The Overthinker's DIET: Cutting Token Calories with DIfficulty-AwarE Training

Weize Chen[*], **Jiarui Yuan**[*], **Tailin Jin, Ning Ding, Huimin Chen, Zhiyuan Liu**[†], **Maosong Sun**
Tsinghua University
{chenwz21, yuanjr22}@mails.tsinghua.edu.cn, liuzy@tsinghua.edu.cn

Abstract

Recent large language models (LLMs) exhibit impressive reasoning but often *overthink*, generating excessively long responses that hinder efficiency. We introduce DIET **(DIfficulty-AwarE Training)**, a framework that systematically cuts these "token calories" by integrating on-the-fly problem difficulty into the reinforcement learning (RL) process. DIET dynamically adapts token compression strategies by modulating token penalty strength and conditioning target lengths on estimated task difficulty, to optimize the performance-efficiency trade-off. We also theoretically analyze the pitfalls of naive reward weighting in group-normalized RL algorithms like GRPO, and propose *Advantage Weighting* technique, which enables stable and effective implementation of these difficulty-aware objectives. Experimental results demonstrate that DIET significantly reduces token counts while simultaneously *improving* reasoning performance. Beyond raw token reduction, we show two crucial benefits largely overlooked by prior work: (1) DIET leads to superior **inference scaling**. By maintaining high per-sample quality with fewer tokens, it enables better scaling performance via majority voting with more samples under fixed computational budgets, an area where other methods falter. (2) DIET enhances the natural positive correlation between response length and problem difficulty, ensuring verbosity is appropriately allocated, unlike many existing compression methods that disrupt this relationship. Our analyses provide a principled and effective framework for developing more efficient, practical, and high-performing LLMs. Our code is available at https://github.com/thunlp/DIET.

1 Introduction

Recent breakthroughs in large language models (LLMs) have yielded remarkable reasoning capabilities, particularly when enhanced through reinforcement learning (RL) from outcome-based rewards (OpenAI, 2024; DeepSeek-AI et al., 2025; Team, 2025). These models excel in complex domains like mathematics and coding, often generating sophisticated reasoning chains (Gandhi et al., 2025; Pan et al., 2025; Luo et al., 2025b). However, this enhanced reasoning frequently comes with a significant side effect: a dramatic increase in response length compared to base or instruction-tuned models. While some verbosity can facilitate complex thought, it often leads to ***overthinking***: models produce excessively long responses, sometimes thousands of tokens, even for simple queries (e.g., "2+3=?") (Chen et al., 2024; Chang et al., 2025; Luo et al., 2025a). This verbosity severely impacts inference latency and computational costs, hindering the practical deployment of these powerful reasoning models. Initial attempts to mitigate overthinking via supervised fine-tuning (SFT), direct preference optimization (DPO), or simple length penalties in RL objectives (Team et al., 2025; Xia et al., 2025; Aggarwal & Welleck, 2025) often struggle, leading to performance degradation, i.e., models that are concise but inaccurate.

[*]Equal Contributions
[†]Corresponding Author

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

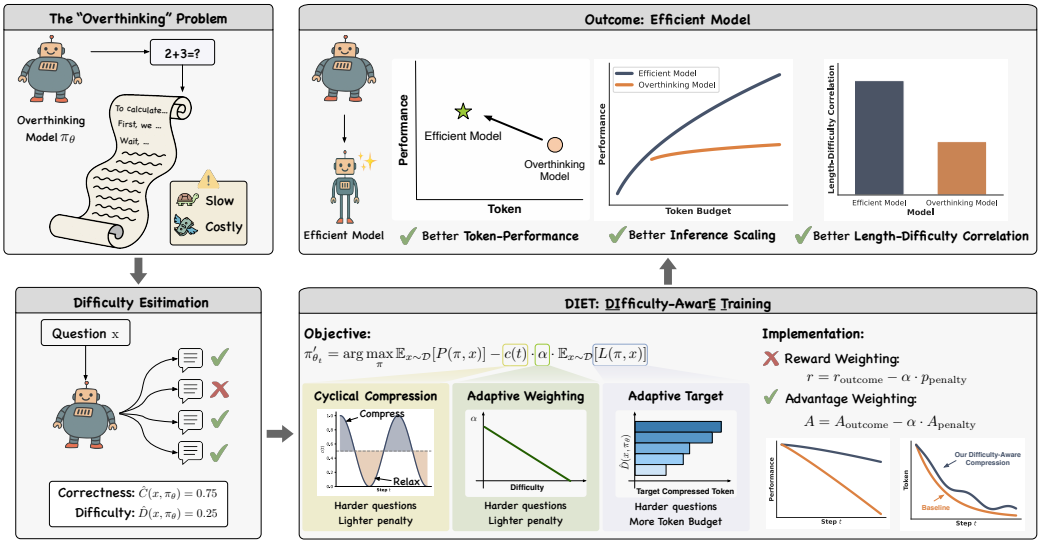

Figure 1: An overview of DIET by mitigating LLM verbosity using difficulty-aware training.

A crucial dimension often overlooked in token compression is the intrinsic link between **problem difficulty** and the *appropriate* level of verbosity. We contend that a "one-size-fits-all" compression strategy is fundamentally flawed. Complex problems may necessitate longer, more detailed reasoning, whereas simpler ones should elicit direct, concise answers. Indeed, as we later show (§2.3), LLMs often naturally exhibit a tendency to use more tokens for problems they find more challenging. Token compression methods that ignore this inherent relationship could force a suboptimal compromise across the difficulty spectrum. We argue that explicitly incorporating on-the-fly difficulty estimation is essential for developing token compression strategies that effectively preserve reasoning capabilities.

To address these challenges, we introduce DIET **(DIfficulty-AwarE Training)**, a framework to systematically "cut token calories" from overthinking LLMs. DIET integrates on-the-fly problem difficulty into the RL training process, dynamically adapting token compression based on estimated task difficulty. This enables DIET to selectively encourage conciseness for simpler problems while preserving necessary verbosity for complex ones, improving the performance-efficiency trade-off and achieving better reasoning with significantly fewer tokens than the base model.

The effective implementation of such difficulty-aware objectives within popular RL algorithms like GRPO (Shao et al., 2024) presents its own challenges. We theoretically analyze the pitfalls of naively applying weighted rewards in these settings and propose a robust *Advantage Weighting* technique that ensures stable and effective token compression training. Beyond standard performance gains, our work reveals two often-overlooked benefits of difficulty-aware compression: **(1)** Critically for practical deployment, we show that DIET leads to superior **inference scaling**. By maintaining high per-sample quality with fewer tokens, it enables significantly better inference scaling performance under small computational budgets, an area where most prior token compression efforts falter or show degradation. **(2)** We demonstrate that DIET not only compresses, but also *enhances* the natural positive correlation between an LLM's response length and problem difficulty: a desirable characteristic for adaptive reasoning that many existing compression methods inadvertently disrupt.

This paper thus formalizes the difficulty-aware token compression problem and presents DIET as a comprehensive solution. Through the core adaptive mechanisms, the enabling Advantage Weighting technique, and by demonstrating unique benefits in enhancing inference scaling and preserving adaptive verbosity, DIET offers a principled and highly effective approach for developing more efficient, practical, and powerful reasoning LLMs.

2 Problem Formulation and Preliminaries

2.1 Token Efficiency in LLMs with Reasoning Capabilities

Formally, we define the token efficiency problem studied in this paper as a multi-objective optimization challenge. Given a capable reasoning LLM policy π_θ and a distribution of reasoning problems

$\mathcal{D}$, we aim to find a policy π'_θ that optimizes both performance and token efficiency:

$$\pi'_\theta = \arg\max_\pi \mathbb{E}_{x\sim\mathcal{D}}[P(\pi, x)] - \alpha \cdot \mathbb{E}_{x\sim\mathcal{D}}[f(L(\pi, x))], \tag{1}$$

where $L(\pi, x)$ represents the expected response length (in tokens) for prompt x under policy π, $P(\pi, x)$ represents the performance (e.g., accuracy on math problems) on task x, $\alpha > 0$ is a coefficient that controls the trade-off between performance and token efficiency, and f is a monotonically increasing transformation function with different implementations in prior work.

The current paradigm of RL from outcome rewards typically focuses solely on maximizing performance without considering token efficiency ($\alpha = 0$), which leads to *overthinking*. Previous approaches to addressing this issue have generally attempted to incorporate token length into the reward function in an intuitive manner:

$$r(x, y) = r_{\text{outcome}}(x, y) - \alpha \cdot f\left(L(y)\right), \tag{2}$$

where $r_{\text{outcome}}(x, y)$ represents the outcome reward, $L(y)$ is the length of response y in tokens. However, the uniform penalty α treats all problems equally, failing to account for varying difficulty levels, and with inappropriate α, it quickly leads to suboptimal tradeoffs, where performance degrades substantially as response length is reduced.

2.2 Model-Based Difficulty Estimation

A critical insight of our work is that the optimal response length for a task should vary according to its difficulty. Intuitively, challenging problems may benefit from extended reasoning, while simpler questions can be answered concisely without sacrificing accuracy. We estimate the difficulty of a given problem based on the performance of the policy model itself during training. Formally, we define the *estimated correctness* for a given prompt x under policy π_θ based on N sampled responses $\{y_i\}_{i=1}^N$:

$$\hat{C}(x, \pi_\theta) = \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}(y_i \text{ is correct}) \quad \text{where } y_i \sim \pi_\theta(\cdot|x). \tag{3}$$

The estimated difficulty $\hat{D}(x, \pi_\theta)$ can then be defined as $1 - \hat{C}(x, \pi_\theta)$. This formulation captures an intuitive notion: tasks that the current policy consistently fails on ($\hat{C} \approx 0$) are considered difficult, while those consistently answered correctly ($\hat{C} \approx 1$) are considered easy.

Notably, popular RL algorithms such as GRPO (Shao et al., 2024) and RLOO (Ahmadian et al., 2024) already require sampling multiple responses ($N > 1$) per prompt within each training batch to estimate advantages. Therefore, computing $\hat{C}(x, \pi_\theta)$ or $\hat{D}(x, \pi_\theta)$ incurs *no computational overhead*, as the necessary samples and correctness evaluations are already part of the core RL algorithm. This makes on-the-fly difficulty estimation highly practical for integration into the training process, as we explore in subsequent sections.

2.3 Preliminary Analysis: Intrinsic Correlation of Response Length and Problem Difficulty

Before introducing our compression methods, we analyze the LLM's natural verbosity relative to task difficulty. Using R1-Distill-Qwen-1.5B and difficulty estimated via Eq. (3), we find that solution length increases with complexity, even without difficulty-aware training.

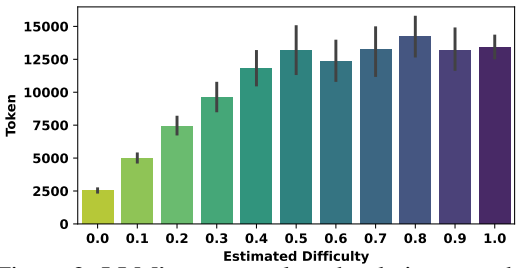

Figure 2: LLM's response length relative to problem difficulty.

Fig. 2 shows a strong positive correlation: average response length increases with estimated problem difficulty, similar findings have been observed in Estermann & Wattenhofer (2025); Wu et al. (2025). This suggests LLMs inherently allocate more tokens to harder problems, likely for more elaborate reasoning. This pivotal observation implies that common "one-size-fits-all" compression strategies, which ignore difficulty, risk either truncating vital reasoning on complex tasks or under-compressing simple ones. Indeed, many prior

compression techniques overlook this, and as shown in §4.4, can distort this natural length-complexity relationship post-training.

This baseline behavior is a cornerstone of our motivation. Since LLMs instinctively use more tokens for harder problems, effective compression should leverage this. We hypothesize that integrating on-the-fly difficulty awareness into compression training enables models to compress more intelligently: compress aggressively on easier problems while preserving token budgets for complex ones, which is the key to achieving superior performance-efficiency.

3 Difficulty-Aware Reinforcement Learning for Token Compression

The baseline analysis in §2.3 shows that LLMs naturally adjust response length based on problem difficulty. This motivates our approach: explicitly incorporating on-the-fly difficulty estimation into the RL objective. Our goal is to train models that compress responses for simpler tasks while retaining reasoning for harder ones. This section presents our methods: we first formulate how to integrate difficulty into the optimization objective, then introduce a key technique for stable implementation within policy gradients, and finally propose a strategy to refine training dynamics. An overview is shown in Fig. 1.

3.1 Formulating Difficulty-Aware Optimization Objectives

The standard RL objective for token compression (Eq. (1)) often focuses on task performance $P(\pi, x)$ with a simple, uniform trade-off α for response length $L(\pi, x)$. To instill difficulty awareness, we propose modifying this objective so that the pressure to be concise adapts to the estimated difficulty of prompt x under the current policy π_θ Eq. (3). We explore two primary strategies for this:

3.1.1 Adaptive Trade-off Parameter $\alpha_{\text{ada}}(x, \pi_\theta)$

The most direct way to instill difficulty awareness into the objective from Eq. (1) is to make its trade-off coefficient α adaptive to the estimated problem difficulty. The intuition is to apply stronger pressure for conciseness (a larger α) when the model finds the problem easy (high $\hat{C}$) and relax it when the model struggles (low $\hat{C}$).

We replace the constant α with an *adaptive* trade-off parameter, $\alpha_{\text{ada}}(x, \pi_\theta)$, calculated using the correctness estimate $\hat{C}(x, \pi_\theta)$:

$$\alpha_{\text{ada}}(x, \pi_\theta) = \alpha_{\text{base}} \cdot w(\hat{C}(x, \pi_\theta)), \tag{4}$$

where $\alpha_{\text{base}} > 0$ is a hyperparameter controlling the overall maximum strength of the efficiency objective, and $w(\cdot)$ is a monotonically increasing function mapping correctness $\hat{C} \in [0, 1]$ to a non-negative weight. A simple and effective choice is the identity function, $w(\hat{C}) = \hat{C}$, resulting in the efficiency pressure scaling linearly from 0 for the hardest problems ($\hat{C} = 0$) up to α_{base} for the easiest ones ($\hat{C} = 1$). For the penalty function $f(L(y_i))$ itself, we adopt the formulation from Team et al. (2025) and denote it as f_{Kimi}:

$$f_{\text{Kimi}}(L(y_i)) = \begin{cases} \gamma_i, & \text{if } \mathbb{I}(y_i \text{ is correct}) = 1 \\ \min(0, \gamma_i), & \text{if } \mathbb{I}(y_i \text{ is correct}) = 0, \end{cases} \tag{5}$$

where $\gamma_i = 0.5 - \frac{L(y_i) - \min_j L(y_j)}{\max_j L(y_j) - \min_j L(y_j) + \epsilon}$. The overall optimization objective effectively becomes maximizing $\mathbb{E}_{x \sim \mathcal{D}}[P(\pi, x) - \mathbb{E}_{y \sim \pi(\cdot|x)}[\alpha_{\text{ada}}(x, \pi) \cdot f_{\text{Kimi}}(L(y))]]$. This approach directly modifies the penalty strength α based on difficulty, prioritizing performance for difficult problems and conciseness for easy ones, while using a fixed form for f.

3.1.2 Dynamic Length Target $f_{\text{dyn}}(x, \pi_\theta)$

An orthogonal strategy is to make the penalty function $f(L(y))$ itself in Eq. (2) inherently adaptive to problem difficulty. Instead of relying on a fixed functional form for f that only considers $L(y)$ (or $L(y_i)$ relative to peers), we now define f to be explicitly conditioned on a dynamic *target length*, $t(x, \pi_\theta)$. This target length depends on the estimated problem difficulty, allowing a larger verbosity "budget" for harder problems.

First, we define how the target length t is determined. Based on the estimated difficulty $\hat{D}(x, \pi_\theta)$, we can sample a target t from a distribution that assigns higher values for higher difficulty:

$$t(x, \pi_\theta) \sim \text{Uniform}(\max(0, L_{\max} \cdot (\hat{D}(x, \pi_\theta) - \delta)), L_{\max} \cdot \hat{D}(x, \pi_\theta)), \tag{6}$$

where $L_{\max}$ and δ are hyperparameters defining the maximum potential target length scale, and a buffer range (e.g., 0.1). This sampling procedure assigns shorter targets for harder problems (high $\hat{D}$) and longer targets for easier ones (low $\hat{D}$).

Next, we define our difficulty-adaptive penalty function, $f_{\text{dyn}}(y_i, x, \pi_\theta, \{y_j\})$. This function quantifies the extent to which the generated length $L(y_i)$ for response $y_i \sim \pi(\cdot|x)$ exceeds its specific target $t(x, \pi_\theta)$, normalized across N sampled responses $\{y_j\}_{j=1}^N$ for the same prompt x. Let $p(y_i, t) = \max(0, L(y_i) - t(x, \pi_\theta))$ be the raw exceedance. Our adaptive penalty function is:

$$f_{\text{dyn}}(y_i, x, \pi_\theta, \{y_j\}) = \frac{p(y_i, t) - \mu_p}{\sigma_p + \epsilon}, \tag{7}$$

where μ_p and σ_p are the mean and standard deviation of $\{p(y_j, t)\}_{j=1}^N$. The overall optimization objective thus becomes maximizing $\mathbb{E}_{x\sim\mathcal{D}, \{y_j\}\sim\pi(\cdot|x)}[P(\pi, x) - \alpha \cdot f_{\text{dyn}}(y_i, t(x, \pi_\theta), \{y_j\})]$. This approach shifts the efficiency goal from minimizing absolute length to minimizing length relative to a difficulty-aware budget. These two strategies, adaptive α and adaptive length targets, can be explored as alternatives or potentially combined.

3.2 Implementing Weighted Objectives with Policy Gradients: The Advantage Weighting

Following the conceptual formulation of difficulty-aware objectives (§ 3.1), we address their RL implementation. We focus on policy gradient (PG) methods, particularly GRPO (Shao et al., 2024), which uses per-prompt, multi-sample ($N > 1$) advantage normalization for stability. Integrating our weighted difficulty-aware penalties requires careful analysis of weight-normalization interactions.

We show that naively combining task rewards (r_{outcome}) with weighted difficulty-dependent penalties into a single reward signal *before* GRPO normalization causes problematic interactions. The combined reward's normalization factor (e.g., its standard deviation) then depends on both penalty and *outcome variances*. Since outcome variance often correlates with problem difficulty (being highest for medium-difficulty problems and lowest for very easy or very hard ones), the intended effect of the penalty weight in the final normalized advantage becomes distorted. Specifically, as derived in Appendix B, the token penalty is *unexpectedly weakened* for problems of modest difficulty where outcome variance is high. Conversely, it can be *unexpectedly exacerbated* for problems where outcome variance is very low (i.e., those the model consistently gets right or wrong), undermining the desired difficulty-aware adaptation.

Remark 1 (Pitfall of Naive Reward Weighting)**.** *In group-normalized PG algorithms like GRPO, combining reward components **before** normalization causes the penalty weight's effect to be distorted by the task's outcome variance, undermining the intended difficulty-aware adaptation.*

To prevent this distortion and ensure correct weighting, we propose *Advantage Weighting*. This method normalizes advantages for the outcome reward ($r_{\text{outcome},i}$) and raw penalty magnitude (p_i) separately, before combining them using the difficulty-aware penalty weight. Specifically, advantages for task outcome and the penalty term are normalized independently using their per-prompt means ($\mu_{\text{outcome}}, \mu_p$) and standard deviations ($\sigma_{\text{outcome}}, \sigma_p$) from N rollouts:

$$\hat{A}_{\text{outcome},i} = \frac{r_{\text{outcome},i} - \mu_{\text{outcome}}}{\sigma_{\text{outcome}} + \epsilon}, \quad \hat{A}_{p,i} = \frac{p_i - \mu_p}{\sigma_p + \epsilon}. \tag{8}$$

Here, p_i is the raw token penalty magnitude (e.g., Eqs. (5) and (7)). The final policy gradient advantage $\hat{A}'_i$ combines the normalized components weighted by α' (e.g., α_{ada} or α from §3.1):

$$\hat{A}'_i = \hat{A}_{\text{outcome},i} - \alpha' \cdot \hat{A}_{p,i}. \tag{9}$$

Thus, each component is scaled by its own variance *before* adaptive weighting. This allows α to correctly modulate the normalized penalty's influence relative to that of the normalized outcome, faithfully reflecting the intended difficulty-aware trade-off and resolving the distortion. Experiments in §4.5 validate this improvement, showing significant benefits over naive reward weighting.

3.3 Refining Training Dynamics: Cyclical Compression Pressure

Although the difficulty-aware objectives (§ 3.1) implemented with Advantage Weighting (§ 3.2) provide robust adaptive compression, constant pressure may risk premature convergence on brevity or hinder exploration. To address this and potentially enhance the performance-efficiency trade-off, we explore temporally modulating the compression intensity during training.

Inspired by annealing schedules, we cyclically vary the difficulty-aware penalty strength using a time-varying cosine modulation factor $c(t) = 0.5\left(1 + \cos\left(\frac{2\pi t}{T}\right)\right)$, where T is the cycle period. This factor $c(t)$ smoothly oscillates between 1 (maximum pressure) and 0 (minimum pressure) and scales the difficulty-aware penalty component within the final advantage calculation (Eq. (9)):

$$\hat{A}'_i(t) = \hat{A}_{\text{outcome},i} - c(t) \cdot \alpha_{\text{ada}}(x, \pi_\theta) \cdot \hat{A}_{p,i}. \tag{10}$$

This temporal variation aims to improve robustness and the final performance-efficiency trade-off, potentially allowing the model to escape local optima related to excessive brevity and better balance reasoning consolidation with conciseness pressure.

3.4 The DIET: DIfficulty-AwarE Training Method

Our approach, DIET, effectively integrates all the elements discussed in this section. Specifically, DIET employs an objective that synergizes the principles of adaptive penalty strength (§3.1.1) and dynamic length target (§3.1.2). This combined objective is implemented using Advantage Weighting (§3.2) and its training is optimized with Cyclical Compression Pressure (§3.3). This holistic strategy, DIET, underpins the best performance-efficiency results presented in our later experiments.

4 Experimental Validation

4.1 Experimental Setup

Base Model and Algorithm. Our experiments use the R1-Distilled Qwen 1.5B model (DeepSeek-AI et al., 2025). For RL algorithm, we employ GRPO (Shao et al., 2024). All proposed RL methods are built on GRPO and use the Advantage Weighting technique (§3.2).

Training. We use the DeepScaleR dataset (Luo et al., 2025b), featuring high-quality mathematical problems of diverse complexities. We use veRL (Sheng et al., 2024) as the training framework, and train models on 8 A100 GPUs. For more details of training, please refer to Appendix D.

Baselines. We compare against: (1) The **Base Model** without any compression. (2) **SFT- & DPO-Based Methods**: Kimi 1.5 SFT (Team et al., 2025) (fine-tuning on shortest correct responses), Kimi 1.5 DPO (Team et al., 2025) (using shortest correct as positive DPO examples), and TokenSkip (Xia et al., 2025) (SFT on responses with redundant tokens removed). (3) **Other RL-Based Methods**: CosFn (Chang et al., 2025), O1-Pruner (Luo et al., 2025a), and Kimi 1.5 RL (Team et al., 2025).

Evaluation. We assess Pass@1 (P@1) and average response length (Tokens, Tok) on MATH 500 (Hendrycks et al., 2021), AIME 2024, AMC 2023, Olympiad Bench (He et al., 2024), and Minerva (Lewkowycz et al., 2022). We sample 32 samples for each question in AIME24, and 10 samples for others to estimate the P@1. Details for evaluation are presented in Appendix E.

4.2 DIET Achieves Improved Performance with Reduced Tokens

Table 1 summarizes the performance and token efficiency of our methods against baselines. The Base Model establishes a strong performance benchmark but with substantial verbosity, underscoring the *overthinking* issue.

Among existing approaches, Kimi 1.5 DPO is a strong baseline that slightly reduces tokens and improves the performance over the base model. TokenSkip achieves extreme token reduction, but its average P@1 drops significantly, demonstrating that aggressive, non-nuanced compression severely degrades reasoning. Standard RL baselines generally achieve more significant token reduction, however, the performances are slightly inferior to the base model. These methods highlight the difficulty of achieving both high performance and low token counts simultaneously without more sophisticated adaptation.

Table 1: Average performance (Pass@1, %) and token length of R1-Distilled Qwen 1.5B trained with different token compression methods. For each benchmark, highest P@1 is bolded. For the "Macro Average" columns, bolding indicates the best P@1 and a favorable performance-efficiency trade-off.

Method	MATH 500		AIME 2024		AMC 2023		Olympiad.		Minerva		Macro Average	
	P@1	Tok	P@1	Tok	P@1	Tok	P@1	Tok	P@1	Tok	P@1	Tok
Base Model	82.1	5534	28.5	16590	62.7	10615	43.5	11587	26.0	7076	48.6	10280
SFT- & DPO-Based												
Kimi 1.5 SFT	68.5	6761	22.0	17400	60.4	9323	39.4	10036	23.6	2804	42.7 $_{-12.1\%}$	9865 $_{-4.0\%}$
Kimi 1.5 DPO	**83.3**	4464	31.7	13389	63.0	8678	**44.5**	9604	26.9	6070	49.9 $_{+2.7\%}$	8441 $_{-17.9\%}$
TokenSkip	64.1	1120	6.8	2231	37.3	1401	25.8	2061	20.7	1674	30.9 $_{-36.4\%}$	1697 $_{-83.5\%}$
RL-Based												
CosFn	75.6	2735	27.5	12492	61.1	6970	42.9	8307	**27.1**	3485	46.8 $_{-3.5\%}$	6798 $_{-33.9\%}$
O1-Pruner	79.1	2531	25.0	8961	62.5	5010	39.0	5242	23.7	2400	45.9 $_{-5.4\%}$	4829 $_{-53.0\%}$
Kimi 1.5 RL	66.3	1552	18.8	9109	44.7	3808	28.5	4774	16.7	1009	35.0 $_{-27.9\%}$	4050 $_{-60.6\%}$
Our Difficulty-Aware Methods												
Dynamic Target (§3.1.2)	82.1	2792	27.7	10288	63.4	6017	43.4	6490	26.3	2700	48.6 $_{+0.0\%}$	5657 $_{-45.0\%}$
Adaptive Weighting (§3.1.1)	82.7	2876	**32.2**	10255	64.4	5819	43.7	6494	26.6	3170	49.9 $_{+2.8\%}$	5723 $_{-44.3\%}$
DIET (§3.1.1+§3.1.2)	83.0	3061	31.8	10578	**65.4**	6425	43.7	6917	26.9	3505	**50.2** $_{+3.3\%}$	**6097** $_{-40.7\%}$

Our **Difficulty-Aware Methods** consistently yield a better performance-efficiency frontier. The *Dynamic Target* approach maintains the Base Model's Macro Average P@1 while reducing average tokens by a substantial 45.0%. The *Adaptive Weighting* method further improves the Macro Average P@1 to 49.9% (+2.8% over Base), matching Kimi 1.5 DPO's performance but with considerably fewer tokens. Significantly, DIET, which leverages both Dynamic Target and Adaptive Weighting, achieves the best P@1. This performance is achieved with an average token count of only 6097, a significant 40.7% reduction compared to the Base Model. Beyond these quantitative gains, our qualitative analysis (Appendix F) further reveals that DIET training progressively refines the model's reasoning style, leading to more structured language, concise calculations, and a marked reduction in unnecessary self-doubt and redundant post-solution exploration.

These results underscore the effectiveness of incorporating nuanced difficulty awareness. While methods like TokenSkip can produce very short answers, they do so at an unacceptable performance drop. Our difficulty-aware methods, especially the combined strategy, demonstrate that it is possible to achieve substantial token reductions while maintaining, and even enhancing the sophisticated reasoning capabilities of the LLM, leading to a superior performance-efficiency trade-off.

4.3 DIET's Advantage in Inference Scaling

An often-overlooked benefit of token compression is its potential to enhance inference scaling performance under a fixed total token budget. Shorter responses allow for more samples to be drawn for techniques like majority voting, which can improve overall accuracy if per-sample quality is maintained. While previous token compression work typically focused on single-sample token counts and Pass@1, we investigate this inference scaling behavior. We show that many existing compression methods fail to translate token savings into improved scaling performance, whereas DIET achieve superior majority voting accuracy, particularly at practical, lower token budgets.

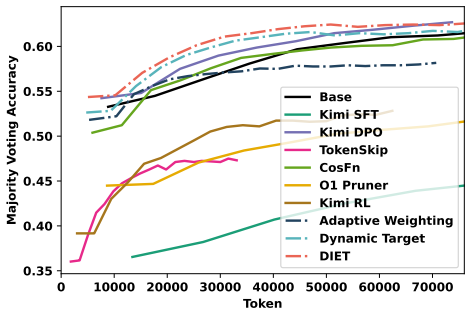

Figure 3: Micro average of majority voting Pass@1 on all the benchmarks.

Fig. 3 plots the micro average of majority voting Pass@1 across all math benchmarks against the total token budget. The results highlight the advantages of our approaches. Methods like TokenSkip and Kimi SFT, despite allowing many samples due to extreme compression, exhibit low and quickly stagnating majority voting accuracy, confirming that their severe per-sample quality degradation undermines scaling benefits. While stronger baselines like Kimi DPO show more respectable scaling, their increase in the scaling performance is slow. The Base Model itself requires a substantial token budget before multiple samples yield significant gains.

In contrast, our difficulty-aware methods demonstrate advantages. Notably, DIET and Dynamic Target achieve significantly higher majority voting accuracy at low token budgets. This demonstrates that DIET effectively preserves per-sample quality, allowing the benefits of increased sampling to manifest early. Although the final accuracies are similar to other baselines, the faster convergence

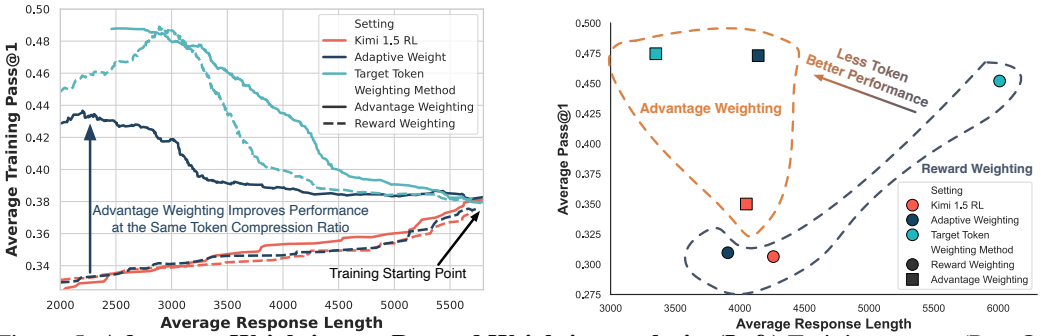

Figure 5: **Advantage Weighting vs. Reward Weighting analysis. (Left)** Training curves (Pass@1 vs. Response Length) demonstrate better performance with Advantage Weighting. **(Right)** Final evaluation results show Advantage Weighting yields superior performance-efficiency points.

makes its scaling more practical. Overall, the results show that our methods effectively translate token savings into improved inference scaling performance, which is crucial for practical applications.

4.4 DIET Enhances Length-Difficulty Correlation

An ideal token compression method should not only reduce verbosity but also preserve, or even enhance, the intelligent allocation of tokens based on problem difficulty, a natural tendency observed in base LLMs (§2.3). Uniform compression risks disrupting this by being overly terse on complex problems or insufficiently concise on simple ones. We investigate this by measuring the Pearson correlation between estimated problem difficulty and generated response length across our test benchmarks; a higher positive correlation indicates more appropriately scaled verbosity.

Fig. 4 demonstrates that our difficulty-aware methods excel at maintaining and enhancing this crucial correlation. Our method, DIET, achieves the highest correlation, surpassing the Base Model's inherent correlation (dashed horizontal line) and other baselines. This indicates that DIET successfully learns to modulate verbosity in tight alignment with problem difficulty, consistent with our design goals. Notably, another strong baseline, CosFn, also shows a high correlation, nearly matching DIET, but as shown in §4.2, it fails to preserve performance when reducing tokens. In contrast, other compression techniques significantly degrade this adaptive characteristic. For instance, Kimi DPO and Kimi SFT show a markedly weaker correlation than the Base Model, implying their compression mechanisms are less sensitive to problem difficulty. O1 Pruner also shows a reduced correlation. This suggests that while these methods reduce tokens, they do so in a more uniform or difficulty-agnostic manner.

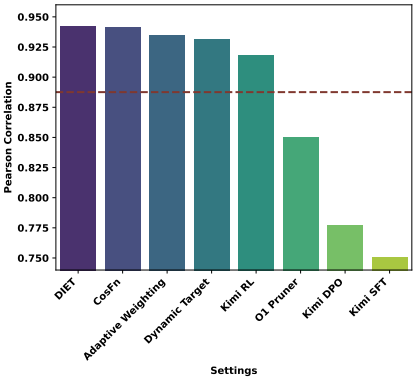

Figure 4: Pearson correlation between problem difficulty and average response length of different methods. All p-values are lower than 0.01. Methods that are not included have lower correlation.

These results underscore that intelligent compression, as achieved by DIET, is not merely about token reduction but about doing so in a way that respects problem complexity. By strengthening the positive relationship between difficulty and response length, DIET ensures a more rational allocation of computational budget during inference, contributing to its robust performance-efficiency.

4.5 Ensuring Stable Difficulty-Aware RL Training with Advantage Weighting

We empirically validate our proposed Advantage Weighting method (§3.2), designed to overcome the signal distortion pitfalls of naive reward weighting within normalized policy gradient algorithms like GRPO. As highlighted in our analysis (§3.2), naive weighting can interact poorly with outcome variance, hindering effective penalty application.

Fig. 5 (left) plots the training dynamics (Average Training Pass@1 vs. Average Response Length). Advantage Weighting (solid lines) consistently maintains higher performance than Reward Weighting

(dashed lines) as models compress responses (moving right-to-left during training). This demonstrates more effective learning during token reduction across all penalty settings when using Advantage Weighting. The final evaluation results (Fig. 5, right) further reinforce this. Models trained with Advantage Weighting achieve superior performance-efficiency trade-offs, occupying the desirable top-left region (high performance, low token count). The DIET without advantage weighting fails to deliver reasonable performance, we therefore omit it from the visualization.

These empirical results strongly support our theoretical analysis (§ 3.2). Advantage Weighting is crucial for the stable and effective implementation of weighted objectives (like difficulty-aware penalties) in normalized PG algorithms. Naive reward weighting leads to suboptimal training and significantly poorer final performance-efficiency outcomes.

5 Related Work

Efficient reasoning in LLMs has recently attracted significant attention, as methods that boost reasoning performance, often via RL with outcome-based rewards, unintentionally induce verbose, overthought outputs. A growing body of work has therefore aimed at compressing the reasoning to reduce inference costs without greatly compromising accuracy.

Prompt-Based Approaches. Initial efforts primarily explore prompt engineering to reduce verbosity. For example, Chain-of-Draft (Xu et al., 2025) and Sketch-of-Thought (Aytes et al., 2025) restructure reasoning by having the model first draft a concise outline before finalizing the answer, while Constrained-CoT (Nayab et al., 2024) imposes length limits via prompts. Though these methods allow quick, zero-shot adjustments, their effectiveness is limited since they do not alter the model's internal parameters.

Training-Based Compression via Supervised and RL Methods. More fundamental approaches modify the model's training process to inherently produce more concise reasoning chains. Supervised fine-tuning (SFT) techniques aim to internalize efficient reasoning patterns by training on compressed or optimized CoT data. SPIRIT-FT (Cui et al., 2025) and Skip-Steps (Liu et al., 2024) train models on reasoning steps deemed crucial. Other SFT approaches focus on distilling longer reasoning chains from capable models into shorter, equivalent ones (Yu et al., 2024; Kang et al., 2024; Munkhbat et al., 2025). Some methods even train models to reason implicitly in latent space, generating concise outputs without explicit step-by-step textual reasoning, such as Coconut (Hao et al., 2024), CCoT (Cheng & Durme, 2024), and Implicit-CoT (Deng et al., 2024).

Reinforcement learning (RL) offers another avenue, often by incorporating length penalties directly into the reward function. This typically involves combining the primary outcome-based reward (e.g., correctness) with a secondary reward term that penalizes longer sequences. Examples include approaches by Arora & Zanette (2025), L1 (Aggarwal & Welleck, 2025), Kimi-1.5 (Team et al., 2025), and work exploring how length penalties can be used to stabilize training (Chang et al., 2025). Our work builds on these efforts by comprehensively introducing difficulty awareness into the compression process, analyzing the impact of data difficulty on compression training, proposing adaptive reward shaping techniques that dynamically adjust token penalties based on on-the-fly difficulty estimation, and addressing methodological issues in applying such rewards within group-normalized RL methods.

6 Conclusion

To combat LLM "overthinking" and its inherent inefficiencies, we introduced DIET (**DIfficulty-AwarE Training**), a framework that intelligently "cuts token calories" by integrating on-the-fly problem difficulty into the RL process for adaptive compression. DIET achieves satisfactory reasoning performance while significantly reducing token counts. Beyond these primary gains, DIET uniquely preserves and enhances the natural positive correlation between response length and problem difficulty, ensuring appropriate verbosity. Furthermore, it translates these efficiencies into superior **inference scaling**, delivering better performance under fixed computational budgets, which is a crucial advantage over prior methods that often falter in this regard. DIET thus offers a principled, effective, and thoroughly-validated strategy for developing more practical, efficient, and ultimately more capable large language models.

Acknowledgement

This work is supported by the National Key R&D Program of China (No.2022ZD0116312), the National Natural Science Foundation of China (No. 62236004), the high-quality development project of MIIT and a grant from the Guoqiang Institute, Tsinghua University. This work is also supported by the AI9Stars community.

References

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

Arash Ahmadian, Chris Cremer, Matthias Gallé, Marzieh Fadaee, Julia Kreutzer, Olivier Pietquin, Ahmet Üstün, and Sara Hooker. Back to basics: Revisiting reinforce-style optimization for learning from human feedback in llms. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 12248–12267. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.662. URL https://doi.org/10.18653/v1/2024.acl-long.662.

Daman Arora and Andrea Zanette. Training language models to reason efficiently. *CoRR*, abs/2502.04463, 2025. doi: 10.48550/ARXIV.2502.04463. URL https://doi.org/10.48550/arXiv.2502.04463.

Simon A Aytes, Jinheon Baek, and Sung Ju Hwang. Sketch-of-thought: Efficient llm reasoning with adaptive cognitive-inspired sketching. *arXiv preprint arXiv:2503.05179*, 2025.

Edward Y. Chang, Yuxuan Tong, Morry Niu, Graham Neubig, and Xiang Yue. Demystifying long chain-of-thought reasoning in llms. *CoRR*, abs/2502.03373, 2025. doi: 10.48550/ARXIV.2502.03373. URL https://doi.org/10.48550/arXiv.2502.03373.

Xingyu Chen, Jiahao Xu, Tian Liang, Zhiwei He, Jianhui Pang, Dian Yu, Linfeng Song, Qiuzhi Liu, Mengfei Zhou, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. Do NOT think that much for 2+3=? on the overthinking of o1-like llms. *CoRR*, abs/2412.21187, 2024. doi: 10.48550/ARXIV.2412.21187. URL https://doi.org/10.48550/arXiv.2412.21187.

Jeffrey Cheng and Benjamin Van Durme. Compressed chain of thought: Efficient reasoning through dense representations. *CoRR*, abs/2412.13171, 2024. doi: 10.48550/ARXIV.2412.13171. URL https://doi.org/10.48550/arXiv.2412.13171.

Yingqian Cui, Pengfei He, Jingying Zeng, Hui Liu, Xianfeng Tang, Zhenwei Dai, Yan Han, Chen Luo, Jing Huang, Zheng Li, Suhang Wang, Yue Xing, Jiliang Tang, and Qi He. Stepwise perplexity-guided refinement for efficient chain-of-thought reasoning in large language models. *CoRR*, abs/2502.13260, 2025. doi: 10.48550/ARXIV.2502.13260. URL https://doi.org/10.48550/arXiv.2502.13260.

DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, and S. S. Li. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *CoRR*, abs/2501.12948, 2025. doi: 10.48550/ARXIV.2501.12948. URL https://doi.org/10.48550/arXiv.2501.12948.

Yuntian Deng, Yejin Choi, and Stuart M. Shieber. From explicit cot to implicit cot: Learning to internalize cot step by step. *CoRR*, abs/2405.14838, 2024. doi: 10.48550/ARXIV.2405.14838. URL https://doi.org/10.48550/arXiv.2405.14838.

Benjamin Estermann and Roger Wattenhofer. Reasoning effort and problem complexity: A scaling analysis in llms. *arXiv preprint arXiv:2503.15113*, 2025.

Kanishk Gandhi, Ayush Chakravarthy, Anikait Singh, Nathan Lile, and Noah D Goodman. Cognitive behaviors that enable self-improving reasoners, or, four habits of highly effective stars. *arXiv preprint arXiv:2503.01307*, 2025.

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. *CoRR*, abs/2412.06769, 2024. doi: 10.48550/ARXIV.2412.06769. URL https://doi.org/10.48550/arXiv.2412.06769.

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Leng Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, Jie Liu, Lei Qi, Zhiyuan Liu, and Maosong Sun. Olympiadbench: A challenging benchmark for promoting AGI with olympiad-level bilingual multimodal scientific problems. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar (eds.), *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024*, pp. 3828–3850. Association for Computational Linguistics, 2024. doi: 10.18653/V1/2024.ACL-LONG.211. URL https://doi.org/10.18653/v1/2024.acl-long.211.

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. In Joaquin Vanschoren and Sai-Kit Yeung (eds.), *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*, 2021. URL https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/be83ab3ecd0db773eb2dc1b0a17836a1-Abstract-round2.html.

Yu Kang, Xianghui Sun, Liangyu Chen, and Wei Zou. C3ot: Generating shorter chain-of-thought without compromising effectiveness. *CoRR*, abs/2412.11664, 2024. doi: 10.48550/ARXIV.2412.11664. URL https://doi.org/10.48550/arXiv.2412.11664.

Aitor Lewkowycz, Anders Andreassen, David Dohan, Ethan Dyer, Henryk Michalewski, Vinay V. Ramasesh, Ambrose Slone, Cem Anil, Imanol Schlag, Theo Gutman-Solo, Yuhuai Wu, Behnam Neyshabur, Guy Gur-Ari, and Vedant Misra. Solving quantitative reasoning problems with language models. In Sanmi Koyejo, S. Mohamed, A. Agarwal, Danielle Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems 35: Annual Conference on Neural Information Processing Systems 2022, NeurIPS 2022, New Orleans, LA, USA, November 28 - December 9, 2022*, 2022. URL http://papers.nips.cc/paper_files/paper/2022/hash/18abbeef8cfe9203fdf9053c9c4fe191-Abstract-Conference.html.

Tengxiao Liu, Qipeng Guo, Xiangkun Hu, Cheng Jiayang, Yue Zhang, Xipeng Qiu, and Zheng Zhang. Can language models learn to skip steps? In Amir Globersons, Lester Mackey, Danielle Belgrave, Angela Fan, Ulrich Paquet, Jakub M. Tomczak, and Cheng Zhang (eds.), *Advances in Neural Information Processing Systems 38: Annual Conference on Neural Information Processing Systems 2024, NeurIPS 2024, Vancouver, BC, Canada, December 10 - 15, 2024*, 2024. URL http://papers.nips.cc/paper_files/paper/2024/hash/504fa7e518da9d1b53a233ed20a38b46-Abstract-Conference.html.

Haotian Luo, Li Shen, Haiying He, Yibo Wang, Shiwei Liu, Wei Li, Naiqiang Tan, Xiaochun Cao, and Dacheng Tao. O1-pruner: Length-harmonizing fine-tuning for o1-like reasoning pruning. *CoRR*, abs/2501.12570, 2025a. doi: 10.48550/ARXIV.2501.12570. URL https://doi.org/10.48550/arXiv.2501.12570.

Michael Luo, Sijun Tan, Justin Wong, Xiaoxiang Shi, William Y. Tang, Manan Roongta, Colin Cai, Jeffrey Luo, Tianjun Zhang, Li Erran Li, Raluca Ada Popa, and Ion Stoica. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl. https://pretty-radio-b75.notion.site/DeepScaleR-Surpassing-O1-Preview-with-a-1-5B-Model-by-Scaling-RL-19681902c1468005bed8ca303013a 2025b. Notion Blog.

Tergel Munkhbat, Namgyu Ho, Seo Hyun Kim, Yongjin Yang, Yujin Kim, and Se-Young Yun. Self-training elicits concise reasoning in large language models. *CoRR*, abs/2502.20122, 2025. doi: 10.48550/ARXIV.2502.20122. URL https://doi.org/10.48550/arXiv.2502.20122.

Sania Nayab, Giulio Rossolini, Giorgio C. Buttazzo, Nicolamaria Manes, and Fabrizio Giacomelli. Concise thoughts: Impact of output length on LLM reasoning and cost. *CoRR*, abs/2407.19825, 2024. doi: 10.48550/ARXIV.2407.19825. URL https://doi.org/10.48550/arXiv.2407.19825.

OpenAI. Learning to reason with llms, 2024. URL https://openai.com/index/learning-to-reason-with-llms/. Accessed: 2025-04-06.

Jiayi Pan, Junjie Zhang, Xingyao Wang, Lifan Yuan, Hao Peng, and Alane Suhr. Tinyzero. https://github.com/Jiayi-Pan/TinyZero, 2025. Accessed: 2025-01-24.

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. *CoRR*, abs/2402.03300, 2024. doi: 10.48550/ARXIV.2402.03300. URL https://doi.org/10.48550/arXiv.2402.03300.

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.

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.

Qwen Team. Qwq-32b: Embracing the power of reinforcement learning, March 2025. URL https://qwenlm.github.io/blog/qwq-32b/.

Yuyang Wu, Yifei Wang, Tianqi Du, Stefanie Jegelka, and Yisen Wang. When more is less: Understanding chain-of-thought length in llms. *arXiv preprint arXiv:2502.07266*, 2025.

Heming Xia, Yongqi Li, Chak Tou Leong, Wenjie Wang, and Wenjie Li. Tokenskip: Controllable chain-of-thought compression in llms. *CoRR*, abs/2502.12067, 2025. doi: 10.48550/ARXIV.2502.12067. URL https://doi.org/10.48550/arXiv.2502.12067.

Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. Chain of draft: Thinking faster by writing less. *CoRR*, abs/2502.18600, 2025. doi: 10.48550/ARXIV.2502.18600. URL https://doi.org/10.48550/arXiv.2502.18600.

Ping Yu, Jing Xu, Jason Weston, and Ilia Kulikov. Distilling system 2 into system 1. *CoRR*, abs/2407.06023, 2024. doi: 10.48550/ARXIV.2407.06023. URL https://doi.org/10.48550/arXiv.2407.06023.

A Limitations

While DIET demonstrates significant advancements in creating more token-efficient and performant reasoning LLMs, we acknowledge certain limitations.

Our empirical validation has primarily focused on mathematical reasoning benchmarks. Although these tasks robustly test complex reasoning and verbosity patterns, the generalization of DIET's benefits to other diverse domains warrants more extensive investigation. The optimal balance between conciseness and necessary verbosity might vary across these different applications. Still, we think that mathematical reasoning is a representative reasoning task that can be used to validate the effectiveness of our methods.

Furthermore, while the principles of our difficulty-aware framework are conceptually orthogonal to many existing RL-based token compression techniques, suggesting potential for synergistic combinations, this work did not investigate such hybrid approaches. The empirical exploration of combining DIET with other methods to potentially achieve further performance enhancements remains an avenue for future research.

Addressing these areas could lead to even more versatile and efficient large language models.

B Derivation of Advantage Distortion under Naive Reward Weighting

This appendix provides the mathematical details supporting Remark 1 in §3.2, demonstrating how naive reward weighting before normalization in algorithms like GRPO distorts the intended effect of a penalty term. To isolate the distortion caused by the normalization procedure itself, we analyze the case with a constant penalty trade-off parameter, α_{base}.

The naive approach combines the outcome reward $r_{\text{outcome},i}$ and the penalty term p_i for response y_i to prompt x into a single reward:

$$r'_i = r_{\text{outcome},i} - \alpha_{\text{base}} \cdot p_i. \tag{11}$$

GRPO then computes the normalized advantage based on the empirical mean $\mu_{r'}$ and standard deviation $\sigma_{r'}$ of these combined rewards $\{r'_j\}_{j=1}^N$ for the N responses sampled for prompt x:

$$\hat{A}'_i = \frac{r'_i - \mu_{r'}}{\sigma_{r'} + \epsilon}. \tag{12}$$

Let $\mu_{\text{outcome}}, \sigma_{\text{outcome}}$ and μ_p, σ_p be the empirical means and standard deviations of the outcome rewards and penalty terms, respectively, within the batch for prompt x. Then $\mu_{r'} = \mu_{\text{outcome}} - \alpha_{\text{base}}\mu_p$. Assuming r_{outcome} and p are approximately independent given the prompt within the batch, the variance of the combined reward is:

$$\sigma^2_{r'} = \text{Var}(r'_i) = \text{Var}(r_{\text{outcome},i} - \alpha_{\text{base}} p_i) \approx \text{Var}(r_{\text{outcome},i}) + \alpha^2_{\text{base}}\text{Var}(p_i) = \sigma^2_{\text{outcome}} + \alpha^2_{\text{base}}\sigma^2_p. \tag{13}$$

Substituting the mean and standard deviation into the advantage calculation:

$$\begin{aligned}
\hat{A}'_i &= \frac{(r_{\text{outcome},i} - \alpha_{\text{base}} p_i) - (\mu_{\text{outcome}} - \alpha_{\text{base}}\mu_p)}{\sqrt{\sigma^2_{\text{outcome}} + \alpha^2_{\text{base}}\sigma^2_p} + \epsilon} \\
&= \frac{(r_{\text{outcome},i} - \mu_{\text{outcome}}) - \alpha_{\text{base}}(p_i - \mu_p)}{\sqrt{\sigma^2_{\text{outcome}} + \alpha^2_{\text{base}}\sigma^2_p} + \epsilon} \\
&= \underbrace{\frac{1}{\sqrt{\sigma^2_{\text{outcome}} + \alpha^2_{\text{base}}\sigma^2_p} + \epsilon}}_{\text{outcome scaling}}(r_{\text{outcome},i} - \mu_{\text{outcome}}) - \underbrace{\frac{\alpha_{\text{base}}}{\sqrt{\sigma^2_{\text{outcome}} + \alpha^2_{\text{base}}\sigma^2_p} + \epsilon}}_{\text{Effective Penalty Scaling: } \hat{\tau}_p}(p_i - \mu_p).
\end{aligned} \tag{14}$$

The crucial term is the effective penalty scaling factor $\hat{\tau}_p$. This factor dictates how strongly the centered penalty term $(p_i - \mu_p)$ contributes to the advantage signal and the subsequent policy gradient update $\nabla_\theta J \propto \mathbb{E}[\nabla_\theta \log \pi_\theta \cdot \hat{A}']$.

Critically, $\hat{\tau}_p$ depends on the task outcome variance $\sigma^2_{\text{outcome}}$. For binary outcome rewards (correct/incorrect), $\sigma^2_{\text{outcome}} = \hat{C}(1 - \hat{C})$, where $\hat{C}$ is the estimated correctness (Eq. 3). This introduces an unintended dependency on problem difficulty, distorting the effect of the constant hyperparameter α_{base}:

- **Easy/Hard Problems ($\hat{C} \approx 1$ or 0):** In these cases, the outcome variance $\sigma^2_{\text{outcome}} \approx 0$. The effective penalty scaling becomes $\hat{\tau}_p \approx \frac{\alpha_{\text{base}}}{\sqrt{\alpha^2_{\text{base}}\sigma^2_p}} = \frac{\alpha_{\text{base}}}{|\alpha_{\text{base}}|\sigma_p} = \frac{1}{\sigma_p}$ (assuming $\alpha_{\text{base}} > 0$). The intended constant penalty weight α_{base} is effectively removed by the normalization, and the penalty's influence is scaled only by its own standard deviation, σ_p. The hyperparameter no longer controls the penalty strength.
- **Intermediate Difficulty Problems ($\hat{C} \approx 0.5$):** Here, the outcome variance $\sigma^2_{\text{outcome}}$ is maximal (≈ 0.25 for binary rewards). The denominator $\sqrt{\sigma^2_{\text{outcome}} + \alpha^2_{\text{base}}\sigma^2_p}$ is larger, meaning the effective penalty scaling $\hat{\tau}_p$ is minimized. The influence of the constant penalty α_{base} is most strongly suppressed precisely when the task difficulty is intermediate.

This analysis reveals that normalizing the combined reward r' distorts the effect of the penalty weight α_{base}. The interaction with the difficulty-dependent outcome variance $\sigma^2_{\text{outcome}}$ prevents α_{base} from applying consistent pressure. The policy gradient updates do not accurately reflect the intended penalty strength, potentially hindering convergence.

The situation becomes even more complex when using the adaptive weight $\alpha_{\text{ada}}(x, \pi_\theta) = \alpha_{\text{base}} \cdot w(\hat{C})$ from the main paper. If we substitute α_{ada} into Eq. 14 and assume a linear weighting function $w(\hat{C}) = \hat{C}$, the effective penalty scaling factor $\hat{\tau}_p$ becomes a non-linear function of correctness $\hat{C}$.

To visualize this, we can analyze its squared form, $\hat{\tau}_p^2$:

$$\hat{\tau}_p^2 = \frac{\alpha_{\text{base}}^2 \hat{C}^2}{\hat{C}(1-\hat{C}) + \alpha_{\text{base}}^2 \hat{C}^2 \sigma_p^2}. \tag{15}$$

Assuming $\hat{C} \neq 0$ for now, this simplifies to:

$$\hat{\tau}_p^2 = \frac{\alpha_{\text{base}}^2}{\frac{1-\hat{C}}{\hat{C}} + \alpha_{\text{base}}^2 \sigma_p^2} = \frac{\alpha_{\text{base}}^2}{\frac{1}{\hat{C}} - 1 + \alpha_{\text{base}}^2 \sigma_p^2}. \tag{16}$$

This relationship is highly non-linear. To illustrate, we can take the case where $\alpha_{\text{base}} = 0.1$ and $\sigma_p^2 = 1$ (a plausible assumption if the penalty term is also normalized, as we have done in Eq. (7)). The resulting function for $\hat{\tau}_p^2$ is plotted in Fig. 6. The curve is far from the linear relationship we desire; the penalty's influence is negligible for most of the difficulty range ($\hat{C} < 0.8$) and then increases sharply only as the problem becomes extremely easy. This interaction obscures the difficulty-aware trade-off that we would like to intervene.

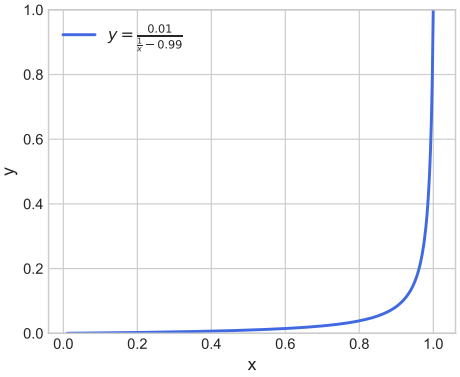

Figure 6: An example plot of Eq. (16) with $\alpha_{\text{base}} = 0.1$ and $\sigma_p^2 = 1$.

The *Advantage Weighting* approach presented in § 3.2 avoids all these issues by normalizing the outcome and penalty advantages separately *before* applying the weight, thus preserving the intended adaptive scaling and ensuring stable, effective training.

C Unbiased Gradient Estimation with Advantage Weighting

In this section, we provide a formal derivation to show that the core structure of our *Advantage Weighting* method provides an unbiased estimate of the true policy gradient. Our goal is to first derive the unbiased estimator and then explain its connection to our practical implementation.

First, we define the optimization objective $J(\theta)$, which combines the task performance reward with the difficulty-aware length penalty:

$$J(\theta) = \mathbb{E}_{x\sim\mathcal{D}, y\sim\pi_\theta(\cdot|x)}\left[r_{\text{total}}(x,y)\right], \quad \text{where} \quad r_{\text{total}}(x,y) = r_{\text{outcome}}(x,y) - \alpha'(x,\pi_\theta)\cdot p(y). \tag{17}$$

Here, r_{outcome} is the task reward, $p(y)$ is the penalty magnitude, and $\alpha'(x,\pi_\theta)$ is the adaptive trade-off parameter, which is treated as a constant for each group of samples generated for a prompt x.

According to the Policy Gradient Theorem, the true gradient of this objective is:

$$\nabla_\theta J(\theta) = \mathbb{E}_{x,y}\left[\nabla_\theta \log \pi_\theta(y|x) \cdot r_{\text{total}}(x,y)\right]. \tag{18}$$

To reduce variance, one can introduce a baseline $B(x)$ that depends on the prompt x but not the specific response y. The gradient estimate remains unbiased because the expectation of the baseline term is zero:

$$\begin{aligned}
\mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[\nabla_\theta \log \pi_\theta(y|x)\cdot B(x)\right] &= B(x)\cdot \mathbb{E}_{y\sim\pi_\theta(\cdot|x)}\left[\nabla_\theta \log \pi_\theta(y|x)\right] \\
&= B(x)\cdot \int_y \pi_\theta(y|x)\nabla_\theta \log \pi_\theta(y|x)\, dy \\
&= B(x)\cdot \int_y \pi_\theta(y|x)\frac{\nabla_\theta \pi_\theta(y|x)}{\pi_\theta(y|x)}\, dy \\
&= B(x)\cdot \int_y \nabla_\theta \pi_\theta(y|x)\, dy \\
&= B(x)\cdot \nabla_\theta \int_y \pi_\theta(y|x)\, dy \\
&= B(x)\cdot \nabla_\theta(1) \\
&= 0
\end{aligned} \tag{19}$$

Thus, the true gradient can be equivalently written as:

$$\nabla_\theta J(\theta) = \mathbb{E}_{x,y}\left[\nabla_\theta \log \pi_\theta(y|x) \cdot (r_{\text{total}}(x,y) - B(x))\right]. \tag{20}$$

The term $A(x,y) = r_{\text{total}}(x,y) - B(x)$ is the advantage function. Our Advantage Weighting method constructs a specific baseline for this multi-component reward setting. For a group of N responses $\{y_i\}_{i=1}^N$ to a prompt x, we define the empirical means $\mu_{\text{outcome}}(x)$ and $\mu_p(x)$. We then construct the baseline as:

$$B(x) = \mu_{\text{outcome}}(x) - \alpha'(x, \pi_\theta) \cdot \mu_p(x). \tag{21}$$

Since this baseline only depends on group-level statistics for a given prompt x, it is a valid baseline. Substituting this into Eq. (20), the advantage for a sample y_i becomes:

$$\begin{aligned} A(x, y_i) &= (r_{\text{outcome},i} - \alpha' p_i) - (\mu_{\text{outcome}} - \alpha' \mu_p) \\ &= (r_{\text{outcome},i} - \mu_{\text{outcome}}) - \alpha'(x, \pi_\theta) \cdot (p_i - \mu_p). \end{aligned} \tag{22}$$

This is precisely the structure of the advantage used in our proposed Advantage Weighting method, it is only that in our practice, we adopt GRPO-style normalization to divide the two terms in the advantage with their standard deviation respectively. The stochastic gradient estimator $\hat{g}_i(\theta) = \nabla_\theta \log \pi_\theta(y_i|x) \cdot A(x, y_i)$ is an *unbiased estimator* of the true policy gradient $\nabla_\theta J(\theta)$. While the core structure of our method is unbiased, our practical implementation follows the common practice of GRPO and normalizes each advantage component by its standard deviation. This final normalization step introduces a known bias to the gradient's magnitude, but follows the more common practice in LLM RL. One can simply remove the standard deviation denominator in Eq. (8) to obtain an unbiased estimator.

D Training Details

For all baseline methods, we follow the hyperparameter settings reported in their original implementations. For our RL-based methods, in the rollout phase, we set the number of rollouts to 8, with a top-p value of 0.95, a temperature of 0.6, and a maximum response length of 8192 tokens. During the training phase, we set α_{base} in Eq. (4) to 0.5, half-cycle of Cyclical Compression Pressure to 100, kl loss coefficient to 0.001, the learning rate to 1e-6, and the batch size to 128.

E Evaluation Detailes

This section of the appendix provides details regarding the evaluation model.

Parameters used for Evaluation: During the evaluation, we employed a temperature of 0.6, a top-p value of 0.95, and a maximum response length of 32,768.

Sample Count for Different Datasets: For the MATH 500, AMC 2023, Olympiad, and Minerva datasets, we adopted a sample count of 10; for the AIME 2024 dataset, we adopted a sample count of 32.

Method of Calculating Pass@1, and Token: For each question, we considered the average accuracy of every sample as Pass@1, the average response length of every sample as Token.

Prompt Used in Evaluation: We utilized the prompt "<Question> Let's think step by step and output the final answer within \\boxed{ }" during the evaluation.

Inference Scaling: During the Inference Scaling evaluation of mathematical problems, we utilized Python's `sympy`[3] module to ascertain the equivalence of two mathematical formulas in LaTeX format. We group the responses with equivalent answers, and select the largest group as the majority voting result. We adopted the LaTeX mathematical formula that appeared most frequently in the k samples after transformation by sympy as the result of Majority Voting. The Inference Scaling Accuracy was computed based on whether the Majority Voting result was equivalent to the Ground Truth.

[3] `https://www.sympy.org/`

Figure 7: Evolution of qualitative characteristics during DIET training, showing the average rate at which model checkpoints satisfy each property better than the Base Model.

F Qualitative Analysis of Behavioral Changes During Training

To understand how our DIET training qualitatively refines the model's verbosity and reasoning style beyond aggregate token counts, we conducted a case study. We aimed to assess specific behavioral changes related to token reduction by defining four qualitative metrics:

1. **Reduced Unnecessary Self-Doubt:** The model exhibits less hesitation or redundant self-correction once a correct reasoning path is identified.
2. **Reduced Post-Solution Exploration:** The model curtails exploration of alternative methods or further elaboration after a correct answer has already been found.
3. **Improved Language Structure:** The model's output is more organized and flows logically, with fewer digressions or poorly structured sentences.
4. **Concise Calculation Process:** Mathematical or logical steps are presented more directly and with less intermediate clutter.

For this analysis, we use questions from AMC 2023, AIME 2024, and MATH 500, and compare responses from the base model against those from various checkpoints of our DIET model during its training process. To evaluate the relative improvement on the aforementioned characteristics, we utilized Gemini 2.5 Pro as a judge to determine if each checkpoint response demonstrated the targeted behavior when compared with response from the base model.

Fig. 7 illustrates the average "Property Satisfaction Rate" for these four characteristics as training progresses. Each curve represents the proportion of cases where the DIET checkpoint was judged superior to the Base Model for that specific characteristic, averaged across datasets.

The trends in Fig. 7 indicate that as DIET training advances, the model progressively improves across these qualitative dimensions. We observe a clear learning curve where the model becomes more adept at producing concise and well-structured language. Notably, the "Reduced Self-Doubt" characteristic shows the most significant impact, with satisfaction rates reaching approximately 90% by the later stages of training. "Structured Language" and "Concise Calculation" also demonstrate substantial gains, with satisfaction rates plateauing around 85% and 83%, respectively. "Less Exploration" after finding a solution also improves steadily, reaching around 80%. These qualitative improvements suggest how DIET shortens the reasoning trajectory.

G Ablation Studies

G.1 Impact of Cyclical Compression Pressure

We analyze the effect of the cyclical compression pressure strategy, detailed in §3.3, by comparing our primary difficulty-aware approaches: Adaptive Weighting, Dynamic Target, and our combined DIET method with and without this temporal modulation.

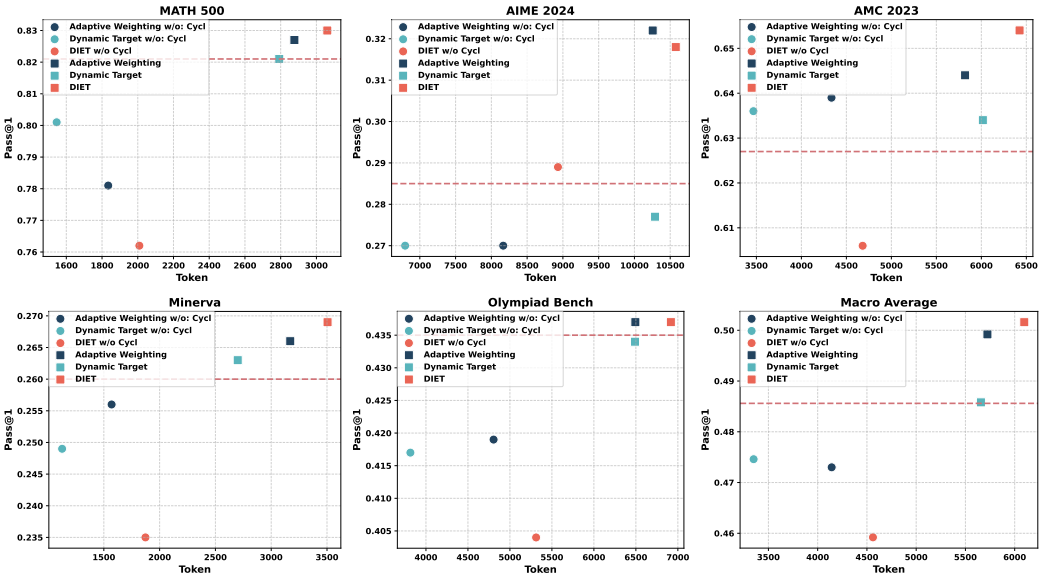

Figure 8: Pass@1 versus average token count for Adaptive Weighting, Dynamic Target, and DIET configurations with and without cyclical compression, across various benchmarks and their Macro Average. The dashed red line indicates the Base Model's Pass@1 performance. Cyclical training generally trades a slight increase in tokens for improved Pass@1.

Table 2: Macro Average Pass@1 (%) and Token Count for the **Dynamic Target** method with varying trade-off parameter α_{base}.

α_{base}	MATH 500		AIME 2024		AMC 2023		Olympiad.		Minerva		Macro Average	
	P@1	Tok	P@1	Tok	P@1	Tok	P@1	Tok	P@1	Tok	P@1	Tok
0.1	83.1	2616	29.7	8852	65.1	5198	35.0	10326	27.1	2951	48.0	5988
0.5	80.1	1546	27.0	6794	63.6	3467	41.7	3814	24.9	1126	47.4	3349
1.0	73.5	1298	16.4	4217	49.5	1966	34.6	2475	22.2	1016	39.2	2194

Fig. 8 illustrates these comparisons. Consistently across the benchmarks, particularly in the Macro Average (bottom right), applying cyclical compression shifts the methods to a higher Pass@1 compared to their non-cyclical counterparts. This performance improvement often allows the cyclical variants to meet or exceed the Base Model's Pass@1 (dashed red line).

This gain in reasoning accuracy typically corresponds to a moderate increase in average token length. For example, in the Macro Average plot, the DIET configuration achieves the highest Pass@1, while DIET w/o Cycl uses fewer tokens but results in the lowest Pass@1. This pattern suggests that while non-cyclical versions offer more aggressive token reduction, the "relax" phases in cyclical training are beneficial for achieving peak performance, justifying its inclusion in our best-performing DIET configurations presented in Table 1. Thus, cyclical compression aids our difficulty-aware methods in achieving a superior balance of high performance and significant token savings relative to the Base Model.

G.2 Impact of Trade-off Parameter α_{base}

To determine an appropriate value for the trade-off parameter α_{base} in Eq. (1), we conducted an ablation study on the Dynamic Target as a preliminary experiment. This preliminary experiment aimed to find a balance between maintaining reasoning performance and achieving significant token reduction. We tested $\alpha_{base} \in \{0.1, 0.5, 1.0\}$, with results shown in Table 2.

As shown in Table 2, increasing α_{base} leads to more aggressive token reduction but also a corresponding decrease in Pass@1 performance. Specifically, on Macro Average, increasing α_{base} from 0.1 to 1.0 reduces tokens significantly from 5988 to 2194, but Pass@1 drops from 48.0% to 39.2%. The

Table 3: Macro Average Pass@1 (%) and Token Count for the ablation of GRPO-style normalization

	MATH 500		AIME 2024		AMC 2023		Olympiad.		Minerva		Macro Average	
	P@1	Tok	P@1	Tok	P@1	Tok	P@1	Tok	P@1	Tok	P@1	Tok
w Norm	83.0	3061	31.8	10578	64.5	6425	43.7	6917	26.9	3505	50.2	6097
w/o Norm	82.5	2856	29.0	9994	66.5	5724	43.8	6384	26.9	3070	49.7	5605

Table 4: Macro Average Pass@1 (%) and Token Count for the **Dynamic Target** method with varying rollout parameter N.

N	MATH 500		AIME 2024		AMC 2023		Olympiad.		Minerva		Macro Average	
	P@1	Tok	P@1	Tok	P@1	Tok	P@1	Tok	P@1	Tok	P@1	Tok
4	78.1	2104	29.1	8709	63.8	4993	42.7	5298	24.8	2270	47.7	4674
8	83.0	3061	31.8	10578	64.5	6425	43.7	6917	26.9	3505	50.2	6097
16	82.2	2613	28.3	9899	66.6	5328	44.0	5970	26.3	2834	49.5	5328

setting of $\alpha_{base} = 0.5$ yields a satisfactory trade-off. Based on this balance, we selected $\alpha_{base} = 0.5$ as the default for all the difficulty-aware training in our main experiments.

G.3 Impact of Normalization Within Advantage Weighting

To test the importance of normalizing the penalty term within our Advantage Weighting framework, we ran an ablation where we removed the GRPO-style normalization from the penalty advantage. The results are shown on Table 3.

As the results show, removing normalization leads to a substantial token reduction but also causes a significant degradation in performance. Some careful tuning of the weighting parameter might mitigate the issue, but normalization is an effective and simple approach. Therefore, we adopt the normalization. We hypothesize that the un-normalized penalty can have extreme values, which overly biases the policy gradient towards generating shorter sequences at the expense of correctness.

G.4 Impact of Rollout Parameter N

N is an important hyperparameter of GRPO, to verify its impact on DIET, we conducted an ablation study, and the results are shown Table 4.

The results show a clear trend. Decreasing the sample count to N=4 results in a noticeable drop in performance. We attribute this to the less stable estimation of problem difficulty and reward baselines, which can introduce noise into the policy gradient updates. Increasing the sample count from N=8 to N=16 does not yield further performance improvements. However, it lowers the number of tokens, potentially because the difficulty estimation is more accurate, allowing for more token reduction for problems with suitable difficulty.Considering the trade-off between performance and efficiency, we select N=8 in our main experiments.

G.5 Impact of Dynamic Length Target Paramter L_{max} and δ

We also conducted ablation study for L_{max} and δ, and the results are shown in Table 5 and Table 6, respectively.

Reducing L_{max} decreases average response length with modest performance degradation, similar to decreasing δ. We chose the current configuration as the optimal balance between performance and token efficiency.

H Scale to Larger Models

To verify the scalability of the DIET, we also conducted experiments on the R1-Distilled Qwen 7B. The experimental results are shown in Table 7. From the results, we draw two key conclusions:

Table 5: Macro Average Pass@1 (%) and Token Count for the **Dynamic Target** method with varying L_{max}.

L_{max}	MATH 500		AIME 2024		AMC 2023		Olympiad.		Minerva		Macro Average	
	P@1	Tok	P@1	Tok	P@1	Tok	P@1	Tok	P@1	Tok	P@1	Tok
2048	82.9	2851	28.8	9762	64.8	5597	43.0	5742	29.0	2973	49.7	5385
8192	83.0	3061	31.8	10578	64.5	6425	43.7	6917	26.9	3505	50.2	6097

Table 6: Macro Average Pass@1 (%) and Token Count for the **Dynamic Target** method with varying δ.

δ	MATH 500		AIME 2024		AMC 2023		Olympiad.		Minerva		Macro Average	
	P@1	Tok	P@1	Tok	P@1	Tok	P@1	Tok	P@1	Tok	P@1	Tok
0.05	82.6	2903	28.3	7630	65.0	5907	43.4	6710	26.3	3303	49.1	5290
0.1	83.0	3061	31.8	10578	64.5	6425	43.7	6917	26.9	3505	50.2	6097

DIET Improves Performance while Reducing Tokens: Our DIET framework achieves the highest average Pass@1 score (65.4%), improving upon the already strong 7B base model (64.4%). Crucially, it simultaneously reduces the average token count This demonstrates that our method effectively enhances both performance and efficiency at a larger scale.

Superior Performance-Efficiency Trade-off: Most other compression methods exhibit a clear trade-off, where significant token reduction leads to a noticeable drop in performance. DIET breaks this trend.

I Functional Form of Dynamic Length Target

As illustrated in Fig. 2, the relationship between difficulty and token count follows a more logarithmic trend rather than a linear one. To explore this, we conducted this experiment. We first fitted a logarithmic curve to the data in Fig. 2, then we scaled and moved the curve to ranges from 0 to 8192 when x is in [0, 1], resulting in $y = 3116.9 * ln(21.33x + 1.66) - 1579.7$. We then used this function to sample target lengths in our Dynamic Target method, instead of sampling from a linear range. The experimental results are shown in Table 8.

The results show that the method is robust to the selection of the target function, the two choices do not differ significantly, and the simple linear function already works well.

Table 7: Average performance and token length of R1-Distilled Qwen 7B trained with RL-based methods

Method	MATH 500		AIME 2024		AMC 2023		Olympiad.		Minerva		Macro Average	
	P@1	Tok	P@1	Tok	P@1	Tok	P@1	Tok	P@1	Tok	P@1	Tok
Base Model	92.1	3921	53.5	13389	82.5	7730	56.4	8890	37.4	4930	64.4	7772
RL-Based												
CosFn	87.1	1397	50.5	8659	78.3	3578	54.0	4593	36.4	1345	61.3	3914
O1-Pruner	71.1	4958	39.2	12054	79.1	6014	50.9	6852	30.5	5277	54.2	7031
Kimi 1.5 RL	64.0	**1124**	44.3	**7903**	70.6	**2783**	46.1	**3225**	32.5	**677**	51.5	**3142**
Our Difficulty-Aware Methods												
Dynamic Target (§3.1.2)	90.4	2121	49.6	8186	79.8	4416	54.3	4677	34.5	1819	61.7	4244
Adaptive Weighting (§3.1.1)	90.6	1782	52.1	8313	80.2	4050	54.9	4838	36.9	1638	62.9	4124
DIET (§3.1.1+§3.1.2)	**92.1**	3187	**57.9**	10124	**82.6**	6075	**56.5**	7026	**37.9**	3695	**65.4**	6021

Table 8: Macro Average Pass@1 (%) and Token Count for different length penalty functions of **Dynamic Target**

	MATH 500		AIME 2024		AMC 2023		Olympiad.		Minerva		Macro Average	
	P@1	Tok	P@1	Tok	P@1	Tok	P@1	Tok	P@1	Tok	P@1	Tok
Logarithmic	83.5	2971	29.8	10890	61.5	6526	43.7	6693	26.5	3319	49.0	6079
Linear	83.0	3061	31.8	10578	64.5	6425	43.7	6917	26.9	3505	50.2	6097

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: We have outlined our main contributions in a point-by-point manner in both the abstract and the instruction sections.

   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: In Appendix A, we provide an analysis of the current limitations of our work.

   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: [Yes]

   Justification: We provide a derivation of advantage distortion under naive reward weighting in Appendix B

   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 provide test details in Appendix E. We will also open-source our evaluation code after proper organization.

   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: [No]

   Justification: We will open-source our code after proper organization.

   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, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We provide detailed training details in Appendix D and detailed evaluation details in Appendix E.

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 conducted a p-value test when computing the Pearson correlation between problem difficulty and average response length and ensured that the p-value is less than 0.01. And we reported error bars in Fig. 2 .

   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: We detail the computational resources used in our experiments in §4.1.

   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: Our research adheres to the NeurIPS Code of Ethics. It does not involve human subjects, personally identifiable information, or sensitive data.

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: Our experiments are conducted using the already widely-used open-source model, which will not induce new societal impact.

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: We use open-source reasoning model for our experiments, which will pose no such risks.

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: All external assets used in our work, including code(e.g., verl, sympy), datasets(e.g., deepscaler), and models(e.g. R1-Distilled Qwen), are properly credited. We ensure that the licenses and terms of use for these resources are explicitly stated and strictly followed.

    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 will open-source our code and provide detailed documentation and comments for ease of use.

    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: Crowdsourcing and human subjects are not involved in our research.

    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: Crowdsourcing and human subjects are not involved in our research.

    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: We only used LLM for writing and editing which does not impact the core methodology.

    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.