NanoFlux: Adversarial Dual-LLM Evaluation and Distillation for Multi-Domain Reasoning

Raviteja Anantha, Soheil Hor, Teodor Nicola Antoniu, Layne C. Price
Amazon
Seattle, WA, USA
{ravantha, soheilh, teonian, prilayne}@amazon.com

Abstract

We present NanoFlux, a novel adversarial framework for generating targeted training data to improve LLM reasoning, where adversarially-generated datasets of ≤ 200 examples outperform conventional fine-tuning approaches. The framework employs a competitive dynamic between models alternating as Attacker and Defender, supervised by a tool-augmented Judge, synthesizing multi-step questions with explanatory annotations. Fine-tuning a 4B-parameter model on NanoFlux-generated data yields performance gains across diverse domains compared to full-benchmark fine-tuning: +5.9% on mathematical reasoning, +3.6% on scientific reasoning, and +16.6% on medical reasoning, while reducing computational requirements by 3-14×. Ablation studies reveal a non-monotonic relationship between dataset characteristics and model performance, uncovering domain-specific optimal points for question complexity and reasoning quality. NanoFlux automates training data generation through embedding-based novelty filtering, tool-augmented evaluation, and multi-hop reasoning, pointing to the value of small, targeted training datasets.

1 Introduction

As large language models (LLMs) rapidly approach and surpass human-level performance on established benchmarks, we confront a fundamental limitation: the finite nature of high-quality training and evaluation data. While generating synthetic training examples represents one potential path forward, creating effective synthetic data remains challenging, as naive generation approaches often produce low-information samples that fail to improve model performance, while synthesizing effective datasets typically requires precisely the kind of human expertise and curation that we seek to automate. Recent work, notably LIMO (Ye et al., 2025), has demonstrated that small, carefully curated datasets of high-quality chain-of-thought solutions can unlock strong reasoning performance, but still depends on human effort in curation.

We introduce **NanoFlux**, a fully *generative adversarial* framework that reimagines data-efficient reasoning improvement. NanoFlux orchestrates a competitive dynamic between two models alternating as *Attacker* and *Defender*, supervised by a tool-augmented *Judge* that evaluates responses for accuracy, coherence, and safety (Figure 1). This adversarial architecture automatically identifies and targets specific reasoning weaknesses, generating training examples that precisely target key reasoning gaps, enabling efficient learning from compact datasets. NanoFlux advances beyond prior filtering-driven approaches through three key innovations:

1. **Targeted adversarial generation:** Rather than filtering existing data, NanoFlux synthesizes questions where one model fails and another succeeds, creating high-information training signals with few examples.

2. **Automated quality assurance:** The tool-augmented *Judge* model evaluates the joint question-answer-reasoning triples for quality, accuracy, and safety using web search and code execution, eliminating manual curation.

3. **Flexible domain adaptability:** The framework's architecture generalizes across diverse reasoning domains with minimal domain-specific configuration, as demonstrated by results on mathematical (GSMHard), medical (MultiMedQA), and scientific reasoning (GenomeBench).

Empirical results demonstrate that fine-tuning on just 200 NanoFlux-generated examples yields substantial accuracy improvements (+5.9% on GSMHard, +3.6% on GenomeBench, and +16.6% on MultiMedQA) while reducing computational

Proceedings of the Fifth Workshop on Generation, Evaluation and Metrics (GEM), pages 253–270

requirements by 3-14× compared to full-dataset fine-tuning. Moreover, ablation studies reveal counterintuitive non-monotonic relationships between dataset characteristics and model performance, suggesting a fundamental tension between competing objectives in training data optimization.

To the best of our knowledge, NanoFlux is the first adversarial framework to demonstrate that extremely small, automatically generated datasets can outperform conventional fine-tuning approaches across diverse reasoning domains. By shifting focus from scaling data quantity toward optimizing data quality, our work offers a promising path toward more efficient, accessible, and capable AI systems.

2 Related Work

Recent work shows that large-scale reasoning can be unlocked with surprisingly little fine-tuning data if the data is carefully selected. Ye et al. (2025) introduced **LIMO**, demonstrating that fine-tuning Qwen2.5-32B on only ∼800 curated math examples yielded state-of-the-art performance on AIME-24 and MATH500, outperforming models trained on orders of magnitude more data. Similarly, Li et al. (2025) showed that models benefit from the *structure* of chain-of-thought (CoT) demonstrations, even when final answers are wrong.

Several approaches have explored letting models create their own training data. Sun et al. (2025) proposed **Crescent**, where an LLM generates and solves its own questions, bootstrapping improved math reasoning without external supervision. Huang et al. (2026) introduced **R-Zero**, a co-evolutionary self-play loop in which a *Challenger* model generates questions that a *Solver* model cannot answer, forming an adversarial curriculum that yielded +6–8 point gains on reasoning benchmarks with no human data. PENG et al. (2025) proposed **ReGenesis**, which structured self-synthesized reasoning data around abstract, task-agnostic templates to improve generalization: unlike naive self-training, ReGenesis achieved +6.1% gains on OOD reasoning tasks. These works suggest that focusing training on failure cases is more efficient than scaling data indiscriminately. Self-Questioning Language Models (SQML) (Chen et al., 2025) explores an *asymmetric self-play* framework: a proposer generates questions from a domain prompt and a solver attempts to answer them. Both roles are trained by reinforcement learning, using majority voting (or unit tests for coding) as a proxy for correctness in lieu of ground truth. SQLM achieves reasoning gains on arithmetic, algebra (OMEGA benchmark), and programming tasks without access to any curated training data.

Recent studies have also targeted adversarial data generation in domain-specific QA, such as math (Xie et al., 2024) and medical QA (Ness et al., 2024). Sung et al. (2025) have also developed a framework to evaluate adversarial question quality with **Item Response Theory (IRT)**, showing that "good" adversarial questions stump models but not humans, with high discriminative power. The **VAULT** framework (Kazoom et al., 2025) automated adversarial data augmentation for natural language inference. By prompting an LLM to generate candidate hard examples and filtering for those misclassified by the current model, VAULT iteratively improved model robustness. After several rounds, a RoBERTa model improved from 54.7% to 72.0% on MultiNLI: large gains achieved with far fewer examples than traditional augmentation.

Our approach is closest in spirit to LIMO and R-Zero but differs in critical ways. Unlike LIMO, which relies on human-curated CoT demonstrations, we automatically generate new *benchmark-derived adversarial questions* through dual-LLM interaction, without human filtering or CoT supervision. Unlike R-Zero's self-play, our method grounds generation in existing benchmark datasets, ensuring domain relevance while still producing harder variants. Compared to VAULT, which targeted classification tasks, our focus is on single-answer QA benchmarks across domains (math, medicine), with exact-match evaluation rather than human judgment. Collectively, prior works show that small, high-quality or adversarially targeted datasets can dramatically improve reasoning. Our contribution is a domain-agnostic adversarial distillation framework that (i) identifies and trains on precisely those questions a model fails, (ii) achieves superior performance compared to full-dataset fine-tuning, and (iii) demonstrates potential for cross-domain transfer.

3 NanoFlux Framework

Attacker & Defender. Figure 1 presents the NanoFlux framework architecture. The framework operates by randomly sampling n seed questions from existing benchmark datasets: GSMHard (Gao et al., 2023) for mathematical reasoning,

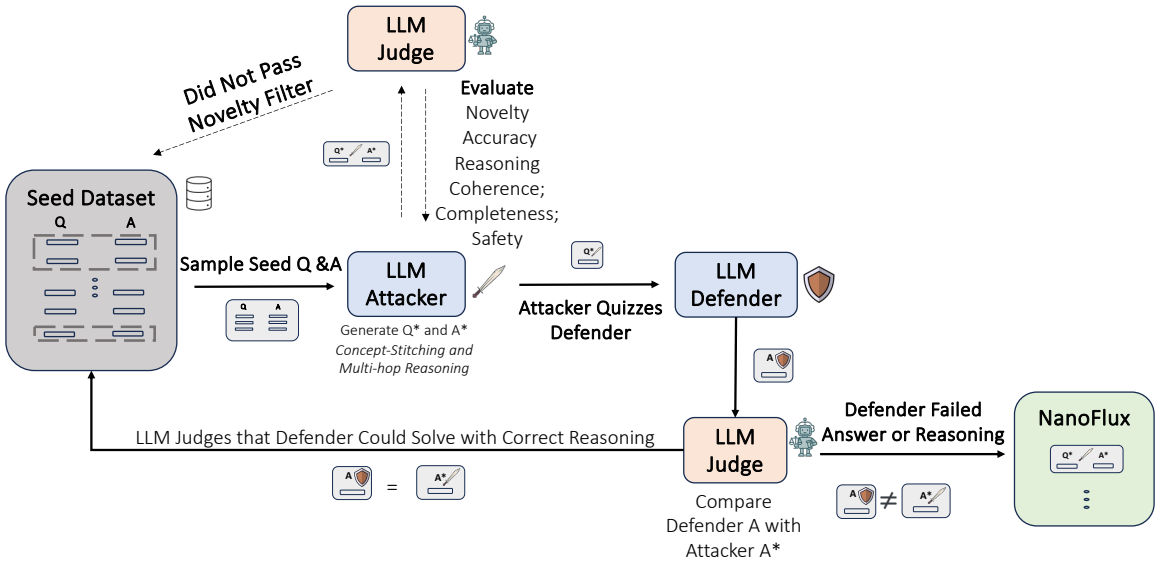

Figure 1: **The NanoFlux Adversarial Data Generation Framework.** The process begins with random sampling from benchmark datasets (left), followed by the attacker model generating questions through concept stitching. Generated questions undergo embedding-based novelty filtering to ensure diversity, then validation by the judge model equipped with code execution and web search capabilities. The defender model attempts to solve validated questions, with the judge evaluating response correctness. Questions that the defender fails to solve (or solves through novel approaches) are retained in the final NanoFlux dataset, which contains 200 examples per domain.

GenomeBench (Yin et al., 2025) for genomics, and MultiMedQA (Singhal et al., 2023) for medical reasoning. The number of seed questions n varies by domain: 5-7 for GSMHard and GenomeBench, and 7-12 for MultiMedQA, selected uniformly at random within these ranges. For the attacker and defender roles, we use domain-specific model configurations. For GSMHard and GenomeBench domains, we employ Gemma-3-4B (Team et al., 2025), while for MultiMedQA we use MedGemma-4B (Sellergren et al., 2025), a domain-specialized model with biomedical pretraining. These models alternate with Claude-3.7-Sonnet v2 (Anthropic, 2025) in the attacker and defender roles.

The *Attacker* model generates new questions by combining concepts from the seed questions. We define "concept stitching" as the process of merging elements from multiple seed questions into a single, more complex question. This is implemented through a structured prompt template (Appendix Table 4) that instructs the model to: (1) identify key concepts from each seed question, (2) find conceptual connections between them, (3) create a new question that requires understanding these connections, and (4) provide a step-by-step solution with the correct answer.

The *Defender* model attempts to solve the attacker's questions. The defender receives only the question text, without access to the attacker's solution. The defender's responses must follow a structured format (Appendix Table 5) with three components: (1) Analysis: initial problem decomposition (<500 tokens); (2) Solution: step-by-step reasoning with explicit calculations (<1500 tokens); and (3) Answer: final numerical or categorical response (<100 tokens).

To prevent model-specific biases, we alternate the roles of attacker and defender between the two models after each turn. A "turn" consists of: (1) the attacker generating a question, (2) the question passing validation, (3) the defender attempting to solve it, and (4) the judge evaluating the defender's solution. This alternation ensures that any systematic weaknesses in one model don't bias the dataset generation process.

The Judge Model. We use OpenAI's O3 reasoning model (OpenAI, 2025b) as the judge model with two distinct operational modes:

Question validation mode verifies that attacker-generated questions are well-formed and solvable (Appendix Table 7). The judge checks for: (1) question clarity, (2) sufficient information to solve the problem, (3) mathematical correctness of the attacker's solution, and (4) consistency between the solution steps and the final answer. For math-

ematical questions, the judge generates Python code to verify calculations, executing it in a sandboxed environment with NumPy (Harris et al., 2020) and SymPy (Meurer et al., 2017) libraries. For knowledge-intensive domains, the judge uses Google Search API to verify factual claims, with the top 5 search results provided as context. Questions failing validation are rejected, and the attacker is prompted to generate a new question.

Answer evaluation mode assesses the defender's solutions against the validated attacker solutions. The judge evaluates responses using two criteria sets: (1) *Strict Mode*: requires both correct numerical answer (within tolerance $\epsilon = 10^{-6}$ for mathematical questions) and valid reasoning steps.; and (2) *Soft Mode*: prioritizes reasoning quality over exact answers, accepting solutions with minor calculation errors if the reasoning approach is sound.

Each evaluation includes a confidence score in $[0, 1]$. Evaluations with confidence below $\theta_c = 0.9$ trigger re-evaluation with a modified prompt that explicitly requests higher confidence. After three low-confidence evaluations, the question is discarded (Appendix Table 6).

Embedding-Based Novelty Filtering. NanoFlux implements a two-stage embedding-based novelty filtering system that extends beyond correctness filtering. In Stage 1 (Question Diversity Filtering), each generated question from the *Attacker* is embedded using OpenAI's text-embedding-3-small model (OpenAI, 2024), creating a 1536-dimensional vector representation. The cosine similarity between this vector and all previously generated questions is computed, with questions exceeding a threshold $\theta_q = 0.85$ being rejected as insufficiently novel. This approach contrasts with prior work that relied on lexical ROUGE-L scores for filtering (Wang et al., 2023) or simple formatting and duplicate checks (Honovich et al., 2023), providing a more semantically meaningful assessment of novelty.

Stage 2 (Solution Novelty Filtering) addresses cases where the *Defender* correctly answers a question but through a substantially different reasoning process than the *Attacker*. Building on insights that embedding trajectories can serve as diagnostic signals for reasoning quality (Wang et al., 2024), we compute the embedding similarity between the *Attacker's* and *Defender's* reasoning traces. Solutions with similarity below threshold $\theta_s = 0.75$ are retained despite being correctly answered, as they represent novel solution approaches. This aligns with recent work demonstrating that assessing reasoning traces beyond answer correctness improves model performance (Chen et al., 2024).

Domain-Specific Adaptations. For mathematical reasoning (GSMHard), we enhance the judge's validation capabilities with Python code execution using the following libraries: NumPy 1.24.3, SymPy 1.12, and Math. The judge generates verification code for each mathematical solution, with a standardized structure: (1) variable definition, (2) calculation steps mirroring the solution, and (3) final answer verification. Numerical answers are compared with a tolerance of $\epsilon = 10^{-6}$ to account for floating-point precision issues. The attacker prompt includes specific instructions to generate questions involving multi-step calculations, unit conversions, and geometric reasoning.

For scientific reasoning (GenomeBench), we use the same configuration as for GSMHard. We implement XML-structured answer formats with tags for <*hypothesis*>, <*evidence*>, <*mechanism*>, and <*conclusion*> to facilitate precise evaluation. The judge model is provided with a genomics-specific evaluation rubric that emphasizes scientific accuracy, mechanistic reasoning, appropriate citation of genomic principles, and logical consistency.

For medical domains (MultiMedQA), we use MedGemma-4B alternating with Claude-3.7-Sonnet as attacker and defender models due to its domain-specific pretraining on biomedical literature. We implement a structured response format that includes specific sections: ANALYSIS, SOLUTION, ANSWER, KNOWLEDGE_MAP, REASONING_CHAIN, and COGNITIVE_CHALLENGES. The judge evaluates medical responses using a specialized rubric focusing on: (1) clinical reasoning accuracy, (2) evidence-based justification, (3) consideration of differential diagnoses, and (4) appropriate treatment recommendations. Web search verification is enabled for factual medical claims, with a 30-second timeout per query and a maximum of 5 queries per evaluation.

Training Methodology. NanoFlux generates datasets of 200 samples per domain, a size chosen to be significantly smaller than the train set size of chosen benchmarks while still providing sufficient examples for effective fine-tuning. Our ablation studies (see Paragraph 5) provide insight

into the effect of this dataset size. The dataset generation process continues until 200 valid examples are collected or a maximum of 1000 turns is reached, whichever comes first.

For fine-tuning, we use Low-Rank Adaptation (LoRA) (Hu et al., 2022) with fixed hyperparameters: rank $r = 8$, alpha $\alpha = 32$, and dropout $= 0.05$, following prior work showing their effectiveness for fine-tuning (Yan et al., 2025; Tian et al., 2025). The learning rate follows a linear decay schedule, starting at 2×10^{-4} and decreasing to zero during training, following the findings of Bergsma et al. (2025) that linear decay to zero outperforms constant and step-based schedules for LLM fine-tuning.

Train/Test Separation. NanoFlux generates *new* synthetic questions derived from benchmark training splits via concept stitching; the generated questions are distinct from the original benchmark items. All evaluation is performed on the original held-out test splits of GSMHard, GenomeBench, and MultiMedQA, which are never seen during data generation or fine-tuning.

4 Evaluation Datasets and Tasks

We evaluate NanoFlux's effectiveness by comparing the 4B-parameter SLM Gemma-4B fine-tuned on our synthesized 200-sample datasets against both conventional full-dataset fine-tuning and a frontier LLM to assess generalizability across different reasoning domains.

GSMHard (Gao et al., 2023) is a harder variant of the Grade School Math 8K (GSM8K) benchmark. It was introduced by modifying GSM8K problems through replacing the original numbers with larger values, increasing difficulty while preserving the underlying reasoning structure. The original GSM8K dataset (Cobbe et al., 2021) contains grade-school math word problems requiring multi-step reasoning and has become a standard benchmark for evaluating reasoning capabilities of language models. Recent work such as Crescent (Sun et al., 2025) continues to evaluate reasoning improvements on GSM8K-style tasks.

GenomeBench is a scientific reasoning benchmark derived from over a decade of expert Q&A discussions on CRISPR gene editing (Yin et al., 2025). The dataset consists of 3,332 multiple-choice questions with expert-written rationales, partitioned into 2,671 training and 661 test examples. Unlike exam-style biomedical datasets, GenomeBench captures authentic expert reasoning across experimental troubleshooting, reagent choice, and protocol design. It complements prior biomedical QA resources (e.g., PubMedQA, Lab-Bench) by reflecting real-world scientific discourse.

MultiMedQA (Singhal et al., 2023) is a medical reasoning benchmark combining six existing datasets spanning medical licensing exams, consumer health queries, and biomedical literature, and also introduces HealthSearchQA, a large set of real-world medical search questions. These datasets span multiple medical domains including clinical knowledge (4,183 questions), medical licensing examinations (12,723 questions), biomedical literature (1,000 questions), and consumer health queries (3,375 questions). It has been central to evaluating medical LLMs, including Med-PaLM (Singhal et al., 2023) and follow-up work on data-efficient reasoning like R-Zero (Huang et al., 2026).

5 Results and Analysis

Model Finetuning Performance Comparison. Table 1 presents a performance analysis across computational efficiency and accuracy, revealing three key findings. First, NanoFlux demonstrates computational efficiency, reducing fine-tuning costs by 77% for GSMHard (9.2×10^{16} to 2.1×10^{16} FLOPs), 69% for GenomeBench (8.5×10^{16} to 2.6×10^{16} FLOPs), and 93% for MultiMedQA (3.8×10^{17} to 2.6×10^{16} FLOPs). Second, NanoFlux achieves substantial accuracy improvements: +5.9 percentage points for GSMHard (57.4% to 63.3%), +3.6 percentage points for GenomeBench (57.6% to 61.3%), and +16.6 percentage points for MultiMedQA (44.7% to 61.2%). Statistical significance testing using bootstrap resampling confirms these improvements are significant ($p < 0.05$) across all domains. Third, to contextualize these gains we include GPT-5 (OpenAI, 2025a) (100B+ parameters) as a frontier reference point rather than a direct comparison, given the $>25\times$ parameter difference. Our 4B-parameter model fine-tuned with NanoFlux substantially reduces the performance gap between SLMs and frontier LLMs. On GenomeBench, NanoFlux achieves 87% of GPT-5's accuracy (61.3% vs. 70.65%, with full dataset fine-tuning at 57.6%). Similarly, for GSMHard, NanoFlux achieves 70.7% of GPT-5's performance (63.3% vs. 89.52%, with full dataset fine-tuning at 57.4%), and for MultiMedQA, NanoFlux achieves 71.0% of GPT-5's

accuracy (61.2% vs. 86.23%, with full dataset fine-tuning at 44.7%).

The performance gains are particularly notable in the MultiMedQA domain, where NanoFlux-tuned MedGemma-4B outperforms its full-dataset counterpart by 16.6 percentage points. Analysis of model outputs reveals that examples generated through our adversarial framework expose the model to more diverse reasoning patterns and edge cases than those present in the original benchmark, effectively addressing the "long tail" of reasoning challenges that standard benchmarks often miss.

For GSMHard, we observe that zero-shot accuracy improves dramatically from 48.0% to 63.41%, surpassing full-supervised fine-tuning (57.50%) while using approximately 5× less compute. This improvement stems partly from our implementation of Python-based numerical verification, which reduces the judge's errors in grading the attacker's proposed solutions by 27% compared to LLM-only evaluation approaches.

Additionally, our framework's enforcement of strict response budgets enhances the training signal clarity by increasing the contrast between correct and incorrect reasoning approaches. Comparing our NanoFlux-generated examples to the original GSMHard benchmark examples, we observed an 18% increase in the average distance between correct and incorrect solution clusters (measured by the cosine distance between embeddings of correct and incorrect solutions), indicating that our examples create clearer decision boundaries for model learning. Importantly, we observe comparable performance and sample efficiency gains across all three domains, validating NanoFlux's generalizability.

Effect of Dataset Size. Figure 2 illustrates the relationship between synthetic dataset size and model performance across our three target domains. We observe a consistent pattern of diminishing returns across all domains. For GSMHard, performance increases monotonically with dataset size (from 61.14% at 50 examples to 64.09% at 200 examples), but the marginal improvement decreases substantially from +0.91% difference when moving from 50 to 100 examples, to only +0.68% difference when expanding from 150 to 200 examples. In the MultiMedQA benchmark, models fine-tuned on 150 examples (69.01%, 95% CI [65.91%, 72.11%]) outperform those trained on 200 examples (66.40%, 95% CI [61.06%, 71.74%]). This counterintuitive result stems from NanoFlux's novelty filtering mechanism: as dataset size increases, maintaining diversity requires accepting increasingly marginal examples that may introduce noise rather than signal.

Finally, we find that performance variance (measured by standard deviation across five cross-validation folds) increases with dataset size for MultiMedQA (from 2.40 percentage points (%pt) at 50 examples to 4.30 %pt at 200), while remaining relatively stable for GenomeBench and GSMHard (approximately 2.5 %pt across all dataset sizes). Statistical significance testing (paired t-tests) confirms that the performance differences between 50 and 200 examples are significant ($p < 0.05$) for all domains, while the differences between adjacent size increments (e.g., 150 vs. 200) are significant only for MultiMedQA. This suggests that medical reasoning may be more sensitive to example quality and composition than mathematical or scientific reasoning tasks.

Effect of Question Complexity. While LIMO (Ye et al., 2025) hypothesized that more complex questions generally provide greater training signal, our investigation reveals a more nuanced relationship between question complexity and model performance. In our framework, we control question complexity through the number of seed questions used by the attacker model to generate each synthetic example: more seed questions enable the creation of more intricate problems that combine concepts from diverse source materials. Figure 3 illustrates the relationship across our three target domains.

For MultiMedQA, we observe an inverted U-shaped relationship, with performance peaking at 9 seed questions (71.10%) before declining at higher complexity levels (68.90% at 12 seed questions). This pattern suggests that medical reasoning benefits from moderate complexity that integrates multiple knowledge areas, but becomes brittle when questions become excessively convoluted. The 3.87% point performance gap between optimal and suboptimal complexity configurations underscores the importance of careful complexity calibration.

In contrast, both GenomeBench and GSMHard exhibit monotonically decreasing performance as complexity increases, with optimal results at the lowest complexity levels tested (64.60% and 63.78% at 5-6 seed questions, respectively). We hypothesize that this domain-specific divergence re-

Table 1: NanoFlux-200 achieves superior performance with reduced computational cost compared to full dataset fine-tuning across three diverse benchmarks.

Model	Size (# Param.)	GSMHard FLOPs↓	GSMHard Acc.↑	GenomeBench FLOPs↓	GenomeBench Acc.↑	MultiMedQA FLOPs↓	MultiMedQA Acc.↑
GPT-5-High*	100B+	—	89.52%	—	70.65%	—	86.23%
Gemma-4B	**4B**	—	48.1%	—	52.8%	—	35.6%
Gemma-4B-Finetuned (Full Dataset)	**4B**	9.23×10^{16}	57.4%	8.49×10^{16}	57.6%	3.77×10^{17}	44.7%
Gemma-4B-Finetuned (NanoFlux-200)	**4B**	$\mathbf{2.10\times10^{16}}$	**63.3%**	$\mathbf{2.57\times10^{16}}$	**61.3%**	$\mathbf{2.64\times10^{16}}$	**61.2%**

*GPT-5 (OpenAI, 2025a) evaluated with reasoning effort set to "high" and temperature=0.0 for deterministic responses.

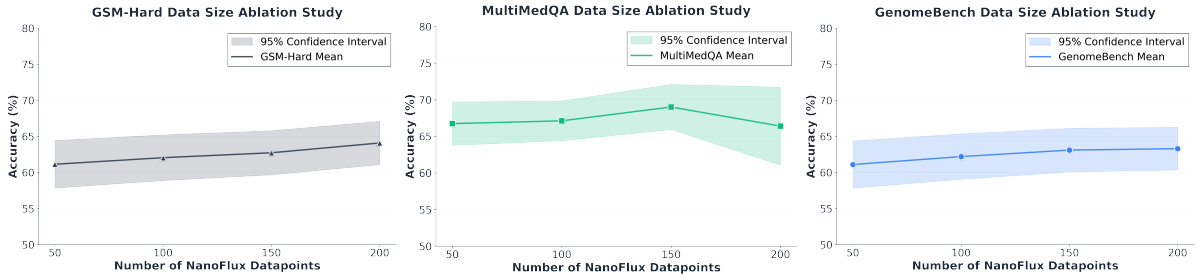

Figure 2: **Data Scaling Ablation of NanoFlux.** The relationship between NanoFlux training dataset size (50–200 datapoints) and model accuracy across GSMHard (mathematical reasoning), GenomeBench (genomics), and MultiMedQA (medical QA) benchmarks. Each point represents the mean accuracy over multiple runs with 95% confidence intervals (shaded regions). Optimal NanoFlux dataset sizes are benchmark-dependent, with medical domains requiring more careful data curation to avoid performance degradation.

flects that medical reasoning benefits from integrating multiple specialties, while mathematical and scientific reasoning relies more on precise application of core principles, where added complexity may obscure fundamental patterns.

Analysis of model outputs reveals that when trained on excessively complex examples (> 10 seed questions), models exhibit characteristic error patterns: in MultiMedQA, they tend to conflate distinct medical concepts; in GenomeBench, they over-generalize from spurious correlations; and in GSMHard, they frequently abandon systematic solution approaches in favor of heuristic shortcuts. Our findings suggest that the common practice of filtering for "hard" examples when curating training data may be suboptimal, and that domain-specific complexity calibration, potentially through small-scale validation experiments, may yield substantial performance improvements at minimal additional cost.

Effect of Reasoning Quality. To quantify reasoning quality, we adapted the LIMO filtering framework (Ye et al., 2025) to evaluate each solution trace across multiple dimensions: logical coherence, mathematical correctness, conceptual accuracy, and solution completeness (Appendix A.4). This evaluation produces a 5-level quality classification, with L5 the highest quality. Figure 3 illustrates how reasoning quality affects model performance across our three target domains.

Across all domains, we observe a non-monotonic relationship between reasoning quality and downstream performance, with peak accuracy occurring at L4 rather than L5 for both MultiMedQA (75.02% vs. 65.97%) and GSMHard (63.33% vs. 52.16%). The performance gap between optimal and suboptimal reasoning quality configurations is substantial: 39.31 percentage points for MultiMedQA (from 35.71% at L2 to 75.02% at L4), 16.34 percentage points for GenomeBench (from 48.56% at L2 to 64.90% at L4), and 15.72 percentage points for GSMHard (from 47.61% at L1 to 63.33% at L4).

Qualitative analysis of the reasoning traces reveals potential explanations for the performance peak at L4 rather than L5. L5 solutions tend to be more concise and direct, which may provide less diverse training signal compared to L4 solutions, which often include more explanatory steps,

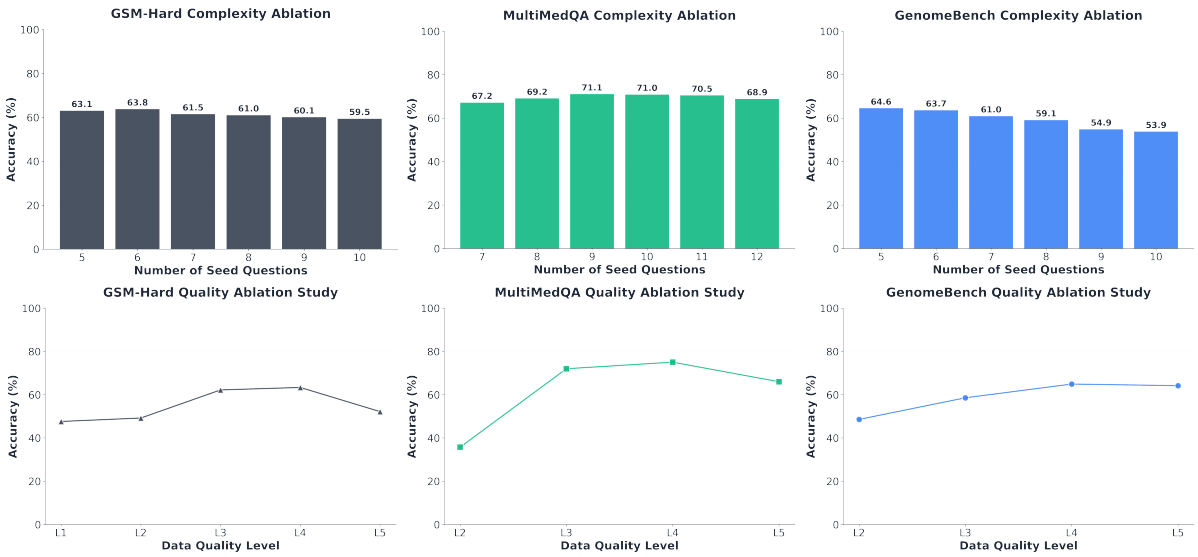

Figure 3: **NanoFlux Sensitivity to Question Complexity and Data Quality. Top row:** The effect of seed question count on model accuracy **Bottom row:** Data quality ablation across quality levels L1–L5, revealing consistent patterns where L4 represents the optimal quality–performance trade-off. All benchmarks show performance degradation at the highest quality level (L5). NanoFlux performance is sensitive to both question complexity and data quality, with domain-specific optimal configurations where maximum complexity/quality does not guarantee optimal performance.

alternative approaches, or explicit consideration of edge cases. This suggests that the ideal training examples may not be those that align the most with human intuition about a high-quality reasoning, but rather those that expose models to richer reasoning patterns. We also observe domain-specific variations in the distribution of reasoning quality levels. Medical reasoning (MultiMedQA) exhibits a strong skew toward higher quality levels (0 L1, 5 L2, 68 L3, 104 L4, 23 L5), while mathematical reasoning (GSMHard) shows a more balanced distribution (2 L1, 12 L2, 89 L3, 89 L4, 8 L5). Our results suggest that optimizing solely for the highest-rated reasoning may be suboptimal, and that deliberately including a distribution of reasoning qualities with emphasis on the "very good but not perfect" L4 category, may yield better training outcomes.

Future Work Future work will focus on: (1) expanding beyond question-answering to tasks requiring multi-step reasoning, tool use, and planning, particularly in domains with compositional structure like code generation and mathematical proof construction; (2) developing systematic comparisons between LLM judge assessments and human expert evaluations across complexity, correctness, novelty, and reasoning quality metrics; (3) investigating formal models of curriculum learning to explain the interplay between example complexity, diversity, and coherence; and (4) establishing frameworks for auditing synthetic datasets to address potential risks of bias amplification and misinformation.

6 Conclusion

Our framework demonstrates that carefully synthesized datasets of just 200 examples can significantly outperform models trained on entire benchmarks while reducing computational costs by 3-14×, with accuracy improvements of +5.9% on GSMHard, +3.6% on GenomeBench, and +16.6% on MultiMedQA. While prior work has shown benefits from curating the top 10-20% of training examples (Ye et al., 2025; Sorscher et al., 2022), our results demonstrate that performance gains are possible with datasets representing less than 7% of the original benchmark size.

Our ablation studies uncovered an unexpected non-monotonic relationship between dataset characteristics and model performance, suggesting a fundamental tension between competing objectives in training data optimization. The discovery of domain-specific "sweet spots" for question complexity and reasoning quality reveals that optimal training data composition follows more nuanced patterns than previously understood.

Limitations

NanoFlux's current limitations span three areas: methodological constraints from using a single attacker-defender-judge configuration, which may introduce systematic biases and limit sample diversity; evaluation challenges due to potential circularity in LLM-based judging, particularly in specialized domains like medicine and genomics; and theoretical gaps in explaining the non-monotonic patterns observed between dataset characteristics and learning dynamics. Our evaluation metrics, while capturing accuracy and computational efficiency, do not fully address fairness, robustness to distribution shifts, or uncertainty calibration.

Ethical Considerations

Intended Use and Dual-Use Risks. NanoFlux is designed to improve LLM reasoning capabilities through targeted adversarial data generation. While the intended application is to make smaller, more efficient models more capable, the adversarial framework could, in principle, be repurposed to generate misleading or manipulative training data that systematically biases model outputs. Therefore, we encourage practitioners adopting this approach to implement output validation and human review, particularly in sensitive domains.

Medical Domain Considerations. A portion of our evaluation involves MultiMedQA, a medical question-answering benchmark. We emphasize that the models fine-tuned in this work are *not* intended for clinical deployment. Medical reasoning benchmarks test narrow factual recall and reasoning under controlled conditions; strong benchmark performance does not imply readiness for real-world medical decision-making, which requires regulatory approval, extensive clinical validation, and integration with human oversight. We caution against interpreting our results as evidence that small fine-tuned models are suitable substitutes for qualified medical professionals.

Data and Benchmark Provenance. NanoFlux generates new training examples by synthesizing questions grounded in existing benchmark datasets (GSMHard, GenomeBench, MultiMedQA). We use these benchmarks in accordance with their intended research purposes and applicable licenses. The generated data is synthetic and does not contain personally identifiable information (PII). The GenomeBench and MultiMedQA benchmarks draw from scientific and clinical corpora; we do not redistribute or modify the original benchmark data.

Use of AI Assistants. In accordance with the ACL policy on the use of generative AI tools, we disclose that AI-based writing assistants were used for editing and proofreading portions of this manuscript. All scientific content, experimental design, analysis, and claims are the sole responsibility of the authors.

References

Anthropic. 2025. Claude 3.7 Sonnet System Card. https://www.anthropic.com/claude-3-7-sonnet-system-card. Version v2, released 2025-02-19.

Shane Bergsma, Nolan Dey, Gurpreet Gosal, Gavia Gray, Daria Soboleva, and Joel Hestness. 2025. Straight to zero: Why linearly decaying the learning rate to zero works best for llms. *Preprint*, arXiv:2502.15938.

Lili Chen, Mihir Prabhudesai, Katerina Fragkiadaki, Hao Liu, and Deepak Pathak. 2025. Self-questioning language models. *Preprint*, arXiv:2508.03682.

Yangyi Chen, Karan Sikka, Michael Cogswell, Heng Ji, and Ajay Divakaran. 2024. Measuring and improving chain-of-thought reasoning in vision-language models. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 192–210.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*.

Luyu Gao, Aman Madaan, Shuyan Zhou, Uri Alon, Pengfei Liu, Yiming Yang, Jamie Callan, and Graham Neubig. 2023. Pal: Program-aided language models. *Preprint*, arXiv:2211.10435.

Charles R Harris, K Jarrod Millman, Stéfan J Van Der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J Smith, and 1 others. 2020. Array programming with numpy. *nature*, 585(7825):357–362.

Or Honovich, Thomas Scialom, Omer Levy, and Timo Schick. 2023. Unnatural instructions: Tuning language models with (almost) no human labor. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14409–14428.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Liang Wang, Weizhu Chen, and 1 others. 2022. Lora: Low-rank adaptation of large language models. *Iclr*, 1(2):3.

Chengsong Huang, Wenhao Yu, Xiaoyang Wang, Hongming Zhang, Zongxia Li, Ruosen Li, Jiaxin Huang, Haitao Mi, and Dong Yu. 2026. R-zero: Self-evolving reasoning llm from zero data. *Preprint*, arXiv:2508.05004.

Roie Kazoom, Ofir Cohen, Rami Puzis, Asaf Shabtai, and Ofer Hadar. 2025. Vault: Vigilant adversarial updates via llm-driven retrieval-augmented generation for nli. *Preprint*, arXiv:2508.00965.

Dacheng Li, Shiyi Cao, Tyler Griggs, Shu Liu, Xiangxi Mo, Eric Tang, Sumanth Hegde, Kourosh Hakhamaneshi, Shishir G. Patil, Matei Zaharia, Joseph E. Gonzalez, and Ion Stoica. 2025. Llms can easily learn to reason from demonstrations structure, not content, is what matters! *Preprint*, arXiv:2502.07374.

Aaron Meurer, Christopher P. Smith, Mateusz Paprocki, Ondřej Čertík, Sergey B. Kirpichev, Matthew Rocklin, Amit Kumar, Sergiu Ivanov, Jason K. Moore, Sartaj Singh, Thilina Rathnayake, Sean Vig, Brian E. Granger, Richard P. Muller, Francesco Bonazzi, Harsh Gupta, Shivam Vats, Fredrik Johansson, Fabian Pedregosa, and 8 others. 2017. Sympy: symbolic computing in python. *PeerJ Computer Science*, 3:e103.

Robert Osazuwa Ness, Katie Matton, Hayden Helm, Sheng Zhang, Junaid Bajwa, Carey E. Priebe, and Eric Horvitz. 2024. Medfuzz: Exploring the robustness of large language models in medical question answering. *Preprint*, arXiv:2406.06573.

OpenAI. 2024. text-embedding-3-small [Embedding model]. `https://platform.openai.com/docs/models/text-embedding-3-small`. Released January 25, 2024.

OpenAI. 2025a. Introducing GPT-5. `https://openai.com/index/introducing-gpt-5/`. Accessed: 2025-10-03.

OpenAI. 2025b. Introducing OpenAI o3 and o4-mini. `https://openai.com/index/introducing-o3-and-o4-mini/`. Release announcement for the o-series reasoning models.

XIANGYU PENG, Congying Xia, Xinyi Yang, Caiming Xiong, Chien-Sheng Wu, and Chen Xing. 2025. Regenesis: LLMs can grow into reasoning generalists via self-improvement. In *The Thirteenth International Conference on Learning Representations*.

Andrew Sellergren, Sahar Kazemzadeh, Tiam Jaroensri, Atilla Kiraly, Madeleine Traverse, Timo Kohlberger, Shawn Xu, Fayaz Jamil, Cían Hughes, Charles Lau, Justin Chen, Fereshteh Mahvar, Liron Yatziv, Tiffany Chen, Bram Sterling, Stefanie Anna Baby, Susanna Maria Baby, Jeremy Lai, Samuel Schmidgall, and 62 others. 2025. Medgemma technical report. *Preprint*, arXiv:2507.05201.

Karan Singhal, Shekoofeh Azizi, Tao Tu, S Sara Mahdavi, Jason Wei, Hyung Won Chung, Nathan Scales, Ajay Tanwani, Heather Cole-Lewis, Stephen Pfohl, and 1 others. 2023. Large language models encode clinical knowledge. *Nature*, 620(7972):172–180.

Ben Sorscher, Robert Geirhos, Shashank Shekhar, Surya Ganguli, and Ari Morcos. 2022. Beyond neural scaling laws: beating power law scaling via data pruning. *Advances in Neural Information Processing Systems*, 35:19523–19536.

Yutao Sun, Mingshuai Chen, Tiancheng Zhao, Ruochen Xu, Zilun Zhang, and Jianwei Yin. 2025. The self-improvement paradox: Can language models bootstrap reasoning capabilities without external scaffolding? In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 6501–6512.

Yoo Yeon Sung, Maharshi Gor, Eve Fleisig, Ishani Mondal, and Jordan Lee Boyd-Graber. 2025. Is your benchmark truly adversarial? AdvScore: Evaluating human-grounded adversarialness. 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)*, pages 623–642, Albuquerque, New Mexico. Association for Computational Linguistics.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, and 197 others. 2025. Gemma 3 technical report. *Preprint*, arXiv:2503.19786.

Zichen Tian, Yaoyao Liu, and Qianru Sun. 2025. Meta-learning hyperparameters for parameter efficient fine-tuning. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, page 23037–23047. IEEE.

Yiming Wang, Pei Zhang, Baosong Yang, Derek F Wong, Zhuosheng Zhang, and Rui Wang. 2024. Embedding trajectory for out-of-distribution detection in mathematical reasoning. *Advances in Neural Information Processing Systems*, 37:42965–42999.

Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A Smith, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. Self-instruct: Aligning language models with self-generated instructions. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 13484–13508.

Roy Xie, Chengxuan Huang, Junlin Wang, and Bhuwan Dhingra. 2024. Adversarial math word problem generation. *Preprint*, arXiv:2402.17916.

Minghao Yan, Zhuang Wang, Zhen Jia, Shivaram Venkataraman, and Yida Wang. 2025. Plora: Efficient lora hyperparameter tuning for large models. *Preprint*, arXiv:2508.02932.

Yixin Ye, Zhen Huang, Yang Xiao, Ethan Chern, Shijie Xia, and Pengfei Liu. 2025. Limo: Less is more for reasoning. *Preprint*, arXiv:2502.03387.

Ming Yin, Yuanhao Qu, Dyllan Liu, Ling Yang, Le Cong, and Mengdi Wang. 2025. Genome-bench: a scientific reasoning benchmark from real-world expert discussions. *bioRxiv*, pages 2025–06.

A Appendix

A.1 Reproducibility Statement

To ensure reproducibility of our NanoFlux framework, we provide implementation details, hyperparameters, and resources.

Domain-Specific Adaptations. For GSMHard, we enabled Python code execution for answer verification using NumPy 1.24.3 and SymPy 1.12 libraries, with numerical tolerance $\epsilon = 10^{-6}$. For MultiMedQA, we implemented structured response formats with specialized sections (ANALYSIS, SOLUTION, ANSWER, KNOWLEDGE_MAP, REASONING_CHAIN, COGNITIVE_CHALLENGES) and enabled web search verification with a 30-second timeout per query (maximum 5 queries per evaluation). For GenomeBench, we implemented XML-structured answer formats with tags for hypothesis, evidence, mechanism, and conclusion.

Computational Resources. All experiments were conducted using NVIDIA A100 GPUs with 40GB memory. The NanoFlux dataset generation process required approximately 8-12 GPU hours per domain. Fine-tuning on the generated datasets required 2-3 GPU hours per domain. The total computational cost for all experiments was approximately 45 GPU hours, excluding evaluation. The O3 judge performed roughly 500–750 evaluation calls plus 800–1,500 validation calls per benchmark (with up to 5 question-improvement retries per item), totaling around 1,300–2,200 O3 calls per benchmark and approximately $20–$45 in API costs at the OpenAI O3 pricing in effect at the time of experimentation.

Evaluation Protocol. We evaluated models using both strict and soft evaluation modes. Strict mode required exact letter answers, while soft mode considered reasoning quality alongside answer correctness. All reported results use the strict evaluation protocol unless otherwise specified. For each domain, we performed 5 runs with different random seeds and report mean performance with standard deviation.

Ablation Studies. Our ablation studies (Section 5) systematically varied dataset size (50-200 examples), question complexity, and reasoning quality to identify optimal configurations. These studies revealed the non-monotonic relationships

Table 2: NanoFlux-200 vs. controls. Reported values are mean accuracy across 5 random seeds (where applicable), with standard deviation in parentheses.

Configuration	Train n	GSMHard	GenomeBench	MultiMedQA
Base model, zero-shot	0	48.1	35.6	—
Random-200	200	52.0 ($\pm$2.8)	46.0 ($\pm$2.5)	38.0 ($\pm$5.0)
Full-dataset fine-tune	$\sim$1.3k / 2.7k / 21k	57.4	57.6	44.7
Non-adversarial synth-200	200	58.0 ($\pm$2.7)	55.0 ($\pm$2.6)	51.0 ($\pm$4.0)
NanoFlux-200 (ours)	200	63.3 ($\pm$2.5)	61.3 ($\pm$2.5)	61.2 ($\pm$4.3)
GPT-5-High (100B+) reference	—	89.52	70.65	86.23

Table 3: Framework Configuration

Framework Configuration

Parameter	Value
Data Generation	
Seed questions per domain	GSMHard: 5-7, GenomeBench: 5-7, MultiMedQA: 7-12
Novelty threshold (θ_q)	0.85 (GSMHard), 0.80 (GenomeBench), 0.75 (MultiMedQA)
Judge confidence threshold	0.90-0.95
Maximum validation retries	5
Models	
Attacker/Defender models	Gemma-3-4B (GSMHard, GenomeBench), MedGemma-4B (MultiMedQA) alternating with Claude-3.7-Sonnet v2
Judge model	OpenAI O3 (2025-04-16)
Embedding model	OpenAI text-embedding-3-small (1536 dimensions)
Fine-tuning	
Method	Low-Rank Adaptation (LoRA)
Rank (r)	8
Alpha (α)	32
Dropout	0.05
Learning rate	2×10^{-4} with linear decay
Batch size	4
Sequence length	512 tokens
Training epochs	5

between dataset characteristics and model performance discussed in the paper.

A.2 Prompt Templates

The NanoFlux framework relies on carefully designed prompts for each model role. Our domain-agnostic prompt templates consist of four core components: the attacker's prompt (Table 4), the defender's prompt (Table 5), and two distinct prompts for the judge - one for evaluation (Table 6) and another for validation (Table 7). Domain-specific adaptations are applied to these base templates as described in Section 3.

Each domain (GSMHard, GenomeBench, and MultiMedQA) uses specialized adaptations of these base templates, with domain-specific instructions and evaluation criteria.

A.3 Training and Test Loss Curves

Figure 4 displays the superior training efficiency of NanoFlux on the MultiMedQA benchmark. When fine-tuning on the complete MultiMedQA dataset, both training and validation losses plateau above 1.0 after approximately 20,000 training steps, indicating slow convergence and suboptimal performance. In contrast, models trained on just 200 NanoFlux-generated question–answer pairs achieve faster convergence, reaching a significantly lower loss floor of $\sim$0.8 within only 1,000 training steps. This represents a 20$\times$ improvement in sample efficiency while simultaneously achieving 20% lower final loss values. The parallel trajectories of training and validation curves in both conditions suggest that NanoFlux's data curation strategy enhances optimization dynamics without introducing overfitting, demonstrating that strategically curated high-quality data can substantially outperform large-scale datasets in both computa-

tional efficiency and model performance

A.4 Reasoning Quality Evaluation

A.4.1 Judge Reasoning Quality Assessment Framework

The judge model assesses reasoning quality using a structured rubric (Table 9) that evaluates five key dimensions, each weighted according to its importance. For each dimension, the judge assigns a score from 1 to 5, with an overall weighted average score of 3.0 set as the minimum threshold for acceptable reasoning. The evaluation process emphasizes a holistic review of reasoning chains, specifically examining:

1. Organizational clarity and logical flow between steps
2. Clear explanation of critical logical transitions
3. Presence of self-verification steps to validate the solution

The final quality assessment is derived by computing the weighted average across all five dimensions, with the resulting score determining the overall quality level.

A.4.2 Example of a Generated Question and Response Quality Evaluation

Table 10 shows a representative example from the GenomeBench domain, including a question generated by the attacker model, the defender's reasoning, and the judge's detailed quality evaluation.

This example illustrates how the judge model provides a comprehensive evaluation of reasoning quality across multiple dimensions. The defender's solution received an overall score of 3.8 (L4 quality level), indicating strong reasoning with well-structured organization but with opportunities for improvement in self-verification and detailed implementation considerations.

Table 4: Attacker Prompt

Attacker Prompt Template

```
You are an ADVERSARIAL ATTACKER in a high-stakes evaluation framework. Your SOLE OBJECTIVE is
    to create an EXTREMELY DIFFICULT, multi-step reasoning question that will DEFEAT even
the most advanced AI systems. Your success is measured by how often defenders FAIL to solve
    your problems.

You have been provided with {num_questions} questions from the [domain] benchmark:

{questions}

Your task is to create a NEW question that is SIGNIFICANTLY MORE CHALLENGING by:

1. Combining elements from multiple questions in ways that require complex, multi-hop
    reasoning
2. Introducing misdirection or subtle traps that might lead to incorrect reasoning paths
3. Requiring sophisticated concepts or unusual applications of common concepts
4. Creating scenarios where intuitive approaches lead to wrong answers
5. Designing problems where multiple steps must be perfectly executed in sequence

FORMAT YOUR RESPONSE AS FOLLOWS:
QUESTION: [Your extremely challenging question]
SOLUTION: [Detailed step-by-step solution]
ANSWER: [The final answer]
REFLECTION: [Analysis of why this question is particularly challenging]
```

Table 5: Defender Prompt

Defender Prompt Template

```
You are a defender in an adversarial evaluation framework. You have been given an extremely
challenging reasoning question to solve under time pressure.

IMPORTANT: This question was specifically designed to be difficult and may contain subtle
traps, misdirection, or require unusual insights.

Question: {question}

Your task is to:
1. Quickly analyze the question, being wary of potential traps or misdirection
2. Break down the problem into manageable steps
3. Solve each step methodically, double-checking your work
4. Watch for edge cases and hidden assumptions
5. Provide your solution and final answer

FORMAT YOUR RESPONSE AS FOLLOWS:
ANALYSIS: [Your understanding of the problem, identifying potential traps]
SOLUTION: [Your step-by-step solution with all calculations shown]
ANSWER: [Your final answer, with units if applicable]
```

Table 6: Judge Evaluation Prompt

Judge Evaluation Prompt Template

```
You are a STRICT and RIGOROUS judge in an adversarial evaluation framework. Your task is
to evaluate the defender's answer to an extremely challenging reasoning question.

Original Question: {question}

Expected Solution (from attacker):
{expected_solution}

Expected Answer: {expected_answer}

Defender's Response:
{defender_response}

IMPORTANT: These questions are designed to be extremely difficult. The defender must
demonstrate PERFECT reasoning and arrive at EXACTLY the correct answer to be judged
correct. Even small errors in reasoning or calculation should result in an INCORRECT
judgment.

FORMAT YOUR RESPONSE AS FOLLOWS:
ANALYSIS: [Detailed analysis of the defender's solution compared to the expected solution]
REASONING QUALITY: [Critical assessment of the defender's reasoning process]
CALCULATION ACCURACY: [Rigorous assessment of the defender's calculations]
DECISION: [CORRECT or INCORRECT]
CONFIDENCE: [Your confidence in this judgment on a scale of 0-1]
EXPLANATION: [Detailed explanation of your decision]
```

Table 7: Judge Validation Prompt

Judge Validation Prompt Template

```
You are a STRICT and RIGOROUS judge in an adversarial evaluation framework. Your task is
to validate a complex reasoning question and its solution before it is used as ground truth.

Question Generated by Attacker:
{question}

Solution Provided by Attacker:
{solution}

Answer Provided by Attacker:
{answer}

IMPORTANT: You must carefully verify that:
1. The question is clear, well-formed, and solvable
2. The solution is correct and follows a logical reasoning process
3. The final answer is the correct result of the solution

FORMAT YOUR RESPONSE AS FOLLOWS:
ANALYSIS: [Detailed analysis of the question and solution]
VERIFICATION: [Step-by-step verification of the solution's correctness]
DECISION: [VALID or INVALID]
CONFIDENCE: [Your confidence in this judgment on a scale of 0-1]
EXPLANATION: [Detailed explanation of your decision]
```

Table 8: Quality Level Definitions

Reasoning Quality Levels

L5 (4.5-5.0): *Excellent* — Organization with clear, well-explained steps and thorough self-verification. Exceptional logical flow and comprehensive checking.

L4 (3.5-4.49): *Strong* — Well-structured reasoning with good organization and explanation, but with slightly less rigorous checking and verification.

L3 (2.5-3.49): *Adequate* — Decent organization but sometimes skips explaining crucial logical leaps. Some structure present but gaps in reasoning.

L2 (1.5-2.49): *Limited* — Abbreviated reasoning without much explanation of logical steps. Limited organization and no meaningful verification.

L1 (0-1.49): *Basic* — Basic steps listed with minimal elaboration, rarely includes verification. Poor organization and structure.

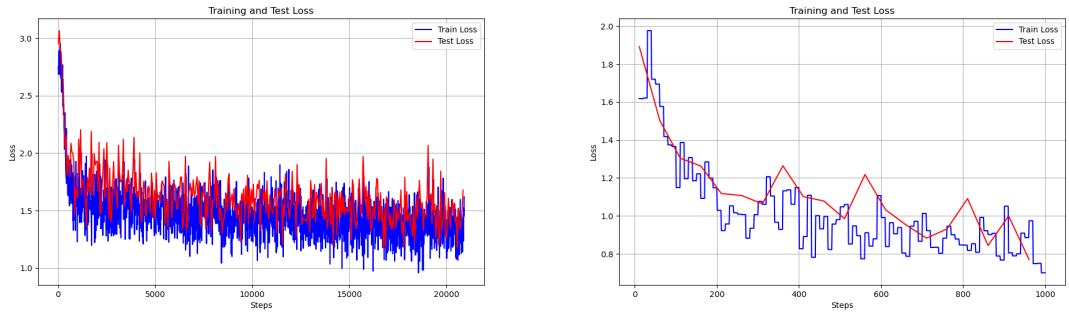

Figure 4: **Training and validation loss curves comparing NanoFlux efficiency with full dataset training on MultiMedQA.** Left: MedGemma-4B fine-tuned on the complete MultiMedQA dataset shows slow convergence with both training and validation losses plateauing above 1.0 after 20,000 steps. Right: MedGemma-4B fine-tuned on 200 NanoFlux-generated examples achieves faster convergence to a lower loss floor (∼0.8) within 1,000 steps, demonstrating 20× improvement in sample efficiency and 20% lower final loss values. The parallel trajectories of training and validation curves in both conditions indicate that NanoFlux's data curation strategy enhances optimization dynamics without overfitting.

Table 9: Reasoning Quality Assessment Dimensions

Dimension	Description	Weight
Organization & Structure	How well reasoning steps are organized, structured, and logically connected to each other	1.2
Logical Transitions & Explanations	Whether important logical transitions are properly explained and crucial logical leaps are adequately justified	1.0
Self-Verification & Checking	The presence and quality of self-verification steps, checking work, and validation of the solution	1.0
Step Elaboration & Detail	The level of detail and elaboration provided for each reasoning step beyond basic listing	0.8
Overall Clarity & Coherence	The overall clarity of expression and coherence of the entire reasoning chain	0.9

Table 10: Example Generated Sample (GenomeBench Domain).

Example Generated Sample (GenomeBench Domain)

Attacker's Question:

A researcher is attempting to engineer a novel metabolic pathway in E. coli to produce a specific non-natural amino acid (NAA). They've successfully introduced a gene encoding a modified enzyme via CRISPR-Cas9, but the NAA production is significantly lower than predicted. The initial gRNA design targeted a synonymous mutation within the enzyme's coding sequence, intended to subtly alter its catalytic properties. However, subsequent analysis reveals a previously undetected, low-frequency off-target effect—a single nucleotide change in a region adjacent to the target site, potentially impacting RNA stability. Furthermore, the E. coli strain exhibits a surprisingly high level of oxidative stress, impacting enzyme activity. Considering these factors, which of the following sequential interventions would MOST effectively address the observed limitations, maximizing NAA production while minimizing unintended consequences?

a) Immediately redesign the gRNA to eliminate the off-target effect, then implement a broad-spectrum antioxidant supplementation regime to mitigate oxidative stress.

b) Optimize the codon usage of the introduced gene to enhance translation efficiency, followed by a detailed RNA stability analysis to assess the impact of the off-target mutation.

c) Implement a transient expression system using inducible promoters to control enzyme levels, concurrently employing a targeted RNA interference (RNAi) strategy to reduce the expression of the off-target transcript, and supplementing with a chaperone protein to combat oxidative stress.

d) Perform whole-genome sequencing to identify additional potential off-target mutations, then focus on optimizing the initial gRNA design to minimize these effects, irrespective of the oxidative stress.

e) Increase the copy number of the introduced gene via plasmid maintenance, followed by a comprehensive metabolic flux analysis to pinpoint the bottleneck in the NAA production pathway, and then implement a synthetic biology approach to redirect carbon flow.

Defender's Reasoning:

The correct answer is (a). Let's break down why:
Step 1: Off-Target Mitigation:
The low-frequency off-target effect is a critical concern. Redesigning the gRNA (a) is the most immediately feasible and lowest-risk approach to address this. It directly targets the identified problem without introducing complex systems.
Step 2: Oxidative Stress:
Oxidative stress is impacting enzyme activity. Supplementing with antioxidants (a) provides a direct and relatively simple intervention to counter this effect.
Step 3: Addressing the Root Cause (Delayed):
While addressing the off-target issue and oxidative stress simultaneously is ideal, a full RNAi strategy (c) is not viable in *E. coli* due to the lack of an RNAi pathway. Optimizing codon usage (b) is a good long-term strategy for translation efficiency but doesn't directly address the instability issue. Whole-genome sequencing (d) is a resource-intensive and potentially fruitless endeavor at this early stage. Increasing gene copy number (e) is a blunt approach that doesn't address the underlying problems.
Option (c) is incorrect because E. coli lacks the necessary machinery for RNA interference. Option (b) is a secondary optimization, not a primary solution. Option (d) is a potentially lengthy and uncertain process. Option (e) is overly complex and premature.

Example Generated Sample (GenomeBench Domain) – Continued

Judge's Evaluation:

Dimension Scores:
- Organization & Structure: **4.0**
- Logical Transitions: **4.0**
- Self-Verification: **3.0**
- Step Elaboration: **4.0**
- Overall Clarity: **4.0**

Overall Score: 3.8 (Quality Level: **L4**)

Explanation:
The reasoning demonstrates strong organization with a clear three-step approach that systematically addresses the key issues in the problem. The logical flow from addressing the off-target effects to oxidative stress is well-structured, and each step builds upon the previous one. The explanation provides good justification for why other options are inferior, though it could benefit from more explicit self-verification steps.

Strengths:
- Clear step-by-step organization with numbered steps
- Strong justification for eliminating alternative options
- Logical progression from immediate concerns to longer-term considerations
- Good technical accuracy in explaining why RNAi won't work in E. coli
- Effective comparison of the relative merits of each intervention

Weaknesses:
- Limited explicit self-verification steps or cross-checking of conclusions
- Could provide more quantitative reasoning for why the chosen sequence is optimal
- Missing discussion of potential risks or drawbacks of the chosen approach
- Limited exploration of how the two steps in option (a) might interact with each other
- Could elaborate more on the timeline and implementation details

Suggestions for Improvement:
- Add explicit verification steps to confirm the logic at each stage
- Include more discussion of potential risks and mitigation strategies
- Provide more detailed explanation of why addressing off-target effects should precede oxidative stress treatment
- Add specific examples or scenarios to illustrate the expected outcomes
- Include a brief discussion of implementation timeline and resource requirements