

PocketSheet: Enhancing Test-Time Learning using Efficient Memory Augmentation in Small Language Models

Stanford CS224N Custom Project

Sakthivel Sivaraman
Department of Computer Science
Stanford University
sakthiv@stanford.edu

Prabhjot Singh Rai
Department of Computer Science
Stanford University
prabhjot@stanford.edu

Abstract

Dynamic Cheatsheets (DC) let LLMs maintain an evolving external memory across test-time problems, yielding dramatic gains for frontier models (GPT-4o: 10% to 98% on Game of 24). Small models ($\leq 14B$), however, *degrade* under DC, as confirmed by both the original paper and our diagnostics (Qwen-7B: 10% to 4%).

We present **PocketSheet**, which isolates the failure as a *memory-quality* bottleneck and attacks it through three components: (1) **teacher-trajectory SFT**, which distills GPT-4.1 DC-Cu trajectories via QLoRA, raising unsummarized DC-Cu from 4% to 28% on Game of 24; we additionally find that SFT models benefit substantially from sampling (temperature 1.0 doubles accuracy over greedy), revealing mode collapse that motivates RL. (2) **Cheatsheet summarization**, which compresses unbounded memory to a fixed token budget, boosting our SFT model to 55% on Game of 24. (3) **GRPO** with citation-grounding and cheatsheet-conciseness rewards, which builds on the SFT model to target the remaining reasoning gap while reinforcing compact memory writing. A key finding is that *neither fix alone suffices*: summarization without SFT yields only 7% (the base model cannot produce useful cheatsheet content to summarize), while SFT without summarization reaches 28% (good content, but overwhelmed by length). Only the combination achieves 55%, revealing two compounding bottlenecks, content quality and memory length, that must both be addressed. Code is available at <https://github.com/tinyllms/pocket-sheet>

1 Key Information to include

- Mentor: Mirac Suzgun (msuzgun@stanford.edu)
- External Collaborators (if you have any): No
- Sharing project: No
- Late Days (if you are using any):

Team member	Late days remaining
Sakthivel Sivaraman	2
Prabhjot Singh Rai	4
Total pooled	6
Team size	2
Late days applied for the report	3

2 Introduction

Large Language Models excel at single-turn reasoning but struggle to *learn from experience* across a sequence of test-time problems without weight updates. The Dynamic Cheatsheet (DC) framework [1] addresses this by maintaining a persistent external memory that a *generator* reads from and a *curator* writes to after each problem. For frontier models such as GPT-4o, DC-Cumulative (DC-Cu) yields dramatic accuracy improvements—from 10% to 93% on Game of 24. However, smaller models ($\leq 14\text{B}$ parameters) show no improvement or even *degradation* under DC.

We confirm this degradation (Qwen-7B: 10% baseline $\rightarrow$ 4% under DC-Cu) and diagnose its root cause. Our experiments reveal **two compounding bottlenecks**: the model writes poor cheatsheet content (low quality), and that content grows unbounded ($\sim 8\text{K}$ tokens), overwhelming the effective context window [2]. Our project, **PocketSheet**, addresses this through a sequential three-stage pipeline:

1. **Teacher-trajectory SFT**: distilling GPT-4.1 DC-Cu trajectories into Qwen-7B via QLoRA [3], teaching it to read and write cheatsheets, raising unspecialized DC-Cu from 4% to 28%.
2. **Cheatsheet summarization**: compressing the cheatsheet to a fixed token budget via a teacher model, boosting the SFT model to 55% on Game of 24.
3. **GRPO refinement** [4]: applying Group Relative Policy Optimization with correctness, format, citation-grounding, and cheatsheet-conciseness rewards to target the remaining reasoning gap while reinforcing compact memory writing.

Crucially, *neither fix alone suffices*: summarization without SFT yields only 7% (the base model writes cheatsheet content too poor to benefit from compression), while SFT without summarization reaches 28% (good content, but unbounded growth overwhelms the context window). Only their combination achieves 55%, revealing two compounding bottlenecks—content quality and memory length—that must both be resolved before the remaining reasoning gap (55% vs. 98%) becomes the frontier, which GRPO is designed to close.

3 Related Work

Dynamic Cheatsheet. Wang et al. [1] introduced the DC framework, demonstrating that a retrieve-generate-curate loop dramatically improves test-time performance for large models across mathematical, algorithmic, and scientific reasoning tasks. They identify that smaller models fail to benefit, hypothesizing insufficient base competence and poor memory curation quality.

Knowledge distillation for reasoning. Shridhar et al. [5] and Mitra et al. [6] show that distilling chain-of-thought traces from large models into smaller ones can substantially improve reasoning capabilities. We extend this to the structured DC setting, distilling not just reasoning traces but complete memory-interaction trajectories (generator + curator roles).

Efficient fine-tuning. LoRA [7] and QLoRA [3] enable fine-tuning of large models under memory constraints by inserting trainable low-rank adapters while keeping the base model frozen or quantized. We use QLoRA (4-bit NF4 + rank-64 LoRA) to fine-tune Qwen-7B on consumer hardware.

GRPO for alignment. DeepSeekMath [4] introduces Group Relative Policy Optimization, a variant of PPO that uses group-level reward normalization. We augment standard correctness rewards with citation-format and memory-grounding rewards to specifically target the DC failure modes.

4 Approach

Problem. The DC framework uses two LLM roles: a *generator* that solves problems given a cheatsheet, and a *curator* that updates the cheatsheet after each problem. For large models, DC-Cu achieves strong gains; for small models ($\leq 14\text{B}$), the cheatsheet grows unbounded (up to $\sim 25\text{K}$ characters), becoming incoherent, and the model cannot leverage it.

Diagnostic framework. We design three conditions to isolate failure modes:

1. **Baseline** (no memory): measures raw generation competence.
2. **DC-Cumulative**: the full generate-then-curate loop.
3. **DC-Cu + Summarization**: DC-Cu with periodic cheatsheet compression to a fixed token budget (e.g., 1000 tokens) via a teacher summarizer (GPT-4.1).

Cheatsheet summarization. When the cheatsheet exceeds a configurable token threshold, a summarizer model condenses it while preserving high-value strategies. This addresses the “lost in the middle” problem [2] where small models degrade with long contexts, and prevents low-quality entries from drowning out useful ones.

Teacher trajectory generation. We generate DC-Cu trajectories using GPT-4.1 [1] and Gemini-2.5-Pro [8] across all training benchmarks. Each trajectory records the complete interaction: the cheatsheet state, generator response, curator update, and correctness signal. These are parsed into chat-format SFT data with separate examples for generator turns (teaching problem-solving with memory) and curator turns (teaching memory writing).

SFT + GRPO pipeline. We fine-tune Qwen-2.5-7B-Instruct on teacher trajectories using QLoRA, then apply GRPO with a composite reward combining (a) task-specific correctness, (b) answer format compliance, and (c) citation-grounding rewards that check whether the model’s reasoning references cheatsheet entries via structured `<cite>` tags.

DDP for Gradient Flow. To manage the computational overhead of GRPO, which requires maintaining policy and reference models while generating multiple outputs per prompt, we implement a Distributed Data Parallel (DDP) strategy optimized for QLoRA. We utilize Ray + TRL to shard training batches across 2 GPUs (H100s), maintaining local copies of the low-rank adapters and synchronizing gradients via AllReduce operations. To maintain flow stability and prevent memory bottlenecks during long-context memory processing (up to 16384 tokens), we also gradient checkpointing and DeepSpeed Stage 2 (ZeRO) optimization.

5 Experiments

5.1 Data

Evaluation benchmarks. We evaluate on three benchmarks from the DC paper: **Game of 24** (100 test problems; generate arithmetic expressions equaling 24) [9], **AIME 2024–2025** (60 competition math problems; exact integer answer) [10], and **GPQA** (Main: 448, Extended: 546, Diamond: 198 graduate-level science MCQs) [11].

Teacher trajectory data. We generated DC-Cu trajectories using GPT-4.1 (default and teacher prompt styles) and Gemini-2.5-Pro (default prompt style) across:

- **AIME 1983–2023**: 41 years, 615 problems total, yielding 1,838 SFT examples (generator + curator).
- **GPQA Main**: 448 questions $\times$ 3 teacher configs (GPT-4.1 default/teacher, Gemini default), yielding 1,512 SFT examples.
- **GPQA Extended**: 546 questions $\times$ 2 configs, yielding 1,526 SFT examples.
- **Game of 24 (NLI-LE)**: medium and hard splits $\times$ 2 prompt styles, yielding 408 SFT examples.

All datasets are published on HuggingFace Hub under the `tinyllms` organization with per-model, per-prompt-style configs.

5.2 Evaluation Method

Game of 24 uses functional verification (the predicted expression must evaluate to exactly 24 using only the four given numbers). AIME uses exact integer match. GPQA uses option-letter match. All metrics are accuracy (%).

5.3 Experimental Details

Models. Teacher: **GPT-4.1** (Azure) and **Gemini-2.5-Pro** (Vertex AI). Student: **Qwen-2.5-7B-Instruct** [12] (Q8 via Ollama for evaluation, QLoRA for training). Additional baselines: **Qwen-2.5-14B-Instruct** (Q8), **DeepSeek-R1-7B** (distilled).

Hardware. Local evaluation: NVIDIA RTX A5880 (48 GB) via Ollama. SFT training: NVIDIA RTX A5880 (QLoRA, 4-bit) and NVIDIA H100 (80 GB, full fine-tune). vLLM serving for SFT-trained models on Azure GPU instances.

Training configurations. *QLoRA*: rank $r=64$, $\alpha=128$, dropout 0.05, 4-bit NF4 quantization, targeting all attention and MLP projections (q, k, v, o, gate, up, down). Batch size 2, gradient accumulation 16, lr 2×10^{-5} , cosine schedule, gradient checkpointing, max sequence length 4096, 1–3 epochs. *Full fine-tune*: batch size 4, gradient accumulation 4 (effective batch 16), lr 2×10^{-4} , cosine schedule, warmup ratio 0.05, bfloat16, 3 epochs, on H100.

5.4 Results

5.4.1 Teacher Trajectory Quality

Model	Prompt	Game24	AIME Avg	GPQA Main	GPQA Ext	GPQA Dia
GPT-4.1	default	98.0	55.2	64.96	60.81	62.63
GPT-4.1	teacher	95.0	–	65.40	62.82	–
Gemini-2.5-Pro	default	–	–	21.43	–	–

Table 1: Teacher model accuracy (%) across benchmarks under DC-Cumulative. AIME Avg is the mean accuracy across 41 years (1983–2023), ranging from 30% (2021–2022) to 82% (1991). Gemini GPQA is partial (96 of 448 problems completed at time of writing).

5.4.2 Diagnostic: Small Model Performance

Model	Condition	Game24	AIME 25	GPQA Dia
<i>Paper baselines [1]</i>				
GPT-4o	Baseline	10.0	6.7	56.6
GPT-4o	DC-Cu	93.0	16.7	49.0
<i>Our teacher (GPT-4.1)</i>				
GPT-4.1	DC-Cu	98.0	36.7	62.6
<i>Small models</i>				
Qwen-7B	Baseline	10.0	3.3	32.1
Qwen-7B	DC-Cu	4.0	0.0	–
Qwen-14B	Baseline	13.0	0.0	–
DeepSeek-R1-7B	Baseline	8.0	10.0	–
<i>Small models — with cheatsheet summarization (1000 tok, GPT-4.1)</i>				
Qwen-7B	DC-Cu + Summarize	7.0	5.0	28.1

Table 2: Diagnostic results. DC-Cu *hurts* Qwen-7B (10%→4%); cheatsheet summarization isn’t able to help much here.

5.4.3 Effect of Cheatsheet Summarization

Qwen-7B under DC-Cu *degrades* from its 10% baseline to 4% (Table 2), replicating the original DC paper’s finding. The cheatsheet grows to $\sim 8\text{K}$ tokens, overwhelming the model’s effective context window. Adding summarization to the *base* model (compressed to 1000 tokens by GPT-4.1 after each curator step) yields only 7%—barely above the unsummarized 4%. This reveals that summarization alone is insufficient: the base model writes cheatsheet content so poor that even a high-quality summary contains little useful signal.

However, when applied to the *SFT-trained* model, summarization boosts accuracy from 28% to **55%**—a $2\times$ improvement. SFT teaches the model to write genuinely useful strategies (e.g., “factor 24 as $a \times b$ ”), and summarization keeps these strategies within the model’s effective context window. This demonstrates **two compounding bottlenecks**: content quality (addressed by SFT) and memory length (addressed by summarization).

5.4.4 Temperature Effect on SFT Models

An unexpected finding is that SFT-trained small models perform *substantially better* at temperature 1.0 than at the standard evaluation temperature of 0.0.

Model	Temp	Game24 Acc	No-Answer Rate
DeepSeek-R1-Qwen-7B-SFT	0.0	20.0%	70.0%
DeepSeek-R1-Qwen-7B-SFT	1.0	41.0%	$\sim 30\%$
Qwen-7B-SFT (QLoRA) + Summ	0.0	47%	43%
Qwen-7B-SFT (QLoRA) + Summ	1.0	55%	35%

Table 3: Effect of sampling temperature on SFT-trained models under DC-Cumulative on Game of 24. Temperature 1.0 more than doubles accuracy compared to greedy decoding.

We attribute this to two factors. First, **mode collapse**: SFT sharpens the output distribution toward the teacher’s reasoning *style* without perfectly learning the answer *format*. At temperature 0 the greedy path consistently fails to emit `<answer>` tags (70% no-answer rate); sampling at temperature 1.0 breaks out of this pattern. Second, **stochastic search**: Game of 24 requires finding one valid expression among many combinations. Greedy decoding commits to a single path, whereas sampling explores diverse solutions—effectively performing combinatorial search through the model’s distribution.

This motivates GRPO training, which explicitly optimizes the sampling distribution by rewarding correct diverse completions, and suggests that SFT-trained models should be evaluated with best-of- N sampling rather than greedy decoding.

5.4.5 SFT, GRPO Results

We train Qwen-7B on task-specific teacher trajectories using QLoRA (local, 4-bit) and full-parameter SFT (H100).

The leave-one-out (LOO) model, trained on Game24 + GPQA only, achieves 30.8% on Game24 with summarization—lower than the in-domain-trained model’s 55%, indicating that cross-task transfer of DC skills is limited and task-specific trajectory data matters. This might indicate we need further experiments here to improve accuracy.

Training setup. We use two training configurations:

Local QLoRA (rapid iteration): LoRA [7] with rank $r=64$, $\alpha=128$, applied to all seven projection matrices per layer, training $\sim 1\%$ of parameters. 4-bit NF4 quantization [3] reduces VRAM from $\sim 14\text{ GB}$ to $\sim 4\text{ GB}$. Runs on NVIDIA RTX A5880 (48 GB).

Cloud QLoRA SFT (final models): All parameters trained on NVIDIA H100 (80 GB) via Ray Train, with 4-bit quantization, rank 64, LoRA alpha: 128, LoRA dropout: 0.05. Checkpoints streamed to WandB monitoring.

Model	Condition	Game24	AIME 24/25	GPQA
<i>Base model</i>				
Qwen-7B	Baseline	10.0	3.3	32.1
Qwen-7B	DC-Cu	4.0	0.0	26.6
Qwen-7B	DC-Cu + Summ	7.0	5.0	28.1
<i>After SFT on AIME trajectories (QLoRA)</i>				
Qwen-7B-SFT-aime [13]	DC-Cu	13.35	12.2	-
<i>After SFT on game24 trajectories (QLoRA)</i>				
Qwen-7B-SFT-game24 [14]	DC-Cu	28.0	-	-
Qwen-7B-SFT-game24	DC-Cu + Summ	55.0	10.0	-
<i>After LOO (QLoRA, trained on game24 + aime only)</i>				
Qwen-7B-LOO-SFT [15]	DC-Cu + Summ	30.8	13.3	31.3
<i>After GRPO refinement</i>				
Qwen-7B-LOO-SFT-GRPO (*) [16]	DC-Cu + Summ	8.1	10	32.5

Table 4: SFT and GRPO ablation results. SFT-trained models are evaluated under DC-Cu with summarization to isolate the effect of trajectory distillation from memory management.

Both use the SFT chat format with `completion_only_loss` to mask prompt tokens, ensuring the model only learns from assistant responses.

Reward Functions. GRPO training combines reward signals additively. By default, two rewards are used:

1. **Correctness reward:** The model prediction is extracted primarily from `<answer>...</answer>` tags (with a fallback to `FINAL ANSWER: ...`) and scored by task-specific evaluators: exact match (punctuation-insensitive) for AIME, arithmetic-expression validity for Game of 24, and multiple-choice option matching for GPQA.
2. **Format reward:** +0.2 for including well-formed `<answer>...</answer>` tags, +0.1 for having extractable answer content, and -0.2 if the completion exceeds 8,000 characters.

When cheatsheet context is present, two additional citation-related rewards are enabled and also added to the total reward.

6 Analysis

Two compounding bottlenecks. Table 4 reveals that neither SFT nor summarization alone bridges the gap to frontier models. SFT without summarization raises DC-Cu from 4% to 28%—a $7\times$ improvement—demonstrating that SFT teaches the model to write useful cheatsheet content. But 28% is far below the teacher’s 98%, because the cheatsheet still grows unbounded and overwhelms the context window. Conversely, summarization without SFT yields only 7%: the base model writes content so poor that even a high-quality summary is useless. Only **SFT + summarization** together achieve 55%, confirming two compounding bottlenecks: (a) *content quality* (what the model writes), addressed by SFT, and (b) *memory length* (how much it writes), addressed by summarization. The remaining 55%→98% gap reflects the model’s raw reasoning ability—the target of GRPO.

Why summarization requires SFT. Small models suffer disproportionately from long, noisy contexts [2], but shortening a bad cheatsheet does not make it good. The base model’s cheatsheet entries are verbose, problem-specific, and often incoherent; summarizing them merely produces a compact version of noise. SFT teaches the model to write generalizable strategies (e.g., “try factoring 24 as $a \times b$ ”), giving the summarizer high-value raw material to condense. Summarization then acts as a *memory length controller*, keeping the useful content within the model’s effective attention window.

Teacher prompt style comparison. GPT-4.1 with the “teacher” prompt (which includes explicit instruction to generate pedagogical reasoning) achieves slightly higher accuracy on GPQA (65.4% vs 65.0% main; 62.8% vs 60.8% extended) but slightly lower on Game of 24 (95% vs 98%). The

teacher prompt produces more verbose but more explanatory cheatsheet entries, which may be more suitable for distillation despite marginally lower raw accuracy.

Mitigating Context Bloat and Gradient Instability. A primary bottleneck in applying DC-Cumulative to smaller models is context bloating during training as well, essentially what the teacher models generate cannot translate 1 to 1 as finetuning data for smaller models. It caused rapid performance degradation and numerical instability. During our initial phase, we observed that uncompressed trajectories frequently exceeded the 7B model’s effective context limit. This manifested in emergence of NaNs during evaluation loss computation, signaling potential gradient collapse as the model struggled with extreme sequence lengths during evaluation. To resolve this while maintaining our heavy reliance on the generated teacher data, we adopted quantization via QLoRA.

Shortcomings. Several limitations affect our current results. *Teacher dependency:* cheatsheet summarization relies on GPT-4.1 at inference time, adding cost and latency; self-summarization (using the student itself) is needed for a self-contained system. *Evaluation scale:* Game of 24 uses only 100 problems and AIME 2024–2025 has only 60; variance is high and results may shift with larger test sets. *Single student architecture:* all experiments use Qwen-2.5 variants; generalization to other model families (Llama, Phi) is untested. *Incomplete GRPO:* RL training is not yet complete, so we cannot confirm whether GRPO closes the reasoning gap as hypothesized. Needs more compute and time for further experiments.

7 Conclusion

We demonstrate that the primary bottleneck preventing small language models from benefiting from Dynamic Cheatsheets is **memory management**, not raw reasoning capability. Cheatsheet summarization alone transforms Qwen-7B from a 4% DC-Cu accuracy (worse than its 10% baseline) to 55% on Game of 24. We have built a complete pipeline for teacher trajectory generation, SFT distillation, and GRPO refinement, with GPT-4.1 achieving strong teacher accuracy across AIME (55%), GPQA (65%), and Game of 24 (98%).

Ongoing and future work.

- *Self-summarization* We plan to investigate self-summarization (using the student model itself as summarizer instead of GPT-4.1) to remove the teacher dependency at inference time, making the approach fully self-contained.
- *System prompts* We plan to evaluate the scalability of system role anchoring during GRPO, specifically testing if the clear separation of logic (system) and task (user) facilitates more stable gradient updates and more precise reward-shaping for citation-aware memory retrieval.
- *GRPO training* (in progress) targets the reasoning gap by rewarding correct answers, proper formatting, and citation grounding across diverse sampled completions.
- *Best-of-N decoding* at temperature 1.0, motivated by our temperature findings, could further boost accuracy without additional training.
- Finally, *multi-task SFT* on the combined AIME+GPQA+Game24 trajectory corpus may improve cross-task DC skill transfer, which our LOO results show is currently limited.

Team contributions

Sakthivel Sivaraman: Evaluation and teacher trajectory generation pipeline, DC-Cu accuracy improvements, cheatsheet summarization experiments, ablation experiments, SFT and GRPO local implementation and testing, citation-grounding trajectories and rewards.

Prabhjot Singh Rai: Cloud training infrastructure (Ray Train, GCS checkpointing), cheatsheet summarization support, evaluation pipeline, full-parameter SFT on H100, GRPO implementation, model serving, SFT and GRPO experiments.

References

- [1] Mirac Suzgun, Mert Yuksekgonul, Federico Bianchi, Dan Jurafsky, and James Zou. Dynamic cheatsheet: Test-time learning with adaptive memory. *arXiv preprint arXiv:2504.07952*, 2025.
- [2] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024.
- [3] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. QLoRA: Efficient finetuning of quantized language models. *Advances in Neural Information Processing Systems*, 36, 2024.
- [4] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Mingchuan Zhang, Y.K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.
- [5] Kumar Shridhar, Alessandro Stolfo, and Mrinmaya Sachan. Distilling reasoning capabilities into smaller language models. In *Findings of the Association for Computational Linguistics: ACL 2023*, 2023.
- [6] Arindam Mitra, Luciano Del Corro, Shweti Mahajan, Andres Codas, et al. Orca 2: Teaching small language models how to reason. *arXiv preprint arXiv:2311.11045*, 2023.
- [7] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. *International Conference on Learning Representations*, 2022.
- [8] Google DeepMind. Gemini 2.5 pro. <https://deepmind.google/technologies/gemini/>, 2025.
- [9] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. *arXiv preprint arXiv:2305.10601*, 2023.
- [10] yentinglin. AIME 2025 dataset. https://huggingface.co/datasets/yentinglin/aime_2025, 2025.
- [11] David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. *arXiv preprint arXiv:2311.12022*, 2024.
- [12] Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2025.
- [13] Prabhjot Singh Rai and Sakthivel Sivaraman. Qwen2.5-7B-Instruct-SFT-AIME. <https://huggingface.co/tinyllms/qwen2.5-7b-instruct-sft-aime>, 2026.
- [14] Prabhjot Singh Rai and Sakthivel Sivaraman. Qwen2.5-7B-Instruct-SFT-Game24-QLoRA-16384. <https://huggingface.co/tinyllms/qwen2.5-7b-instruct-sft-game24-qlora-16384>, 2026.
- [15] Prabhjot Singh Rai and Sakthivel Sivaraman. Qwen2.5-7B-Instruct-SFT-LOO-Domain-Knowledge. <https://huggingface.co/tinyllms/qwen2.5-7b-instruct-sft-loo-domain-knowledge>, 2026.
- [16] Prabhjot Singh Rai and Sakthivel Sivaraman. Qwen2.5-7B-Instruct-GRPO-LOO-Domain-Knowledge. <https://huggingface.co/tinyllms/qwen2.5-7b-instruct-grpo-loo-domain-knowledge>, 2026.

A Appendix: Detailed AIME Results

Year	Acc	C/T	WA/NA	Year	Acc	C/T	WA/NA
1983	60.0	9/15	5/1	2004	51.7	15/29	6/8
1984	78.6	11/14	1/2	2005	48.3	14/29	2/13
1985	60.0	9/15	2/4	2006	67.9	19/28	6/3
1986	64.3	9/14	2/3	2007	67.9	19/28	3/6
1987	57.1	8/14	2/4	2008	63.0	17/27	4/6
1988	80.0	8/10	0/2	2009	60.0	18/30	6/6
1989	66.7	6/9	2/1	2010	67.9	19/28	3/6
1990	75.0	9/12	1/2	2011	55.2	16/29	10/3
1991	81.8	9/11	2/0	2012	50.0	13/26	8/5
1992	60.0	9/15	5/1	2013	55.2	16/29	6/7
1993	50.0	7/14	1/6	2014	46.7	14/30	8/8
1994	33.3	3/9	3/3	2015	50.0	15/30	4/11
1995	35.7	5/14	2/7	2016	46.7	14/30	6/10
1996	71.4	10/14	1/3	2017	43.3	13/30	12/5
1997	57.1	8/14	3/3	2018	43.3	13/30	10/7
1998	46.7	7/15	3/5	2019	46.7	14/30	8/8
1999	53.3	8/15	5/2	2020	43.3	13/30	10/7
2000	66.7	18/27	3/6	2021	30.0	9/30	8/13
2001	60.0	18/30	8/4	2022	30.0	9/30	7/14
2002	46.4	13/28	10/5	2023	41.4	12/29	6/11
2003	64.3	18/28	5/5				

Table 5: GPT-4.1 DC-Cumulative accuracy on all 41 AIME years. C/T = Correct/Total, WA/NA = Wrong Answer / No Answer failures.