, maximum iterations N_{max}
Ensure: Improved rule R_{best} with performance metrics

Initialize $R_{current} \leftarrow R_0$, $R_{best} \leftarrow R_0$, trajectory $T \leftarrow \emptyset$
Evaluate initial performance: $M_0 \leftarrow \text{EvaluateRule}(R_0, D)$
$M_{current} \leftarrow M_0$, $M_{best} \leftarrow M_0$
for $i = 1$ to N_{max} **do**
  // Performance Evaluation & Behavioral Analysis
  Compute confusion matrix from $M_{current}$
  $B \leftarrow \text{AnalyzeBehavior}(M_{current})$ // Categorize: over-conservative, over-aggressive, etc.
  // Trajectory-Aware Rule Modification
  Update trajectory: $T \leftarrow T \cup \{(R_{current}, M_{current}, B)\}$
  Generate improvement prompt with $(R_{current}, B, T)$
  $R_{new} \leftarrow \text{LLM}(\text{prompt})$ // Generate improved rule code
  // Evaluation & Rule Tracking
  $M_{new} \leftarrow \text{EvaluateRule}(R_{new}, D)$
  if M_{new}.F1_score $>$ M_{best}.F1_score **then**
    $R_{best} \leftarrow R_{new}$, $M_{best} \leftarrow M_{new}$
  end if
  // Update Current Rule for Next Iteration
  if M_{new}.F1_score $>$ $M_{current}$.F1_score **then**
    $R_{current} \leftarrow R_{new}$, $M_{current} \leftarrow M_{new}$
  end if
end for
return R_{best}, M_{best}, trajectory T

prevents cyclical improvements and promotes convergent optimization behavior toward the specified performance target. Additionally, the iterative improvement pipeline using LLMs shares some similar spirits with the *automatic heuristic design*, where LLMs automatically propose new algorithms, implement them, get the evaluation feedback, and then revise them based on the feedback (Liu et al., 2024c; Romera-Paredes et al., 2024; Aglietti et al., 2024; Ma et al., 2024). The proposed algorithms in these methods are largely maintained and decided by a heuristic strategy (e.g,. evolution computation) to complete the entire procedure. In comparison, we cast the LLM-assisted rule learning into a standardized machine learning pipeline.

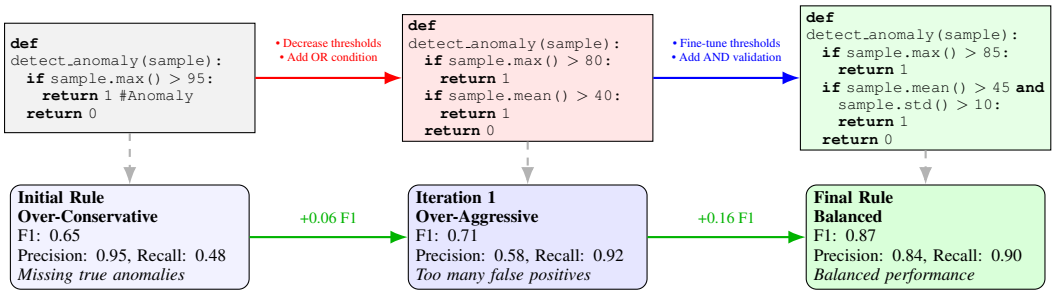

Figure 2: A simplified illustration of iterative rule improvements: The performance evaluation based on the labeled dataset identifies behavioral patterns of the rule (over-conservative, over-aggressive, etc.) and LLMs apply targeted modifications to refine the rule, achieving the target performance.

B.2.3 Complete Learning Framework for Logic-based Rules

During the iterative rule improvement pipeline, the invoked LLM revises the current logic-based rules using the provided "direction" (e.g., decreasing the threshold), which is derived from the behavioral analysis (e.g., over-conservative) in the performance evaluation. This "direction" shares similar spirit with the gradient descent when minimizing the loss function during the machine learning (ML) procedure (Yang et al., 2023). Inspired by this similarity between our LLM-assisted im-

provement pipeline and the gradient descent method in classical ML, we develop a systematic procedure to learn a logic-based rule for anomaly detection that follows established learning methodology with training/test splits and early stopping mechanisms. A comprehensive comparison between our proposed method and classical ML learning procedure is included in Table 2.

To be more specific, the complete learning framework for logic-based rules consists of:

1. **Data Split:** The dataset is divided into training and test sets following standard ML.
2. **Two-Phase Learning per Rule:** For each rule, the procedure first generates an initial rule prototype through pattern extraction and feature engineering as in Sec. B.2.1. It then randomly divides the training set into learning and validation set, and then conducts multi-epoch learning. In each epoch, the learning procedure implements the iterative improvement pipeline (behavioral analysis $\rightarrow$ LLM-driven rule modification $\rightarrow$ performance evaluation) on training data as in Sec. B.2.2, followed by an evaluation on the validation set. The epoch is accepted when the performance gap between the training and validation sets is within a threshold.
3. **Early Stopping & Convergence:** For each set of logic-based rules, learning terminates when either the target F1-score is achieved, maximum epochs are reached, or no improvement is observed for a patience period.
4. **Multiple Rule Generation & Selection:** The procedure generates multiple rules from the same training dataset to account for the stochastic nature of LLMs. After generating multiple candidate rules, the final rule is selected based on a held-out test dataset, ensuring unbiased model selection and avoiding overfitting.

This learning procedure integrates the interpretability advantages of logic-based rules, supported by LLM, with the well-established learning methodology of classical machine learning, ensuring the generalizability and preventing overfitting.

Table 2: Comparison between Classical ML and Our Rule Learning Method

Classical ML Framework	
Optimization	Minimize loss function
Parameters	Continuous (weights, biases)
"Gradient"	Mathematical: $\nabla L = \partial L / \partial \theta$
Update	Numerical: $\theta = \theta - \alpha \nabla L$
Learning Rate	Fixed/adaptive scalar
Memory	Momentum, optimizer states
Epoch	Sequential gradient descent
Starting Point	Often randomly-selected
Convergence	Loss threshold
Our Rule Learning Method	
Optimization	Maximize F1 score
Parameters	Discrete (symbolic logic rules, thresholds)
"Gradient"	Semantic: targeted modification from rule analysis
Update	Symbolic: LLM-based rule modification
Learning Rate	LLM's contextual reasoning
Memory	Trajectory of rule attempts & performances
Epoch	Iterative improvement pipeline (Sec. B.2.2)
Starting Point	LLM-generated prototypes (Sec. B.2.1)
Convergence	Target F1 score achieved

At the end of this part, we explain why implementing LLM-driven logic-based rule learning instead of supervised learning or traditional heuristic search procedures using the labeled data attained in Sec. B.1.

- First, regarding the learning mechanism, traditional rule induction (e.g., Decision Trees) is constrained by pre-defined feature vectors, whereas our approach performs *simultaneous feature synthesis and rule induction*, i.e., generating novel programmatic features on the fly. Similarly, unlike numerical optimization which targets parametric tuning, LLMs optimize the *logical structure* itself, utilizing semantic priors to prune the combinatorial search space efficiently.
- Second, the learned rules are explicit and human-readable, enabling verification and optional revision by human experts. In contrast, machine learning models are generally black boxes and difficult to interpret in production.
- Third, although LLMs exhibit promising anomaly detection capabilities, their detections are not fully reliable and can exhibit inconsistencies or hallucinations. Logic-based rules learned from the labeled dataset "distill" the predominant patterns, while supervised learning methods may overfit to noise in the training data.
- Lastly, time series data in supply chain management (as well as other domains) could exhibit strong non-stationarity: data during holiday weeks or promotions differ significantly from that in normal weeks. This non-stationarity leads to distribution shifts, under which supervised learning methods degrade while our learned rules, with deterministic and consistent logic, exhibit strong robustness. We include numerical results to support the advantages of our learned rules over supervised learning methods in Sec. C.

B.3 Augmentation Stage

In this stage, we implement a semantic categorization to enhance the interpretability of the rules through two steps: **1. Rule Analysis & Category Taxonomy:** Our method analyzes the learned rules to extract the detection principles using LLMs, identifying strategies such as z-score thresholds, ratio comparisons, and trend detection. Based on the rule analysis, the LLM generates a taxonomy of anomalous categories accounting for business contexts, e.g., "Critical Stock-Out Crisis," "Moderate Stock Pressure," "Escalating Stock-Out Emergency." **2. Rule Enhancement & Category Assignment:** The learned rules are enriched to classify anomalies into the identified categories without modifying their detection logic, supported by the LLM. The enhanced rules return both the detection results and the category classifications, enabling more informed analysis. The in-production rules learned and categorized exhibit hierarchical structures and encode confidential business insights, and therefore, we present simplified examples in Table 3 for illustration.

This categorization enhancement provides critical context for human analysts by explaining not just *that* an anomaly occurred, but *what type* of anomaly was detected. Importantly, this semantic layer is added without sacrificing detection performance, as the underlying detection logic remains unchanged. The output categorization enables more targeted remediation strategies and improved root cause analysis by distinguishing between different categories of anomalies.

Table 3: Simplified Rule Examples and Their Categories

Category	**Rules of Anomalies**
Critical Stock-Out Crisis	`current_value >= 80.0`
	`ratio of current_value and values[-2]> 10 && current_value >= 50`
Moderate Stock Pressure	`current_value >= 35 && z_score >= 5.0`
	`recent_mean < 3.0 && current_value >= 12`

C Results

We include more details of the in-production results in Table 1. Due to business confidentiality, we do not include detailed descriptions of this metric here. Typically, a relatively high value of this metric is flagged as an anomaly, though recent trends and historical value ranges must also be considered. The task is to determine whether the current week (the last timestamp) indicates an

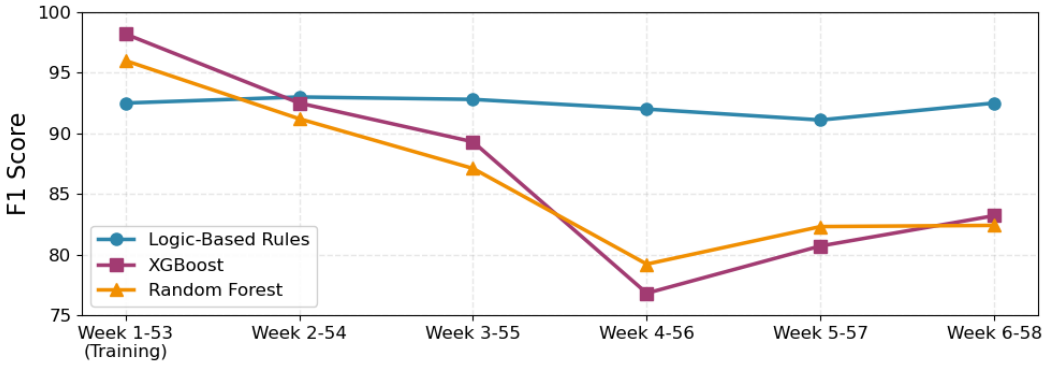

Figure 3: The comparison between the learned rules and supervised learning models trained on the label dataset. Week 56 includes a holiday that causes demand spikes across products, degrading the performance of supervised learning models because of the high anomaly rate.

anomaly for each ASIN. The evaluation is based on a dataset containing 10K products with 53 weeks of data (≈1 year). The dataset is labeled by multimodal (vision-language) LLMs and reviewed by human experts as described in Sec. B.1. To prevent data leakage, the evaluation dataset does not overlap with the training set.

The compared baseline methods include:

- Amazon's in-production anomaly detection at aggregated category levels, which is a modified Isolation Forest (iForest) (Liu et al., 2008) with business context manually encoded.
- Unsupervised deep learning models including VAE-LSTM Hybrid (Lin et al., 2020), and Anomaly Transformer (Xu et al., 2022).
- Direct LLMs using the same input as in the labeling stage (Sec. B.1), i.e., we reproduce the results. The employed multimodal LLMs include 1) Anthropic Claude Sonnet 4, 2) Amazon Nova Pro, and 3) Meta Llama 3.2.

In Table 1, the execution time represents the computation cost for detecting anomalies across the entire dataset. The implementation is hosted on an AWS SageMaker ml.r7i.48xlarge instance type. The experimental results provide the following insights:

1. Deep learning methods achieve unsatisfactory results due to low precision, demonstrating over-sensitivity. Specifically, the business context reveals that some trend-breaking patterns (e.g., sharp decreases) are not actual anomalies, yet these unsupervised methods lack awareness of this context, resulting in false positives.
2. Amazon's in-production anomaly detection method (based on iForest) is specifically designed for aggregated category levels, achieving balanced performance with both reasonable precision and recall, since business context has been *manually* encoded into the algorithm. However, its performance still degrades at the individual ASIN level since time series data exhibit significantly higher uncertainty and volatility, thus providing minimal detectable patterns for iForest to leverage.
3. Multimodal LLMs are required to *reproduce* their detection results. Although the detection accuracy of using LLMs directly outperforms other methods, the results demonstrate that the detection outputs are not fully reproducible. This non-deterministic nature creates risks of missed alarms or wasted efforts in production. Additionally, the computational latency and costs associated with LLMs are critical concerns, since Amazon requires weekly anomaly detection across millions of ASINs.

Overall, our learned logic-based rules outperform other baseline methods and have been deployed in production.

Additionally, we conducted an ablation study to compare our logic-based rule learning (Sec. B.2) against supervised learning models trained on the exact same labels generated in Stage 1. This reformulates the problem as a binary classification task; among the tested models, XGBoost achieved the highest accuracy, followed by Random Forest. We employed a rolling 53-week evaluation window to explicitly test robustness against temporal distribution shifts. The results (Fig. 3) reveal a critical divergence: during Week 56, a holiday event caused widespread demand spikes, significantly altering the anomaly distribution. While supervised models degraded in performance due to their sensitivity to the anomaly rate observed during training, our logic-based rules maintained consistent high performance. This demonstrates that our framework distills fundamental detection logic robust to non-stationarity, whereas supervised methods risk overfitting to the specific distribution statistics of the training period.