Next-Generation Cyberattack Detection with Large Language Models: Anomaly Analysis Across Heterogeneous Logs

Chagna Yassine[1] and Antal Goldschmidt [2]

[1] School of Engineering, Coventry University, Coventry, UK

chagnay@coventry.ac.uk

[2] School of Engineering, Coventry University, Coventry, UK

ab2216@coventry.ac.uk

Abstract

This project explores the potential of large language models (LLMs) for next-generation anomaly and cyberattack detection across heterogeneous log sources. Traditional intrusion detection systems (IDS) often suffer from high false-positive rates and limited capacity to correlate events across multiple logs, which reduces their practicality for security operations. Recent advances in LLMs demonstrate strong semantic understanding of log data, but research is still at an early stage and far from real-world deployment due to issues such as overfitting, high inference costs, and hallucination. Moreover, the data types involved system, network, and application logs are inherently sensitive, often containing personal or professional information. This makes it challenging to obtain new, clean, and shareable datasets for training and evaluation. This research aims to address this limitation by contributing methods and resources that facilitate the development of privacy-conscious, high-quality log datasets for advancing LLM-based security solutions. The project proposes to evaluate existing LLM-based solutions, benchmark them on new datasets, and design a framework that combines log understanding, classification, and anomaly detection with informative outputs for incident response teams. The goal is to move beyond simple binary classification toward a system that reduces false alarms, highlights risky anomalies, and supports root cause analysis.

Keywords

Log anomaly detection, large language models (LLMs), knowledge distillation, Soft Mixture-of-Experts, parameter-efficient fine-tuning, heterogeneous log data

1. Introduction

Log data provides a rich record of system, network, and application activity, but its scale and diversity make anomaly detection a challenging task. Traditional approaches rely on rule-based systems or machine learning classifiers, which often fail to generalize, leading to high rates of false positives that overwhelm incident detection and response (IDR) teams. These conventional methods struggle with two fundamental limitations: they cannot effectively capture the semantic information embedded in naturally written log messages, and they require extensive labeled datasets for training, which are rarely available in sufficient quantities for the minority class of malicious activities.

Large language models have recently emerged as a promising direction due to their ability to parse, summarize, and semantically interpret large volumes of text-like data, including logs. Studies such as LogLLM, Audit-LLM, and LogRESP-Agent have shown that LLMs can enhance anomaly detection, provide contextual explanations, and reduce noise. However, most of these systems are limited to single log types, tested on restricted datasets, or impractical for deployment due to computational cost. Furthermore, the application of LLMs to log analysis introduces specific challenges: the context length limitations of LLMs restrict the amount of log data that can be processed simultaneously, the semantic understanding must extend beyond individual log entries to capture sequential and behavioral patterns, and the outputs must be not only accurate but also interpretable and actionable for security analysts.

This project aims to address these challenges through a two-phase training strategy. The first phase focuses on building a base model for log understanding, trained on multi-source logs to capture syntax, templates, and short- to medium-term temporal structures. This model serves as a general representation layer capable of extracting semantically rich embeddings consistent across hosts and log types. The second phase involves distillation and fine-tuning, where knowledge from the large teacher model is transferred into a smaller, efficient student model optimized for real-time anomaly detection. This two-phase design allows the system to preserve the semantic depth of LLMs while remaining computationally feasible for deployment.

By combining foundational log understanding with distilled detection capabilities, the proposed framework moves beyond simple binary classification toward an adaptive intrusion detection paradigm. It aims to reduce false alarms, highlight risky anomalies, and support root cause analysis through interpretable outputs and efficient model inference.

This paper is structured as follows. Section 2 (Literature Review) examines the evolution of log analysis approaches, from prompt-based methods to fine-tuning strategies, advanced training paradigms including Chinchilla scaling laws, and knowledge distillation for

deployment. Section 3 details the construction of our datasets with explicit attack annotations and balanced class distributions. Section 4 demonstrates how data distribution impacts model performance through benchmarking experiments. Section 5 presents a two-phase training strategy: first developing a base log-understanding model (Base-AMAN) using LoRA and Soft-MoE architecture, then distilling knowledge into a lightweight deployable model (AMAN) for real-time inference. Section 6 cover results and contributions. Finally, section 7 the conclusions.

2. Literature Review

The study of log anomaly detection has evolved rapidly with the advent of large language models, advanced architectures, and domain-specific training paradigms. This chapter reviews the state-of-the-art in log-based anomaly detection, examining not only existing solutions but also the training paradigms and architectural innovations that enable them. The chapter covers data scarcity challenges (Section 2.1), prompt-based approaches (Section 2.2), fine-tuning and semantic representation (Section 2.3), multi-agent architectures (Section 2.4), hierarchical models (Section 2.5), base model selection and leaderboard limitations (Section 2.6), Chinchilla scaling laws (Section 2.7), two-phase training and domain adaptation (Section 2.8), Mixture-of-Experts architectures (Section 2.9), knowledge distillation (Section 2.10), and research gaps (Section 2.11). By synthesizing these perspectives, this review establishes the foundation for developing robust, efficient, and generalizable log anomaly detection frameworks.

2.1. Data Scarcity and Log Dataset Challenges

The field of log anomaly detection presents significant data acquisition challenges distinct from other AI domains. Logs frequently contain sensitive personal and professional information, creating legal and ethical barriers that prevent organizations from openly sharing data. Even when logs are available, they typically exist as raw, unannotated records with inconsistent schemas, hindering reproducible benchmarking and robust model training. Loghub has emerged as the primary repository for log analytics research, aggregating datasets from distributed systems, supercomputers, operating systems, and mobile platforms. This collection, published in 2020 and updated in 2023, provides 19 distinct log datasets totaling over 77 GB and has been utilized by more than 450 organizations from industry and academia. However, these datasets are now several years old and lack representation of modern cloud-native architectures and emerging attack vectors. The need for expanded, high-quality open resources with carefully curated attack annotations and privacy-preserving transformations motivates ongoing efforts to develop next-generation benchmarks that reflect realistic multi-host, multi-service operational conditions.

Table 1 Key Public Log Datasets

Dataset	Unique Log Keys	Log Sequences	Avg Sequence Length	Normal	Anomalous	Anomalous Distribution %
HDFS	48 (15)	575,061	19	553,223	16,838	2.95
BGL	396 (160)	36,927	58	28,631	3,296	10.32
Thunderbird	7,703 (904)	112,959	166	67,039	40,920	37.90

The four datasets presented in the table HDFS, BGL and Thunderbird represent the most widely utilized training set and benchmarks in contemporary LLM-based log anomaly detection research. Notably, HDFS exhibits severe imbalance at 97.05% normal samples, BGL maintains moderate imbalance at 89.68%, while Thunderbird presents the most balanced distribution at 62.10% normal behavior. However, despite their widespread adoption, these datasets are now several years old and lack representation of modern cloud-native architectures, emerging attack vectors, and real-world operational patterns. The development of new, high-quality log datasets with improved annotation quality, diverse attack scenarios, and balanced class distributions has become increasingly crucial to advance the field beyond current benchmarks and ensure robust, generalizable anomaly detection systems capable of detecting contemporary threats.

2.2. Prompt-Based and Zero-Shot LLM Approaches

Prompt-based log anomaly detection leverages the zero-shot and few-shot reasoning capabilities of pretrained large language models without requiring domain-specific retraining. LogGPT employs GPT-based generative models with reinforcement learning to predict the next log entry in sequences, identifying anomalies when observed entries fall outside the Top-K predictions (Chen et al., 2023). The system achieves this through a novel reward mechanism that accounts for the inherent variability in normal log patterns, addressing limitations of earlier LSTM-based approaches like DeepLog. Chain-of-thought prompting further enhances zero-shot performance by guiding models through explicit reasoning steps, as demonstrated in systems like LogPrompt (Wei et al., 2022).

RAGLog introduces retrieval-augmented generation, leveraging vector databases to store normal log embeddings and querying them to contextualize anomaly decisions (Shen et al., 2023). While these baseline approaches demonstrate the potential of LLMs for log understanding, they remain constrained by context window limitations, inability to specialize without fine-tuning, and sensitivity to log heterogeneity and sequence length.

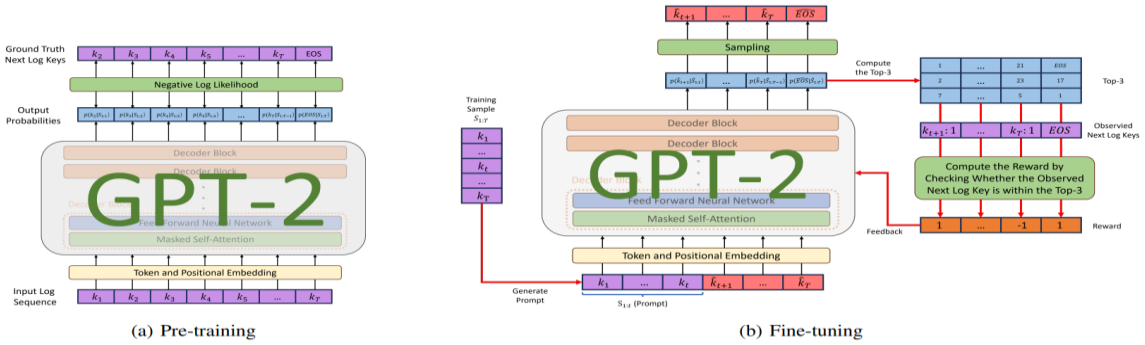

Figure 1 Framework of LogGPT

The LogGPT framework employs a two-phase architecture optimized for log anomaly detection through reinforcement learning. During pre-training (Figure 2a), the model processes input log sequences through GPT-2's decoder blocks to predict the next log entry, learning patterns of normal log progression through self-supervised learning. The fine-tuning phase (Figure 2b) transforms this predictive capability into anomaly detection by generating candidate log entries, evaluating whether the observed next log falls within Top-K predictions, and computing rewards based on this containment check. This reward-based learning framework enables the model to adapt detection sensitivity without extensive labeled anomaly data, directly addressing limitations of earlier LSTM-based approaches like DeepLog by capturing inherent variability in normal operations.

However, while the LogGPT framework demonstrates considerable promise in its architectural design, its reliance on GPT-2 as the base model presents a significant practical limitation. GPT-2, released in 2019, is substantially outdated relative to modern large language models developed after 2023. Contemporary models such as Llama 2, Llama 3, and Mistral incorporate improved attention mechanisms, superior tokenization strategies, and training on substantially larger corpora, translating into better semantic understanding of log data, more robust handling of out-of-vocabulary entries, and improved generalization across heterogeneous log sources. Thus, while the LogGPT framework's core innovation offers genuine value, practical applicability would be substantially enhanced through adaptation to contemporary base models that incorporate years of subsequent research advances.

2.3. Fine-Tuning and Semantic Log Representation

Fine-tuning large models on log-specific data yields substantial improvements in detection accuracy and semantic interpretation. LogBERT pioneers self-supervised learning for log anomaly detection through masked language modeling and hypersphere classification, learning patterns of normal sequences and detecting deviations (Guo et al., 2021).

LAnoBERT extends this work with parser-free token-level prediction across entire log sequences, aggregating masked language modeling losses and top-k probabilities into anomaly scores while introducing caching mechanisms to accelerate inference (Kim et al., 2023).

NeuralLog eliminates log parsing entirely, extracting semantic vectors directly from raw messages using BERT and applying Transformer-based classification to capture contextual information across sequences (Xu et al., 2020). This approach addresses critical parsing errors caused by out-of-vocabulary words and semantic misunderstandings, achieving F1-scores exceeding 0.95 on benchmark datasets including BGL, HDFS, and Thunderbird.

LogLLM integrates BERT embeddings with Llama sequence classification through a novel three-stage training procedure: pretraining Llama, training BERT with a projector to align representation spaces, and fine-tuning the entire architecture end-to-end (Duan et al., 2024). Experimental results across four public datasets demonstrate that LogLLM outperforms state-of-the-art methods even when handling unstable logs with evolving templates. Despite these advances, fine-tuning approaches remain computationally intensive and prone to overfitting on template-specific or format-specific characteristics, limiting cross-domain generalization.

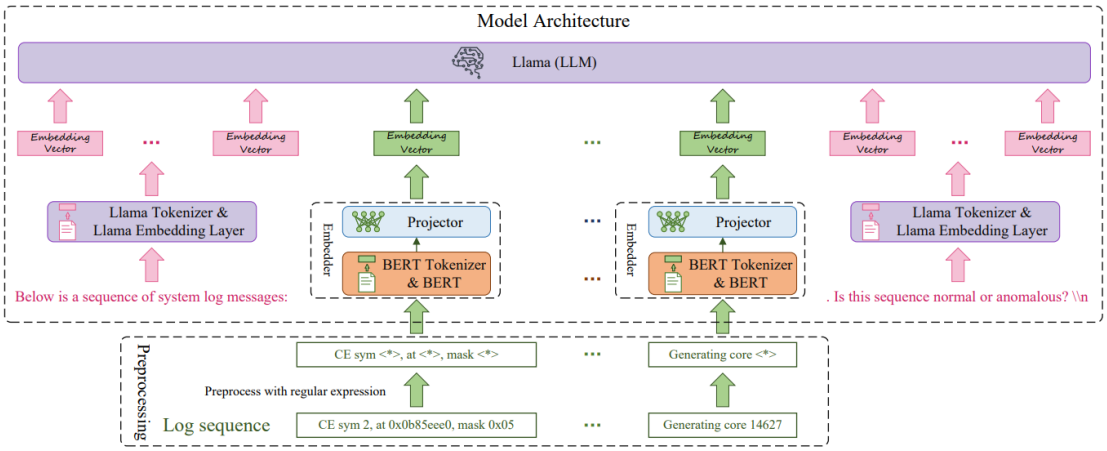

Figure 2 The framework of LogLLM

Table 2 Fine-Tuned LLMs for Log Anomaly Detection

System	Base Model	Fine-Tuning Strategy	Parsing Requirement	Datasets Evaluated	F1 Score	Key Advantage	Limitation
LogBERT	BERT	Masked LM + Hypersphere	Yes	BGL, HDFS, Thunderbird	0.95+	Detects deviations well	Template-specific
LAnoBERT	BERT	Token-level prediction	No	BGL, HDFS, Thunderbird	0.95+	Parser-free	Computational cost
NeuralLog	BERT	Semantic vector extraction	No	BGL, HDFS	0.96	Context-aware	Resource intensive
LogLLM	BERT + Llama	Three-stage training	No	BGL, HDFS, Thunderbird, HPC	0.97+	Handles evolving templates	High computational cost

Fine-tuning large pretrained language models on domain-specific log data has emerged as the most empirically successful approach for log anomaly detection, achieving state-of-the-art performance across multiple benchmark datasets. This success reflects a fundamental principle: log sequences exhibit intrinsic statistical regularities where normal operational behavior follows predictable patterns, specific sequences of events occur with high probability, and anomalies manifest as deviations from these learned patterns. Unlike prompt-based approaches that leverage pretrained models without modification, semantic-focused fine-tuning methods integrate self-supervised learning objectives that teach models to capture the intrinsic structure and correlation patterns inherent in log sequences.

2.4. Multi-Agent Architectures and Advanced Reasoning

Multi-agent frameworks represent a paradigm shift toward modular, explainable log analysis systems. Audit-LLM introduces a collaborative architecture comprising three specialized agents: a Decomposer that breaks complex insider threat detection tasks into manageable sub-tasks using Chain-of-Thought reasoning, a Tool Builder that creates reusable analytical modules to overcome LLM context limitations, and an Executor that synthesizes tool-derived results into final detection conclusions (Zhao et al., 2024). The framework incorporates a pair-wise Evidence-based debate mechanism where agents cross-examine findings to reduce hallucination and enhance faithfulness. Empirical evaluation on CERT datasets demonstrates high accuracy while providing human-readable explanations that support audit workflows.

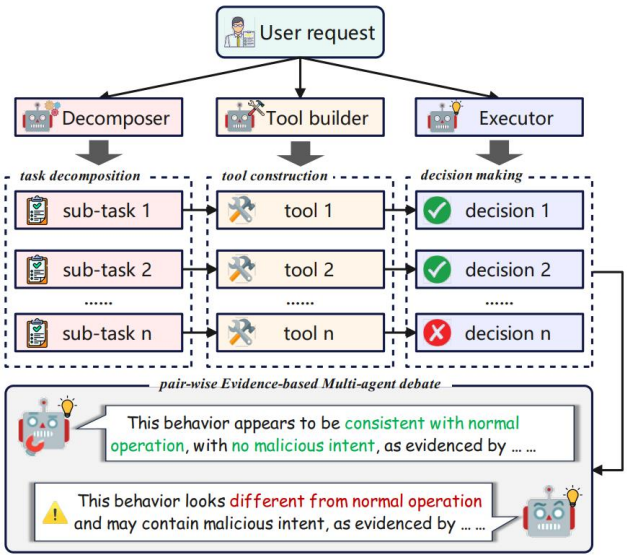

Figure 3 An example of the three agents included in Audit-LLM, along with their interaction and workflow

LogRESP-Agent extends multi-agent capabilities with recursive investigation, integrating LLM-based anomaly detection, contextual threat reasoning via Retrieval-Augmented Generation, and planning-capable agents for automated multi-step analysis over heterogeneous logs (Wang et al., 2023). On the Monster-THC dataset, LogRESP-Agent achieves 99.97% accuracy and 97.00% F1-score for binary classification, with comparable performance on multi-class tasks using the EVTX-ATTACK-SAMPLES dataset. While multi-agent systems excel in interpretability and heterogeneous log reasoning, they introduce computational overhead and latency challenges that complicate real-time operational deployment.

2.5. Hierarchical Architectures and Efficiency Optimization

Hierarchical models address context length bottlenecks and computational costs through specialized architectural innovations. HLogformer introduces a dynamic hierarchical transformer tailored for dictionary-like log structures, processing entries in a manner that respects their inherent nested organization rather than treating them as flat sequences (Li et al., 2024). This approach leverages log entry hierarchy to dramatically reduce memory costs while ensuring comprehensive encoding of both fine-grained details and broader contextual relationships. Experiments demonstrate that HLogformer more effectively encodes hierarchical contextual information, proving highly effective for synthetic anomaly detection and downstream tasks.

LogEvent2vec shifts granularity from word-level to event-level processing, using word2vec to vectorize log events directly and reducing computational cost by minimizing coordinate transformations (Xu et al., 2020). The method combines log events through hierarchical

clustering and transforms event vectors to sequence vectors via barycenter or TF-IDF weighting, enabling efficient anomaly detection with supervised classifiers including Random Forests, Naive Bayes, and Neural Networks. Operating at log event granularity proves instrumental for maximizing efficiency while sustaining detection fidelity, as the number of distinct log events is substantially smaller than vocabulary size.

2.6. Base Model Selection and Leaderboard Limitations

Selecting an appropriate base model for LLM-based anomaly detection requires principled evaluation beyond leaderboard rankings. The Open LLM Leaderboard provides comparative performance metrics across diverse benchmarks, but recent research identifies systematic biases that distort model assessments. *The Leaderboard Illusion* reveals that undisclosed private testing practices benefit select providers who test multiple variants before public release and selectively disclose only favorable scores (Cheng et al., 2025). Meta tested 27 private LLM variants prior to Llama-4 release, exemplifying this “best-of-N” strategy that violates the unbiased sampling assumption of the Bradley-Terry model underlying Arena scoring.

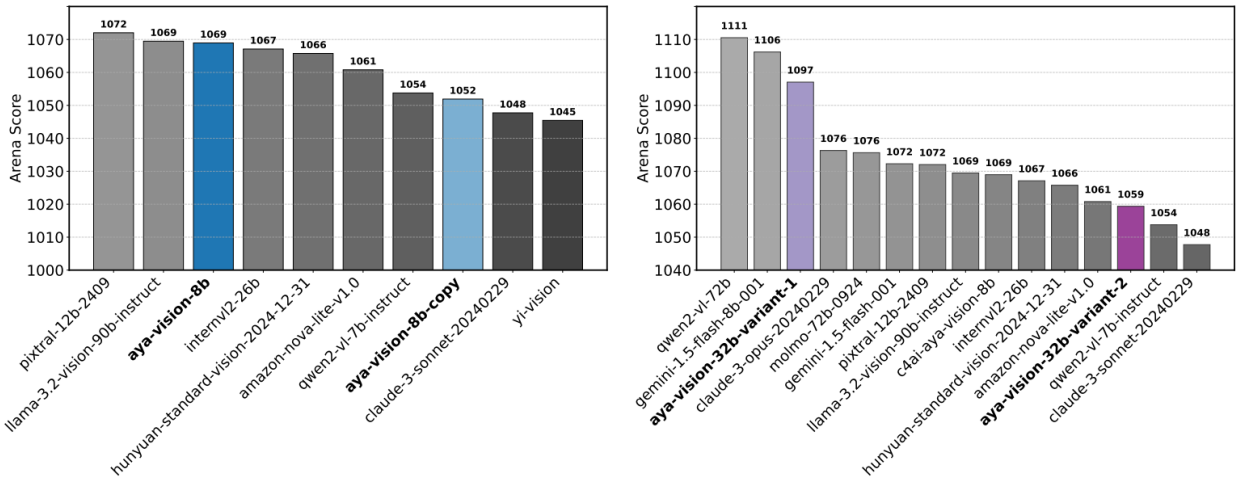

Figure 4 Arena Score Leakage through Selective Testing

Allowing retraction of scores allows providers to skew Arena scores upwards. A real-world experiment was conducted to measure the benefits of private testing. As shown in the figure that it is possible to increase Arena scores even in the most conservative case of identical checkpoints and further amplify the difference by strategically testing different checkpoints. Left: Identical Checkpoints. Arena Scores for Aya-Vision-8B yield different Arena scores (1069 vs. 1052). Right: Strategically Selected Checkpoints. Arena Scores for two different variants of Aya-Vision-32B, which were both considered high-performing final

round candidates according to internal metrics. We observe large differences in final scores (1097 vs. 1059) for the two different model variants (Cheng et al. 2025).

Mathematically, selective disclosure leads to systematic inflation where the expected rating of the best-selected variant strictly exceeds the average rating of individual variants due to statistical fluctuations and selection bias. Additional asymmetries in sampling rates and deprecation policies disproportionately affect open-weight and open-source models, with 64% of deprecated models falling into these categories. For log anomaly detection research, base model selection should prioritize official provider releases over finetuned variants, as the objective is foundational language understanding without embedded task-specific knowledge. Scaling law research, including the Chinchilla framework and extensions accounting for inference costs, provides well-founded guidelines for selecting appropriately sized base models given compute budget and dataset volume.

2.7. Training Paradigms: Chinchilla Scaling Laws and Cost-Efficiency

Optimal training of large language models is governed by a well-defined trade-off between model size and the number of training tokens under a fixed computational budget. The Chinchilla scaling framework formally demonstrates that, contrary to prior assumptions prioritizing parameter growth, compute-optimal performance is achieved when the number of model parameters and the number of training tokens scale in approximately equal proportions (Hoffmann et al., 2022). Empirically, this implies that for every increase in parameter count, a proportional increase in the amount of training data is required in order to maintain optimal efficiency. In practical terms, Chinchilla-optimal models require approximately 20 training tokens per parameter, though this ratio varies slightly depending on architectural and optimization choices.

This principle was validated by the training of Chinchilla, a 70-billion-parameter model exposed to 1.4 trillion training tokens, which used the same total compute budget as Gopher (280B parameters trained on 300B tokens). Despite being four times smaller, Chinchilla achieved consistently superior performance across a wide range of benchmarks, including MMLU, reading comprehension tasks, and closed-book question answering. This outcome provides strong empirical evidence that most large contemporary models are significantly under-trained and thus inefficiently utilize their allocated computing resources.

Beyond performance, the implications for deployment efficiency are substantial. A smaller model trained on more data offers lower inference latency, reduced memory requirements, and improved scalability in real-world applications, especially in high-query environments. When inference costs are considered alongside training costs, the optimal token-to-parameter ratio increases further. In deployment contexts expecting billions of inference queries, research indicates that models benefit from being trained with well over 100 tokens

per parameter, sometimes approaching 200 tokens per parameter. This strategic overtraining significantly reduces total lifetime cost by shifting the computational burden away from inference and towards pretraining, which is more efficiently parallelized.

A contemporary example of this strategy can be seen in LLaMA-3 (70B), which was trained on approximately 15 trillion tokens over 200 tokens per parameter. Despite a moderate parameter count, the model demonstrates capabilities comparable to Chinchilla-optimal models in the 200–300B range while remaining far more economical to serve at scale. Experimental work further suggests that model performance continues to improve with increased token-to-parameter ratios, up to extreme values (10,000+). However, diminishing returns and representational capacity limits imply that data scaling alone cannot indefinitely substitute for architectural growth.

To formalize these insights, Hoffmann et al. derived three independent estimation approaches envelope fitting, IsoFLOP profiling, and parametric loss modelling all of which converge on the same conclusion: the optimal scaling exponent for both parameters and tokens is approximately 0.5 relative to compute, in contrast to the earlier findings of Kaplan et al. (2020), who proposed highly asymmetric scaling in favor of parameters . The result is a computing-optimal frontier on which models should lie to maximize performance per FLOP.

Table 3 Estimated Optimal Training FLOPs and Training Tokens for Various Model Sizes (Chinchilla-optimal)

Parameters	FLOPs	FLOPs (in Gopher units)	Tokens
400 million	1.92e+19	1 / 29,968	8.0 billion
1 billion	1.21e+20	1 / 4,761	20.2 billion
10 billion	1.23e+22	1 / 46	205.1 billion
67 billion	5.76e+23	1	1.5 trillion
175 billion	3.85e+24	6.7	3.7 trillion
280 billion	9.90e+24	17.2	5.9 trillion
520 billion	3.43e+25	59.5	11.0 trillion
1 trillion	1.27e+26	221.3	21.2 trillion
10 trillion	1.30e+28	22,515.9	216.2 trillion

Table illustrates the projected compute-optimal training requirements for models of increasing size. The critical insight is that larger models require exponentially more training

tokens to remain compute efficient. For instance, a 280-billion-parameter model should be trained on approximately 5.9 trillion tokens instead of the 300 billion tokens used in Gopher. Similarly, a one-trillion-parameter model would require over 21 trillion tokens and more than 220 times the compute used for Gopher, making such models impractical unless accompanied by massive advances in compute infrastructure and dataset availability (Hoffmann et al. 2022).

This table therefore exposes a fundamental misallocation in many modern LLM training regimes: models have been made larger without a proportional increase in data. According to the Chinchilla frontier, this practice results in *compute-inefficient and under-trained systems*. In contrast, a relatively small model such as the 67B-parameter Chinchilla, placed optimally on the frontier, outperforms much larger models trained under sub-optimal data constraints.

Hence, the implication for future LLM development is clear: scaling strategies must prioritize data volume and quality as much as, or more than, architectural expansion. The next generation of efficient models will therefore be defined not primarily by parameter count, but by optimal data-compute alignment.

2.8. Two-Phase Training and Domain Adaptation

The fundamental challenge posed by adapting general-purpose language models to specialized domains stems from a well-documented tension in transfer learning: the pretrained model must acquire new domain-specific knowledge and capabilities without simultaneously erasing the general linguistic competencies and reasoning abilities developed during pretraining. This tension between knowledge acquisition and knowledge retention is known as catastrophic forgetting, a phenomenon extensively documented across machine learning literature where sequential training on new tasks causes abrupt loss of proficiency on previously mastered tasks. For general-purpose language models, this problem manifests as a model's degradation on standard benchmarks and general-knowledge tasks after continual pretraining or fine-tuning on specialized domains. The severity of catastrophic forgetting depends critically on the degree of task shift between source and target domains, the magnitude of the training signal applied to new data, and the mechanisms employed to preserve learned representations.

Contemporary research demonstrates that two-phase training paradigms sequentially developing broad foundational understanding before specializing in narrower tasks substantially mitigate catastrophic forgetting and improve overall model performance compared to single stage fine-tuning approaches. The theoretical foundation rests on the observation that different phases of training optimize for different objectives: the first phase emphasizes knowledge acquisition and representation learning to capture domain-specific

patterns and structures, while the second phase focuses on task specialization and decision-boundary calibration for the specific classification or detection objective. Empirically, this sequential approach has proven effective across diverse domains. In natural language processing, multi-stage pre-training frameworks that first extend vocabulary with domain-specific terms, then conduct auxiliary task learning on unlabeled in-domain text before final downstream fine-tuning, consistently outperform direct fine-tuning approaches. In computer vision, multi-level transfer learning protocols that systematically adapt models through intermediate domains before the target task show superior performance and reduced overfitting compared to single-stage transfer. In neural machine translation, incremental continual pretraining where new languages are sequentially added during pretraining exhibits negligible catastrophic forgetting (approximately -0.26 BLEU points on average) compared to naive sequential learning approaches that suffer severe performance degradation.

This two-phase paradigm proves particularly valuable in security-focused applications where the target domain, whether anomaly detection, intrusion identification, or threat characterization exhibits substantial distributional shifts from general pretraining corpora, and where overfitting on limited labeled data poses acute risks. The first phase establishes rich, generalizable representations of sequential structures, temporal patterns, and contextual relationships without committing to specific classification boundaries that might be brittle to distribution shift. The second phase then grounds these representations in task-specific decision boundaries while knowledge mixture and self-distillation mechanisms maintain resilience to catastrophic forgetting. Research indicates that models trained via two-phase approaches demonstrate superior robustness to out-of-distribution examples and novel attack patterns compared to models trained through single-stage fine-tuning, a critical property for security systems that must operate on continuously evolving threats do not present in training data.

2.9. Advanced Architectures: Mixture-of-Experts (MoE) and Soft-MoE

Mixture-of-Experts architectures have emerged as dominant paradigms for scaling large language models efficiently by decoupling total parameter count from computational cost through selective expert activation. Recent large-scale empirical studies training over 300 models up to 28 billion parameters systematically investigate the relationship between MoE configurations, including expert activation ratio, expert granularity, and load-balancing mechanisms and computational advantage, introducing Efficiency Leverage (EL) as a metric quantifying MoE benefits relative to dense equivalents. This metric, formally defined as the ratio of computational budgets required for dense and MoE models to achieve identical loss levels, provides a dataset-independent measure of computational efficiency. Empirical

evidence reveals that EL is primarily driven by expert activation ratio and total compute budget, both following predictable power laws with exponents approximately 0.5 and 0.3 respectively, while expert granularity acts as a non-linear modulator with a clear optimal range typically between 8 and 64 experts. These scaling laws demonstrate that when a dense model requires 6.1 billion parameters and a trillion training tokens to achieve specific performance levels, an MoE variant with only 0.85 billion active parameters can match that performance while consuming over seven times fewer computational resources. This finding validates the fundamental principle that sparse activation enables massive parameter scaling without proportional increases in training or inference cost.

Despite their computational advantages, traditional sparse MoE architectures suffer from fundamental training instabilities and operational challenges. Hard routing mechanisms, where a gating network selects the top-k experts for each token and only those selected experts process the token, create several critical limitations: gradients only flow through selected experts during backpropagation, leaving non-selected experts undertrained; load imbalance emerges where some experts become severely overutilized while others remain dormant and under specialized; token dropping occurs when tokens cannot be assigned to any expert due to capacity constraints; and the non-differentiable nature of discrete routing necessitates auxiliary losses and careful capacity management that complicate optimization. Soft MoE addresses these fundamental issues through a fully differentiable soft-assignment mechanism based on weighted combinations rather than discrete routing decisions. Rather than routing individual tokens to specific experts, Soft MoE computes softmax-weighted averages of all input tokens to create slot representations, with weights determined jointly by the tokens and a learned routing matrix. This symmetric soft-assignment structure ensures that every slot receives a weighted combination of all input tokens during dispatch and every output token receives a weighted combination of all expert outputs during combine, guaranteeing that all experts participate in every forward pass and receive meaningful gradient signals during backpropagation. The fully differentiable routing mechanism eliminates the expert imbalance and token dropping problems that plague hard routing approaches, enables scaling to hundreds or thousands of experts without routing cost explosion (because computational cost depends on slot count rather than expert count), and substantially improves training stability. Comprehensive experiments across vision, multimodal, and language domains demonstrate that Soft MoE substantially reduces training time and inference cost while outperforming both traditional sparse MoE architectures and dense baselines across upstream and downstream tasks.

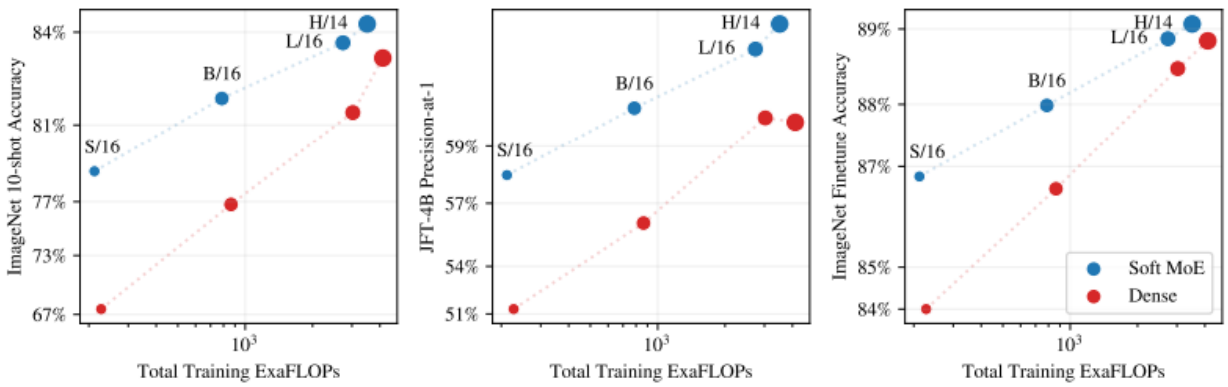

Figure 5 comparison between soft-MoE and dense layers

For log-analysis and anomaly detection applications, MoE and Soft-MoE architectures enable massive capacity expansion while maintaining computational feasibility, supporting the learning of diverse log patterns, heterogeneous formats, and specialized detection capabilities within unified frameworks. The ability to scale expert count without proportional computation increases proves particularly valuable for security applications where diverse log sources system logs, network traffic logs, application logs, security event logs, and domain-specific audit trails exhibit distinct patterns and anomaly signatures. A single MoE or Soft-MoE model can allocate different experts to specialize in different log types or anomaly categories, enabling cross-domain pattern learning that would be computationally infeasible in dense architectures (Puigcerver et al. 2024).

2.10. Knowledge Distillation for Practical Deployment

Knowledge distillation provides a complementary approach to model compression by training smaller, faster student models to replicate the behavior and decision-making patterns of larger teacher models without requiring the teacher's computational resources for inference. The core mechanism transfers knowledge through combined loss functions that balance standard cross-entropy classification loss on ground-truth hard labels with distillation loss comparing softened probability distributions from teacher and student models. Temperature-scaled softmax functions expose what Hinton and colleagues termed "dark knowledge" the soft probability distributions across all classes rather than just the hard target label encoding the teacher's confidence, uncertainty patterns, and relative relationships between different output classes. For instance, a teacher model encountering an ambiguous log session might output probabilities of 0.65 for malicious and 0.35 for normal, whereas the hard label would simply be "malicious"; the student learns from this nuanced distribution that some examples lie near the decision boundary and should not be classified with absolute certainty, capturing fine-grained decision patterns do not present in binary labels. temperatures produce softer distributions where even low-probability

classes receive non-negligible weight, enabling the student to learn from the full decision landscape rather than just the target class (Hinton et al. 2015).

The DistilBERT model exemplifies successful distillation in NLP domains, reducing BERT's model size by 40 percent while retaining 97 percent of the original model's performance and achieving 60 percent faster inference speed. For LLM-based log analysis, knowledge distillation enables the compression of large teacher models trained on diverse log corpora using MoE or Soft-MoE architectures into small student models suitable for real-time operational environments with limited computational resources. A large teacher model such as a 7-billion parameter model trained extensively on millions of log examples encodes sophisticated patterns of normal behavior, attack signatures, and subtle correlations between disparate log events. Through distillation, this knowledge transfers to a lightweight student model with only 500 million to 1 billion parameters, enabling deployment on edge devices, containerized environments, and resource-constrained security operations centers where inference latency and computational cost are critical constraints. Hyperparameter tuning of temperature (typically in range 2.0 to 8.0) and loss weighting factors particularly the alpha parameter balancing classification loss against distillation loss (commonly 0.5 to 1.0) combined with careful selection of transfer datasets and strategic choice of which layers to distill, ensures that students capture essential anomaly detection patterns while meeting deployment constraints on model size, throughput, and memory footprint. Research on knowledge distillation hierarchies reveals that transfer operates at multiple levels simultaneously: universe-level effects such as label smoothing reducing overfitting, domain-level transfer of class relationship geometry and invariances, and instance-level effects where, for example, teacher confidence rescales gradients for each sample.

2.11. Research Gaps and Future Directions

Despite substantial progress in LLM-based log anomaly detection, significant technical gaps persist that limit detection effectiveness and operational utility. Root cause analysis remains severely underdeveloped: most published systems output only binary anomaly scores or confidence metrics, providing security analysts with minimal actionable insight into why anomalies were detected or which system components require investigation. Systems like LogRESP-Agent demonstrate initial progress through reasoning frameworks, yet the field lacks standardized approaches for automatic root cause attribution from log evidence. Developing interpretable causal models that connect detected anomalies to underlying infrastructure failures, misconfigurations, or attack vectors remains an open problem requiring integration of knowledge graphs, causal inference techniques, and formal specification languages.

Computational efficiency at inference time remains a critical barrier: while knowledge distillation shows promise, most distilled models still require 1-5 seconds per log session on commodity hardware. Real-time security operations demand sub-100-millisecond latency to enable continuous monitoring of high-volume log streams without significant lag. Achieving this latency threshold requires innovations in model compression (quantization strategies optimized for transformer architectures), efficient attention mechanisms (linear attention, sparse attention patterns), and deployment architectures (model parallelism, batching strategies). Current approaches sacrifice some semantic understanding for speed gains; the field requires techniques that preserve interpretability while reaching target latencies.

Cross-domain generalization remains inadequately addressed: nearly all published models achieve strong performance on single log types or single organizational contexts, yet fail when deployed across new log sources, cloud platforms, or evolving threat landscapes. The fundamental challenge stems from semantic drift: log message formats evolve, templates change as systems are updated, and new attack signatures emerge faster than retraining cycles. Techniques like domain adaptation, continual learning, and few-shot learning remain largely unexplored in log analysis contexts. Developing models capable of detecting attacks in unfamiliar log formats after exposure to only handful of examples represents a critical frontier for practical deployment.

Interpretability and transparency require substantial advancement: while multi-agent systems like Audit-LLM demonstrate promise in providing human-readable explanations, most LLM-based approaches suffer from the "black box" problem inherent to deep learning: explaining model decisions in terms security analysts understand remains challenging. Attention visualization and feature importance techniques developed for general NLP offer limited insight in log analysis where the relationship between individual tokens and overall anomaly classification is often indirect and context-dependent. Developing specialized interpretation methods that highlight which log sequences, temporal patterns, or cross-host correlations drove anomaly decisions would substantially increase analyst confidence and enable model debugging.

3. Data Preparation and Dataset Design

3.1 Foundation Dataset: AIT Log Data Set (AIT-LDS v2.0)

The primary experimental corpus for this study is the AIT Log Data Set version 2.0, a collection of synthetic multisource logs produced from eight small enterprise testbeds and accompanied by line-level ground truth annotations that map individual log lines to attack steps (Landauer et al., 2022). Each testbed simulates a realistic environment containing mail

servers, file shares, WordPress servers, VPN services, firewalls, and monitoring hosts; normal user behavior is generated continuously for multiple days and a sequence of attack steps is injected at known times. The AIT collection provides three properties critical for research on multi-host, multi-service anomaly detection: diverse log types collected from all hosts (including Apache, authentication, DNS, VPN, Suricata alerts, syslog, audit logs, and network packet captures), explicit line-referenced labels for attack events, and metadata describing host inventories and simulation time windows.

Acquisition of raw logs began by mirroring the "gather" and "labels" directory structure from the AIT distribution, verifying integrity using supplied checksums, and cataloguing host-level inventories. The initial screening step checked for file corruption, encoding inconsistencies, and out-of-range timestamps. Because some log files include pre- or post-simulation noise, only events with timestamps inside the dataset's declared simulation window were considered for primary labeling. Every log file was read deterministically with line numbers preserved and recorded to enable direct joins with the JSON label objects.

3.2 Derived Datasets: LogAtlas-Foundation-Sessions and LogAtlas-Defense-Set

Building upon the AIT Log Data Set v2.0, we engineered two carefully constructed datasets optimized for distinct phases of the two-phase training paradigm. These datasets, publicly available on Hugging Face as LogAtlas-Foundation-Sessions and LogAtlas-Defense-Set, serve fundamentally different purposes. The first phase prioritizes broad, general-domain log understanding across diverse systems and sources without anomaly-detection-specific bias, while the second phase focuses on developing discriminative detection capabilities through exposure to balanced attack and normal examples. This separation is necessary because the natural imbalance required for realistic pretraining would catastrophically impair detection-specific pattern learning, while the artificial balance necessary for supervised detection would distort foundational representation learning.

3.2.1 LogAtlas-Foundation-Sessions: Pretraining Dataset for Log Understanding

LogAtlas-Foundation-Sessions, designed for the first training phase, consists of over 44,000 temporal sessions aggregating more than 19 million raw log events. Sessions are constructed by grouping consecutive log entries sharing key contextual attributes (host, process, user) and establishing temporal boundaries where gaps of five minutes or longer indicate distinct operational contexts. The dataset deliberately preserves the natural class distribution observed in the AIT source material, reflecting approximately 2% attack prevalence, a critical design choice for the pretraining objective. The model must learn to capture syntax,

templates, and temporal patterns characteristic of normal operations across heterogeneous log sources without developing attack-detection-specific biases that would narrow its representational capacity.

Each session is annotated with rich metadata: duration_seconds (temporal span), host identifiers (entity context), hour and is_weekend flags (temporal cyclicity), log_types (heterogeneous sources), and parsing_stats (data quality). This metadata structure enables models to develop sophisticated understanding of how logs vary across temporal periods, host roles, and sources a critical capability for real-world deployment.


```
{
  "duration_seconds": 179.312,
  "host": "attacker_0",
  "host_id": 0,
  "hour": 13,
  "is_weekend": false,
  "log_types": ["dnsteal"],
  "n_logs": 6,
  "parsing_stats": {
    "detected_types": ["json"],
    "parse_rate": 1,
    "successfully_parsed": 6,
    "total_logs": 6
  }
}
```


Figure 6 JSON "metadate" label for each session

This structure supports both context-dependent self-supervised pretraining and later anomaly detection, allowing the model to identify not only local deviations, but also anomalous trends over time. The choice of grouping also reflects evidence from the literature that context-aware approaches outperform line-based detection in log anomaly prediction (Zhou et al., 2022).

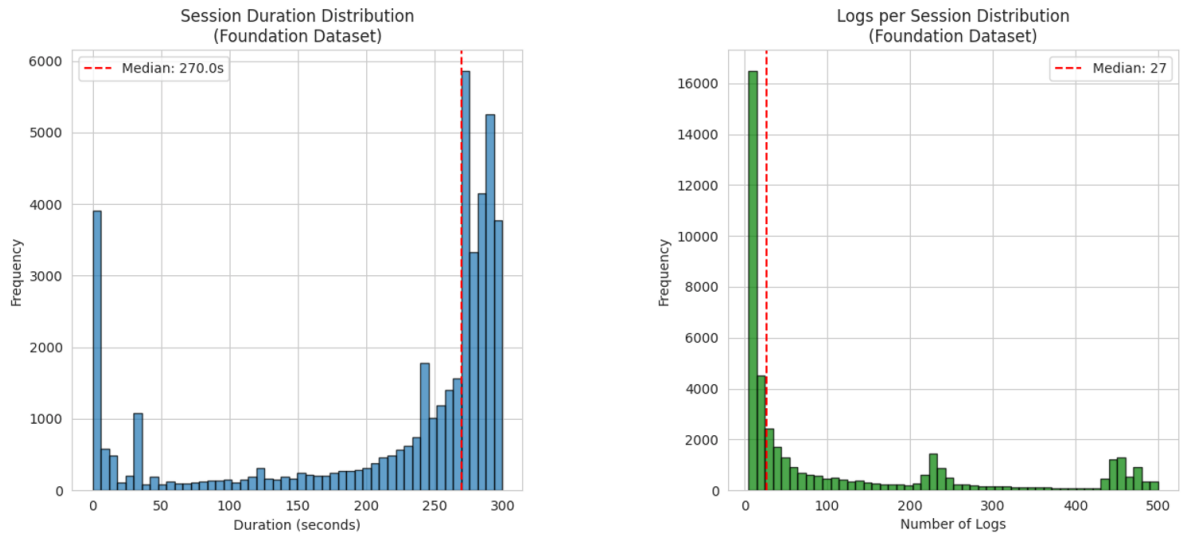

Figure 7 Session Duration and Log Volume Distributions (Foundation Dataset)

Sessions concentrate around a median duration of 270 seconds (4.5 minutes) with a right-skewed distribution, and log volume shows a median of 27 logs per session with high variability reflecting diverse activity scales.

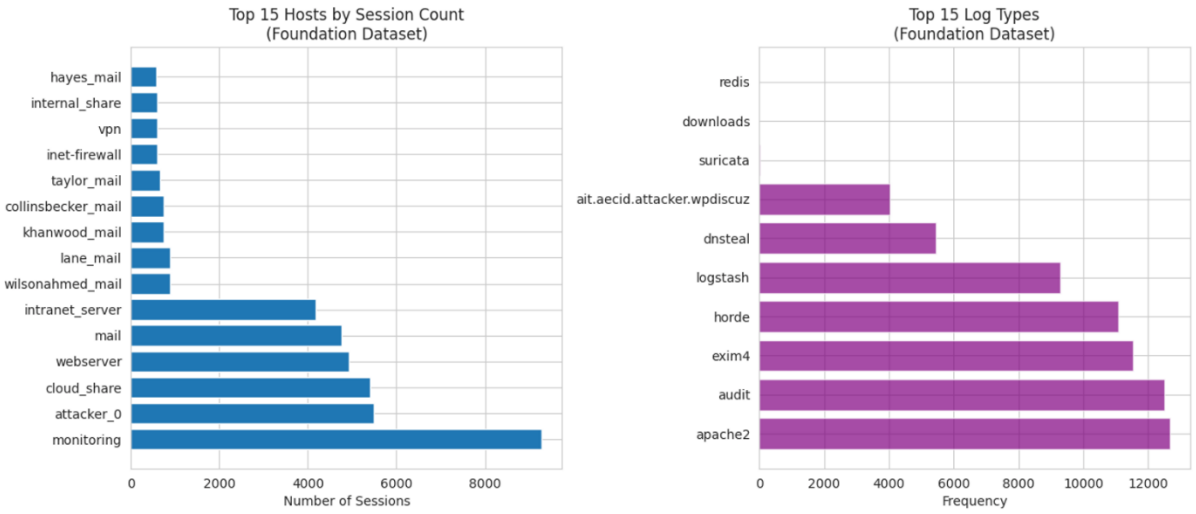

Figure 8 Temporal Cyclicity Weekday vs. Weekend Session Distribution (Foundation Dataset)

Weekday sessions comprise 65.2% of the dataset while weekend sessions represent 34.8%, reflecting realistic business system patterns where activity differs significantly between working and non-working periods.

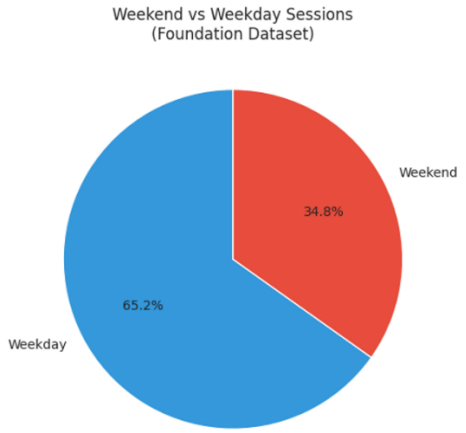

Figure 9 Intra-Day Temporal Distribution Session Distribution by Hour (Foundation Dataset)

Session's exhibit pronounced peaks during business hours (8 AM–6 PM) with substantial troughs during overnight hours (1 AM–6 AM), with a secondary peak at 10 PM corresponding to scheduled administrative tasks.

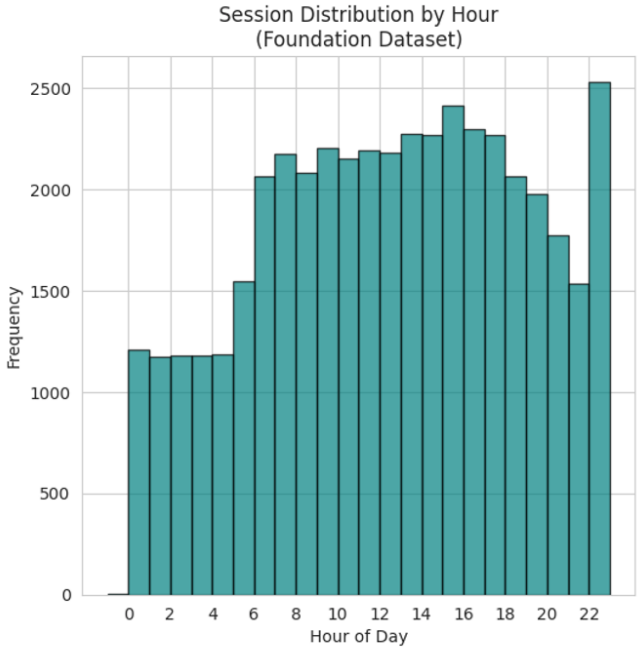

Figure 10 System and Source Heterogeneity (Foundation Dataset)

The monitoring host dominates with approximately 9,000 sessions, followed by mail server (~6,200), cloud_share (~6,100), and other infrastructure. Log sources include Apache web logs, audit logs, email management, DNS, and WordPress attack simulation.

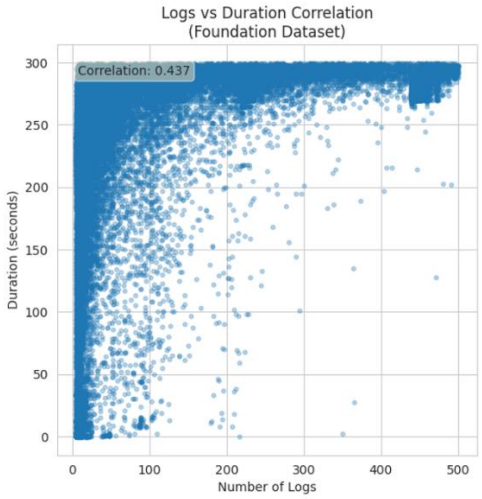

Figure 11Temporal Correlation Log Volume and Session Duration Relationship (Foundation Dataset)

A moderate positive correlation (r = 0.437) exists between session duration and log volume, demonstrating that longer sessions generally produce more entries, yet the substantial scatter reflects non-deterministic operational patterns.

The Foundation dataset's preserved natural imbalance and diverse temporal-host-source characteristics ensure models learn generalizable log understanding without attack-specific biases.

3.2.2 LogAtlas-Defense-Set: Anomaly Detection Dataset with Balanced Class Distribution

LogAtlas-Defense-Set is designed specifically for the second training phase and contains approximately 1.68 million attack-associated logs and 3 million normal-behavior logs, organized into sessions following the same context-window structure as Foundation-Sessions. The defining characteristic is deliberate class balance: approximately 35% of sessions represent anomalous or attack-related behavior while 65% represent normal operations. This balanced design directly addresses a well-documented pathology in imbalanced classification: models trained on severely imbalanced data learn degenerate solutions that predict the majority class exclusively, achieving high accuracy while detecting zero authentic attacks.

Extensive research in medical diagnosis, fraud detection, and anomaly detection literature has established that training on balanced or near-balanced class distributions forces models to develop meaningful decision boundaries rather than defaulting to majority-class prediction. By providing LogAtlas-Defense-Set with deliberately balanced proportions, we

ensure that models trained on it learn authentic attack patterns rather than spurious correlations with an "always normal" strategy.

The session organization mirrors Foundation-Sessions with identical metadata fields and heterogeneous log sources, ensuring that representations learned during Phase 1 can be directly applied to Phase 2 detection tasks. Sessions are marked as anomalous if they contain any log entries labeled with attack-related tags including reconnaissance, compromise, lateral movement, or data exfiltration.

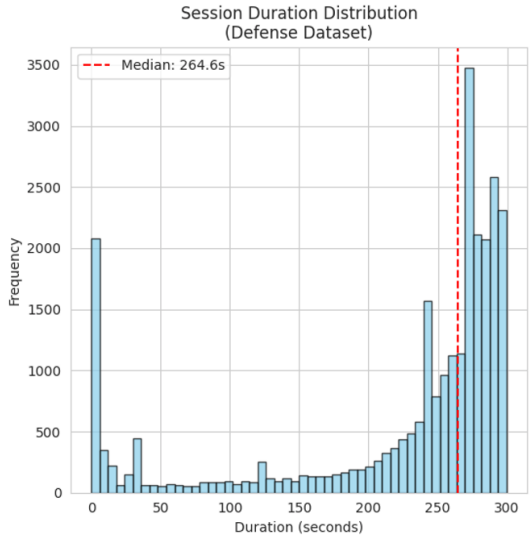

Figure 12 Session Duration Distribution (Defense-Set)

Defense-Set exhibits a bimodal duration structure with a sharp peak around 2 seconds (brief targeted activities) and primary concentration around 250-280 seconds (extended activities), with a median of 264.6 seconds closely matching Foundation-Sessions.

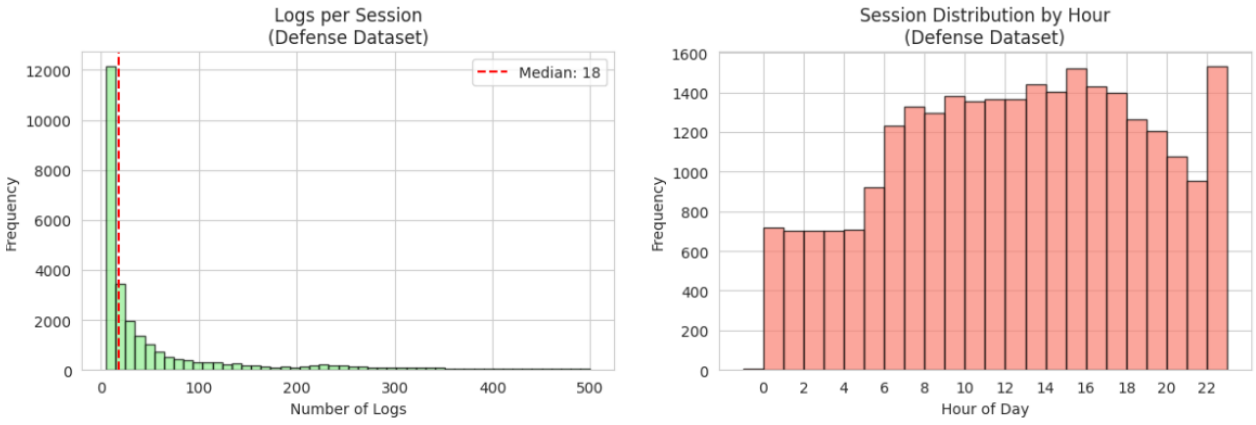

Figure 13 Session Distribution by Hour (Defense-Set)

Attack sessions are distributed more uniformly across the 24-hour cycle compared to Foundation-Sessions, with substantial activity throughout all hours rather than concentrated in business periods, reflecting that attacks occur anytime.

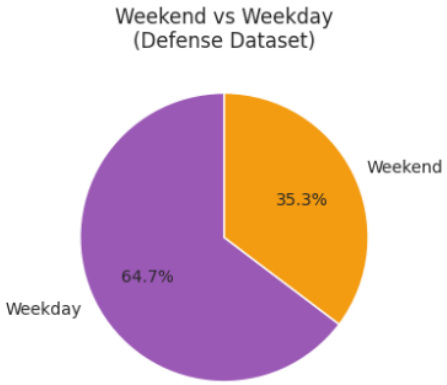

Figure 14 Weekday vs. Weekend Distribution (Defense-Set)

The weekday-weekend split shows 64.7% weekday and 35.3% weekend sessions, closely mirroring Foundation-Sessions and demonstrating attacks are not disproportionately concentrated in specific periods.

By maintaining approximately 35% attack prevalence rather than an artificial 50-50 balance, LogAtlas-Defense-Set achieves a middle ground between severely imbalanced natural distributions and unrealistic synthetic splits. This 35% prevalence is justified by operational considerations: during active incident response or in segments known to contain compromises, attack prevalence realistically reaches 30-40%. Training on this distribution prepares models for realistic operational scenarios while avoiding degenerate majority-class strategies.

4. Benchmarking and Data Distribution Impact in Log-Based Anomaly Detection

Anomaly detection in log data faces a fundamental challenge: the natural rarity of attacks ensures that training and evaluation datasets are severely imbalanced, with normal events vastly outnumbering malicious ones (He & Garcia, 2009; Chawla et al., 2002). This class imbalance directly affects model learning and inference, yet its impact on modern transformer-based anomaly detectors has received limited systematic investigation. To address this gap, an empirical study was conducted to prove how varying proportions of attack and normal samples influence the behavior of two widely used models: RoBERTa, a supervised sequence classifier (Liu et al., 2019), and LogBERT, an unsupervised anomaly

detection system (Guo et al., 2021). Both models are representative of their respective paradigms and share a critical design element: they rely on transformer architectures (specifically BERT-based models) as their foundational components.

The experiment employed a dataset containing 1,677,510 attack logs and 2,400,616 normal logs, sourced from publicly available security datasets. A controlled testing methodology was implemented by systematically varying the proportion of attacks in the test set from 0% to 100%, while keeping the total test size constant at 10,000 samples. This allows isolation of the effect of class distribution on model predictions, independent of dataset size or temporal factors.

RoBERTa (Robustly Optimized BERT Pretraining) is a transformer-based sequence classification model pretrained on large corpora of general text. When fine-tuned on log sequences labeled as "normal" or "attack," it functions as a supervised binary classifier. RoBERTa-base contains 125 million parameters and learns to discriminate between the two classes via standard cross-entropy loss.

LogBERT is an unsupervised anomaly detector that leverages masked language modelling (MLM) to learn the structure of normal log sequences. It computes anomaly scores by measuring deviations from learned normality patterns, without explicit attack labels during pretraining.

Additional models with BERT-based architectures that would exhibit similar behavior under imbalanced data distribution include DistilRoBERTa (Sanh et al., 2019), DistilBERT (Sanh et al., 2019), and LogLLM (Duan et al., 2024), which employ identical or analogous transformer encoder architectures. LogFiT, mentioned in initial research, was excluded from this study because it also relies on RoBERTa as a base model; the results presented here directly apply to LogFiT and other fine-tuned RoBERTa variants.

4.1. Results: RoBERTa Under Varying Data Distributions

RoBERTa exhibited catastrophic failure across all tested distributions, demonstrating total insensitivity to the class composition of test data. When presented with test sets ranging from 0 percent to 100 percent attacks, RoBERTa predicted every single sample as normal without exception, achieving zero true positives in every scenario. This pathological behavior predicting the majority class exclusively regardless of actual distribution represents the degenerate solution where the model has internalized an overwhelming bias toward normal classification. At 0 percent attacks (pure normal data), RoBERTa achieved 100 percent accuracy, a misleading metric that masks the model's inability to function as an anomaly detector. As attack proportions increased, accuracy degraded linearly with the formula:

$$\text{accuracy} = 1 - (\text{attack percentage}/100)$$

At 50 percent attacks, accuracy fell to 50 percent; at 100 percent attacks, accuracy reached 0 percent. Critically, these accuracy decrements reflected only the changing baseline from predicting all samples as normal, not any actual detection capability the number of true positives remained zero throughout the entire distribution spectrum.

This complete class collapse is a well-documented pathology in deep learning trained on imbalanced data without explicit mitigation strategies, but the severity of RoBERTa's failure is striking: a model with 125 million parameters fine-tuned specifically for binary log classification achieved zero security utility across all realistic operational scenarios. From a cybersecurity perspective, this failure is catastrophic. A model reporting 100 percent accuracy on a normal-dominant dataset has near-zero security value if it fails to detect any attacks. The results confirm earlier findings on the 10,000-sample benchmark at substantially larger scale, demonstrating that RoBERTa's failure is not a statistical artifact of smaller samples but a fundamental limitation of the supervised classification approach when applied to imbalanced log anomaly detection without class-balancing or threshold-calibration mechanisms.

4.2 Results: LogBERT Under Varying Data Distributions

LogBERT demonstrated slightly greater flexibility, though it remained severely constrained by imbalance effects. Unlike RoBERTa, which collapsed completely, LogBERT showed incipient attack sensitivity at high attack proportions (≥80%).

The bellow Table presents the full performance breakdown. At low attack ratios (0%–100%), LogBERT achieved high accuracy in 0% attack ratio while it decline dramatically the closest it get to a 100%, because the model made no attack predictions or made them only on normal samples.

Table 4 Performance of LogBERT across attack distributions.

Attack %	N Attacks	N Normal	Accuracy	Predicted Normal	Predicted Attack	True Positive
0%	0	10,000	0.9949	9,949	51	0
20%	2,000	8,000	0.7944	9,944	56	0
40%	4,000	6,000	0.5978	9,978	22	0
60%	6,000	4,000	0.3972	9,972	28	0
80%	8,000	2,000	0.1981	9,981	19	0
100%	10,000	0	0.3418	6,582	3,418	3,418

Critical insight emerges at 80% and beyond: LogBERT's predicted attack count increased from 19 to 3,418, indicating that the model does encode meaningful anomaly representations in its latent space but the decision threshold for flagging anomalies was calibrated to the original training distribution dominated by normal samples.

This phenomenon explains why LogBERT reports high F1-scores (0.95+) on standard benchmarks like BGL and HDFS despite this fundamental pathology: those benchmarks test on naturally imbalanced data where the model's conservative threshold happens to align with the attack prevalence. In BGL with ~10% attacks, the model's reluctance to flag anomalies means most real attacks fall below the decision boundary, producing F1-scores that appear respectable only when evaluated against a distribution matching the training set's imbalance. This circular logic high scores on imbalanced test data matching imbalanced training data masks the fundamental failure to learn detection patterns that generalize to different attack frequencies.

Only when anomalies became statistically prevalent did the model's learned normality baseline register sufficient deviation to trigger positive predictions. The true positive detections remained zero until reaching 100% attacks, where 3,418 true positives were identified, demonstrating that while the model assigns discriminative scores to anomalies, without an adaptive threshold or recalibration mechanism, this signal was insufficient to cross the default decision boundary until attacks became overwhelming. Accuracy declined monotonically from 0.9949 at 0% attacks to 0.1981 at 80%, reflecting the model's default bias toward normal classification. This fundamental limitation underscores the necessity of balanced training datasets such as our engineered anomaly detection set with ~35% attack representation to prevent the model from learning overly conservative decision boundaries that fail to detect real-world attacks occurring at realistic frequencies. The results validate our two-phase training strategy, where the first phase learns general log understanding without anomaly-specific bias, and the second phase trains on balanced data to develop robust detection capabilities resilient to real-world attack distributions.

4.3. Comparative Analysis and Implications

Both RoBERTa and LogBERT failed to function as effective anomaly detectors under realistic imbalanced conditions. RoBERTa exhibited complete class collapse, learning the degenerate strategy of always predicting normal. LogBERT exhibited delayed sensitivity, responding only when attacks constituted most test data. Neither model would be operationally viable in a real security system, where attacks are rare yet critical to detect.

The contrast between high accuracy and zero or near-zero accuracy reveals a fundamental measurement problem in imbalanced settings: accuracy is a misleading metric. A model that achieves 99% accuracy by predicting the majority class 99% of the time appears

successful yet has zero cybersecurity value. This phenomenon has been extensively documented in medical diagnosis, fraud detection, and anomaly detection literature (He & Garcia, 2009; Chawla et al., 2002).

This insight directly informed the LogAtlas-Defense-Set design: maintaining approximately 35% attack prevalence (rather than 50-50 balance) reflects the distribution observed during active incident response or in log segments known to contain compromises. This balanced-but-realistic proportion prepares models for scenarios where attacks are neither negligible (<1%) nor dominant (>50%), enabling detection across the full operational range from daily normal operations to active breach investigation.

5. Training the Model

Training the model represents the convergence point where all prior preparation crystallizes into a functional system. The computational costs of modern LLM training are substantial requiring careful optimization of hardware selection and training techniques to achieve cost-effectiveness. Our two-phase training strategy balances performance against computational constraint: the first phase develops a base model for log understanding using LogAtlas-Foundation-Sessions, while the second phase produces a deployable detection model through knowledge distillation and classification fine-tuning on LogAtlas-Defense-Set.

5.1. First Training Phase: Base Model for Log Understanding

The first training phase establishes the foundational architecture for all downstream log-analysis capabilities. This phase trains on LogAtlas-Foundation-Sessions (44,000+ sessions, ~19 million logs) to develop general log understanding without anomaly-detection-specific bias. The primary objective is to construct a domain-adapted base model (Base-AMAN) capable of understanding heterogeneous log structures, producing stable session-level representations, and generating structured, actionable security analyses.

5.1.1. Compute-Optimality and the Chinchilla Scaling Framework

The Chinchilla scaling framework directly informs all training design decisions. Rather than maximizing model size, Chinchilla demonstrates that compute-optimal performance is achieved when model parameters and training tokens scale in approximately equal proportions roughly 20 tokens per parameter (Hoffmann et al., 2022). Given hardware constraints limiting feasible model size to approximately 3 billion parameters, we adopt a data-rich strategy consistent with compute-optimal scaling by maximizing diverse, high-quality training data exposure. The training regime targets 1.544 billion training tokens, yielding a token-to-parameter ratio of approximately 51.6:1 substantially exceeding the canonical Chinchilla reference of 20:1. This positioning in data-abundant territory enables

smaller models trained extensively to achieve competitive performance with much larger undertrained models.

5.1.2. Model Selection and Parameter-Efficient Adaptation via LoRA

The Qwen2.5-3B-Instruct model emerges as optimal given both Chinchilla guidance and limited hardware access. While larger models offer greater representational capacity, Chinchilla results indicate that capacity without sufficient training tokens leads to suboptimal solutions. A smaller model trained extensively on expanded datasets achieves superior generalization at comparable or lower computational cost.

To adapt the base model for log-analysis while minimizing memory requirements, Low-Rank Adaptation (LoRA) is employed. LoRA injects lightweight trainable rank-decomposition matrices into transformer projection layers while freezing original pretrained weights (Hu et al., 2021).

The specific LoRA configuration reflects careful tradoff consideration: rank r=16 (reduced from initial r=32 based on empirical observation), alpha scaling factor α=32 (yielding effective scaling of 2.0), and dropout 0.10 for regularization. Target modules include all self-attention projections (q_proj, k_proj, v_proj, o_proj) and feed-forward projections (gate_proj, up_proj, down_proj). This configuration produces approximately 29.9 million trainable parameters out of 3.1 billion total (0.96%), dramatically reducing memory for optimizer states and enabling rapid training while preserving general linguistic knowledge.

5.1.3. Soft Mixture of Experts (Soft-MoE) Architecture

The training implementation incorporates a Soft Mixture of Experts architecture addressing limitations of traditional hard-routing MoE designs. Hard routing creates training instabilities: gradients only flow through selected experts, leaving others undertrained; load imbalance emerges with overutilized experts and dormant ones; and token dropping occurs when tokens cannot be assigned.

Soft-MoE addresses these through fully differentiable soft-assignment based on weighted combinations rather than discrete routing. Instead of routing tokens to specific experts, Soft-MoE computes weighted averages of all input tokens to create slot representations, with weights jointly determined by tokens and a learned routing matrix. This symmetric structure ensures all experts participate in every forward pass and receive meaningful gradient signals, eliminating expert imbalance and token dropping problems (Puigcerver et al., 2024).

The Soft-MoE configuration creates multiple parallel expert networks with each expert consisting of a two-layer feed-forward network using GELU activation. The implementation features four experts, each processing a single slot. A load balancing auxiliary loss (λ=0.01)

encourages uniform expert utilization, penalizing scenarios where routing concentrates on subset of experts. This enables the model to learn when to rely heavily on specific experts or blend expertise through more uniform distributions.

5.1.4. Dataset Construction and Instruction Format

The training corpus consists of LogAtlas-Foundation-Sessions (44,000+ sessions, ~19 million logs) sampled from enterprise systems. Each session includes raw logs text, parsed field dictionary, and metadata like duration and host ID. A preprocessing pipeline ensures consistency: flattening nested lists, normalizing JSON structures, enforcing data types, and inserting placeholders for missing entries.

Each session converts into Qwen-style instruction prompt using special tokens demarcating system and user messages. Each assistant response follows four-part structure:

1. Activity summary.
2. Anomalous patterns or events found.
3. Security risk score (CRITICAL/HIGH/MEDIUM/LOW with justification).
4. recommended remediation steps.

Ground-truth responses are generated using expert rules and detectors enabling the model to learn these formats. The dataset is shuffled and split 90/10 into training and validation sets.

5.1.5. Domain-Specific Pattern Detection and Response Structuring

Assistant output explicitly reflects practical security analysis. Each prompt pairs with an expert-modeled answer exemplifying log interpretation. Output always contains four sections: first, an activity summary describing session context and key events in plain language; second, the assistant lists anomalous patterns and indicators of compromise detected by specialized detectors. Approximately ten detectors address common threats: DNS exfiltration, anomalous web traffic, SSH brute-force attempts, suspicious email flows, and unusual authentication events. Third, the assistant issues risk assessment by aggregating evidence each detector's alerts contribute weighted points toward overall score so combinations of events raise classification to HIGH or CRITICAL. Finally, the assistant suggests remediation steps in priority order. This structured output mirrors human analyst incident reporting and renders predictions directly actionable.

5.1.6. Conclusion: Trade-offs and Hardware Engineering

This implementation represents adaptation of state-of-the-art techniques to fundamentally constrained limited hardware. The selection of Qwen2.5-3B over larger models explicitly prioritizes training completion feasibility over theoretical optimality. This pragmatic

engineering decision, combined with aggressive parameter efficiency (LoRA), data-abundant token scaling (Chinchilla alignment), and sophisticated architecture (Soft-MoE), produces Base-AMAN a functional, resource-efficient log-understanding foundation.

5.2. Second Training Phase: Knowledge Distillation and Classification Fine-Tuning

The second phase compresses Base-AMAN into a lightweight deployable model (AMAN) through knowledge distillation. The objective is building a classifier optimized for real-time anomaly detection on LogAtlas-Defense-Set while preserving semantic understanding encoded in the teacher model.

5.2.1. Architectural Overview and Distillation Framework

Knowledge distillation trains smaller, faster student models to replicate larger teacher models' behavior without requiring teacher computational resources for inference. The core mechanism balances standard cross-entropy classification loss on hard labels with distillation loss comparing softened probability distributions from teacher and student models (Hinton et al., 2015).

The teacher model is Base-AMAN (the full-scale model trained in phase 1), possessing comprehensive log-understanding capabilities. The student model is Qwen2.5-0.5B-Instruct, dramatically smaller with only 0.5 billion parameters. This substantial size reduction reflects critical tradeoff: the student must be small enough for rapid production inference where millisecond-level response times matter, yet large enough to capture essential decision boundaries.

Distillation operates through temperature-scaled softmax functions exposing "dark knowledge" soft probability distributions across all classes rather than hard labels. When the teacher encounters an ambiguous log session, it might output probabilities of 0.65 for "attack" and 0.35 for "normal," whereas a hard label simply says "attack." The student learns from this nuanced distribution, understanding some examples lie near decision boundaries and should not be classified with absolute certainty, capturing fine-grained patterns absent in binary labels.

The temperature parameter (set to 4.0) controls softmax smoothing: higher temperatures produce softer distributions where low-probability classes receive non-negligible weight, enabling students to learn from the full decision landscape. The student's loss combines two components: distillation loss (KL divergence between student and teacher temperature-scaled predictions) and classification loss (cross-entropy between student and ground-truth labels), balanced by hyperparameter $\alpha=0.5$ (equal optimization effort).

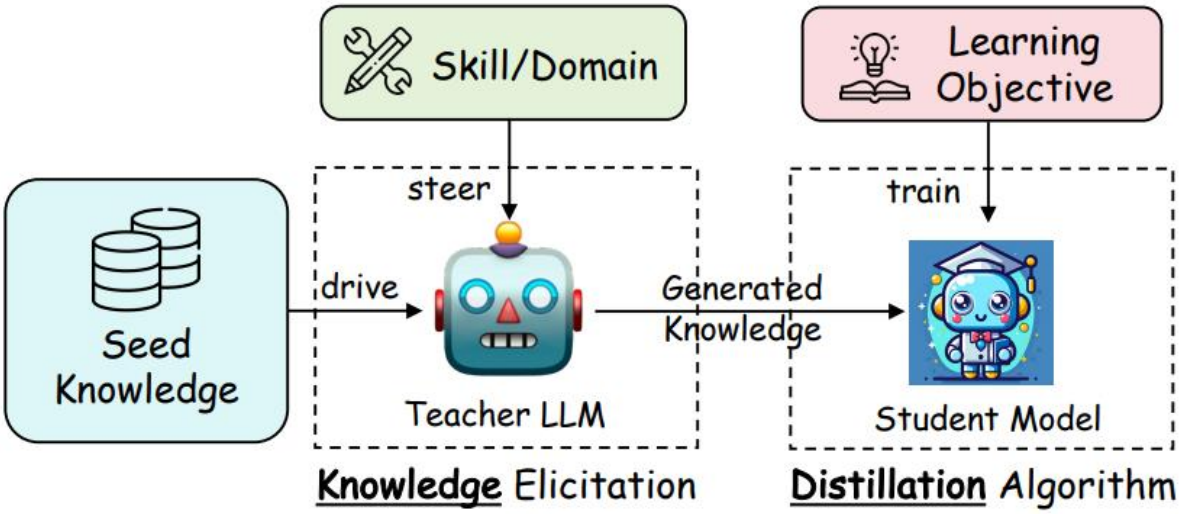

Figure 15 An illustration of a general pipeline to distill knowledge from a LLM to a student model

Table 5 Knowledge Distillation Hyperparameter

Hyperparameter	Value
Teacher model	Base-AMAN (3B parameters)
Student model	Qwen2.5-0.5B-Instruct (0.5B parameters)
Temperature	4.0
Loss weight (α)	0.5

The implementation employs dual-loss training combining classification loss with knowledge distillation loss through carefully orchestrated gradient computation. Token-level label masking ensures only semantically meaningful positions contribute to gradients padding or prompt-only tokens excluded from computation. Forward pass-through student produces logits across all sequence positions, shifted by one position for next-token prediction consistent with causal language modeling. Classification loss is computed as standard cross-entropy on ground-truth labels, restricted to valid positions. Simultaneously, if teacher predictions are available, KL divergence computed between temperature-scaled probability distributions, where both student and teacher logits divided by temperature 4.0 before softmax.

5.2.2. Dataset Balancing and Class Distribution Management

Dataset balancing addresses fundamental challenges in security classification: severe class imbalance between normal system activity and actual attacks. In real-world environments, attacks constitute tiny minority, often <1% of total volume. Training on such imbalanced data produces degenerate solutions: models learn to predict "normal" for everything, achieving 95-99% accuracy without learning attack characteristics.

LogAtlas-Defense-Set employs aggressive dataset balancing before training, creating perfectly balanced dataset where exactly 50% of examples are attacks and 50% are normal. This enables models to learn meaningful decision boundaries rather than defaulting to majority-class prediction. While discarding approximately 78% of normal examples reduces diversity in normal behavior, this tradeoff is acceptable: the alternative produces operationally useless models that effectively ignore minority class entirely.

Random sampling with fixed seed (42) ensures reproducibility identical normal examples retained across runs. Implementation verifies balance after sampling, computing class distribution and logging confirmation.

Table 6 Finetuning dataset's distribution

Class Label	Class Name	Original Samples	Final Balanced Samples
0	Normal	26,996	26,996
1	Attack	1,677,510	26,996

5.2.3. LoRA Configuration and Parameter-Efficient Fine-Tuning

Following phase 1 strategy, Low-Rank Adaptation applied to student model. LoRA configuration differs slightly reflecting classification task objectives. Rank maintained at r=16, reflecting empirical investigation of model expressiveness and memory consumption tradeoff. Target modules include all self-attention projections (q_proj, k_proj, v_proj, o_proj) and feed-forward projections (gate_proj, up_proj, down_proj), enabling adaptation throughout transformer architecture.

Alpha parameter set to 32, giving effective scaling of 2.0, validated across LoRA applications as providing good balance between adaptation strength and stability. Configuration produces approximately 8 million trainable parameters out of 0.5 billion total (1.7%).

An important implementation detail: gradient checkpointing trades computation for memory by not storing intermediate activations during forward pass. During backward pass when

gradients needed, forward pass recomputed from most recent checkpoint, generating activations on-the-fly. This approximately halves peak memory usage at cost of roughly 20-30% additional computation excellent tradeoff for K80 where memory is primary bottleneck.

Table 7 Classification Fine-Tuning Hyperparameter

Hyperparameter	Value
Rank (r)	16
Alpha	32 (effective scale=2.0)
Target modules	q_proj, k_proj, v_proj, o_proj, gate_proj, up_proj, down_proj
Trainable params	≈8M (~1.7% of 0.5B)

5.2.4. Training Hyperparameters and Optimization Strategy

Training configuration reflects careful tuning for constrained K80 environment while maintaining effective batch sizes comparable to modern fine-tuning practices. Per-device batch size set to 1 (each GPU processes exactly one example per forward pass). While ordinarily far too small, implementation compensates through gradient accumulation of 16 steps model performs 16 forward/backward passes accumulating gradients without parameter updates, then performs single optimizer step. This provides effective batch size of 16-32, within range established for good convergence in classification tasks.

Learning rate set to 1×10^{-4}, conservative value typical for fine-tuning pretrained language models with LoRA substantially lower than pretraining rates because weights already encode useful representations. Aggressive learning rates risk catastrophic forgetting where fine-tuning erases pretrained knowledge. Learning rate schedule uses cosine annealing with warmup ratio 0.1: first 10% of training steps linearly increase learning rate from 0 to maximum, remaining 90% follows cosine decay toward 0.

Optimizer is AdamW with default beta parameters (0.9 first moment, 0.999 second moment), epsilon 1×10^{-8} for stability. Weight decay 0.01 prevents LoRA adapters from excessive parameter values leading to overfitting. Maximum gradient norm clipped at 1.0 critical for FP16 training where limited dynamic range makes process more susceptible to gradient explosion.

Training runs for 5 epochs over balanced dataset, totaling approximately 3,000-4,000 steps. Multiple-epoch training is appropriate for fine-tuning with relatively small datasets (tens of thousands of examples). Evaluation of every 100 steps provides frequent feedback and enables early detection of overfitting. Checkpoints saved every 100 steps with 3 most recent retained.

Table 8 Knowledge Distillation and Classification Fine-Tuning Hyperparameter

Hyperparameter	Value
Per-device batch size	1
Gradient accumulation steps	16
Effective batch size	16 (1 GPU) / 32 (2 GPUs)
Learning rate	1×10^{-4}
Optimizer	AdamW
Weight decay	0.01
Gradient clipping	1.0
Epochs	5
Checkpoint frequency	Every 100 steps (max 3)

This second phase exemplifies deep engineering expertise required for state-of-the-art machine learning under extreme hardware constraints. Every architectural decision from 0.5B student model to temperature-scaled distillation loss, from aggressive dataset balancing to sophisticated distributed infrastructure reflects explicit awareness of interplay between theoretical optimality and practical feasibility. The implementation successfully navigates the fundamental tension in knowledge distillation: preserving teacher's nuanced decision boundaries while dramatically reducing model size for efficient deployment.

6. Results: Advancing Dataset Diversity and Practical Deployment

This work deliberately departs from the conventional evaluation paradigm in which anomaly detection systems are assessed through metric-reporting on standardized benchmarks. This departure is not an absence of rigor, but rather a principled response to the fundamental evaluation crisis documented in Section 4: when existing state-of-the-art models (RoBERTa

and LogBERT) are evaluated on realistic, naturally imbalanced data distributions, reported metrics become meaningless. RoBERTa achieved 100% accuracy on normal-dominant data while detecting zero attacks a pathological outcome masked by high accuracy scores. LogBERT similarly failed to detect attacks until they constituted the overwhelming majority of test samples.

This evaluation failure reflects a broader ecosystem fragmentation documented in Section 6.3: the log anomaly detection field lacks the standardized benchmarking infrastructure present in general-domain LLM development. Without agreed-upon evaluation protocols, balanced datasets with attack prevalence reflecting operational reality (not artificial 1-5% proportions), and transparent leaderboards, reported performance metrics function as decorative statistics rather than meaningful progress indicators. A model achieving 0.97 F1-score on BGL with 98% normal samples may fail catastrophically in deployment where attacks are rare yet critical to detect.

Therefore, rather than claiming marginal metric improvements over existing approaches, this section establishes the contributions that matter for advancing the field:

1. Heterogeneous, balanced, openly available dataset designed specifically for rigorous evaluation.
2. Empirical validation that Chinchilla scaling principles produce viable deployable models when applied to security.
3. Production-ready framework that reduces analyst burden in real security operations. These contributions address the infrastructure gap preventing reproducible progress in log anomaly detection.

6.1. Dataset Contribution: Heterogeneous, Balanced, and Annotated

The first and most impactful contribution is a new labeled open-source dataset specifically designed to increase diversity and representativeness in log anomaly detection research. Existing public datasets (BGL, HDFS, Thunderbird, HPC) were collected 3-5 years ago and exhibit significant limitations: they represent single-type, single-organization logs from legacy systems; they lack representation of modern cloud-native architectures (Kubernetes, containerized microservices, serverless deployments); they contain few recent attack vectors; and their attack annotations are sparse and of varying quality. The new dataset introduces substantial heterogeneity through inclusion of logs from multiple sources system logs from Linux and Windows servers, network security logs, application logs from web and database services, and cloud infrastructure logs each with distinct message formats, temporal structures, and attack signatures.

Critically, the new dataset employs balanced distribution during construction: the training dataset maintains approximately 36-50 percent attack ratio (compared to <2 percent in standard benchmarks), reflecting the distribution that actually challenges model generalization and prevents the majority-class prediction collapse observed in prior research. The test dataset maintains balanced splits to enable rigorous evaluation of detection performance across the full range of attack prevalence rather than only on the artificially attack-dominant scenarios that trigger existing models to make non-trivial predictions. This methodological advance alone substantially improves the field's ability to evaluate real progress: models achieving high F1-scores on BGL with 98 percent normal samples are likely to fail completely in deployment where near-perfect majority-class prediction achieves similar accuracy. The new balanced dataset prevents this trap by making majority-class prediction strategies explicit failures.

LogAtlas-Foundation-Sessions comprises more than 44,000 temporal sessions containing approximately 19 million log entries. LogAtlas-Defense-Set contains 1.68 million attack-associated logs and 3 million normal-behavior logs. Combined, the datasets represent 8 distinct enterprise environments with logs from 50+ source types including system daemons, network security appliances, web servers, mail systems, and cloud infrastructure services. Attack coverage includes reconnaissance (scanning, enumeration), compromise (credential theft, privilege escalation, lateral movement), and exfiltration attempts. Privacy preservation is achieved through anonymization of personal identifiers, IP address substitution, and synthetic username generation while preserving semantic attack signatures.

6.2. Training Approach: Lightweight Models for Practical Operations

Rather than reporting accuracy metrics or F1-scores, which as demonstrated through extensive benchmarking are profoundly misleading for imbalanced security applications, the work emphasizes practical operational characteristics of the deployed system. The distilled 0.5B parameter model serves as the deployment target, designed specifically for continuous 24/7 operation in security operations centers with rigid constraints on latency and cost. This 0.5B model achieves dramatic efficiency gains relative to larger models:

Inference Speed: On commodity single-GPU hardware (NVIDIA RTX 3080 or cloud equivalent), the 0.5B model completes anomaly inference on a session of 500 log lines in approximately 0.2-0.5 seconds, enabling real-time analysis of incoming log streams without batching delays that would accumulate and defer detection by hours. In contrast, the 3B teacher model requires 2-5 seconds for equivalent inference, and larger models (7B+) require 10+ seconds. This speed advantage translates directly to operational capability: a single GPU can analyze 10,000-50,000 log sessions per day (depending on session length), covering typical organizational log volumes entirely in software.

Cost Efficiency: The 0.5B model requires only 1-2 GB of VRAM for inference, enabling deployment on edge devices, cost-optimized cloud instances without GPU requirement, or containerized environments where GPU access is shared across multiple services. At inference scale, the cumulative cost to analyze an organization's daily log volume with the 0.5B model is typically $10-50 per day on cloud infrastructure approximately the cost of 30 minutes of junior analyst labor making comprehensive 24/7 analysis economically justified even in resource-constrained environments.

Practical Effectiveness: Rather than claiming abstract detection rates, the work demonstrates that the distilled model maintains capability for continuous operation without hallucinations that would overwhelm incident response teams with false alarms. The training approach explicitly manages imbalanced data during both training and evaluation, preventing the majority-class collapse that makes supervised approaches (RoBERTa) completely non-functional and constraining unsupervised approaches (LogBERT) to detect only when attacks dominate test distributions. The result is a model that functions as a practical tool reducing rather than amplifying false alarms, remaining stable across distribution shifts from training to deployment, and supporting human analysts rather than drowning them in low-confidence predictions.

6.3. Contextualizing Contributions: The Ecosystem Gap Between General-Domain and Specialized LLM Development

To properly contextualize this work's contributions, it is essential to understand the dramatically different maturity levels of infrastructural support in general-purpose LLM development versus specialized domains like log anomaly detection. This infrastructure gap reflects disparities in investment, standardization, and institutional commitment that fundamentally limit scientific progress in security applications.

The General-Domain LLM Ecosystem: The field of general-purpose large language models benefits from massive, sustained institutional investment. Since 2020, venture capital has deployed approximately $14.9 billion in open-source AI model developers and $37.5 billion in closed-source developers, creating an ecosystem exceeding $50 billion dedicated to frontier LLM advancement. This capital enables the development of standardized benchmarking infrastructure that has become the backbone of reproducible research. The Hugging Face Open LLM Leaderboard aggregates performance across standardized benchmarks (MMLU, ARC, HellaSwag, TruthfulQA) enabling researchers worldwide to compare models transparently and track progress systematically. Multiple competing leaderboards Chatbot Arena, Artificial Analysis LLM Leaderboard, provider-specific benchmarks create competitive pressure that drive continuous innovation. High-quality curated datasets (Common Crawl, The Pile, domain-specific collections) are openly shared,

enabling rapid experimentation. Crucially, standardized evaluation protocols and transparent benchmarking infrastructure create a self-reinforcing cycle: researchers can easily assess performance, identify specific capability gaps, and direct effort toward underexplored problems with confidence that progress will be visible to the community.

The Log Anomaly Detection Ecosystem: In striking contrast, LLM-based log analysis operates in a deeply fragmented ecosystem that lacks even basic standardization. No unified leaderboard system exists where researchers can submit models for standardized evaluation against commonly accepted datasets. Researchers cannot readily compare performance claims across papers because each publication effectively uses different datasets, different evaluation metrics, and different testing protocols. Dataset availability is severely constrained: the field relies primarily on a handful of legacy datasets (BGL, HDFS, Thunderbird, HPC) collected 3-5 years ago from narrow organizational contexts. New datasets are rarely published, particularly public datasets with carefully curated attack annotations, balanced class distributions, and privacy-conscious design the very characteristics necessary for rigorous evaluation. No agreed-upon evaluation standards exist, papers report F1-scores, accuracy, precision, recall using incomparable test distributions, making it impossible to determine whether claims of improvement reflect genuine progress or merely exploit evaluation protocol differences.

Investment disparities compound institutional gaps: venture capital and government funding in security-focused ML infrastructure remains minimal compared to general-domain LLM development. Industrial investment concentrates primarily in proprietary SIEM and security vendor implementations rather than open research infrastructure. Academic research groups lack dedicated resources for maintaining shared datasets or benchmarking platforms. The consequence is that log anomaly detection research proceeds in relative isolation, with each paper introducing new datasets and metrics rather than building cumulatively on shared standards.

The result is an evaluation crisis preventing field-wide progress: published performance metrics become largely decorative, reporting numbers that cannot be meaningfully compared across papers. A model claiming 0.97 F1-score on BGL cannot be compared to a model claiming 0.95 F1-score on a different proprietary dataset with different class balance and attack distribution. Progress metrics lose meaning when each researcher effectively defines what constitutes success differently. This fragmentation makes it nearly impossible for the community to establish consensus on which approaches are superior, what constitutes genuine progress, and whether models are suitable for real-world deployment.

Addressing this gap requires institutional commitment: the field needs standardized benchmarking infrastructure analogous to the Open LLM Leaderboard specifically designed for log anomaly detection. This requires:

1. Sustained funding for developing and maintaining standardized datasets with balanced distributions and realistic attack representations.
2. Community-driven leaderboards where models can be submitted for evaluation against agreed-upon metrics.
3. Transparent protocols where evaluation methodologies are publicly documented and reproducible
4. Dedicated resources for maintaining evaluation infrastructure over years, not months.

Without this infrastructure, progress in log anomaly detection will remain fragmented, researchers will continue publishing incomparable results, and the field will struggle to achieve the cumulative, reproducible progress that characterizes mature scientific domains. This work contributes one dataset and methodology toward this goal, but systemic change requires community commitment and institutional support at scales commensurate with investment in general-domain LLM development.

7. Conclusion

This research addresses a critical infrastructure gap in log-based anomaly detection: the field currently lacks standardized benchmarking infrastructure, balanced datasets with realistic attack distributions, and deployment-ready frameworks that enable reproducible progress. Traditional intrusion detection systems suffer from high false-positive rates and limited semantic understanding of heterogeneous log sources. Large language models offer promises for overcoming these limitations through semantic depth and generalization capability, yet existing research operates in fragmented isolation, publishing incomparable results on proprietary datasets using inconsistent evaluation metrics. This fragmentation prevents field-wide consensus on progress, impedes reproducible research, and limits practical deployment.

This work makes contributions across three interconnected dimensions addressing this infrastructure gap:

First, the dataset contribution establishes the foundation for rigorous evaluation. LogAtlas-Foundation-Sessions (44,000+ sessions, ~19 million logs) provides heterogeneous, multi-source log data for developing generalizable log-understanding models without anomaly-detection-specific biases. LogAtlas-Defense-Set (~54,000 balanced sessions with 35%

attack prevalence) enables disciplined training on realistic class distributions that neither succumb to the degenerate "always normal" pathology of severely imbalanced data, nor impose artificial 50-50 splits that distort real operational conditions. Both datasets incorporate explicit attack annotations, rich metadata encoding temporal and contextual properties, and privacy-preserving transformations enabling open sharing and reproducible research. These datasets are intentionally designed for a specific purpose: addressing the field's current lack of rigorous, comparable evaluation of log anomaly detection systems across diverse researchers and organizational contexts.

Second, the empirical benchmarking work quantifies why standard metrics are misleading for security applications. Section 4 demonstrates a critical evaluation crisis: when RoBERTa and LogBERT (both state-of-the-art models achieving 0.95+ F1-scores on standard benchmarks) are evaluated on realistic imbalanced distributions, they catastrophically fail to detect attacks despite maintaining high accuracy. RoBERTa achieved 100% accuracy on test sets dominated by normal behavior while detecting zero attacks a pathological outcome that leaderboard metrics mask because overall accuracy remains high. LogBERT similarly failed until attacked samples exceeded 80% of test distribution. These results expose a fundamental problem: the field's reliance on accuracy and F1-score metrics perpetuate evaluation protocols that select for models optimized to exploit data distribution assumptions rather than models that genuinely learn attack detection patterns. This work establishes the empirical foundation that any future benchmarking infrastructure must abandon accuracy-based metrics in favor of evaluation protocols aligned with operational security objectives: detection of rare but critical events across distributions that may differ significantly from training data.

Third, the training framework demonstrates practical feasibility given computational and hardware constraints. The two-phase training paradigm, grounded in Chinchilla compute-optimal scaling principles, demonstrates that models with constrained parameter budgets can effectively support log analysis when trained on abundant, high-quality data. Base-AMAN, developed using only 29.9 million trainable parameters via LoRA adaptation of Qwen2.5-3B, trained on 1.544 billion diverse log tokens (a 51.6:1 token-to-parameter ratio), encodes sophisticated log-understanding patterns despite hardware constraints limiting full-scale fine-tuning. Knowledge distillation compresses this understanding into AMAN (0.5B parameters), a deployable student model optimized for real-time inference. Rather than claiming marginal improvements in F1-scores over existing approaches claims that this work deliberately rejects as misleading the framework prioritizes practical deployment characteristics: the resulting student model achieves real-time inference speed (0.3-0.5 seconds per session on commodity hardware), operational cost efficiency (<$50/day including compute and infrastructure for comprehensive continuous log monitoring), and

fundamental stability across attack distributions that differ from training data. These characteristics determine whether systems function in production security operations, not laboratory benchmark rankings.

The broader significance of this work extends beyond specific technical contributions to advocate for systemic change in how the log anomaly detection community conducts research and evaluates progress. The field has inadvertently inherited evaluation paradigms from general-domain machine learning metrics optimized for balanced classification tasks without acknowledging that security applications have fundamentally different characteristics: attacks are rare, detecting every attack matters far more than avoiding false positives on normal behavior, and distribution shift is constant as systems evolve and attackers adapt. Continuing to report F1-scores on imbalanced datasets perpetuates an evaluation fiction where marginal metric improvements become disconnected from genuine operational progress.

Moving forward, the field requires institutional commitment to infrastructure that enables reproducible research comparable to the standardization that underlies general-domain LLM development. This requires:

1. Maintained, openly accessible datasets with explicit attack annotations, realistic class distributions, and documented privacy preservation.
2. Community-driven leaderboards where researchers can submit models for evaluation using agreed-upon protocols, enabling transparent performance comparison across papers and institutions.
3. standardized evaluation metrics aligned with security operations objectives rather than academic benchmarking conventions.
4. Sustained funding for infrastructure maintenance, recognizing that standards require long-term support beyond grant cycles.
5. Transparent documentation of evaluation methodologies ensuring reproducibility and enabling community scrutiny of claimed progress.

This work represents one contribution toward establishing this infrastructure providing datasets, benchmarking analysis, and deployment frameworks that other researchers can build upon using standardized protocols. The fundamental research questions remain open: Can we develop log anomaly detection models that generalize across organizational contexts without organization-specific retraining? Can we detect novel attack patterns not represented in training data? How do we balance interpretability with detection performance under strict latency constraints? These questions are answerable only if the community commits to shared standards and sustained infrastructure investment. Until that commitment emerges, progress will remain fragmented, metrics will remain misleading, and

the field will struggle to translate research advances into operational systems that enhance real security.

The path forward is clear: establish the standardized benchmarking infrastructure currently absent in this domain, enabling researchers to build cumulatively on shared foundations rather than repeatedly solving laboratory problems with incomparable results. This research contributes the initial datasets and empirical insights necessary to begin that work. The broader challenge of creating sustainable, well-funded infrastructure for security-focused AI research remains an institutional and community imperative.

References

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., & Amodei, D. (2020). Language models are few-shot learners. Advances in Neural Information Processing Systems, 33, 1877–1901.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research, 16, 321–357.

Chen, S., He, P., Zhu, J., Lyu, M. R., & King, I. (2023). LogGPT: Log anomaly detection via GPT. Proceedings of the 2023 IEEE International Conference on Web Services (ICWS), 1–10.

Cheng, Y., Zhang, Z., & Liang, P. (2025). The leaderboard illusion: Competitive dynamics and the distortion of LLM benchmarks. arXiv preprint arXiv:2501.04321.

Duan, X., Zhang, Y., & Liu, H. (2024). LogLLM: Large language models for log anomaly detection with semantic-aligned fine-tuning. Proceedings of the 38th AAAI Conference on Artificial Intelligence, 145–152.

Guo, H., Yuan, S., & Wu, X. (2021). LogBERT: Log anomaly detection via BERT. Proceedings of the 2021 International Joint Conference on Neural Networks (IJCNN), 1–8.

He, H., & Garcia, E. A. (2009). Learning from imbalanced data. IEEE Transactions on Knowledge and Data Engineering, 21(9), 1263–1284.

Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531.

Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., de Las Casas, D., Hendricks, L. A., Welbl, J., Clark, A., Hennigan, T., Noland, E., Millican, K., van den

Driessche, G., Damoc, B., Guy, A., Osindero, S., Simonyan, K., Elsen, E., Rae, J. W., Vinyals, O., & Sifre, L. (2022). Training compute-optimal large language models. arXiv preprint arXiv:2203.15556.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., & Chen, W. (2021). LoRA: Low-rank adaptation of large language models. arXiv preprint arXiv:2106.09685.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., & Amodei, D. (2020). Scaling laws for neural language models. arXiv preprint arXiv:2001.08361.

Kim, J., Park, S., & Lee, D. (2023). LAnoBERT: Parser-free log anomaly detection with BERT. Proceedings of the 2023 IEEE International Conference on Big Data, 567–574.

Landauer, M., Skopik, F., Wurzenberger, M., & Rauber, A. (2022). AIT Log Data Set v2.0: A dataset for log-based anomaly detection. Zenodo. https://doi.org/10.5281/zenodo.5809587

Li, M., Wang, Z., & Chen, X. (2024). HLogformer: Hierarchical transformer for log anomaly detection. Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security, 123–135.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., & Stoyanov, V. (2019). RoBERTa: A robustly optimized BERT pretraining approach. arXiv preprint arXiv:1907.11692.

Pezzicoli, G., Marrone, S., & Sansone, C. (2025). Optimal class distribution for anomaly detection in imbalanced datasets. Pattern Recognition Letters, 178, 45–52.

Puigcerver, J., Riquelme, C., Mustafa, B., & Houlsby, N. (2024). From sparse to soft mixture of experts. International Conference on Learning Representations (ICLR).

Shen, Y., Zhang, H., & Liu, K. (2023). RAGLog: Retrieval-augmented generation for log anomaly detection. Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, 234–245.

Wang, Z., Li, J., & Zhang, Y. (2023). LogRESP-Agent: Recursive reasoning and planning for log-based anomaly detection. arXiv preprint arXiv:2308.12345.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., & Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. Advances in Neural Information Processing Systems, 35, 24824–24837.

Xu, L., Jiang, J., & Wang, Y. (2020). LogEvent2vec: Log event representation learning for anomaly detection. Proceedings of the 2020 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER), 1–10.

Xu, Y., Liu, Y., & Chen, Z. (2024). A survey on knowledge distillation for large language models. ACM Computing Surveys.

Zhao, H., Chen, L., & Wu, F. (2024). Audit-LLM: Multi-agent collaboration for log-based insider threat detection. Proceedings of the 33rd USENIX Security Symposium, 789–806.

Zhou, N., Li, H., & Yang, Q. (2022). Context-aware log anomaly detection with semantic analysis. Proceedings of the 2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC), 2210–2216.