Think Before You Prune: Selective Self-Generated Calibration for Pruning Large Reasoning Models

Anonymous ACL submission

Abstract

Large Reasoning Models (LRMs) have demonstrated remarkable performance on complex reasoning benchmarks. However, their long chain-of-thought reasoning processes incur significant inference overhead. Pruning has emerged as a promising approach to reducing computational costs. However, existing efforts have primarily focused on large language models (LLMs), while pruning LRMs remains unexplored. In this work, we conduct the first empirical study on pruning LRMs and show that directly applying existing pruning techniques fails to yield satisfactory results. Our findings indicate that using self-generated reasoning data for calibration can substantially improve pruning performance. We further investigate how the difficulty and length of reasoning data affect pruning outcomes. Our analysis reveals that challenging and moderately long self-generated reasoning data serve as ideal calibration data. Based on these insights, we propose a Selective Self-Generated Reasoning (SSGR) data construction strategy to provide effective calibration data for pruning LRMs. Experimental results on the DeepSeek-R1-Distill model series validate that our strategy improves the reasoning ability of pruned LRMs by 10–13% compared to general pruning methods.

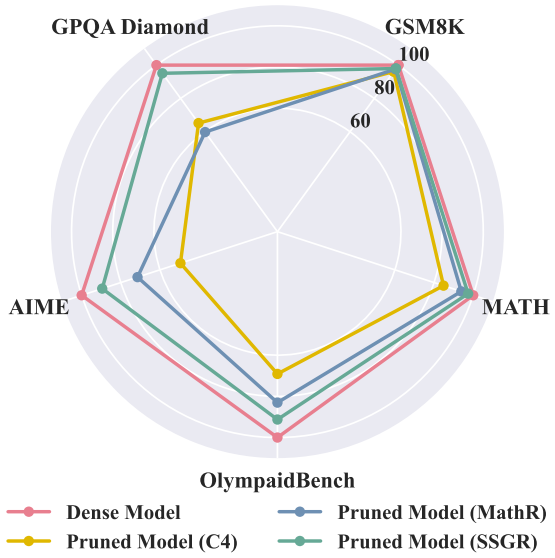

Figure 1: Performance of the pruned DeepSeek-R1-Distill-Qwen-7B model using SparseGPT with different calibration data (C4, MathR, and SSGR) across five reasoning benchmarks: GSM8K, MATH500, OlympiadBench, AIME, and GPQA Diamond.

1 Introduction

Recently, Large Reasoning Models (LRMs) such as OpenAI's o1/o3 (OpenAI, 2024, 2025) and the DeepSeek-R1 (Guo et al., 2025) series have achieved remarkable performance on challenging reasoning benchmarks, especially in competitive mathematics, programming, and science question answering. Unlike traditional Large language models (LLMs), these LRMs, after undergoing large-scale reinforcement learning, have learned to perform effective deep thinking through chain-of-thought (CoT) before producing the final answer (Chen et al., 2025a). However, the long CoT reasoning of LRMs significantly increases inference overhead (Ma et al., 2025). Striking a balance between accuracy and efficiency remains an open research question (Chen et al., 2025b; Luo et al., 2025a; Wang et al., 2025; Zhang et al., 2025).

In contrast to shortening CoTs at inference time, reducing model parameters offers a more fundamental and general solution to improving efficiency. Among various methods, pruning stands out as a promising strategy, which reduces model latency and computational cost by eliminating redundant parameters while preserving performance, and has been widely applied to LLMs (Zheng et al., 2024; Li et al., 2024b, 2025). Traditional magnitude-based pruning methods perform poorly on LLMs (Frantar and Alistarh, 2023). Most existing methods (Frantar and Alistarh, 2023; Sun et al., 2024; Gao et al., 2024) estimate parameter importance using a limited amount of data, referred to as

calibration data. The latest pruning methods (Zhou et al., 2025; Li et al., 2025) have achieved nearly lossless pruning performance. However, these approaches are proposed for LLMs, and their applicability to LRMs remains uninvestigated.

In this work, we present the first empirical study on pruning LRMs, revealing that directly applying existing LLM pruning methods can lead to substantial degradation in LRMs' reasoning performance. Surprisingly, we find that simply altering the calibration data can lead to dramatic improvements. Previous studies (Williams and Aletras, 2024; Bandari et al., 2024; Kong et al., 2025) have shown that the choice of calibration data significantly affects performance on downstream tasks. Shin et al. (2024) observe that minimizing reconstruction error is not always optimal and can even lead to overfitting on the calibration data. They recommend using self-generated calibration data to mitigate this issue. Subsequently, Ji et al. (2025) and Williams et al. (2024) propose their self-generated calibration data strategies, resulting in improved pruning performance. However, these studies have been conducted only on LLMs, and the impact of calibration data on pruning LRMs remains unknown.

To comprehensively investigate the impact of calibration data on pruning LRMs, we design experiments to explore the impact of different types of calibration data on pruning performance. Our empirical results indicate that choosing self-generated reasoning data as calibration data achieves the optimal pruning performance for LRMs. Furthermore, we investigate the effects of data difficulty and reasoning length on pruning performance, finding that challenging reasoning data with moderate length serves as the ideal calibration data. Building on this, we propose a Selective Self-Generated Reasoning (SSGR) data construction strategy that provides effective calibration data for pruning LRMs, aiming to minimize the degradation in reasoning performance after pruning. To evaluate the effectiveness of our proposed calibration data construction method, we conduct experiments on the DeepSeek-R1-Distill model series. As shown in Figure 1, the results demonstrate that our method significantly outperforms baseline calibration data, offering valuable insights and guidance for pruning LRMs.

Overall, our contributions are as follows:

- We conduct the first empirical study on pruning LRMs and find that conventional pre-training data is no longer suitable as calibration data, while self-generated reasoning data emerges as the optimal choice.
- Further exploration of self-generated reasoning data reveals that challenging and moderately long reasoning samples are more effective. Based on this, we propose a Selective Self-Generated Reasoning (SSGR) data construction strategy.
- Extensive experiments demonstrate that our SSGR data consistently outperforms baseline calibration data in pruning performance.

2 Related work

Large Reasoning Models OpenAI o1 series models (OpenAI, 2024), which are trained through large-scale reinforcement learning, perform long internal CoTs for thinking (Jaech et al., 2024) before generating responses, achieving outstanding performance in complex reasoning tasks spanning mathematics, coding, science, and beyond. The success of o1 marks a pivotal shift from LLMs to LRMs. Many studies (Team, 2025a; Team et al., 2025; Seed et al., 2025; Wen et al., 2025) have made great efforts to replicate the performance of o1. DeepSeek-R1 (Guo et al., 2025) achieves state-of-the-art performance by integrating cold-start data and multi-stage training. QwQ (Team, 2025b) delivers comparable performance with just 32B parameters, further highlighting the effectiveness of scaling reinforcement learning in training LRMs. Additionally, DeepSeek leverages distilled data from DeepSeek-R1 to perform supervised fine-tuning of small models (Llama and Qwen model series), also enabling test-time scaling. Several open-source studies (Muennighoff et al., 2025; Ye et al., 2025) have demonstrated that even a small amount of high-quality distilled data is enough to train competitive LRMs.

LLM Pruning LLM pruning can be broadly categorized into unstructured pruning (Frantar and Alistarh, 2023; Sun et al., 2024; Yin et al., 2024; Ilin and Richtarik, 2025) and structured pruning (Ma et al., 2023; Gao et al., 2024; Sengupta et al., 2025; Li et al., 2025) approaches. We primarily focus on the former. SparseGPT (Frantar and Alistarh, 2023) was the first to conduct experiments on LLMs, revealing that traditional magnitude-based pruning becomes ineffective at this scale. It introduces the use of second-order gradient information to assess parameter importance. Wanda (Sun et al., 2024) proposes a simple and effective approach that estimates weight importance using the prod-

uct of the weight magnitude and the norm of the corresponding input activations, without any retraining or weight updates. RIA (Zhang et al., 2024) integrates weight, input, and output activations as a pruning metric. Recent works (Yin et al., 2024; Lu et al., 2024; Cunegatti et al., 2025) have focused on allocating layer-wise pruning ratios instead of applying a uniform sparsity ratio across all layers. OWL (Yin et al., 2024) introduces outlier-weighted layer-wise sparsity, which proportionally links each layer's sparsity to the observed outliers. Evopress (Sieberling et al., 2024) leverages evolutionary algorithms to implement a dynamic compression framework for LLMs, while NEURONAL (Cunegatti et al., 2025) adaptively selects the optimal hyperparameters for block-wise and row-wise sparsity ratios.

Calibration data Calibration data is a critical component of pruning methods. Williams and Aletras (2024) find that post-training pruning is sensitive to calibration data, which significantly affects downstream performance. Shin et al. (2024) observe that the reconstruction error objective causes overfitting on calibration data and suggest using self-generated calibration data to mitigate this issue. Bandari et al. (2024) points out that the commonly used C4 dataset is not optimal. Arithmetic datasets are more effective as calibration data than pretraining datasets, and calibration data in the ICL format tends to have broader applicability. Ji et al. (2025) demonstrate that calibration data more similar to the training distribution yields better pruning performance and proposes a self-generated calibration data synthesis strategy. Our work presents the first investigation into the impact of calibration data on pruning LRMs. We further examine how data difficulty and reasoning length influence pruning performance and propose an effective calibration data construction strategy.

3 The Impact of Calibration Data for Pruning LRMs.

3.1 Expermental Details

Dense Model and Pruning Method We adopt the powerful open-source reasoning model DeepSeek-R1-Distill-Qwen-7B (Guo et al., 2025) as our dense model. For pruning, we use SparseGPT (Frantar and Alistarh, 2023), a widely used and effective method. Experiments are conducted under a 50% unstructured sparsity setting.

Calibration Data We follow prior works and randomly sample 128 sequences as calibration data. To further highlight the impact of different calibration data on the pruned model, We extend the length of each sequence to 16,384 tokens, whereas it is typically set to 2,048 tokens. We sample calibration data from different types of data, including:

- **Pre-training Data.** It is typically considered a primary source of calibration data. Following Ji et al. (2025), We select three commonly used pre-training datasets: C4 (Raffel et al., 2020), Wikipedia, and DCLM (Li et al., 2024a).
- **Math Reasoning Data.** The goal of pruning LRMs is to reduce the loss of reasoning capabilities in pruned model. A straightforward idea is to use reasoning data as calibration data. Therefore, we select GSM8K (Cobbe et al., 2021) and MATH (Hendrycks et al., 2021), two widely used mathematical reasoning datasets, as the reasoning data source. GSM8K is a high-quality elementary school math word problem dataset, while MATH is a challenging high school-level competition math problem dataset, covering seven subjects and five difficulty levels. Each problem in both datasets includes a complete step-by-step solution. We select their training sets and combine them into the Mathematical Reasoning (MathR) Dataset.
- **Self-Generated Reasoning Data.** Previous work (Ji et al., 2025) in LLM pruning has shown that using self-generated calibration data yields better results. We aim to explore and compare this approach in LRM pruning. For each problem in the MathR dataset, we sample 16 responses using the dense model with a temperature of 0.6, top-p of 0.95, and a maximum generation length of 32768 tokens. We discard responses that are overly repetitive, lack proper reasoning, or fail to follow the specified answer format. As a result, we obtain a new dataset, referred to as the Self-Generated Reasoning (SGR) Dataset. Apart from the experiments in Section 3.4, we adopt the first correct response for each problem as the final solution.

Evaluation We evaluate the performance of the pruned models on several reasoning benchmarks, including GSM8K (Cobbe et al., 2021), MATH500 (Lightman et al., 2024), AIME, which contains 60 problems from the 2024 and 2025 AIME competitions, and OlympiadBench (He et al., 2024). In addition, we evaluate the pruned model's reasoning generalization ability beyond mathematics using GPQA Diamond (Rein et al.,

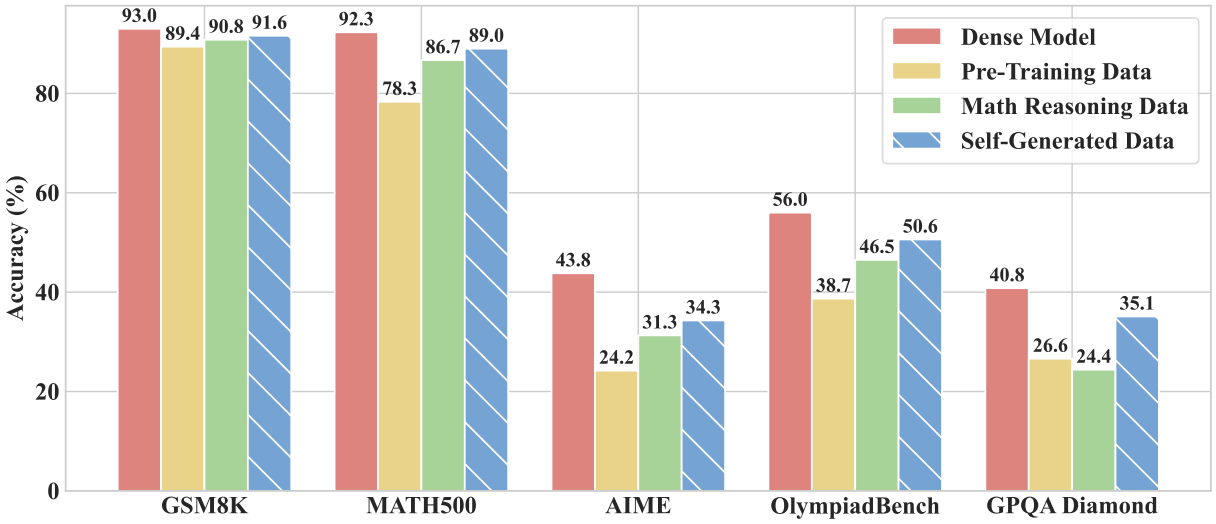

Figure 2: Comparison of pruning performance using three different types of calibration data for the dense model. For pre-training data (C4, Wikipedia, and DCLM), we report the highest accuracy among the three on each benchmark.

2024), a benchmark consisting of 198 PhD-level science questions spanning Biology, Chemistry, and Physics. All evaluations are conducted in a zero-shot CoT setting. To reduce output truncation and provide sufficient room for complex reasoning, we set the maximum output length to 32768 tokens. The sampling temperature and top-p are set to 0.6 and 0.95, respectively. We generate four responses per problem for all benchmarks and report the average pass@1 as the evaluation metric. Our evaluation framework is built upon the open-source codebase of DeepScaleR (Luo et al., 2025b).

3.2 Self-Generated Reasoning Data: A Superior Choice for Calibration

We prune the dense model using three types of calibration data, and the performance of the pruned models is shown in Figure 2.

Pre-training data is not suited as calibration data for pruning LRMs. The pruned model using pre-training data for calibration achieves 89.4% on GSM8K, with only a 3.3% drop in performance. This indicates that using pre-training data yields relatively small loss on straightforward reasoning tasks. However, on more challenging benchmarks, the pruned model suffers a significant performance degradation, with accuracy dropping by over 10% across the board. Notably, the pruned model attains only 24.2% on AIME, suggesting a substantial loss in complex reasoning capability. As shown in Table 3, Variation exists across different pre-training datasets. Using C4 as calibration data achieves the best overall performance, whereas Wikipedia leads to the lowest, falling 5.6% behind C4 on average. DCLM surpasses C4 on AIME and matches its overall performance.

Sampling calibration data from math reasoning data helps preserve more mathematical reasoning capability. Compared to pre-training data, using math reasoning data for calibration yields a 7.1%–8.4% improvement in accuracy on complex mathematical reasoning tasks across MATH500, AIME, and OlympiadBench. This benefit can be attributed to the presence of explicit reasoning steps in the data, which help the model retain more mathematical reasoning capability during pruning. However, the performance on GPQA Diamond, a scientific benchmark spanning multiple disciplines, drops significantly, falling even 2.2% below that achieved when using pre-training data for pruning. This suggests that while effective for mathematical tasks, math reasoning data lacks generalization and fails to transfer reasoning ability across domains.

Self-generated reasoning data stands out as the optimal choice of calibration data for pruning LRMs. Using self-generated reasoning data for calibration consistently yields the best performance across all benchmarks, with the exception of AIME, where the accuracy drop remains within 6%. It not only performs well on mathematical tasks, but also demonstrates strong cross-domain generalization. On GPQA Diamond, it achieves 35.1%, outperforming Math Reasoning Data by 10.7%. These results clearly demonstrate that self-generated reasoning data can further bridge the gap between the pruned and original model, effectively preserving

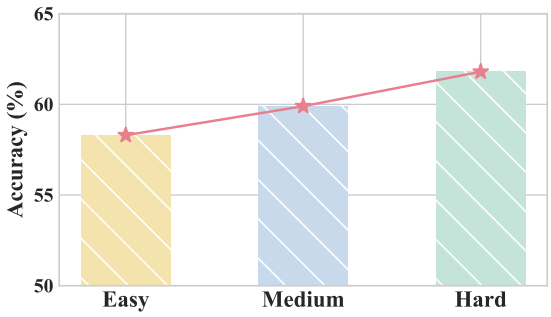

Figure 3: Comparison of Pruning Performance Using Calibration Data of Varying Difficulty Levels (*Easy*, *Medium*, and *Hard*).

	AIME	Olympiad	GPQA	AVG
Correctness				
True Sample	**36.7**	**52.1**	35.0	**41.3**
False Sample	35.4	51.0	**35.4**	40.6
Reasoning Length				
Random	36.7	**52.1**	35.0	41.3
Short	33.3	50.8	34.6	39.6
Meidum	**40.4**	51.1	**35.2**	**42.2**
Long	32.9	50.6	37.8	40.4

Table 1: Impact of Sample Correctness and Reasoning Length on Pruning Performance.

most of the model's reasoning capability.

The key characteristic of LRMs is to produce a long thinking process before arriving at an answer. Each response in the self-generated reasoning dataset similarly includes this thinking process. We find that if we remove the thinking process from the response and use the modified response for calibration, the pruning performance drops significantly, showing no improvement compared to the results obtained with MathR data, as shown in Table 3. This suggests that the thinking process in the response is essential for use as calibration data. The self-generated "think-then-answer" reasoning format significantly enhances pruning performance, allowing the pruned model to retain a problem-solving pattern that more closely resembles that of the original model.

3.3 Challenging Calibration Data Enhances Pruning Performance

During model training, challenging examples are often used to enhance reasoning capabilities (Ye et al., 2025). But does a similar principle apply to calibration data for pruning LRMs? Therefore, we propose a reasonable hypothesis: using challenging reasoning data for calibration can lead to better performance in pruned models.

To validate this hypothesis, we first need to categorize the data based on difficulty. The most straightforward approach is to rely on the difficulty labels provided within the datasets themselves. For instance, GSM8K consists of elementary school math problems and is undoubtedly the simplest. In the MATH dataset, each problem is assigned a difficulty level, determined from a human perspective: the easiest are labeled as Level 1, while the most difficult are labeled as Level 5. Based on these labels, we can perform a rough division of the data by difficulty. However, such a human-defined classification is inadequate and unreliable. There is a significant gap between what humans and models perceive as difficult. Moreover, different models may have entirely different notions of difficulty. Problems that a strong model can easily solve may be highly challenging for a weaker one. Therefore, the difficulty of the data should be determined by the model itself. We define a question's difficulty based on the accuracy of responses generated by the dense model. Specifically, questions that are always answered correctly (i.e., 100% accuracy) are labeled as *Easy*; those with a majority of correct responses (i.e., accuracy > 50%) are considered *Medium*; and those with fewer correct responses (i.e., accuracy $\leq$ 50%) are classified as *Hard*.

The average pruning performance using calibration data of varying difficulty levels is shown in Figure 3. More details regarding the performance of different difficulty levels can be found in the Table 4. **As the difficulty of the calibration data increases, the performance of the pruned model improves accordingly.** Specifically, on GSM8K and MATH500, even easy calibration data yields strong results. However, for more challenging benchmarks like AIME and GPQA, using hard calibration data leads to significant gains, improving performance by 6.3% and 8.6%, respectively, compared to using easy calibration data. This highlights the potential of letting the model identify and use challenging examples as calibration data, offering a practical and effective path toward better pruning performance.

3.4 Proper Reasoning Length Promotes Effective Pruning

It can be observed from Table 4 that more challenging samples generally exhibit longer reasoning lengths. This naturally leads to the question of

whether extending the reasoning length of calibration data can further enhance pruning performance.

In practice, the lengths of reasoning vary significantly among different responses to the same question. By selecting different responses, it is possible to construct calibration datasets with varying reasoning lengths. However, for difficult questions, the set of responses often includes incorrect answers. Consequently, we first analyze whether incorrect answers can also serve as effective calibration data. Following prior practice, we select the first correct response as the true sample and the first incorrect response as the false sample. These are then used separately as calibration data for pruning. As shown in Table 1, pruning with incorrect samples results in lower performance compared to using correct samples. Since they are generated by the dense model, incorrect responses still contribute positively to pruning, indicating that self-generated reasoning data, regardless of correctness, can be useful for pruning. However, since incorrect responses fail to reach the correct final answer, they inherently contain flawed reasoning paths and introduce a certain level of noise. As a result, the overall quality of these samples is inherently lower compared to that of correct responses.

Among the correct responses, we sort them based on their token lengths and select the shortest, median, and longest ones to construct three reasoning datasets of varying lengths, denoted as *Short*, *Medium*, and *Long*, respectively. Additionally, we include *Random*, which randomly selects samples from the correct responses. As shown in Table 1, longer reasoning does not necessarily lead to better pruning performance. The Medium sample outperforms both Short and Long by 2.6% and 1.8% on average, respectively. This indicates that **moderately long reasoning tends to serve as the most effective calibration data, while overly short or excessively long responses may degrade pruning performance.** A proper reasoning length strikes a balance: it captures sufficient intermediate thinking steps without introducing unnecessary verbosity, thus yielding better results.

4 Method

Based on the empirical study presented in Section 3, we propose a calibration data construction strategy for pruning LRMs. Liu et al. (2025) observed that channel magnitudes and activation distributions exhibit highly similar patterns across reasoning datasets from different domains (e.g., math, code). Therefore, we consider that using a single domain is sufficient for constructing calibration data. Specifically, for a problem q from data source $\mathcal{D}$, we perform n independent sampling runs using the dense model, resulting in a response set $\mathcal{R} = \{r_1, r_2, r_3, .., r_n\}$. Let k denote the number of correct responses in $\mathcal{R}$. We compute the correctness ratio as $c = \frac{k}{n}$. If $c > \tau$, where τ is a difficulty threshold, we discard the problem q from the dataset. If $c \leq \tau$, we retain q and proceed to filter $\mathcal{R}$. We first remove incorrect responses based on exact answer matching or pattern verification. Then, we further filter out low-quality responses, such as those that are repetitive, lack coherent reasoning, or do not conform to the expected answer format, resulting in a refined response set $\mathcal{R}^{'} \subseteq \mathcal{R}$. From the filtered set $\mathcal{R}^{'}$, we select the response with the median token length as the final target response for q. Formally, let $\mathcal{R}' = \{r_1, r_2, ..., r_{n'}\}$ be sorted such that their token lengths satisfy $l_1 \leq l_2 \leq \cdots \leq l_{n'}$, where l_i = length(r_i). We define the median response as $r_{median} = r_m$, where $m = \left\lfloor \frac{n'}{2} \right\rfloor$. Let the total calibration data length be l, and the length of question q be l_q. We define the available length for response as $l' = l - l_q$. The final target response $\hat{r}$ for question q is selected as:

$$\hat{r} = \begin{cases} r_{median}, & \text{if } \ell_m \leq l', \\ \arg\max_{\substack{r_i \in \mathcal{R}' \\ \ell_i \leq l}} \ell_i, & \text{if } \ell_m > l' \text{ and } \exists \ell_i \leq l', \\ \arg\min_{r_i \in \mathcal{R}'} \ell_i, & otherwise. \end{cases} \tag{1}$$

When the median-length response is shorter than or equal to l', it is chosen directly. If the median response is longer than l', we select the longest response within the length limit l'. If no response is shorter than or equal to l', we fall back to the shortest response available. This strategy balances verbosity and conciseness, and avoids selecting responses that are either too short to reflect reasoning or too long to be practical. The resulting set of problem–response pairs constitutes our selective self-generated reasoning (SSGR) calibration data.

5 Experiments

5.1 Experimental Details

To evaluate the effectiveness of our proposed calibration data construction method, we apply it to several LRMs, including DeepSeek-R1-Distill-Qwen-7B and DeepSeek-R1-Distill-Llama-8B. As

Model	Method	Data	GSM8K	MATH500	AIME	Olympiad	GPQA	AVG
DeepSeek-R1-Distill-Qwen-7B	Dense		93.0	92.3	43.8	56.0	40.8	65.2
	SparseGPT	C4	89.4	78.3	21.7	38.7	26.6	50.9
		MathR	90.8	86.7	31.3	46.5	24.4	55.9
		SGR	**91.6**	89.0	34.3	50.6	34.8	60.1
		SSGR	91.1	**89.9**	**39.2**	**51.1**	**38.8**	**62.0**
	Wanda	C4	88.1	73.5	13.3	35.1	16.9	45.4
		MathR	90.0	84.6	27.5	45.5	26.9	54.9
		SGR	**90.5**	86.3	30.0	46.3	30.1	56.6
		SSGR	90.4	**87.3**	**32.5**	**48.3**	**31.7**	**58.0**
DeepSeek-R1-Distill-Llama-8B	Dense		91.6	90.3	45.4	54.4	42.4	64.8
	SparseGPT	C4	76.4	56.0	5.8	27.7	14.1	36.0
		MathR	80.4	70.6	19.6	38.8	9.5	43.8
		SGR	84.2	76.0	22.5	41.2	17.3	48.2
		SSGR	**84.8**	**77.1**	**26.3**	**41.5**	**18.2**	**49.6**
	Wanda	C4	69.0	44.8	5.0	21.6	8.6	29.8
		MathR	74.8	58.8	8.3	27.7	9.6	35.8
		SGR	76.7	61.8	**10.0**	28.6	**11.4**	37.7
		SSGR	**76.8**	**63.4**	**10.0**	**29.9**	10.9	**38.2**

Table 2: Pruning performance under a 50% sparsity ratio with different calibration data on DeepSeek-R1-Distill-Qwen-7B and DeepSeek-R1-Distill-Llama-8B. MathR, SGR, and SSGR refer to Math Reasoning, Self-Generated Reasoning, and our Selective Self-Generated Reasoning Data, respectively. Olympiad and GPQA correspond to OlympiadBench and GPQA Diamond. The best results are shown in bold.

described in Section 3.1, we use pre-training data, math reasoning data, and self-generated reasoning data as baseline sources for calibration data, with C4 selected to represent the pre-training data. We adopt two widely used pruning methods, SparseGPT and Wanda, to prune the dense models. In our main experiments, we report model performance under a 50% sparsity setting. The pruned models are evaluated on five reasoning benchmarks: GSM8K, MATH500, OlympiadBench, AIME, and GPQA. During the construction of the selective self-generated reasoning data, MathR is chosen as the data source, and 16 responses are sampled for each problem. We set the difficulty threshold to 0.75 to retain a broader pool of candidate samples.

5.2 Overall Performance

We report the main results in Table 2. Overall, our calibration data construction method outperforms other baseline calibration data on complex reasoning benchmarks, and is compatible with different pruning methods. On DeepSeek-R1-Distill-Qwen-7B, the Selective Self-Generated Reasoning data (SSGR) achieves an average performance improvement of 11.9% to 13.6% over pre-training data (C4) and 3.1% to 6.1% over math reasoning data. This clearly demonstrates the effectiveness of using self-generated reasoning data for calibration. Moreover, SSGR brings an additional 1.4% to 1.9% average gain over the unfiltered Self-Generated Reasoning (SGR) data, highlighting the necessity of our reasoning data selection strategy. On DeepSeek-R1-Distill-Llama-8B, using SSGR as the calibration data also outperforms all other baseline data. SSGR improves performance by 8.4% to 13.6% over C4 and by 2.4% to 5.8% over MathR, and also yields a 0.5% to 1.4% gain compared to SGR. However, it is evident that DeepSeek-R1-Distill-Llama-8B experiences a more significant performance drop.When using SparseGPT with SSGR as the calibration data, it suffers a 15.2% performance loss, whereas DeepSeek-R1-Distill-Qwen-7B only shows a 3.2% loss. Using Wanda results in even greater performance degradation. This suggests that DeepSeek-R1-Distill-Llama-8B is more difficult to compress, and improvements from calibration data alone are insufficient; more effective pruning techniques are necessary to achieve better results.

The pruning performance gains from self-generated reasoning data can be attributed to the fact that the generated responses guide the model to retain similar reasoning patterns during pruning, thereby reducing the discrepancy between the pruned and dense models. Moreover, by selecting responses with higher difficulty and appropriate length, we ensure the data better serves as effective calibration data for pruning LRMs. In addition, although the selected calibration data is derived

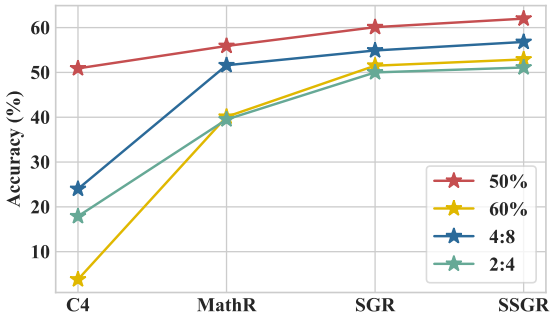

Figure 4: Pruning performance under various pruning settings with different calibration data.

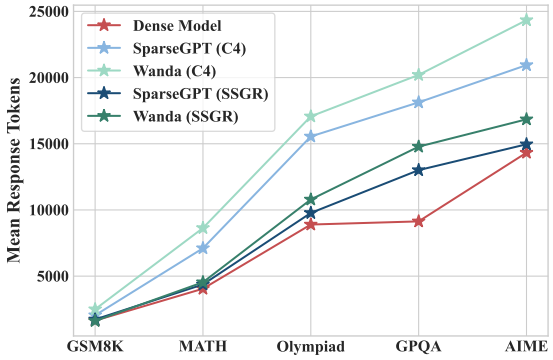

Figure 5: Average response length of pruned LRMs across different benchmarks.

entirely from mathematical problems, the pruned models still exhibit strong performance on out-of-domain tasks such as GPQA, indicating the generalizability of our data construction method.

6 Discussion

6.1 Is the Calibration Data Construction Method Applicable to Other Pruning Settings?

We further validate the effectiveness of the selective self-generated reasoning (SSGR) data under a broader range of pruning settings. As shown in Figure 4, when applying SparseGPT with unstructured 60% pruning and semi-structured 4:8 and 2:4 sparsity settings, SSGR consistently outperforms all baseline calibration data. Notably, at higher sparsity levels and under semi-structured pruning, using pre-training data for calibration leads to a sharp drop in pruning performance. This further demonstrates that pre-training data is ill-suited for calibrating pruned LLMs. In contrast, calibration with SGR data significantly mitigates this issue, yielding a 30.9%–47.7% improvement over C4. Moreover, the SSGR subset derived from SGR offers an additional performance gain of 1.1%–1.9% over SGR. This further demonstrates the strong generalizability of SSGR across various settings.

6.2 How Does Pruning Affect the Generation Length of LRMs?

Reasoning length is a key factor affecting the efficiency of inference in large reasoning models. While pruning reduces the model size by decreasing the number of parameters, its actual impact on inference speed remains uncertain. In Figure 5, we report average response lengths of pruned LRMs. On simple reasoning tasks, the pruned models produce outputs nearly identical in length to those of the dense model. However, for complex tasks, pruned models tend to generate longer responses. Pruned LRMs calibrated with SSGR data produce significantly shorter outputs compared to those calibrated with C4. Pruned LRMs with less performance degradation tend to generate outputs closer in length to the dense model. Therefore, in practical applications, it is crucial to minimize the performance degradation of pruned LRMs, not only to ensure better pruning effectiveness but also to avoid generating excessively long responses that can lead to inference delays. Recent studies (Hou et al., 2025; Yi and Wang, 2025) have also explored methods to compress reasoning length. It is worth noting that model compression and chain-of-thought (CoT) compression are orthogonal and can be applied together. This opens up the possibility of building reasoning models that are both parameter-efficient and capable of fast inference.

7 Conclusion

In this work, we conduct the first empirical investigation into pruning large reasoning models. Using pre-training data directly as calibration input for pruning results in a substantial degradation in the reasoning performance of LRMs. In contrast, leveraging self-generated reasoning data for calibration notably enhances pruning performance. We further observe that self-generated reasoning data with higher difficulty and moderate length serves as ideal calibration data. Building on these insights, we propose a selective self-generated reasoning (SSGR) data construction strategy to provide effective calibration data for pruning LRMs. Experiments on the DeepSeek-R1-Distill-Qwen and DeepSeek-R1-Distill-Llama model series show that our SSGR strategy significantly outperforms baseline calibration data, providing practical insights and guidance for pruning large reasoning models.

Limitations

Although our proposed Selective Self-Generated Reasoning (SSGR) data construction strategy has made notable progress in pruning LRMs, there are still some limitations:

- Due to computational resource constraints, we conduct experiments only on LLMs up to 14B.
- This work focuses exclusively on textual reasoning models. We have not yet extended our method to multimodal reasoning or agent scenarios, which we leave for future exploration.

References

Abhinav Bandari, Lu Yin, Cheng-Yu Hsieh, Ajay Jaiswal, Tianlong Chen, Li Shen, Ranjay Krishna, and Shiwei Liu. 2024. Is c4 dataset optimal for pruning? an investigation of calibration data for llm pruning. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 18089–18099.

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

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. 2025b. Do not think that much for 2+3=? on the overthinking of o1-like llms. *Preprint*, arXiv:2412.21187.

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

Elia Cunegatti, Leonardo Lucio Custode, and Giovanni Iacca. 2025. Zeroth-order adaptive neuron alignment based pruning without re-training. In *Sparsity in LLMs (SLLM): Deep Dive into Mixture of Experts, Quantization, Hardware, and Inference*.

Elias Frantar and Dan Alistarh. 2023. Sparsegpt: Massive language models can be accurately pruned in one-shot. In *International Conference on Machine Learning*, pages 10323–10337. PMLR.

Shangqian Gao, Chi-Heng Lin, Ting Hua, Zheng Tang, Yilin Shen, Hongxia Jin, and Yen-Chang Hsu. 2024. DISP-LLM: Dimension-independent structural pruning for large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

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

Chaoqun He, Renjie Luo, Yuzhuo Bai, Shengding Hu, Zhen Thai, Junhao Shen, Jinyi Hu, Xu Han, Yujie Huang, Yuxiang Zhang, et al. 2024. Olympiadbench: A challenging benchmark for promoting agi with olympiad-level bilingual multimodal scientific problems. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3828–3850.

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. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

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*.

Ivan Ilin and Peter Richtarik. 2025. Thanos: A block-wise pruning algorithm for efficient large language model compression. *arXiv preprint arXiv:2504.05346*.

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

Yixin Ji, Yang Xiang, Juntao Li, Qingrong Xia, Ping Li, Xinyu Duan, Zhefeng Wang, and Min Zhang. 2025. Beware of calibration data for pruning large language models. In *The Thirteenth International Conference on Learning Representations*.

Jun Kong, Xinge Ma, Jin Wang, and Xuejie Zhang. 2025. Sample-aware adaptive structured pruning for large language models. *Preprint*, arXiv:2503.06184.

Guanchen Li, Yixing Xu, Zeping Li, Ji Liu, Xuanwu Yin, Dong Li, and Emad Barsoum. 2025. T\'yr-the-pruner: Unlocking accurate 50% structural pruning for llms via global sparsity distribution optimization. *arXiv preprint arXiv:2503.09657*.

Jeffrey Li, Alex Fang, Georgios Smyrnis, Maor Ivgi, Matt Jordan, Samir Yitzhak Gadre, Hritik Bansal, Etash Kumar Guha, Sedrick Keh, Kushal Arora, Saurabh Garg, Rui Xin, Niklas Muennighoff, Reinhard Heckel, Jean Mercat, Mayee F Chen, Suchin Gururangan, Mitchell Wortsman, Alon Albalak, Yonatan Bitton, Marianna Nezhurina, Amro Kamal Mohamed Abbas, Cheng-Yu Hsieh, Dhruba Ghosh, Joshua P Gardner, Maciej Kilian, Hanlin Zhang, Rulin Shao, Sarah M Pratt, Sunny Sanyal, Gabriel Ilharco, Giannis Daras, Kalyani Marathe, Aaron Gokaslan, Jieyu

Zhang, Khyathi Chandu, Thao Nguyen, Igor Vasiljevic, Sham M. Kakade, Shuran Song, Sujay Sanghavi, Fartash Faghri, Sewoong Oh, Luke Zettlemoyer, Kyle Lo, Alaaeldin El-Nouby, Hadi Pouransari, Alexander T Toshev, Stephanie Wang, Dirk Groeneveld, Luca Soldaini, Pang Wei Koh, Jenia Jitsev, Thomas Kollar, Alex Dimakis, Yair Carmon, Achal Dave, Ludwig Schmidt, and Vaishaal Shankar. 2024a. Datacomp-LM: In search of the next generation of training sets for language models. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

Wei Li, Lujun Li, Mark G. Lee, and Shengjie Sun. 2024b. Adaptive layer sparsity for large language models via activation correlation assessment. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2024. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Ruikang Liu, Yuxuan Sun, Manyi Zhang, Haoli Bai, Xianzhi Yu, Tiezheng Yu, Chun Yuan, and Lu Hou. 2025. Quantization hurts reasoning? an empirical study on quantized reasoning models. *arXiv preprint arXiv:2504.04823*.

Haiquan Lu, Yefan Zhou, Shiwei Liu, Zhangyang Wang, Michael W Mahoney, and Yaoqing Yang. 2024. Alphapruning: Using heavy-tailed self regularization theory for improved layer-wise pruning of large language models. *Advances in Neural Information Processing Systems*, 37:9117–9152.

Haotian Luo, Li Shen, Haiying He, Yibo Wang, Shiwei Liu, Wei Li, Naiqiang Tan, Xiaochun Cao, and Dacheng Tao. 2025a. O1-pruner: Length-harmonizing fine-tuning for o1-like reasoning pruning. *Preprint*, 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. 2025b. Deepscaler: Surpassing o1-preview with a 1.5b model by scaling rl. Notion Blog.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. 2023. Llm-pruner: On the structural pruning of large language models. *Advances in neural information processing systems*, 36:21702–21720.

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*.

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*.

OpenAI. 2024. Introducing openai o1 preview.

OpenAI. 2025. Openai o3-mini system card.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2024. GPQA: A graduate-level google-proof q&a benchmark. In *First Conference on Language Modeling*.

ByteDance Seed, Yufeng Yuan, Yu Yue, Mingxuan Wang, Xiaochen Zuo, Jiaze Chen, Lin Yan, Wenyuan Xu, Chi Zhang, Xin Liu, et al. 2025. Seed-thinking-v1. 5: Advancing superb reasoning models with reinforcement learning. *arXiv preprint arXiv:2504.13914*.

Ayan Sengupta, Siddhant Chaudhary, and Tanmoy Chakraborty. 2025. You only prune once: Designing calibration-free model compression with policy learning. In *The Thirteenth International Conference on Learning Representations*.

Sungbin Shin, Wonpyo Park, Jaeho Lee, and Namhoon Lee. 2024. Rethinking pruning large language models: Benefits and pitfalls of reconstruction error minimization. *arXiv preprint arXiv:2406.15524*.

Oliver Sieberling, Denis Kuznedelev, Eldar Kurtic, and Dan Alistarh. 2024. Evopress: Towards optimal dynamic model compression via evolutionary search. *arXiv preprint arXiv:2410.14649*.

Mingjie Sun, Zhuang Liu, Anna Bair, and J Zico Kolter. 2024. A simple and effective pruning approach for large language models. In *The Twelfth International Conference on Learning Representations*.

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

NovaSky Team. 2025a. Sky-t1: Train your own o1 preview model within $450. https://novasky-ai.github.io/posts/sky-t1. Accessed: 2025-01-09.

Qwen Team. 2025b. Qwq-32b: Embracing the power of reinforcement learning.

Zhihai Wang, Jie Wang, Jilai Pan, Xilin Xia, Huiling Zhen, Mingxuan Yuan, Jianye Hao, and Feng Wu. 2025. Accelerating large language model reasoning via speculative search. *Preprint*, arXiv:2505.02865.

Liang Wen, Yunke Cai, Fenrui Xiao, Xin He, Qi An, Zhenyu Duan, Yimin Du, Junchen Liu, Lifu Tang, Xiaowei Lv, et al. 2025. Light-r1: Curriculum sft, dpo and rl for long cot from scratch and beyond. *arXiv preprint arXiv:2503.10460*.

Miles Williams and Nikolaos Aletras. 2024. On the impact of calibration data in post-training quantization and pruning. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10100–10118.

Miles Williams, George Chrysostomou, and Nikolaos Aletras. 2024. Self-calibration for language model quantization and pruning. *arXiv preprint arXiv:2410.17170*.

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

Jingyang Yi and Jiazheng Wang. 2025. Shorterbetter: Guiding reasoning models to find optimal inference length for efficient reasoning. *arXiv preprint arXiv:2504.21370*.

Lu Yin, You Wu, Zhenyu Zhang, Cheng-Yu Hsieh, Yaqing Wang, Yiling Jia, Gen Li, Ajay Jaiswal, Mykola Pechenizkiy, Yi Liang, et al. 2024. Outlier weighed layerwise sparsity (owl): A missing secret sauce for pruning llms to high sparsity. In *The Forty-First International Conference on Machine Learning*.

Jintian Zhang, Yuqi Zhu, Mengshu Sun, Yujie Luo, Shuofei Qiao, Lun Du, Da Zheng, Huajun Chen, and Ningyu Zhang. 2025. Lightthinker: Thinking step-by-step compression. *Preprint*, arXiv:2502.15589.

Yingtao Zhang, Haoli Bai, Haokun Lin, Jialin Zhao, Lu Hou, and Carlo Vittorio Cannistraci. 2024. Plug-and-play: An efficient post-training pruning method for large language models. In *The Twelfth International Conference on Learning Representations*.

Haizhong Zheng, Xiaoyan Bai, Xueshen Liu, Zhuoqing Mao, Beidi Chen, Fan Lai, and Atul Prakash. 2024. Learn to be efficient: Build structured sparsity in large language models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*.

Changhai Zhou, Qian Qiao, Weizhong Zhang, and Cheng Jin. 2025. Large language model compression with global rank and sparsity optimization. *arXiv preprint arXiv:2505.03801*.

A More Experiments

To further demonstrate the effectiveness of our method, we conduct experiments on Qwen3-14B. As shown in Table 5, SSGR consistently achieves the best performance, significantly outperforming C4 and MathR. Moreover, we apply our approach to the recent pruning method OWL (Yin et al., 2024). As reported in Table 6, SSGR remains effective, demonstrating strong generalization across different pruning strategies.

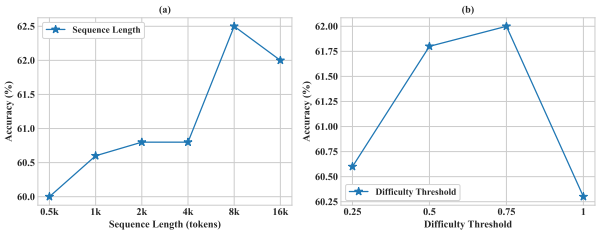

Figure 6: Performance under different sequence lengths and different difficulty thresholds.

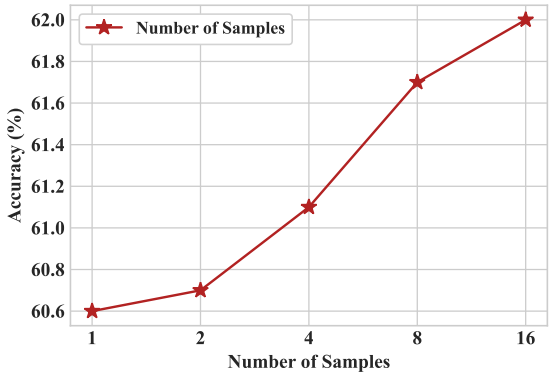

Figure 7: Performance under different numbers of samples.

B Ablation Study

The results are shown in Figure 6. As the sequence length increases, pruning performance generally improves. However, performance at 16k is slightly lower than at 8k, which may be because the average length of the calibration dataset is closer to 8k, leading to better alignment and performance. When no difficulty threshold is applied ($\tau = 1$), pruning performance drops significantly. Conversely, setting the threshold too low ($\tau = 0.25$) leads to too few reasoning samples and insufficient diversity, which harms performance. A moderate difficulty threshold ($\tau = 0.75$ or 0.5) achieves the best results by balancing difficulty and diversity.

C Discussion on Number of Samples

We discuss the effect of the number of samples. As shown in Figure 8, increasing the number of samples consistently improves the reasoning performance of the pruned model. This highlights the importance of selecting an appropriate answer from multiple generated responses. Notably, under resource-constrained settings, the model still achieves strong performance even with only a single sampled response per question. This further demonstrates the critical role of self-generated calibration data in pruning reasoning models.

Data	GSM8K	MATH500	AIME	Olympiadbench	GPQA	AVG
C4	89.4	78.3	21.7	38.7	26.6	50.9
Wikipedia	87.1	72.6	17.5	33.7	15.5	45.3
DCLM	89.1	77.7	24.2	38.2	24.6	50.8
MathR	90.8	86.7	31.3	46.5	24.4	55.9
SGR	91.6	89.0	34.3	50.6	34.8	60.1
SGR *w/o thinking*	90.4	84.9	31.3	46.4	25.6	55.7

Table 3: Pruning performance of different datasets under unstructured 50% sparsity on SparseGPT

Difficulty Level	GSM8K	MATH500	AIME	Olympiadbench	GPQA	AVG	Token Length
Easy	91.3	88.9	30.8	49.6	30.8	58.3	2005
Meidum	91.0	**90.5**	35.4	50.3	32.3	59.9	3972
Hard	**91.5**	89.8	**37.1**	**51.3**	**39.4**	**61.8**	7375

Table 4: Pruning performance using calibration data of varing difficulty levels under unstructured 50% sparsity on SparseGPT

D Discussion on alternative CoT data generation methods

As shown in Table 7, we generated reasoning data using Qwen3-14B (a stronger reasoning model). However, the pruning performance was inferior compared to data self-generated by DeepSeek-R1-Distill-Qwen-7B, particularly in terms of generalization, with a 10.1% drop on GPQA. This further highlights the necessity of using self-generated reasoning data for effective pruning.

E Case Study

As shown in Figure 8, we present a case study comparing the reasoning behaviors of pruned models calibrated with different datasets. The results demonstrate notable differences in the coherence and correctness of the generated reasoning, underscoring the importance of high-quality calibration data for pruning reasoning models.

Data	GSM8K	MATH500	AIME	Olympiadbench	GPQA	AVG
Qwen3-14B (SparseGPT)						
C4	94.7	84.8	28.3	49.3	45.5	60.5
MathR	94.6	92.4	51.8	57.6	47.0	68.7
SGR	**94.9**	92.4	55.0	59.3	52.0	70.7
SSGR	94.7	**93.2**	**58.4**	**60.4**	**55.1**	**72.4**

Table 5: Pruning performance of different datasets under unstructured 50% sparsity on Qwen3-14B

Data	GSM8K	MATH500	AIME	Olympiadbench	GPQA	AVG
DeepSeek-R1-Distill-Qwen-7B (OWL)						
Dense	93.0	92.3	43.8	56.0	40.8	65.2
C4	89.0	75.1	17.1	36.3	20.3	47.6
MathR	90.4	85.2	28.6	45.8	25.8	55.2
SGR	91.0	87.4	32.3	48.1	32.7	58.3
SSGR	**91.1**	**88.2**	**35.8**	**50.1**	**35.6**	**60.2**

Table 6: Pruning performance of different datasets under unstructured 50% sparsity on OWL

Generation Source	GSM8K	MATH500	AIME	Olympiadbench	GPQA	AVG
DS-Distill-Qwen-7B	91.1	89.9	39.2	51.1	38.8	**62.0**
Qwen3-14B	89.7	87.9	35.8	49.4	28.7	58.3

Table 7: Pruning performance of different generation source.

Question

What is the least positive integer multiple of 30 that can be written with only the digits 0 and 2?

Pruned model (C4)'s Response:

Let's think step by step. <think> Okay, so I need to find the smallest positive multiple of 30 that only uses the digits 0 and 2. Hmm, let's start by recalling that a multiple of 30 must be divisible by both 3 and 10. That means the number has to end with a 0 (since it's divisible by 10) and the sum of its digits must be a multiple of 3 (divisible by 3). \First, since the number can only have 0s and 2s, the last digit has to be 0. So whatever number I come up with, it must end with a 0. Also, the digits can only be 0 or 2, so the number can't have any other digits. That simplifies things a bit because I don't have to worry about other digits interfering with divisibility. Since the number must end with 0, the last digit is fixed. The rest of the digits can be 0 or 2. The digit sum (sum of all digits) must be a multiple of 3. So, for example, if the number is 20, the digit sum is 2, which isn't divisible by 3. If it's 200, the digit sum is 2, still not divisible by 3. If it's 220, the digit sum is 4, not divisible by 3. 2020: sum is 4, same problem. 2220: sum is 6, which is divisible by 3. Wait, 2220 is a multiple of 30? Let me check.\\textcolor{blue}{2220 divided by 30 is 74, so yes, 2220 is 30 times 74.} But is there a smaller number? Let's see. Let's check numbers with fewer digits first.....\But wait, maybe there's a smaller number with more digits but smaller in value? For example, 2220 is 2220, but maybe a number like 2020 is smaller? Wait, 2020 is 2020, which is smaller than 2220. But 2020's digit sum is 2 + 0 + 2 + 0 = 4, which isn't divisible by 3. So even though 2020 is smaller, it's not a multiple of 30. Similarly, 220 is 220, which is smaller than 2220, but digit sum 4. So 2220 is the first one that works. But let's check if there's a smaller number with more digits but smaller in value..... So maybe 2220 is the answer. \textcolor{red}{But wait, let's check if there's a three-digit number. Wait, three digits: 200, 220, 2220?}...Thus, the angle between the lines is \boxed{**220**} degrees}.

Pruned model (SSGR)'s Response:

Let's think step by step. Okay, so I need to find the angle between two lines given by these equations. Let me start by understanding what each line is. The problem says that the first line is defined by 2x = 3y = -z, and the second one is 6x = -y = -4z. Hmm, okay. So both are lines in 3D space, right? Because they have three variables x, y, z. So each equation here is like a parametric equation for the line. \First, maybe I should find the direction vectors of each line. Because once I have the direction vectors, I can use the dot product formula to find the angle between them. That makes sense. So direction vectors are key here. Let me recall that for a line defined by equations like ax = by = cz, the direction vector is (a, b, c). Wait, is that right? Let me check. ..... Alternatively ..... \Alternatively, ... Let me verify. ..... Wait..... Wait, ... But this seems complicated, and the problem probably expects the answer to be 90 degrees. But why? Because the dot product is zero. So maybe the answer is 90 degrees. So perhaps the problem is designed so that the direction vectors are perpendicular. Therefore, the angle is 90 degrees. So maybe the answer is 90 degrees. Therefore, boxed answer is \boxed{90}. Wait, but let me check again. Maybe I made a mistake in direction vectors. Wait, if the direction vectors are perpendicular, then the angle is 90 degrees. So if the dot product is zero, then yes. So in that case, the answer is 90 degrees. Therefore, the angle is 90 degrees. So I think that's correct. Therefore, the answer is \boxed{90} degrees.**Final Answer**\boxed{90} </think>\.....Thus, the angle between the lines is \boxed{**90**} degrees.

Figure 8: Case study comparing the reasoning behaviors of pruned models calibrated with C4 and SSGR.