Optimal Sparsity of Mixture-of-Experts Language Models for Reasoning Tasks

Taishi Nakamura [1 2] **Satoki Ishikawa** [1] **Masaki Kawamura** [1] **Takumi Okamoto** [1 2] **Daisuke Nohara** [1]
Jun Suzuki [3 2 4] **Rio Yokota** [1 2]

Abstract

Empirical scaling laws have driven the evolution of large language models (LLMs), yet their coefficients shift whenever the model architecture or data pipeline changes. Mixture-of-Experts (MoE) models, now standard in state-of-the-art systems, introduce a new sparsity dimension that current dense-model frontiers overlook. We investigate how MoE sparsity influences two distinct capability regimes: memorization and reasoning. We train families of MoE Transformers that systematically vary total parameters, active parameters, and top-k routing while holding the compute budget fixed. For every model we record pre-training loss, downstream task loss, and task accuracy, allowing us to separate the train-test generalization gap from the loss-accuracy gap. Memorization benchmarks improve monotonically with total parameters, mirroring training loss. By contrast, reasoning performance saturates and can even regress despite continued gains in both total parameters and training loss. Altering top-k alone has little effect when active parameters are constant, and classic hyperparameters such as learning rate and initialization modulate the generalization gap in the same direction as sparsity. Neither post-training reinforcement learning (GRPO) nor extra test-time compute rescues the reasoning deficit of overly sparse models.

1. Introduction

The recent evolution of large language models (LLMs) has been driven by empirical scaling laws that link training loss to model size, dataset size, and compute budget. Kaplan et al. showed that these laws hold across seven orders of magnitude, establishing them as a reliable extrapolation tool for dense Transformers (Kaplan et al., 2020). Subsequent work by Hoffmann et al. demonstrated that scaling curves can be inverted to choose the compute-optimal combination of parameters and tokens for a fixed budget (Hoffmann et al., 2022). Together, these results have made scaling analysis a cornerstone of model planning at both academic and industrial labs.

Yet the coefficients of the scaling laws are not universal. Highly expressive models trained under different optimizers or architectures often follow the same loss trajectory but diverge substantially on downstream reasoning benchmarks (Liu et al., 2023). Brandfonbrener et al. extend the classic laws with loss-to-loss prediction, showing that the mapping between training and test distributions admits its own power law when the distributions differ substantially (Brandfonbrener et al., 2025). These observations imply that optimal budgets must be re-estimated whenever we modify the model or the data pipeline.

A particularly compelling architectural modification is the Mixture-of-Experts (MoE) paradigm, offering high capacity at fixed FLOPs by routing each token through a sparse subset of experts (Shazeer et al., 2017; Lepikhin et al., 2021; Fedus et al., 2021). Modern flagship models, e.g., Gemini 2.5/1.5 Pro (Gemini Team, 2025; 2024), DeepSeek-V2/V3 (DeepSeek-AI, 2024; 2025b), Qwen-2.5/3 (Qwen, 2025; Qwen Team, 2025), and Mixtral-8x22B (Jiang et al., 2024), now rely on MoE as a de-facto standard for economical scaling. Abnar et al. derive a parameters-vs-FLOPs frontier and locate an optimal sparsity for a given compute budget (Abnar et al., 2025). These findings emphasize that the classical dense-model frontier is an incomplete picture, and one must account for architectural knobs such as MoE sparsity and top-k routing.

Furthermore, loss-based scaling curves do not always predict the performance on downstream tasks. Jelassi et al. report that increasing MoE sparsity improves memorization benchmarks, but saturates for reasoning performance (Jelassi et al., 2025). However, the *Mixture of Parrots* paper (Jelassi et al., 2025) only explores the number of active vs. total

[1]Institute of Science Tokyo, Tokyo, Japan [2]Research and Development Center for Large Language Models, National Institute of Informatics, Tokyo, Japan [3]Tohoku University, Sendai, Japan [4]RIKEN, Tokyo, Japan. Correspondence to: Taishi Nakamura <taishi@rio.scrc.iir.isct.ac.jp>, Rio Yokota <rioyokota@rio.scrc.iir.isct.ac.jp>.

The second AI for MATH Workshop at the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. Copyright 2025 by the author(s).

parameters, ignoring the effect of routing strategies beyond standard top-2 routing. They also do not consider the effect of reinforcement learning and test-time compute on their reasoning benchmarks. Evaluating reasoning performance immediately after pre-training overlooks both the benefits of post-training adaptation and the leverage of additional test-time compute. Post-training methods such as GRPO, which use reinforcement signals to encourage coherent chain-of-thought generation, sharpen a model's reasoning on complex tasks (OpenAI, 2024b; DeepSeek-AI, 2025a). Beyond these refinements, models can further improve outputs at test time by adopting calibrated decoding strategies that mirror how humans pause to reconsider difficult problems. These test-time approaches not only boost routine benchmark performance but, when properly tuned, substantially enhance multi-step mathematical reasoning, demonstrating that adaptive computing at test time is a powerful complement to both model scale and post-training adaptation.

In this paper, we aim to identify how the optimal sparsity of MoE changes between memorization (TriviaQA, HellaSwag) and reasoning (GSM8K, GSM-Plus) tasks. We train families of MoEs varying not only the total vs. active parameters, but also the number of top-k experts. For each model, we measure the loss on the pre-training data, the task loss on the downstream benchmarks, and the accuracy on those benchmarks. This allows us to disentangle the generalization gap between the train vs. test loss, and the gap between loss vs. accuracy. For both memorization and reasoning benchmarks, the train loss decreases monotonically with the total parameters. The task loss and accuracy follow the same monotonic trend as the train loss for memorization benchmarks. In contrast, for reasoning benchmarks, the task loss and accuracy diverge from the monotonic trend as the total parameters increase and training loss decreases. We found that changing the k in top-k routing itself has a negligible effect if the number of active parameters is kept constant. We also consider classic generalization-gap controls by sweeping the learning rate and initialization, and show that their effects align strikingly with the generalization-gap caused by sparsity. This confirms that the gap between the performance on memorization vs. reasoning tasks can be induced not only by sparsity of the MoE, but also classical hyperparameters like learning rate and initialization. We further investigate whether applying GRPO or additional test-time compute could recover the poor reasoning ability of sparser models. Our results show that the gap between memorization and reasoning performance caused by increased sparsity remains unchanged even after GRPO and increased test-time compute. This means that finding the optimal sparsity of the MoE during pre-training is crucial for training a reasoning model under a fixed compute budget.

2. Background and Related Work

2.1. Mixture of Experts

MoE Architecture. Mixture-of-Experts (MoE) networks were introduced by (Jacobs et al., 1991; Jordan & Jacobs, 1994) and later brought to large-scale neural language modeling by Shazeer et al. (2017). Within the Transformer architecture (Vaswani et al., 2017), MoE layers have proven especially effective, scaling to hundreds of billions of parameters while maintaining manageable training costs (Lepikhin et al., 2021; Fedus et al., 2021; Du et al., 2022; Zoph et al., 2022). Consequently, modern state-of-the-art language models, including Gemini 2.5/1.5 Pro (Gemini Team, 2025; 2024) DeepSeek-V2/V3 (DeepSeek-AI, 2024; 2025b), Qwen-2.5/3 (Qwen, 2025; Qwen Team, 2025), and Mixtral-8x22B (Jiang et al., 2024), rely heavily on MoE layers to achieve superior performance under fixed inference budgets. In an MoE layer, a learnable router assigns each token to a sparse subset of experts. Let $\mathbf{x} \in \mathbb{R}^{d_h}$ be a token representation and $\{\text{FFN}(\mathbf{x})_i\}_{i=1}^{n}$ the n feed-forward experts. For top-k routing, the router produces scores $\mathbf{s} = \mathbf{x}^\top \mathbf{W}_{\text{router}} \in \mathbb{R}^n$, and selects the indices $\mathcal{K}$ of the k largest components, then normalizes them: $g(\mathbf{x})_i = \frac{\exp(s_i)}{\sum_{j\in\mathcal{K}} \exp(s_j)}$ if $i \in \mathcal{K}$ and $g(\mathbf{x})_i = 0$ otherwise. The layer output is the weighted sum of the chosen experts: $\mathbf{y} = \sum_{i=1}^{n} g(\mathbf{x})_i \, \text{FFN}(\mathbf{x})_i$. Modern MoE models typically supplement the token-level cross-entropy loss with two auxiliary terms: a load-balancing loss $\mathcal{L}_{\text{LB}}$, which prevents expert collapse (Shazeer et al., 2017), and a router-z loss $\mathcal{L}_{\text{RZ}}$, which penalizes large router logits for better numerical stability and gradient flow (Zoph et al., 2022). The combined training loss is expressed as $\mathcal{L} = \mathcal{L}_{\text{CE}} + \alpha\mathcal{L}_{\text{LB}} + \beta\mathcal{L}_{\text{RZ}}$, where α and β are hyperparameters that control the relative importance of each term in the objective function. This formulation is widely used in recent MoE-based language models and remains unchanged throughout the experiments.

2.2. Scaling Laws of LLMs

Scaling Laws for MoE. Existing scaling laws demonstrate power-law relationships between model performance, parameter count, dataset size, and compute budget (Kaplan et al., 2020; Hoffmann et al., 2022). Scaling laws for MoE models have similarly explored how total parameter count and expert granularity jointly affect scaling behavior (Clark et al., 2022; Ludziejewski et al., 2024). Building on this, Frantar et al. derived sparsity-aware scaling exponents that bridge dense and sparse regimes (Frantar et al., 2024), while Abnar et al. empirically charted the optimal trade-offs between total parameters and FLOPs per token in MoE settings (Abnar et al., 2025). Complementary theoretical and empirical work shows that adding experts tends to improve memorization more than reasoning, motivating new, generalized scaling frameworks that address scaling laws for

reasoning performance (Jelassi et al., 2025).

Task Loss. Since the scaling law for next-token prediction loss does not necessarily align with downstream task loss, it may not be reliable for predicting benchmark performance (Grattafiori et al., 2024). Some work has tried to model downstream accuracy with an exponential curve, but accuracy is only predictable when we average over many tasks and carefully choose which ones to include (Gadre et al., 2024). Another line of research instead first quantifies how downstream task loss scales with parameters and data, then converts predicted losses into accuracy estimates, achieving under two points of absolute error for mid-scale models using minimal extra compute (Bhagia et al., 2024). Prior work observes that downstream task loss relates to pre-training loss, where the shifts depend on the minimal achievable losses determined by the intrinsic complexity and distributional mismatch between the pre-training and downstream datasets (Brandfonbrener et al., 2025). Because scaling laws differ across tasks, the optimal scaling strategy may also vary; for example, knowledge-based QA tasks are "capacity-hungry," benefiting more from larger model sizes, whereas code-related tasks are "data-hungry," benefiting more from increased training data (Roberts et al., 2025).

Overfitting of LLMs. Standard scaling laws imply that one can indefinitely increase model and dataset size without worrying about overfitting; however, if we need to worry about generalization and overfitting, these scaling laws break down (Caballero et al., 2023). While some work has shown that large language models can generalize to mathematical reasoning tasks (Wang et al., 2025), others have found that even the latest state-of-the-art models may still overfit on the GSM8K benchmark (Zhang et al., 2024; Mirzadeh et al., 2024). However, it remains unclear whether this overfitting stems from overtraining or from having too many parameters. For overtraining, empirical studies in data-constrained pre-training show that repeated passes over the same data yield negligible improvements after four epochs (Muennighoff et al., 2023), with validation loss on high-quality scientific text starting to rise at the beginning of the fifth epoch (Taylor et al., 2022), and severe accuracy drops observed under extreme token scarcity when training is extended well beyond four to five epochs (Xue et al., 2023). Outside of pre-training contexts, overtraining has been shown to degrade performance after post-quantization or post-training, despite continued reductions in pre-training loss (Kumar et al., 2025; Springer et al., 2025).

2.3. Post Training and Test-Time Compute (TTC)

Reinforcement Learning (RL) post-training has long been a predominant approach for improving LLMs. Proximal Policy Optimization (PPO) (Schulman et al., 2017) forms the backbone of RLHF pipelines, from the original GPT alignment work (Ouyang et al., 2022) to the GPT-4 family of models (OpenAI, 2024a). More recently, Group Relative Policy Optimization (GRPO) was introduced as a variant of PPO that replaces the value function baseline with a group-relative advantage estimator, thereby improving memory efficiency and stabilizing updates; this approach already powers frontier-scale systems such as DeepSeek-R1, achieving state-of-the-art results on mathematical-reasoning benchmarks (Shao et al., 2024; DeepSeek-AI, 2025a).

Complementary to these training-time advances, scaling *test-time compute* (TTC) offers an orthogonal approach. TTC denotes accuracy gains obtained *without* updating model parameters, simply by allocating more inference resources, e.g., running longer chains of thought (OpenAI, 2024b; Muennighoff et al., 2025b), sampling larger candidate pools (Li et al., 2022; Wang et al., 2023; Brown et al., 2024; Schaeffer et al., 2025), or performing explicit search-and-verify steps (Lightman et al., 2024; Shinn et al., 2024; Snell et al., 2025; Inoue et al., 2025). Among these, *self-consistency*, repeated sampling with majority-vote aggregation, has emerged as a strong TTC baseline (Wang et al., 2023).

3. Experiments

In this section, we empirically demonstrate the scaling of downstream task performance through a systematic investigation of memorization-reasoning benchmarks in MoE LLMs.

3.1. Experimental Setup

We use the Mixtral (Jiang et al., 2024) architecture, a Transformer backbone with RMSNorm (Zhang & Sennrich, 2019), SwiGLU activations (Shazeer, 2020), and rotary positional embeddings (Su et al., 2024). Each feed-forward block is a sparsely gated MoE layer, gated by the dropless token-choice top-k routing (Gale et al., 2023). All models use $L = 16$ layers, following Muennighoff et al. (2025a). We sweep three architectural hyperparameters: (i) the model width $d \in \{512, 1024, 2048\}$; (ii) the number of experts per layer $E \in \{8, 16, 32, 64, 128, 256\}$; and (iii) the top-$k$ experts per token $k \in \{2, 4, 8, 16\}$. Each feed-forward network has a hidden dimension of $2d$. When $d = 512$ and $d = 1024$, we train every combination of E and k. For $d = 2048$, we limit the search to $E \leq 128$ due to computational resource constraints. We train with AdamW (Loshchilov & Hutter, 2019) using a peak learning rate of 4×10^{-4}, a 2k-step linear warm-up followed by cosine decay, and a weight decay of 0.1.

Hyperparameter Study. To isolate optimization effects, we reuse the same 125 B-token corpus. For all HP runs,

we fix $E = 16$, $k = 2$, and train two widths, $d_{\text{model}} \in \{512, 1024\}$, with the same FFN expansion factor 2. We vary (i) LM-head initialization schemes, (ii) peak learning rate, and (iii) AdamW ϵ. Further implementation and environmental details are deferred to Appendix A.3.

Pre-training Datasets. We use a balanced mixture of general-domain and mathematics-centric corpora, totaling 125 B-tokens. High quality web text (43 B) comes from deduplicated DCLM (Zyphra, 2024), the Flan-decontaminated Dolmino subset, and WebInstructFull. Mathematics (32 B) combines OLMo OpenWebMath and Algebraic-Stack (Soldaini et al., 2024), FineMath-4+ (Liu et al., 2024), the MathPile commercial subset (Wang et al., 2024), the math split of Dolmino-Mix-1124 (OLMo, 2025), OpenMathInstruct-1/2 (Toshniwal et al., 2024b;a), StackMathQA, Orca-Math (Mitra et al., 2024), and GSM8K train (Cobbe et al., 2021). STEM Literature & Reference (42 B) consists of arXiv, pes2o, Wikipedia, and Dolma–books (Soldaini et al., 2024). Finally, we add Code from the StackExchange code subset. See Appendix A.1 for complete statistics. See Appendix A.1 for complete statistics.

Evaluation Protocol. We evaluate four capability areas with standard few-shot prompts. General Knowledge and Reasoning: BBH (Suzgun et al., 2023) (3-shot CoT). Mathematical Reasoning: GSM8K (Cobbe et al., 2021) (4-shot) and GSM-Plus (Li et al., 2024) (5-shot CoT). Reading Comprehension: TriviaQA (Joshi et al., 2017) with 4-shot prompting. Commonsense Reasoning: HellaSwag (Zellers et al., 2019), and XWinograd (EN) (Tikhonov & Ryabinin, 2021), each under a 4-shot prompting setup. See Appendix 3 for further details.

3.2. Downstream Performance Does Not Necessarily Improve with Total Parameter Size

In this section, we examine how the expert sparsity in MoE models affects the relationship between pre-training loss and downstream performance. We train a series of models with controlled sparsity levels and measure their performance on the representative downstream tasks. Our analysis shows that while increasing the total number of parameters reduces pre-training loss, downstream task loss on mathematical reasoning worsens beyond a certain model size.

Task Loss Computation. Following Brandfonbrener et al. (2025) and Grattafiori et al. (2024), we compute cross-entropy only over the answer tokens by concatenating the prompt with the ground-truth answer. For multiple-choice datasets (e.g., HellaSwag, TriviaQA) the target sequence is the correct answer string, as in Bhagia et al. (2024). For open-ended mathematics datasets—GSM8K, and GSM-Plus —we likewise compute cross-entropy directly against the ground-truth answer tokens.

Training Loss and Validation Loss. Figure 1 presents the training and validation losses when fixing the top-k/MoE layer width constant and increasing only the number of experts (and hence the total parameter count). As the total parameter count grows, both training and validation losses decrease. Therefore, in terms of pre-training loss, increasing total parameters (thereby raising sparsity) reduces pre-training loss, which is consistent with prior work.

Experiments with Task Loss Next, we examine how the downstream task loss responds to increases in the total parameter count. Figure 2 shows task loss on several benchmarks as we vary only the number of experts, holding both top-k and each MoE layer widths constant. On TriviaQA and HellaSwag, lower pre-training loss reduces task loss, indicating that larger total parameter models yield better results on these datasets. In contrast, for GSM8K and GSM-Plus, further reductions in pre-training loss do not translate into improved task loss; in some cases, the task loss actually worsens. On XWinograd as well, one of the reasoning tasks, we observe a modest downward trend in performance. However, task loss displays considerable variability. We hypothesize that this observation arises because the ground-truth answers are only a few tokens long. These results suggest that, once top-k and layer width are fixed, an optimal number of experts exists for each task, and adding more beyond that point can harm performance on GSM8K and GSM-Plus.

Dependence on Active Parameter. Can we avoid a decline in performance as the total number of experts increases? Figure 2 shows that models with more active parameters begin to overfit at a lower pre-training loss and reach a lower minimum task loss at their optimal expert counts. Consequently, improving results on GSM8K and GSM-Plus requires tuning not only the total number of experts but also the top-k size. Whether a similar trend occurs on other reasoning benchmarks, including code-generation tasks, remains an open question.

Downstream Accuracy. The decline in math-task performance as total parameters increase is not limited to task loss; it also consistently holds for downstream accuracy (Figure 3). For TriviaQA and HellaSwag, accuracy improves monotonically as training loss decreases. By contrast, on GSM8K, further reductions in pre-training loss do not always translate to higher accuracy. When the number of active parameters is held constant, over-optimizing pre-training loss can indeed harm performance. Figure 4 plots benchmark error rate against pre-training loss, including intermediate checkpoints. We can find a sparsity dependence for reasoning-oriented tasks such as GSM8K, XWinograd, and BBH. These results

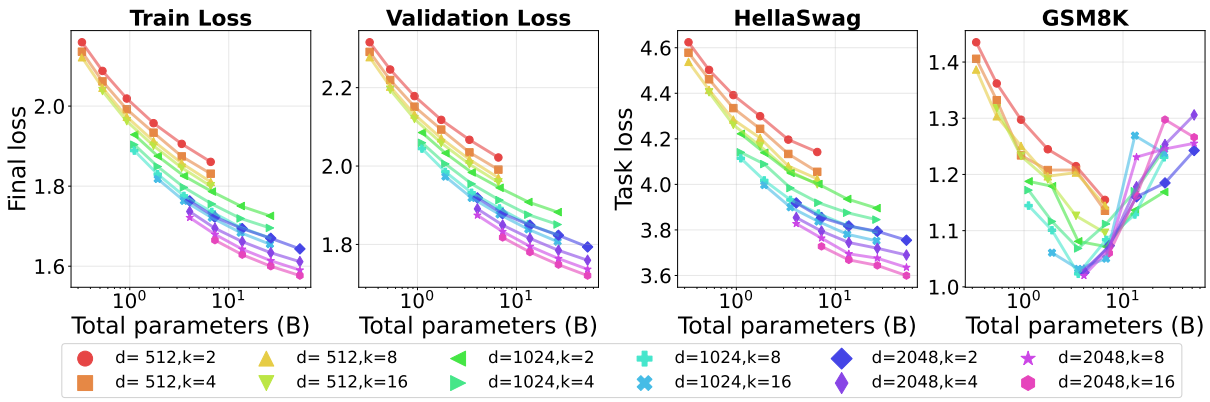

Figure 1. **Although training and validation loss decrease as the total number of parameters grows, the task loss on GSM8K can sometimes worsen with larger models.** Training and validation losses steadily decrease as total or active parameters increase. The HellaSwag task loss follows this scaling trend, whereas GSM8K task loss worsens once total parameters exceed a threshold, yet continues to improve when active parameters are scaled up.

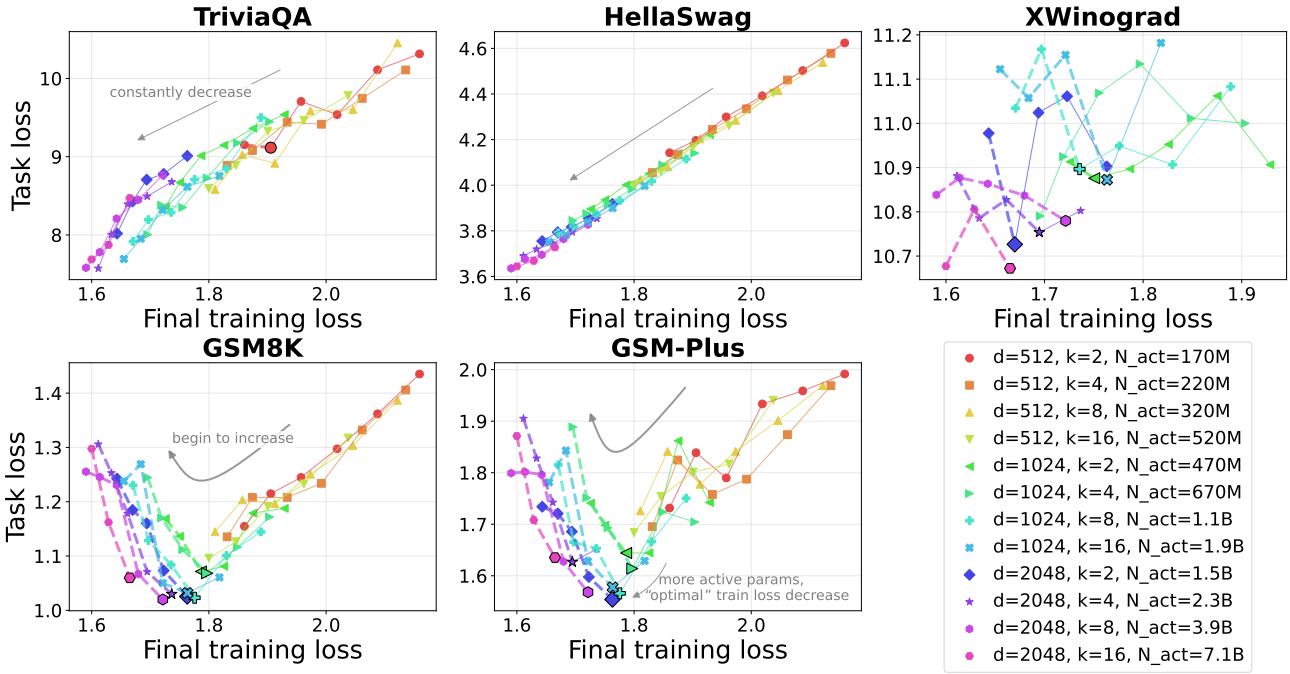

Figure 2. **For GSM8K and GSM-Plus, once the training loss drops below a certain point, the task loss starts to increase.** Results of scaling total parameters by increasing the number of experts, with model width and top-k held constant. For TriviaQA, HellaSwag, and task loss falls monotonically as training loss decreases. By contrast, GSM8K and GSM-Plus show a U-shaped trend: task loss declines with training loss only until a threshold, beyond which further reductions in training loss hurt task performance. That threshold moves lower as active parameter count increases, models with more active parameters achieve a lower optimal task loss. No such active parameters dependence appears for TriviaQA, HellaSwag. On XWinograd, a modest downward trend in performance can still be observed, while this trend is not as clean as the others.

suggest that, for MoE models, downstream accuracy can deviate from the predictions of conventional scaling laws, and these deviations may vary across different tasks.

3.3. Optimal Sparsity for Iso-FLOP Budgets

We next analyze model quality under a constant compute budget, that is, along *IsoFLOP* contours (Hoffmann et al., 2022; Abnar et al., 2025). For a fixed per-token FLOP count, we vary only the *sparsity configuration*: the number of experts E and the top-k value, while holding the hidden dimension and sequence length.

Figure 3. **Downstream accuracy when scaling total parameters via expert count with width and top-k fixed.** TriviaQA and HellaSwag exhibit steadily improving accuracy as pre-training loss decreases, whereas GSM8K shows a non-monotonic trend: further reductions in pre-training loss do not always improve accuracy and can even degrade performance.

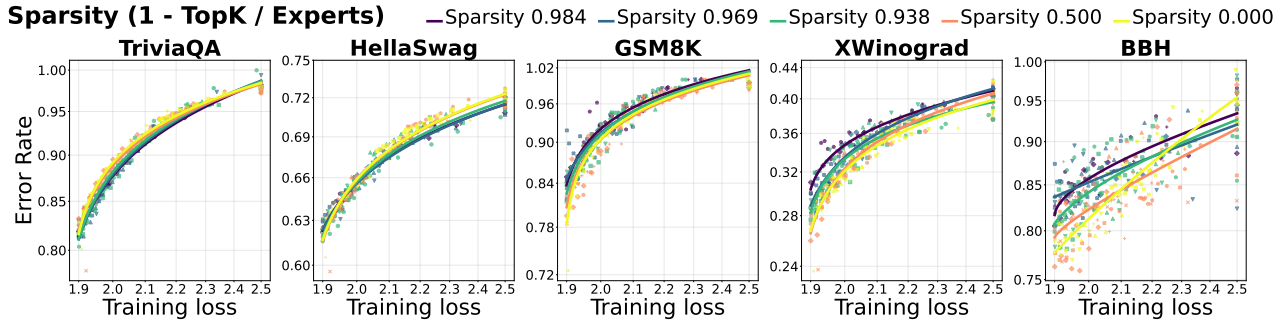

Figure 4. **Effect of sparsity on performance across different tasks** We vary sparsity (1 - top-k/Experts) and plot the relationship between pre-training loss and benchmark error rate, including intermediate checkpoints. For TriviaQA and HellaSwag, the error rate clearly tracks training loss and is largely insensitive to sparsity. In contrast, reasoning-intensive tasks such as GSM8K, XWinograd, and BBH exhibit a strong dependence of error rate on sparsity.

In Figure 5, we plot the task-specific optimal density (i.e. 1-TopK/Experts) against model performance under a fixed FLOPs budget. For QA benchmarks such as TriviaQA and HellaSwag, lower density (higher sparsity) consistently yields lower task loss and higher accuracy. This pattern aligns with prior studies showing that, when FLOPs are fixed to be constant, sparse models outperform denser models on QA tasks (Abnar et al., 2025). By contrast, on mathematical-reasoning benchmarks like GSM8K and GSM-Plus, the relationship between density and performance depends on the available compute. At lower FLOPs, increasing sparsity still reduces loss and improves accuracy; however, once the FLOPs budget grows, denser models begin to perform better, achieving both lower loss and higher accuracy. This shift indicates that the optimal model density for reasoning tasks depends on compute budget: when a lot of FLOPs are available, a denser models may be preferable.

3.4. Impact of TTC and Post-Training on Downstream Performance

Test-Time Compute and RL post-training are standard for boosting reasoning on tasks such as mathematical problem solving. We therefore investigated whether performance declines reported above persist when applying (a) Test-Time Compute (TTC) and (b) RL post-training (GRPO). In Test-Time Compute, we evaluated GSM8K(COBBE ET AL., 2021) in a purely zero-shot setting using Self-Consistency (SC) decoding(Wang et al., 2023), generating 2^7 independent continuations per problem and selecting the most frequent answer. In Post-Training, we fine-tuned each model on the GSM8K training dataset using the GRPO algorithm (Shao et al., 2024). We followed the settings of Zhao et al. (2025) including reward function and fixed the learning rate constant across all model configurations.

As illustrated in Figure 6, neither Test-Time Compute nor GRPO mitigates the GSM8K performance drop that arises when total parameters increase. In other words, although both methods consistently improve overall performance, they do not eliminate the inverted U-shaped relationship between training loss and task accuracy.

3.5. Influence of Optimization Hyperparameter

Thus far, we have demonstrated that the structure of the model, particularly the degree of sparsity, can lead to differences in reasoning performance on downstream tasks, even when the models converge to the same training loss. Such differences are similar to generalization, in which a

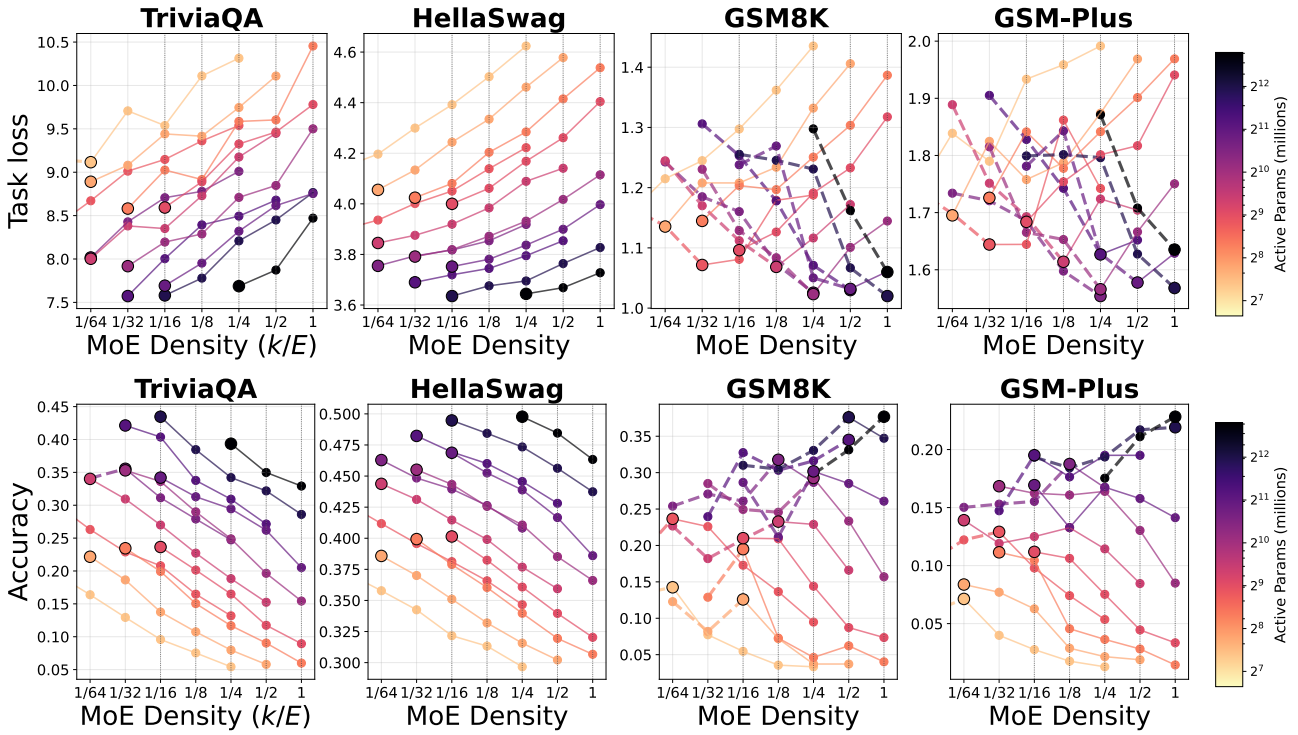

Figure 5. **At fixed active parameter counts, higher sparsity (lower density) consistently improves performance, but at larger active parameter counts, GSM8K and GSM-PLUS shift their optima back toward dense models.** Task loss (top row) and Accuracy (bottom row) against MoE Density k/E for a fixed active parameter budget.In the left two tasks (TRIVIAQA, HELLASWAG), increasing sparsity consistently lowers task loss and raises accuracy across all active parameter budgets, in contrast, in the right two tasks (GSM8K, GSM-PLUS), once active parameter counts become large, this trend reverses and denser models begin to outperform their sparser counterparts.

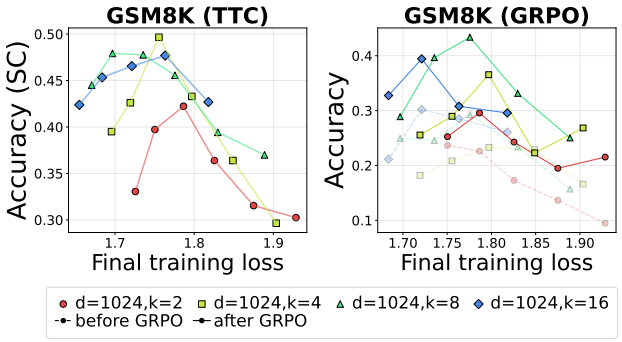

Figure 6. **Effect of Test-Time Compute and GRPO on the loss–accuracy trade-off.** Although both methods yield performance improvements that scale with model size, the loss–accuracy trade-off on GSM8K remains. Left: Final training loss vs. accuracy under Test-Time Compute (Self-Consistency). Right: Final training loss vs. accuracy after GRPO post-training.

model's behavior on unseen data reflects implicit inductive biases rather than mere fit to the training data. Studies on neural network generalization have long recognized that not only architectural choices, but also optimization dynamics (i.e., differences in hyperparameter settings, regularization schemes, and optimizer algorithms), play an important role in shaping these inductive biases. Motivated by this insight, we examine the learning-rate scale, which is critical to generalization (Keskar et al., 2017; Li et al., 2019; Yang & Hu, 2021). Our goal is to investigate how these choices influence the model's ability to transfer to downstream tasks, beyond what is captured by pre-training loss alone.

Figure 7 illustrates our empirical findings, obtained using a MoE architecture with 16 experts. By varying the learning rate, we evaluate performance on both QA benchmarks (TriviaQA, HellaSwag) and reasoning benchmarks (GSM8K, BBH). While QA benchmark performance like TriviaQA and HellaSwag remain largely invariant to these hyperparameters, reasoning benchmark performance like GSM8K and BBH are sensitive to the learning rates: when models converge to the same training loss, trainings with lower learning rates and smaller initialization scales yield superior downstream accuracy. These observations carry an important implication. Studies on generalization in large-scale language models should incorporate rigorous reasoning benchmarks (such as GSM8K and BBH) rather than relying solely on validation loss curves or standard QA tasks to fully capture the impact of optimization-induced implicit biases. This enables a more precise analysis on the

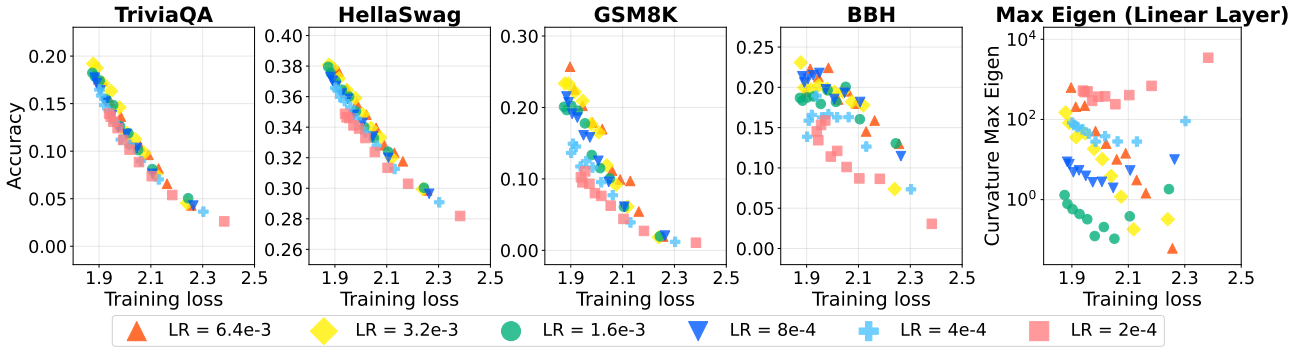

Figure 7. **For reasoning tasks like GSM8K and BBH, the relationship between training loss and downstream performance is dependent on the choice of optimization hyperparameters.** The learning rate also impact downstream accuracy. For the maximum eigenvalue, we evaluated the maximum eigenvalue of fisher information matrix under a K-FAC approximation (Martens & Grosse, 2015; Eschenhagen et al., 2023). Following (Grosse et al., 2023), we calculate the maximum eigenvalues only for linear layers. We find that higher learning rates lead to a lower maximum eigenvalue, which is consistent with existing research indicating that convergence to flatter minima improves generalization (Hochreiter & Schmidhuber, 1997; Keskar et al., 2017; Jiang et al., 2020).

generalization of LLMs.

4. Discussion and Limitations

Dataset In this study, we trained our model on a diverse set of mathematics datasets, ranging from web-sourced collections to those generated synthetically, as detailed in Table 1 of the Appendix. Our findings merely demonstrate that, under this particular mix of datasets, the model overfits on GSM8K. Thus, alternative dataset combinations, such as those without synthetic data or employing higher quality collections, may not exhibit the same overfitting behavior. All models are trained on a 125B-token corpus. This corpus is Chinchilla-optimal for dense models of comparable activated size (Hoffmann et al., 2022), yet two orders of magnitude smaller than the multi-trillion-token budgets now common for state-of-the-art MoE LLMs (DeepSeek-AI, 2025b; Qwen Team, 2025). Appendix C.4 shows that ablating the web-vs-math ratio up to a 1 T-token scale does *not* improve GSM8K, implying that sheer data volume is less critical than high-quality reasoning data. Recent large models such as OLMo-2 and Qwen-3 adopt a multi-stage curriculum training, general web pre-training followed by mid-training on math and CoT data (OLMo, 2025; Qwen Team, 2025); we avoid this design to keep a fixed data distribution and a clean link between pre-train loss and downstream accuracy, but exploring staged curricula remains important future work. These caveats render our conclusions suggestive rather than prescriptive and motivate verification at trillion-token scale with richer reasoning corpora.

Model We build on the Mixtral backbone and adopt the *fine-grained expert* segmentation of DeepSeek-MoE: each feed-forward block is split into $g = 2$, so the effective expert count becomes $E \times g$ while the total parameter budget stays fixed. In conjunction with standard top-k routing strategy ($k \in \{2, 4, 8, 16\}$) and the auxiliary importance / load-balance loss of Shazeer et al. (2017), our hyperparameter sweep evaluates configurations with up to 256 active experts. This is contrast to contemporary MoE variants such as Qwen-3, which primarily differ from Mixtral by the integration of only QK-Norm and a global load-balancing regularizer. These modifications are negligible in comparison to the scale of changes evaluated in our experiments. For scaling, the number of experts is more influential than minor structural details. We explore configurations with up to 256 experts and a top-16 routing strategy, which offers a sufficiently broad range for our purposes. We acknowledge that gating design choices, such as the formulation of the load-balance loss, might affect how expert scaling influences performance; we leave this for future work. The patterns we report are intended as provisional observations rather than definitive rules. We encourage further studies to examine these effects at larger model scales.

5. Conclusion

In this paper, we investigated the optimal sparsity of MoE language models through the lens of downstream task performance. By training families of Mixtral-style MoEs with various number of experts, top-k routing, and model width, and by evaluating them across pre-training, GRPO post-training, and test-time compute, we show that the classical "more experts is better" rule holds for knowledge-oriented benchmarks such as TriviaQA and HellaSwag, but not for mathematical reasoning benchmarks. On reasoning tasks, downstream task loss starts to rise, and accuracy to fall, once total parameters grow at a certain point; in this regime, models with more active parameters may achieve lower optimal task loss, whereas those with extreme sparsity over-fit despite lower pre-training loss. Neither reinforcement-learning post-training nor additional test-time compute removes this

trade-off. These findings update current scaling practice. When computational budget is fixed, allocating FLOPs to extra experts improves memorization, but improving reasoning ability requires matching growth in active parameters or even shifting toward denser MoE layers once enough compute is available.

Author Contributions

Taishi Nakamura prepared the pretraining datasets, conducted all pre-training experiments and evaluations (excluding test-time-compute), and co-designed the overall experimental setup. Satoki Ishikawa co-designed the experiments and formulated the overall research strategy. Masaki Kawamura initiated the post-training and test-time-compute (TTC) experiments. Takumi Okamoto conducted the post-training experiments and carried out the Max-Eigen (linear-layer) experiments. Daisuke Nohara conducted TTC experiments. Rio Yokota and Jun Suzuki provided guidance and oversight throughout the project. All authors contributed to manuscript writing and approved the final version.

Acknowledgements

This work was supported by the "R&D Hub Aimed at Ensuring Transparency and Reliability of Generative AI Models" project of the Ministry of Education, Culture, Sports, Science and Technology. We used ABCI 3.0 provided by AIST and AIST Solutions with support from "ABCI 3.0 Development Acceleration Use. This work used computational resources TSUBAME4.0 supercomputer provided by Institute of Science Tokyo through the HPCI System Research Project (Project ID: hp240170).

References

Abnar, S., Shah, H., Busbridge, D., Ali, A. M. E., Susskind, J., and Thilak, V. Parameters vs flops: Scaling laws for optimal sparsity for mixture-of-experts language models. *International Conference on Machine Learning*, 2025.

Bhagia, A., Liu, J., Wettig, A., Heineman, D., Tafjord, O., Jha, A. H., Soldaini, L., Smith, N. A., Groeneveld, D., Koh, P. W., et al. Establishing task scaling laws via compute-efficient model ladders. arXiv:2412.04403, 2024.

Brandfonbrener, D., Anand, N., Vyas, N., Malach, E., and Kakade, S. M. Loss-to-loss prediction: Scaling laws for all datasets. *Transactions on Machine Learning Research*, 2025.

Brown, B., Juravsky, J., Ehrlich, R., Clark, R., Le, Q. V., Ré, C., and Mirhoseini, A. Large language monkeys: Scaling inference compute with repeated sampling. arXiv:2407.21787, 2024.

Caballero, E., Gupta, K., Rish, I., and Krueger, D. Broken neural scaling laws. In *International Conference on Learning Representations*, 2023.

Clark, A., De Las Casas, D., Guy, A., Mensch, A., Paganini, M., Hoffmann, J., Damoc, B., Hechtman, B., Cai, T., Borgeaud, S., et al. Unified scaling laws for routed language models. In *International Conference on Machine Learning*, 2022.

Cobbe, K., Kosaraju, V., Bavarian, M., Chen, M., Jun, H., Kaiser, L., Plappert, M., Tworek, J., Hilton, J., Nakano, R., et al. Training verifiers to solve math word problems. arXiv:2110.14168, 2021.

DeepSeek-AI. DeepSeek-V2: A strong, economical, and efficient mixture-of-experts language model. arXiv:2405.04434, 2024.

DeepSeek-AI. DeepSeek-R1: Incentivizing reasoning capability in llms via reinforcement learning. arXiv:2501.12948, 2025a.

DeepSeek-AI. DeepSeek-V3 technical report. arXiv:2412.19437, 2025b.

Du, N., Huang, Y., Dai, A. M., Tong, S., Lepikhin, D., Xu, Y., Krikun, M., Zhou, Y., Yu, A. W., Firat, O., et al. GLaM: Efficient scaling of language models with mixture-of-experts. In *International Conference on Machine Learning*, 2022.

Eschenhagen, R., Immer, A., Turner, R., Schneider, F., and Hennig, P. Kronecker-factored approximate curvature for modern neural network architectures. In *Advances in Neural Information Processing Systems*, 2023.

Fedus, W., Zoph, B., and Shazeer, N. M. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *J. Mach. Learn. Res.*, 2021.

Frantar, E., Ruiz, C. R., Houlsby, N., Alistarh, D., and Evci, U. Scaling laws for sparsely-connected foundation models. In *International Conference on Learning Representations*, 2024.

Gadre, S. Y., Smyrnis, G., Shankar, V., Gururangan, S., Wortsman, M., Shao, R., Mercat, J., Fang, A., Li, J., Keh, S., et al. Language models scale reliably with overtraining and on downstream tasks. arXiv:2403.08540, 2024.

Gale, T., Narayanan, D., Young, C., and Zaharia, M. MegaBlocks: Efficient Sparse Training with Mixture-of-Experts. *Proceedings of Machine Learning and Systems*, 2023.

Gao, L., Tow, J., Abbasi, B., Biderman, S., Black, S., DiPofi, A., Foster, C., Golding, L., Hsu, J., Le Noac'h, A., et al. The language model evaluation harness, 2024.

Gemini Team. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. arXiv:2403.05530, 2024.

Gemini Team. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. arXiv:2507.06261, 2025.

Grattafiori, A., Dubey, A., Jauhri, A., Pandey, A., Kadian, A., Al-Dahle, A., Letman, A., Mathur, A., Schelten, A., Vaughan, A., et al. The Llama 3 herd of models. arXiv:2407.21783, 2024.

Grosse, R., Bae, J., Anil, C., Elhage, N., Tamkin, A., Tajdini, A., Steiner, B., Li, D., Durmus, E., Perez, E., et al. Studying large language model generalization with influence functions. arXiv:2308.03296, 2023.

Hochreiter, S. and Schmidhuber, J. Flat minima. *Neural computation*, 1997.

Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., de las Casas, D., Hendricks, L. A., Welbl, J., Clark, A., et al. An empirical analysis of compute-optimal large language model training. In *Advances in Neural Information Processing Systems*, 2022.

Inoue, Y., Misaki, K., Imajuku, Y., Kuroki, S., Nakamura, T., and Akiba, T. Wider or deeper? scaling llm inference-time compute with adaptive branching tree search. arXiv:2503.04412, 2025.

Jacobs, R. A., Jordan, M. I., and Barto, A. G. Task decomposition through competition in a modular connectionist architecture: The what and where vision tasks. *Cognitive science*, 1991.

Jelassi, S., Mohri, C., Brandfonbrener, D., Gu, A., Vyas, N., Anand, N., Alvarez-Melis, D., Li, Y., Kakade, S. M., and Malach, E. Mixture of parrots: Experts improve memorization more than reasoning. In *International Conference on Learning Representations*, 2025.

Jiang, A. Q., Sablayrolles, A., Roux, A., Mensch, A., Savary, B., Bamford, C., Chaplot, D. S., de las Casas, D., Hanna, E. B., Bressand, F., et al. Mixtral of experts. arXiv:2401.04088, 2024.

Jiang, Y., Neyshabur, B., Mobahi, H., Krishnan, D., and Bengio, S. Fantastic generalization measures and where to find them. In *International Conference on Learning Representations*, 2020.

Jordan, M. I. and Jacobs, R. A. Hierarchical mixtures of experts and the em algorithm. *Neural computation*, 1994.

Joshi, M., Choi, E., Weld, D., and Zettlemoyer, L. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2017.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. Scaling laws for neural language models. arXiv:2001.08361, 2020.

Keskar, N. S., Mudigere, D., Nocedal, J., Smelyanskiy, M., and Tang, P. T. P. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations*, 2017.

Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., and Iwasawa, Y. Large language models are zero-shot reasoners. In *Advances in Neural Information Processing Systems*, 2022.

Kumar, T., Ankner, Z., Spector, B. F., Bordelon, B., Muennighoff, N., Paul, M., Pehlevan, C., Re, C., and Raghunathan, A. Scaling laws for precision. In *International Conference on Learning Representations*, 2025.

Lepikhin, D., Lee, H., Xu, Y., Chen, D., Firat, O., Huang, Y., Krikun, M., Shazeer, N., and Chen, Z. {GS}hard: Scaling giant models with conditional computation and automatic sharding. In *International Conference on Learning Representations*, 2021.

Li, Q., Cui, L., Zhao, X., Kong, L., and Bi, W. GSM-plus: A comprehensive benchmark for evaluating the robustness of LLMs as mathematical problem solvers. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024.

Li, Y., Wei, C., and Ma, T. Towards explaining the regularization effect of initial large learning rate in training neural networks. In *Advances in Neural Information Processing Systems*, 2019.

Li, Y., Choi, D., Chung, J., Kushman, N., Schrittwieser, J., Leblond, R., Eccles, T., Keeling, J., Gimeno, F., Dal Lago, A., et al. Competition-level code generation with alphacode. *Science*, 2022.

Lightman, H., Kosaraju, V., Burda, Y., Edwards, H., Baker, B., Lee, T., Leike, J., Schulman, J., Sutskever, I., and Cobbe, K. Let's verify step by step. In *International Conference on Learning Representations*, 2024.

Liu, H., Xie, S. M., Li, Z., and Ma, T. Same pre-training loss, better downstream: Implicit bias matters for language models. In *International Conference on Machine Learning*, 2023.

Liu, Y., Jin, R., Shi, L., Yao, Z., and Xiong, D. FineMath: A fine-grained mathematical evaluation benchmark for chinese large language models. arXiv:2403.07747, 2024.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Ludziejewski, J., Krajewski, J., Adamczewski, K., Pióro, M., Krutul, M., Antoniak, S., Ciebiera, K., Król, K., Odrzygóźdź, T., Sankowski, P., et al. Scaling laws for fine-grained mixture of experts. In *International Conference on Machine Learning*, 2024.

Martens, J. and Grosse, R. Optimizing neural networks with kronecker-factored approximate curvature. In *International Conference on Machine Learning*, 2015.

Mirzadeh, I., Alizadeh, K., Shahrokhi, H., Tuzel, O., Bengio, S., and Farajtabar, M. Gsm-symbolic: Understanding the limitations of mathematical reasoning in large language models. arXiv:2410.05229, 2024.

Mitra, A., Khanpour, H., Rosset, C., and Awadallah, A. Orca-math: Unlocking the potential of slms in grade school math. arXiv:2402.14830, 2024.

Muennighoff, N., Rush, A., Barak, B., Le Scao, T., Tazi, N., Piktus, A., Pyysalo, S., Wolf, T., and Raffel, C. A. Scaling data-constrained language models. In *Advances in Neural Information Processing Systems*, 2023.

Muennighoff, N., Soldaini, L., Groeneveld, D., Lo, K., Morrison, J., Min, S., Shi, W., Walsh, E. P., Tafjord, O., Lambert, N., et al. OLMoE: Open mixture-of-experts language models. In *International Conference on Learning Representations*, 2025a.

Muennighoff, N., Yang, Z., Shi, W., Li, X. L., Fei-Fei, L., Hajishirzi, H., Zettlemoyer, L., Liang, P., Candès, E., and Hashimoto, T. s1: Simple test-time scaling. arXiv:2501.19393, 2025b.

OLMo, T. 2 OLMo 2 furious. arXiv:2501.00656, 2025.

OpenAI. GPT-4 technical report. arXiv:2303.08774, 2024a.

OpenAI. Openai o1 system card. arXiv:2412.16720, 2024b.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Gray, A., et al. Training language models to follow instructions with human feedback. In *Advances in Neural Information Processing Systems*, 2022.

Qwen. Qwen2.5 technical report. arXiv:2412.15115, 2025.

Qwen Team. Qwen3 technical report. arXiv:2505.09388, 2025.

Roberts, N., Chatterji, N., Narang, S., Lewis, M., and Hupkes, D. Compute optimal scaling of skills: Knowledge vs reasoning. arXiv:2503.10061, 2025.

Schaeffer, R., Kazdan, J., Hughes, J., Juravsky, J., Price, S., Lynch, A., Jones, E., Kirk, R., Mirhoseini, A., and Koyejo, S. How do large language monkeys get their power (laws)? In *International Conference on Machine Learning*, 2025.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. arXiv:1707.06347, 2017.

Shao, Z., Wang, P., Zhu, Q., Xu, R., Song, J., Bi, X., Zhang, H., Zhang, M., Li, Y. K., Wu, Y., et al. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. arXiv:2402.03300, 2024.

Shazeer, N. Glu variants improve transformer. arXiv:2002.05202, 2020.

Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., and Dean, J. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017.

Shinn, N., Cassano, F., Gopinath, A., Narasimhan, K., and Yao, S. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, 2024.

Snell, C. V., Lee, J., Xu, K., and Kumar, A. Scaling LLM test-time compute optimally can be more effective than scaling parameters for reasoning. In *International Conference on Learning Representations*, 2025.

Soldaini, L., Kinney, R., Bhagia, A., Schwenk, D., Atkinson, D., Authur, R., Bogin, B., Chandu, K., Dumas, J., Elazar, Y., et al. Dolma: an open corpus of three trillion tokens for language model pretraining research. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024.

Springer, J. M., Goyal, S., Wen, K., Kumar, T., Yue, X., Malladi, S., Neubig, G., and Raghunathan, A. Overtrained language models are harder to fine-tune. arXiv:2503.19206, 2025.

Su, J., Ahmed, M., Lu, Y., Pan, S., Bo, W., and Liu, Y. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 2024.

Suzgun, M., Scales, N., Schärli, N., Gehrmann, S., Tay, Y., Chung, H. W., Chowdhery, A., Le, Q., Chi, E., Zhou, D., et al. Challenging BIG-bench tasks and whether chain-of-thought can solve them. In *Findings of the Association for Computational Linguistics: ACL 2023*, 2023.

Takano, R., Takizawa, S., Tanimura, Y., Nakada, H., and Ogawa, H. Abci 3.0: Evolution of the leading ai infrastructure in japan. arXiv:2411.09134, 2024.

Taylor, R., Kardas, M., Cucurull, G., Scialom, T., Hartshorn, A., Saravia, E., Poulton, A., Kerkez, V., and Stojnic, R. Galactica: A large language model for science. arXiv:2211.09085, 2022.

Tikhonov, A. and Ryabinin, M. It's all in the heads: Using attention heads as a baseline for cross-lingual transfer in commonsense reasoning. In *Findings of the Association for Computational Linguistics (ACL)*, 2021.

Toshniwal, S., Du, W., Moshkov, I., Kisacanin, B., Ayrapetyan, A., and Gitman, I. OpenMathInstruct-2: Accelerating AI for math with massive open-source instruction data. In *Workshop on Mathematical Reasoning and AI at NeurIPS'24*, 2024a.

Toshniwal, S., Moshkov, I., Narenthiran, S., Gitman, D., Jia, F., and Gitman, I. OpenMathInstruct-1: A 1.8 million math instruction tuning dataset. In *Neural Information Processing Systems Datasets and Benchmarks Track*, 2024b.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. u., and Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

Wang, X., Wei, J., Schuurmans, D., Le, Q. V., Chi, E. H., Narang, S., Chowdhery, A., and Zhou, D. Self-consistency improves chain of thought reasoning in language models. In *International Conference on Learning Representations*, 2023.

Wang, X., Antoniades, A., Elazar, Y., Amayuelas, A., Albalak, A., Zhang, K., and Wang, W. Y. Generalization v.s. memorization: Tracing language models' capabilities back to pretraining data. In *International Conference on Learning Representations*, 2025.

Wang, Z., Li, X., Xia, R., and Liu, P. Mathpile: A billion-token-scale pretraining corpus for math. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024.

Xue, F., Fu, Y., Zhou, W., Zheng, Z., and You, Y. To repeat or not to repeat: Insights from scaling llm under token-crisis. In *Advances in Neural Information Processing Systems*, 2023.

Yang, G. and Hu, E. J. Tensor programs iv: Feature learning in infinite-width neural networks. In *International Conference on Machine Learning*, 2021.

Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A., and Choi, Y. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 2019.

Zhang, B. and Sennrich, R. Root Mean Square Layer Normalization. In *Advances in Neural Information Processing Systems*, 2019.

Zhang, H., Da, J., Lee, D., Robinson, V., Wu, C., Song, W., Zhao, T., Raja, P. V., Zhuang, C., Slack, D. Z., et al. A careful examination of large language model performance on grade school arithmetic. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024.

Zhao, R., Meterez, A., Kakade, S., Pehlevan, C., Jelassi, S., and Malach, E. Echo chamber: Rl post-training amplifies behaviors learned in pretraining. arXiv:2504.07912, 2025.

Zoph, B., Bello, I., Kumar, S., Du, N., Huang, Y., Dean, J., Shazeer, N., and Fedus, W. St-moe: Designing stable and transferable sparse expert models. arXiv:2202.08906, 2022.

Zyphra. dclm-dedup. https://huggingface.co/datasets/Zyphra/dclm-dedup, 2024. Accessed: 2025-05-16.

Table 1. Break-down of the 125 B-token pre-training corpus.

Source	Type	Tokens	Corpus	Hugging Face
High Quality Web				
DCLM-Deduped	High quality web	33.5B	788.5B	Zyphra/dclm-dedup
Flan decontaminated	High quality web	9.2B	18.5B	allenai/dolmino-mix-1124
WebInstructFull	High quality web	14.7M	29.7M	TIGER-Lab/WebInstructFull
STEM Literature & Reference				
peS2o	Academic papers	31.1B	62.9B	allenai/dolma
ArXiv	STEM papers	11.0B	22.2B	allenai/dolma
Wikipedia	Encyclopedic	2.3B	4.7B	-
Wikipedia & Wikibooks	Encyclopedic	1.9B	3.9B	allenai/dolma
Project Gutenberg	Books	2.7B	5.5B	allenai/dolma
Mathematics				
OpenWebMath	Math	6.6B	13.4B	allenai/dolma
Algebraic Stack	Math	6.6B	13.3B	allenai/dolma
FineMath-4+	Math	5.1B	10.3B	HuggingFaceTB/finemath
MathPile commercial subset train split	Math	4.5B	9.2B	GAIR/MathPile_Commercial
TinyGSM-MIND	Synthetic math	3.4B	6.9B	allenai/olmo-mix-1124
OpenMathInstruct-2	Synthetic math	2.6B	5.2B	nvidia/OpenMathInstruct-2
MathCoder2 Synthetic	Synthetic Math	2.0B	4.1B	allenai/olmo-mix-1124
StackMathQA	Math	529.6M	1070.0M	math-ai/StackMathQA
NaturalReasoning	General reasoning	506.0M	1022.2M	facebook/natural_reasoning
NuminaMath-CoT train split	CoT reasoning	221.0M	446.4M	AI-MO/NuminaMath-CoT
OpenMathInstruct-1 train split	Synthetic math	168.4M	340.2M	nvidia/OpenMathInstruct-1
TuluMath	Synthetic math	123.9M	250.4M	allenai/olmo-mix-1124
Metamath OWM-filtered	Math	42.3M	85.4M	allenai/olmo-mix-1124
Orca-Math	Synthetic math	33.5M	67.7M	microsoft/orca-math-word-problems-200k
Dolmino SynthMath	Synthetic math	15.7M	31.7M	allenai/olmo-mix-1124
GSM8K train split	Math	1.4M	2.8M	allenai/dolmino-mix-1124
GSM8K train split	Math	1.4M	2.8M	openai/gsm8k
CodeSearchNet-owmfilter	Math	1.1M	2.2M	allenai/dolmino-mix-1124
Code				
StackExchange	CodeText	725.1M	1464.8M	allenai/dolmino-mix-1124
Grand total		**125.0B**	**973.4B**	

A. Training Setup

A.1. Pre-training Dataset Details

Table 1 summarizes the pre-training corpus: for each subset, it lists the Hugging Face repository, split identifier, and public URL, alongside the original size and the number of subsampled tokens we used (125 B tokens in the 99:1 train/validation split, as counted by the llm-jp tokenizer v3 with 99,487 tokens). Thus, the total token budget is fixed in strict accordance with Kaplan's scaling law (Kaplan et al., 2020), meaning the observed loss increase (and the accompanying puzzling overfitting that mirrors behavior recently reported by (OLMo, 2025; OpenAI, 2024a)) cannot be attributed to any change in data volume.

A.2. Post-Training Details

We use GRPO(Shao et al., 2024) with a batch size of 1024, train for 15 epochs, and truncate prompts and generated sequences to 512 and 1024 tokens respectively. The actor's learning rate is fixed at 5×10^{-6}; the temperature is set to 1.0, the KL-penalty coefficient to 10^{-3}, and 5 samples are used per prompt. Optimisation employs Adam with $\beta = (0.9, 0.999)$, $\epsilon = 10^{-8}$, and weight decay of 10^{-2}. Following Zhao et al. (2025), we implemented a code-execution-based evaluator supporting TinyGSM-style and OpenMathInstruct-1 outputs. For a width of 2048 with 16 or 64 experts, we swept the learning rate (Fig. 8) and subsequently fixed it to 5×10^{-6} for all GRPO experiments.

Table 2. Detailed composition of the 125 B-token pre-training corpus *without* GSM8K and its synthetic variants (used for the ablation in Section C.3). Token counts and raw corpus sizes are listed for each source, following the same category structure as Table 1.

Source	Type	Tokens	Corpus	Hugging Face
High Quality Web				
DCLM-Deduped	High quality web	33.5B	788.5B	Zyphra/dclm-dedup
Flan decontaminated	High quality web	9.2B	18.5B	allenai/dolmino-mix-1124
WebInstructFull	High quality web	14.7M	29.7M	TIGER-Lab/WebInstructFull
STEM Literature & Reference				
peS2o	Academic papers	31.1B	62.9B	allenai/dolma
ArXiv	STEM papers	11.0B	22.2B	allenai/dolma
Wikipedia	Encyclopedic	2.3B	4.7B	-
Wikipedia & Wikibooks	Encyclopedic	1.9B	3.9B	allenai/dolma
Project Gutenberg	Books	2.7B	5.5B	allenai/dolma
Mathematics				
OpenWebMath	Math	8.2B	13.4B	allenai/dolma
Algebraic Stack	Math	8.1B	13.3B	allenai/dolma
FineMath-4+	Math	6.3B	10.3B	HuggingFaceTB/finemath
MathPile commercial subset train split	Math	5.6B	9.2B	GAIR/MathPile_Commercial
MathCoder2 Synthetic	Synthetic Math	2.5B	4.1B	allenai/olmo-mix-1124
StackMathQA	Math	653.9M	1070.0M	math-ai/StackMathQA
NaturalReasoning	General reasoning	624.7M	1022.2M	facebook/natural_reasoning
NuminaMath-CoT train split	CoT reasoning	272.8M	446.4M	AI-MO/NuminaMath-CoT
TuluMath	Synthetic math	153.0M	250.4M	allenai/olmo-mix-1124
Metamath OWM-filtered	Math	52.2M	85.4M	allenai/olmo-mix-1124
Orca-Math	Synthetic math	41.4M	67.7M	microsoft/orca-math-word-problems-200k
CodeSearchNet-owmfilter	Math	1.1M	2.2M	allenai/dolmino-mix-1124
Code				
StackExchange	CodeText	725.1M	1464.8M	allenai/dolmino-mix-1124
Grand total		**125.0B**	**961.0B**	

A.3. Implementation & Training Environment

We executed all pre-training runs on the ABCI 3.0 supercomputer (Takano et al., 2024), equipped with NVIDIA H200 GPUs with board-level power capped at 500 W per GPU. TTC experiments were conducted on the TSUBAME 4.0 supercomputer at the Global Scientific Information and Computing Center, Institute of Science Tokyo. They used NVIDIA H100 SXM5 94 GB GPUs (four GPUs per node) and InfiniBand NDR200 interconnects for inter-node communication.

For pre-training, we extended the Megatron-LM[1] codebase to add functionality needed for this study, with support for pipeline, tensor, and expert parallelism. Reinforcement learning experiments were implemented using GRPO (Shao et al., 2024) on top of the VerL[2] framework. Model quality was assessed using lm-evaluation-harness[3] and LargeLanguageMonkeys[4].

B. Evaluation Setup

We evaluate our models using the lm-evaluation-harness framework (Gao et al., 2024) across four key capability areas. All evaluations employ standard few-shot prompting strategies unless otherwise specified.

We assess logical reasoning capabilities using Big Bench Hard (BBH) (Suzgun et al., 2023) with 3-shot Chain-of-Thought (CoT) prompting. Mathematical problem-solving is evaluated using GSM8K (Cobbe et al., 2021) with 4-shot prompting and

[1] https://github.com/NVIDIA/Megatron-LM
[2] https://github.com/volcengine/verl
[3] https://github.com/EleutherAI/lm-evaluation-harness
[4] https://github.com/ScalingIntelligence/large_language_monkeys

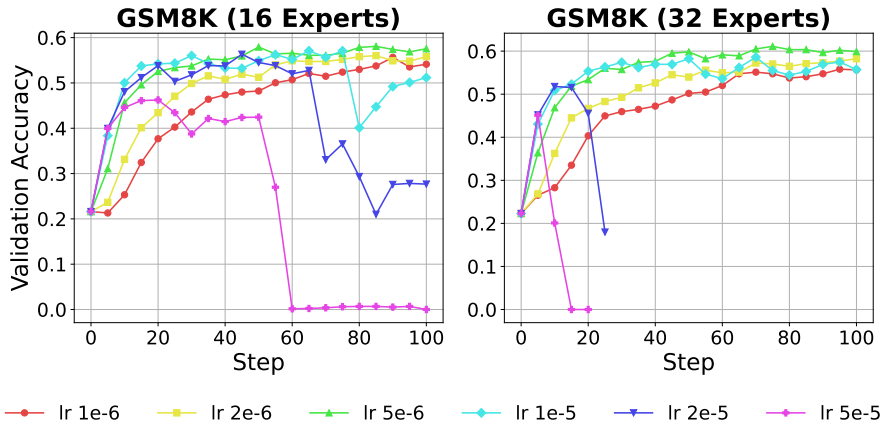

Figure 8. **Learning-rate sweep for width** $= 2048$**.** We varied the number of experts and swept the learning rate. For both 16 and 32 experts, 5×10^{-6} produces the most stable training.

Table 3. Evaluation Benchmark Details

Dataset	**TriviaQA**	**HellaSwag**	**XWinograd**	**BBH**	**GSM8K**	**GSM-Plus**
Task	QA	MRC	Commonsense Reasoning	Logical Reasoning	Math Reasoning	Math Reasoning
Language	EN	EN	EN	EN	EN	EN
# Instances	17,944	10,042	2,325	6,511	1,319	10,552
Few-shot #	4	4	4	3	4 (0 for TTC)	5
Metric	Accuracy	Accuracy	Accuracy	CoT Acc.	Accuracy	CoT Acc.

GSM-Plus (Li et al., 2024) with 5-shot CoT prompting. We evaluate comprehension abilities using TriviaQA (Joshi et al., 2017) with 4-shot prompting. Common sense reasoning is assessed through HellaSwag (Zellers et al., 2019) and XWinograd (EN) (Tikhonov & Ryabinin, 2021), both using 4-shot prompting setups. For Test-Time Compute (TTC) experiments specifically, GSM8K evaluation is conducted under a zero-shot setting. To accommodate the variety of valid answer formats, we extend the strict match patterns provided by the `lm-evaluation-harness` beyond the standard implementation. Our matching criteria accept both the standard GSM8K format (`####`) and GSM8K-CoT formats prefixed with "The answer is" or "Answer:".

Table 3 provides comprehensive details for all evaluation benchmarks.

C. Additional Experiments

C.1. GRPO

Training on MATH 500 Dataset Following the analysis presented in Section 3.4, the inverted U-shaped relationship between training loss and task accuracy persists even after applying GRPO. To verify that this phenomenon is not due to performing GRPO on the GSM8K dataset, we conducted additional GRPO experiments on the MATH 500 dataset (Lightman et al., 2024). As illustrated in Figure 9, GRPO on the MATH dataset yields consistent results with those obtained on the GSM8K dataset, confirming that this inverted U-shaped relationship is robust across different GRPO training datasets.

C.2. Test-Time Compute

Evaluation Setup We evaluated both GSM8K(Cobbe et al., 2021) in a purely zero-shot setting using Self-Consistency (SC) decoding(Wang et al., 2023), generating 2^7 independent continuations per problem and selecting the most frequent answer with 128 samples per problem. Specifically, for each prompt we generated up to 1,024 tokens under temperature 0.6 and nucleus sampling (top-$p = 0.95$), drawing 128 independent continuations and selecting the most frequent answer.

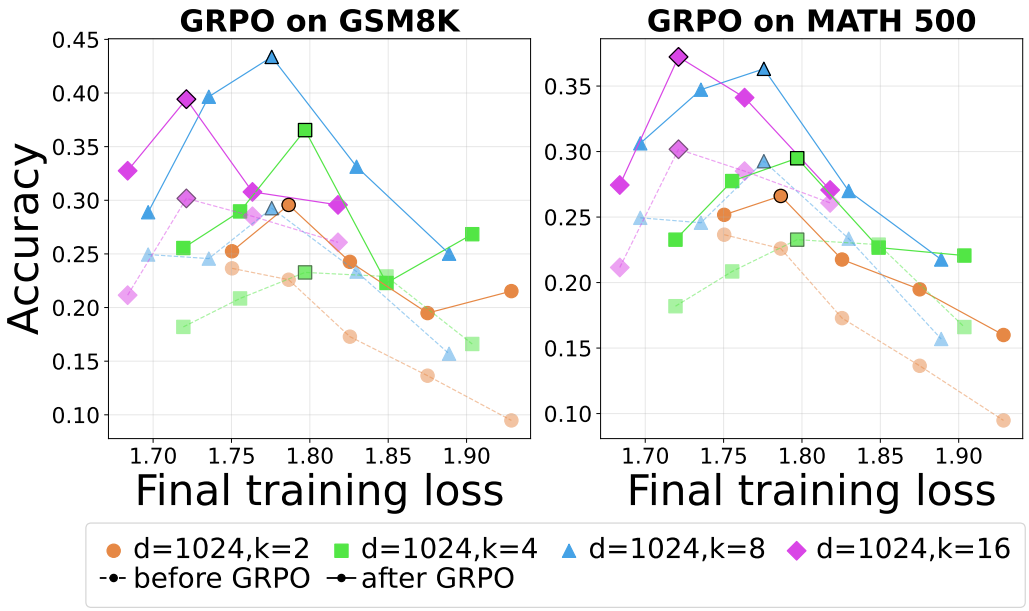

Figure 9. **Comparison of GSM8K accuracy for models fine-tuned with GRPO on different training datasets (left: GSM8K, right: MATH 500).** Performance decline is consistently observed across different training datasets.

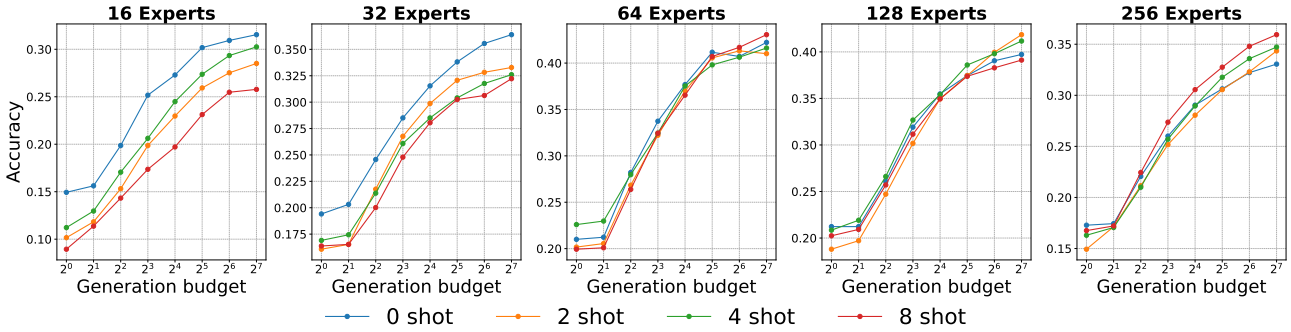

Figure 10. **GSM8K accuracy of model (d=1024) across different shot counts.** Because few shot performance is unstable and dropped significantly for models with a small number of experts, zero shot is used for Test-Time Compute.

Zero-shot VS Few-shot To set up Test Time Compute appropriately, we investigate how varying the number of prompt shots affected each expert's behavior (Figure 10). Few shot performance is unstable and dropped significantly for models with a small number of experts, so we use zero shot inference for Test Time Compute. When few shot chain of thought is used to standardize answer formats, the provided demonstration steps can be internalized as a fixed reasoning pattern by the model. As a result, the model's inherent inference capabilities may not be fully expressed, and its ability to generalize to novel problems could be hindered (Kojima et al., 2022).

Temperature Figure 11 shows that the inverted U-shaped performance-decline trend holds across every temperature setting, indicating that sampling temperature does not affect this behavior. This suggests that, although temperature controls inference randomness, the primary drivers of performance decline are inherent to model architecture rather than temperature settings.

Evaluation of Larger Generation Budget We extended the sample size used for Test-Time Compute as described in Section 3.4, generating a larger set of candidate responses. We then measured the resulting accuracy across different

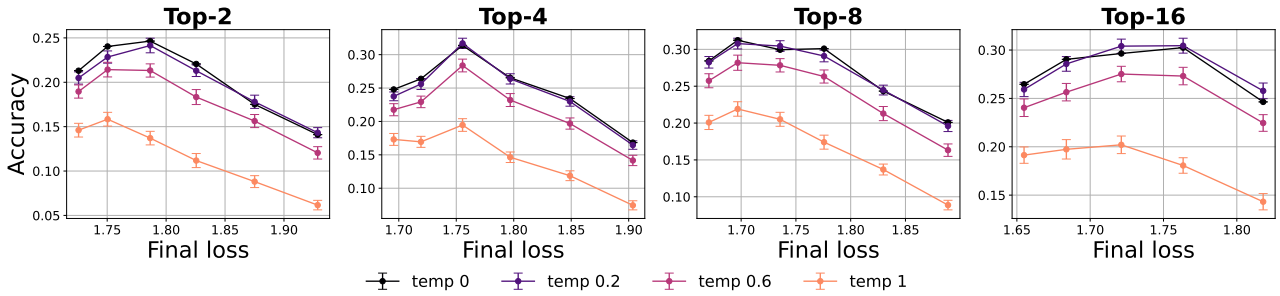

Figure 11. **Comparison of performance decline across different temperature settings (pass@1, d=1024).** A consistent performance decline is observed regardless of temperature, and overall accuracy increases as temperature decreases (i.e., approaches greedy).

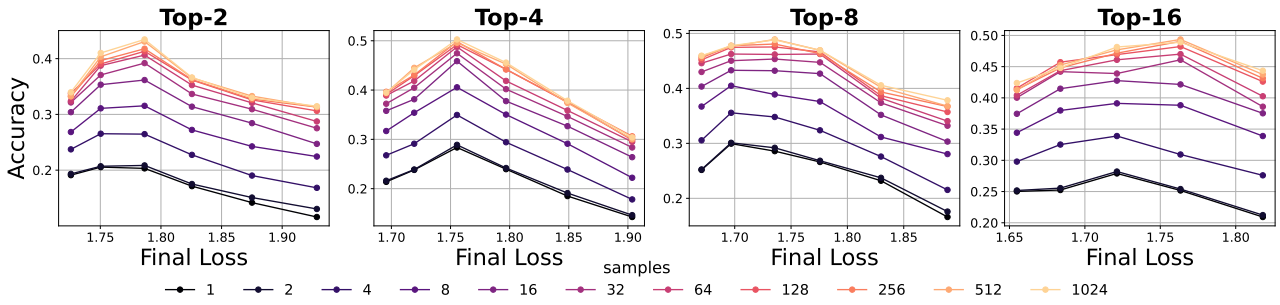

Figure 12. **Accuracy across generation budgets with increased sample counts.** With an active parameter count of 8 (top 8), the performance decline is gradually alleviated as the budget increases, whereas with an active parameter count of 2 (top 2), the decline is amplified, resulting in a more pronounced U shaped trend.

generation budgets to assess how increased sampling influences performance (Figure 12). For an active parameter count of 8 (top-8), the performance decline is gradually mitigated, whereas for an active parameter count of 2 (top-2), the decline is instead amplified, resulting in a more pronounced U-shaped trend. Although increasing the sample count further may provide additional insights, it remains challenging to identify a consistent mitigation pattern across all models.

Increasing Top-k During Inference We compared the performance under TTC for model with a hidden dimension of 2048, 128 experts, and top-2 routing by varying the inference-time top-k parameter. (Figure 13) Specifically, although doubling top-k sometimes yielded temporary improvements in Pass@1, applying TTC ultimately showed that the original top-2 setting maintained the highest performance, suggesting that no fundamental performance gain occurs.

C.3. GSM8K Overfitting Analysis

To investigate whether our model overfits to GSM8K due to the inclusion of GSM8K training data and its synthetic derivatives, we conducted an ablation experiment removing GSM8K-related datasets from our pre-training corpus as listed in Table 2.

We removed TinyGSM-MIND, both GSM8K train split instances, Dolmino SynthMath, OpenMathInstruct-1, and OpenMathInstruct-2, which contain either the original GSM8K training data or synthetic problems derived from it.

The results are shown in Figure 14. We observe that the trends with respect to sparsity on GSM8K remain unchanged, both for Pass@1 and TTC metrics. This indicates that while GSM8K training data and its synthetic derivatives do improve GSM8K scores, they do not alter the underlying performance trends. However, after post-training, we observe some changes in these trends, which we leave as future work to investigate further.

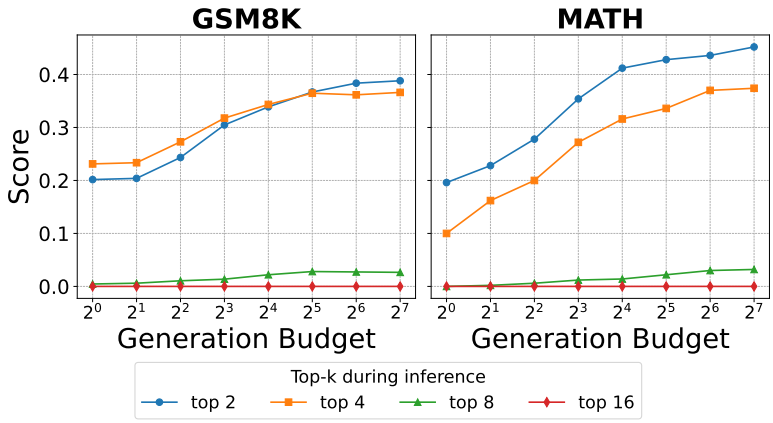

Figure 13. **Increasing the top-k parameter only at inference time does not improve performance.** Performance comparison under TTC for a Mixture-of-Experts model (hidden dimension 2048, 128 experts, top-2) as the top-k parameter is increased. While doubling k can occasionally improve Pass@1, applying TTC ultimately shows that the original top-2 configuration delivers the highest performance.

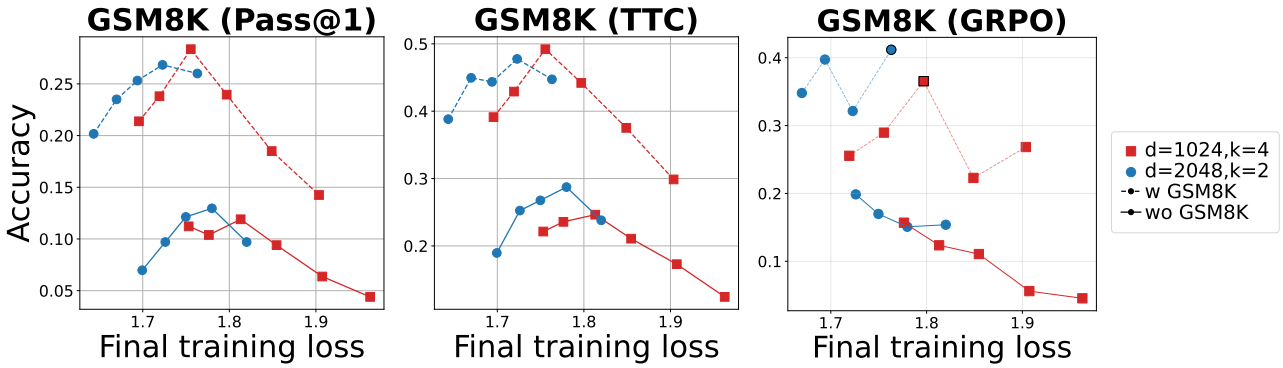

Figure 14. GSM8K performance without GSM8K-related training data: Pass@1 (left), TTC with 128 budget (center), and after GRPO (right)

C.4. Dataset Ablation

Table 1 shows a comparison between training on the full 1T tokens, training without web data, and our main experimental setup used in this study. Even when training on 1T tokens, we find that GSM8K scores do not improve proportionally to the number of tokens. Additionally, we compare experiments focused on math-centered training without mixing web data. Here, we observe that even with 1T token training, GSM8K scores do not improve commensurate with the token count. Furthermore, while math-focused training degrades performance on other tasks, it does not improve GSM8K performance.

Figure 15 presents the ablation results.

C.5. GSM8K Problem Analysis

We investigated whether models with varying numbers of experts exhibit differences in their ability to solve specific problems on the GSM8K dataset.

Figure 16 shows the results. We observe that different sparsity levels solve different instances of the problems.

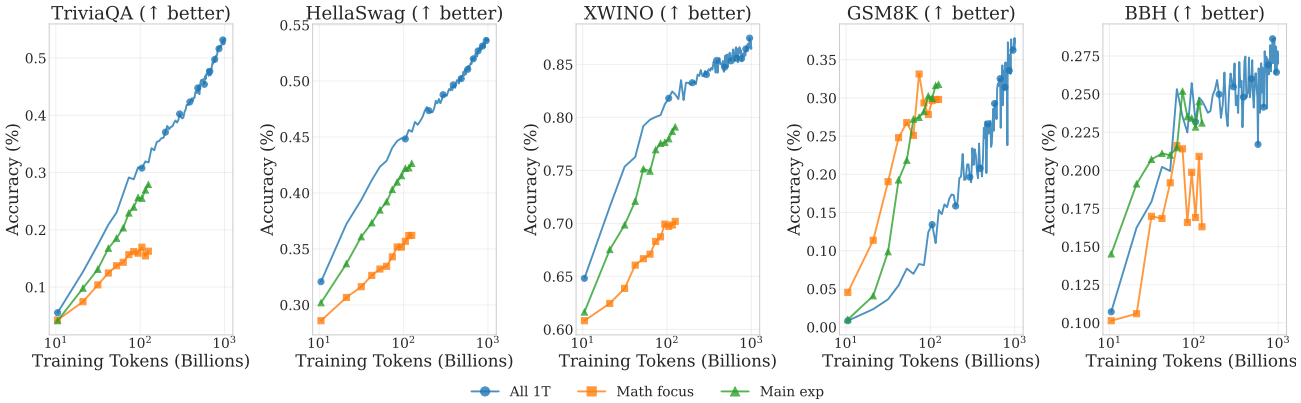

Figure 15. **Performance comparison across different model configurations as a function of training tokens.** The graph shows ablation results on various datasets, illustrating the effect of expert count and routing strategies on model performance.

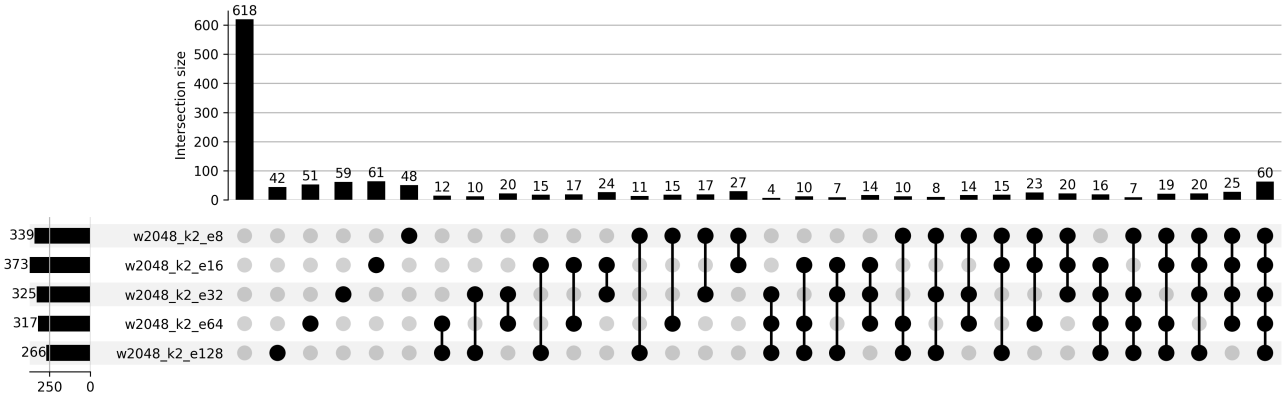

Figure 16. **Analysis of solvable problems across different numbers of experts on GSM8K.** This graph displays the number of problems that were commonly solvable or unsolvable across models with varying numbers of experts.