Too Long, Do Re-weighting for Efficient LLM Reasoning Compression

Anonymous ACL submission

Abstract

Large Language Models (LLMs) have recently achieved remarkable progress on complex reasoning tasks by leveraging extended Chain-of-Thought (CoT) techniques. These reasoning processes can be roughly categorized into System-1 (fast and intuitive) and System-2 (slow and deliberate) paradigms. However, excessive reliance on lengthy System-2-style reasoning during inference can produce extremely long outputs, thereby reducing efficiency. In this work, we propose Thinking Length Data Re-weighting (TLDR), that does not rely on sophisticated data annotations or interpolation between multiple models. We continuously balance the weights between the model's System-1 and System-2 data to eliminate redundant reasoning processes while preserving the model's reasoning capability. We validate our method across multiple base models, including Deepseek-R1-Distilled Qwen models, as well as on a diverse benchmarks with varying difficulty levels. Our method significantly reduces the number of output tokens by nearly 40% while maintaining the accuracy of the reasoning. Our code and data are at link: https://anonymous.4open.science/r/TLDR_Review-BBE5/.

1 Introduction

Recent efforts have developed reasoning-oriented Large Language Models (LLMs) capable of solving complex reasoning tasks. Earlier large language models (Llama-3 Team, 2024; Yang et al., 2024) largely rely on System-1 reasoning to generate, yet tend to be less effective on tasks requiring sustained, multi-step reasoning. In contrast, models exhibiting more System-2 reasoning behavior, such as DeepSeek-R1 (DeepSeek-AI, 2025), extend reasoning chains to encourage deliberate thinking through iterative self-assessment, error correction, and verification, albeit at the cost of increased redundancy and much higher computational overhead.

However, reasoning LLMs often over-deliberate even on simple problems (Chen et al., 2025b; Wang et al., 2025), resulting in unnecessary exploration and planning that deteriorate the efficiency and practicality. To mitigate this issue, two types of approaches have been explored: *training-free methods* (Xu et al., 2025b; Yao et al., 2025; Han et al., 2024) and *training-based methods*. Some training-free methods regulate internal states during inference—for example, via prompting or confidence-based output compression (Chen et al., 2025a; Azizi et al., 2025). Some training-free methods based on architectural or parametric interventions, such as model merging (Wu et al., 2025; Kimi-1.5 Team, 2025), modify the LLM's weights or decoding strategies to achieve conciseness. Although these methods offer immediate efficiency gains by constraining the inference process, they do not alter the model parameters or its intrinsic reasoning capabilities, and their performance is sensitive to both parameter settings and model choice.

On the other hand, *training-based methods* improve reasoning conciseness via generation path optimization. These methods primarily focus on sampling and synthesizing relatively concise reasoning paths through various strategies. By performing reinforcement learning (RL) (Meng et al., 2024; Hou et al., 2025; Luo et al., 2025b; Aggarwal and Welleck, 2025) or supervised fine-tuning (SFT) (Chen et al., 2025b), models learn to generate more concise yet correct reasoning paths. RL-based methods (Aggarwal and Welleck, 2025; Hou et al., 2025) yield more accurate and concise solutions by penalizing redundancy in System-2 reasoning. However, the efficiency gain is at the cost of a computationally intensive training phase; specifically, achieving an optimal Pareto front between reasoning performance and inference efficiency necessitates a vast number of optimization iterations. SFT-based methods, despite their effectiveness, typically require careful collection of problems and precise control of the data ratio for different lengths

to achieve good results, incurring the cost of complex data construction and parameter tuning. For example, TOPS (Yang et al., 2025) requires preprocessing steps to manually label SFT data to construct length-sensitive models, while CoT-Valve (Ma et al., 2025) generates data by creating intermediate models through model interpolation for sampling. This construction process is often tedious (Yang et al., 2025), or challenging to maintain high quality (Ma et al., 2025).

To optimize the trade-off between reasoning accuracy and token efficiency, we conduct an empirical study to evaluate how fine-tuning with various CoT styles across different problem difficulties affects model performance. As illustrated in Figure 1, our findings reveal that while short CoTs on simple tasks can effectively simulate System-1 intuition, long CoT sequences on complex problems are indispensable for sustaining performance and eliciting System-2 reasoning capabilities. Motivated by these findings, we propose a dynamic **T**hinking **L**ength **D**ata **R**e-Weighting (**TLDR**), which dynamically balances the model's complex reasoning using long CoT and efficient reasoning using short CoT data, enabling the model to eliminate redundant cognitive processes. For training, we construct mixed datasets comprising System-1 short CoT for simple problems and System-2 long CoT for complex reasoning. The model is guided to dynamically recalibrate the data ratio after each cycle of reasoning compression. We evaluate the utility of System-1 data via an efficiency metric and System-2 data via an accuracy metric, balancing between inference speed and reasoning depth.

Compared to various methods requiring fine-tuning data with different reasoning lengths (Ma et al., 2025; Yang et al., 2025), our method enables dynamic learning by utilizing the self-sampled long CoT model and the short CoT data constructed by the original instruct/base model. Among the training-based methods, our method achieves shorter training time and higher efficiency than ThinkPrune and L1, while also delivering superior performance. Through experiments on DeepSeek-Distill-7B/14B, our model achieves high compression ratio while maintaining reasoning accuracy on the DeepSeek Distilled Qwen-7B/14B models.

Our main contributions are summarized as follows: (1) We empirically analyze the interaction between CoT length and problem difficulty, showing that short CoT supports efficient System-1 reasoning on simple tasks, while long CoT is crucial for System-2 reasoning on complex problems. (2) We propose **TLDR**, a dynamic training framework that adaptively balances short and long CoT data to reduce redundant reasoning without sacrificing accuracy. (3) TLDR avoids manually curated length-sensitive data and heavy reinforcement learning by combining self-sampled long CoT with short CoT from base models, achieving higher reasoning compression while maintaining strong performance on DeepSeek-Distill-7B/14B.

2 Related Work

2.1 Efficient System-2 Reasoning

Despite the enhanced generalization of System 2 reasoning, the auto-regressive nature of LLMs imposes a heavy computational burden (Chen et al., 2025b; Wang et al., 2025). To improve efficiency, recent methods have introduced *adaptive reasoning budgets* to dynamically allocate resources. Within this category, training-free methods like CoD and TALE-EP (Xu et al., 2025b; Han et al., 2024) impose budget constraints, while budget-sensitive models—such as L1, TOPS, O1-Pruner, and Kimi-k1.5 (Aggarwal and Welleck, 2025; Yang et al., 2025; Luo et al., 2025b; Kimi-1.5 Team, 2025)—incorporate length penalties during post-training. Some work (Ma et al., 2025; Jiang et al., 2025; Yu et al., 2025) synthesizes diverse-length CoT data, while TOPS (Yang et al., 2025) samples budget-sensitive versions using a data model, and C3oT (Kang et al., 2024) compresses original LLM output. While prior studies (Yang et al., 2025; Ma et al., 2025) developed multi-length CoT datasets for adaptive reasoning, the scaling effects of reasoning-chain length on the accuracy-efficiency trade-off remain under-explored. We bridge this gap by analyzing how training on short CoT data influences token compression and accuracy retention across mathematical benchmarks of varying difficulty.

2.2 Data Re-weight of LLM Training

Data quality and composition are pivotal throughout the pre-training and post-training phases. In pre-training, these factors are primarily managed via data filtering and re-weighting. While extensive research focuses on filtering pipelines—including language identification (Laurençon et al., 2023; PaLM Team, 2022), quality assessment (Raffel et al., 2023; Rae et al., 2022), content moderation (Xu et al., 2021; Longpre et al., 2023), and deduplication (Hernandez et al., 2022; Lee et al., 2022)—to

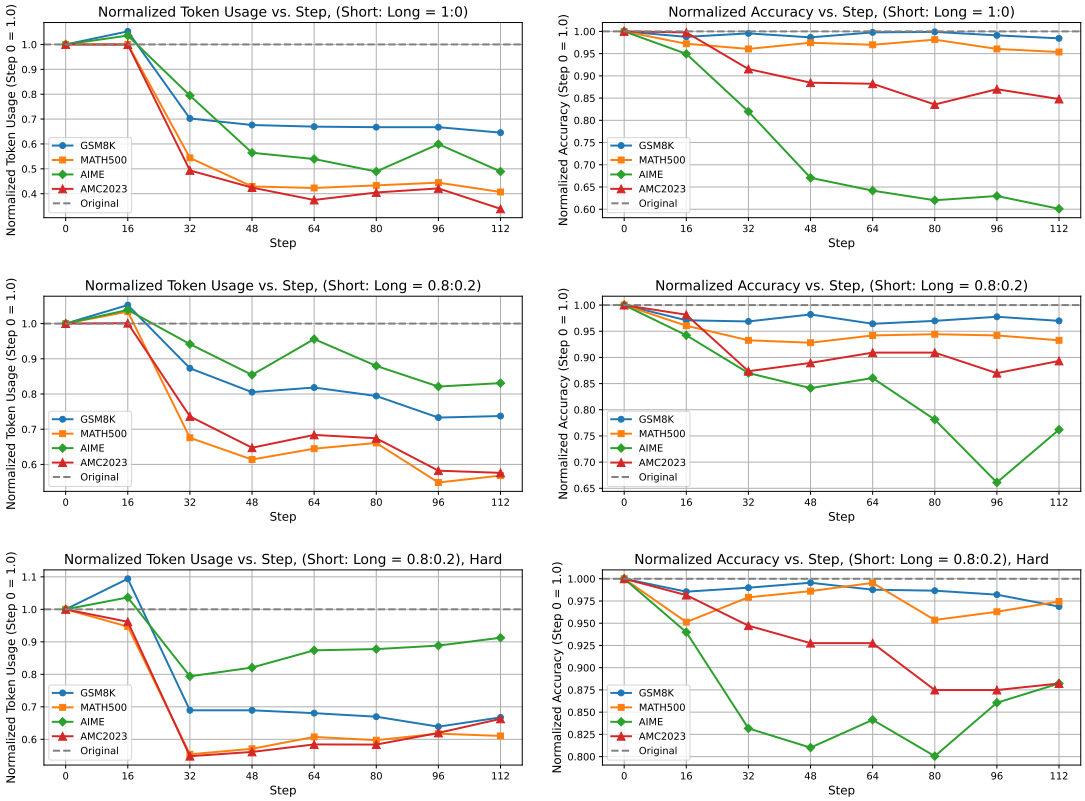


Figure 1: Empirical study on System-1/2 data composition. We assessed normalized token/accuracy usage: Len_{norm} and normalized token accuracy: Acc_{norm} on datasets of various question difficulty, spanning from GSM8K to AIME. The details of Acc_{norm} /Len_{norm} metric refer to Appendix A.

enhance training efficiency (Liu et al., 2024; Albalak et al., 2024), these static approaches often lack adaptability. Such rigidity can lead to the inadvertent removal of valuable samples (Muennighoff et al., 2023) and the introduction of systematic biases (Gururangan et al., 2022; Dodge et al., 2021). Similarly, in the post-training stage, an appropriate proportion of data with varying characteristics is crucial for optimizing final performance. For example, DeepMath-103K generates a large volume of data with evenly distributed difficulty for training (He et al., 2025). SRPO designs a dynamic sampling approach to filter out samples that are consistently answered correctly, thereby improving inference efficiency (Zhang et al., 2025). To the best of our knowledge, we are the first to introduce a re-weighting mechanism into thinking compression. By employing simple strategies to construct short and long CoT, we enable the model to dynamically compress its reasoning process.

3 Empirical Study on Data Composition for Reasoning Compression

We aim to investigate which data regimes induce rapid, intuitive reasoning (System-1) versus sustained, deliberative reasoning (System-2). Prior approaches such as CoT-Valve (Ma et al., 2025) and TOPS (Yang et al., 2025) did not distinguish problem difficulty, constructing and mixing CoTs of varying lengths for all problems indiscriminately. In contrast, we decouple problem difficulty from CoT length by pairing competition-level problems (Muennighoff et al., 2025) and elementary arithmetic problems (Cobbe et al., 2021) with either long or short CoTs. This design allows us to systematically study which data characteristics selectively elicit System-1 or System-2 reasoning.

Our findings reveal that **short CoT on simple problems data effectively enhances models with rapid and intuitive reasoning capabilities**. We define this combination of elementary problems paired with short chains of thought as **System-1 data**. We leverage concise solutions derived from simple GSM8K questions to fine-tune the model, subsequently monitoring token compression and accuracy across GSM8K, MATH500, AMC, and AIME. As illustrated in Figure 1, fine-tuning on these direct responses achieves consistent length reduction across all difficulty levels. However, this gain in efficiency entails a significant trade-off: a substantial degradation in reasoning on complex problems. Our results suggest that while such data

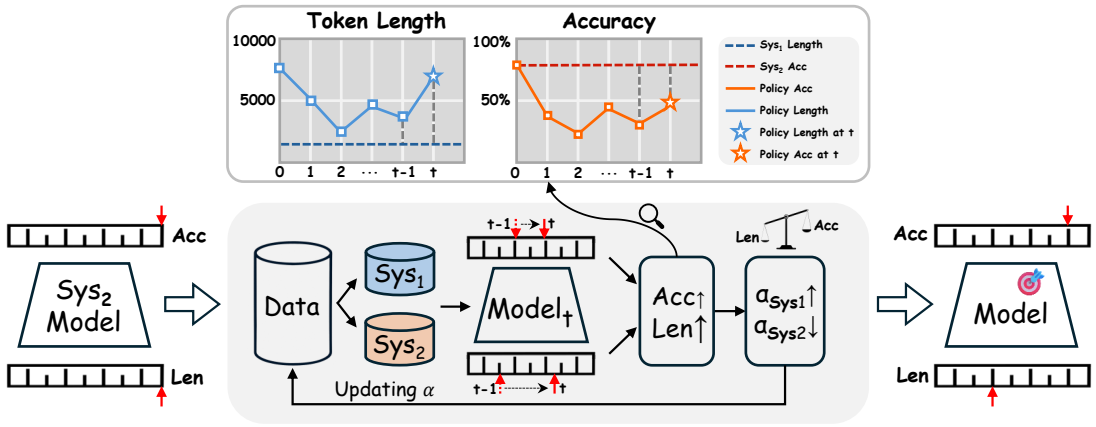


Figure 2: Overview of TLDR. Starting with a System-2 model, we iteratively update it on both System-12 samples. Their ratios are adjusted every few steps based on current average accuracy and token length from the validation set.

reinforces System-1 capabilities—rapid, intuitive processing—it significantly erodes System-2 reasoning, the slow and deliberative analytical capacity essential for solving challenging tasks. Conversely, our results demonstrate that **long CoT on hard problems is indispensable for sustaining model performance on challenging tasks**. We evaluate the trade-off between token compression and accuracy by blending s1-style long CoT from difficult problems (Muennighoff et al., 2025) with System-1 data (ratio 1 : 4). Crucially, our comparison in Figure 1 reveals that replaying long CoT from simple tasks fails to recover reasoning depth. Only long CoT originating from complex problems, which we define as **System-2 data**, effectively mitigates performance loss. This confirms that System-2 capability depends on the deliberate reasoning process triggered by difficulty, rather than merely on the length in the CoT.

Building upon the aforementioned observations, we transition from static data mixing of System-1/2 data to a more principled framework. Specifically, we propose a dynamic re-weighting framework designed to achieve a harmonious balance between inference efficiency and analytical depth. This strategy functions by selectively allocating cognitively demanding System-2 pathways to complex challenge reasoning tasks while streamlining routine tasks through efficient System-1 responses.

4 Reasoning Compression via Dynamic Re-weighting

4.1 Formalizing the Optimization Objective

The thinking compression problem is formalized as an optimization task designed to determine the ideal balance between System-1 and System-2 reasoning. By training on mixed datasets, the model is expected to converge toward the superior performance characteristics of both systems across specific evaluation metrics.

To operationalize this objective, consider a model p_θ and an input problem x. We define $\ell(y) \in \mathbb{Z}^+$ as the token length and $\mathbb{I}(y) \in \{0, 1\}$ as the correctness indicator of the generated output y. The theoretical upper bounds for System-1 and System-2 capabilities are denoted as ϕ_i^* (for $i \in \{1, 2\}$). The overall optimization objective is to minimize the gap between the current model's performance and these ideal bounds:

$$\min_{\theta, \alpha \in (0,1)} \mathcal{L}(\theta, \alpha) = \sum_{i \in 1,2} \alpha_i \cdot \underbrace{\left(\phi_i^* - \phi_i^\theta\right)}_{\delta_i}, \quad (1)$$

where ϕ_1^θ can be regarded as a metric for measuring the efficiency of the System-1 reasoning. ϕ_2^θ can be regarded as an accuracy metric. In this way, the overall optimization objective is to jointly and explicitly minimize the gap between the current model and the efficiency upper bound of System-1, as well as the reasoning capability upper bound of System-2; the objective is to optimally and coherently harmonize reasoning process.

$$\phi_1^\theta = -\mathbb{E}_{x \in D_{\text{dev}}, y \sim p_\theta(x)}\, \ell(y), \quad (2)$$

$$\phi_2^\theta = \mathbb{E}_{x \in D_{\text{dev}}, y \sim p_\theta(x)}\, \mathbb{I}(y). \quad (3)$$

4.2 Overall Pipeline of TLDR

Setup: Prepare For System-1/2 Data. To emulate the rapid and intuitive processing characteristic of System-1, we sample reasoning paths using a short CoT model, utilizing elementary arithmetic problems from the GSM8K dataset as the

Algorithm 1 Overall Pipeline of TLDR: Thinking Compression via Dynamic Re-weighting

Require: $\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_{\text{dev}}$; Total steps T; Interval T_d; Step size η; Initial θ_l (Sys-2) and θ_s (Sys-1).

// Step 1: Capability Boundary Estimation

Compute upper bounds ϕ_1^*, ϕ_2^* (Eq. (4-5)); Init System-1/2 data probability simplex $\alpha_t = [0.5, 0.5]$, when $t = 0$, And $\theta_{\text{proxy}} \leftarrow \theta_l$.

// Step 2: Dynamic Re-weighting SFT

for $t = 1$ **to** T/T_d **do**

  Data Sampling: Sample mini-batch $\mathcal{B} \sim \mathcal{D}$ with mixture ratio α_{t-1}.

  Model Update: Update θ_p for T_d steps on $\mathcal{B}$ via standard SFT loss.

  Capability Evaluation: Evaluate $\phi_1^{\theta_p}$ and $\phi_2^{\theta_p}$ on $\mathcal{D}_{\text{dev}}$.

  Weight Update: Estimate gains λ_i using Eq. (6-7); Update $\alpha_t[i] \propto \alpha_{t-1}[i] \cdot e^{-\eta\lambda_i}$.

  Checkpointing: Save θ_p as a candidate for Pareto frontier evaluation.

end for

Ensure: Checkpoint θ^* on the Pareto frontier of $\mathcal{D}_{\text{dev}}$ with optimal efficiency-accuracy trade-off.

primary task source. This process yields a dataset $\mathcal{D}_1 = \{(Q_s, C_s)\}$, comprising `<simple question, short CoT>` data pairs. Conversely, to encapsulate the deliberate and rigorous logical reasoning of the original System-2 model, we employ the uncompressed long CoT model to sample reasoning trajectories from the s1 (Muennighoff et al., 2025) prompt set. By leveraging this raw computational depth, we curate $\mathcal{D}_2 = \{(Q_h, C_l)\}$, consisting of `<hard question, long CoT>` data pairs.

Step 1: Establishing Capability Bounds. To balance reasoning depth with inference efficiency, we optimize the data mixture ratio during post-training. Our goal is to align the model θ with the theoretical upper bounds ϕ_i^*. Specifically, we define the System-2 performance bound ϕ_2^* based on the maximum accuracy of long CoT sequences, and the System-1 efficiency bound ϕ_1^* based on the minimum token length of short CoT sequences:

$$\phi_1^* = -\mathbb{E}_{x\sim\mathcal{D}_{\text{dev}}}\, \ell(y_s), \tag{4}$$

$$\phi_2^* = \mathbb{E}_{x\sim\mathcal{D}_{\text{dev}}}\, \mathbb{I}(y_l), \tag{5}$$

where y_s and y_l denote the outputs from the short and long CoT systems, respectively.

Step 2: Reasoning Compression via Dynamic Re-weighting. We initialize the SFT process with an equal data ratio α_0 for System-1 and System-2 datasets. To adaptively minimize the gap δ_i defined in Eq. (1), we implement a periodic reweighting mechanism every T_d steps. We update the sampling simplex α using the Exponential Gradient method (see Appendix B.1 for formal derivation). Specifically, we estimate the potential gains for each system, λ_1 and λ_2, by evaluating the current model θ_p against the established bounds:

$$\lambda_1 = \max\left(\frac{\phi_1^* - \phi_1^{\theta_p}}{\phi_1^{\theta_s} - \phi_1^{\theta_l}},\ 0\right), \tag{6}$$

$$\lambda_2 = \max\left(\frac{\phi_2^* - \phi_2^{\theta_p}}{\phi_2^{\theta_l} - \phi_2^{\theta_s}},\ 0\right). \tag{7}$$

These gains reflect the normalized distance to the target performance relative to the margin between specialized system proxies. This dynamic adjustment allows the model to adaptively prioritize reasoning depth or efficiency based on real-time validation feedback, as illustrated in Algorithm 1, which describes computational workflow. During the final model selection, we adopt an efficiency-first Pareto optimality criterion: we select the checkpoint that minimizes average output length among all candidates that maintain baseline System-2 accuracy on the validation set. As shown in Figure 2, this dynamic reweighting enables the model to compress its reasoning trajectories—producing significantly fewer tokens—while preserving the original logical rigor and accuracy.

5 Experiments

Datasets and Metrics. Following prior efforts, we evaluate TLDR on several widely-used benchmarks that span a broad range of difficulty levels, including ASDiv (Miao et al., 2021), GSM8K (Cobbe et al., 2021), MATH-500 (Hendrycks et al., 2021), AIME2024 (AI-MO Team, 2024a), AMC (AI-MO Team, 2024b) and MinervaMath (Lewkowycz et al., 2022) in Table 1. To ensure the stability of the evaluation, we performed multiple samplings for each

Model	Accuracy							Generation Length							P.D.R.
	ASDiv	GSM8K	MATH	AIME	AMC	Minerva	**Avg.**	ASDiv	GSM8K	MATH	AIME	AMC	Minerva	**A.C.R.**	
DS-Qwen-7B Models															
Original	86.8	89.4	86.8	42.9	81.5	46.0	**72.2**	769	554	2861	6820	4510	3347	**0%**	0%
TALE-EP	80.4	89.1	84.3	40.0	80.0	42.3	**69.3**	509	450	1994	6520	3892	2242	**22.3%**	**21.4%**
ConciseCoT	86.0	89.5	86.2	41.7	79.6	46.0	**71.5**	532	457	2330	6587	4245	3347	**12.7%**	**12.5%**
Avg-Merging	92.8	70.1	58.6	0.05	39.6	29.8	**48.4**	622	8552	8540	8501	8542	8544	**3.2%**	**2.14%**
Ties-Merging	74.4	69.7	59.8	13.6	42.5	23.2	**47.2**	1114	2475	4086	6767	5195	4306	**0.1%**	**0.06%**
TD-Merging	75.9	72.3	65.4	14.6	45.6	24.3	**49.6**	1036	2073	2934	5483	3698	2938	**8.3%**	**5.7%**
Seal	86.8	89.4	89.4	43.3	77.8	47.8	**72.4**	591	773	2661	6871	4740	3413	**5.1%**	**5.1%**
Overthink	86.6	89.6	87.2	38.7	79.6	45.2	**71.1**	773	555	2898	6766	4558	3407	**0.1%**	**0.09%**
ThinkPrune	90.6	92.1	91.0	43.3	86.2	45.6	**74.8**	653	587	2379	6207	3739	2762	**12.6%**	**12.6%**
CoT-Valve	59.4	88.4	84.2	41.2	80.6	41.9	**65.9**	140	514	2144	6397	4278	2172	**26.8%**	**24.4%**
TLDR	93.0	87.7	87.4	41.2	83.1	41.0	**72.3**	147	253	1556	6368	3386	1451	**44.9%**	**44.9%**
DS-Qwen-14B Models															
Original	80.5	92.5	86.4	43.4	79.6	48.2	**71.7**	476	679	2951	6701	4584	3270	**0%**	0%
TALE-EP	77.5	92.4	85.4	49.2	80.3	50.0	**72.5**	369	555	2248	6551	4179	2731	**15.4%**	**15.4%**
ConciseCoT	74.0	92.4	85.6	51.6	82.3	47.1	**72.2**	369	555	2066	6267	3878	2605	**18.8%**	**18.8%**
Avg-Merging	94.8	90.3	73.0	10.8	55.0	44.1	**61.3**	167	366	5158	6364	5668	1084	**30.5%**	**26.1%**
Ties-Merging	79.6	91.3	82.6	25.4	72.5	37.1	**64.8**	242	542	1919	5913	3158	1850	**31.8%**	**28.7%**
TD-Merging	80.7	91.8	84.8	25.4	75.3	34.9	**65.4**	274	467	1870	5747	3182	1877	**33.0%**	**30.1%**
Seal	78.2	90.4	84.8	41.2	78.5	47.5	**70.1**	452	465	2193	6671	4032	2911	**14.3%**	**13.9%**
Overthink	79.3	92.3	88.0	45.8	82.8	45.6	**72.3**	451	679	2893	6700	4464	3715	**1.6%**	**1.3%**
ThinkPrune	80.6	93.7	89.0	50.8	88.7	50.7	**75.6**	379	563	2177	5778	3327	2234	**22.8%**	**22.8%**
CoT-Valve	72.9	92.0	87.0	45.0	83.5	47.8	**71.4**	204	576	2652	6686	4392	2833	**16.7%**	**16.6%**
TLDR	88.0	90.9	86.6	43.3	83.8	48.7	**73.5**	158	240	2092	6403	3839	2177	**35.8%**	**35.8%**

Table 1: Performance comparison of TLDR with baselines. The accuracy is measured by sampling multiple responses from the LLMs and taking the average to reduce variance. A.C.R means the token compression ratio computed by Eq. (8). P.D.Ratio are compression ratio that incorporates accuracy metrics, as defined in Eq. (9).

dataset and took the average accuracy. For GSM8K, MATH-500, and MinervaMath, we sampled each question 4 times and took the average accuracy of the 4 solutions. For AIME24 and AMC23, we sampled each problem 8 times and took the average accuracy of the 8 solutions. The token count was calculated using the corresponding tokenizer of the language model.

We evaluate TLDR and other baseline methods by jointly considering accuracy preservation and compression ratio. The Average Compression Ratio (A.C.R) measures the reduction in response length relative to the original output in N benchmarks, and is formally defined as:

$$\text{A.C.R.} = \frac{1}{N}\sum_{i=1}^{N}\max\left(\frac{L^i_{\text{orig}} - L^i_{\text{curr}}}{L^i_{\text{orig}}}, 0\right), \quad (8)$$

where L_{orig} and L_{curr} denote the token counts of the original and the processed responses, respectively. Then, the P.D.R., Performance-Discounted Compression Ratio, across a set of N benchmarks is calculated as:

$$\text{P.D.R.} = \max(\frac{\mathcal{A}_{\text{curr}}}{\mathcal{A}_{\text{orig}}}, 1) \times \text{A.C.R.}, \quad (9)$$

where $\mathcal{A}_{\text{curr}}$ and $\mathcal{A}_{\text{orig}}$ is the average accuracy of current model and original model. P.D.R. is used as an integrated metric that simultaneously accounts for accuracy and compression efficiency.

Baselines. We compared two types of baselines: *training-free methods* and *training-based methods*. We include three categories of training-free baselines as discussed in Section 2.1: prompt-based methods, model-merging–based methods, and steering-based methods. For prompt-based methods, we adopt TALE-EP (Han et al., 2024) and ConciseCoT (Lee et al., 2025), both of which encourage the model to generate correct solutions in a more concise manner by explicitly constraining or guiding the reasoning process through prompts. For model-merging–based methods, we consider Average Merging (Avg-Merging), Ties-Merging, and Ties-DARE Merging (TD-Merging), as introduced in the Kimi-k1.5 Technical Report (Kimi-1.5 Team, 2025) and the Long2Short Technical Report (Wu et al., 2025). These methods combine multiple pretrained models with different characteristics into a single model without additional training. For steering-based methods, we adopt Seal (Chen et al., 2025a), which extracts hidden states from

Model	Accuracy					Generation Length				
	GSM8K	MATH	AIME	AMC	Avg.	GSM8K	MATH	AIME	AMC	A.C.R.↑
DS-7B	89.4	86.8	42.9	81.5	75.2	554	2861	6820	4510	–
Compression by SFT on Static Dataset										
Direct-Mixture	87.1	84.8	39.7	73.1	**71.2**	236	1221	5322	2560	44.98%
CoT-Valve	88.4	84.2	41.2	80.6	**73.6**	514	2144	6397	4278	10.91%
TOPS	85.9	89.4	43.3	77.8	**74.1**	336	2145	7024	4378	16.08%
Compression by Simple Curriculum Data Schedule										
L->S	86.9	83.0	39.5	76.8	**71.5**	231	1335	5684	2841	41.33%
S->L	84.0	82.0	41.2	80.0	**71.8**	266	1766	6729	3968	25.90%
Random	84.0	81.0	37.5	79.6	**70.5**	246	1416	5828	2925	38.92%
Ours										
TLDR	87.7	87.4	41.2	83.1	**74.9**	253	1556	6368	3386	38.95%

Table 2: Performance comparison of baseline methods: static data compression methods and curriculum learning data mixing methods. Direct-Mixture denotes directly mixing our System-1/2 data.

Method	Time ↓	Avg. Acc ↑	A.C.R ↑
TLDR (Ours)	**88.1**	72.3	**44.9%**
TOPS	201.6	67.1	24.4%
CoT-Valve	240.6	65.9	26.8%
L1	640.3	–	–
ThinkPrune	704.1	74.8	22.8%

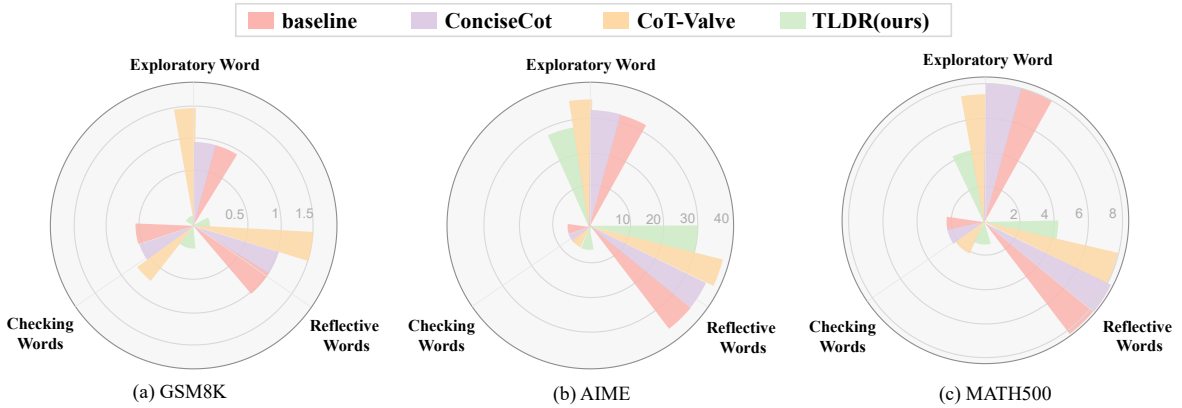
Table 3: Training time comparison (GPU Hours) on A100 GPUs. TLDR significantly reduces training costs while maintaining competitive performance.

the model to construct steering vectors and applies them during decoding to control the model's reasoning behavior. For *training-based methods*, In addition to prior work such as CoT-Valve and TOPS that constructs SFT datasets with diverse reasoning lengths, we also include approaches that incorporate alternative reward-based methods. $\text{SimPO}_{\text{shortest}}$ was introduced in Overthink (Chen et al., 2025b) to adjust the effectiveness of the RL algorithm by reject sampling. ThinkPrune (Hou et al., 2025) uses progressive compression of RL training length to improve the effectiveness of context utilization during exploration via vast RL iterations. We provide more training details, evaluation details and baseline reproduce details in Appendix B.2-B.4.

Main Results. As shown in Table 1, our method TLDR achieves the best performance on the P.D. Ratio, which jointly considers accuracy and compression, outperforming all previous raining-based and training-free methods. While maintaining comparable accuracy, TLDR consistently achieves an A.C.R. and P.D.Ratio of over 40% across six diverse datasets. Notably, these results outperform several established RL-based baselines, including Overthink, CoT-Valve, TOPS, and ThinkPrune, demonstrating superior efficiency in reasoning compression. Notably, on relatively simple datasets that are prone to overthinking, such as ASDiv and GSM8K, TLDR attains higher compression ratios while preserving or even improving accuracy, demonstrating its effectiveness in reduce unnecessary reasoning.

Ablation Studies and Analysis. To demonstrate the advantages of our dynamic re-weighting method, we evaluate it against three categories of baselines: (*i*) Static Compression Methods: Direct-Mixture, CoT-Valve and TOPS, which served as representative Long-to-Short datasets for analyzing static compression in SFT data; (*ii*) Curriculum-based Dynamic Methods: $L \rightarrow S$ and $S \rightarrow L$ linearly adjust the sampling probabilities of System-1 and System-2 data over training steps, enabling a transition from large to small or from small to large in search of an optimal balance; and (*iii*) Random Re-weighting, which explores various data ratios by periodically resetting the sampling weights.

As illustrated in Table 2, while static baselines (*i*) like CoT-Valve and TOPS achieve competitive average accuracies of 73.6% and 74.1% respectively, they are inherently limited by their fixed distributions. Specifically, these static datasets struggle to maintain an optimal balance across diverse downstream tasks; for instance, TOPS fails to exhibit any compression effect on the AIME and AMC datasets, even leading to an increase in generation length. Furthermore, although baselines (*ii*) and (*iii*) attempt to identify an optimal data mix, they exhibit poor convergence. Their average accuracies (70.5%–71.8%) represent a significant decline of 3.4% to 4.7% compared to the static baselines. In contrast, our proposed **TLDR** method strikes a superior balance, attaining an average accuracy of



(a) GSM8K (b) AIME (c) MATH500

Figure 3: Frequency comparison of different keywords. The figure illustrates the distribution of exploratory, checking, and reflective keywords across datasets. *Exploratory Keywords: wait*, *Reflective Word: but*, *Checking Words: make sure/confirm/verify/check*, TLDR significantly reduces the presence of such words, reflecting its ability to produce streamlined and efficient reasoning steps.

74.9% (comparable to the Original model's 75.2%) with a substantial 38.9% reduction in average generation length. These results contend that achieving a precise balance between System-1 and System-2 data necessitates a dynamic re-weighting mechanism that leverages real-time training signals as a guiding heuristic to optimize both performance and compression efficiency.

Efficiency Discussion. We evaluate TLDR against three representative training-based baselines: ThinkPrune, L1, and CoT-Valve. As illustrated in Table 3, TLDR exhibits a decisive advantage in training efficiency. Specifically, TLDR requires only 88.1 GPU hours, achieving a significant speedup of approximately **8.0**× over ThinkPrune (704.1 hours) and **7.3**× over L1 (640.3 hours). While RL-based methods such as ThinkPrune and L1 necessitate extensive sampling iterations to satisfy specific length constraints or quotas, TLDR streamlines the optimization process by leveraging the rapid convergence of SFT coupled with dynamic reweighting. Furthermore, unlike CoT-Valve, which incurs substantial overhead from generating diverse CoT trajectories via model interpolation, our approach maintains a more compact and efficient training pipeline. It is worth noting that although methods like ThinkPrune attempt to optimize solutions within a constrained budget, they often suffer from prohibitive computational costs and lack the granularity to achieve a fine-grained Pareto optimal trade-off. In contrast, TLDR not only drastically reduces the training footprint but also achieves a superior A.C.R of 44.9%, effectively balancing task performance with inference-time economy. For a more detailed performance comparison with L1 performance, we provide analysis in Appendix C.1.

Analysis of Thinking Patterns. By comparing our method with other thinking compression techniques (Xu et al., 2025a), we find that TLDR effectively reduces internal redundancy by minimizing reliance on reasoning patterns like reflection or checking in GSM8K and MATH500 benchmarks. This avoids excessive computational cost while crucially retaining complex reasoning behaviors for challenging problems to preserve System-2 capabilities, as shown Figure 3. Furthermore, the effectiveness of our System-1 data synthesis is validated through supplementary evaluations on non-mathematical Out-of-Distribution (OOD) benchmarks. Detailed sensitivity analysis of data sources, discussions on OOD generalization, and case studies are provided in Appendix C.2, C.3 and D.

6 Conclusion

This paper introduces TLDR, an innovative method designed to compress the reasoning processes of LLMs without sacrificing accuracy. By dynamically re-weighting the influence of System 1 (concise reasoning) and System 2 (detailed reasoning) data during the training process, TLDR allows LLMs to eliminate unnecessary steps for simpler problems while still engaging in deep contemplation for complex tasks. Unlike existing compression methods that necessitate extensive data collection and sensitive hyperparameter tuning, TLDR offers a streamlined, robust, and practical framework for developing LLMs that effectively balance computational efficiency with reasoning accuracy.

Limitation

Current validation has primarily focused on benchmarks with clear logical structures. While the results are promising, the effectiveness of this dynamic ratio-based pipeline in more open-ended or creative reasoning tasks, where the distinction between redundancy and necessary elaboration, which is hard to define, remains an area for exploration.

References

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

AI-MO Team. 2024a. AIME 2024: AIMO Validation Dataset. Hugging Face Dataset. Available at AI-MO/aimo-validation-aime.

AI-MO Team. 2024b. AMC 2023: AIMO Validation Dataset. Hugging Face Dataset. Available at AI-MO/aimo-validation-amc.

Alon Albalak, Yanai Elazar, Sang Michael Xie, Shayne Longpre, Nathan Lambert, Xinyi Wang, Niklas Muennighoff, Bairu Hou, Liangming Pan, Haewon Jeong, Colin Raffel, Shiyu Chang, Tatsunori Hashimoto, and William Yang Wang. 2024. A Survey on Data Selection for Language Models. *arXiv preprint arXiv:2402.16827*.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. 2021. Program Synthesis with Large Language Models. *arXiv preprint arXiv:2108.07732*.

Seyedarmin Azizi, Erfan Baghaei Potraghloo, and Massoud Pedram. 2025. Activation Steering for Chain-of-Thought Compression. *arXiv preprint arXiv:2507.04742*.

Mark Chen and Jerry Tworek. 2021. Evaluating Large Language Models Trained on Code. *arXiv preprint arXiv:2107.03374*.

Runjin Chen, Zhenyu Zhang, Junyuan Hong, Souvik Kundu, and Zhangyang Wang. 2025a. Seal: Steerable Reasoning Calibration of Large Language Models for Free. *arXiv preprint arXiv:2504.07986*.

Xingyu Chen and 1 others. 2025b. Do NOT Think That Much for 2+3=? On the Overthinking of o1-Like LLMs. *arXiv preprint arXiv:2412.21187*.

Jeffrey Cheng and Benjamin Van Durme. 2024. Compressed Chain of Thought: Efficient Reasoning Through Dense Representations. *arXiv preprint arXiv:2412.13171*.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. 2021. Training Verifiers to Solve Math Word Problems.

DeepSeek-AI. 2025. DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *arXiv preprint arXiv:2501.12948*.

Yuntian Deng, Yejin Choi, and Stuart Shieber. 2024. From Explicit CoT to Implicit CoT: Learning to Internalize Cot Step by Step. *arXiv preprint arXiv:2405.14838*.

Jesse Dodge, Maarten Sap, Ana Marasović, William Agnew, Gabriel Ilharco, Dirk Groeneveld, Margaret Mitchell, and Matt Gardner. 2021. Documenting Large Webtext Corpora: A Case Study on the Colossal Clean Crawled Corpus. *arXiv preprint arXiv:2104.08758*.

Amir Globerson, Terry Koo, Xavier Carreras, and Michael Collins. 2007. Exponentiated Gradient Algorithms for Log-linear Structured Prediction. In *Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML 2007), Corvallis, Oregon, USA, June 20-24, 2007*, volume 227 of *ACM International Conference Proceeding Series*, pages 305–312. ACM.

Daya Guo, Qihao Zhu, Dejian Yang, Zhenda Xie, Kai Dong, Wentao Zhang, Guanting Chen, Xiao Bi, Y. Wu, Y. K. Li, Fuli Luo, Yingfei Xiong, and Wenfeng Liang. 2024. DeepSeek-Coder: When the Large Language Model Meets Programming – The Rise of Code Intelligence. *arXiv preprint arXiv:2401.14196*.

Suchin Gururangan, Dallas Card, Sarah Dreier, Emily Gade, Leroy Wang, Zeyu Wang, Luke Zettlemoyer, and Noah A. Smith. 2022. Whose Language Counts as High Quality? Measuring Language Ideologies in Text Data Selection. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2562–2580, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Tingxu Han, Zhenting Wang, Chunrong Fang, Shiyu Zhao, Shiqing Ma, and Zhenyu Chen. 2024. Token-budget-aware LLM Reasoning. *arXiv preprint arXiv:2412.18547*.

Shibo Hao, Sainbayar Sukhbaatar, DiJia Su, Xian Li, Zhiting Hu, Jason Weston, and Yuandong Tian. 2024. Training Large Language Models to Reason in A Continuous Latent Space. *arXiv preprint arXiv:2412.06769*.

Zhiwei He, Tian Liang, Jiahao Xu, Qiuzhi Liu, Xingyu Chen, Yue Wang, Linfeng Song, Dian Yu, Zhenwen Liang, Wenxuan Wang, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. 2025. DeepMath-103K: A Large-Scale, Challenging, Decontaminated, and Verifiable Mathematical Dataset for Advancing Reasoning. *arXiv preprint arXiv:2504.11456*.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021. Measuring Mathematical Problem Solving with The Math Dataset. *arXiv preprint arXiv:2103.03874*.

Danny Hernandez, Tom Brown, Tom Conerly, Nova DasSarma, Dawn Drain, Sheer El-Showk, Nelson Elhage, Zac Hatfield-Dodds, Tom Henighan, Tristan Hume, Scott Johnston, Ben Mann, Chris Olah, Catherine Olsson, Dario Amodei, Nicholas Joseph, Jared Kaplan, and Sam McCandlish. 2022. Scaling Laws and Interpretability of Learning from Repeated Data. *arXiv preprint arXiv:2205.10487*.

Bairu Hou, Yang Zhang, Jiabao Ji, Yujian Liu, Kaizhi Qian, Jacob Andreas, and Shiyu Chang. 2025. ThinkPrune: Pruning Long Chain-of-Thought of LLMs via Reinforcement Learning. *arXiv preprint arXiv:2504.01296*.

Yuxuan Jiang, Dawei Li, and Frank Ferraro. 2025. DRP: Distilled Reasoning Pruning with Skill-aware Step Decomposition for Efficient Large Reasoning Models. *arXiv preprint arXiv:2505.13975*.

Yu Kang, Xianghui Sun, Liangyu Chen, and Wei Zou. 2024. C3oT: Generating Shorter Chain-of-Thought without Compromising Effectiveness. *arXiv preprint arXiv:2412.11664*.

Kimi-1.5 Team. 2025. Kimi k1. 5: Scaling Reinforcement Learning with LLMs. *arXiv preprint arXiv:2501.12599*.

Hugo Laurençon, Lucile Saulnier, Thomas Wang, Christopher Akiki, Albert Villanova del Moral, Teven Le Scao, Leandro Von Werra, Chenghao Mou, Eduardo González Ponferrada, Huu Nguyen, Jiyung Sarria, Niklas Muennighoff, and 1 others. 2023. The BigScience ROOTS Corpus: A 1.6TB Composite Multilingual Dataset. *arXiv preprint arXiv:2303.03915*.

Ayeong Lee, Ethan Che, and Tianyi Peng. 2025. How Well do LLMs Compress Their Own Chain-of-Thought? A Token Complexity Approach. *arXiv preprint arXiv:2503.01141*.

Katherine Lee, Daphne Ippolito, Andrew Nystrom, Chiyuan Zhang, Douglas Eck, Chris Callison-Burch, and Nicholas Carlini. 2022. Deduplicating Training Data Makes Language Models Better. *arXiv preprint arXiv:2107.06499*.

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. 2022. Solving quantitative reasoning problems with language models. In *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*.

Yang Liu, Jiahuan Cao, Chongyu Liu, Kai Ding, and Lianwen Jin. 2024. Datasets for Large Language Models: A Comprehensive Survey. *arXiv preprint arXiv:2402.18041*.

Llama-3 Team. 2024. The Llama 3 Herd of Models. *arXiv preprint arXiv:2407.21783*.

Shayne Longpre, Gregory Yauney, Emily Reif, Katherine Lee, Adam Roberts, Barret Zoph, Denny Zhou, Jason Wei, Kevin Robinson, David Mimno, and Daphne Ippolito. 2023. A Pretrainer's Guide to Training Data: Measuring the Effects of Data Age, Domain Coverage, Quality, & Toxicity. *arXiv preprint arXiv:2305.13169*.

Haotian Luo, Haiying He, Yibo Wang, Jinluan Yang, Rui Liu, Naiqiang Tan, Xiaochun Cao, Dacheng Tao, and Li Shen. 2025a. Ada-R1: Hybrid-CoT via Bi-Level Adaptive Reasoning Optimization. *arXiv preprint arXiv:2504.21659*.

Haotian Luo, Li Shen, Haiying He, Yibo Wang, Shiwei Liu, Wei Li, Naiqiang Tan, Xiaochun Cao, and Dacheng Tao. 2025b. O1-Pruner: Length-Harmonizing Fine-Tuning for O1-Like Reasoning Pruning. *arXiv preprint arXiv:2501.12570*.

Xinyin Ma, Guangnian Wan, Runpeng Yu, Gongfan Fang, and Xinchao Wang. 2025. CoT-Valve: Length-Compressible Chain-of-Thought Tuning. *arXiv preprint arXiv:2502.09601*.

Yu Meng, Mengzhou Xia, and Danqi Chen. 2024. SimPO: Simple Preference Optimization with a Reference-Free Reward. *arXiv preprint arXiv:2405.14734*.

Shen-Yun Miao, Chao-Chun Liang, and Keh-Yih Su. 2021. A Diverse Corpus For Evaluating and Developing English Math Word Problem Solvers. *arXiv preprint arXiv:2106.15772*.

Niklas Muennighoff, Alexander M. Rush, Boaz Barak, Teven Le Scao, Aleksandra Piktus, Nouamane Tazi, Sampo Pyysalo, Thomas Wolf, and Colin Raffel. 2023. Scaling Data-Constrained Language Models. *arXiv preprint arXiv:2305.16264*.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. 2025. s1: Simple test-time scaling. *arXiv preprint arXiv:2501.19393*.

PaLM Team. 2022. PaLM: Scaling Language Modeling with Pathways. *arXiv preprint arXiv:2204.02311*.

Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, and 1 others. 2022. Scaling Language Models: Methods, Analysis & Insights from Training Gopher. *arXiv preprint arXiv:2112.11446*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *arXiv preprint arXiv:1910.10683*.

Yue Wang, Qiuzhi Liu, Jiahao Xu, Tian Liang, Xingyu Chen, Zhiwei He, Linfeng Song, Dian Yu, Juntao Li, Zhuosheng Zhang, Rui Wang, Zhaopeng Tu, Haitao Mi, and Dong Yu. 2025. Thoughts Are All Over the Place: On the Underthinking of o1-Like LLMs. *arXiv preprint arXiv:2501.18585*.

Han Wu, Yuxuan Yao, Shuqi Liu, and 1 others. 2025. Unlocking Efficient Long-to-Short LLM Reasoning with Model Merging. *arXiv preprint arXiv:2503.20641*.

Heming Xia, Yongqi Li, Chak Tou Leong, Wenjie Wang, and Wenjie Li. 2025. Tokenskip: Controllable Chain-of-Thought Compression in LLMs. *arXiv preprint arXiv:2502.12067*.

Albert Xu, Eshaan Pathak, Eric Wallace, Suchin Gururangan, Maarten Sap, and Dan Klein. 2021. Detoxifying Language Models Risks Marginalizing Minority Voices. *arXiv preprint arXiv:2104.06390*.

Haotian Xu, Xing Wu, Weinong Wang, Zhongzhi Li, Da Zheng, Boyuan Chen, Yi Hu, Shijia Kang, Jiaming Ji, Yingying Zhang, and 1 others. 2025a. RedStar: Does Scaling Long-CoT Data Unlock Better Slow-Reasoning Systems? *arXiv preprint arXiv:2501.11284*.

Silei Xu, Wenhao Xie, Lingxiao Zhao, and Pengcheng He. 2025b. Chain of Draft: Thinking Faster By Writing Less. *arXiv preprint arXiv:2502.18600*.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

Wenkai Yang, Shuming Ma, Yankai Lin, and Furu Wei. 2025. Towards Thinking-Optimal Scaling of Test-Time Compute for LLM Reasoning. *arXiv preprint arXiv:2502.18080*.

Yuxuan Yao, Shuqi Liu, Zehua Liu, Qintong Li, Mingyang Liu, Xiongwei Han, Zhijiang Guo, Han Wu, and Linqi Song. 2025. Activation-Guided Consensus Merging for Large Language Models. *arXiv preprint arXiv:2505.14009*.

Yiyao Yu, Yuxiang Zhang, Dongdong Zhang, Xiao Liang, Hengyuan Zhang, Xingxing Zhang, Ziyi Yang, and 1 others. 2025. Chain-of-Reasoning: Towards Unified Mathematical Reasoning in Large Language Models via a Multi-Paradigm Perspective. *arXiv preprint arXiv:2501.11110*.

Xiaojiang Zhang, Jinghui Wang, Zifei Cheng, Wenhao Zhuang, Zheng Lin, Minglei Zhang, Shaojie Wang, and 1 others. 2025. SRPO: A Cross-Domain Implementation of Large-Scale Reinforcement Learning on LLM. *arXiv preprint arXiv:2504.14286*.

A Metric

To quantify the efficiency and performance of the TLDR framework, we employ the following metrics:

Normalized Performance Metrics. To facilitate a fair comparison across heterogeneous datasets, we introduce Normalized Accuracy and Normalized Token Length, defined as:

$$\text{Acc}_{\text{norm}} = \frac{\mathcal{A}_{\text{curr}}}{\mathcal{A}_{\text{orig}}}, \quad \text{Len}_{\text{norm}} = \frac{L_{\text{curr}}}{L_{\text{orig}}}, \tag{10}$$

where $\mathcal{A}$ represents the accuracy score and L represents the average token length. These metrics provide a relative measure of performance retention and resource savings compared to the baseline.

Compression Rate (C.R.). The Compression Rate (C.R.) measures the reduction in response length relative to the original output. To ensure the metric remains non-negative, we define it as:

$$\eta = \max\left(\frac{L_{\text{orig}} - L_{\text{curr}}}{L_{\text{orig}}}, 0\right), \tag{11}$$

where L_{orig} and L_{curr} denote the token counts of the original and the processed responses, respectively.

The Average Compression Rate (A.C.R) across a set of N benchmarks is calculated as the arithmetic mean:

$$\text{A.C.R} = \frac{1}{N}\sum_{i=1}^{N}\eta_i. \tag{12}$$

B Implementation Details

B.1 Derivation Details: Gradient Derivation for Exponentiated Update of α_i

Gradient-based Weight Updates of TLDR. We consider the loss function:

$$\min_{\theta,\alpha\in(0,1)} \mathcal{L}(\theta,\alpha) = \sum_{i\in\{1,2\}} \alpha_i \cdot \underbrace{\left(\phi_i^* - \phi_i^\theta\right)}_{\delta_i}. \tag{13}$$

Assuming θ is fixed, δ_i can be treated as a constant. Thus, L is linear in α_i. α_i is required to be non-negative, and $\alpha_1 + \alpha_2 = 1$.

To maintain the constraints of the simplex without explicit projection, we employ the **Exponentiated Gradient (EG)** (Globerson et al., 2007) algorithm. For loss $\mathcal{L}(\alpha) = \sum \alpha_i\delta_i$, the gradient with respect to α_i is:

$$\frac{\partial\mathcal{L}}{\partial\alpha_i} = \delta_i. \tag{14}$$

The standard EG update rule is defined as:

$$\begin{aligned}\alpha'_{t+1}[i] &= \alpha_t[i]\exp\left(-\eta\frac{\partial\mathcal{L}}{\partial\alpha_i}\right)\\ &= \alpha_t[i]\exp(-\eta\delta_i),\end{aligned} \tag{15}$$

where $\eta > 0$ is the learning rate. This update naturally preserves the non-negativity of the weights.

Gradient Stabilization and Thresholding. In practice, the raw difference δ_i may exhibit high variance due to sampling noise in the validation set. To improve stability and focus on components that require improvement, we apply a normalization and thresholding operation. We define the stabilized gradient signal λ_i as:

$$\lambda_i = \max\left(\frac{\delta_i}{\text{scale}_i}, 0\right), \tag{16}$$

where scale_i is a normalization factor. Specifically, for our two-system setup, the signals are formulated to capture the relative gap:

$$\lambda_1 = \max\left(\frac{\phi_1^* - \phi_1^{\theta_p}}{\phi_1^{\theta_s} - \phi_1^{\theta_l}}, 0\right), \tag{17}$$

$$\lambda_2 = \max\left(\frac{\phi_2^* - \phi_2^{\theta_p}}{\phi_2^{\theta_l} - \phi_2^{\theta_s}}, 0\right). \tag{18}$$

Normalized Update Rule. By substituting the stabilized signal λ_i back into the EG framework and enforcing the sum-to-one constraint, we obtain the final normalized update rule:

$$\alpha_{t+1}[i] = \frac{\alpha_t[i]\exp(-\eta\lambda_i)}{\sum_{j\in\{1,2\}}\alpha_t[j]\exp(-\eta\lambda_j)}. \tag{19}$$

B.2 Training Details

To ensure the reproducibility of our experiments, we provide the comprehensive training configuration and the specific protocol used for model selection.

Training Infrastructure. The experiments were conducted using a distributed setup across two machines, each equipped with 8 NVIDIA 80GB GPUs. We decoupled the workload by dedicating one machine to `vLLM` inference for real-time validation, while the other handled the training process. Parameter synchronization between the training and inference clusters was executed every n steps utilizing `vLLM`'s synchronization interface to ensure consistency during evaluation.

Hyperparameters and Protocol. The models were trained for a total of $T = 500$ steps with a constant learning rate of 1×10^{-5}. We employed a balanced data mixture with a 0.5:0.5 ratio as init weights. To monitor the trade-off between reasoning performance and efficiency, we performed evaluation of the model every 32 steps on a validation set. This validation set consists of 512 problems sampled from historical AIME exams (1983–2023).

Checkpoint Selection Strategy. Our selection criterion prioritizes model efficiency without sacrificing essential reasoning capabilities. We define the optimal checkpoint as the one achieving the shortest average token length among all candidates that maintain at least 80% of the accuracy, as shown in Eq. (20) of the original baseline model on the validation set. Formally, let $\mathcal{A}_{\text{val}}$ and $\mathcal{A}_{\text{long}}$ denote the validation accuracy of the current checkpoint and the Long-CoT baseline respectively; we select:

$$\theta^* = \arg\min_{\theta} \{\text{Len}(\theta) \mid \mathcal{A}_{\text{val}}(\theta) \geq 0.8 \cdot \mathcal{A}_{\text{long}}\} \tag{20}$$

B.3 Data Construction Detail

For long CoT, we use the prompt from dataset s1.1 (Muennighoff et al., 2025). Each sample is generated 8 times using the original model. For short CoT, to avoid inconsistencies in the system prompt format, we adopt the short CoT construction method from AdaR1 (Luo et al., 2025a). We annotate 10 randomly selected questions from GSM8K using the instruct model, then fine-tune the long CoT model to overfit on them. For the GSM8K training set, we sample and retain only the examples with correct answers.

B.4 Baseline Reproduce Details

This section provides the technical specifications and hyperparameter settings used to reproduce the baseline models evaluated in our study.

OverThink. For the MATH12K dataset, we sample each problem 8 times. The shortest correct response is designated as the *chosen* sample, while the longest is selected as the *rejected* sample. The model is fine-tuned for 1 epoch.

ThinkPruner. We utilize the competition-level training data as specified in the original work. The model is trained for 10 epochs with a learning rate of 1×10^{-6} and a maximum response length of 4,096 tokens. We strictly follow the original early-stopping strategy to identify the optimal checkpoint for evaluation.

CoT-Valve. Since the original report does not cover all datasets, we reproduced the results using their publicly released `Mix-Chain-Z-GSM8K` dataset. Adhering to the official training protocol, we fine-tuned all models using LoRA ($r = 2$) for 5 epochs. Experiments were conducted on a cluster of 8 NVIDIA 80GB GPUs.

L1. In our reproduction on the 7B System-2 model, we utilized the *L1-Exact* reward function with a token length constraint between 100 and 4,096 tokens. The token difference penalization parameter α is set to 0.0003, consistent with the original paper. We append the prescribed prompt "Think for n_{token} tokens" to each query. During inference, the token budget is aligned with the average token count of our proposed method across all evaluated benchmarks.

Seal. The original Seal implementation relied on the `transformers` framework's `.generate` method for inference. To align evaluation settings and evaluate Seal across a broader range of datasets, we employed the **EasySteer** framework for reproduction, as it supports **vLLM** inference. Following the protocol of Seal, we extracted steering vectors using 1,000 samples.

Avg./Ties/Ties-Dare-Merging. Following the default configurations and all hyperparameter details specified in the `Model-Merging` repository and its accompanying technical report, we reproduced the results utilizing the implementation provided.

Ablation Baseline: Curriculum-based Dynamic. For the baseline models ($L \rightarrow S$ and $S \rightarrow L$): We adopt the same training step size as TLDR. The ratio of System-1 to System-2 data is adjusted every 100 steps. For the Curriculum Learning ($L \rightarrow S$) setup: The initial ratio of System-1 to System-2 data is $1:0$. We linearly decrease this ratio every 50 steps to $0.9:0.1$, continuing this progression until it eventually reaches $0:1$.For the baseline ($S \rightarrow L$) setup: The initial ratio starts at $0:1$. It is then linearly adjusted to $0.1:0.9$ and continues increasing until the final ratio reaches $1:0$.

B.5 Evaluation Detail

We use the DeepSeek-R1-Distill model and apply a temperature setting of 0.7 for evaluating all models. We limit the context window to 8K tokens for all datasets. We adopt nucleus sampling with top-p = 0.9 and use identical decoding parameters across all models. Meanwhile, considering the relatively small sizes of the AMC and AIME datasets, we sample 8 responses per question and compute the average.

Evaluation Framework. We use *skythought-eval*[1] as the framework, which supports efficient evaluation of long CoT reasoning with vLLM. The version of vLLM we use is 0.6.3.

Evaluation Dataset Detail. The mathematical datasets employed in this study are detailed in the following sections.

ASDiv A diverse corpus of 2,305 simple English math word problems designed to evaluate various MWP solvers across diverse text patterns and elementary school problem types.

GSM8K A high-quality benchmark consisting of 8,500 human-written grade school math word problems. The test set of 1,319 questions emphasizes multi-step sequential reasoning and natural language solutions.

MATH500 A rigorous collection of 500 high school competition-level problems across seven subjects, including Algebra and Geometry, requiring advanced mathematical reasoning and generalization.

AIME2024 A dataset featuring 30 problems from the 2024 American Invitational Mathematics Examination, specifically curated to test deep mathematical insight and complex problem-solving skills.

AMC Comprising 40 selected problems from the AMC12 (2022 and 2023) sourced from the AoPS wiki, targeting advanced high-school level mathematics.

MinervaMath A specialized dataset containing 272 high-difficulty math problems designed to challenge the limits of modern reasoning models.

C Further and More Granular Analysis

C.1 Comparison with Token-Budget-Aware Models

We compared our redundancy reduction method with both quota-controlled models and reasoning models under the same token budget, in order to evaluate the effectiveness of our approach relative to explicit quota-based control, as summarized in Table 4.

In our experiments, to enable a fair comparison, the quota-controlled baseline methods such as L1 (Aggarwal and Welleck, 2025) were evaluated under three different inference budgets. Under Budget_1, each dataset were allowed the same token quota as TLDR. Under Budget_2, each dataset were allowed 1.2× the average TLDR output tokens. Under Budget_3, each dataset were restricted to at most 0.8× the TLDR average output tokens. These settings allowed us to systematically evaluate the performance of L1 under varying token constraints and compare it with our TLDR method. The results show that our method achieves higher reasoning accuracy than all the L1 (Aggarwal and Welleck, 2025) baselines under approximately the same token quota. As shown in Table 4, across different inference budgets, L1 fails to achieve compression rates comparable to TLDR, with observed rates of 23.80%, 28.01%, and 21.41%, all lower than that of TLDR. This discrepancy is particularly pronounced on simpler datasets such as GSM8K, where the rigid quota-controlled mechanism of L1 leads to inefficient use of computational resources.

Furthermore, our approach demonstrates more efficient utilization of context length and does not require explicitly specifying a reasoning quota, offering a more flexible and adaptive inference mechanism. TLDR demonstrates stronger compression efficiency on simple problems.

C.2 Discussion on Data Source Sensitivity

Our previous exploratory experiments helped delineate the respective roles and difficulty ranges of System-1 and System-2 data. Building on this, we further investigate whether the difficulty levels chosen for System-1 and System-2 data construction influence the final TLDR performance.

Concretely, we collect question data spanning different difficulty levels and categorize the sources into three tiers: Easy (GSM8K),

[1] https://github.com/NovaSky-AI/SkyThought

Model	Accuracy					Generation Length				
	GSM8K	MATH	AIME	AMC	Avg.	GSM8K	MATH	AIME	AMC	A.C.R.↑
DS-Qwen-7B Models										
Original	89.4	86.8	42.9	81.5	**75.2**	554	2861	6820	4510	–
TLDR	87.7	87.4	41.2	83.1	**74.8**	253	1556	6368	3386	**32.89%**
L1-Burget$_1$	86.4	88.6	42.2	84.6	**75.4**	301	2301	5875	3784	**23.80%**
L1-Burget$_2$	86.4	87.6	45.1	84.6	**75.9**	312	1831	5675	3807	**28.01%**
L1-Burget$_3$	86.1	88.4	45.5	83.3	**75.8**	292	2589	6007	3746	**21.41%**

Table 4: Performance comparison of TLDR with the budget-aware baseline, L1 (Aggarwal and Welleck, 2025). The accuracy is measured by sampling multiple responses from the LLMs to reduce variance. The terms Budget$_1$, Budget$_2$, and Budget$_3$ refer to different budget settings for L1. Budget$_1$ is assigned the mean TLDR output token for each dataset, Budget$_2$ allows 1.2× the average TLDR output tokens for each dataset, while Budget$_3$ allows fewer tokens than the TLDR average, set to 0.8×.

Model	Accuracy					Generation Length				
	GSM8K	MATH	AIME	AMC	Avg.	GSM8K	MATH	AIME	AMC	A.C.R.↑
System-1 Data Source Ablation										
Original	89.4	86.8	42.9	81.5	**60.1**	554	2861	6820	4510	–
-Easy	87.7	87.4	41.2	83.1	**59.9**	253	1556	6368	3386	**32.87%**
-Medium	88.2	86.2	41.5	31.3	**61.8**	318	2083	6604	3945	**21.37%**
Hard	83.6	80.2	30.0	65.3	**64.8**	495	2970	6874	4947	**2.64%**
System-2 Data Source Ablation										
Original	89.4	86.8	42.9	81.5	**60.1**	554	2861	6820	4510	–
-Easy	83.9	86.8	42.5	83.4	**74.2**	446	2639	6580	4047	**10.26%**
-Medium	91.6	87.6	40.4	81.5	**75.3**	342	2761	6553	4116	**13.61%**
-Hard	87.7	87.4	41.2	83.1	**74.8**	253	1556	6368	3386	**32.87%**

Table 5: Ablation study on the difficulty levels used for constructing System-1 and System-2 data. TLDR models are trained using variants of System-1 and System-2 data constructed from questions of different difficulty levels.

Medium (MATH500 training set), and Hard (s1-style prompts). Based on these categories, we perform a difficulty ablation by replacing the original question sources used to construct System-1 and System-2 data with alternatives from other difficulty levels. In our experimental design, we replace the question sources used for System-1 data construction with Medium and Hard problems, and replace the question sources used for System-2 data construction with Easy and Medium problems. This results in four additional variant configurations, enabling us to systematically analyze the impact of difficulty mismatches between System-1 and System-2 data on TLDR performance.

System-1 data uses simpler problems, enabling better token compression. We examine how the composition of System-1 data affects the generalization of compression. As shown in the Table 5, replacing the questions used to construct System-1 data with more difficult medium or hard questions actually reduces the final compression efficiency of TLDR. Specifically, the number of tokens in the TLDR, compressed models A.C.R. decreases from 32.87% to 21.37% and 2.64%. Our experiments demonstrate that constructing compression data primarily from *lower-difficulty* problems can effectively reduce the token count of *high-difficulty* problems. **System-2 data needs to be sampled from more difficult problems to maintain accuracy.** In contrast, using easier questions for System-2 data impairs the recovery of original reasoning performance during long CoT sampling. As shown in Table 5, replacing the questions used to construct System-2 data with Medium or Easy questions substantially reduces the final compression efficiency of TLDR, with the A.C.R. dropping from 32.87% to 13.61% and 10.26%, respectively. These experiments further validate the importance of per-

Model	Leetcode	MBPP	HumanEval	**Average**
	Pass@1↑ / Tokens↓	Pass@1↑ / Tokens	Pass@1↑ / Tokens↓	Pass@1↑ / Tokens↓
Original	33.3 / 7088	61.4 / 1739	67.6 / 2692	54.1 / 3839
TLDR	**34.4 / 6793**	**64.3 / 1234**	**73.1 / 2536**	**57.3 / 3521**
Δ	+1.1 / -295	+2.9 / -505	+5.5 / -156	+3.2 / -318

Table 6: Performance comparison between R1-Distill-Qwen-7B and TLDR across three popular coding benchmarks. Pass@1 accuracy is reported alongside the average number of tokens generated.

forming dynamic re-weighting by using sufficiently challenging data in TLDR.

C.3 Discussion and Analysis on the Non-Math Domain Benchmark

To evaluate the model's generalization capabilities beyond the mathematical domain, we benchmarked its performance on **HumanEval** (Chen and Tworek, 2021), **MBPP** (Austin et al., 2021), and **LeetCode** (Guo et al., 2024). These datasets allow us to analyze code reasoning performance and observe how effectively the model extends its logic to non-mathematical tasks. The comprehensive results of this evaluation are presented in Table 6.

The experimental results demonstrate that under Out-of-Distribution evaluation conditions, the reasoning compression strategy—initially optimized on mathematical datasets—generalizes remarkably well to complex programming challenges. This strategy effectively mitigates reasoning redundancy, shifting the model's processing from a verbose, step-by-step System-2 deliberation toward a more streamlined and efficient System-1 logical synthesis. As a result, we observe a substantial reduction in output token counts accompanied by a simultaneous boost in Pass@1 accuracy. Specifically, we achieved consistent gains in both computational efficiency and task performance across the **HumanEval**, **MBPP**, and **LeetCode** benchmarks, underscoring the cross-domain robustness of our approach.

D Case Study

To further elucidate the behavioral divergence between the baseline and TLDR strategies, we perform a qualitative analysis using the DeepSeek-R1-Distill-Qwen-7B model across three representative benchmarks: GSM8K, AIME, and MATH500. As illustrated in Figures 4–6, the baseline model frequently exhibits verbose reasoning trajectories, often cluttered with redundant loops or speculative fillers. Conversely, TLDR distills the reasoning process into a significantly more concise format without compromising logical rigor or final accuracy. These cases underscore TLDR's efficacy in filtering superfluous cognitive overhead—such as excessive self-correction or circular explorations—resulting in a more streamlined and purposeful inference path.

E More Related Work

A parallel category of work explores the construction of *efficient representations* to optimize reasoning costs. TokenSkip (Xia et al., 2025) prioritizes data based on token importance, leading to more streamlined reasoning paths. Shifting from the discrete token level to the continuous embedding level, COCONUT (Hao et al., 2024) explores efficient reasoning within the latent space. This direction is further extended by ICoT-KD (Deng et al., 2024) and CCoT (Cheng and Van Durme, 2024), which develop specialized reasoning strategies in the hidden space. More recently, Token Assorted has been proposed to strike a balance between interpretability and efficiency by hybridizing hidden-space and text-based reasoning.

E.1 LLM Utilization

During the preparation of this manuscript, the authors employed the LLM Gemini (specifically the Gemini 3 Flash variant) for the purpose of language editing and grammatical refinement.

Generated Answer For GSM8K

Question:
Michael is traveling on the interstate at an average speed of 50km/hr (taking into account all the necessary stops for gas, relaxation, etc.). If he goes on like this for 3 days, and the distance between Alaska and Texas is 6000 km, what percentage of this distance has he covered?

Baseline (842 tokens)

Step 1: 114 tokens

Okay, let me try to figure out how much of the distance from Alaska to Texas Michael has covered. So, Michael is driving at an average speed of 50 km per hour, and he's doing this for 3 days. The total distance between Alaska and Texas is 6000 km. I need to find out what percentage of that 6000 km he's covered.\n\nFirst, I think I should calculate the total distance Michael has driven in those 3 days. To do that, I need to know how many hours he's been driving each day. Hmm, the problem says he's taking into account all stops for gas, relaxation, etc., so I guess he's driving non-stop each day.

Step 2: 87 tokens

But wait, how many hours is that? Oh, the problem doesn't specify the number of hours he drives each day. That's a bit confusing. Maybe I'm supposed to assume a certain number of hours per day? Or perhaps the 50 km/h speed is his average speed including stops, so I can still use it to find the total distance.\n\nWait, maybe I can just calculate the total distance based on his average speed and the time. But without knowing the exact number of hours, I can't do that.

.....

Step 13: 40 tokens

Alternatively, maybe the problem meant 3 hours each day, but that's unclear. Wait, perhaps the problem is in a different context, like driving across multiple states for 3 days, but without knowing the daily driving time, it's hard to say.

Step 14: 49 tokens

Maybe I should proceed with the initial assumption of 24 hours per day, even though it's unrealistic, and provide that answer.\n\nSo, final calculation: 50 km/h * 72 h = 3600 km. 3600/6000 = 0.6 => 60%.\n</think>\n\nThe final answer is 60%.

TLDR(248 tokens)

Step 1: 82 tokens

Let's break down the problem step by step.
1. Calculate the total distance Michael will travel: Michael is traveling at an average speed of 50 km/hr for 3 days. To find the total distance, we multiply the speed by the number of days: Total distance = 50km/hr × 3 days = 150km.

Step 2: 150 tokens

2. Calculate the percentage of the distance between Alaska and Texas that Michael has covered: The distance between Alaska and Texas is 6000 km. To find the percentage of this distance that Michael has covered, we use the formula:

$$\text{Percentage} = \frac{\text{Distance covered}}{\text{Total distance}} \times 100$$

Substituting the values:

$$\text{Percentage} = \frac{150}{6000} \times 100 = 2.5$$

Step 3: 16 tokens

So, Michael has covered 2.5% of the distance between Alaska and Texas.

Figure 4: Comparison of reasoning processes on the GSM8K dataset.

Generated Answer For AIME

Question:
Let B be the set of rectangular boxes with surface area 54 and volume 23. Let r be the radius of the smallest sphere that can contain each of the rectangular boxes that are elements of B. The value of r^2can be written as $\frac{p}{q}$, where p and q are relatively prime positive integers. Find p + q

Baseline (7482 tokens)

Step 1: 87 tokens

Observe that the \"worst\" possible box is one of the maximum possible length. \nBy symmetry, the height and the width are the same in this antioptimal box. (If the height and width weren't the same, the extra difference between them could be used to make the length longer.) Thus, let the width and height be of length a and the length be L. \nWe're given that the volume is 23; thus, $a^2L=23$.

Step 2: 168 tokens

We're also given that the surface area is $54=2\\cdot27$; thus, $a^2+2aL=27$. \nFrom the first equation, we can get $L=\\dfrac{23}{a^2}$. We do a bunch of algebra:
\n\\begin{align*}\nL&=\\dfrac{23}{a^2}
27&=a^2+2aL
\\\\\n&=a^2+2a\\left(\\dfrac{23}{a^2}\\right)
\\\\\n&=a^2+\\dfrac{46}a \\\\\n27a&=a^3+46
\\\\\na^3-27a+46&=0. \\\\\n\\end{align*}\nWe can use the Rational Root Theorem and test a few values. It turns out that $a=2$ works. We use synthetic division to divide by $a-2$:
\n[Asdf.png](https://artofproblemsolving.com/wiki/index.php/File:Asdf.png)\nAs we expect, the remainder is 0, and we are left with the polynomial $x^2+2x-23$.

.....

Step 27: 168 tokens

To find the maximum m for $x^3-mx^2+27x-23=0$\nrewrite m as function of x and calculate derivatives to get maximum value, \\[m(x) =-x + 27x^{-1} - 23x^{-2}\\]\n\\[m'(x) = -1 - 27x^{-2} - 46x^{-3} = 0\\]\n\\[x^3 -27x+46=0\\]\n\\[(x-2)(x^2+2x-23)=0\\]\nwhen x = 2, \\[m= 2 + \\frac{27}{2} - \\frac{23}{4} = \\frac{39}{4}\\]\nthe rest is similar to solution 3\n~[luckuso](https://artofproblemsolving.comhttps://artofproblemsolving.com/wiki/index.php/User:Cyantist)

TLDR(5960 tokens)

Step 1: 84 tokens

Observe that the \"worst\" possible box is one of the maximum possible length. \nBy symmetry, the height and the width are the same in this antioptimal box. (If the height and width weren't the same, the extra difference between them could be used to make the length longer.) Thus, let the width and height be of length a and the length be L. \nWe're given that the volume is 23; thus, $a^2L=23$.

Step 2: 155 tokens

We're also given that the surface area is $54=2\\cdot27$; thus, $a^2+2aL=27$. \nFrom the first equation, we can get $L=\\dfrac{23}{a^2}$. We do a bunch of algebra:
\n\\begin{align*}\nL&=\\dfrac{23}{a^2}
\\\\\n27&=a^2+2aL
\\\\\n&=a^2+2a\\left(\\dfrac{23}{a^2}\\right)
\\\\\n&=a^2+\\dfrac{46}a \\\\\n27a&=a^3+46
\\\\\na^3-27a+46&=0. \\\\\n\\end{align*}\nWe can use the Rational Root Theorem and test a few values. It turns out that $a=2$ works. We use synthetic division to divide by $a-2$:
\n[Asdf.png](https://artofproblemsolving.com/wiki/index.php/File:Asdf.png)\nAs we expect, the remainder is 0, and we are left with the polynomial $x^2+2x-23$.

.....

Step 8: 60 tokens

To find the maximum m for $x^3-mx^2+27x-23=0$\nrewrite m as function of x and calculate derivatives to get maximum value, \\[m(x) =-x + 27x^{-1} - 23x^{-2}\\]\n\\[m'(x) = -1 - 27x^{-2} -46x^{-3} = 0\\]\n\\[x^3 - 27x+46=0\\]\n\\[(x-2)(x^2+2x-23)=0\\]\nwhen x = 2, \\[m= 2 + \\frac{27}{2} - \\frac{23}{4} = \\frac{39}{4}\\]\nthe rest is similar to solution 3.

Figure 5: Comparison of Reasoning process on AIME: Baseline vs. TLDR.

Generated Answer For MATH500

Question:
The set of points (x,y,z) that satisfy\n\\[2x = 3y = -z\\]is a line.\n\nThe set of points (x,y,z) that satisfy\n\\[6x = -y = -4z\\]is another line.\n\nFind the angle between these lines, in degrees.

Baseline (1165 tokens)

Step 1: 138 tokens

Okay, so I've got this problem where there are two lines defined by sets of equations, and I need to find the angle between them. Hmm, let me see. The first line is given by the equations 2x = 3y = -z, and the second line is 6x = -y = -4z. I remember that to find the angle between two lines, I need to find the angle between their direction vectors. So, first step is probably to figure out the direction vectors of each line.\n\nLet me think about the first line: 2x = 3y = -z. I think that when an equation like this is given, it's setting all these expressions equal to some parameter, maybe t.

Step 2: 237 tokens

So, I can write each variable in terms of t. Let me try that. Let's let 2x = 3y = -z = t. Then, x would be t/2, y would be t/3, and z would be -t. So, the parametric equations for the first line would be x = t/2, y = t/3, z = -t. \n\nNow, the direction vector of a line given in parametric form is the coefficients of the parameter t. So, looking at x = (1/2)t, y = (1/3)t, z = (-1)t, the direction vector for the first line is (1/2, 1/3, -1). Hmm, that seems a bit messy with fractions. Maybe I can scale it up to make it simpler. If I multiply each component by 6, which is the least common multiple of denominators 2 and 3, I get (3, 2, -6). So, the direction vector for the first line is (3, 2, -6). That makes things easier.\n\nAlright, moving on to the second line: 6x = -y = -4z. Similarly, I can set all of these equal to another parameter, maybe s. So, 6x = -y = -4z = s.

.....

Step 10: 68 tokens

3. Calculate the dot product of the direction vectors \\((3, 2, -6)\\) and \\((2, -12, -3)\\):\n - \\(u \\cdot v = 3 \\cdot 2 + 2 \\cdot (-12) + (-6) \\cdot (-3) = 6 - 24 + 18 = 0\\).\n\nSince the dot product is zero, the vectors are perpendicular, meaning the angle between the lines is 90 degrees.\n\n\\[\n\\boxed{90}

TLDR(204 tokens)

Step 1: 44 tokens

To find the angle between the two lines, we first need to determine their direction vectors. For the first line, given by \\(2x = 3y = -z\\), we can express it in parametric form. Let \\(2x = 3y = -z = t\\).

Step 2: 52 tokens

Then, we have:\n\n\\[\nx = \\frac{t}{2}, \\quad y = \\frac{t}{3}, \\quad z = -t\n\\]\n\nThus, the direction vector \\(\\mathbf{v}\\) of the first line is \\(\\left(\\frac{1}{2}, \\frac{1}{3}, -1\\right)\\).\n\nFor the second line, given by \\(6x = -y = -4z\\), we can express it in parametric form as well.

Step 3: 37 tokens

Let \\(6x = -y = -4z = s\\). Then, we have:\n\n\\[\nx = \\frac{s}{6}, \\quad y = -s, \\quad z = \\frac{s}{-4}\n\\]\n\nThus, the direction vector \\(\\mathbf{w}\\) of the second line is \\(\\left(\\frac{1}{6}, -1, -\\frac{1}{4}\\right)\\)

Step 4: 45 tokens

Next, we use the dot product formula to find the angle between the two direction vectors. The dot product \\(\\mathbf{v} \\cdot \\mathbf{w}\\) is calculated as follows:\n\n\\[\n\\mathbf{v} \\cdot \\mathbf{w} =\\left(\\frac{1}{2}\\right)\\left(\\frac{1}{6}\\right) + \\left(\\frac{1}{3}\\right)(-1) + (-1)\\left(-\\frac{1}{4}\\right) = \\frac{1}{12} - \\frac{1}{3} + \\frac{1}{4}\n\\]

Step 5: 26 tokens

Simplifying the expression: [\n\\frac{1}{12} - \\frac{4}{12} + \\frac{3}{12} = \\frac{0}{12} = 0\n\\]\n\nSince the dot product is zero, the vectors are perpendicular, and thus the angle between the lines is \\(90^\\circ\\).

Figure 6: Comparison of Reasoning process on MATH500: Baseline vs. TLDR.