Less Data, More Security: Advancing Cybersecurity LLMs Specialization via Resource-Efficient Domain-Adaptive Continuous Pre-training with Minimal Tokens

Salahuddin Salahuddin[a,b,c], Ahmed Hussain[b], Jussi Löppönen[a], Toni Jutila[a]

[a]*SSH Communications Security, Helsinki, Finland*
[b]*Networked Systems Security (NSS) Group, KTH Royal Institute of Technology, Stockholm, Sweden*
[c]*Aalto University, Espoo, Finland*

Abstract

While Large Language Models (LLMs) demonstrate exceptional natural language capabilities, general-purpose models often lack the specialized domain knowledge necessary for effective cybersecurity analysis. In this work, we investigate Domain-Adaptive Continuous Pretraining (DAP) as a methodology for enhancing cybersecurity understanding in pretrained LLMs. We systematically adapted three decoder-based architectures—Llama-3.1-8B, DeepSeek-R1-Distill-Qwen-14B, and Llama-3.3-70B-Instruct—using a curated 126-million-word cybersecurity corpus from standards, academic literature, and various other sources. Our approach employed constrained training parameters and distributed Fully Sharded Data Parallel (FSDP) training to balance domain specialization with knowledge preservation. Evaluation across three cybersecurity benchmarks, namely, CTI-MCQ, CyberMetric, and SecEval, demonstrates consistent improvements post-adaptation. Notably, our Llama-3.3-70B-Ins-DAP model achieves state-of-the-art performance with accuracies of 0.718, 0.933, and 0.864, respectively, surpassing parameter-efficient baseline (accuracies $\approx$ 0.485-0.618) and specialized models including Llama-Primus-Base (trained on 2.77 billion tokens) and Foundation-Sec-8B (trained on 5 billion tokens), despite utilizing only 118.8 million tokens—representing a 23-to-42-fold reduction in training data. We demonstrate that targeted continuous pretraining enables effective cybersecurity domain adaptation with computational feasibility, providing a foundation for specialized AI assistants in threat analysis, vulnerability assessment, and security documentation, while challenging prevailing assumptions about data requirements for LLM specialization.

Keywords:
Domain Adaptation, Natural Language Processing, Generative AI, Cybersecurity, Large Language Models, Domain Adaptive continuous Pretraining

1. Introduction

The digitization of critical infrastructure, services, and data has dramatically increased the importance of cybersecurity across all sectors of society. As digital systems become increasingly complex, with numerous nodes exchanging information across networks with diverse configurations and access levels, the challenge of maintaining security grows exponentially. Traditional cybersecurity approaches, illustrated in Figure 1, employ reactive methodologies centered on continuous network monitoring, comprehensive log analysis, and signature-based threat detection to identify and respond to security incidents.

These conventional frameworks rely heavily on security experts who manually configure firewalls, enforce security policies, and analyze extensive datasets to prevent malware infiltration and mitigate vulnerabilities. While these established approaches provide valuable capabilities for tracking data flow and generating analytical logs, they face significant limitations when confronted with the sheer volume of information requiring real-time processing to detect sophisticated threats and vulnerabilities effectively.

The inherent reactive nature of traditional cybersecurity solutions introduces critical operational challenges. The reliance on human expertise for complex analysis tasks creates bottlenecks that are susceptible to cognitive bias, human error, and fatigue-induced oversight. Furthermore, the labor-intensive nature of manual threat assessment and policy enforcement limits scalability and response time, potentially leaving systems vulnerable during the critical window between the emergence of a threat and its detection.

Cybersecurity encompasses multiple complex domains, including wireless systems, networking infrastructure, and cloud technologies, requiring extensive expertise across various disciplines. The intricate and volatile nature of cyberspace makes it difficult to detect hidden threats without thorough analysis. Conventional methods typically identify only obvious threats, consuming a substantial amount of time and resources.

Recent advancements in Deep Learning (DL) architectures have revolutionized various domains, including finance, healthcare, education, and customer service. Corporations are increasingly integrating intelligent chatbots powered by Large Language Models (LLMs), i.e., Generative AI (GenAI), to achieve greater efficiency. These agents can process vast amounts of data instantaneously, comprehend complex queries, and provide precise responses that often resolve problems

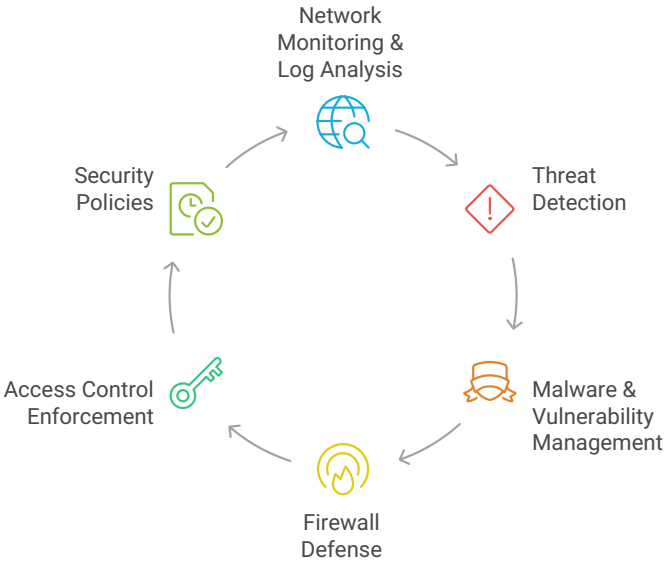

Figure 1: Traditional reactive cybersecurity framework illustrating the conventional workflow from threat detection through policy enforcement.

without requiring human intervention. The versatility of GenAI extends to the field of Computer Science (CS), where it assists programmers and analysts with code writing and reviewing [1], as well as data processing and analytics [2, 3].

These recent developments in GenAI have produced models capable of processing textual information through natural queries, employing vectorized databases for enhanced context through Retrieval Augmented Generation (RAG) [4], and providing coherent, logical responses. These characteristics make modern LLMs particularly suitable for strengthening security within digital systems, as they can efficiently process system configurations and logs, provide concise reports, and respond immediately to critical scenarios.

Despite their potential, general-purpose LLMs lack the specialized knowledge required for effective cybersecurity analysis. We address this limitation by enhancing the cybersecurity domain knowledge of general-purpose LLMs through Domain-Adaptive Continuous Pretraining (DAP), providing them with the expertise necessary to understand domain nuances while avoiding overfitting or knowledge loss. Our approach employs constrained training parameters and appropriately sized datasets to achieve this balance. Specifically, we address the following:

> **Research Question:** *Can efficient domain adaptation enable cybersecurity specialization of pretrained LLMs using substantially smaller datasets than traditional pretraining approaches?*

To investigate this systematically, we undertake four principal objectives: (i) preparing a comprehensive cybersecurity corpus, (ii) selecting appropriate open-access LLMs for adaptation, (iii) implementing an efficient domain-adaptation pipeline, and (iv) rigorously assessing the resulting models through established benchmark datasets.

Contribution. This paper addresses the gap between general-purpose LLMs and domain-specific cybersecurity requirements through a novel systematic domain adaptation methodology. Our investigation demonstrates that specialized cybersecurity knowledge can be effectively instilled in pretrained models using carefully curated, resource-efficient training approaches. Our primary contributions are summarized as follows:

1. We develop a systematic methodology for cybersecurity domain adaptation by instilling specialized knowledge through constrained training parameters, conservative learning rates, and limited training epochs (2-3).

2. We curate a specialized 126-million-word cybersecurity dataset from authoritative standards (e.g., ISO, NIST), academic literature, and technical documentation, providing comprehensive domain coverage while maintaining computational feasibility for domain adaptation.

3. We demonstrate competitive performance using substantially smaller datasets (118.8 million tokens) compared to existing specialized models (up to 5 billion tokens), challenging prevailing assumptions about data requirements for effective LLM domain specialization.

4. We systematically evaluate domain adaptation effectiveness across three decoder-based architectures (*Llama-3.1-8B*, *DeepSeek-R1-Distill-Qwen-14B*, and *Llama-3.3-70B-Instruct*), providing empirical insights into scale-dependent learning dynamics and optimal resource allocation strategies.

5. We provide empirical evidence demonstrating that DAP significantly outperforms parameter-efficient baselines (Parameter Efficient Fine-Tuning (PEFT)/Low-Rank Adaptation (LoRA)) in cybersecurity tasks.

Our domain-adapted models achieve higher performance across established benchmarks (CTI-MCQ, CyberMetric, SecEval), with the *Llama-3.3-70B-Ins-DAP* model obtaining accuracies of 0.718, 0.933, and 0.864, respectively, surpassing specialized cybersecurity models including *Llama-Primus-Base* [5] and *Foundation-Sec-8B* [6]. Additionally, our approach introduces key technical advances, where we utilize Fully Sharded Data Parallel (FSDP) technique for computational scalability, conservative hyperparameter selection to mitigate catastrophic forgetting while enabling effective knowledge acquisition, and a comprehensive evaluation framework spanning multiple cybersecurity subdomains. These contributions establish foundational capabilities for developing specialized cybersecurity Artificial Intelligence (AI) assistants capable of threat analysis, vulnerability assessment, and security documentation generation.

Paper Organization. Section 2 provides the theoretical foundation covering transformer architectures, domain adaptation methodologies, and GenAI applications in cybersecurity. Sections 3 and 4 present our systematic domain adaptation

framework and distributed training implementation. Section 5 evaluates model performance through comprehensive benchmarking against baseline and specialized cybersecurity models. Section 6 synthesizes empirical findings within the broader domain-specific LLM research landscape. Section 7 consolidates contributions and identifies future research directions in efficient domain adaptation methodologies.

2. Preliminaries and Related Work

This section establishes the conceptual foundations necessary for understanding the domain adaptation of LLMs for cybersecurity applications. We begin by examining the evolution of language understanding technologies, proceed to analyze transformer architectures that power modern language models, explore domain adaptation methodologies, and review relevant research in applying these technologies to cybersecurity challenges. Following this, we introduce an evaluation framework comprising five dimensions for comparing LLMs domain adaptation in cybersecurity. Finally, we review existing research in cybersecurity-focused language model adaptation, organizing approaches by architectural paradigm and identifying research gaps that motivate our investigation into data-efficient domain specialization.

2.1. Language Understanding

Natural Language Processing (NLP) represents a core domain of AI focused on processing natural text for decision-making applications. Traditional statistical techniques such as N-Gram and TF-IDF have been employed for tasks including text classification, information retrieval, and language modeling [7, 8, 9]. Despite their utility for applications such as spam filtering, these classical approaches exhibit significant limitations: they struggle with high-dimensional data, demonstrate vulnerability to data sparsity, and lack a meaningful contextual understanding of semantic relationships. The emergence of DL architectures enabled a fundamental change in NLP. Deep Neural Networks (DNNs) such as Recurrent Neural Networks (RNNs) [10] and Long Short-Term Memory (LSTM) [11] transcended the constraints of statistical methods by efficiently processing large text corpora and extracting semantic meaning. These architectures have demonstrated exceptional performance across diverse applications, including speech recognition [12], sequence-to-sequence modeling for machine translation [13], and Named Entity Recognition (NER) [14].

2.2. Transformer Architectures

The transformer architecture introduced by Vaswani et al. [15] fundamentally revolutionized language processing. Unlike previous sequential processing methods, transformers utilize self-attention mechanisms within an encoder-decoder framework, allowing for parallel text processing to generate contextually rich representations. Transformer models generate output by predicting sequential tokens until reaching a designated end token or the model's context limit. For each prediction, the model produces a probability distribution across its vocabulary, selecting the most likely token via predefined algorithms. While context windows constrain token generation capacity, various methodologies exist to expand effective context, including input truncation and positional embedding adjustments [16].

Encoder-Decoder Framework. The transformer architecture's encoder-decoder framework enables parallel processing through multi-head attention mechanisms. The encoder transforms input text into high-dimensional, context-rich internal representations (hidden states), which the decoder then processes to generate appropriate output text for tasks such as language translation [17].

During training, input sequences pass through an embedding layer that converts text into dense vectors of uniform length, capturing semantic and syntactic characteristics. These embeddings traverse attention mechanisms and feed-forward networks to produce context-enriched encodings, which the decoder's cross-attention mechanism subsequently utilizes. The decoder's masked self-attention ensures that each token attends only to itself and previously generated tokens, enabling coherent response generation.

2.3. Large Language Models

LLMs typically employ transformer architectures that are pretrained on extensive text corpora collected from diverse sources. While maintaining architectural similarities to standard transformers, these models incorporate numerous attention layers that enable the extraction of complex contextual and syntactic information. Contemporary LLMs generally utilize either encoder-only or decoder-only architectures, although some implementations, such as T5 [18] and BART [19], leverage both components. The T5 approach, for instance, conceptualizes all natural language tasks as text-to-text transformations, enabling fine-tuning for specific applications, such as question answering or summarization, following general pretraining.

Encoder-only Models. Encoder-only architectures stack multiple transformer encoder components, as exemplified by Bidirectional Encoder Representations from Transformers (BERT) [20] and Robustly Optimized BERT Approach (RoBERTa) [21]. These models transform raw text into semantically rich encodings that capture the relationships between tokens. Such architectures excel at tasks requiring deep textual understanding for contextual prediction, including NER and sentiment analysis [22]. Encoder-based models typically train through Masked Language Modeling (MLM), wherein certain tokens are randomly masked before embedding, with the model learning to infer these masked elements. This approach enables the model to develop robust semantic representations that can subsequently support classification tasks [23].

Decoder-only Models. Decoder-only models specialize in generative tasks, including text generation and language modeling. Unlike encoder-only architectures that primarily encode contextual information, these models learn to produce coherent, contextually enriched outputs based on input sequences. Meta's Large Language Model Meta AI (LLaMA) [24] and OpenAI's Generative Pre-trained Transformer (GPT) 4 [25]

exemplify this architecture, demonstrating exceptional performance across programming, question-answering, and language translation tasks [26]. These architectures employ autoregressive methodologies, predicting each subsequent token based on preceding tokens through Causal Language Modeling (CLM). During training, the model learns to predict the next token for each position in the sequence, optimizing weights to capture contextual patterns, enabling coherent output generation.

2.4. Evaluation Framework for Cybersecurity Domain Adaptation

To compare LLMs domain adaptation in cybersecurity, we establish an evaluation framework based on five dimensions that influence practical deployment and research advancement:

- **Data Requirements.** Dataset size (tokens/words), source diversity (e.g., standards, literature, documentation), and the relationship between data scale and performance. This criterion is crucial for resource-constrained scenarios where large corpora may be unavailable.
- **Computational Resources.** GPU requirements, training duration, memory footprint, and costs. These factors determine both accessibility and feasibility for academic groups and smaller organizations.
- **Architecture.** Encoder-only (classification, extraction), decoder-only (generation, conversation), or encoder-decoder designs, and whether employing full continuous pretraining versus PEFT [33] methods.
- **Domain Coverage.** Breadth and depth of cybersecurity knowledge, ranging from narrow subdomains (e.g., secure code generation, threat intelligence) to comprehensive multi-disciplinary coverage.
- **Evaluation.** Benchmark diversity and breadth (single versus multiple datasets) affecting reliability and generalizability of performance claims.

These criteria guide our comparative analysis and identify research gaps motivating our investigation.

2.5. Related Work

We review cybersecurity-focused language model adaptation and organizing approaches by architectural paradigm.

2.5.1. Traditional Deep Learning Approaches

Traditional DL methods demonstrate strong performance on specific tasks: Lee et al. [34] developed SeqDroid for Android malware detection, while Tang et al. [35] employed Gated Recurrent Unit Recurrent Neural Network (GRU-RNN) for Software Designed Networks (SDN) intrusion detection. However, these approaches lack natural language comprehension, which limits their applicability to semantic tasks such as documentation analysis or threat report generation. Recent LLM advances address these limitations. Diaf et al. [36] achieved 98% detection accuracy combining LLMs with LSTM for Internet of Things (IoT) traffic analysis, integrating both encoder and decoder architectures.

2.5.2. Encoder-only Architectures

Encoder-only models excel at representation and classification but are constrained in generative capabilities. Aghaei et al. [27] developed SecureBERT through BERT adaptation with specialized tokenization, achieving superior sentiment assessment and terminology recognition compared to SecBERT [37]. Their subsequent SecureBERT 2.0 [28] builds upon this foundation by leveraging the efficient ModernBERT architecture, which is pre-trained on a comprehensive corpus of 13.6B tokens spanning both cybersecurity text and code. Bayer et al. [29] introduced CySecBERT, trained on diverse sources (e.g., web articles, social media, papers, vulnerability databases), explicitly validating catastrophic forgetting mitigation (a notable contribution to general capability preservation). Ranade et al. [30] developed CyBERT for threat intelligence, demonstrating enhanced NER and Cybersecurity Knowledge Graph (CKG) completion performance. While these approaches provide strong domain-specific understanding with moderate computational requirements and open-source availability, architectural constraints limit deployment to discriminative tasks.

2.5.3. Decoder-only Architectures

Decoder architectures enable generative applications beyond classification. He et al. [31] proposed SafeCoder, which combines instruction tuning with security-focused fine-tuning on secure/insecure code pairs, thereby increasing StarCoder-1B's secure generation capability from 62.9% to 92.1%. Shestov et al. [38] fine-tuned WizardCoder [32] for Java vulnerability detection with high Receiver Operating Characteristic Area Under the Curve (ROC-AUC) and improved convergence. Fayyazi et al. [39] augmented GPT-3.5 with RAG for MITRE AT&ACK analysis, demonstrating decoder superiority over encoder-only models post-Supervised Fine-Tuning (SFT). While these approaches demonstrate strong performance with modest computational requirements (1B-scale models) or no retraining (RAG), they either target narrow domains (e.g., code security) or rely on proprietary models, thereby limiting reproducibility.

2.5.4. Cybersecurity Specialization

Yu et al. [5] introduced Llama-Primus-Base, the most comparable work to our investigation, which employs a similar DAP methodology using 2.77 billion tokens from Primus-FineWeb. The model achieves 0.667 (CTI-MCQ), 0.866 (CyberMetric), and 0.500 (SecEval) accuracy under 5-shot evaluation with full open-source availability. Similarly, Kassianik et al. [6] developed Llama-3.1-FoundationAI-SecurityLLM-Base-8B through continued pretraining on 5 billion tokens, achieving scores of 0.662 (CTI-MCQ) and 0.848 (CyberMetric) under a 5-shot evaluation. Both approaches enable comprehensive coverage but raise questions about data efficiency, specifically whether equivalent specialization can be achieved through more focused curation.

Table 1: Comparative analysis of cybersecurity domain-adaptive Language Models. Our approach demonstrates efficient and generalizable specialization, contrasting with prior task-specific or data-intensive approaches.

Ref. (Model)	Domain Adaptation	Generative Capability	Efficiency Focused	Adaptation Method	Evaluation Focus / Objective	Dataset Size	Training Duration [Hrs]	Epochs	Resources	Parameter Size
[27] (SecureBERT, 2022)	✓	✗	✗	Domain-Adaptive Pretraining (DAPT)	Cyber text understanding and NER	1.1B words	100	-	8 Tesla V100 GPUs	~125M (same as RoBERTa-base)
[28] (SecureBERT 2.0, 2025)	✓	✗	✓	ModernBERT encoder pretraining on cyber text + code	Document embedding, entity recognition, vulnerability detection	≈13.6B text + 53M code tokens	-	20	Multi-GPU infrastructure (specific GPU type not mentioned)	- (based on ModernBERT)
[29] (CySecBERT, 2022)	✓	✗	✗	DAPT on cybersecurity corpus	Context modeling and CTI classification	~528M tokens (4.3M entries)	-	30	4 Nvidia Tesla V100 GPUs	~110M (same as BERT-base)
[30] (CyBERT, 2021)	✓	✗	✗	Fine-tuning via MLM on CTI corpus	Entity recognition, attack classification, and CKG completion	~17K documents	-	4 (for NER task)	-	~110 Million
[31] (SafeCoder, 2024)	✓	✓	✗	Instruction + Security Fine-tuning	Secure code generation and safety alignment	1.2K pairs (367 tokens avg.)	-	2 (5 for CodeLlama-7B)	3 H100 (80GB) + 8 A100 (40GB) GPUs	Varies by base model (1B to 7B)
[32] (WizardCoder, 2024)	✓	✓	✓	LoRA Fine-tuning for Vulnerability Detection	Vulnerability detection in Java code (balanced & imbalanced)	1,334 (X1: Vulnerability pairs) + 22,945 (X1 + P3: With safe code added)	-	50	-	~13B (base WizardCoder) + 25M (LoRA)
[5] (Llama-Primus-Base, 2025)	✓	✓	✗	Continued Pretraining (PRIMUS-SEED + FINEWEB)	Cyber reasoning and knowledge comprehension	~2.77B tokens	30	2	4 nodes, each with 8x H200 GPUs (32 H200 GPUs total)	8 billion
[6] (Foundation-Sec-8B, 2025)	✓	✓	✗	Continued Pretraining on curated cybersecurity corpus	Cyber reasoning, vulnerability classification, secure code QA	~5B tokens	-	-	Multi-node compute cluster (using DeepSpeed)	9 billion
This work	✓	✓	✓	**Domain-Adaptive Continuous Pretraining (DAP)**	**General Cybersecurity Awareness**	**118.8M tokens**	**0.65 hours (for 8B), 7 hours (for 14B), 21 hours (for 70B)**	**2**	**32 NVIDIA A10G (for 8B), 16 NVIDIA H100 (for 70B and 14B)**	**8B, 14B, 10B**

2.5.5. Research Gap

Data Efficiency. Existing comprehensive approaches employ massive datasets (Llama-Primus: 2.77B tokens; FoundationAI: 5B tokens), while efficient approaches target narrow subdomains. Our work investigates whether broad adaptation is achievable with 118.8 million tokens (a 23-fold reduction) while maintaining comprehensive coverage across standards, literature, and documentation.

Scale-Dependent Learning Dynamics. Limited investigation exists into adaptation effectiveness across model scales under comparable training regimes. Our systematic evaluation across three architectures (Llama-3.1-8B, DeepSeek-R1-Distill-Qwen-14B, Llama-3.3-70B-Instruct) provides empirical insights into optimal resource allocation strategies.

Catastrophic Forgetting. While CySecBERT explicitly validated general capability preservation, most domain adaptation research does not address any potential degradation of foundational language understanding. Our conservative training approach—employing low learning rates (1×10^{-6}), limited epochs (2-3), and frozen embeddings—was specifically designed to mitigate this risk, though comprehensive validation on general benchmarks remains a direction for future work due to resource constraints (as discussed in Section 6).

Our positioning within this landscape emphasizes practical efficiency, achieving competitive cybersecurity specialization with substantially reduced data requirements while maintaining decoder-only generative capabilities essential for tasks such as threat analysis, vulnerability assessment, and security documentation generation.

3. Methodology

This section presents our proposed LLMs DAP methodology for cybersecurity applications. Our end-to-end pipeline is depicted in Figure 2, which illustrates the systematic workflow comprising seven integrated components that transform raw cybersecurity data into specialized domain-adapted model weights.

The pipeline orchestrates three primary phases: *initialization*, *distributed training*, and *persistence*. The initialization phase establishes the foundation by loading pretrained model architectures and their associated tokenizers, followed by preprocessing domain-specific data into standardized formats suitable for parallel processing. The distributed training phase implements FSDP to enable efficient weight distribution and gradient synchronization across GPU clusters, addressing the computational demands of large-scale language models. Training orchestration manages the iterative optimization process, including forward propagation, loss computation, gradient backpropagation, and periodic checkpointing for fault tolerance and reliability. The persistence phase preserves the complete training state and converts adapted weights into formats compatible with downstream evaluation and deployment.

This modular pipeline architecture enables scalable domain adaptation while maintaining reproducibility and computational efficiency across varying model scales and hardware configurations.

3.1. Methodological Rationale

Our investigation employs DAP rather than adapter-based approaches such as PEFT [33] or LoRA [40] to achieve a fundamentally different objective: instilling an intrinsic understanding of cybersecurity within the model's foundational representations. While adapter-based methods demonstrate strong performance for task-specific optimization (e.g., vulnerability classification, log parsing), they introduce specialized parameters on top of frozen base models without modifying the underlying domain comprehension. This architectural constraint limits their capacity to embed deep semantic understanding and domain-specific reasoning patterns into the model's core representations.

DAP, conversely, enables comprehensive knowledge integration through continued pretraining on domain-specific

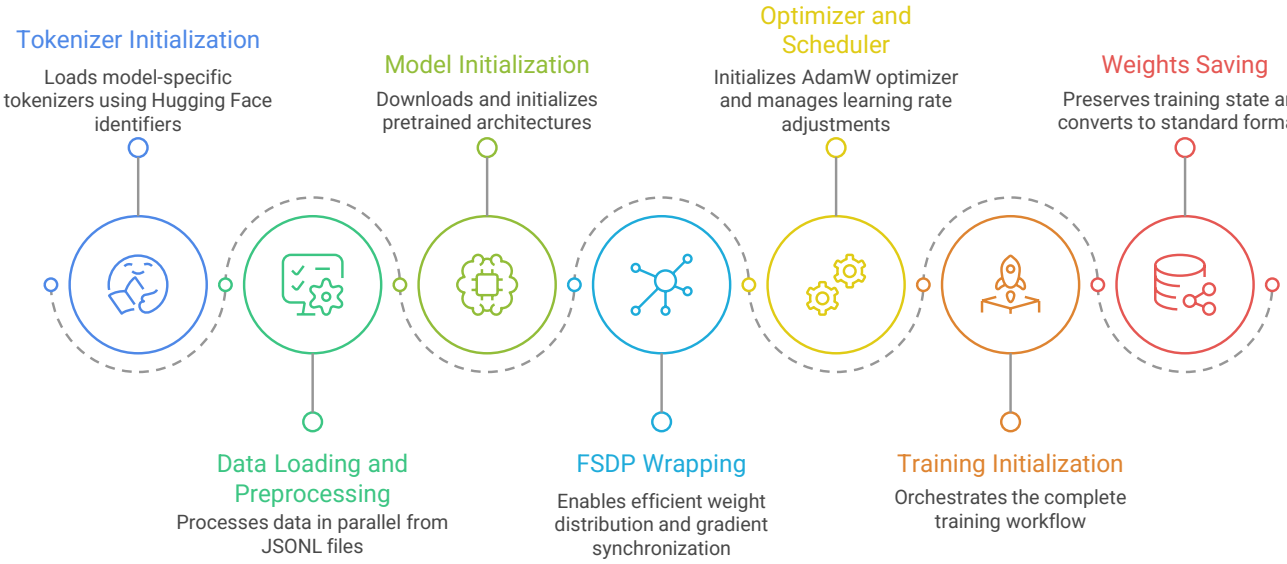

Figure 2: Our domain-adaptive continuous pretraining pipeline. The workflow illustrates the systematic transformation from raw cybersecurity corpus collection through preprocessing, JSONL formatting, and distributed training to produce specialized domain-adapted model weights.

corpora, allowing the model to internalize cybersecurity terminology, conceptual relationships, and contextual patterns at the representational level. This foundational enhancement proves particularly valuable for complex reasoning tasks that require nuanced domain understanding, such as threat analysis, security documentation generation, and multi-step vulnerability assessments. The resulting domain-adapted models subsequently serve as robust foundations for downstream specialization through instruction tuning, PEFT, or task-specific fine-tuning, enabling efficient adaptation to specific applications including intrusion detection, secure code generation, and compliance analysis.

Our methodology thus prioritizes building foundational cybersecurity-aware language models that possess inherent domain expertise, creating versatile bases for subsequent specialization across diverse cybersecurity applications rather than optimizing for any single narrow task.

3.2. Dataset Preparation

The foundation of our DAP methodology rests upon a meticulously curated cybersecurity corpus that achieves an optimal balance between domain specificity and comprehensive coverage. In contrast to pretraining datasets that encompass petabyte-scale collections, our domain adaptation strategy employs a focused dataset specifically designed to instill domain-specific terminologies and contextual nuances. This deliberate constraint in dataset scope serves dual purposes: ensuring sufficient exposure to cybersecurity concepts while aiming to reduce the risk of the model overspecializing and losing its general language understanding capabilities, though preservation of general capabilities was not empirically validated in this work.

Data Curation. We developed a custom cybersecurity corpus through the systematic collection of materials from three primary categories. Standards and regulations form the cornerstone of our dataset, incorporating authoritative documents from recognized bodies such as NIST and ENISA. These regulatory frameworks provide a solid foundation in formal cybersecurity terminology and compliance requirements. Research papers sourced from academic platforms, including arXiv, contribute cutting-edge insights and emerging threat landscapes to our corpus. Technical literature encompassing books on network security, malware analysis, and related domains completes our dataset, ensuring comprehensive coverage of both theoretical foundations and practical applications. This targeted curation approach ensures a thorough representation of cybersecurity subdomains while maintaining a dataset scope suitable for domain adaptation rather than full model retraining.

Data Processing Pipeline. Raw documents undergo systematic preprocessing through a pipeline designed to maximize data quality while preserving contextual integrity. The process begins with extracting text from PDF documents and text files, followed by conversion to JSON format for structured representation. Subsequently, we apply Regular Expression (RegEx) patterns for comprehensive noise removal, eliminating extraneous elements such as page numbers, citations, and empty lines that could interfere with the learning process. The cleaned data then undergoes transformation to JSONL format, ensuring compatibility with modern training frameworks. Throughout this process, we maintain paragraph-level segmentation to preserve contextual coherence, recognizing that cybersecurity concepts often require surrounding context for proper interpretation. Notably, no manual labeling is required, as models undergo unsupervised training via CLM, which significantly reduces preprocessing overhead while maintaining training effectiveness.

3.3. Model Selection and Architecture

Our methodology employs decoder-only architectures, chosen for their demonstrated performance in autoregressive text generation tasks compared to encoder-only or encoder-decoder alternatives. This architectural decision stems from decoder models' streamlined approach to next-token prediction, which aligns naturally with our objective of generating contextually appropriate cybersecurity responses. The selection process prioritizes three critical criteria that balance practical constraints with performance requirements.

Computational efficiency remains paramount given resource constraints typical in research environments, necessitating models that deliver optimal performance without requiring extensive Graphical Processing Unit (GPU) clusters. Finally, we consider the ratio of pretraining corpus size to model parameters, ensuring that the selected models possess a sufficient knowledge base to support effective domain adaptation.

Initial experiments utilize LLaMA 3.2 variants with 1B and 3B parameters, models pretrained on approximately 9 trillion tokens. This extensive pretraining provides a substantial knowledge foundation while maintaining computational feasibility for domain adaptation. Subsequent experiments scale to larger models, like 8B parameter models, allowing systematic evaluation of performance improvements achievable through increased model capacity. This progressive scaling strategy enables empirical determination of the optimal trade-off between computational requirements and domain adaptation effectiveness.

3.4. Domain-Adaptive Training Framework

Our training methodology leverages distributed multi-GPU processing to accommodate the computational demands of modern LLMs while implementing carefully calibrated strategies to preserve general knowledge during domain specialization.

Training Configuration. The configuration begins with tokenization using pretrained tokenizers without modification, based on the reasonable assumption that tokenizers trained on trillions of tokens possess comprehensive vocabulary coverage for our domain-specific corpus. This approach avoids potential vocabulary fragmentation that could arise from retraining tokenizers on limited domain data. Specifically, we freeze the embedding layer throughout training to preserve learned token representations acquired during pretraining. This strategy prevents degradation of fundamental language understanding while allowing higher layers to adapt to domain-specific patterns.

Learning rate selection requires particular attention in domain adaptation scenarios. We deliberately employ small learning rates to facilitate gradual weight adjustments, with the intention of mitigating catastrophic forgetting that could result from aggressive parameter updates, although this preservation has not been empirically validated. Similarly, we limit training to a minimal number of epochs, typically one to three complete passes through the dataset, thereby aiming to avoid overfitting on domain-specific patterns that could compromise generalization capability.

Training Process. The domain adaptation process follows a systematic protocol designed for reproducibility and optimal results. Training commences with the initialization of pretrained model weights, establishing the foundation of general language understanding upon which domain knowledge is built. Following initialization, we freeze the embedding layer to maintain vocabulary representations while allowing other parameters to adapt. Training proceeds with our carefully controlled Learning Rate (LR) schedule, ensuring gradual weight adjustments that preserve existing knowledge while incorporating domain expertise. Upon completion, adapted model weights are saved in appropriate formats such as binary or safe tensors, facilitating efficient deployment and further experimentation.

3.5. Evaluation Framework

Our comprehensive evaluation methodology includes both training-phase metrics and post-training benchmarks, providing a multifaceted assessment of domain adaptation effectiveness across various dimensions of model performance.

Training Phase Evaluation. During DAP, we continuously monitor perplexity as the primary optimization metric, calculated as $\text{PPL} = \exp\left(-\frac{1}{N}\sum_{i=1}^{N}\log_2 q(x_i)\right)$, where N represents the total number of tokens in the sequence and $q(x_i)$ denotes the probability assigned by the model to token x_i. Lower perplexity values indicate higher model confidence in next-token prediction, serving as a proxy for improved domain understanding. However, we recognize that perplexity alone cannot fully assess contextual comprehension or practical applicability, necessitating comprehensive post-training evaluation.

Post-Training Assessment. Domain-adapted models undergo rigorous evaluation using established cybersecurity benchmarks to quantify improvement in domain-specific understanding. Performance assessment on datasets such as CyberMetric [41] employs Multiple Choice Question (MCQ) format questions that test various aspects of cybersecurity knowledge. We conduct direct accuracy comparisons between original and adapted models on identical evaluation sets, providing clear quantitative measures of domain adaptation effectiveness.

4. Implementation

The domain-adaptive pretraining (DAP) of large language models (LLMs) presents substantial computational challenges due to their massive parameter counts and quadratic attention complexity $O(n^2)$, where n represents the prompt length. For instance, GPT-2 employs 768-dimensional vectors per token to capture embedding and positional information, resulting in prohibitive resource requirements that necessitate distributed training environments with weight sharding and aggregation mechanisms. Our training implementation employs a mixed-precision strategy to optimize computational efficiency while preserving numerical stability. Models are loaded in half-precision formats

(FP16/BF16) to reduce memory footprint, while most computations are performed in half-precision for enhanced throughput. Critical operations, including loss calculation, utilize full-precision (FP32) with gradient scaling to prevent numerical instability and gradient underflow that would otherwise compromise training convergence.

This section presents a comprehensive implementation framework for a cybersecurity-focused DAP that utilizes AWS SageMaker's P5 instances [42] equipped with 8 NVIDIA H100 GPUs, addressing both technical constraints and domain-specific requirements.

4.1. Dataset Construction and Preprocessing

The absence of comprehensive cybersecurity-focused datasets led to the construction of a custom corpus comprising 126 million words (132 million tokens using LLaMA tokenizer). This corpus integrates diverse sources to ensure comprehensive domain coverage while maintaining linguistic variety essential for effective model adaptation.

Dataset Coverage. Our curated dataset encompasses three primary categories, each contributing unique contextual information: (i) *Standards & Regulations:* Industry-recognized documents from organizations such as ISO provide formal compliance frameworks and security guidelines. These documents enable the model to extract contextual information and generate responses adhering to established standards, particularly valuable for compliance analysis during SFT for downstream tasks. (ii) *Papers:* Academic publications encompass cutting-edge cybersecurity research, providing formal, well-structured, fact-based content. This exposure enables the model to learn academic writing styles, facilitating the generation of concise summaries and technical reports grounded in sound reasoning and logic. (iii) *Books:* Technical literature presents concepts at varying complexity levels with extended context lengths. This diversity enables the model to develop longer contextual memory while learning to generate user-friendly responses across different writing styles, from descriptive to explanatory formats.

During our experimental phase, the Primus-Seed and Primus-FineWeb [5] datasets became available. While Primus-FineWeb's 2.57 billion tokens posed risks of overfitting, the expert-curated Primus-Seed (0.2 billion tokens) dataset, containing content from reputable sources including MITRE, was preserved for potential integration in subsequent fine-tuning stages or downstream task optimization.

4.1.1. Pre-processing

The heterogeneous nature of our curated documents required comprehensive preprocessing to eliminate noise and standardize formats. Our pipeline addresses three critical aspects:

Data Filtering. Documents collected from sources, including the NIST Open Document Library and arXiv repositories, underwent multi-stage filtering. Manual curation combined with AI-based classification addressed the challenge of ambiguous domain categorization, particularly for research papers with multiple keywords. The classification prompt specifically evaluated strict adherence to cybersecurity or its subdomains. RegEx filtering eliminated patterns already learned during pretraining (i.e., email formats, phone numbers, and references), maintaining minimal noise-to-content ratios to prevent memorization of irrelevant patterns.

Data Formatting. The multi-format corpus (PDFs, text files, structured and unstructured content) required standardization for tokenization. Text extraction and filtering preceded segmentation into fixed sequence lengths, minimizing padding overhead during tokenization. The processed content was formatted as JSON Lines (JSONL) entries (depicted in Figure 3), facilitating efficient data loading.

```
{"text": "reprehenderit dolore consectetur
 ↪  ea \\n ut elit cillum in minim aliquip
 ↪  in id voluptate qui eu"}
{"text": "cupidatat eu in quis officia in
 ↪  \\n irure eiusmod qui enim aute quis
 ↪  officia tempor dolor"}
{"text": "mollit \\n occaecat magna ullamco
 ↪  aute quis tempor consectetur dolor
 ↪  commodo in laborum laborum cupidatat
 ↪  minim"}
```

Figure 3: Sample JSONL structure.

Context window constraints of LLMs necessitated sequence truncation to specified lengths (1024 or 2048 tokens), as illustrated in Figure 4. This approach simplifies context retention across sequences by eliminating random splits.

```
tensor([128000, 345, 887, 29892, 278, 13,
 ↪  ..., 128001, 128001])
tensor([128000, 1045, 2572, 1037, 2173,
 ↪  2000, 2424, ..., 128001, 128001])
tensor([128000, 2023, 2003, 1037, 3978,
 ↪  7099, 6251, 6251, ..., 128001, 128001])
```

Figure 4: Sample tokenized input IDs (LLaMA3 tokenizer).

Data Splitting. The dataset allocation follows a 90-10 train-test split, with the training portion further stratified into three variants with $\approx$ 1 million, $\approx$ 50 million, and $\approx$ 118.8 million tokens, respectively. Random shuffling during variant creation prevents ordering bias, ensuring robust learning across different model scales.

4.2. Domain Adaptive Continuous Pretraining Pipeline

Model Selection. Model selection balanced multiple constraints: architectural diversity, open-source availability, and computational feasibility. While state-of-the-art models like GPT-4 and Claude 3.7 [43] remain proprietary, we selected open-source alternatives demonstrating strong performance across mathematical, linguistic, and reasoning tasks. Initial pipeline validation utilized LLaMA-3.2-3B before proceeding to larger architectures. Table 2 presents the final model

selection with corresponding dataset allocations, designed to evaluate scale-dependent learning dynamics.

Table 2: Approximate training-data size for each model.

Model name	Approximate dataset size in million [$\times 10^6$] tokens
Llama-3.1-8B	1
DeepSeek-R1-Distill-Qwen-14B (DSR1D-Qwen-14B)	50
Llama-3.3-70B-Instruct (Llama-3.3-70B-Ins)	118.8

Resource requirements for model loading follows: Model Size $= 4N \times 10^9$ Bytes, where $N \in \{1, 3, 8, 13, 14, 32, 70, 671, \dots\}$ represents billions of parameters. Full training memory requirements approximate:

$$\text{Memory Requirement} \approx \alpha \times \text{Model Size}, \quad \text{where } \alpha \in [3, 5] \tag{1}$$

This scaling factor accounts for gradient storage, activations, and optimizer states during forward and backward passes. While half-precision formats (*float16* or *bfloat16* [44]) reduce memory requirements by 50%, models like DeepSeek-R1-671B remain computationally prohibitive given current resource constraints.

4.2.1. Hyperparameters Configuration

Hyperparameter selection influences learning dynamics, particularly for LLMs, where the massive number of parameters amplifies the risks of overfitting. Our configuration below prioritizes stability while enabling effective domain knowledge acquisition.

Learning Rate. Configured at $\approx 1 \times 10^{-6}$, this conservative value prevents overfitting while enabling gradual adaptation to domain-specific patterns. This selection balances the competing objectives of knowledge acquisition and retention.

Maximum Sequence Length. Constrained to $\{1024, 2048\}$ tokens due to memory limitations in attention computation. Longer sequences enhance contextual understanding through self-attention mechanisms but proportionally increase memory consumption for attention matrices, activations, and gradients.

Batch Size. Per-device batch sizes of $\{1, 2\}$ accommodate GPU memory constraints. Distributed training across 4-8 GPU instances achieves effective parallelism. For instance, 4 compute instances with 4 GPUs each yield an effective batch size of $1 \times 4 \times 4 = 16$.

Gradient Accumulation Step. Mini-batch processing with gradient accumulation maintains computational efficiency while achieving effects comparable to larger batch sizes. Total effective batch size is calculated as Total Batch Size $=$ MBS $\times$ GAS $\times$ World Size. Where MBS denotes per-device mini-batch size, GAS represents gradient accumulation steps, and World Size is defined as World Size = Total Devices $\times$ GPUs per Device.

Additional configurations include (i) maximum 2-3 training epochs to prevent domain overfitting, (ii) cosine learning rate scheduling with warm-up to prevent exploding gradients, (iii) and AdamW optimizer [45] providing stable weight updates through decoupled weight decay regularization.

4.3. Model Fine-tuning

The distributed training pipeline addresses the computational infeasibility of single-device LLM training through systematic parallelization strategies. Our implementation leverages PyTorch's FSDP framework, combining the benefits of data and model parallelism through weight sharding with distributed batch processing. The implementation comprises the following key components:

1. **Tokenizer Initialization.** The AutoTokenizer API from the Transformers library loads model-specific tokenizers via Hugging Face identifiers (e.g., *meta-llama/Llama-3.1-8B*) or local storage paths for pre-saved configurations.

2. **Data Loading and Preprocessing.** PyTorch's DistributedSampler and DataLoader utilities enable parallel data processing from pre-formatted JSONL files. The pipeline generates identical input-output sequence pairs for unsupervised DAP, returning properly formatted tensor batches.

3. **Model Initialization.** AutoModelForCausalLM downloads and initializes pretrained architectures from Hugging Face Hub, automatically managing weight shard distribution across available devices.

4. **FSDP Wrapping.** The FullyShardedDataParallel API enables efficient weight distribution and gradient synchronization. This approach supersedes the Transformers Trainer utility due to enhanced customization capabilities for our specific requirements.

5. **Optimizer and Scheduler.** AdamW optimizer initialization via PyTorch's optim utility receives model parameters for weight updates. The learning rate scheduler manages adaptive rate adjustments throughout training phases.

6. **Training Initialization.** A custom Trainer class orchestrates the complete training workflow: dataset iteration, forward pass computation, loss calculation, gradient backpropagation, weight updates, and checkpoint persistence. Critical recovery capabilities store current step and world size information, enabling mid-epoch training resumption after interruptions.

7. **Weights Saving.** Checkpoint management preserves a comprehensive training state, including model weights, optimizer states, and scheduler configurations, using PyTorch's distributed API in sharded format. Post-training conversion via Transformers' *save_pretrained* API produces standard formats compatible with *from_pretrained* or *AutoModelForCausalLM* loading mechanisms, facilitating seamless downstream evaluation using utilities like *generate*.

5. Performance Evaluation

The evaluation of domain-adapted LLMs requires substantial computational resources due to their scale and complexity. For instance, loading the *Llama-3.1-8B* model in half-precision

requires approximately 29.9 GB of memory, calculated as GPU Memory Required $> 8 \times 10^9 \times 2$ bytes ≈ 29.9 GB. Larger models demand proportionally greater resources, necessitating cloud-based evaluation infrastructure. Although inference requirements are less stringent than domain adaptation, the GPU must simultaneously accommodate model parameters, tokenized inputs, and intermediate computations, making traditional Central Processing Unit (CPU)-based evaluation infeasible without quantization[1].

5.1. Experimental Setup

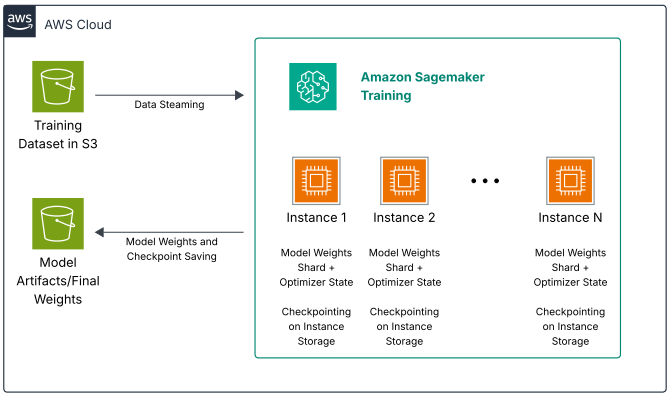

Figure 5: AWS reference architecture for distributed training and evaluation infrastructure.

Our distributed training and evaluation pipeline leverages the AWS reference architecture, illustrated in Figure 5, which is orchestrated through Amazon SageMaker to manage computational resources and coordinate data flow. The training dataset is stored persistently in an AWS S3 bucket and streamed directly to the SageMaker training cluster during the adaptation process. We employ a distributed data-parallel strategy, wherein SageMaker provisions multiple instances (Instance 1 through Instance N) that operate in parallel. Consistent with the FSDP methodology detailed in Section 4, model parameters (Model Weights Shard) and optimizer states are sharded across all available GPUs on these instances. Training checkpoints are saved to instance storage for efficient state recovery in case of interruptions. Upon completion, the final model artifacts, including the adapted weights, are aggregated and stored in a specific S3 bucket for subsequent evaluation and deployment.

For inference evaluation, our pipeline utilizes AWS SageMaker inference endpoints with mixed-precision configurations (*FP32*/*FP16*/*BF16*) employed during DAP and half-precision formats for memory-optimized evaluation. Table 3 presents the instance allocation strategy for each LLM architecture.

The *ml.g5.12xlarge* instances utilize NVIDIA A10G GPUs (24 GiB of memory per GPU, totaling 96 GiB), which are sufficient for 8B and 14B models in half-precision. The 70B model requires *ml.p4d.24xlarge* instances with NVIDIA A100 GPUs,

[1]Quantization can lead to performance degradation.

Table 3: Model to Instance Mapping (Inference)

Model	Instance
Llama-3.1-8B	*ml.g5.12xlarge*
DSR1D-Qwen-14B	*ml.g5.12xlarge*
Llama-3.3-70B-Ins	*ml.p4d.24xlarge*

optimized for *bfloat16* operations and providing enhanced computational capacity. Table 4 details the hardware specifications. The theoretical compute capacity per instance is computed as Compute Capacity = Num of GPUs $\times$ TFLOPS per GPU. Our implementation utilizes the SageMaker SDK with HuggingFace Transformers (version 4.48.0), PyTorch 2.3.0, and Python 3.11, enabling direct model deployment and evaluation through the established Application Programming Interfaces (APIs).

Table 4: AWS instance specifications [46]. (OD: On-Demand)

Instance	# GPUs	GPU Type	Mem/GPU (GiB)	TFLOPS (FP16)	(OD) Price/hr
ml.g5.12xlarge	4	NVIDIA A10G	24	31.52	$7.09
ml.p4d.24xlarge	8	NVIDIA A100	40	19.5	$25.251

Baseline Configuration Parameters. To validate the efficacy of full-parameter adaptation, we compared our approach against a PEFT baseline using LoRA. The LoRA configuration utilized a rank of $r = 16$, $\alpha = 32$, and dropout of 0.05, targeting the query, key, value, and output projection modules within the attention mechanism.

5.2. Decoding Configuration

To ensure reproducible and deterministic evaluation across all benchmarks, we employ greedy decoding with fixed parameters: `do_sample = False` and `temperature = 0`. This configuration eliminates stochastic sampling and probability scaling, producing a single deterministic output per prompt. By removing decoding randomness, our evaluation isolates model behavior and domain knowledge acquisition from variability introduced by sampling strategies, enabling consistent cross-benchmark comparisons. All reported results reflect single-pass evaluations under this standardized configuration.

5.3. Evaluation Benchmarks

We assess cybersecurity domain adaptation using three specialized datasets that comprehensively evaluate different aspects of cybersecurity knowledge. The CTI-MCQ dataset [47] comprises 2500 multiple-choice questions specifically designed to evaluate Cyber Threat Intelligence (CTI) understanding, testing models' comprehension of threat identification, analysis methodologies, and intelligence-driven security practices. The CyberMetric benchmark [41] comprises 2000 multiple-choice questions that assess general knowledge across various cybersecurity domains, offering comprehensive coverage of fundamental security principles, best practices, and technical competencies. Finally, the SecEval dataset [48] comprises over 2000 multiple-choice questions that focus on broad cybersecurity knowledge evaluation, spanning various security subdomains, including vulnerability assessment, risk management, and defensive strategies.

These benchmarks span cybersecurity standards, network and cloud security, cryptography, identity and access management, and threat mitigation strategies. Due to computational constraints, we evaluated the 70B model on reduced samples for CyberMetric (1000 questions) while maintaining full evaluation sets for other models and datasets.

5.4. Results and Analysis

We evaluate the model performance using standard accuracy metrics, i.e., Accuracy $= \frac{1}{N}\sum_{i=1}^{N} 1(y_i = \hat{y}_i)$. Here y_i and $\hat{y}_i$ are the ground truth and predictions from the models, respectively. N is the total data samples in the evaluation dataset. If the prediction and ground truth are the same $1(y_i = \hat{y}_i)$ evaluate to 1 otherwise 0. For SecEval, which permits multiple correct answers, we employ the Jaccard Index for per-question accuracy, i.e., Jaccard Index$(GT, Pred) = \frac{|GT \cap Pred|}{|GT \cup Pred|}$, where GT is the ground truth set and $Pred$ is the model response (extracted) answer set.

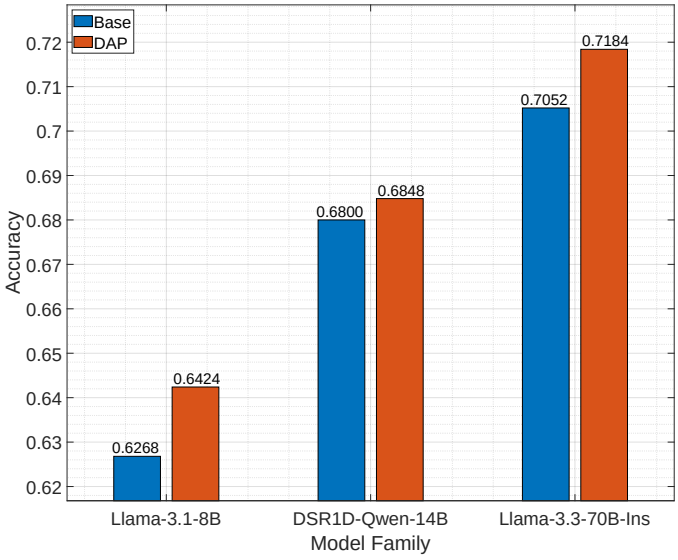

Figure 6: Performance comparison of base and domain-adapted models on CTI-MCQ benchmark. Domain adaptation yields consistent improvements across all architectures, with accuracy gains of 1.6% for Llama-3.1-8B, 0.5% for DSR1D-Qwen-14B, and 1.3% for Llama-3.3-70B-Ins.

Performance on CTI-MCQ. Figure 6 demonstrates the effectiveness of domain-adaptive continuous pretraining across all evaluated model architectures on the CTI-MCQ benchmark, which specifically assesses Cyber Threat Intelligence understanding. The results reveal consistent performance improvements following domain adaptation, with notable variations in the magnitude of improvement across different model scales. The Llama-3.1-8B model exhibits the most substantial relative improvement, achieving a 1.6% accuracy gain from 0.6268 to 0.6424, representing a meaningful enhancement in cybersecurity knowledge acquisition despite the model's smaller parameter count. The DSR1D-Qwen-14B architecture demonstrates more modest but consistent improvement with a 0.5% increase from 0.6800 to 0.6848. In contrast, the largest model, Llama-3.3-70B-Ins, achieves a 1.3% improvement, rising from 0.7052 to 0.7184, and ultimately attains the highest performance across all evaluated architectures. These results validate the efficacy of our domain adaptation methodology in enhancing specialized cybersecurity comprehension while maintaining architectural diversity in performance gains.

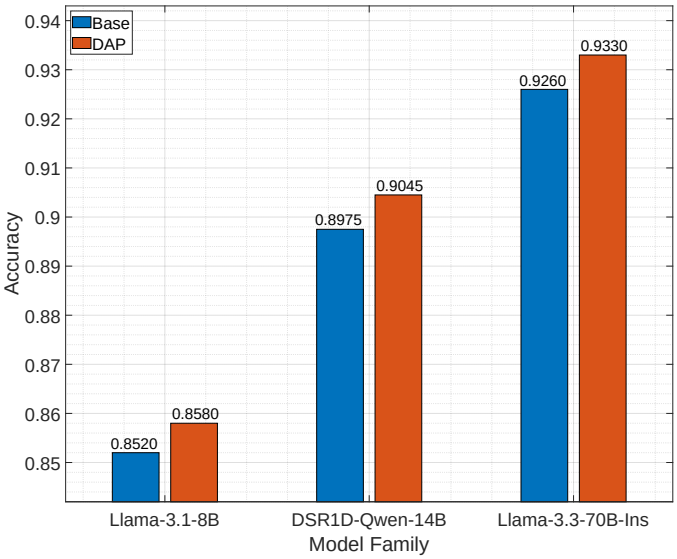

Figure 7: Performance comparison of base and domain-adapted models on CyberMetric (2000) benchmark. DAP consistently improves cybersecurity knowledge across all LLMs, with accuracy gains ranging from 0.6% to 0.7%.

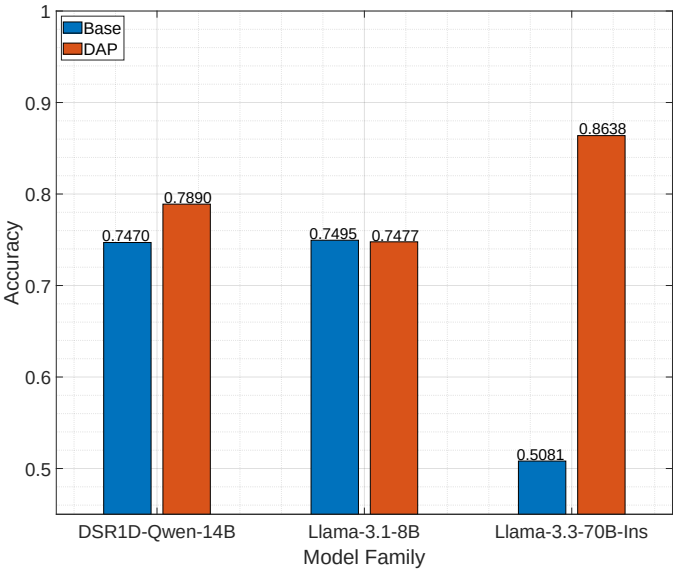

Figure 8: Performance comparison of base and domain-adapted models on the SecEval benchmark. Domain adaptation demonstrates significant improvements for DSR1D-Qwen-14B (0.747 to 0.789) and Llama-3.3-70B-Ins (0.508 to 0.864), while Llama-3.1-8B shows minimal change.

Performance on CyberMetric. Figure 7 illustrates the performance characteristics of our domain-adapted models on the CyberMetric benchmark, which evaluates general cybersecurity knowledge across diverse security domains. The results demonstrate remarkably consistent improvement patterns, with all three architectures achieving accuracy gains within a range of 0.6% to 0.7%. Specifically, the Llama-3.1-8B

model improves from 0.8520 to 0.8580, while the DSR1D-Qwen-14B achieves enhancement from 0.8975 to 0.9045, and the Llama-3.3-70B-Ins advances from 0.9260 to 0.9330. These improvements across different model scales imply that our domain adaptation approach effectively instills general cybersecurity knowledge regardless of architectural complexity. Notably, all models achieve relatively high baseline performance on this benchmark, with the domain adaptation process providing incremental yet consistent enhancements that validate the robustness of our training methodology across diverse cybersecurity knowledge domains.

Performance on SecEval. Figure 8 presents the most different results across architectures within our evaluation framework, revealing complex interactions between baseline model capabilities and domain adaptation efficacy on the SecEval benchmark. This evaluation dataset employs a sophisticated multi-answer format utilizing Jaccard Index scoring, which demands advanced instruction-following and contextual reasoning capabilities. The results demonstrate three distinct adaptation patterns that highlight fundamental relationships between pre-existing model alignment and domain specialization potential.

The DSR1D-Qwen-14B architecture demonstrates moderate yet meaningful progression from 0.7470 to 0.7890, reflecting its intermediate position between raw pretraining and full instruction alignment, with knowledge distillation foundations providing a stable platform for domain enhancement. In contrast, the Llama-3.1-8B model demonstrates minimal variation, i.e, ≈ 0.002 difference, between the base and DAP model. This indicates that this intermediate-scale architecture maintains a robust baseline performance that proves relatively resistant to further domain-specific enhancements, which could be attributed to the relatively small size of the DAP dataset, i.e., 1 million tokens, for an LLM with 8 billion parameters.

Remarkably, the Llama-3.3-70B-Ins model exhibits a significant improvement, increasing from 0.5081 to 0.8638, representing a 70% relative increase, which is the most substantial performance gain observed across our entire evaluation framework. This enhancement reveals that despite the model's extensive parameter count and sophisticated pre-training, the baseline architecture exhibited significant deficiencies in complex multi-answer reasoning tasks, which domain adaptation effectively addressed.

Key Takeaway. The aforementioned outcomes challenge conventional assumptions about model scale, training data requirements, and inherent capabilities. Our findings demonstrate that even large, instruction-tuned models can maintain specific domain-related weaknesses that targeted, resource-efficient continuous pretraining can systematically address. The substantial performance gains achieved with our focused adaptation approach, utilizing significantly smaller datasets than traditional domain specialization methods, demonstrate the viability of efficient domain adaptation strategies that prioritize judicious dataset curation over sheer data volume.

Semantic Similarity Analysis. To assess the contextual understanding capabilities of our domain-adapted models, we conducted semantic similarity evaluation for the 8B and 14B parameter architectures across all three cybersecurity benchmark datasets. During inference, models were prompted to generate explanatory responses of 20-30 tokens justifying their answer selections. We computed cosine similarity scores between the model-generated explanations and the ground-truth correct answers to quantify the degree of semantic alignment.

Figure 9 demonstrates consistent enhancement in contextual understanding following domain adaptation across all evaluated benchmarks. The results confirm that domain-adapted variants consistently achieve higher median cosine similarity scores compared to their baseline counterparts, with the most substantial improvements observed on the SecEval benchmark. Notably, the Llama-3.1-8B-DAP model exhibits an increase from 0.33 to 0.70 on SecEval, representing a 112% improvement that highlights the profound impact of cybersecurity-focused continuous pretraining on semantic comprehension. It is worth noting that despite having a 0.002 accuracy difference between the base and our DAP model (Figure 8), the cosine similarity is higher. Hence, indicating that the loss in accuracy might have stemmed from the limited domain knowledge in the small DAP dataset rather than the loss in domain understanding.

The DSR1D-Qwen-14B architecture displays more modest but consistent improvements across all benchmarks, with enhancements ranging from 0.05 to 0.06. These findings corroborate our primary performance metrics, providing additional evidence that domain adaptation successfully instills specialized cybersecurity knowledge while enhancing the models' ability to generate contextually appropriate explanations. Resource constraints precluded similar analysis for the 70B model due to the computational overhead associated with extended token generation at scale, as detailed in Section 6.2.

These results validate the effectiveness of our domain adaptation approach, demonstrating that targeted fine-tuning on cybersecurity-specific datasets significantly enhances model performance in specialized domains while maintaining computational efficiency compared to training larger, general-purpose models from scratch.

6. Discussion

The proliferation of LLMs in NLP has established their capability for generating contextually coherent responses across diverse domains. However, the computational and data requirements for training domain-specific models from scratch remain prohibitive. We demonstrated that DAP offers a viable pathway for specializing general-purpose models to cybersecurity applications in a data-efficient manner, providing foundational capabilities for subsequent task-specific adaptations through SFT or Reinforcement Learning (RL).

Our investigation employed continuous pretraining rather than adapter-based methods, prioritizing the comprehensive integration of domain knowledge despite the increased computational demands. This approach enables models to extract and comprehend domain-specific semantics from natural language—a prerequisite for complex downstream tasks,

Table 5: Performance comparison of cybersecurity domain adaptation approaches across CTI-MCQ, CyberMetric, and SecEval. Our DAP methodology achieves state-of-the-art results while utilizing substantially smaller training datasets compared to existing specialized frameworks. Models are categorized by foundational training approaches, with "Base Alignment" indicating pre-existing instruction-following capabilities and "Extra Tuning" denoting domain-specific adaptations.

Ref.	Model	Size [B]	CTI-MCQ	CyberMetric	SecEval	Base Alignment	Extra Tuning
[41]	Mixtral-8×7B-Instruct	45	–	0.911	–	SFT + DPO	None
	Falcon-180B-Chat	180	–	0.871	–	SFT (Chat-style)	None
	Mistral-7B-Instruct-v0.2	7	–	0.764	–	SFT + DPO	None
	Gemma-1.1-7B-it	7	–	0.758	–	SFT + (novel) RLHF	None
[47]	Llama-3-70B-Instruct	70	0.657	–	–	SFT + (multi-round) DPO	None
[47, 41]	Llama-3-8B-Instruct	8	0.613	0.731	–	SFT + RLHF	None
[49]	Mistral-7B-v0.1	7	–	–	0.437	None (pretrain only)	None
	Qwen-7B	7	–	–	0.314	None (pretrain only)	None
[5]	Llama-Primus-Base (Cybersecurity specialized)	8	0.667	0.866	0.500	SFT + RLHF	Continuous Pretraining
[6]	Foundation-Sec-8B (Cybersecurity specialized)	8	0.676	0.851	–	SFT + RLHF	Continuous Pretraining
Baseline	Llama-3-8B-Base	8	0.499	0.485	0.620	None (pretrain only)	PEFT + Continuous Pretraining
This work	Llama-3.1-8B-DAP	8	0.642	0.858	0.748	None (pretrain only)	DAP
	DSR1D-Qwen-14B-DAP	14	0.685	0.904	0.789	KD + SFT	DAP
	Llama-3.3-70B-Ins-DAP	70	**0.718**	**0.933**	**0.864**	SFT + RLHF	DAP

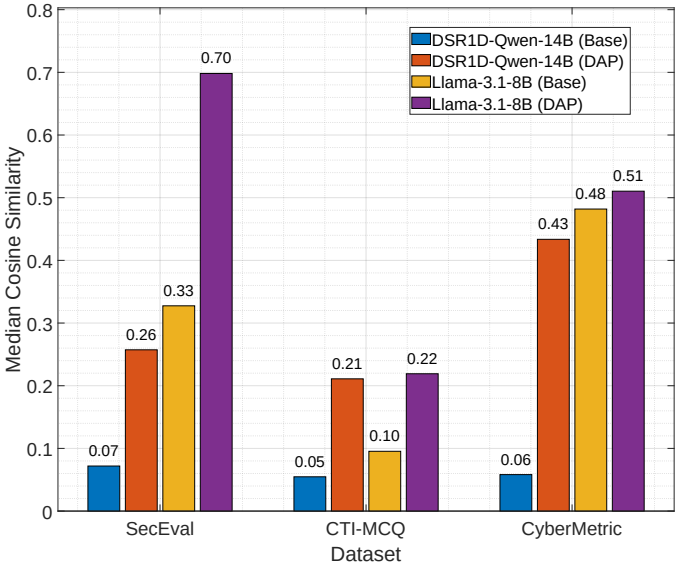

Figure 9: Semantic similarity analysis comparing model explanations to correct answers across cybersecurity benchmarks. Domain-adapted models consistently demonstrate enhanced contextual understanding.

including security report analysis and document summarization. The custom dataset, comprising 126 million words from technical documents, standards, and research papers, was deliberately constrained to preserve general language capabilities while instilling cybersecurity-specific nuances.

Key Findings. The empirical evaluation reveals nuanced patterns in model adaptation efficacy across different architectures and dataset scales. Performance improvements on domain-specific benchmarks validate the effectiveness of DAP, with the magnitude of gains indicating both successful specialization and opportunities for further optimization through enhanced data diversity and quality.

- **Scale-Dependent Learning Dynamics.** The differential performance across model sizes illuminates critical relationships between architectural scale and training data requirements. The 8-billion-parameter model exhibited performance regression on SecEval (Figure 8), which we attribute to the insufficient training data relative to the model's capacity. This model received only 1 million tokens compared to 50-100 times more for larger variants, suggesting that effective domain adaptation requires careful calibration of dataset size to parameter count. This observation underscores the importance of proportional data allocation strategies when adapting models of varying scales.

- **Unsupervised Fine-Tuning Constraints.** The moderate performance improvements likely reflect inherent limitations in unsupervised continuous pretraining. Weight updates without explicit supervision may compromise instruction-following capabilities acquired through prior SFT and RL stages, particularly for complex multi-answer reasoning tasks, such as those in SecEval. Subsequent instruction tuning on curated datasets emphasizing reasoning and task-specific competencies could address these limitations and unlock additional performance gains.

- **Hyperparameter Trade-offs.** Our conservative hyperparameter configuration, particularly the low learning rate (1×10^{-6}) and limited training epochs (2-3), was specifically designed to mitigate catastrophic forgetting of general language capabilities. While this approach successfully prevented severe performance degradation on general tasks (as evidenced by the maintenance of instruction-following abilities), it may have constrained the magnitude of domain-specific improvements that could be achieved through more aggressive adaptation strategies. Future work validating general capability preservation through comprehensive benchmarking would enable more informed hyperparameter optimization.

Comparison to Existing Frameworks. Table 5 provides a comprehensive comparative evaluation of our domain adaptation methodology against existing cybersecurity-focused LLMs and open-source baseline architectures. All models are evaluated on identical benchmark datasets to ensure fair comparison across different architectural paradigms and training approaches. The comparative analysis reveals several insights into the effectiveness and efficiency of our DAP framework.

State-of-the-Art Performance. Our domain-adapted models demonstrate superior performance across all evaluated

benchmarks. The *Llama-3.3-70B-Ins-DAP* model achieves state-of-the-art results with accuracies of 0.718 on CTI-MCQ, 0.933 on CyberMetric, and 0.864 on SecEval, surpassing all baseline configurations. These results exceed the performance of general-purpose models, including the 45-billion parameter *Mixtral-8x7B-Instruct* and the substantially larger 180-billion parameter *Falcon-180B-Chat* [41]. This demonstrates that targeted domain adaptation achieves superior effectiveness with more efficient parameter utilization.

Data Efficiency Advantages. The comparison with specialized cybersecurity models reveals particularly notable findings regarding training data requirements. Our approach significantly outperforms *Llama-Primus-Base* [5] across all benchmarks, despite utilizing substantially smaller training datasets: 118.8 million tokens compared to their 2.77 billion-token corpus (a 23-fold reduction). The performance differential is most pronounced on SecEval, where our 70B model achieves 0.864 compared to *Llama-Primus-Base*'s 0.500, representing a 72.8% relative improvement.

Similar efficiency patterns emerge when comparing against *Foundation-Sec-8B* [6], which employed approximately 5 billion tokens for training. Despite this substantial data advantage, our *Llama-3.1-8B-DAP* achieves superior accuracy on CyberMetric (0.858 versus 0.848) while utilizing only 1 million tokens. While *Foundation-Sec-8B* maintains a slight edge on CTI-MCQ (0.662 versus 0.642), our intermediate *DSR1D-Qwen-14B-DAP* surpasses it (0.685) using just 50 million tokens. This represents a 100-fold reduction in training data compared to *Foundation-Sec-8B*'s 5 billion tokens, demonstrating that strategic dataset curation and model selection can compensate for substantially smaller corpus sizes.

Domain Adaptation Value. Comparison with pretrain-only baselines from [49] shows the clear value of domain-specific adaptation beyond general pretraining. Our intermediate *DSR1D-Qwen-14B-DAP* not only significantly outperforms established pre-training-only architectures on SecEval (achieving 0.789 compared to 0.437 for *Mistral-7B-v0.1* and 0.314 for *Qwen-7B*), but also delivers improvements over its own base model (0.747). Similarly, *DSR1D-Qwen-14B-DAP* shows notable improvement over its base model (0.789 versus 0.747). This performance difference indicates that while general language understanding provides a foundation, targeted domain knowledge is essential for complex cybersecurity reasoning tasks.

Intermediate-Scale Efficiency. The *DSR1D-Qwen-14B-DAP* model exhibits remarkable efficiency, achieving competitive performance across all benchmarks (CTI-MCQ: 0.685, CyberMetric: 0.904, SecEval: 0.789) despite its intermediate parameter size. This effectiveness likely stems from its knowledge distillation foundation, where structured reasoning capabilities inherited from larger teacher models provide enhanced adaptation potential. Compared to instruction-tuned baselines from [47], our 14B model outperforms the *Llama-3-70B-Instruct* on CTI-MCQ while maintaining substantial computational efficiency advantages.

Zero-Shot Generalization. Our models achieve superior performance through zero-shot evaluation, while comparison baselines including *Llama-Primus-Base* and *Foundation-Sec-8B* employ 5-shot prompting strategies. The 5-shot approach provides task format examples that enable the model to infer appropriate response structures augmented through pattern recognition. In contrast, our domain-adapted models achieve competitive or superior results without such assistance. Despite this methodological difference (zero-shot versus 5-shot evaluation), our DAP models demonstrate higher performance across all three benchmarks. This achievement is particularly notable given the substantial disparity in training data volumes, suggesting that careful dataset curation and model selection effectively compensate for limited data availability. Figure 10 visualizes these performance comparisons and highlights the performance of our approach.

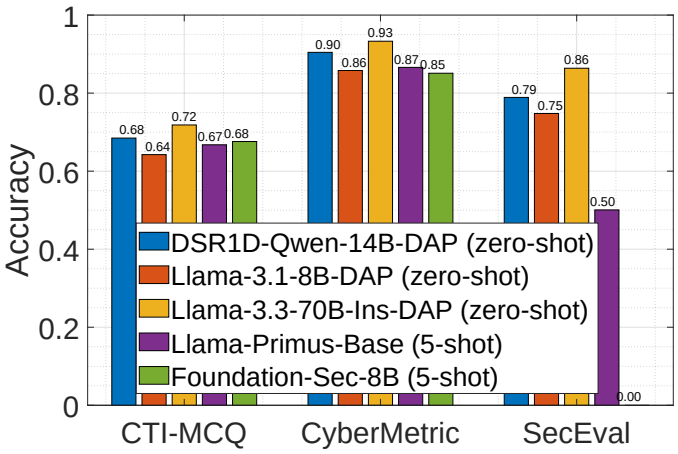

Figure 10: Comparative performance of domain-adapted models against *Llama-Primus-Base* across three cybersecurity benchmarks. Our zero-shot domain-adapted models consistently outperform the specialized 5-shot *Llama-Primus-Base* and *Foundation-Sec-8B*, with *Llama-3.3-70B-Ins-DAP* achieving high performance across all evaluated datasets.

Comparison to Parameter-Efficient Methods. We validated our approach against a PEFT baseline using LoRA [40], as presented in Table 5. While adapter-based methods offer computational efficiency, the LoRA baseline yielded limited accuracies of 0.499, 0.485, and 0.620 across CTI-MCQ, CyberMetric, and SecEval, respectively. In contrast, our DAP approach achieved significantly superior performance (up to 0.642, 0.858, and 0.748). These results confirm that operating on top of frozen base representations is insufficient for deep domain specialization, whereas DAP successfully instills foundational cybersecurity awareness. Future research integrating adapter-based methods atop our domain-adapted models could leverage both this comprehensive domain understanding and efficient task-specific adaptation.

6.1. Implications

The successful demonstration of efficient domain adaptation carries significant implications for both industrial applications and academic research trajectories.

Industrial Use Cases. The primary application domain centers on enhancing digital system security through generative AI capabilities. When incorporating accepted cybersecurity

standards and regulations, adapted models enable automated compliance analysis, identifying policy gaps and conformity issues within corporate frameworks. These AI capabilities extend to vulnerability detection and mitigation, offering both syntactic and logical bug fixes to preempt security compromises.

Natural language interfaces enable cybersecurity professionals to perform sophisticated analyses of system logs and configuration files through conversational queries, allowing for seamless interaction. The models' contextual understanding facilitates identification of suspicious activities and vulnerabilities, enabling proactive threat mitigation. Security experts can leverage these capabilities to address potential vulnerabilities before they are exploited, thereby supporting comprehensive security postures against various threats, including identity theft and system compromise.

Academic Research. This paper establishes foundational insights into efficient LLM domain adaptation using constrained datasets. The adapted models serve as springboards for future investigations, including the refinement of specialized tasks through parameter-efficient methods, such as adapter-based approaches (e.g., PEFT and LoRA). Notably, the 14-billion-parameter model achieved performance approaching that of the 70-billion variant, suggesting potential efficiency gains through architectural optimization. This observation warrants systematic investigation across broader evaluation frameworks to validate the generalizability of these findings. The moderate performance differentials may reflect limitations in dataset scope rather than fundamental constraints, indicating opportunities for targeted dataset enhancement strategies.

The demonstrated viability of small-scale domain adaptation challenges prevailing assumptions about data requirements for effective LLM specialization, opening new research avenues for resource-constrained model development and deployment.

6.2. Challenges, Limitations, and Future Directions

Computational Resource Constraints. The primary challenges centered on hardware resource availability for training large-scale language models. The 70 billion-parameter model proved especially demanding, requiring scaling from 8 NVIDIA H100 instances by factors of 2 to 3 to achieve sufficient computational capacity, even in half-precision mode. Such scaling dramatically increased computational costs, while still necessitating a reduction in the context window to manage memory constraints.

In contrast, models with 8 and 14 billion parameters proved more tractable, though context windows remained constrained to 1024-2048 tokens to avoid memory overflow on lower-tier instances. These reduced context windows inherently limit the models' ability to process extended prompts and extract richer contextual information, which may impact their performance on tasks that require a broader understanding of context. While public cloud platforms offer access to higher-tier GPU resources that could alleviate these constraints, the associated costs substantially exceeded available research budgets.

Cost-Benefit Analysis. The instances operate on an hourly billing model where total cost scales linearly with training duration and instance specifications. This cost structure necessitates careful consideration of trade-offs between computational efficiency and resource allocation strategies. Our cost analysis reveals significant scaling dependencies across model architectures. The DSR1D-Qwen-14B model, deployed on dual ml.p5.48xlarge instances (approximately $63.296 per hour [46]), required 420 minutes (7 hours) of training time, resulting in domain adaptation costs of $886.144 (excluding taxes). As illustrated in Figure 11, training duration exhibits exponential growth characteristics with respect to model parameters and dataset size, directly impacting computational expenditure. Table 6 provides a comprehensive breakdown of training costs across all evaluated architectures.

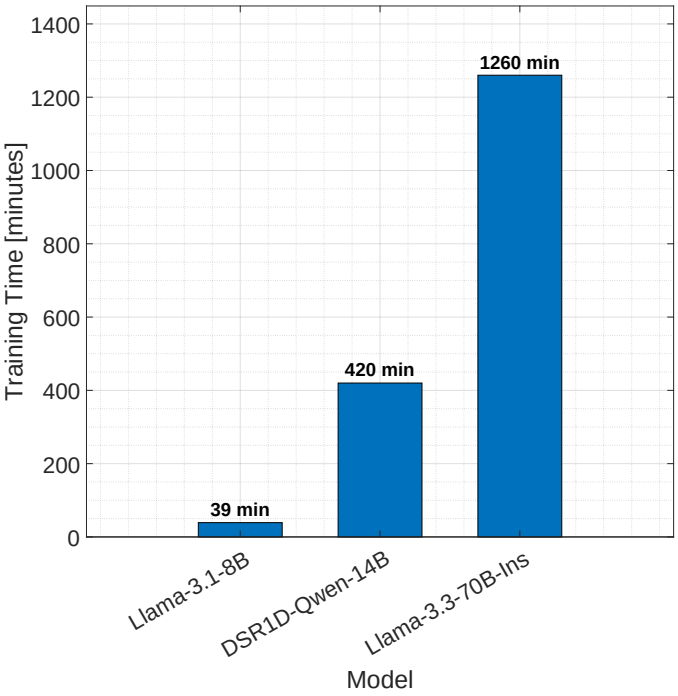

Figure 11: Training time comparison across model architectures demonstrating exponential scaling of computational requirements with model size.

The resource scaling relationship becomes particularly pronounced when comparing intermediate and large-scale architectures. While the 14B model demonstrates manageable computational requirements, the 70B architecture necessitates substantially enhanced infrastructure allocation, including extended training durations and higher-tier instance configurations. This scaling challenge led to total project costs of approximately $23,000 (excluding taxes) when accounting for all training experiments and preliminary trials. The final configurations reported in this work, however, cost $3,581.44 (excluding taxes).

Table 6: Total training cost estimation across different model architectures.

Model	Running Time (hours)	Estimated Total Cost (USD)
Llama-3.1-8B	0.65	$36.87
DSR1D-Qwen-14B	7.00	$886.14
Llama-3.3-70B-Ins	21.00	$2,658.43

Limitations. Several methodological constraints warrant

discussion to contextualize our findings appropriately.

Comparison with Parameter-Efficient Baseline. To provide a comprehensive comparative assessment, we evaluated adapter-based baselines using LoRA [40] and PEFT [33] methods. This baseline evaluation yielded accuracies of 0.499 on CTI-MCQ, 0.485 on CyberMetric-2000, and 0.620 on SecEval (with 790 exact matches). These results fall significantly below the performance achieved by our DAP approach, empirically validating our hypothesis in Section 3.1 that DAP is essential for instilling intrinsic domain understanding. However, as this evaluation was conducted on an Llama-3.1-8B-Base model, it remains an open question whether larger models could compensate for the limitations of adapter-based methods. Consequently, future work must investigate the impact of different model sizes on PEFT performance to determine whether parameter efficiency scales comparably to full-domain adaptive pretraining in cybersecurity contexts.

Non-Uniform Dataset Allocation. A further limitation arises from the non-uniform dataset allocation across model scales. The 8B model received 1 million tokens, the 14B model received 50 million tokens, and the 70B model received 118.8 million tokens. This configuration stemmed from both methodological intent and resource feasibility, aiming to examine how varying corpus sizes influence domain adaptation efficiency across different parameter scales. The smaller dataset for the 8B model was deliberately employed to represent a minimal-data scenario for assessing adaptation efficiency boundaries, while larger variants explored scale-sensitive performance characteristics with proportionally increased data availability.

Although this design provided valuable insights into data-performance dynamics, it also introduced variability that constrains direct cross-architecture comparisons. The reduced gains observed for the 8B model on SecEval (Figure 8) suggest that the corpus may have been insufficient relative to its parameter capacity, indicating that model scale and dataset size are interdependent factors in achieving effective adaptation. Accordingly, these findings should be interpreted within the context of exploratory efficiency analysis rather than controlled equivalence testing. Future investigations should employ proportionally scaled datasets to enable balanced evaluation across architectures and isolate the independent effects of parameter count on adaptation efficacy.

Architectural Heterogeneity. An additional methodological limitation relates to our analysis of scale-dependent learning dynamics. While our investigation evaluated architectures at 8B, 14B, and 70B parameter scales, these models are not from a homologous family. Our evaluation encompasses Llama-3.1-8B (base-pretrained), DSR1D-Qwen-14B (knowledge-distilled), and Llama-3.3-70B-Ins (instruction-tuned), each representing distinct architectural lineages with different pretraining corpora, optimization strategies, and foundational alignments. This architectural heterogeneity introduces significant confounding variables. Consequently, observed performance differences cannot be attributed exclusively to the parameter scale, as they reflect the combined influence of model capacity, pretraining methodology, and base alignment approaches. A rigorous assessment of pure scaling effects would require controlled comparison across a single, consistent model family (e.g., Llama-3.1-8B, Llama-3.1-70B) with identical pretraining and alignment procedures.

Catastrophic Forgetting Assessment. A primary limitation of this investigation is the absence of empirical validation for catastrophic forgetting. While our conservative training configuration (specifically the low learning rate of approximately 1×10^{-6} and limited training epochs of 2 to 3) was intentionally designed to mitigate degradation of general linguistic capabilities, its success was not quantitatively measured. As noted in Section 3.5, resource constraints precluded comprehensive benchmarking of our DAP variants against their baselines on established general-domain frameworks such as SuperGLUE [50].

Future Research Directions. Several promising avenues exist to build upon and extend our foundational results.

Catastrophic Forgetting Validation. A priority is addressing the primary methodological limitation of this study through rigorous quantitative assessment of catastrophic forgetting. Future work must benchmark our DAP variants against their baselines on established general-domain benchmarks (e.g., MMLU [51], SuperGLUE [50], HellaSwag [52]). This validation is essential to empirically confirm preservation of foundational language competencies and establish a complete impact profile for our domain adaptation methodology. Such an evaluation would enable more informed optimization of training hyperparameters, potentially allowing for more aggressive adaptation strategies if general capabilities prove to be robustly preserved.

Controlled Scaling Analysis. To properly isolate the impact of parameter scale from architectural artifacts, subsequent investigation should be conducted across a homologous model family (e.g., Llama-3.1-8B, Llama-3.1-70B, Llama-3.3-405B) with consistent pretraining and alignment procedures. This controlled comparison would enable definitive conclusions regarding optimal model scale for cybersecurity domain adaptation and inform resource allocation strategies for organizations with varying computational budgets.

Expanded Model Architectures. Exploring model selection beyond the evaluated Llama-3.1-8B, DeepSeek-R1-Distill-Qwen-14B, and Llama-3.3-70B-Instruct architectures could yield improved outcomes. Particular interest lies in incorporating models with inherent reasoning capabilities such as Chain-of-Thought (CoT) architectures, which may demonstrate enhanced performance on complex multi-step cybersecurity reasoning tasks. Alternative parameter scales between our evaluated points (e.g., 32B, 40B) may reveal optimal efficiency frontiers balancing performance and computational requirements.

Dataset Enhancement. Dataset expansion represents a promising avenue for performance improvement. The varying dataset sizes employed in our investigation (potentially insufficient for the 8B model) suggest that broader data collection encompassing wider cybersecurity topics could provide richer contextual and semantic information. Systematic exploration of dataset composition effects (e.g., standards-heavy versus literature-heavy corpora) could inform optimal curation strategies. Additionally, incorporating more recent cybersecurity content and emerging threat landscapes would enhance model

relevance for evolving security challenges.

Advanced Training Methodologies. Training methodology refinements, such as knowledge distillation, could optimize the domain adaptation process. Teacher-student architectures that integrate original model knowledge into loss optimization could potentially mitigate concerns about overfitting and catastrophic forgetting. Other promising techniques include curriculum learning (progressively introducing complexity during adaptation) and mixture-of-experts approaches (specialized sub-networks for different cybersecurity subdomains). Resource constraints prevented the exploration of these techniques in the current study; however, they represent valuable directions for enhancing adaptation efficiency and effectiveness.

Real-World Deployment. While our benchmark-based evaluation demonstrates foundational cybersecurity knowledge acquisition, validation in operational environments represents a necessary next step. Future work should assess model performance on real-world security workflows, including the analysis of intrusion detection logs, vulnerability assessment reporting, and Security Operations Center (SOC) alert triage. Such an evaluation would establish practical applicability beyond controlled benchmark environments and identify task-specific fine-tuning requirements for production deployment.

7. Conclusion

We investigated the potential of pretrained LLMs in cybersecurity through domain-adaptive training, demonstrating that specialized domain knowledge can be effectively instilled through resource-efficient methodologies. Our approach achieves state-of-the-art performance across established cybersecurity benchmarks, utilizing 118.8 million tokens compared to 2.77-5 billion tokens employed by existing specialized frameworks, representing a 23- to 42-fold reduction in training data requirements. This efficiency gain challenges prevailing assumptions about data volume requirements for effective domain specialization, demonstrating that curation of high-quality materials can compensate for a limited corpus.

While adapter-based PEFT methods provided a baseline with limited performance, we used DAP to develop foundational domain understanding, essential for contextualizing prompts and generating coherent, logical responses. The experimental results validate our approach across three architectures (Llama-3.1-8B, DeepSeek-R1-Distill-Qwen-14B, and Llama-3.3-70B-Instruct), with the largest model achieving accuracies of 0.718, 0.933, and 0.864 on CTI-MCQ, CyberMetric, and SecEval benchmarks, respectively. These results surpass both the parameter-efficient baselines and specialized models, including Llama-Primus-Base and Foundation-Sec-8B, despite substantially reduced training overhead.

The domain-adapted models developed through this work provide robust foundations for a wide range of downstream applications, including vulnerability and threat analysis, security document summarization, and compliance assessment. Such specialized models offer practical pathways to enhance organizational security postures while maintaining computational feasibility in resource-constrained environments. Future refinement through SFT and RL techniques could further enhance reasoning capabilities, enabling advanced decision-making through improved semantic understanding and contextual analysis. Our findings establish that efficient, targeted domain adaptation represents a viable alternative to data-intensive pretraining approaches for cybersecurity LLM specialization.

References

[1] M.-F. Wong, S. Guo, C.-N. Hang, et al., Natural Language Generation and Understanding of Big Code for AI-Assisted Programming: A Review, Entropy 25 (6) (2023). doi:10.3390/e25060888.

[2] J. P. Inala, C. Wang, S. M. Drucker, et al., Data Analysis in the Era of Generative AI, ArXiv abs/2409.18475 (2024).

[3] M. Mitra, M. G. De Vos, N. Cortinovis, et al., Generative AI for Research Data Processing: Lessons Learnt From Three Use Cases, in: 2024 IEEE 20th International Conference on e-Science (e-Science), 2024, all Open Access, Green Open Access. doi:10.1109/e-Science62913.2024.10678704.

[4] P. Lewis, E. Perez, A. Piktus, et al., Retrieval-augmented generation for knowledge-intensive NLP tasks, in: Proceedings of the 34th International Conference on Neural Information Processing Systems, NIPS '20, Curran Associates Inc., Red Hook, NY, USA, 2020.

[5] Y.-C. Yu, T.-H. Chiang, C.-W. Tsai, et al., Primus: A Pioneering Collection of Open-Source Datasets for Cybersecurity LLM Training, arXiv preprint arXiv:2502.11191 (2025).

[6] Kassianik et al., Llama-3.1-foundationai-securityllm-base-8b technical report, arXiv preprint arXiv:2504.21039 (2025).

[7] Z. Yun-tao, G. Ling, W. Yong-cheng, An improved TF-IDF approach for text classification, Journal of Zhejiang University-Science A 6 (1) (2005) 49–55.

[8] J. Ramos, et al., Using tf-idf to determine word relevance in document queries, in: Proceedings of the first instructional conference on machine learning, Vol. 242, Citeseer, 2003, pp. 29–48.

[9] P. F. Brown, V. J. Della Pietra, P. V. deSouza, et al., Class-Based *n*-gram Models of Natural Language, Computational Linguistics 18 (4) (1992) 467–480.
URL https://aclanthology.org/J92-4003/

[10] J. L. Elman, Finding structure in time, Cognitive science 14 (2) (1990).

[11] S. Hochreiter, J. Schmidhuber, Long Short-Term Memory, Neural Computation 9 (8) (1997). doi:10.1162/neco.1997.9.8.1735.

[12] A. Graves, A. rahman Mohamed, G. E. Hinton, Speech recognition with deep recurrent neural networks, 2013, pp. 6645–6649.
URL https://api.semanticscholar.org/CorpusID:206741496

[13] I. Sutskever, O. Vinyals, Q. V. Le, Sequence to sequence learning with neural networks, in: Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2, NIPS'14, MIT Press, Cambridge, MA, USA, 2014, p. 3104–3112.

[14] G. Lample, M. Ballesteros, S. Subramanian, et al., Neural Architectures for Named Entity Recognition, in: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, San Diego, California, 2016, pp. 260–270. doi:10.18653/v1/N16-1030.

[15] A. Vaswani, N. Shazeer, N. Parmar, et al., Attention is All you Need, in: Advances in Neural Information Processing Systems, Vol. 30, 2017.
URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[16] Y. Ding, L. L. Zhang, C. Zhang, et al., LongRoPE: extending LLM context window beyond 2 million tokens, in: Proceedings of the 41st International Conference on Machine Learning, ICML'24, JMLR.org, 2024.

[17] Q. Wang, B. Li, T. Xiao, et al., Learning Deep Transformer Models for Machine Translation, in: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Florence, Italy, 2019, pp. 1810–1822. doi:10.18653/v1/P19-1176.

[18] C. Raffel, N. Shazeer, A. Roberts, et al., Exploring the limits of transfer learning with a unified text-to-text transformer, J. Mach. Learn. Res. 21 (1) (Jan. 2020).

[19] M. Lewis, Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, arXiv preprint arXiv:1910.13461 (2019).

[20] J. Devlin, M.-W. Chang, K. Lee, et al., BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Association for Computational Linguistics, Minneapolis, Minnesota, 2019, pp. 4171–4186. doi:10.18653/v1/N19-1423.

[21] Y. Liu, RoBERTa: A robustly optimized bert pretraining approach, arXiv preprint arXiv:1907.11692 364 (2019).

[22] K. L. Tan, C. P. Lee, K. S. M. Anbananthen, et al., RoBERTa-LSTM: A Hybrid Model for Sentiment Analysis With Transformer and Recurrent Neural Network, IEEE Access 10 (2022) 21517–21525. doi:10.1109/ACCESS.2022.3152828.

[23] Z. Lan, M. Chen, S. Goodman, et al., ALBERT: A Lite BERT for Self-Supervised Learning of Language Representations, in: International Conference on Learning Representations (ICLR), 2020.
URL https://openreview.net/forum?id=H1eA7AEtvS

[24] H. Touvron, T. Lavril, G. Izacard, et al., LLaMA: Open and Efficient Foundation Language Models, ArXiv abs/2302.13971 (2023).

[25] J. Achiam, S. Adler, S. Agarwal, et al., Gpt-4 technical report, arXiv preprint arXiv:2303.08774 (2023).

[26] S. Zhu, L. Pan, D. Jian, et al., Overcoming language barriers via machine translation with sparse Mixture-of-Experts fusion of large language models, Inf. Process. Manage. 62 (3) (Apr. 2025). doi:10.1016/j.ipm.2025.104078.

[27] E. Aghaei, X. Niu, W. Shadid, et al., SecureBERT: A Domain-Specific Language Model for Cybersecurity, in: Security and Privacy in Communication Networks, Springer Nature Switzerland, Cham, 2023, pp. 39–56.

[28] E. Aghaei, S. Jain, P. Arun, et al., SecureBERT 2.0: Advanced Language Model for Cybersecurity Intelligence, arXiv preprint arXiv:2510.00240 (2025).

[29] M. Bayer, P. Kuehn, R. Shanehsaz, et al., CySecBERT: A Domain-Adapted Language Model for the Cybersecurity Domain, ACM Trans. Priv. Secur. 27 (2) (Apr. 2024). doi:10.1145/3652594.

[30] P. Ranade, A. Piplai, A. Joshi, et al., CyBERT: Contextualized Embeddings for the Cybersecurity Domain, in: 2021 IEEE International Conference on Big Data (Big Data), 2021, pp. 3334–3342. doi:10.1109/BigData52589.2021.9671824.

[31] J. He, M. Vero, G. Krasnopolska, et al., Instruction tuning for secure code generation, in: Proceedings of the 41st International Conference on Machine Learning, ICML'24, JMLR.org, 2024.

[32] Z. Luo, C. Xu, P. Zhao, et al., WizardCoder: Empowering Code Large Language Models with Evol-Instruct 2024 (2024) 27168–27188.
URL https://proceedings.iclr.cc/paper_files/paper/2024/file/72eba29737f9c3a5a4ce8cdb7b667145-Paper-Conference.pdf

[33] N. Houlsby, A. Giurgiu, S. Jastrzebski, et al., Parameter-Efficient Transfer Learning for NLP, in: Proceedings of the 36th International Conference on Machine Learning, Vol. 97 of Proceedings of Machine Learning Research, PMLR, 2019, pp. 2790–2799.
URL https://proceedings.mlr.press/v97/houlsby19a.html

[34] W. Y. Lee, J. Saxe, R. Harang, SeqDroid: Obfuscated Android Malware Detection Using Stacked Convolutional and Recurrent Neural Networks, Springer International Publishing, Cham, 2019, pp. 197–210. doi:10.1007/978-3-030-13057-2_9.

[35] T. A. Tang, D. McLernon, L. Mhamdi, et al., Intrusion Detection in SDN-Based Networks: Deep Recurrent Neural Network Approach, Springer International Publishing, Cham, 2019, pp. 175–195. doi:10.1007/978-3-030-13057-2_8.

[36] A. Diaf, A. A. Korba, N. E. Karabadji, et al., Beyond detection: Leveraging large language models for cyber attack prediction in iot networks, in: 2024 20th International Conference on Distributed Computing in Smart Systems and the Internet of Things (DCOSS-IoT), IEEE, 2024, pp. 117–123.

[37] H. Huang, Y. Wang, SecBERT: Privacy-preserving pre-training based neural network inference system, Neural Networks 172 (2024) 106135.

[38] A. Shestov, A. Cheshkov, R. Levichev, et al., Finetuning large language models for vulnerability detection, arXiv preprint arXiv:2401.17010 (2024).

[39] R. Fayyazi, R. Taghdimi, S. J. Yang, Advancing TTP Analysis: Harnessing the Power of Large Language Models with Retrieval Augmented Generation, in: 2024 Annual Computer Security Applications Conference Workshops (ACSAC Workshops), 2024, pp. 255–261. doi:10.1109/ACSACW65225.2024.00036.

[40] Yu et al., Low-Rank Adaptation of Large Language Model Rescoring for Parameter-Efficient Speech Recognition, in: IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), 2023. doi:10.1109/ASRU57964.2023.10389632.

[41] N. Tihanyi, M. A. Ferrag, R. Jain, et al., CyberMetric: a benchmark dataset based on retrieval-augmented generation for evaluating LLMs in cybersecurity knowledge, in: 2024 IEEE International Conference on Cyber Security and Resilience (CSR), IEEE, 2024, pp. 296–302.

[42] Amazon Sagemaker, accessed: 05 May 2025.
URL https://aws.amazon.com/sagemaker/

[43] Anthropic Claude 3.7, accessed: 05 May 2025.
URL https://www.anthropic.com/news/claude-3-7-sonnet

[44] Medium – Difference between float16 and bfloat16, accessed: 05 May 2025.
URL https://medium.com/@furkangozukara/what-is-the-dif...d75ac7ec30fa

[45] I. Loshchilov, F. Hutter, Decoupled Weight Decay Regularization, in: International Conference on Learning Representations, 2017.

[46] Amazon Web Services, SageMaker AI Pricing, https://aws.amazon.com/sagemaker-ai/pricing/, accessed: 2025-05-26 (n.d.).

[47] M. T. Alam, L. Nguyen, D. Bhusal, et al., CTIBench: a benchmark for evaluating LLMs in cyber threat intelligence, in: Proceedings of the 38th International Conference on Neural Information Processing Systems, NIPS '24, Curran Associates Inc., Red Hook, NY, USA, 2025.

[48] SecEval Benchmark - GitHub Repository, accessed: 05 May 2025.
URL https://xuanwuai.github.io/SecEval/

[49] G. Li, Y. Li, G. Wang, et al., SecEval Leaderboard: A Comprehensive Benchmark for Evaluating Cybersecurity Knowledge of Foundation Models, https://xuanwuai.github.io/SecEval/leaderboard.html (2024).

[50] A. Wang, Y. Pruksachatkun, N. Nangia, et al., SuperGLUE: a stickier benchmark for general-purpose language understanding systems, Curran Associates Inc., Red Hook, NY, USA, 2019.

[51] G. Son, H. Lee, S. Kim, et al., KMMLU: Measuring Massive Multitask Language Understanding in Korean, in: Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers), Association for Computational Linguistics, Albuquerque, New Mexico, 2025. doi:10.18653/v1/2025.naacl-long.206.

[52] R. Zellers, A. Holtzman, Y. Bisk, et al., HellaSwag: Can a Machine Really Finish Your Sentence?, in: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Florence, Italy, 2019. doi:10.18653/v1/P19-1472.