ARM: Adaptive Reasoning Model

Anonymous Author(s)
Affiliation
Address
email

Abstract

While large reasoning models demonstrate strong performance on complex tasks, they lack the ability to adjust reasoning token usage based on task difficulty. This often leads to the "overthinking" problem—excessive and unnecessary reasoning—which, although potentially mitigated by human intervention to control the token budget, still fundamentally contradicts the goal of achieving fully autonomous AI. In this work, we propose ***Adaptive Reasoning Model*** (ARM), a reasoning model capable of adaptively selecting appropriate reasoning formats based on the task at hand. These formats include three efficient ones—*Direct Answer*, *Short CoT*, and *Code*—as well as a more elaborate format, *Long CoT*. To train ARM, we introduce ***Ada-GRPO***, an adaptation of Group Relative Policy Optimization (GRPO), which addresses the format collapse issue in traditional GRPO. Ada-GRPO enables ARM to achieve high token efficiency, reducing tokens by an average of $\sim 30\%$, and up to $\sim 70\%$, while maintaining performance comparable to the model that relies solely on *Long CoT*. All the resources will be released.

1 Introduction

The emergence of large reasoning models (LRMs) such as OpenAI-o1 [16] and DeepSeek-R1 [9] has advanced problem-solving via test-time scaling [3; 50], where Long Chain-of-Thought (Long CoT) boosts performance by generating more tokens. Yet, trained mainly on reasoning-heavy tasks, LRMs often apply Long CoT indiscriminately, causing "overthinking" [4; 37]—excessive token use with little gain and potential noise [46; 7]. Prior approaches reduce tokens through budget estimation [1; 41] or length-constrained training [13], but these rely heavily on human knowledge of the tasks. A more autonomous solution is adaptive token control, where the model itself employs concise reasoning for easy tasks and deliberate reasoning for hard ones.

In this work, we propose ***Adaptive Reasoning Model*** (ARM), a reasoning model capable of adaptively selecting reasoning formats based on task difficulty, balancing both performance and computational efficiency. ARM supports four reasoning formats: three efficient ones—*Direct Answer*, *Short CoT*, and *Code*—and one elaborate format, *Long CoT*. To train ARM, we adopt a two-stage training framework. In Stage 1, we apply supervised fine-tuning (SFT) to equip the language model with a foundational understanding of four reasoning formats. In Stage 2, we introduce Ada-GRPO, an adaptation of Group Relative Policy Optimization (GRPO) [33], which encourages efficient format selection while preserving accuracy as the primary objective. Ada-GRPO is designed to address two key issues: *1)* The uniform distribution of reasoning formats regardless of task difficulty observed during the SFT stage; *2)* The format collapse problem in GRPO, where *Long CoT* gradually dominates as training progresses, leading to the diminished use of other, more efficient formats. Extensive evaluations show that ARM trained with Ada-GRPO achieves comparable performance while using $\sim 30\%$ fewer tokens than GRPO, across both in-domain and out-of-domain tasks in commonsense, mathematical, and symbolic reasoning.

Submitted to NeurIPS 2025 Workshop on Efficient Reasoning. Do not distribute.

2 Method

We propose Adaptive Reasoning Model (ARM), a reasoning model designed to optimize effectiveness and efficiency by adaptively selecting reasoning formats. Specifically, ARM is trained in two stages: *1)* **Stage 1: Supervised Fine-tuning (SFT) for Reasoning Formats Understanding**: In this stage, we use 10.8K diverse questions, each annotated with solutions in four distinct reasoning formats, to fine-tune the model and build a foundational understanding of different reasoning strategies. *2)* **Stage 2: Reinforcement Learning (RL) for Encouraging Efficient Format Selection**: We adopt an adapted version of the GRPO algorithm, named Ada-GRPO, to train the model to be capable of selecting more efficient reasoning formats over solely *Long CoT*, while maintaining accuracy.

2.1 Stage 1: SFT for Reasoning Formats Understanding

In this stage, we leverage SFT as a cold start to introduce the model to various reasoning formats it can utilize to solve problems. These formats include three efficient reasoning formats *Direct Answer*, *Short CoT*, and *Code*, as well as the elaborate reasoning format *Long CoT*. We use special tokens (e.g., <*Code*></*Code*>) to embrace thinking rationale. Specifically, *1)* ***Direct Answer*:** This format provides a direct answer without any reasoning chain, making it the most efficient in terms of token usage. *2)* ***Short CoT*:** This format begins with a short reasoning and then provides an answer, which has been proved effective in mathematical problems [43]. *3)* ***Code*:** This format adopts code-based reasoning, which has proven effective across a variety of tasks due to its structured process [44; 45; 20]. *4)* ***Long CoT*:** This format involves a more detailed, iterative reasoning process, thus incurs higher token usage. It is suited for tasks requiring advanced reasoning capabilities, such as self-reflection and alternative generation, where those more efficient formats fall short [27; 9; 49].

2.2 Stage 2: RL for Encouraging Efficient Format Selection

After SFT, the model learns to respond using various reasoning formats but lacks the ability to adaptively switch between them based on the task (see Section 3.2 for details). To address this, we propose **Ada**ptive GRPO (**Ada-GRPO**), which enables the model to dynamically select appropriate reasoning formats according to the task difficulty through a format diversity reward mechanism.

GRPO In traditional GRPO [33], the model samples a group of outputs $O = \{o_1, o_2, \cdots, o_G\}$ for each question q, where G denotes the group size. For each o_i, a binary reward r_i is computed using a rule-based reward function that checks whether the prediction $pred$ matches the ground truth gt:

$$r_i = \mathbb{1}_{\texttt{passed}(gt, pred)}. \tag{1}$$

However, since traditional GRPO solely optimizes for accuracy, it leads, in our setting, to overuse of the highest-accuracy format while discouraging exploration of alternative reasoning formats. Specifically, if *Long CoT* achieves higher accuracy than other formats, models trained with GRPO tend to increasingly reinforce it, leading to an over-reliance on *Long CoT* and reduced exploration of more efficient alternatives. We refer to this phenomenon as **Format Collapse**, which ultimately hinders the model's ability to develop adaptiveness. We further analyze this in Section 3.2.

Ada-GRPO We introduce Ada-GRPO to mitigate format collapse by amplifying rewards for underrepresented reasoning formats, ensuring their persistence during training. Formally, we rescale the reward r_i as

$$r'_i = \alpha_i(t)\, r_i, \quad \alpha_i(t) = \frac{G}{F(o_i)} \cdot decay_i(t), \quad decay_i(t) = \tfrac{F(o_i)}{G} + \tfrac{1}{2}\Big(1 - \tfrac{F(o_i)}{G}\Big)\big(1 + \cos\big(\pi \tfrac{t}{T}\big)\big)\,, \tag{2}$$

where $F(o_i)$ denotes the number of times the reasoning format corresponding to o_i appears within its group O, and t represents the training step. $\alpha_i(t)$ is a format diversity scaling factor that gradually decreases from $\frac{G}{F(o_i)}$ at the beginning of training ($t = 0$) to 1 at the end of training ($t = T$).

We introduce $\alpha_i(t)$ to extend GRPO into **Ada-GRPO**, enabling models to adaptively select reasoning formats. Specifically, $\alpha_i(t)$ consists of two components: *1)* **Format Diversity Scaling Factor** $\frac{G}{F(o_i)}$: To prevent premature convergence on the highest-accuracy format (i.e., format collapse to *Long CoT*), we upweight rewards for less frequent formats to encourage exploration. *2)* **Decay Factor** $decay_i(t)$: To avoid long-term misalignment caused by over-rewarding rare formats, this term gradually reduces the influence of diversity over time. For example, $\frac{G}{F(o_i)}$ might make the model favor a lower-accuracy

Table 1: Performance of various models across evaluation datasets. "#Tokens" refers to the token cost for each model on each dataset. For each model, $k = 1$ corresponds to pass@1, and $k = 8$ corresponds to maj@8. When $k = 8$, the token cost is averaged over a single output to facilitate clear comparison. "†" denotes in-domain tasks, while "‡" denotes out-of-domain tasks. "Δ" represents the difference between ARM and Qwen2.5$_{\text{SFT+GRPO}}$, calculated by subtracting the accuracy of Qwen2.5$_{\text{SFT+GRPO}}$ from that of ARM, with the token usage expressed as the ratio of tokens saved by ARM compared to Qwen2.5$_{\text{SFT+GRPO}}$, with all settings based on $k = 8$ to ensure a stable comparison. We also report results for ARM-3B and ARM-14B in Appendix Table 3.

Models		**Accuracy (↑)**								**#Tokens (↓)**							
		Easy		Medium				Hard	Avg.	Easy		Medium				Hard	Avg.
	k	CSQA†	OBQA‡	GSM8K†	MATH†	SVAMP‡	BBH‡	AIME'25‡		CSQA†	OBQA‡	GSM8K†	MATH†	SVAMP‡	BBH‡	AIME'25‡	
GPT-4o	1	85.9	94.2	95.9	75.9	91.3	84.7	10.0	76.8	192	165	287	663	156	278	984	389
o1-preview	1	85.5	95.6	94.2	92.6	92.7	91.8	40.0	84.6	573	492	456	1863	489	940	7919	1819
o4-mini-high	1	84.7	96.0	96.9	97.7	94.0	92.2	96.7	94.0	502	289	339	1332	301	755	9850	1910
DeepSeek-V3	1	82.4	96.0	96.5	91.8	93.7	85.8	36.7	83.3	231	213	236	887	160	400	2992	732
DeepSeek-R1	1	83.3	94.8	96.4	97.1	96.0	85.0	70.0	88.9	918	736	664	2339	589	1030	9609	2270
DS-R1-Distill-1.5B	1	47.6	48.6	79.4	84.6	86.7	53.5	20.0	60.1	987	1540	841	3875	606	3005	13118	3425
DS-R1-Distill-7B	1	64.9	77.4	90.0	93.6	90.3	72.1	40.0	75.5	792	928	574	3093	315	1448	12427	2797
DS-R1-Distill-14B	1	80.6	93.2	94.0	95.5	92.7	80.4	50.0	83.8	816	750	825	2682	726	1292	11004	2585
DS-R1-Distill-32B	1	83.2	94.6	93.5	93.0	92.0	86.3	56.7	85.6	674	698	438	2161	283	999	11276	2361
Qwen2.5-7B	1	76.7	78.6	81.6	50.1	81.0	51.7	3.3	60.4	64	83	156	376	99	182	767	247
	8	82.0	86.4	89.9	64.7	89.7	62.0	3.3	68.3	66	74	156	370	92	183	881	260
Qwen2.5-7B$_{\text{SFT}}$	1	80.8	81.2	54.4	30.4	76.0	48.2	0	53.0	136	150	184	348	126	245	1239	347
	8	83.9	84.6	79.4	42.4	88.0	56.0	0	62.0	141	137	185	361	141	274	1023	323
Qwen2.5-7B$_{\text{SFT+GRPO}}$	1	83.1	82.2	92.8	79.4	93.7	64.3	16.7	73.2	491	651	739	1410	587	1133	3196	1173
	8	83.7	84.6	94.8	84.9	95.3	69.3	20.0	76.1	496	625	745	1415	586	1135	3145	1164
ARM-7B	1	86.1	84.4	89.2	73.9	92.0	61.4	16.7	72.0	136	159	305	889	218	401	3253	766
	8	85.7	85.8	93.7	82.6	95.3	67.9	20.0	75.9	134	154	297	893	218	413	3392	786
Δ		+2.0	+1.2	-1.1	-2.3	0	-1.4	0	-0.2	−73.0%	−75.4%	−60.1%	−36.9%	−62.8%	−63.6%	+7.9%	−32.5%

format like *Short CoT* over *Long CoT* simply because it appears less frequently and thus receives a higher reward. While such exploration is beneficial early in training, it can hinder convergence later. The decay mechanism mitigates this by promoting diversity initially, then shifting focus to accuracy again as training progresses. We adopt the traditional advantage formula in GRPO. Please refer to Appendix C for more details.

3 Experiment

3.1 Experimental Setup

Training To assess the effectiveness of our method across models of different sizes, we select Qwen2.5-Base as the backbone model. We use AQuA-Rat [22] as the SFT dataset, as its answers can be naturally transformed into four distinct reasoning formats. In addition to the *Direct Answer* and *Short CoT* rationales provided with the dataset, we utilize GPT-4o [26] and DeepSeek-R1 [9] to supplement the *Code* and *Long CoT* rationales, resulting in a training set containing 3.0K multiple-choice and 7.8K open-form questions. Appendix D provides further details on the generation and filtering process. In Stage 2, to prevent data leakage, we employ three additional datasets exclusively for the RL stage. These datasets cover a range of difficulty levels, from relatively simple commonsense reasoning tasks to more complex mathematical reasoning tasks, including CommonsenseQA (CSQA) [39], GSM8K [6], and MATH [12], collectively comprising 19.8K verifiable question-answer pairs. Please refer to Appendix E for dataset details, Appendix F for implementation details, and Appendix G for inference.

Baselines In addition to backbone models, we compare ARM with models trained using alternative algorithms that may enable adaptive reasoning capabilities. Specifically, **Qwen2.5$_{\text{SFT}}$** refers to the backbone model trained on the AQuA-Rat dataset used in Stage 1. In this setting, we explore whether language models can master adaptive reasoning through a straightforward SFT strategy. For **Qwen2.5$_{\text{SFT+GRPO}}$**, we examine whether SFT models, further trained with GRPO, can better understand different reasoning formats and whether this approach empowers them to select appropriate reasoning formats based on rule-based rewards.

Evaluation Datasets To evaluate reasoning, we use in-domain and out-of-domain datasets spanning commonsense, mathematical, and symbolic tasks. CommonsenseQA (CSQA) [39] and OpenBookQA (OBQA) [25] represent intuitive commonsense tasks. Mathematical reasoning is assessed with SVAMP [30], GSM8K [6], MATH [12], and the competition-level AIME'25 [8]. For symbolic reasoning, we adopt Big-Bench-Hard (BBH) [38]. We categorize datasets into three levels: *easy* (commonsense), *medium* (math + symbolic), and *hard* (AIME'25).

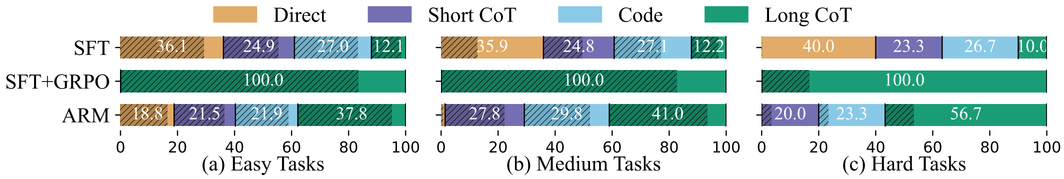

Figure 1: Format distribution by task difficulty with Qwen2.5-7B. The hatched areas indicate the percentage of correct answers that were generated using the selected reasoning format.

3.2 Main Results

Alongside our baselines, we include several state-of-the-art general models, including GPT-4o [26] and DeepSeek-V3 [23], as well as reasoning models o1-preview [27], o4-mini-high [28], and DeepSeek-R1 [9], along with several DeepSeek-R1-Distill-Qwen (DS-R1-Distill) models ranging from 1.5B to 32B [9]. We report our results in Table 1, and we have the following findings:

SFT teaches models formats but not how to choose among them. We find that SFT improves performance on easy commonsense tasks but hurts medium and hard ones. To analyze why, we examine the distribution of reasoning formats at inference. Figure 1 shows that SFT models allocate formats nearly uniformly, with most outputs in *Direct Answer* and few in *Long CoT*, regardless of difficulty. This overuse of *Direct Answer*—which performs poorly on medium tasks (35.2% accuracy)—undermines reasoning ability and overall performance. Thus, SFT teaches formats but fails to promote adaptive selection as task complexity increases.

GRPO does improve reasoning capabilities, but it tends to rely on *Long CoT* to solve all tasks. We observe that models trained with GRPO achieve significant improvements across all tasks, yet the token cost remains substantial, especially for the two easier tasks. Further analysis reveals that *Long CoT* is predominantly used in the inference stage, as shown in Figure 1. This behavior stems from the nature of GRPO (i.e., format collapse discussed in Section 2.2), where models converge to the format with the highest accuracy (i.e., *Long CoT*) early in training ($\sim$10 steps in our experiment). As a result, GRPO also fails to teach models how to select a more efficient reasoning format based on the task.

ARM is able to adaptively select reasoning formats based on task difficulty, while achieving comparable accuracy across all tasks compared to GRPO and using significantly fewer tokens. As shown in Table 1, ARM experiences an average performance drop of less than 1% compared to the model trained with GRPO, yet it saves more than 30% of the tokens. Specifically, ARM demonstrates a clear advantage on easy tasks, saving over 70% of tokens while maintaining comparable accuracy. This advantage extends to medium tasks as well. For the more challenging AIME'25 task, ARM adapts to the task difficulty by increasingly selecting *Long CoT*, thereby avoiding performance degradation on harder tasks. Figure 1 further confirms that ARM is able to gradually adopt more advanced reasoning formats and discards simpler ones as task difficulty increases.

3.3 Further Analysis

We provide more features of ARM and further analysis in Appendix B, including *1)* ARM also supports Instruction-Guided Mode, effective when specified formats are suitable for the tasks at hand, and Consensus-Guided Mode, which maximizes performance at higher token cost; *2)* Ada-GRPO yields a $\sim 2\times$ training speedup over traditional GRPO; and *3)* Ada-GRPO demonstrates robustness across both instruction-tuned and reasoning backbones.

4 Conclusion

In this work, we propose ***Adaptive Reasoning Model*** (ARM), which adaptively selects reasoning formats based on task difficulty. ARM is trained with Ada-GRPO, a GRPO variant that addresses format collapse via a format diversity reward and achieves a $\sim 2\times$ training speedup. Experiments show that ARM maintains performance comparable to the GRPO-trained model relying solely on *Long CoT*, while significantly improving token efficiency. By adopting the adaptive reasoning format selection strategy, ARM effectively mitigates the overthinking problem and offers a novel, efficient approach to reducing unnecessary reasoning overhead.

References

[1] Pranjal Aggarwal and Sean Welleck. L1: Controlling how long a reasoning model thinks with reinforcement learning. *arXiv preprint arXiv:2503.04697*, 2025.

[2] Daman Arora and Andrea Zanette. Training language models to reason efficiently. *arXiv preprint arXiv:2502.04463*, 2025.

[3] Qiguang Chen, Libo Qin, Jinhao Liu, Dengyun Peng, Jiannan Guan, Peng Wang, Mengkang Hu, Yuhang Zhou, Te Gao, and Wangxiang Che. Towards reasoning era: A survey of long chain-of-thought for reasoning large language models. *arXiv preprint arXiv:2503.09567*, 2025.

[4] Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, et al. Do not think that much for 2+ 3=? on the overthinking of o1-like llms. *arXiv preprint arXiv:2412.21187*, 2024.

[5] Jeffrey Cheng and Benjamin Van Durme. Compressed chain of thought: Efficient reasoning through dense representations. *arXiv preprint arXiv:2412.13171*, 2024.

[6] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, et al. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

[7] Chenrui Fan, Ming Li, Lichao Sun, and Tianyi Zhou. Missing premise exacerbates overthinking: Are reasoning models losing critical thinking skill? *arXiv preprint arXiv:2504.06514*, 2025.

[8] Google. Aime problems and solutions, 2025. URL https://artofproblemsolving.com/wiki/index.php/AIME_Problems_and_Solutions.

[9] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[10] Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. Token-budget-aware llm reasoning. *arXiv preprint arXiv:2412.18547*, 2024.

[11] Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. Training large language models to reason in a continuous latent space. *arXiv preprint arXiv:2412.06769*, 2024.

[12] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. *NeurIPS*, 2021.

[13] Bairu Hou, Yang Zhang, Jiabao Ji, Yujian Liu, Kaizhi Qian, Jacob Andreas, and Shiyu Chang. Thinkprune: Pruning long chain-of-thought of llms via reinforcement learning. *arXiv preprint arXiv:2504.01296*, 2025.

[14] 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. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

[15] Jingcheng Hu, Yinmin Zhang, Qi Han, Daxin Jiang, Xiangyu Zhang, and Heung-Yeung Shum. Open-reasoner-zero: An open source approach to scaling up reinforcement learning on the base model. *arXiv preprint arXiv:2503.24290*, 2025.

[16] Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

[17] Bowen Jin, Hansi Zeng, Zhenrui Yue, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*, 2025.

[18] Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. T\" ulu 3: Pushing frontiers in open language model post-training. *arXiv preprint arXiv:2411.15124*, 2024.

[19] Ayeong Lee, Ethan Che, and Tianyi Peng. How well do llms compress their own chain-of-thought? a token complexity approach. *arXiv preprint arXiv:2503.01141*, 2025.

[20] Junlong Li, Daya Guo, Dejian Yang, Runxin Xu, Yu Wu, and Junxian He. Codei/o: Condensing reasoning patterns via code input-output prediction. *arXiv preprint arXiv:2502.07316*, 2025.

[21] Zhong-Zhi Li, Duzhen Zhang, Ming-Liang Zhang, Jiaxin Zhang, Zengyan Liu, Yuxuan Yao, Haotian Xu, Junhao Zheng, Pei-Jie Wang, Xiuyi Chen, et al. From system 1 to system 2: A survey of reasoning large language models. *arXiv preprint arXiv:2502.17419*, 2025.

[22] Wang Ling, Dani Yogatama, Chris Dyer, and Phil Blunsom. Program induction by rationale generation: Learning to solve and explain algebraic word problems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 158–167, 2017.

[23] Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*, 2024.

[24] Yan Ma, Steffi Chern, Xuyang Shen, Yiran Zhong, and Pengfei Liu. Rethinking rl scaling for vision language models: A transparent, from-scratch framework and comprehensive evaluation scheme. *arXiv preprint arXiv:2504.02587*, 2025.

[25] Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2381–2391, 2018.

[26] OpenAI. Hello GPT-4o, 2024. URL `https://openai.com/index/hello-gpt-4o/`.

[27] OpenAI. Learning to reason with llms, 2024. URL `https://openai.com/index/learning-to-reason-with-llms`.

[28] OpenAI. Introducing openai o3 and o4-mini, 2025. URL `https://openai.com/index/introducing-o3-and-o4-mini/`.

[29] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[30] Arkil Patel, Satwik Bhattamishra, and Navin Goyal. Are nlp models really able to solve simple math word problems? In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2080–2094, 2021.

[31] Xiao Pu, Michael Saxon, Wenyue Hua, and William Yang Wang. Thoughtterminator: Benchmarking, calibrating, and mitigating overthinking in reasoning models. *arXiv preprint arXiv:2504.13367*, 2025.

[32] Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 3505–3506, 2020.

[33] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Y Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[34] Zhenyi Shen, Hanqi Yan, Linhai Zhang, Zhanghao Hu, Yali Du, and Yulan He. Codi: Compressing chain-of-thought into continuous space via self-distillation. *arXiv preprint arXiv:2502.21074*, 2025.

[35] Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv:2409.19256*, 2024.

[36] DiJia Su, Sainbayar Sukhbaatar, Michael Rabbat, Yuandong Tian, and Qinqing Zheng. Dualformer: Controllable fast and slow thinking by learning with randomized reasoning traces. In *The Thirteenth International Conference on Learning Representations*, 2024.

[37] Yang Sui, Yu-Neng Chuang, Guanchu Wang, Jiamu Zhang, Tianyi Zhang, Jiayi Yuan, Hongyi Liu, Andrew Wen, Hanjie Chen, Xia Hu, et al. Stop overthinking: A survey on efficient reasoning for large language models. *arXiv preprint arXiv:2503.16419*, 2025.

[38] Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, et al. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*, 2022.

[39] Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. 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)*, pages 4149–4158, 2019.

[40] Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, et al. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*, 2025.

[41] Qwen Team. Qwen3 technical report, 2025. URL https://github.com/QwenLM/Qwen3/blob/main/Qwen3_Technical_Report.pdf.

[42] Luong Trung, Xinbo Zhang, Zhanming Jie, Peng Sun, Xiaoran Jin, and Hang Li. Reft: Reasoning with reinforced fine-tuning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7601–7614, 2024.

[43] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022.

[44] Nathaniel Weir, Muhammad Khalifa, Linlu Qiu, Orion Weller, and Peter Clark. Learning to reason via program generation, emulation, and search. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=te6VagJf6G.

[45] Jiaxin Wen, Jian Guan, Hongning Wang, Wei Wu, and Minlie Huang. Codeplan: Unlocking reasoning potential in large language models by scaling code-form planning. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=dCPF1wlqj8.

[46] Siye Wu, Jian Xie, Jiangjie Chen, Tinghui Zhu, Kai Zhang, and Yanghua Xiao. How easily do irrelevant inputs skew the responses of large language models? In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=S7NVVfuRv8.

[47] Fengli Xu, Qianyue Hao, Zefang Zong, Jingwei Wang, Yunke Zhang, Jingyi Wang, Xiaochong Lan, Jiahui Gong, Tianjian Ouyang, Fanjin Meng, et al. Towards large reasoning models: A survey of reinforced reasoning with large language models. *arXiv preprint arXiv:2501.09686*, 2025.

[48] Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. Chain of draft: Thinking faster by writing less. *arXiv preprint arXiv:2502.18600*, 2025.

[49] Weihao Zeng, Yuzhen Huang, Qian Liu, Wei Liu, Keqing He, Zejun Ma, and Junxian He. Simplerl-zoo: Investigating and taming zero reinforcement learning for open base models in the wild. *arXiv preprint arXiv:2503.18892*, 2025.

[50] Qiyuan Zhang, Fuyuan Lyu, Zexu Sun, Lei Wang, Weixu Zhang, Zhihan Guo, Yufei Wang, Irwin King, Xue Liu, and Chen Ma. What, how, where, and how well? a survey on test-time scaling in large language models. *arXiv preprint arXiv:2503.24235*, 2025.

[51] Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, and Zheyan Luo. Llamafactory: Unified efficient fine-tuning of 100+ language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 3: System Demonstrations)*, pages 400–410, 2024.

Appendix

A Related Work

A.1 Reinforcement Learning for Improving Reasoning

Reinforcement Learning (RL) has demonstrated significant potential in enhancing the problem-solving abilities of large language models (LLMs) across various domains [29; 42; 17]. Recently, Reinforcement Learning with Verifiable Rewards (RLVR) has gained substantial attention for advancing LLM capabilities [18; 9; 21], resulting in the development of large reasoning models (LRMs) [47] such as OpenAI-o1 [16] and DeepSeek-R1 [9]. Based on simple rule-based rewards, RLVR algorithms such as Group Relative Policy Optimization (GRPO) [33] enable models to use Long Chain-of-Thought (Long CoT) [49; 15]. This facilitates deep reasoning behaviors, such as searching, backtracking, and verifying through test-time scaling [3; 50]. However, these models also suffer from significant computational overhead due to extended outputs across all tasks, leading to inefficiency associated with the "overthinking" phenomenon [4; 31; 37]. Verbose and redundant outputs can obscure logical clarity and hinder the model's ability to solve problems effectively [46; 7].

A.2 Efficiency in Large Language Models

Recently, many studies have focused on improving the reasoning efficiency in LLMs. Some prompt-guided methods [10; 48; 19] explicitly instruct LLMs to generate concise reasoning outputs by controlling input properties such as task difficulty and response length. Other approaches [11; 5; 34] explore training LLMs to reason in latent space, generating the *direct answer* without the need for detailed language tokens. Several techniques have also been proposed to reduce inference costs by controlling or pruning output length, either by injecting multiple reasoning formats during the pre-training stage [36] or by applying length penalties during the RL stage [40; 2; 1; 13]. Many of these methods aim to strike a trade-off between token budget and reasoning performance by shortening output lengths, often relying on clear estimations of the token budget for each task or requiring specialized, length-constrained model training. However, in reality, such estimations are not always accurate, and what we ultimately expect is for models to adaptively regulate their token usage based on the complexity of the task at hand. Therefore, in this work, we propose a novel training framework that enables models to adaptively select suitable reasoning formats for given tasks by themselves, optimizing both performance and computational efficiency.

B Analysis

B.1 Reasoning Mode Switching

Table 2: Accuracy (Acc.) and token usage (Tok.) for the three reasoning modes supported by ARM-7B. In the Consensus-Guided Mode, the percentage of *Long CoT* usage indicates how often the model resorts to *Long CoT* when simpler reasoning formats fail to reach a consensus.

ARM-7B	**Easy**				**Medium**								**Hard**		**Avg.**	
	CSQA†		OBQA‡		GSM8K†		MATH†		SVAMP‡		BBH‡		AIME'25‡			
	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.	Acc.	Tok.
Adaptive	86.1	136	84.4	159	89.2	305	73.9	889	92.0	218	61.4	401	16.7	3253	72.0	766
Inst$_{Direct}$	84.1	10	81.8	10	22.9	11	23.1	13	67.0	11	44.7	21	0	12	46.2	13
Inst$_{Short\ CoT}$	81.3	33	77.4	35	85.0	124	70.9	633	86.7	66	49.7	101	10.0	2010	65.9	428
Inst$_{Code}$	84.4	140	81.6	147	84.2	285	65.9	559	88.3	182	57.9	344	10.0	1821	67.5	497
Inst$_{Long\ CoT}$	84.0	259	87.4	294	91.8	426	77.2	1220	94.3	340	66.9	660	20.0	4130	74.5	1047
Consensus	85.8	228	87.0	260	92.9	777	78.4	2281	95.7	433	66.4	1039	20.0	7973	75.2	1856
Long CoT Usage	12.9%		21.4%		79.8%		79.2%		36.3%		56.3%		100%		55.1%	

ARM is capable of autonomously selecting appropriate reasoning formats (Adaptive Mode), while also supporting explicit guidance to reason in specified formats (Instruction-Guided Mode) or through consensus between different reasoning formats (Consensus-Guided Mode). Specifically, *1)* **Adaptive**

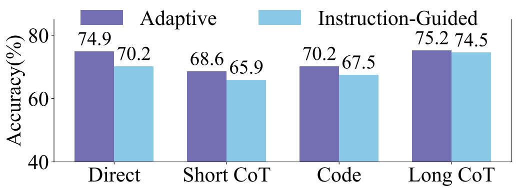

Figure 2: Accuracy comparison between ARM's **Adaptive** and **Instruction-Guided** modes. The figure shows average accuracy across evaluation datasets, with *Direct Answer* applied only to commonsense and symbolic tasks, as it does not appear in mathematical tasks in Adaptive mode.

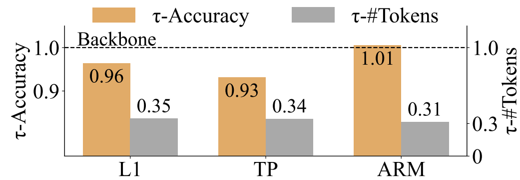

Figure 3: Relative accuracy and token usage of different models compared to their backbone models on CSQA. "L1" denotes L1-Exact [1], and "TP" denotes THINKPRUNE [13]. "τ-Accuracy" and "τ-#Tokens" are reported relative to each model's backbone after RL training.

Mode: In this mode, ARM autonomously selects the reasoning format for each task, which is also the default reasoning mode if not specified in this paper. *2)* **Instruction-Guided Mode:** In this mode, a specific token (e.g., <*Long CoT*>) is provided as the first input, forcing ARM to reason in the specified format. *3)* **Consensus-Guided Mode:** In this mode, ARM first generates answers using the three simpler reasoning formats (i.e., *Direct Answer*, *Short CoT*, and *Code*) and checks for consensus among them. If all formats agree, the consensus answer is adopted as the final result. Otherwise, ARM defaults to *Long CoT* for the final answer, treating the task as sufficiently complex.

To evaluate the performance and effectiveness of the proposed reasoning modes, we conduct experiments across various evaluation datasets. Table 2 presents the results for ARM-7B. Specifically: *1)* **Adaptive Mode strikes a superior balance between high accuracy and efficient token usage across all datasets, demonstrating its ability to adaptively select the reasoning formats.** *2)* **Instruction-Guided Mode offers a clear advantage when the assigned reasoning format is appropriate.** For example, *Direct Answer* is sufficient for commonsense tasks, while *Code*, due to its structured nature, performs better on symbolic reasoning tasks compared to *Direct Answer* and *Short CoT*. Furthermore, $\text{Inst}_{\textit{Long CoT}}$ achieves better performance (74.5%) than the same-sized model trained on GRPO (73.2% in Table 1). This demonstrates that Ada-GRPO does not hinder the model's *Long CoT* reasoning capabilities. We further validate this by analyzing the reflective words used by ARM-7B and Qwen2.5-$7\text{B}_{\text{SFT+GRPO}}$ in Appendix H. *3)* **Consensus-Guided Mode, on the other hand, is performance-oriented, requiring more tokens to achieve better performance.** This mode leverages consensus across multiple formats to mitigate bias and uncertainty present in any single format, offering greater reliability, particularly for reasoning tasks that demand advanced cognitive capabilities, where simpler formats may fall short. This is evidenced by the fact that *Long CoT* is less likely to be used for easy tasks, but is highly likely to be selected for medium tasks and even used 100% of the time for the most difficult AIME'25 task.

B.2 Effectiveness of Adaptive Format Selection

To verify that ARM's format selection indeed adapts to the task at hand rather than relying on random selection, we compare ARM's **Adaptive Mode** with **Instruction-Guided Mode**. In Instruction-Guided Mode, the reasoning format is fixed and manually specified, providing a strong baseline to test whether adaptive selection offers real benefits over using a uniform format across tasks. We report the accuracy of both modes in Figure 2. We observe that the accuracy of the reasoning formats selected in Adaptive Mode is higher than that in Instruction-Guided Mode. Specifically, Adaptive Mode improves accuracy by 4.7% on *Direct Answer*, by 2.7% on both *Short CoT* and *Code*, and even yields a slight improvement on *Long CoT*. These results confirm that ARM is not randomly switching formats but is instead learning to select an appropriate one for each task.

B.3 Comparison of Ada-GRPO and GRPO

We find that, compared to GRPO, ARM trained with Ada-GRPO achieves comparable performance on the evaluation dataset while achieving approximately a $\sim 2\times$ speedup in training time. To understand the source of this efficiency, we compare the training dynamics of Ada-GRPO and GRPO across different model sizes, focusing on accuracy, response length, and training time, as shown in Figure 4.

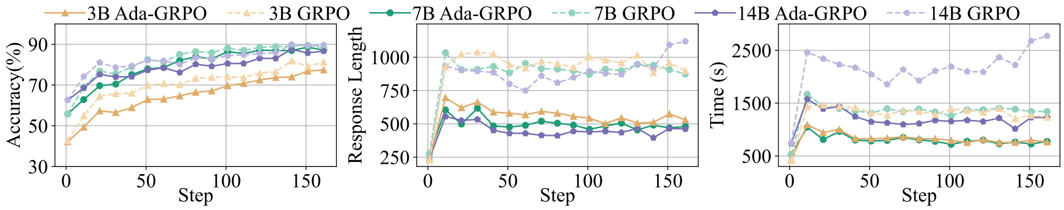

Figure 4: Performance on the training set across different model sizes trained with Ada-GRPO and GRPO. Except for the implementation of the algorithm, all hyperparameters are kept the same.

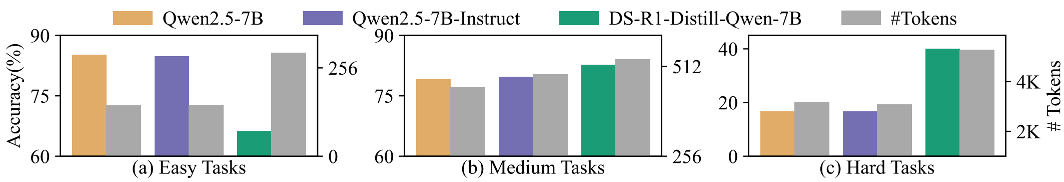

Figure 5: ARMs' performance across different backbones. Base and instruction-tuned models perform similarly, while DS-R1-Distill improves on medium and hard tasks but struggles on easy ones.

The results highlight the following advantages of Ada-GRPO: *1)* **Comparable Accuracy.** Although Ada-GRPO initially lags behind GRPO in accuracy due to suboptimal reasoning format selection in the early training steps, both methods converge to similar final accuracy across all model sizes. This demonstrates that Ada-GRPO does not compromise final performance. *2)* **Half Response Length.** While GRPO uses *Long CoT* uniformly across all tasks, Ada-GRPO adaptively selects reasoning formats based on task difficulty. Due to the length efficiency of *Direct Answer*, *Short CoT*, and *Code*, Ada-GRPO ultimately reduces the average response length to roughly half that of GRPO. *3)* **Half Training Time Cost.** Since the majority of training time is spent on response generation during the roll-out stage, reducing response length directly translates into lower time cost. As a result, Ada-GRPO achieves approximately a $\sim 2\times$ speedup compared to GRPO. Overall, Ada-GRPO maintains strong performance while significantly reducing computational overhead, underscoring its efficiency and reliability for training.

B.4 Comparison of Backbone Models

Beyond the base model, we further analyze the impact of different backbone models, including instruction-tuned and DS-R1-Distill variants. Figure 5 reports accuracy and token usage across *easy*, *medium*, and *hard* tasks. We observe that base and instruction-tuned models have a highly similar performance. This suggests that RL effectively bridges the gap left by instruction tuning, enabling base models to achieve comparable performance, consistent with findings from previous work [17]. In contrast, the DS-R1-Distill variant performs notably better on medium and hard tasks, benefiting from distilled knowledge from the stronger DeepSeek-R1 model, though at the expense of increased token cost. However, it performs significantly worse on easy tasks, even with excessive token usage, resulting from the overthinking phenomenon. For more discussion and case studies on the overthinking phenomenon, please refer to Appendix I.

B.5 Comparison of ARM and Length-Penalty-Based Strategies

To examine whether previously proposed length-penalty-based strategies—proven effective in complex reasoning—remain effective for easier tasks, we evaluate two representative methods, L1 [1] and THINKPRUNE[13], on the CSQA dataset. Since both methods are based on the DS-R1-Distill model, we ensure a fair comparison by also evaluating the version of ARM trained on the same backbone. We report the relative accuracy and token usage of all three models compared to their respective backbone models in Figure 3. When using the minimum allowed lengths specified in the official settings of L1 and THINKPRUNE, both methods exhibit performance drops. In contrast, ARM maintains strong performance while using relatively fewer tokens, demonstrating its ability to balance reasoning efficiency and effectiveness. Please refer to Appendix J for more details.

Table 3: Performance of ARM-3B and ARM-14B across evaluation datasets.

Models	k	Accuracy (↑) Easy CSQA†	Easy OBQA‡	Medium GSM8K†	Medium MATH†	Medium SVAMP‡	Medium BBH‡	Hard AIME'25‡	Avg.	#Tokens (↓) Easy CSQA†	Easy OBQA‡	Medium GSM8K†	Medium MATH†	Medium SVAMP‡	Medium BBH‡	Hard AIME'25‡	Avg.
Qwen2.5-3B	1	66.5	65.8	66.9	37.7	71.3	38.4	0	49.5	97	120	150	419	76	232	1393	355
	8	75.5	77.4	80.9	50.8	83.7	47.1	0	59.3	96	100	149	424	85	240	1544	377
Qwen2.5-3B$_{\text{SFT}}$	1	72.8	72.4	35.7	20.9	62.3	37.4	0	43.1	99	108	145	229	126	311	694	245
	8	75.5	77.4	56.0	27.6	74.7	43.5	0	50.7	97	103	132	231	108	309	537	217
Qwen2.5-3B$_{\text{SFT+GRPO}}$	1	79.7	79.0	88.7	66.6	92.0	52.6	6.7	66.5	425	501	788	1586	630	994	3027	1136
	8	80.3	80.0	91.4	74.0	94.7	56.2	6.7	69.0	429	506	802	1590	638	996	3247	1172
ARM-3B	1	79.8	78.0	83.8	62.9	89.7	50.0	6.7	64.4	118	156	346	1013	264	436	2958	756
	8	80.1	78.0	90.8	72.8	95.0	53.8	6.7	68.2	123	169	359	1036	246	430	3083	778
Δ		-0.2	-2.0	-0.6	-1.2	+0.3	-2.4	0	-0.8	-71.3%	-66.6%	-55.2%	-34.8%	-61.4%	-56.8%	-5.1%	-33.6%
Qwen2.5-14B	1	79.9	83.8	84.9	52.7	84.7	56.8	3.3	63.7	56	60	132	335	77	139	611	201
	8	83.8	90.2	92.3	68.4	91.7	67.4	3.3	71.0	55	60	131	325	81	131	735	217
Qwen2.5-14B$_{\text{SFT}}$	1	81.8	88.0	62.6	37.4	84.0	53.5	0	58.2	155	140	161	276	152	254	527	238
	8	85.0	91.4	86.4	48.8	91.7	64.4	3.3	67.3	149	141	165	288	140	247	493	232
Qwen2.5-14B$_{\text{SFT+GRPO}}$	1	85.4	93.0	94.8	81.7	93.7	70.5	20.0	77.0	558	531	693	1805	565	945	4031	1304
	8	85.8	94.2	96.1	87.1	95.3	77.0	20.0	79.4	552	537	696	1810	565	943	3723	1261
ARM-14B	1	85.3	91.8	92.5	79.1	93.3	66.6	20.0	75.5	146	128	294	903	212	420	3871	853
	8	85.6	91.8	96.3	86.4	95.7	72.1	23.3	78.7	145	134	293	910	189	415	3996	869
Δ		-0.2	-2.4	+0.2	-0.7	+0.4	-4.9	+3.3	-0.7	−73.7%	−75.0%	−57.9%	−49.7%	−66.5%	−56.0%	+7.3%	−31.1%

C Details of Ada-GRPO

C.1 Training Objective

Following GRPO [33], the group advantage $\hat{A}_{i,k}$ for all tokens in each output is computed based on the group of reshaped rewards $\mathbf{r}' = \{r'_1, r'_2, \cdots, r'_G\}$:

$$\hat{A}_{i,k} = \frac{r'_i - \text{mean}(\{r'_1, r'_2, \cdots, r'_G\})}{\text{std}(\{r'_1, r'_2, \cdots, r'_G\})}. \tag{3}$$

We optimize the policy model π using the Ada-GRPO objective:

$$\begin{aligned}\mathcal{J}_{\text{Ada-GRPO}}(\theta) =& \mathbb{E}\left[q \sim P(Q), \{o_i\}_{i=1}^{G} \sim \pi_{\theta_{\text{old}}}(O|q)\right] \left[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{|o_i|}\sum_{k=1}^{|o_i|}\left\{\min\left[\frac{\pi_\theta(o_{i,k}|q, o_{i,<k})}{\pi_{\theta_{\text{old}}}(o_{i,k}|q, o_{i,<k})}\hat{A}_{i,k},\right.\right.\right.\\ &\left.\left.\left.\text{clip}\left(\frac{\pi_\theta(o_{i,k}|q, o_{i,<k})}{\pi_{\theta_{\text{old}}}(o_{i,k}|q, o_{i,<k})}, 1-\epsilon, 1+\epsilon\right)\hat{A}_{i,k}\right] - \beta\,\text{KL}\left[\pi_\theta \parallel \pi_{\text{ref}}\right]\right\}\right],\end{aligned} \tag{4}$$

where π_{ref} denotes the reference model, and the KL divergence term KL serves as a constraint to prevent the updated policy from deviating excessively from the reference. The advantage estimate $\hat{A}_{i,k}$ is computed based on a group of rewards $\{r'_1, r'_2, \cdots, r'_G\}$ associated with the responses in O, as defined in equation 3.

C.2 Decay Factor

In Ada-GRPO, the decay factor $decay_i(t)$ is introduced to regulate the influence of the format diversity scaling factor during training. Without decay, the model may continue to overly reward less frequent reasoning formats even after sufficient exploration, misaligning with our objective. To evaluate the effectiveness of the decay mechanism, we track the test set performance across three in-domain datasets (CSQA, GSM8K, and MATH) using checkpoints saved every 25 training steps for models trained with and without decay. As shown in Figure 6, models trained without decay exhibit larger performance fluctuations in test accuracy, indicating unstable exploration. In contrast, the decay mechanism stabilizes training, resulting in smoother and more consistent improvements in accuracy during the middle and later training stages.

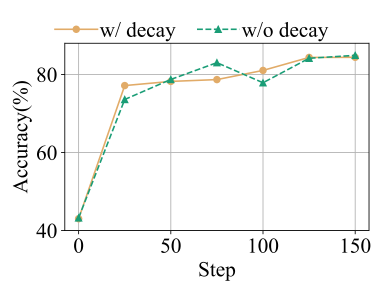

Figure 6: Test set accuracy with and without the decay mechanism.

Table 4: Dataset in each training stage.

Dataset	Answer Format	Size
Stage 1: Supervised Finetuning		
AQuA-Rat	Multiple-Choice	3.0K
	Open-Form	7.8K
		10.8K
Stage 2: Reinforcement Learning		
CSQA	Multiple-Choice	4.9K
GSM8K	Open-Form	7.4K
MATH	Open-Form	7.5K
		19.8K

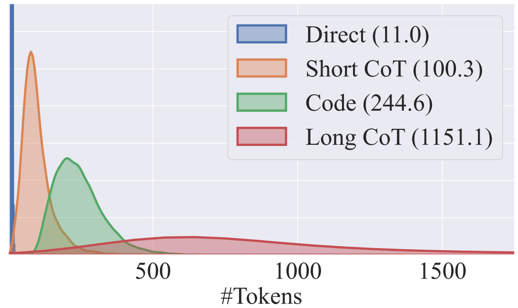

Figure 7: Token count distribution across reasoning formats in the SFT dataset AQuA-Rat, with brackets indicating average counts.

D Details of Processing SFT Dataset

D.1 Prompt List

We use gpt-4o-2024-11-20 to generate *Code* reasoning rationales. Following previous work [44], we ask the model to return the output as a dictionary containing all intermediate and final outputs, which is beneficial for emulating the generated program's execution.

```
For the following questions and answers, generate a function that
solves the question. The function should return a dictionary with the
field 'answer': <answer>, as well as the values for intermediate
decisions. Ensure that both the function and its call are wrapped in <
CODE>...</CODE>, and that the emulation of its execution is wrapped in
 <OUTPUT>...</OUTPUT>.

{examples}

Question:
{question}
Answer:
{rational}
#### {ground_truth}
```

We use Deepseek-R1 to generate *Long CoT* rationales.

```
Put the answer in format "<ANSWER>...</ANSWER>".

{question}
```

D.2 Filter Out Rationales

For *Code* rationales, we utilize a Python interpreter to execute each generated code snippet. We apply the following filters: *1)* execution failure, *2)* missing *answer* key, *3)* inconsistencies between intermediate steps and execution results, and *4)* mismatches between the predicted and ground-truth answers. For *Long CoT* rationales, we filter out those with incorrect answers. Token count distribution across reasoning formats in the SFT dataset can be seen in Figure 7.

E Examples in Training

The question-answer format and size of the datasets in each training stage can be found in Table 4.

E.1 Examples in SFT Stage

We provide four reasoning formats for the SFT dataset AQuA-Rat [22], including *Direct Answer*, *Short CoT*, *Code*, *Long CoT*. Here is an example.

```
Question:
You collect baseball cards. Suppose you start out with 15. Maria takes
 half of one more than the number of baseball cards you have. Since
you're nice, you give Peter 1 baseball card. Since his father makes
baseball cards, Paul decides to triple your baseball cards. How many
baseball cards do you have at the end?

Direct Answer:
18

Code:
def calculate_baseball_cards():
    # Initial number of baseball cards
    initial_cards = 15

    # Maria takes half of one more than the number of baseball cards
    maria_takes = (initial_cards + 1) / 2
    cards_after_maria = initial_cards - maria_takes

    # Peter takes 1 baseball card
    cards_after_peter = cards_after_maria - 1

    # Paul triples the number of baseball cards
    final_cards = cards_after_peter * 3

    return {
        'maria_takes': maria_takes,
        'cards_after_maria': cards_after_maria,
        'cards_after_peter': cards_after_peter,
        'answer': final_cards
    }

>>> calculate_baseball_cards()

output = {
    'maria_takes': 8.0,
    'cards_after_maria': 7.0,
    'cards_after_peter': 6.0,
    'answer': 18.0
}

Short CoT:
Start with 15 baseball cards. Maria takes half of one more than the
number of baseball cards you have. So Maria takes half of 15 + 1 which
 is 8, so you're left with 15 - 8 = 7. Peter takes 1 baseball card
from you: 7 - 1 = 6 baseball cards. Paul triples the number of
baseball cards you have: 6 * 3 = 18 baseball cards. So you have 18 at
the end. The answer is 18.

Long CoT:
Okay, let me try to work through this problem step by step. So, the
question is about collecting baseball cards, and there are a series of
 steps that happen. Let me read it again to make sure I understand
each part.

Alright, starting with 15 cards. Then Maria takes half of one more
than the number I have. Wait, that phrasing is a bit tricky. So Maria
takes half of (one more than my current number of cards). Let me parse
 that. If I have N cards, Maria takes (N +1)/2? Hmm. Let me double-
check. The problem says: "Maria takes half of one more than the number
```

```
 of baseball cards you have." So, the expression would be ( (number
you have + 1 ) ) / 2. Right. So that's (current cards +1) divided by
two. Then after that, I give Peter 1 card. Then Paul triples my cards.
 So, let's break it down step by step.

Starting with 15 cards. Let's note each step:

1. Initial cards: 15.

2. Maria takes half of one more than what I have. So, first, calculate
 one more than my current count: 15 +1 =16. Then she takes half of
that. So 16 /2 =8. So Maria takes 8 cards. Wait, but when she takes
them, does that mean my total decreases by 8? Yes, I think so. Because
 she's taking them from me. So subtract 8 from my current count.

So after Maria takes them: 15 -8 =7. Now I have 7 cards left.

Wait a minute, wait. Wait, when Maria takes half of one more than the
number I have, which is (15+1)/2=8. So she's taking 8 cards. So yes,
subtract 8 from my current 15. That leaves me with 7.

Then, the next step: since I'm nice, I give Peter 1 baseball card. So
subtract 1 more. 7 -1 =6.

Then Paul decides to triple my baseball cards. So multiply my current
count by 3. 6 *3 =18.

So the answer should be 18. Let me go through the steps again to check
.

Original: 15.

After Maria: (15 +1)/2 =8. So 15-8=7.

After giving Peter 1: 7-1=6.

After Paul triples: 6*3=18.

Yes, that seems right. Did I misinterpret any step? Let's double-check
 each part.

"Maria takes half of one more than the number of baseball cards you
have." So when Maria takes, it's (my current cards +1)/2. So when I
have 15, that's (15+1)/2=8. She takes 8, so I have 7 left. Correct.

Then giving Peter 1: 7-1=6. Then Paul triples to 18. Yep. That seems
correct. So the final answer is 18.
```

E.2 Examples in RL Stage

The RL dataset includes CSQA [39], GSM8K [6], and MATH [12], with examples provided below.

```
<CommonsenseQA>
What is the base of a vase called?
(A)transistor  (B)vessel  (C)box  (D)pie  (E)seabase
ground truth: B

<GSM8K>
Natalia sold clips to 48 of her friends in April, and then she sold
half as many clips in May. How many clips did Natalia sell altogether
in April and May?
ground truth: 72

<MATH>
```

Table 5: Definitions and results of reflection-related ratios on AIME'25.

Ratio Name	Formula	Qwen2.5-7B$_{\text{SFT+GRPO}}$	ARM-7B
reflection_ratio	$\frac{\mathcal{N}_{ref}}{\mathcal{N}}$	93.8	95.0
correct_ratio_in_reflection_texts	$\frac{\mathcal{N}_{ref+}}{\mathcal{N}_{ref}}$	14.2	13.9

```
Rationalize the denominator: $\frac{1}{\sqrt{2}-1}$. Express your
answer in simplest form.
ground truth: $\boxed{\sqrt{2}+1}$
```

F Implementation Details

Our training is performed using 8 NVIDIA A800 GPUs. The following settings are also applied to other baselines for fair comparisons.

F.1 Stage 1: SFT

We utilize the open-source training framework LLAMAFACTORY [51] to perform SFT. The training is conducted with a batch size of 128 and a learning rate of 2e-4. We adopt a cosine learning rate scheduler with a 10% warm-up period over 6 epochs. To enhance training efficiency, we employ parameter-efficient training via Low-rank adaptation (LoRA) [14] and DeepSpeed training with the ZeRO-3 optimization stage [32]. As a validation set, we sample 10% of the training data and keep the checkpoint with the lowest perplexity on the validation set for testing and the second stage.

F.2 Stage 2: RL

We utilize the open-source training framework VeRL [35] to perform RL. During training, we use a batch size of 1024 and generate 8 rollouts per prompt ($G = 8$), with a maximum rollout length of 4096 tokens. The model is trained with a mini-batch size of 180, a KL loss coefficient of 1e-3, and a total of 9 training epochs. The default sampling temperature is set to 1.0.

G Inference

During inference, we set the temperature to 0.7 and top-p to 1.0. For all evaluation datasets, we use accuracy as the metric. In addition to pass@1, to reduce bias and uncertainty associated with single generation outputs and to enhance the robustness of the results [50], we further use majority@k (maj@k), which measures the correctness of the majority vote from k independently sampled outputs. For inference on the three backbone models, we use an example with a short-cot-based answer within the prompt to guide the model toward specific answer formats while preserving its original reasoning capabilities as much as possible.

H Details of Reflective Words

To evaluate models' *Long CoT* reasoning capabilities, we focus on their use of specific *reflective words* that signal backtracking and verifying during the reasoning process. Following prior work [24], we consider a curated list of 17 reflective words: ["*re-check*", "*re-evaluate*", "*re-examine*", "*re-think*", "*recheck*", "*reevaluate*", "*reexamine*", "*reevaluation*", "*rethink*", "*check again*", "*think again*", "*try again*", "*verify*", "*wait*", "*yet*", "*double-check*", "*double check*"]. We adopt two evaluation metrics: `reflection_ratio`, measuring the proportion of outputs containing at least one reflective word, and `correct_ratio_in_reflection_texts`, assessing the correctness within reflective outputs. The formulas for these metrics are summarized in Table 5, where $\mathcal{N}$ denotes the total number of responses, $\mathcal{N}_{ref}$ the number of responses containing reflective words, and $\mathcal{N}_{ref+}$ the number of correct reflective responses.

Given its competition-level difficulty, we conduct our analysis on AIME'25 using ARM-7B and Qwen2.5-7B$_{\text{SFT+GRPO}}$. For ARM-7B, we use the Instruction-Guided Mode (Inst$_{\textit{Long CoT}}$) to specifically assess its *Long CoT* reasoning. The results, averaged over 8 runs, are reported in Table 5. As shown, both models exhibit a high frequency of reflective word usage, with `reflection_ratio` exceeding 93%, indicating that reflection behavior is well-integrated during *Long CoT* reasoning. The `correct_ratio_in_reflection_texts` remains comparable for both models, and relatively low due to the high complexity of the AIME'25 tasks. These results demonstrate that Ada-GRPO does not hinder the model's *Long CoT* reasoning capabilities.

I Details of the Overthinking Phenomenon

Overthinking refers to the phenomenon where LLMs apply unnecessarily complex reasoning to simple tasks, leading to diminishing returns in performance [37]. As demonstrated in Table 1 and 2, using *Long CoT*, despite incurring higher computation costs, significantly enhances model performance on tasks requiring complex mathematical reasoning, such as MATH. However, as mentioned in Section 3.2 and B.4, longer responses do not consistently lead to better performance for all task types. In this section, we analyze the overthinking phenomenon in depth, focusing on how overly complex reasoning formats can hurt performance when applied to certain tasks.

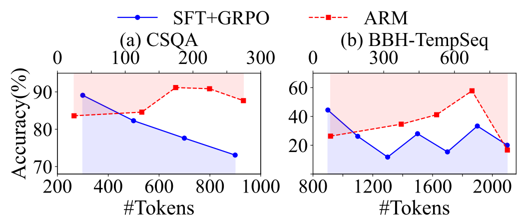

Figure 8: Overthinking in 7B model performance across two representative datasets. We remove the extreme data points and ensure that sufficient data points fall within the intervals.

I.1 Analysis

We analyze the evaluation datasets and illustrate the "overthinking" phenomenon using two representative datasets: CSQA and TemporalSequences in BBH Benchmark. From Figure 8 (note that the horizontal axis scales differ between *SFT+GRPO* and ARM.), we observe that *SFT+GRPO*, which relies heavily on *Long CoT*, shows a significant drop in accuracy as the model generates lengthy responses. It indicates that the model starts generating excessive information that does not contribute to task resolution. In contrast, ARM achieves better performance with fewer tokens for simpler tasks. This analysis reinforces the idea that longer reasoning does not equate to better performance [7; 37], and the optimal reasoning format depends on task type. ARM provides a more flexible, task-specific solution to avoid overthinking, ultimately improving performance. Please refer to Appendix I.3 for a detailed example of how *Long CoT* errs in CSQA, while a simpler reasoning format with lower token usage successfully solves the problem.

I.2 Data Examples

```
<BBH-Temporal Sequences>
Today, Ashley went to the bookstore. Between what times could they
have gone?
We know that: Ashley woke up at 5am. Jason saw Ashley buying lunch at
the deli from 11am to 1pm. Emily saw Ashley driving to the water park
from 1pm to 3pm. Kimberly saw Ashley stretching at a yoga studio from
3pm to 6pm.
The bookstore was closed after 6pm. Between what times could Ashley
have gone to the bookstore?
(A) 1pm to 3pm
(B) 11am to 1pm
(C) 3pm to 6pm
(D) 5am to 11am
ground truth: D
```

I.3 An Example of Overthinking in CSQA

```
Where does a maid empty a trash can?
(A)bus stop  (B)corner  (C)hockey game  (D)motel  (E)alley

Long CoT used by +SFT+GRPO:
<LONG_COT>
Okay, let's see. The question is where a maid would empty a trash can.
 Hmm, pick one of the options given. Let's break down the terms.

First, a maid is someone who works in a hotel or other environment,
such as cleaning. Their job might involve domestic duties for a
specific kind of client, perhaps upper class, but specific job would
involve handling various items. So having cleaning duties like trash
removal.

Now, traditionally, maid service might be focused on particular areas-
probably the rooms in a hotel, apartment, or maybe an individual
client if they work as an in-home maid. So for that_RESULTS, would
probably be... Alright. Let's look at the choices:

(A) Bus stop. Seems like they shouldn't dispose of anything there. A
bus stop would have trash cans, but personally I doubt a maid would
dump trash in a bus station. That's probably not part of their
responsibilities.

(B) Corner. Hmm. A literal street corner, perhaps maybe a public
restroom? Or other outdoor spot? But where tomorrow's trash can? I can
't imagine a location in caring for trash outside normal public areas.
 Maybe mega cities and sanitation handling upgrades, but again not a
typical everyday task for a maid.

(C) Hockey Game. Hmm. Trash removal at a hockey game... Never heard of
 a maid working at a hockey game. Probably no. The grounds might be
cleaned by different personnel, like stadium employees, most likely
more specialized. Hell, playerName duplication, maybe the trash cans
outside the arena get cleaned? That probably exclude maid's work issue
. So probably can be discounted.

(D) Motel. Hmm. Maybe the version of this? Or hotels. Wait, motel is
final answer. But wait, official work responsibility. The main point
here is where professional maid service has responsibilities. For
example, in hotels-motel rooms (though maybe not modern motels, which
had more community facilities?) The specific task is they probably
need to take trash cans to the disposal area-usually outside the room
in compilations. So perhaps dump it into a proper receptacle. That
would happen in backend trash can disposal, but Solutions associated
with cleaning, not directly in guest areas.

(E) Alley. Hmm. Alley avoids touristy trash? Wait, perhaps like
backstage scenario. Or alleyways can be garbage collection areas. But
how typical? Probably the communities are serviced by regular
collection with professionals. For same reasons as airliner (like
trash in the alley). The main thing is that probably would move trash
to the nearest dumpster, possibly in an alley. Though not a public
area of the house. Though again, problem is which alley. It's not
nearly as common would be this responsibility. Probably more likely
that a maid would take the trash to a dumpster outside. So option E
probably can be considered.

Wait, not sure, but none of the options are obvious, but Zeeland E.
Because the equipment is usually moved from rooms to a centralized
dumpster. Probably near the alley. Although may not be the primary
course of action. However, this is what might be discussed. Because
depending on the structure. For example, perhaps motel rooms need to
dispose garbage in dumpster, making alley collection possible.
```

```
But options like motel and motel would probably be the right answer.
But museum? Not typical. So perhaps neither of the answers given. Wait
, maybe E.
</LONG_COT>
<ANSWER>
E
</ANSWER>

Short CoT used by ARM:
<SHORT_COT>
A maid works in a motel. A maid may empty a trash can in a motel.
The answer is D
</SHORT_COT>
<ANSWER>
D
</ANSWER>
```

J Details of Length-Penalty-Based Strategies

To ensure fair comparisons, we follow the official settings of L1 [1] and THINKPRUNE [13], adopting their specified minimum allowed lengths when evaluating on easy tasks. We set the temperature to 0.6 and top-p to 0.95, consistent with both papers. Specifically, we use `L1-Qwen-1.5B-Exact`[1] at 512 tokens for L1 and `DeepSeek-R1-Distill-Qwen-1.5B-thinkprune-iter2k`[2] for THINKPRUNE.

K Limitations

Dependency on Predefined Reasoning Formats In this work, we focus on four commonly used reasoning formats that generalize well across a wide range of reasoning tasks. However, we acknowledge that certain tasks may benefit from more specialized or nuanced reasoning strategies beyond this predefined set. Our reliance on predefined formats is primarily due to the limited capabilities of current models, which may struggle to autonomously identify or switch between diverse reasoning formats, let alone new reasoning formats. As a result, we define the formats in advance and introduce them through SFT to help the model establish a clear understanding of each reasoning type. We believe that as model capabilities continue to improve, future work can explore enabling models to autonomously select or even invent new reasoning formats without relying on predefined structures.

Lack of Hard Task Data in Training Unlike some length-penalty-based strategies, our training setup does not include hard datasets such as prior AIME tasks, which may place our model at a disadvantage on hard tasks compared to methods like L1 [1] and THINKPRUNE [13] that incorporate such data. Nevertheless, ARM still shows clear improvements on base models and maintains stable performance on R1-distilled models on AIME 2025, demonstrating its potential on hard tasks. We expect that incorporating harder data into training would further enhance performance. However, due to the high computational cost of reinforcement learning—and the current version of ARM being an early exploration aimed at evaluating generalization across tasks while improving token cost efficiency—we leave this extension to future work.

[1] https://huggingface.co/l3lab/L1-Qwen-1.5B-Exact

[2] https://huggingface.co/Shiyu-Lab/DeepSeek-R1-Distill-Qwen-1.5B-thinkprune-iter2k

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Our main contributions are detailed in Section 1. See Section 3 and Section B for more experimental evidence and analysis.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We discuss the limitations of the work in Appendix K.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

Justification: The paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: We explain our settings as well as the hyperparameters in Section 3, Appendix F, and Appendix J for all our experiments.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [Yes]

Justification: We will release our data and code to facilitate reproduction and future research.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyper-parameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We explain our settings as well as the hyperparameters. Details are summarized in Section 3, Appendix F, and Appendix J for all our experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We use some strategies such as "majority@k" to reduce bias and uncertainty to enhance the robustness of the results.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).

- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Details are provided in Section B.3 and Appendix F.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We followed the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

    Justification: There is no societal impact of the work performed.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The research involves publicly available datasets and standard models, posing no significant misuse risks, thus no specific safeguards were necessary.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We credit the owners of all code, models and data used in this work. All relevant papers are cited, and we have adhered to the licenses and terms of use associated with these assets.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We provide the detailed documentation alongside the new assets for reproducibility.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.

- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.