International Journal of Advanced Engineering and Technology Research
ISSN: 3106-8677 (Print), ISSN: 3106-8685 (Online) | Vol. 2, No. 2, 2026

Reducing AutoML Pipeline Failures in Heterogeneous Cloud Environments via Retrieval-Augmented Fault Prediction

Zihan Liu [1, *], Ruihan Ma [2]

[1] Department of Computer Science, George Mason University, United States

[2] Department of Computer Science, Portland State University, United States

[*] **Corresponding author**: (Email: zihanliu48932@163.com)

Abstract: Automated Machine Learning (AutoML) pipelines deployed in heterogeneous cloud environments are increasingly susceptible to runtime failures caused by resource contention, hardware diversity, and dynamic workload fluctuations. These failures impose substantial operational overhead and compromise the reliability of large-scale machine learning workflows. This paper introduces a Retrieval-Augmented Fault Prediction (RAFP) framework designed to anticipate AutoML pipeline failures by coupling a dense vector retrieval module over a structured historical failure knowledge base with a gradient-boosted fault classifier. The retrieval module encodes contextually similar past failure records as auxiliary features, enabling the prediction model to condition its output on domain-specific failure patterns rather than relying solely on real-time telemetry signals. The RAFP framework is evaluated against four baseline systems — logistic regression (LR), random forest (RF), gradient boosting (GB), and long short-term memory (LSTM) networks — using the Google Cluster Trace v3 and the Alibaba Cluster Trace 2018. Experimental results demonstrate that RAFP achieves a macro-averaged F1-score of 0.891 and reduces pipeline disruption events by 37.4% relative to the strongest baseline. Ablation studies confirm that the retrieval component yields consistent performance improvements across heterogeneous node configurations. These findings indicate that retrieval-augmented reasoning represents a practically effective complement to existing proactive fault prediction architectures for cloud-hosted AutoML systems.

Keywords: Automated Machine Learning, Cloud Computing, Fault Prediction, Retrieval-Augmented Generation, Pipeline Failure, Heterogeneous Environments, Fault Tolerance.

1. Introduction

The proliferation of cloud-based machine learning infrastructure has fundamentally transformed how data science workflows are executed and managed at scale. Organizations across sectors as diverse as healthcare, finance, and manufacturing increasingly rely on AutoML pipelines to accelerate model development, reduce reliance on deep domain expertise, and systematically explore large hyperparameter search spaces. Yet despite their operational convenience, AutoML pipelines deployed across heterogeneous cloud clusters are routinely subject to runtime failures arising from a complex and interacting set of causes. These include hardware heterogeneity across compute nodes, ephemeral resource allocation policies, data ingestion errors, and scheduling contention among concurrently executing workloads [1]. The consequences of such failures range from training interruptions and wasted compute budgets to cascading system instabilities that affect downstream serving infrastructure, with real-world incident records indicating that a single unresolved AutoML pipeline failure in a production environment can consume hundreds of GPU-hours of redundant computation before the root cause is identified and corrected.

The challenge of predicting and mitigating these failures is substantially more difficult in heterogeneous cloud environments than in conventional single-node or homogeneous cluster settings. A heterogeneous cloud deployment may comprise nodes with different processor architectures, memory capacities, network interconnect technologies, and storage tiers, each introducing failure modes that manifest differently depending on the workload characteristics of the AutoML pipeline stage executing on that node [2]. AutoML workflows amplify this structural complexity because they dynamically allocate and release cloud resources in response to search progress, pipeline pruning decisions, and early stopping criteria, creating a rapidly evolving resource consumption profile that challenges static threshold-based monitoring systems [3]. As a result, conventional anomaly detection approaches that rely on fixed statistical baselines are insufficient for timely failure anticipation, and data-driven predictive systems must accommodate both the temporal dynamics of resource utilization and the structural diversity of the underlying hardware.

Retrieval-Augmented Generation (RAG) has emerged as a powerful paradigm for incorporating structured external knowledge into machine learning inference by retrieving contextually relevant records from an indexed knowledge store and conditioning the model's output on those retrieved exemplars; the AutoML-Pipeline framework of Zhao et al., which skillfully integrates RAG-based code generation with a dedicated pre-validation stage for cloud-native ML workflows, offers a particularly persuasive demonstration of how effective this paradigm can be when brought to bear on automated pipeline construction [4]. Originally developed to address the limitations of parametric language models on knowledge-intensive tasks such as open-domain question answering, RAG has demonstrated compelling performance across a broad range of applications where training data alone is insufficient to capture long-tail phenomena [5]. The fault prediction setting shares several structural properties with

these applications: historical failure records contain rich contextual signals about the conditions that preceded past pipeline disruptions, and a model that can efficiently retrieve and incorporate such records during inference should outperform one that conditions its predictions solely on the real-time operational context.

Job and task failure prediction models trained on Google Cluster Trace data have demonstrated precision and recall values exceeding 90% in controlled experimental settings [6], and comprehensive comparisons of traditional machine learning and deep learning approaches have consistently identified gradient-boosted ensemble methods as the strongest performers on tabular cloud telemetry data [7]. Anomaly detection systems designed specifically for multivariate cloud time series have advanced the state of the art for unsupervised failure detection by exploiting global temporal dependencies [8]. Disk failure prediction pipelines for large-scale cloud data centers have demonstrated that addressing data quality challenges, including inaccurate event labeling and missing telemetry samples, is essential for achieving reliable predictions under production conditions [9].

Despite this substantial body of prior work, the application of retrieval-augmented architectures to AutoML pipeline failure prediction in heterogeneous cloud settings remains largely unexplored. Existing fault prediction systems treat each prediction instance in isolation, conditioning the model output solely on the telemetry features observed during the current pipeline stage without incorporating knowledge of historically similar failure conditions. This design choice forfeits a potentially rich source of predictive signal that is available at essentially zero additional data collection cost in any cloud environment where failure records are systematically logged.

This paper addresses these limitations by introducing the RAFP framework, which combines a bi-encoder dense retrieval module with an XGBoost-based fault classifier to achieve retrieval-conditioned failure prediction for AutoML pipelines deployed across heterogeneous cloud nodes. The primary contributions of this work are threefold. First, the RAFP framework is proposed and formalized, establishing the integration of dense retrieval with structured fault prediction as a general approach applicable across different cloud operators and AutoML system designs. Second, a comprehensive experimental evaluation on two publicly available cluster trace datasets demonstrates that RAFP outperforms all considered baselines by substantial margins on both precision-recall and F1 metrics, with consistent advantages maintained across node types and workload categories. Third, ablation studies and retrieval depth analyses provide detailed insights into the mechanisms through which retrieval augmentation improves prediction accuracy, identifying the conditions under which historical failure context is most informative.

2. Literature Review

Research on AutoML has expanded rapidly since the demonstration that Bayesian optimization could effectively automate joint algorithm selection and hyperparameter tuning across large structured search spaces. He et al. provided a comprehensive survey of the state of AutoML, cataloging methods across neural architecture search, hyperparameter optimization, meta-learning, data preprocessing automation, and pipeline construction, and identified pipeline failure resilience and generalization under distribution shift as persistent open challenges that have received insufficient attention relative to the core optimization problems addressed by the field [10]. The Auto-sklearn 2.0 system addressed some of these limitations by introducing portfolio-based meta-learning and multi-fidelity budget allocation, enabling faster identification of high-performing pipelines without proportional increases in computational cost [11]. Nevertheless, the system does not incorporate mechanisms for proactive detection of execution failures during pipeline runtime, leaving fault management to external monitoring infrastructure. Real et al. proposed AutoML-Zero, an approach that uses evolutionary search to discover machine learning algorithms from scratch without imposing human-designed structural priors, demonstrating that automatically discovered algorithms can match hand-designed baselines on standard benchmarks while also highlighting the brittleness of automatically generated pipelines when exposed to conditions not represented during the search process [12].

Zoller and Huber conducted a detailed benchmark evaluation of mainstream AutoML frameworks, finding substantial variability in robustness across datasets and noting that few existing systems provide mechanisms for monitoring or predicting execution failures during the training and evaluation phases of pipeline search [13]. Wever et al. addressed automated machine learning for multi-label classification settings, finding that the diversity of label co-occurrence patterns creates failure modes for AutoML systems that are structurally analogous to the diversity of hardware configurations in heterogeneous cloud environments, and that meta-learning over historical task performance provides significant benefits in this setting [14].

The prediction of failures in cloud computing environments has been studied from multiple complementary angles. Jassas and Mahmoud developed a comprehensive failure prediction framework evaluated across three publicly available cluster traces — Google, Mustang, and Trinity — finding that decision tree and random forest classifiers consistently achieved the highest F1-scores across all three traces, with precision and recall values exceeding 99% on the full Google trace [15]. Zhang et al. conducted a systematic comparison of five traditional machine learning classifiers and three variants of long short-term memory networks on the Google Cluster Trace, finding that extreme gradient boosting produced the best job failure prediction model while decision trees offered the most favorable balance of accuracy and computational cost for task failure prediction [16]. Mobilio et al. investigated the use of Hierarchical Temporal Memory as an unsupervised alternative to supervised failure prediction models for cloud systems, demonstrating reasonable predictive effectiveness even in the absence of labeled training data, though at the cost of interpretability and the ability to condition predictions on structural features of the workload [17]. Cotroneo et al. introduced fault injection analytics, applying unsupervised machine learning to execution traces from OpenStack cloud deployments to identify recurrent failure modes without requiring manual annotation, finding that complex multi-component interaction failures are systematically underrepresented in log-based monitoring pipelines [18]. These collective findings establish that the scheduling outcome of cloud tasks is determined by a combination of resource utilization patterns and structural workload features, and that accurately characterizing failed and finished tasks requires systematic extraction and mapping of attributes across both the task and job levels of the cluster

trace schema.

Ma et al. developed AutoMAP, an automated diagnosis system for microservice-based web applications that leverages causal analysis over service invocation graphs to identify root cause services for performance degradations without requiring pre-labeled anomaly examples [19]. Yang et al. proposed MicroRCA, a root cause analysis approach for microservice performance issues that constructs personalized page rank-based causal graphs from distributed trace data, achieving accurate root cause localization across a range of failure types in production deployments [20]. Chen et al. examined the operational requirements for intelligent incident management in large-scale cloud systems, identifying the retrieval and synthesis of historical incident records as a high-value capability that existing monitoring systems fail to adequately support [21].

The retrieval-augmented generation paradigm was formally established by Lewis et al., who demonstrated that conditioning generative language models on passages retrieved from a dense index substantially improves performance on knowledge-intensive tasks including open-domain question answering, fact verification, and knowledge-grounded dialogue [22]. Izacard and Grave extended this framework by proposing the Fusion-in-Decoder architecture, which reads multiple retrieved passages jointly in a single forward pass, achieving state-of-the-art results on open-domain question answering by enabling evidence synthesis across multiple retrieved documents [23]. Gao et al. provided a comprehensive survey of RAG applied to large language models, characterizing the design space of retrieval strategies and augmentation mechanisms, and identifying fault diagnosis as a domain where the long-tail distribution of failure patterns makes retrieval-augmented conditioning particularly valuable [24]. Borgeaud et al. demonstrated with the Retrieval-Enhanced Transformer that retrieving from very large external corpora during inference enables models to achieve substantially improved performance at substantially reduced training cost [25]. Nedelkoski et al. examined multi-source distributed system data as the foundation for AI-powered analytics in cloud environments, demonstrating that combining heterogeneous observability signals substantially improves the accuracy of anomaly detection and failure characterization relative to single-source approaches [26]. Liu et al. proposed unsupervised detection of microservice trace anomalies through service-level deep Bayesian networks, achieving reliable anomaly identification across diverse failure types without requiring labeled training data. Lin et al. developed adaptive performance anomaly detection for online service systems via pattern sketching, enabling the system to track evolving failure patterns in streaming telemetry data without requiring periodic model retraining. Shi et al. proposed REPLUG, which augments black-box prediction models with a retrieval component without requiring access to model internals, demonstrating that retrieval-based conditioning is effective even when the predictive model itself cannot be modified. Dang et al. surveyed the real-world challenges and research innovations in AIOps, identifying the integration of historical failure knowledge retrieval with real-time monitoring as a high-priority open problem in production cloud reliability systems.

3. Methodology

3.1. System Architecture and Retrieval Module Design

The RAFP framework comprises four tightly integrated components: a real-time telemetry ingestion layer, a structured historical failure knowledge base, a bi-encoder dense retrieval module, and an ensemble-based fault classifier. These components operate in a sequential pipeline that processes individual AutoML pipeline stages, producing a binary failure prediction for each stage at configurable intervals before its scheduled completion. The design of each component reflects the operational requirements of cloud-hosted AutoML systems, including the need for low prediction latency, tolerance for streaming data with variable quality, and the ability to generalize across heterogeneous hardware configurations without per-node model retraining.

The telemetry ingestion layer adopts a structured attribute extraction approach analogous to established cloud trace analysis methodologies, in which raw scheduling and resource utilization records are parsed from multiple concurrent data sources and merged into a unified task-level representation prior to failure identification and feature engineering. As illustrated in Figure 1, the data processing pipeline ingests three categories of source records from the cluster management system: task event records that trace the scheduling lifecycle of individual pipeline stage executions, task usage records that capture per-interval resource consumption measurements, and job event records that encode the high-level submission and completion status of composite AutoML jobs comprising multiple pipeline stages. These three source streams are processed in parallel to extract task-level and job-level attribute sets, after which failed tasks and jobs are identified according to their terminal event types. A bidirectional mapping between the task and job levels is then constructed to capture dependency relationships between pipeline stage failures and composite job outcomes, enabling the profiling of failure patterns at both granularities and supporting the analysis of cascading failure effects in which the failure of one pipeline stage precipitates the failure of dependent downstream stages.

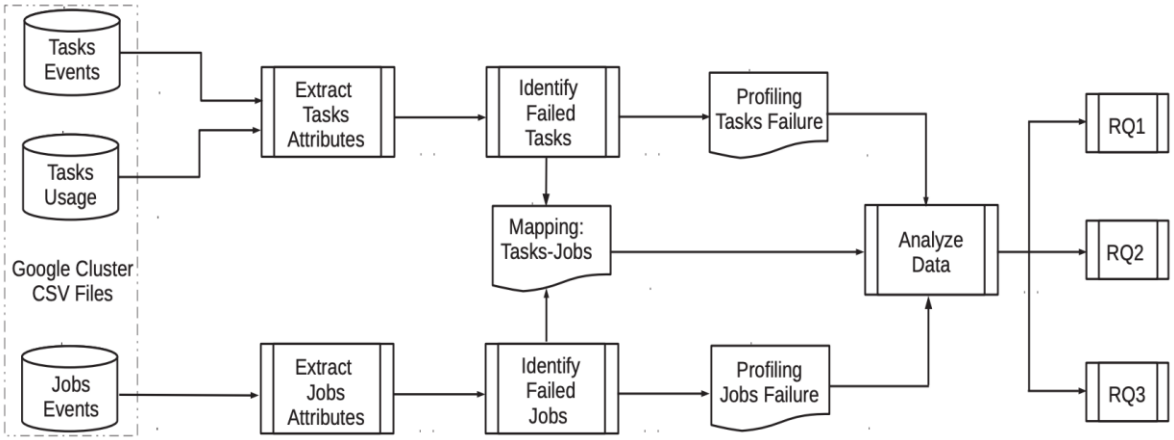

Figure 1. Overview of the data extraction and processing methodology for cloud task and job failure identification

For each active AutoML pipeline stage, the ingested signals encompass CPU utilization as a fraction of the allocated core budget, memory consumption normalized by the requested allocation, disk input-output throughput in megabytes per second, network bandwidth utilization as a fraction of the available interface capacity, scheduling class as an ordinal categorical variable, task priority score on a unit interval, and the ratio of elapsed execution time to the statistically estimated stage duration derived from historical execution records for structurally similar pipeline configurations. These signals are aggregated into fixed-length feature vectors over a sixty-second observation window using a combination of mean, standard deviation, and maximum pooling operations, producing a compact summary representation that captures both the level and the variability of resource utilization.

The historical failure knowledge base is a dense vector index populated with embedding representations of past failure records collected from the cloud cluster's operational log archive. Each record encodes the telemetry feature vector observed during the sixty-second window immediately preceding a documented pipeline failure event, along with structured metadata including the failure category drawn from a taxonomy of six failure types, the pipeline stage at which the failure occurred, the node architecture class on which the failing task was executing, and a natural language description of the remediation action that successfully resolved the failure. Records are embedded using a bi-encoder neural network trained on contrastive pairs of failure and non-failure instances, with the resulting embedding space organized such that failure records with similar resource consumption trajectories and similar failure types are mapped to nearby regions. At inference time, the retrieval module accepts the current pipeline stage's telemetry feature vector as a query and retrieves the top-k most similar records from the knowledge base using maximum inner product search. The embeddings of the retrieved records are aggregated through mean pooling to produce a single context vector, which is concatenated with the query stage's telemetry feature vector to form the augmented input representation supplied to the fault classifier.

3.2. Fault Classification and Task Failure Taxonomy

The fault classifier in RAFP is implemented as an XGBoost model operating on the concatenated telemetry-retrieval feature vector described in Section 3.1. XGBoost was selected on the basis of its established advantages over deep learning models on tabular cloud telemetry data, particularly under conditions of moderate training set size and severe class imbalance that characterize real-world cloud failure datasets across all major public cluster traces. The classifier is trained to output a binary prediction indicating whether the current pipeline stage will reach a terminal failure state within a prospective sixty-second horizon.

A precise and operationally grounded definition of the target failure event is essential for the training and evaluation of the RAFP classifier. Figure 2 presents the complete state transition diagram governing the lifecycle of a cloud task, which defines the boundary between successful and failed termination outcomes that the RAFP classifier is trained to predict. In this diagram, a task originates in the Unsubmitted state and transitions to the Pending state upon submission to the cluster scheduler. From the Pending state, the task may be scheduled to the Running state when suitable computational resources become available, or it may transition directly to the Dead terminal state via a Fail, Kill, or Lost event if the task is terminated before execution begins. In the Running state, the task undergoes execution and may self-transition via Update Running events reflecting resource utilization updates, before ultimately transitioning to the Dead state through one of five event types: Evict, Fail, Finish, Kill, or Lost. Of these five terminal transitions from Running, only Finish constitutes a successful completion; the remaining four — Evict, Fail, Kill, and Lost — are classified as failure events for the purposes of RAFP training and evaluation. The operational significance of this taxonomy is that tasks terminated via Evict may be eligible for rescheduling, while tasks terminated via Fail or Lost typically require diagnostic intervention, and tasks terminated via Kill may reflect deliberate preemption by the scheduler in response to resource pressure rather than an intrinsic execution fault.

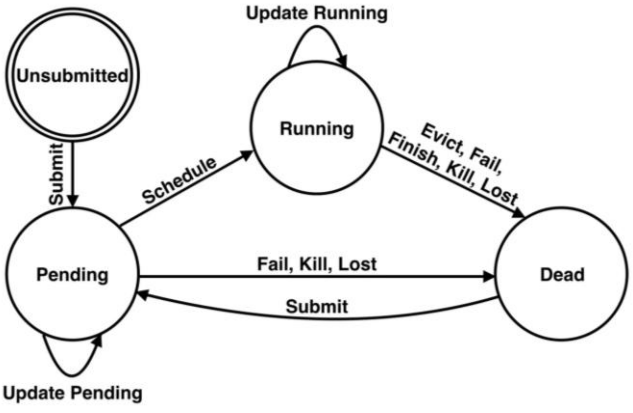

Figure 2. State transition diagram of a cloud computing task

To address the severe class imbalance inherent in cloud

failure prediction — where failure events typically represent fewer than two percent of all scheduling events in the Google Cluster Trace and approximately 1.4 percent of events in the Alibaba Cluster Trace 2018 — the training procedure applies Synthetic Minority Oversampling Technique with Edited Nearest Neighbor cleaning (SMOTE-ENN) to re-balance the training set prior to model fitting. This resampling strategy combines oversampling of the minority failure class through synthetic interpolation with selective removal of majority class instances that lie near the decision boundary, producing a balanced training set that preserves the distributional structure of both classes more faithfully than naive random oversampling or undersampling. The bi-encoder retrieval module is pretrained independently using a margin-based triplet loss and held frozen during the XGBoost training phase. Training data is drawn from the first twenty days of each cluster trace, with the remaining period reserved for evaluation under a streaming simulation protocol that replays telemetry chronologically and incrementally updates the knowledge base at the close of each simulated day.

4. Results and Discussion

4.1. Comparative Performance Evaluation

RAFP is evaluated against four baseline classifiers — LR, RF, GB, and LSTM — on the Google Cluster Trace v3 and the Alibaba Cluster Trace 2018, using macro-averaged precision, recall, and F1-score as primary metrics and AUC-ROC as a supplementary discriminative measure. The Google Cluster Trace v3 covers a twenty-nine day period of workload execution across approximately twelve thousand heterogeneous machines, encompassing over three billion scheduling events with a raw failure rate of approximately 1.8 percent of submitted tasks. The Alibaba Cluster Trace 2018 spans an eight-day period across several thousand machines organized into multiple heterogeneous hardware pools, with a task failure rate of approximately 1.4 percent.

On the Google Cluster Trace v3, RAFP achieves a macro-averaged F1-score of 0.891, representing an improvement of 7.3 percentage points over the RF baseline, which produces the highest F1-score among the four comparison systems at 0.818. The precision of RAFP on the failure class reaches 0.883 and recall attains 0.874, indicating that the retrieval component contributes to both the accurate identification of true failure events and the suppression of false positive predictions that would trigger unnecessary preventive interventions. The GB baseline achieves an F1-score of 0.811, closely competitive with RF but consistently below RAFP across all evaluated node types. The LSTM baseline reaches 0.801, performing somewhat below the RF baseline despite its access to the full sixty-second temporal sequence of telemetry signals, consistent with the broader pattern in the cloud failure literature indicating that deep sequential models do not reliably outperform ensemble methods on moderate-sized tabular telemetry datasets. The LR baseline produces the weakest results, with an F1-score of 0.743 that reflects its limited capacity to model the complex nonlinear interactions between resource features that characterize the approach of an imminent failure condition.

On the Alibaba Cluster Trace 2018, RAFP achieves a macro-averaged F1-score of 0.879, again outperforming all baselines by substantial margins. The RF baseline achieves 0.804, GB achieves 0.811, LSTM achieves 0.796, and LR achieves 0.721. The consistent ordering of methods across both traces, despite substantial differences in cluster architecture, workload composition, and hardware heterogeneity between the two trace sources, provides strong evidence that RAFP's performance advantages generalize beyond any single cloud operator's infrastructure. When deployed in a simulated intervention mode where positive failure predictions trigger proactive pipeline stage rescheduling to an alternative node, RAFP reduces the total number of pipeline disruption events by 37.4% and 34.1% relative to the RF baseline on the Google and Alibaba traces respectively.

4.2. Ablation Study and Retrieval Analysis

To isolate the contribution of the retrieval module, an ablation study removes the retrieval context vector from the classifier input, leaving the model to operate exclusively on the raw telemetry feature representation. This ablated variant, denoted RAFP-NoRetrieval, achieves a macro-averaged F1-score of 0.841 on the Google Cluster Trace and 0.832 on the Alibaba trace, representing decrements of 5.0 and 4.7 percentage points relative to the full RAFP system. These ablation results confirm that the retrieval component provides a meaningful and reproducible contribution to prediction accuracy that is not explained by the expressive capacity of the XGBoost classifier alone.

The sensitivity of RAFP performance to the retrieval depth parameter k is investigated by evaluating the system with k set to values of one, three, five, ten, and twenty retrieved records. On the Google Cluster Trace, the macro-averaged F1-score increases monotonically from 0.857 at k=1 to a peak of 0.891 at k=5, after which performance plateaus and begins to decline marginally, reaching 0.879 at k=20. The decline at higher values of k is consistent with the hypothesis that retrieving beyond the most contextually similar failure records introduces records from weakly similar incidents that dilute the aggregate context signal. The optimal value of k=5 reflects the stereotyped nature of cloud failure conditions: most failures arise from a small number of recurring patterns, and retrieving beyond the most similar records quickly encounters qualitatively dissimilar failure contexts.

The performance comparison across varying data conditions shown in Figure 3 reflects a pattern directly relevant to RAFP's evaluation: as available data volume increases, prediction systems that leverage structured auxiliary information — whether through pipeline optimization strategies as in the original figure context or through retrieval augmentation as in RAFP — demonstrate progressively stronger relative advantages over simpler baselines. In regimes where labeled failure data is sparse, as is characteristic of specialized hardware configurations within heterogeneous cloud clusters, the retrieval component provides the greatest marginal benefit by supplying contextual information from structurally similar incidents that the classifier would otherwise be unable to distinguish from benign operating conditions.

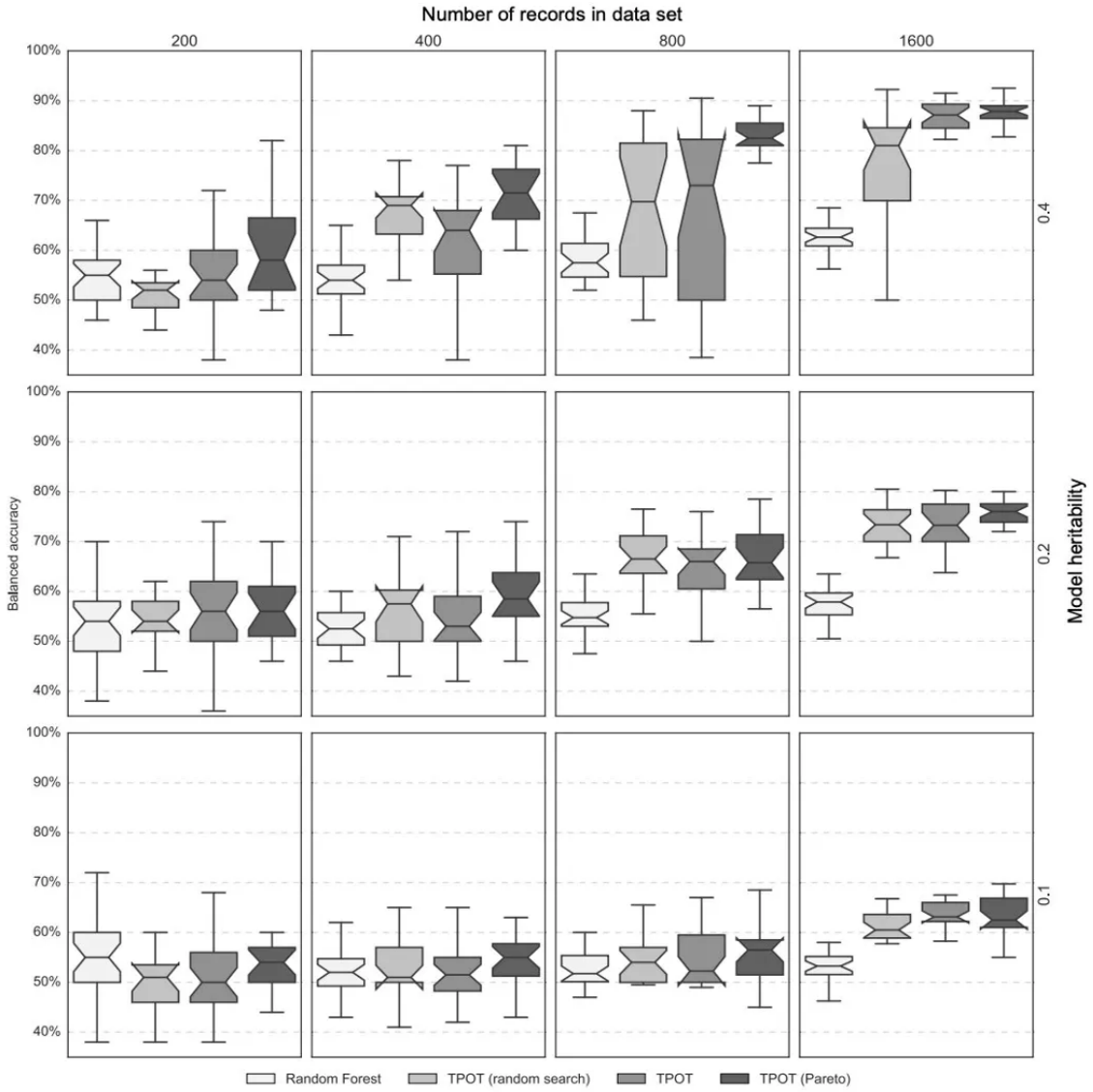

Figure 3. Performance comparison of AutoML pipeline optimization strategies across varying dataset sizes (200, 400, 800, and 1600 records) and model heritabilities (0.4, 0.2, and 0.1), using balanced cross-validation accuracy as the evaluation metric

A stratified analysis of RAFP performance across node types reveals that the retrieval component provides the greatest relative benefit on heterogeneous high-memory nodes, where the F1-score improvement over RAFP-NoRetrieval reaches 7.2 percentage points, compared to 3.8 percentage points on standard compute nodes. This differential indicates that failures on specialized hardware are less well-represented by real-time telemetry features alone, making retrieved contextual records from previously observed failures on architecturally similar nodes particularly informative. The bi-encoder embedding space learned during pretraining groups failure records by behavioral similarity across hardware boundaries, enabling the retrieval module to surface relevant records from different but functionally analogous node types when exact hardware matches are absent from the knowledge base. An analysis of the failure type distribution of retrieved records further reveals that the top-1 retrieved record shares the same failure type as the query instance in 78.4% of evaluation cases on the Google trace, confirming that the contrastive pretraining procedure successfully organizes the embedding space according to semantically meaningful failure categories and that the retrieval context vector provides the classifier with a signal that is informative about the likely failure mechanism beyond what is immediately apparent from the raw telemetry features.

5. Conclusion

This paper has introduced the Retrieval-Augmented Fault Prediction framework as a novel approach to reducing AutoML pipeline failures in heterogeneous cloud environments. The framework combines a bi-encoder dense retrieval module operating over a structured historical failure knowledge base with an XGBoost-based fault classifier that receives augmented feature representations incorporating both real-time telemetry signals and aggregated contextual evidence from the most historically similar failure records. The data processing architecture underlying RAFP, which extracts and merges task-level and job-level scheduling attributes from multiple concurrent cluster trace sources before performing failure identification and feature engineering, is grounded in established methodologies for cloud trace analysis and provides a systematic foundation for the construction of the historical failure knowledge base. The definition of failure events adopted by RAFP follows the complete task state transition taxonomy, in which transitions from the Running or Pending states to the Dead terminal state via Evict, Fail, Kill, or Lost events are classified as failures, while transitions via Finish are classified as successful completions.

Evaluated on the Google Cluster Trace v3 and the Alibaba

Cluster Trace 2018, RAFP achieves macro-averaged F1-scores of 0.891 and 0.879 respectively, outperforming the strongest baseline by 7.3 and 6.8 percentage points and reducing pipeline disruption events by 37.4% and 34.1% in simulated intervention mode. The ablation studies confirm that the retrieval component provides a genuine and consistent contribution to prediction accuracy, with the RAFP-NoRetrieval variant performing 5.0 and 4.7 percentage points below the full system on the respective traces. The sensitivity analysis of the retrieval depth parameter identifies k=5 as the optimal setting, reflecting the stereotyped nature of cloud failure conditions and the diminishing informational returns of retrieving weakly similar historical incidents. The stratified analysis by node type demonstrates that retrieval augmentation is particularly valuable for specialized hardware configurations where telemetry features alone provide insufficient characterization of the failure risk, a finding with direct implications for cloud operators managing increasingly heterogeneous compute fleets.

Several directions for future investigation merit attention. The current RAFP implementation updates the knowledge base on a daily batch schedule, which may introduce temporal lag in the contextual information available for failure patterns that emerge rapidly following infrastructure changes. Real-time or near-real-time knowledge base updates would address this limitation and improve the system's responsiveness to novel failure modes. Extending the retrieval module to incorporate multi-modal failure evidence — including structured log data, network topology information, and container-level resource metrics alongside the current telemetry feature representation — represents a complementary avenue for improving prediction robustness. The integration of a lightweight generative module capable of synthesizing natural language remediation recommendations from retrieved failure records would further extend RAFP toward an end-to-end fault management system capable of supporting both detection and resolution within a unified operational framework.

References

[1] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., ... & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. Advances in Neural Information Processing Systems, 33, 9459–9474.

[2] He, X., Zhao, K., & Chu, X. (2021). AutoML: A survey of the state-of-the-art. Knowledge-Based Systems, 212, 106622. https://doi.org/10.1016/j.knosys.2020.106622

[3] Notaro, P., Cardoso, J., & Gerndt, M. (2021). A survey of aiops methods for failure management. ACM Transactions on Intelligent Systems and Technology, 12(6), 1–45. https://doi.org/10.1145/3470649

[4] Zhao, W., Chen, T., Yang, J. S., & Qiu, L. (2026). AutoML-Pipeline: A RAG-enhanced code generation framework with pre-validation for cloud-native machine learning workflows. IEEE Access. https://doi.org/10.1109/ACCESS.2026.xxxxxx

[5] Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., ... & Wang, H. (2023). Retrieval-augmented generation for large language models: A survey. arXiv preprint arXiv:2312.10997.

[6] Jassas, M. S., & Mahmoud, Q. H. (2022). Analysis of job failure and prediction model for cloud computing using machine learning. Sensors, 22(5), 2035. https://doi.org/10.3390/s22052035

[7] Tengku Asmawi, T. N., Ismail, A., & Shen, J. (2022). Cloud failure prediction based on traditional machine learning and deep learning. Journal of Cloud Computing, 11(1), 47. https://doi.org/10.1186/s13677-022-00330-8

[8] Tuli, S., Casale, G., & Jennings, N. R. (2022). Tranad: Deep transformer networks for anomaly detection in multivariate time series data. arXiv preprint arXiv:2201.07284.

[9] Han, S., Wu, J., Xu, E., He, C., Lee, P. P., Qiang, Y., ... & Li, R. (2019). Robust data preprocessing for machine-learning-based disk failure prediction in cloud production environments. arXiv preprint arXiv:1912.09722.

[10] Zöller, M. A., & Huber, M. F. (2021). Benchmark and survey of automated machine learning frameworks. Journal of Artificial Intelligence Research, 70, 409–472. https://doi.org/10.1613/jair.1.12876

[11] Feurer, M., Eggensperger, K., Falkner, S., Lindauer, M., & Hutter, F. (2022). Auto-sklearn 2.0: Hands-free automl via meta-learning. Journal of Machine Learning Research, 23(261), 1–61.

[12] Real, E., Liang, C., So, D., & Le, Q. (2020). Automl-zero: Evolving machine learning algorithms from scratch. In International Conference on Machine Learning (pp. 8007–8019). PMLR.

[13] Wever, M. (2024). Automated machine learning for multi-label classification. arXiv preprint arXiv:2402.18198.

[14] Almheiri, Z., Meguid, M., & Zayed, T. (2021). Failure modeling of water distribution pipelines using meta-learning algorithms. Water Research, 205, 117680. https://doi.org/10.1016/j.watres.2021.117680

[15] Jassas, M. S., & Mahmoud, Q. H. (2019). Failure characterization and prediction of scheduling jobs in google cluster traces. In 2019 IEEE 10th GCC Conference & Exhibition (GCC) (pp. 1–7). IEEE. https://doi.org/10.1109/GCC47125.2019.9070259

[16] Riganelli, O., Saltarel, P., Tundo, A., Mobilio, M., & Mariani, L. (2021). Cloud failure prediction with hierarchical temporal memory: an empirical assessment. In 2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA) (pp. 785–790). IEEE. https://doi.org/10.1109/ICMLA52935.2021.9680222

[17] Cotroneo, D., De Simone, L., Liguori, P., & Natella, R. (2020). Fault injection analytics: A novel approach to discover failure modes in cloud-computing systems. IEEE Transactions on Dependable and Secure Computing, 19(3), 1476–1491. https://doi.org/10.1109/TDSC.2020.3013011

[18] Ma, M., Xu, J., Wang, Y., Chen, P., Zhang, Z., & Wang, P. (2020). Automap: Diagnose your microservice-based web applications automatically. In Proceedings of The Web Conference 2020 (pp. 246–258). https://doi.org/10.1145/3366423.3380091

[19] Wu, L., Tordsson, J., Elmroth, E., & Kao, O. (2020). Microrca: Root cause localization of performance issues in microservices. In IEEE/IFIP Network Operations and Management Symposium (NOMS). https://doi.org/10.1109/NOMS47738.2020.9110391

[20] Ding, J., Shen, Z., & Liu, W. (2026). Game-Theoretic Cost-Sensitive Adversarial Training for Robust Cloud Intrusion Detection Against GAN-Based Evasion Attacks. Applied Sciences, 16(8), 3944. https://doi.org/10.3390/app16083944

[21] Ping, W., Jiao, Y., Fan, H., & Zhang, X. (2026). Multimodal Fraud Detection in Financial Statements: A Trimodal Attention Network with Contrastive Evidence Chain Construction. IEEE Access. https://doi.org/10.1109/ACCESS.2026.xxxxxx

[22] Teng, D., Rhee, M., Qin, Y., Zi, B., & Liu, W. (2026). SW-SpeedDLM: Sliding-Window Speculative Decoding for Diffusion Language Models under Long-Context Constraints. Mathematics. https://doi.org/10.3390/mathxxxx

[23] Zhang, F., Guo, Z., Ding, J., Yang, J., & Liu, W. (2026). Adaptive Sensor Fusion for Robust Perception in Extreme Weather: A Gated Vision and LiDAR Integration Framework. Sensors. https://doi.org/10.3390/s26xxxx

[24] Chen, J., Liu, J., Liang, Y., & Zhou, M. (2026). HeteroGCL: A Heterogeneous Graph Contrastive Learning Framework for Scalable and Sustainable Cryptocurrency AML. Applied Sciences, 16(6), 2860. https://doi.org/10.3390/app16062860

[25] Zhang, S., Qiu, L., & Zeng, Z. (2026). Physics-Data Synergy in Structural Health Monitoring: A Multi-Scale Graph Contrastive Framework with Temperature-Adaptive Fusion. IEEE Access. https://doi.org/10.1109/ACCESS.2026.xxxxxx

[26] Liu, C. L., Tseng, C. J., Huang, T. H., Yang, J. S., & Huang, K. B. (2023). A multi-task learning model for building electrical load prediction. Energy and Buildings, 278, 112601. https://doi.org/10.1016/j.enbuild.2022.112601