DeepSeek-V3 Technical Report

DeepSeek-AI

research@deepseek.com

Abstract

We present DeepSeek-V3, a strong Mixture-of-Experts (MoE) language model with 671B total parameters with 37B activated for each token. To achieve efficient inference and cost-effective training, DeepSeek-V3 adopts Multi-head Latent Attention (MLA) and DeepSeekMoE architectures, which were thoroughly validated in DeepSeek-V2. Furthermore, DeepSeek-V3 pioneers an auxiliary-loss-free strategy for load balancing and sets a multi-token prediction training objective for stronger performance. We pre-train DeepSeek-V3 on 14.8 trillion diverse and high-quality tokens, followed by Supervised Fine-Tuning and Reinforcement Learning stages to fully harness its capabilities. Comprehensive evaluations reveal that DeepSeek-V3 outperforms other open-source models and achieves performance comparable to leading closed-source models. Despite its excellent performance, DeepSeek-V3 requires only 2.788M H800 GPU hours for its full training. In addition, its training process is remarkably stable. Throughout the entire training process, we did not experience any irrecoverable loss spikes or perform any rollbacks. The model checkpoints are available at https://github.com/deepseek-ai/DeepSeek-V3.

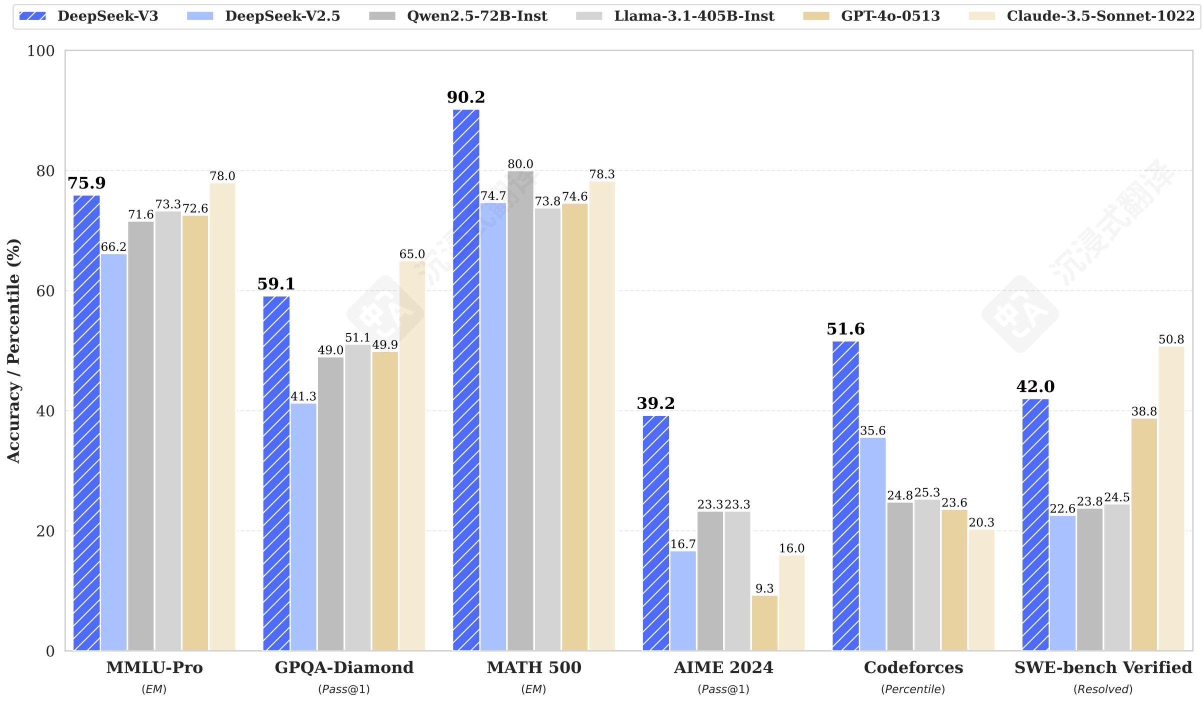


Figure 1 | Benchmark performance of DeepSeek-V3 and its counterparts.

DeepSeek-V3 技术报告

DeepSeek-AI

research@deepseek.com

摘要

我们介绍了 DeepSeek-V3，一个强大的专家混合（MoE）语言模型，总参数量为 671B，每个 token 激活参数量为 37B。为了实现高效的推理和成本效益的训练，DeepSeek-V3 采用了多头潜在注意力（MLA）和 DeepSeekMoE 架构，这些架构在 DeepSeek-V2 中得到了彻底的验证。此外，DeepSeek-V3 创新了一种无辅助损失的负载均衡策略，并设定了多 token 预测训练目标以实现更强的性能。我们在 14.8 万亿个多样且高质量的 token 上预训练 DeepSeek-V3，然后进行监督微调和强化学习阶段，以充分发挥其能力。综合评估表明，DeepSeek-V3 优于其他开源模型，并实现了与领先闭源模型相当的性能。尽管性能优异，DeepSeek-V3 的完整训练仅需 2.788M H800 GPU 小时。此外，其训练过程非常稳定。在整个训练过程中，我们没有遇到任何不可恢复的损失峰值或进行任何回滚。模型检查点可在 https://github.com/deepseek-ai/DeepSeek-V3处获取。

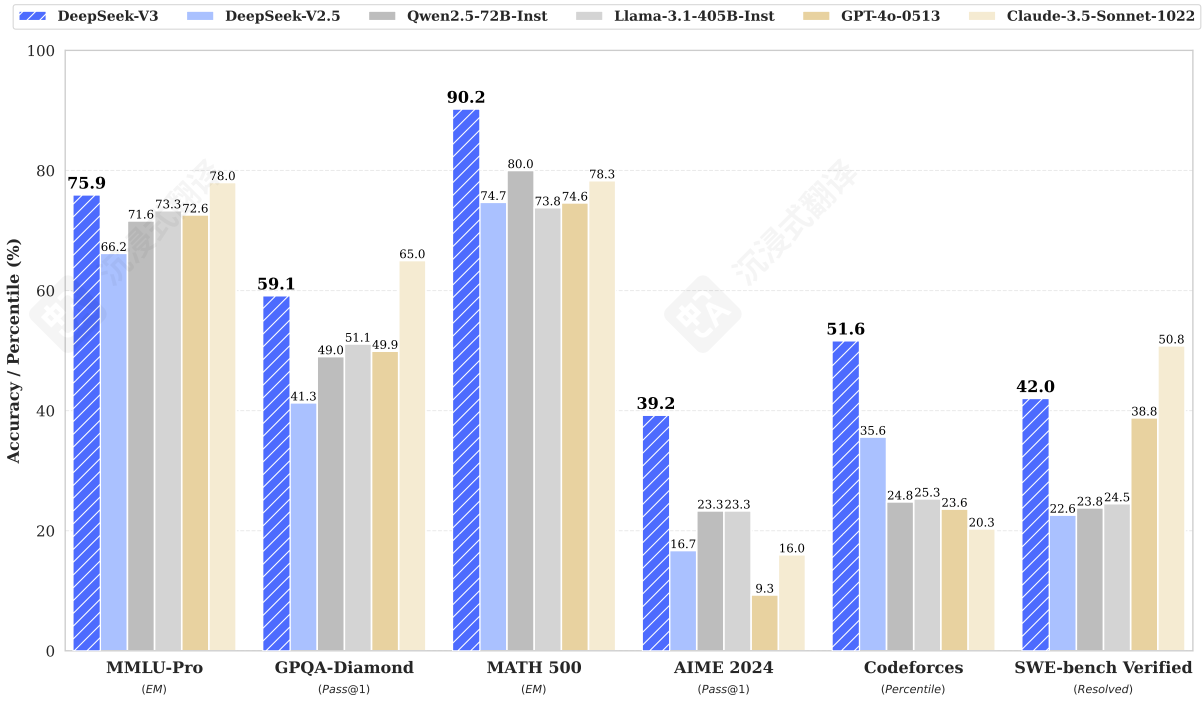


图 1 | DeepSeek-V3及其对应模型的基准性能。

Contents

目录

1. Introduction

In recent years, Large Language Models (LLMs) have been undergoing rapid iteration and evolution (Anthropic, 2024; Google, 2024; OpenAI, 2024a), progressively diminishing the gap towards Artificial General Intelligence (AGI). Beyond closed-source models, open-source models, including DeepSeek series (DeepSeek-AI, 2024a,b,c; Guo et al., 2024), LLaMA series (AI@Meta, 2024a,b; Touvron et al., 2023a,b), Qwen series (Qwen, 2023, 2024a,b), and Mistral series (Jiang et al., 2023; Mistral, 2024), are also making significant strides, endeavoring to close the gap with their closed-source counterparts. To further push the boundaries of open-source model capabilities, we scale up our models and introduce DeepSeek-V3, a large Mixture-of-Experts (MoE) model with 671B parameters, of which 37B are activated for each token.

With a forward-looking perspective, we consistently strive for strong model performance and economical costs. Therefore, in terms of architecture, DeepSeek-V3 still adopts Multi-head Latent Attention (MLA) (DeepSeek-AI, 2024c) for efficient inference and DeepSeekMoE (Dai et al., 2024) for cost-effective training. These two architectures have been validated in DeepSeek-V2 (DeepSeek-AI, 2024c), demonstrating their capability to maintain robust model performance while achieving efficient training and inference. Beyond the basic architecture, we implement two additional strategies to further enhance the model capabilities. Firstly, DeepSeek-V3 pioneers an auxiliary-loss-free strategy (Wang et al., 2024a) for load balancing, with the aim of minimizing the adverse impact on model performance that arises from the effort to encourage load balancing. Secondly, DeepSeek-V3 employs a multi-token prediction training objective, which we have observed to enhance the overall performance on evaluation benchmarks.

In order to achieve efficient training, we support the FP8 mixed precision training and implement comprehensive optimizations for the training framework. Low-precision training has emerged as a promising solution for efficient training (Dettmers et al., 2022; Kalamkar et al., 2019; Narang et al., 2017; Peng et al., 2023b), its evolution being closely tied to advancements in hardware capabilities (Luo et al., 2024; Micikevicius et al., 2022; Rouhani et al., 2023a). In this work, we introduce an FP8 mixed precision training framework and, for the first time, validate its effectiveness on an extremely large-scale model. Through the support for FP8 computation and storage, we achieve both accelerated training and reduced GPU memory usage. As for the training framework, we design the DualPipe algorithm for efficient pipeline parallelism, which has fewer pipeline bubbles and hides most of the communication during training through computation-communication overlap. This overlap ensures that, as the model further scales up, as long as we maintain a constant computation-to-communication ratio, we can still employ fine-grained experts across nodes while achieving a near-zero all-to-all communication overhead. In addition, we also develop efficient cross-node all-to-all communication kernels to fully utilize InfiniBand (IB) and NVLink bandwidths. Furthermore, we meticulously optimize the memory footprint, making it possible to train DeepSeek-V3 without using costly tensor parallelism. Combining these efforts, we achieve high training efficiency.

During pre-training, we train DeepSeek-V3 on 14.8T high-quality and diverse tokens. The pre-training process is remarkably stable. Throughout the entire training process, we did not encounter any irrecoverable loss spikes or have to roll back. Next, we conduct a two-stage context length extension for DeepSeek-V3. In the first stage, the maximum context length is extended to 32K, and in the second stage, it is further extended to 128K. Following this, we conduct post-training, including Supervised Fine-Tuning (SFT) and Reinforcement Learning (RL) on the base model of DeepSeek-V3, to align it with human preferences and further unlock its potential. During the post-training stage, we distill the reasoning capability from the DeepSeek-R1 series of models, and meanwhile carefully maintain the balance between model accuracy

1. 简介

近年来，大型语言模型（LLMs）正在经历快速迭代和演进（Anthropic，2024；Google，2024；OpenAI，2024a），逐步缩小与通用人工智能（AGI）的差距。除了闭源模型，开源模型，包括DeepSeek系列（DeepSeek-AI，2024a,b,c；Guo等人，2024）、LLaMA系列（AI@Meta，2024a,b；Touvron等人，2023a,b）、Qwen系列（Qwen，2023，2024a,b）和Mistral系列（Jiang等人，2023；Mistral，2024），也在取得显著进展，努力缩小与它们的闭源版本之间的差距。为了进一步推动开源模型能力的边界，我们扩展了我们的模型并推出了DeepSeek-V3，这是一个拥有671B参数的大型专家混合（MoE）模型，其中每个token有37B参数被激活。

以前瞻性的视角，我们始终致力于实现强大的模型性能和经济的成本。因此，在架构方面，DeepSeek-V3 仍然采用多头潜在注意力（MLA）（DeepSeek-AI，2024c）以实现高效的推理，并采用 DeepSeekMoE（Dai 等人，2024）以实现经济高效的训练。这两种架构已在 DeepSeek-V2（DeepSeek-AI，2024c）中得到验证，证明它们能够在实现高效的训练和推理的同时保持强大的模型性能。除了基本架构之外，我们实施了两种额外的策略，以进一步提升模型能力。首先，DeepSeek-V3 引领了一种无辅助损失的策略（Wang 等人，2024a）用于负载均衡，旨在最大限度地减少因鼓励负载均衡而产生的对模型性能的不利影响。其次，DeepSeek-V3 采用多标记预测训练目标，我们观察到这有助于提升在评估基准上的整体性能。

为了实现高效的训练，我们支持FP8混合精度训练，并对训练框架进行了全面的优化。低精度训练已成为高效训练的一种有前景的解决方案（Dettmers等人，2022年；Kalamkar等人，2019年；Narang等人，2017年；Peng等人，2023b年），其发展紧密依赖于硬件能力的进步（Luo等人，2024年；Micikevicius等人，2022年；Rouhani等人，2023a年）。在这项工作中，我们介绍了一个FP8混合精度训练框架，并且首次在极大规模模型上验证了其有效性。通过支持FP8计算和存储，我们实现了加速训练和减少GPU内存使用。对于训练框架，我们设计了DualPipe算法以实现高效的流水线并行，该算法具有更少的流水线气泡，并通过计算-通信重叠在训练过程中隐藏了大部分通信。这种重叠确保了，随着模型的进一步扩展，只要我们保持计算与通信的比率恒定，我们仍然可以在节点之间使用细粒度专家，同时实现近零的全对全通信开销。此外，我们还开发了高效的跨节点全对全通信内核，以充分利用InfiniBand（IB）和NVLink带宽。此外，我们还精心优化了内存占用，使得无需使用昂贵的张量并行即可训练DeepSeek-V3。结合这些努力，我们实现了高训练效率。

在预训练阶段，我们在 14.8T 的高质量多样化 token 上训练 DeepSeek-V3。预训练过程非常稳定。在整个训练过程中，我们没有遇到任何不可恢复的损失峰值，也不需要回滚。接下来，我们对 DeepSeek-V3 进行了分两阶段的上下文长度扩展。在第一阶段，最大上下文长度扩展到 32K，在第二阶段，进一步扩展到 128K。随后，我们对 DeepSeek-V3 的基础模型进行后训练，包括监督微调（SFT）和强化学习（RL），以使其与人类偏好保持一致并进一步释放其潜力。在后训练阶段，我们从 DeepSeek-R1 系列模型中蒸馏推理能力，同时小心地保持模型准确性与

Training Costs	Pre-Training	Context Extension	Post-Training	Total
in H800 GPU Hours	2664K	119K	5K	2788K
in USD	$5.328M	$0.238M	$0.01M	$5.576M

Table 1 | Training costs of DeepSeek-V3, assuming the rental price of H800 is $2 per GPU hour.

and generation length.

We evaluate DeepSeek-V3 on a comprehensive array of benchmarks. Despite its economical training costs, comprehensive evaluations reveal that DeepSeek-V3-Base has emerged as the strongest open-source base model currently available, especially in code and math. Its chat version also outperforms other open-source models and achieves performance comparable to leading closed-source models, including GPT-4o and Claude-3.5-Sonnet, on a series of standard and open-ended benchmarks.

Lastly, we emphasize again the economical training costs of DeepSeek-V3, summarized in Table 1, achieved through our optimized co-design of algorithms, frameworks, and hardware. During the pre-training stage, training DeepSeek-V3 on each trillion tokens requires only 180K H800 GPU hours, i.e., 3.7 days on our cluster with 2048 H800 GPUs. Consequently, our pre-training stage is completed in less than two months and costs 2664K GPU hours. Combined with 119K GPU hours for the context length extension and 5K GPU hours for post-training, DeepSeek-V3 costs only 2.788M GPU hours for its full training. Assuming the rental price of the H800 GPU is $2 per GPU hour, our total training costs amount to only $5.576M. Note that the aforementioned costs include only the official training of DeepSeek-V3, excluding the costs associated with prior research and ablation experiments on architectures, algorithms, or data.

Our main contribution includes:

Architecture: Innovative Load Balancing Strategy and Training Objective

- On top of the efficient architecture of DeepSeek-V2, we pioneer an auxiliary-loss-free strategy for load balancing, which minimizes the performance degradation that arises from encouraging load balancing.
- We investigate a Multi-Token Prediction (MTP) objective and prove it beneficial to model performance. It can also be used for speculative decoding for inference acceleration.

Pre-Training: Towards Ultimate Training Efficiency

- We design an FP8 mixed precision training framework and, for the first time, validate the feasibility and effectiveness of FP8 training on an extremely large-scale model.
- Through the co-design of algorithms, frameworks, and hardware, we overcome the communication bottleneck in cross-node MoE training, achieving near-full computation-communication overlap. This significantly enhances our training efficiency and reduces the training costs, enabling us to further scale up the model size without additional overhead.
- At an economical cost of only 2.664M H800 GPU hours, we complete the pre-training of DeepSeek-V3 on 14.8T tokens, producing the currently strongest open-source base model. The subsequent training stages after pre-training require only 0.1M GPU hours.

Post-Training: Knowledge Distillation from DeepSeek-R1

- We introduce an innovative methodology to distill reasoning capabilities from the long-Chain-of-Thought (CoT) model, specifically from one of the DeepSeek R1 series models, into standard LLMs, particularly DeepSeek-V3. Our pipeline elegantly incorporates the

训练成本	预训练上下文扩展后训练			总计
在 H800 GPU 小时	2664K	119K	5K	2788K
in USD	$5.328M	$0.238M	$0.01M	$5.576M

Table 1 | Training costs of DeepSeek-V3, assuming the rental price of H800 is $2 per GPU hour.

and generation length.

我们在一系列综合基准测试中评估了 DeepSeek-V3。尽管其训练成本经济，但全面评估表明，DeepSeek-V3-Base 是目前最强的开源基础模型，尤其是在代码和数学方面。其聊天版本也优于其他开源模型，并在一系列标准和开放式基准测试中达到了与领先闭源模型（包括 GPT-4o 和 Claude-3.5-Sonnet）相当的性能。

最后，我们再次强调 DeepSeek-V3 的经济训练成本，如表 1 所示，这是通过我们对算法、框架和硬件的优化协同设计实现的。在预训练阶段，每个万亿 token 的 DeepSeek-V3 训练仅需 180K H800 GPU 小时，即在我们拥有 2048 个 H800 GPU 的集群上只需 3.7 天。因此，我们的预训练阶段在不到两个月内完成，成本为 2664K GPU 小时。加上用于上下文长度扩展的 119K GPU 小时和用于后训练的 5K GPU 小时，DeepSeek-V3 的完整训练成本仅为 2.788M GPU 小时。假设 H800 GPU 的租赁价格为每 GPU 小时 2 美元，我们的总训练成本仅为 5.576M 美元。请注意，上述成本仅包括 DeepSeek-V3 的官方训练，不包括先前在架构、算法或数据上的研究和消融实验的成本。

我们的主要贡献包括：

架构：创新的负载均衡策略和训练目标

- 在DeepSeek-V2的高效架构基础上，我们开创了一种无辅助损失的负载均衡策略，该策略最大限度地减少了因鼓励负载均衡而导致的性能下降。
- 我们研究了一种多标记预测（MTP）目标，并证明其对模型性能有益。它也可以用于推测解码以加速推理。

Pre-Training: Towards Ultimate Training Efficiency

- 我们设计了一种FP8混合精度训练框架，并首次验证了在极其大规模模型上进行FP8训练的可行性和有效性。
- 通过算法、框架和硬件的协同设计，我们克服了跨节点MoE训练中的通信瓶颈，实现了接近完全的计算-通信重叠。这显著提高了我们的训练效率，降低了训练成本，使我们能够在不增加额外开销的情况下进一步扩大模型规模。
- 仅花费2.664M H800 GPU小时的经济成本，我们完成了DeepSeek-V3在14.8T tokens上的预训练，生成了目前最强的开源基模型。预训练后的后续训练阶段仅需0.1M GPU小时。

后训练：从DeepSeek-R1进行知识蒸馏

- 我们引入了一种创新的方法，将长链式思维（CoT）模型的推理能力从DeepSeek R1系列模型之一蒸馏到标准LLM中，特别是DeepSeek-V3。我们的流程优雅地结合了

verification and reflection patterns of R1 into DeepSeek-V3 and notably improves its reasoning performance. Meanwhile, we also maintain control over the output style and length of DeepSeek-V3.

Summary of Core Evaluation Results

- **Knowledge**: (1) On educational benchmarks such as MMLU, MMLU-Pro, and GPQA, DeepSeek-V3 outperforms all other open-source models, achieving 88.5 on MMLU, 75.9 on MMLU-Pro, and 59.1 on GPQA. Its performance is comparable to leading closed-source models like GPT-4o and Claude-Sonnet-3.5, narrowing the gap between open-source and closed-source models in this domain. (2) For factuality benchmarks, DeepSeek-V3 demonstrates superior performance among open-source models on both SimpleQA and Chinese SimpleQA. While it trails behind GPT-4o and Claude-Sonnet-3.5 in English factual knowledge (SimpleQA), it surpasses these models in Chinese factual knowledge (Chinese SimpleQA), highlighting its strength in Chinese factual knowledge.
- **Code, Math, and Reasoning**: (1) DeepSeek-V3 achieves state-of-the-art performance on math-related benchmarks among all non-long-CoT open-source and closed-source models. Notably, it even outperforms o1-preview on specific benchmarks, such as MATH-500, demonstrating its robust mathematical reasoning capabilities. (2) On coding-related tasks, DeepSeek-V3 emerges as the top-performing model for coding competition benchmarks, such as LiveCodeBench, solidifying its position as the leading model in this domain. For engineering-related tasks, while DeepSeek-V3 performs slightly below Claude-Sonnet-3.5, it still outpaces all other models by a significant margin, demonstrating its competitiveness across diverse technical benchmarks.

In the remainder of this paper, we first present a detailed exposition of our DeepSeek-V3 model architecture (Section 2). Subsequently, we introduce our infrastructures, encompassing our compute clusters, the training framework, the support for FP8 training, the inference deployment strategy, and our suggestions on future hardware design. Next, we describe our pre-training process, including the construction of training data, hyper-parameter settings, long-context extension techniques, the associated evaluations, as well as some discussions (Section 4). Thereafter, we discuss our efforts on post-training, which include Supervised Fine-Tuning (SFT), Reinforcement Learning (RL), the corresponding evaluations, and discussions (Section 5). Lastly, we conclude this work, discuss existing limitations of DeepSeek-V3, and propose potential directions for future research (Section 6).

2. Architecture

We first introduce the basic architecture of DeepSeek-V3, featured by Multi-head Latent Attention (MLA) (DeepSeek-AI, 2024c) for efficient inference and DeepSeekMoE (Dai et al., 2024) for economical training. Then, we present a Multi-Token Prediction (MTP) training objective, which we have observed to enhance the overall performance on evaluation benchmarks. For other minor details not explicitly mentioned, DeepSeek-V3 adheres to the settings of DeepSeek-V2 (DeepSeek-AI, 2024c).

2.1. Basic Architecture

The basic architecture of DeepSeek-V3 is still within the Transformer (Vaswani et al., 2017) framework. For efficient inference and economical training, DeepSeek-V3 also adopts MLA and DeepSeekMoE, which have been thoroughly validated by DeepSeek-V2. Compared with DeepSeek-V2, an exception is that we additionally introduce an auxiliary-loss-free load balancing

将 R1 的验证和反思模式整合到 DeepSeek-V3 中，并显著提升了其推理性能。同时，我们也控制了 DeepSeek-V3 的输出风格和长度。

核心评估结果摘要

- **知识**: (1) 在 MMLU、MMLU-Pro 和 GPQA 等教育基准测试中，DeepSeek-V3 的表现优于所有其他开源模型，在 MMLU 上达到 88.5，在 MMLU-Pro 上达到 75.9，在 GPQA 上达到 59.1。其性能与 GPT-4o 和 Claude-Sonnet-3.5 等领先闭源模型相当，缩小了该领域开源模型与闭源模型之间的差距。(2) 在事实性基准测试中，DeepSeek-V3 在 SimpleQA 和 Chinese SimpleQA 上均表现出色。虽然它在英语事实知识（SimpleQA）方面落后于 GPT-4o 和 Claude-Sonnet-3.5，但在中文事实知识（Chinese SimpleQA）方面超越了这些模型，突显了其在中文事实知识方面的优势。

- **代码、数学和推理**: (1) DeepSeek-V3 在所有非长-CoT 开源和闭源模型中，在数学相关基准测试中实现了最先进的性能。值得注意的是，它在特定基准测试（如 MATH-500）上甚至优于 o1-preview，展示了其强大的数学推理能力。(2) 在编程相关任务中，DeepSeek-V3 成为编码竞赛基准测试（如 LiveCodeBench）表现最佳的模型，巩固了其在该领域的领先地位。对于工程相关任务，虽然 DeepSeek-V3 的表现略低于 Claude-Sonnet-3.5，但它仍然显著优于所有其他模型，展示了其在多样化技术基准测试中的竞争力。

在本文的其余部分，我们首先详细阐述我们的 DeepSeek-V3 模型架构（第 2 节）。随后，我们介绍我们的基础设施，包括我们的计算集群、训练框架、FP8 训练支持、推理部署策略以及我们对未来硬件设计的建议。接下来，我们描述我们的预训练过程，包括训练数据构建、超参数设置、长上下文扩展技术、相关评估以及一些讨论（第 4 节）。此后，我们讨论我们的后训练工作，包括监督微调（SFT）、强化学习（RL）、相应的评估和讨论（第 5 节）。最后，我们总结这项工作，讨论 DeepSeek-V3 现有的局限性，并提出未来研究的潜在方向（第 6 节）。

2. 架构

我们首先介绍了 DeepSeek-V3 的基本架构，其特点是采用多头潜在注意力 (MLA) (DeepSeek-AI, 2024c) 以实现高效推理，以及 DeepSeekMoE (Dai 等人, 2024) 以实现经济高效的训练。然后，我们提出了一种多标记预测 (MTP) 训练目标，我们观察到该目标在评估基准上提升了整体性能。对于其他未明确提及的细节，DeepSeek-V3 遵循 DeepSeek-V2 的设置 (DeepSeek-AI, 2024c)。

2.1. 基本架构

DeepSeek-V3 的基本架构仍然在 Transformer (Vaswani 等人, 2017) 框架内。为了实现高效推理和经济高效的训练，DeepSeek-V3 也采用了 MLA 和 DeepSeekMoE，这些技术已经在 DeepSeek-V2 中得到了充分验证。与 DeepSeek-V2 相比，一个例外是我们额外引入了一种无辅助损失负载均衡

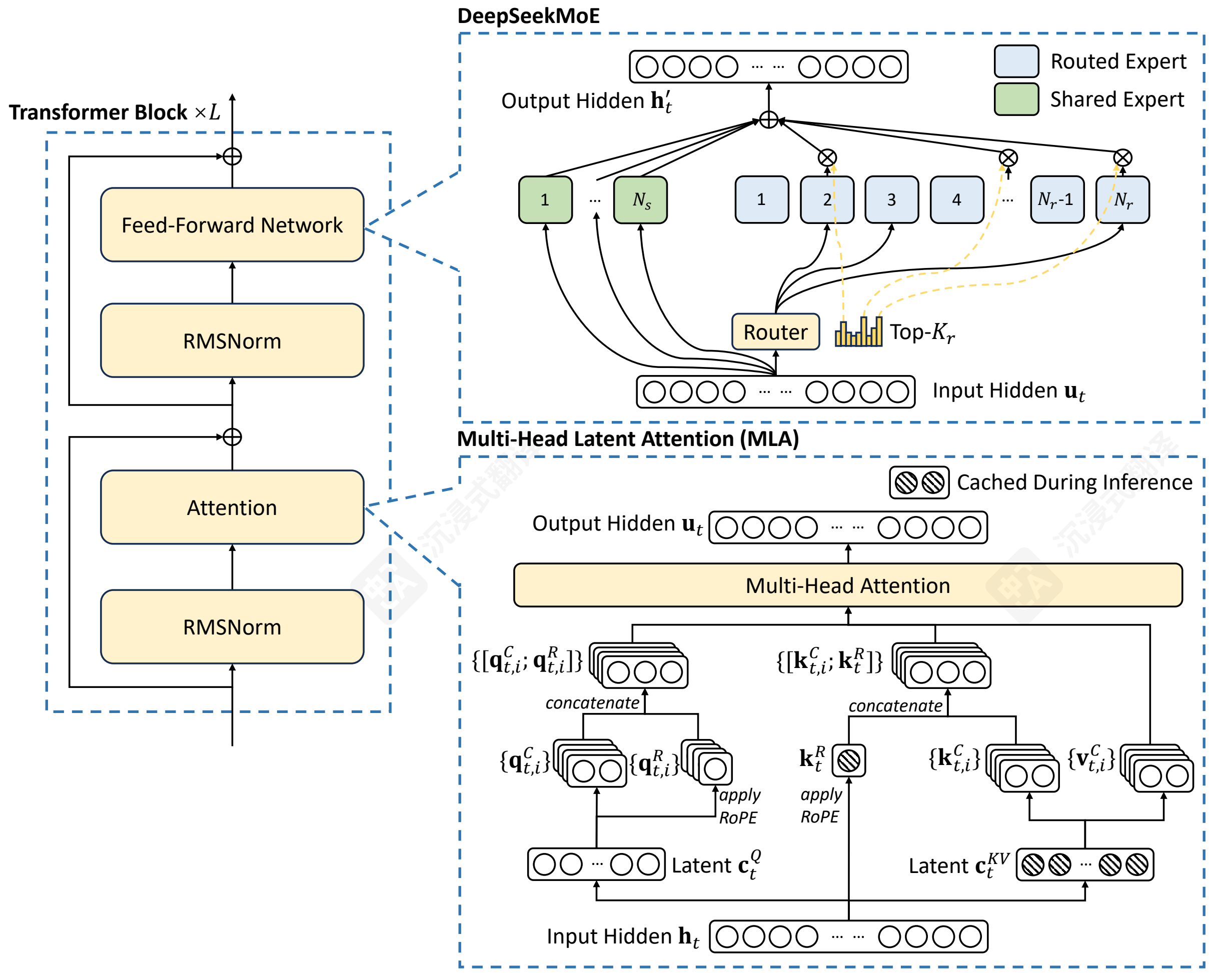


Figure 2 | Illustration of the basic architecture of DeepSeek-V3. Following DeepSeek-V2, we adopt MLA and DeepSeekMoE for efficient inference and economical training.

strategy (Wang et al., 2024a) for DeepSeekMoE to mitigate the performance degradation induced by the effort to ensure load balance. Figure 2 illustrates the basic architecture of DeepSeek-V3, and we will briefly review the details of MLA and DeepSeekMoE in this section.

2.1.1. *Multi-Head Latent Attention*

For attention, DeepSeek-V3 adopts the MLA architecture. Let d denote the embedding dimension, n_h denote the number of attention heads, d_h denote the dimension per head, and $\mathbf{h}_t \in \mathbb{R}^d$ denote the attention input for the t-th token at a given attention layer. The core of MLA is the low-rank joint compression for attention keys and values to reduce Key-Value (KV) cache during inference:

$$\boxed{\color{blue}{\mathbf{c}_t^{KV}}} = W^{DKV}\mathbf{h}_t, \tag{1}$$

$$[\mathbf{k}^C_{t,1}; \mathbf{k}^C_{t,2}; ...; \mathbf{k}^C_{t,n_h}] = \mathbf{k}^C_t = W^{UK}\mathbf{c}^{KV}_t, \tag{2}$$

$$\boxed{\color{blue}{\mathbf{k}^R_t}} = \operatorname{RoPE}(W^{KR}\mathbf{h}_t), \tag{3}$$

$$\mathbf{k}_{t,i} = [\mathbf{k}^C_{t,i}; \mathbf{k}^R_t], \tag{4}$$

$$[\mathbf{v}^C_{t,1}; \mathbf{v}^C_{t,2}; ...; \mathbf{v}^C_{t,n_h}] = \mathbf{v}^C_t = W^{UV}\mathbf{c}^{KV}_t, \tag{5}$$

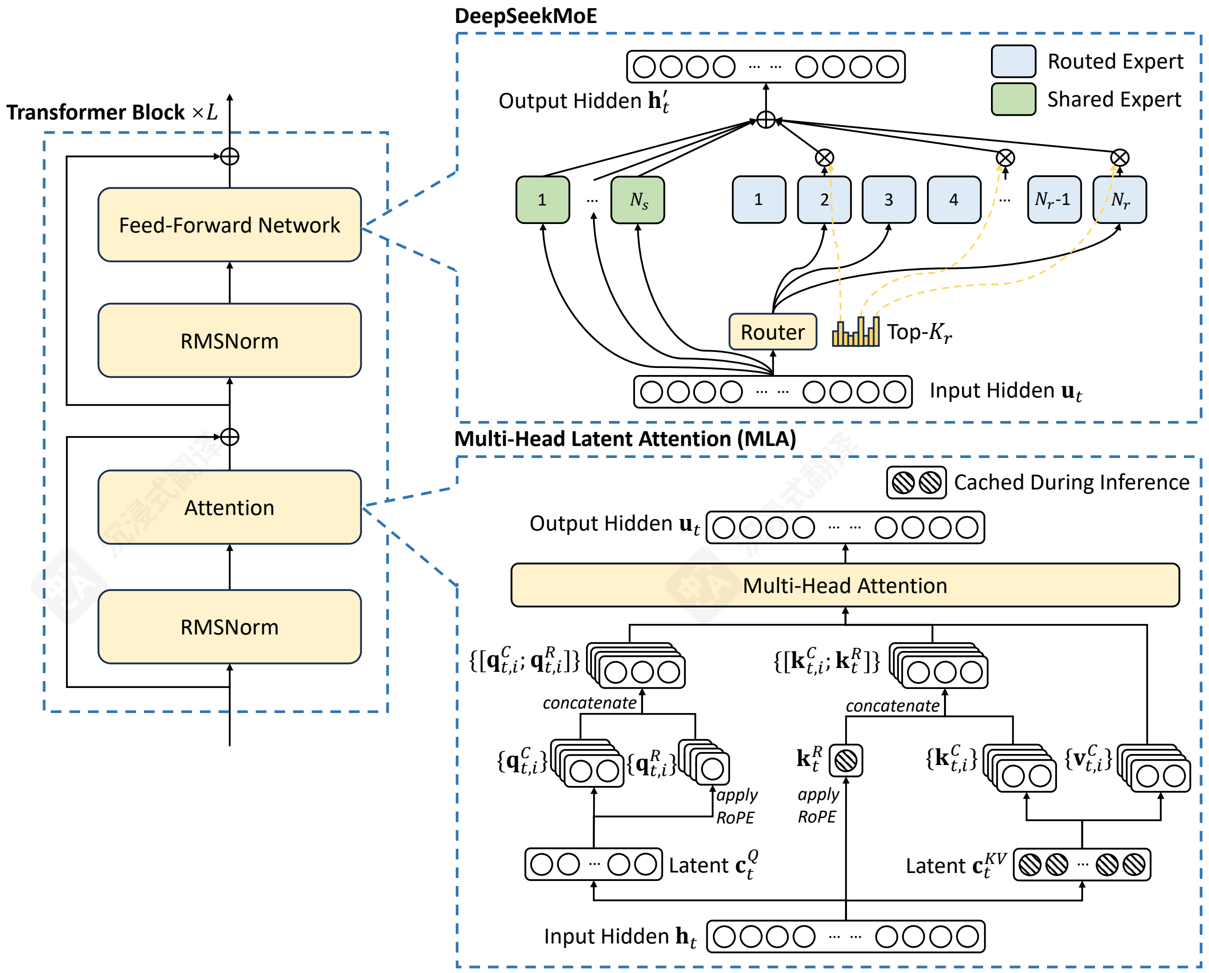


图 2 | DeepSeek-V3的基本架构图。继DeepSeek-V2之后，我们采用MLA和DeepSeek MoE以实现高效的推理和经济的训练。

策略（Wang等人，2024a）用于DeepSeekMoE以减轻因确保负载平衡而产生的性能下降。图2说明了DeepSeek-V3的基本架构，在本节中我们将简要回顾MLA和DeepSeekMoE的细节。

2.1.1. 多头潜在注意力

对于注意力机制，DeepSeek-V3采用了MLA架构。令 d 表示嵌入维度，n_h 表示注意力头的数量，d_h 表示每个头的维度，$\mathbf{h}_t \in \mathbb{R}^d$表示在给定注意力层中第$t$个token的注意力输入。MLA的核心是对注意力键和值进行低秩联合压缩，以减少推理过程中的键值（KV）缓存：

$$\left|\color{blue}{\mathbf{c}_t^{KV}}\right| = W^{DKV}\mathbf{h}_t, \tag{1}$$

$$[\mathbf{k}^C_{t,1}; \mathbf{k}^C_{t,2}; ...; \mathbf{k}^C_{t,n_h}] = \mathbf{k}^C_t = W^{UK}\mathbf{c}^{KV}_t, \tag{2}$$

$$\left|\color{blue}{\mathbf{k}^R_t}\right| = \operatorname{RoPE}(W^{KR}\mathbf{h}_t), \tag{3}$$

$$\mathbf{k}_{t,i} = [\mathbf{k}^C_{t,i}; \mathbf{k}^R_t], \tag{4}$$

$$[\mathbf{v}^C_{t,1}; \mathbf{v}^C_{t,2}; ...; \mathbf{v}^C_{t,n_h}] = \mathbf{v}^C_t = W^{UV}\mathbf{c}^{KV}_t, \tag{5}$$

where $\mathbf{c}_t^{KV} \in \mathbb{R}^{d_c}$ is the compressed latent vector for keys and values; $d_c (\ll d_h n_h)$ indicates the KV compression dimension; $W^{DKV} \in \mathbb{R}^{d_c \times d}$ denotes the down-projection matrix; $W^{UK}, W^{UV} \in \mathbb{R}^{d_h n_h \times d_c}$ are the up-projection matrices for keys and values, respectively; $W^{KR} \in \mathbb{R}^{d_h^R \times d}$ is the matrix used to produce the decoupled key that carries Rotary Positional Embedding (RoPE) (Su et al., 2024); $\mathrm{RoPE}(\cdot)$ denotes the operation that applies RoPE matrices; and $[\cdot;\cdot]$ denotes concatenation. Note that for MLA, only the blue-boxed vectors (i.e., $\mathbf{c}_t^{KV}$ and $\mathbf{k}_t^R$) need to be cached during generation, which results in significantly reduced KV cache while maintaining performance comparable to standard Multi-Head Attention (MHA) (Vaswani et al., 2017).

For the attention queries, we also perform a low-rank compression, which can reduce the activation memory during training:

$$\mathbf{c}_t^Q = W^{DQ} \mathbf{h}_t, \tag{6}$$

$$[\mathbf{q}_{t,1}^C; \mathbf{q}_{t,2}^C; ...; \mathbf{q}_{t,n_h}^C] = \mathbf{q}_t^C = W^{UQ} \mathbf{c}_t^Q, \tag{7}$$

$$[\mathbf{q}_{t,1}^R; \mathbf{q}_{t,2}^R; ...; \mathbf{q}_{t,n_h}^R] = \mathbf{q}_t^R = \mathrm{RoPE}(W^{QR} \mathbf{c}_t^Q), \tag{8}$$

$$\mathbf{q}_{t,i} = [\mathbf{q}_{t,i}^C; \mathbf{q}_{t,i}^R], \tag{9}$$

where $\mathbf{c}_t^Q \in \mathbb{R}^{d_c'}$ is the compressed latent vector for queries; $d_c' (\ll d_h n_h)$ denotes the query compression dimension; $W^{DQ} \in \mathbb{R}^{d_c' \times d}, W^{UQ} \in \mathbb{R}^{d_h n_h \times d_c'}$ are the down-projection and up-projection matrices for queries, respectively; and $W^{QR} \in \mathbb{R}^{d_h^R n_h \times d_c'}$ is the matrix to produce the decoupled queries that carry RoPE.

Ultimately, the attention queries ($\mathbf{q}_{t,i}$), keys ($\mathbf{k}_{j,i}$), and values ($\mathbf{v}_{j,i}^C$) are combined to yield the final attention output $\mathbf{u}_t$:

$$\mathbf{o}_{t,i} = \sum_{j=1}^{t} \mathrm{Softmax}_j \left(\frac{\mathbf{q}_{t,i}^T \mathbf{k}_{j,i}}{\sqrt{d_h + d_h^R}} \right) \mathbf{v}_{j,i}^C, \tag{10}$$

$$\mathbf{u}_t = W^O [\mathbf{o}_{t,1}; \mathbf{o}_{t,2}; ...; \mathbf{o}_{t,n_h}], \tag{11}$$

where $W^O \in \mathbb{R}^{d \times d_h n_h}$ denotes the output projection matrix.

2.1.2. *DeepSeekMoE with Auxiliary-Loss-Free Load Balancing*

Basic Architecture of DeepSeekMoE. For Feed-Forward Networks (FFNs), DeepSeek-V3 employs the DeepSeekMoE architecture (Dai et al., 2024). Compared with traditional MoE architectures like GShard (Lepikhin et al., 2021), DeepSeekMoE uses finer-grained experts and isolates some experts as shared ones. Let $\mathbf{u}_t$ denote the FFN input of the t-th token, we compute the FFN output $\mathbf{h}_t'$ as follows:

$$\mathbf{h}_t' = \mathbf{u}_t + \sum_{i=1}^{N_s} \mathrm{FFN}_i^{(s)}(\mathbf{u}_t) + \sum_{i=1}^{N_r} g_{i,t} \, \mathrm{FFN}_i^{(r)}(\mathbf{u}_t), \tag{12}$$

$$g_{i,t} = \frac{g_{i,t}'}{\sum_{j=1}^{N_r} g_{j,t}'}, \tag{13}$$

$$g_{i,t}' = \begin{cases} s_{i,t}, & s_{i,t} \in \mathrm{Topk}(\{s_{j,t} | 1 \leqslant j \leqslant N_r\}, K_r), \\ 0, & \text{otherwise}, \end{cases} \tag{14}$$

$$s_{i,t} = \mathrm{Sigmoid}\left({\mathbf{u}_t}^T \mathbf{e}_i \right), \tag{15}$$

where $\mathbf{c}_t^{KV} \in \mathbb{R}^{d_c}$ 是用于键和值的压缩潜在向量； $d_c (\ll d_h n_h)$ 表示KV压缩维度； $W^{DKV} \in \mathbb{R}^{d_c \times d}$ 表示下投影矩阵； $W^{UK}, W^{UV} \in \mathbb{R}^{d_h n_h \times d_c}$分别是用于键和值的上投影矩阵； $W^{KR} \in \mathbb{R}^{d^R{}_h \times d}$是用于生成携带旋转位置嵌入（RoPE）的解耦键的矩阵（Su等人，2024）；RoPE$(\cdot)$ 表示应用RoPE矩阵的操作；并且 $[\cdot;\cdot]$ 表示拼接。请注意，对于MLA，在生成过程中只需要缓存蓝色框内的向量（即 $\mathbf{c}_t^{KV}$和 $\mathbf{k}_t^R$），这显著减少了KV缓存，同时保持了与标准多头注意力（MHA）相当的性能（Vaswani等人，2017）。

对于注意力查询，我们还执行低秩压缩，这可以减少训练期间的激活内存：

$$\mathbf{c}_t^Q = W^{DQ} \mathbf{h}_t, \tag{6}$$

$$[\mathbf{q}_{t,1}^C; \mathbf{q}_{t,2}^C; ...; \mathbf{q}_{t,n_h}^C] = \mathbf{q}_t^C = W^{UQ} \mathbf{c}_t^Q, \tag{7}$$

$$[\mathbf{q}_{t,1}^R; \mathbf{q}_{t,2}^R; ...; \mathbf{q}_{t,n_h}^R] = \mathbf{q}_t^R = \mathrm{RoPE}(W^{QR} \mathbf{c}_t^Q), \tag{8}$$

$$\mathbf{q}_{t,i} = [\mathbf{q}_{t,i}^C; \mathbf{q}_{t,i}^R], \tag{9}$$

在 $\mathbf{c}_t^Q \in \mathbb{R}^{d_c'}$ 是查询的压缩潜在向量； $d_{\prime c} (\ll d_h n_h)$表示查询压缩维度； $W^{DQ} \in \mathbb{R}^{d_c' \times d}$, $W^{UQ} \in \mathbb{R}^{d_h n_h \times d_{\prime c}}$分别是查询的下投影和上投影矩阵；而 $W^{QR} \in \mathbb{R}^{d^R{}_h n_h \times d_{\prime c}}$是用于生成携带RoPE的解耦查询的矩阵。

最终，注意力查询 ($\mathbf{q}_{t,i}$)、键 ($\mathbf{k}_{j,i}$) 和值 ($\overset{C}{\mathbf{v}_j}\, i,$) 被组合以产生最终的注意力输出$\mathbf{u}_t$:

$$\mathbf{o}_{t,i} = \sum_{j=1}^{t} \mathrm{Softmax}_j (\frac{\mathbf{q}_{t,i}^T \mathbf{k}_{j,i}}{\sqrt{d_h + d_h^R}}) \mathbf{v}_{j,i}^C, \tag{10}$$

$$\mathbf{u}_t = W^O [\mathbf{o}_{t,1}; \mathbf{o}_{t,2}; ...; \mathbf{o}_{t,n_h}], \tag{11}$$

其中$W^O \in \mathbb{R}^{d \times d_h n_h}$ 表示输出投影矩阵。

2.1.2. *DeepSeekMoE* 带有辅助损失无负载均衡

DeepSeekMoE的基本架构。 对于前馈网络（FFNs），DeepSeek-V3采用了DeepSeekMoE架构（Dai等人，2024年）。与传统的MoE架构（如GShard（Lepikhin等人，2021年））相比，DeepSeekMoE使用更细粒度的专家，并将一些专家隔离为共享专家。令 $\mathbf{u}_t$ 表示第t个token的FFN输入，我们按如下方式计算FFN输出$\mathbf{h}_t'$：

$$\mathbf{h}_t' = \mathbf{u}_t + \sum_{i=1}^{N_s} \mathrm{FFN}_i^{(s)}(\mathbf{u}_t) + \sum_{i=1}^{N_r} g_{i,t} \, \mathrm{FFN}_i^{(r)}(\mathbf{u}_t), \tag{12}$$

$$g_{i,t} = \frac{g_{i,t}'}{\sum_{j=1}^{N_r} g_{j,t}'}, \tag{13}$$

$$g_{i,t}' = \begin{cases} s_{i,t}, & s_{i,t} \in \mathrm{Topk}(\{s_{j,t} | 1 \leqslant j \leqslant N_r\}, K_r), \\ 0, & \text{otherwise}, \end{cases} \tag{14}$$

$$s_{i,t} = \mathrm{Sigmoid}\left({\mathbf{u}_t}^T \mathbf{e}_i \right), \tag{15}$$

where N_s and N_r denote the numbers of shared experts and routed experts, respectively; $\text{FFN}_i^{(s)}(\cdot)$ and $\text{FFN}_i^{(r)}(\cdot)$ denote the i-th shared expert and the i-th routed expert, respectively; K_r denotes the number of activated routed experts; $g_{i,t}$ is the gating value for the i-th expert; $s_{i,t}$ is the token-to-expert affinity; $\mathbf{e}_i$ is the centroid vector of the i-th routed expert; and $\text{Topk}(\cdot, K)$ denotes the set comprising K highest scores among the affinity scores calculated for the t-th token and all routed experts. Slightly different from DeepSeek-V2, DeepSeek-V3 uses the sigmoid function to compute the affinity scores, and applies a normalization among all selected affinity scores to produce the gating values.

Auxiliary-Loss-Free Load Balancing. For MoE models, an unbalanced expert load will lead to routing collapse (Shazeer et al., 2017) and diminish computational efficiency in scenarios with expert parallelism. Conventional solutions usually rely on the auxiliary loss (Fedus et al., 2021; Lepikhin et al., 2021) to avoid unbalanced load. However, too large an auxiliary loss will impair the model performance (Wang et al., 2024a). To achieve a better trade-off between load balance and model performance, we pioneer an auxiliary-loss-free load balancing strategy (Wang et al., 2024a) to ensure load balance. To be specific, we introduce a bias term b_i for each expert and add it to the corresponding affinity scores $s_{i,t}$ to determine the top-K routing:

$$g'_{i,t} = \begin{cases} s_{i,t}, & s_{i,t} + b_i \in \text{Topk}(\{s_{j,t} + b_j | 1 \leqslant j \leqslant N_r\}, K_r), \\ 0, & \text{otherwise.} \end{cases} \tag{16}$$

Note that the bias term is only used for routing. The gating value, which will be multiplied with the FFN output, is still derived from the original affinity score $s_{i,t}$. During training, we keep monitoring the expert load on the whole batch of each training step. At the end of each step, we will decrease the bias term by γ if its corresponding expert is overloaded, and increase it by γ if its corresponding expert is underloaded, where γ is a hyper-parameter called bias update speed. Through the dynamic adjustment, DeepSeek-V3 keeps balanced expert load during training, and achieves better performance than models that encourage load balance through pure auxiliary losses.

Complementary Sequence-Wise Auxiliary Loss. Although DeepSeek-V3 mainly relies on the auxiliary-loss-free strategy for load balance, to prevent extreme imbalance within any single sequence, we also employ a complementary sequence-wise balance loss:

$$\mathcal{L}_{\text{Bal}} = \alpha \sum_{i=1}^{N_r} f_i P_i, \tag{17}$$

$$f_i = \frac{N_r}{K_r T} \sum_{t=1}^{T} \mathbb{1}\left(s_{i,t} \in \text{Topk}(\{s_{j,t} | 1 \leqslant j \leqslant N_r\}, K_r)\right), \tag{18}$$

$$s'_{i,t} = \frac{s_{i,t}}{\sum_{j=1}^{N_r} s_{j,t}}, \tag{19}$$

$$P_i = \frac{1}{T} \sum_{t=1}^{T} s'_{i,t}, \tag{20}$$

where the balance factor α is a hyper-parameter, which will be assigned an extremely small value for DeepSeek-V3; $\mathbb{1}(\cdot)$ denotes the indicator function; and T denotes the number of tokens in a sequence. The sequence-wise balance loss encourages the expert load on each sequence to be balanced.

其中 N_s 和 N_r 分别表示共享专家和路由专家的数量；$\text{FFN}_i^{(s)}(\cdot)$ 和 $\text{FFN}_i^{(r)}(\cdot)$ 分别表示第 i 个共享专家和第 i 个路由专家；K_r 表示激活的路由专家的数量；$g_{i,t}$ 是第 i 个专家的门控值；$s_{i,t}$ 是 token 到专家的亲和度；$\mathbf{e}_i$ 是第 i 个路由专家的质心向量；$\text{Topk}(\cdot, K)$ 表示由第 t 个 token 和所有路由专家计算出的亲和度分数中前 K 个最高分数组成的集合。与 DeepSeek-V2 略有不同，DeepSeek-V3 使用 sigmoid 函数来计算亲和度分数，并对所有选定的亲和度分数进行归一化以产生门控值。

无辅助损失的负载均衡。 对于 MoE 模型，专家负载的不平衡会导致路由崩溃（Shazeer 等人，2017）并降低具有专家并行性的场景中的计算效率。传统解决方案通常依赖于辅助损失（Fedus 等人，2021；Lepikhin 等人，2021）来避免不平衡负载。然而，过大的辅助损失会损害模型性能（Wang 等人，2024a）。为了在负载均衡和模型性能之间实现更好的权衡，我们开创了一种无辅助损失的负载均衡策略（Wang 等人，2024a）来确保负载均衡。具体来说，我们为每个专家引入一个偏差项 b_i，并将其添加到相应的亲和度分数 $s_{i,t}$ 中，以确定 Top-K 路由：

$$g'_{i,t} = \begin{cases} s_{i,t}, & s_{i,t} + b_i \in \text{Topk}(\{s_{j,t} + b_j | 1 \leqslant j \leqslant N_r\}, K_r), \\ 0, & \text{otherwise.} \end{cases} \tag{16}$$

请注意，偏差项仅用于路由。将与 FFN 输出相乘的门控值仍然来自原始亲和度分数 $s_{i,t}$。在训练过程中，我们持续监控每个训练步骤中整个批次中每个专家的负载。在每个步骤结束时，如果其对应的专家过载，我们将偏差项减少 γ；如果其对应的专家负载不足，我们将偏差项增加γ，其中 γ 是一个称为偏差更新速度的超参数。通过动态调整，DeepSeek-V3 在训练过程中保持专家负载均衡，并比通过纯辅助损失鼓励负载均衡的模型取得了更好的性能。

互补序列辅助损失。 尽管 DeepSeek-V3 主要依赖于无辅助损失的策略进行负载均衡，但为了防止任何单个序列内出现极端不平衡，我们还采用了一种互补序列平衡损失：

$$\mathcal{L}_{\text{Bal}} = \alpha \sum_{i=1}^{N_r} f_i P_i, \tag{17}$$

$$f_i = \frac{N_r}{K_r T} \sum_{t=1}^{T} \mathbb{1}\left(s_{i,t} \in \text{Topk}(\{s_{j,t} | 1 \leqslant j \leqslant N_r\}, K_r)\right), \tag{18}$$

$$s'_{i,t} = \frac{s_{i,t}}{\sum_{j=1}^{N_r} s_{j,t}}, \tag{19}$$

$$P_i = \frac{1}{T} \sum_{t=1}^{T} s'_{i,t}, \tag{20}$$

其中平衡因子 α 是一个超参数，对于 DeepSeek-V3 将被分配一个极小的值；$1(\cdot)$ 表示指示函数；而 T 表示序列中的 token 数量。序列平衡损失鼓励每个序列上的专家负载均衡。

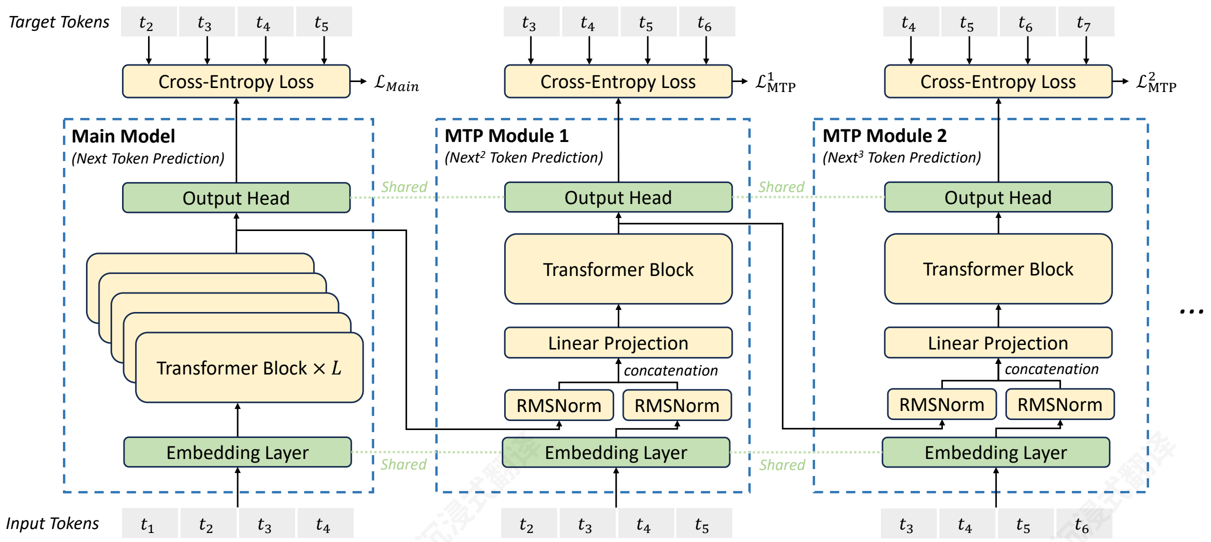


Figure 3 | Illustration of our Multi-Token Prediction (MTP) implementation. We keep the complete causal chain for the prediction of each token at each depth.

Node-Limited Routing. Like the device-limited routing used by DeepSeek-V2, DeepSeek-V3 also uses a restricted routing mechanism to limit communication costs during training. In short, we ensure that each token will be sent to at most M nodes, which are selected according to the sum of the highest $\frac{K_r}{M}$ affinity scores of the experts distributed on each node. Under this constraint, our MoE training framework can nearly achieve full computation-communication overlap.

No Token-Dropping. Due to the effective load balancing strategy, DeepSeek-V3 keeps a good load balance during its full training. Therefore, DeepSeek-V3 does not drop any tokens during training. In addition, we also implement specific deployment strategies to ensure inference load balance, so DeepSeek-V3 also does not drop tokens during inference.

2.2. Multi-Token Prediction

Inspired by Gloeckle et al. (2024), we investigate and set a Multi-Token Prediction (MTP) objective for DeepSeek-V3, which extends the prediction scope to multiple future tokens at each position. On the one hand, an MTP objective densifies the training signals and may improve data efficiency. On the other hand, MTP may enable the model to pre-plan its representations for better prediction of future tokens. Figure 3 illustrates our implementation of MTP. Different from Gloeckle et al. (2024), which parallelly predicts D additional tokens using independent output heads, we sequentially predict additional tokens and keep the complete causal chain at each prediction depth. We introduce the details of our MTP implementation in this section.

MTP Modules. To be specific, our MTP implementation uses D sequential modules to predict D additional tokens. The k-th MTP module consists of a shared embedding layer $\text{Emb}(\cdot)$, a shared output head $\text{OutHead}(\cdot)$, a Transformer block $\text{TRM}_k(\cdot)$, and a projection matrix $M_k \in \mathbb{R}^{d\times 2d}$. For the i-th input token t_i, at the k-th prediction depth, we first combine the representation of the i-th token at the $(k-1)$-th depth $\mathbf{h}_i^{k-1} \in \mathbb{R}^d$ and the embedding of the $(i+k)$-th token $Emb(t_{i+k}) \in \mathbb{R}^d$

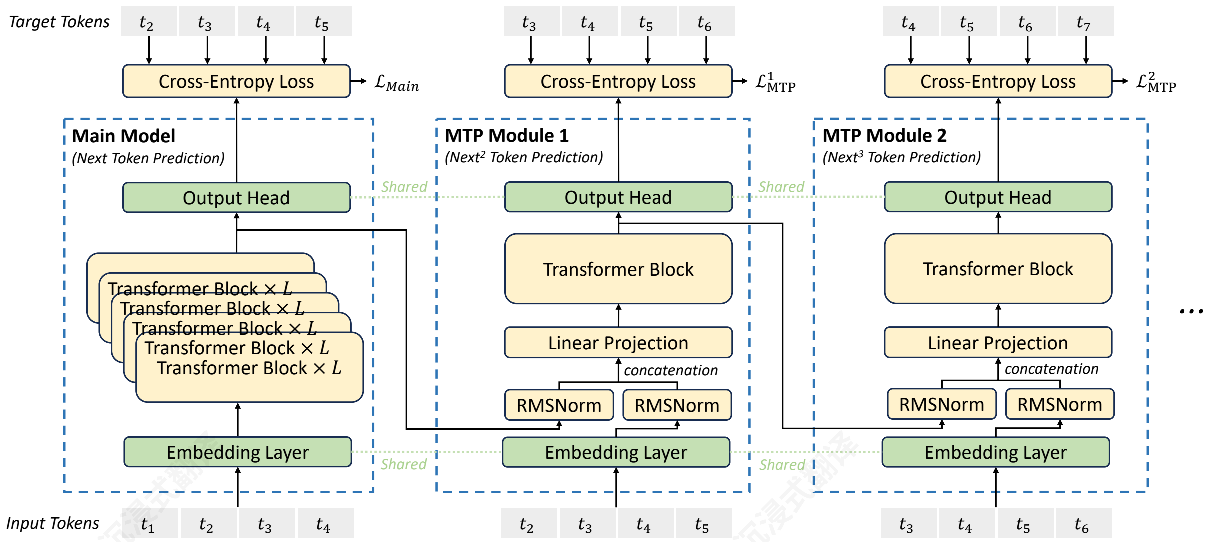


图 3 | 我们多标记预测（MTP）实现的插图。我们保留每个深度每个标记预测的完整因果链。

节点限制路由。 与 DeepSeek-V2 使用的设备限制路由类似，DeepSeek-V3 也使用一种限制路由机制来限制训练过程中的通信成本。简而言之，我们确保每个标记最多会被发送到 M 个节点，这些节点是根据每个节点上分布的专家的最高 $\frac{K_r}{M}$ 亲和度分数之和选择的。在此约束下，我们的 MoE 训练框架几乎可以实现完全的计算-通信重叠。

无标记丢弃。 由于有效的负载均衡策略，DeepSeek-V3 在整个训练过程中保持良好的负载均衡。因此，DeepSeek-V3 在训练过程中不会丢弃任何标记。此外，我们还实施了特定的部署策略来确保推理负载均衡，因此 DeepSeek-V3 在推理过程中也不会丢弃标记。

2.2. 多标记预测

受 Gloeckle 等人 (2024) 启发，我们研究了为 DeepSeek-V3 设置多标记预测 (MTP) 目标，该目标将预测范围扩展到每个位置的多个未来标记。一方面，MTP 目标使训练信号更加密集，并可能提高数据效率。另一方面，MTP 可能使模型能够预先规划其表示，以更好地预测未来标记。图 3 展示了我们对 MTP 的实现。与 Gloeckle 等人 (2024) 并行地使用独立的输出头预测 D 附加标记不同，我们顺序地预测附加标记，并在每个预测深度保持完整的因果链。我们在本节中介绍了我们的 MTP 实现细节。

MTP 模块。 具体来说，我们的 MTP 实现使用 D 顺序模块来预测 D附加标记。第 k- 个 MTP 模块由一个共享嵌入层 $\text{Emb}(\cdot)$、一个共享输出头 $\text{OutHead}(\cdot)$、一个 Transformer 块 $\text{TRM}_k(\cdot)$ 和一个投影矩阵 $M_k \in \mathbb{R}^{d\times 2d}$组成。对于第 i- 个输入标记 t_i，在第 k- 个预测深度，我们首先组合第 i- 个标记在第 $(k-1)$- 个深度 $\mathbf{h}_i^{k-1} \in \mathbb{R}^d$ 的表示和第 $(i+k)$- 个标记 $Emb(t_{i+k}) \in \mathbb{R}^d$

with the linear projection:

$$\mathbf{h}_i'^k = M_k[\text{RMSNorm}(\mathbf{h}_i^{k-1}); \text{RMSNorm}(\text{Emb}(t_{i+k}))], \tag{21}$$

where $[\cdot;\cdot]$ denotes concatenation. Especially, when $k = 1$, $\mathbf{h}_i^{k-1}$ refers to the representation given by the main model. Note that for each MTP module, its embedding layer is shared with the main model. The combined $\mathbf{h}_i'^k$ serves as the input of the Transformer block at the k-th depth to produce the output representation at the current depth $\mathbf{h}_i^k$:

$$\mathbf{h}_{1:T-k}^k = \text{TRM}_k(\mathbf{h}_{1:T-k}'^k), \tag{22}$$

where T represents the input sequence length and $_{i:j}$ denotes the slicing operation (inclusive of both the left and right boundaries). Finally, taking $\mathbf{h}_i^k$ as the input, the shared output head will compute the probability distribution for the k-th additional prediction token $P_{i+1+k}^k \in \mathbb{R}^V$, where V is the vocabulary size:

$$P_{i+k+1}^k = \text{OutHead}(\mathbf{h}_i^k). \tag{23}$$

The output head OutHead$(\cdot)$ linearly maps the representation to logits and subsequently applies the Softmax$(\cdot)$ function to compute the prediction probabilities of the k-th additional token. Also, for each MTP module, its output head is shared with the main model. Our principle of maintaining the causal chain of predictions is similar to that of EAGLE (Li et al., 2024b), but its primary objective is speculative decoding (Leviathan et al., 2023; Xia et al., 2023), whereas we utilize MTP to improve training.

MTP Training Objective. For each prediction depth, we compute a cross-entropy loss $\mathcal{L}_{\text{MTP}}^k$:

$$\mathcal{L}_{\text{MTP}}^k = \text{CrossEntropy}(P_{2+k:T+1}^k, t_{2+k:T+1}) = -\frac{1}{T}\sum_{i=2+k}^{T+1} \log P_i^k[t_i], \tag{24}$$

where T denotes the input sequence length, t_i denotes the ground-truth token at the i-th position, and $P_i^k[t_i]$ denotes the corresponding prediction probability of t_i, given by the k-th MTP module. Finally, we compute the average of the MTP losses across all depths and multiply it by a weighting factor λ to obtain the overall MTP loss $\mathcal{L}_{\text{MTP}}$, which serves as an additional training objective for DeepSeek-V3:

$$\mathcal{L}_{\text{MTP}} = \frac{\lambda}{D}\sum_{k=1}^{D} \mathcal{L}_{\text{MTP}}^k. \tag{25}$$

MTP in Inference. Our MTP strategy mainly aims to improve the performance of the main model, so during inference, we can directly discard the MTP modules and the main model can function independently and normally. Additionally, we can also repurpose these MTP modules for speculative decoding to further improve the generation latency.

3. Infrastructures

3.1. Compute Clusters

DeepSeek-V3 is trained on a cluster equipped with 2048 NVIDIA H800 GPUs. Each node in the H800 cluster contains 8 GPUs connected by NVLink and NVSwitch within nodes. Across different nodes, InfiniBand (IB) interconnects are utilized to facilitate communications.

与线性投影：

$$\mathbf{h}_i'^k = M_k[\text{RMSNorm}(\mathbf{h}_i^{k-1}); \text{RMSNorm}(\text{Emb}(t_{i+k}))], \tag{21}$$

其中 $[\cdot;\cdot]$ 表示拼接。特别是，当 $k = 1$ 时，$\mathbf{h}_i^{k-1}$ 指的是由主模型给出的表示。请注意，对于每个 MTP 模块，其嵌入层与主模型共享。组合的 $\mathbf{h}_i'^k$ 作为第 k 层 Transformer 块的输入，以产生当前深度的输出表示 $\mathbf{h}_i^k$：

$$\mathbf{h}_{1:T-k}^k = \text{TRM}_k(\mathbf{h}_{1:T-k}'^k), \tag{22}$$

其中 T 表示输入序列长度，$_{i:j}$ 表示切片操作（包括左右边界）。最后，以 $\mathbf{h}_i^k$ 作为输入，共享的输出头将计算第 k 个附加预测标记 $P_{i+1+k}^k \in \mathbb{R}^V$ 的概率分布，其中 V 是词汇量大小：

$$P_{i+k+1}^k = \text{OutHead}(\mathbf{h}_i^k). \tag{23}$$

输出头 OutHead$(\cdot)$ 将表示线性映射到 logits，然后应用 Softmax$(\cdot)$ 函数来计算第 k 个附加标记的预测概率。此外，对于每个 MTP 模块，其输出头与主模型共享。我们保持预测因果链的原则与 EAGLE (Li et al., 2024b) 类似，但其主要目标是推测解码 (Leviathan et al., 2023; Xia et al., 2023)，而我们利用 MTP 来改进训练。

MTP 训练目标。 对于每个预测深度，我们计算交叉熵损失 L$_k$ MTP：

$$\mathcal{L}_{\text{MTP}}^k = \text{CrossEntropy}(P_{2+k:T+1}^k, t_{2+k:T+1}) = -\frac{1}{T}\sum_{i=2+k}^{T+1} \log P_i^k[t_i], \tag{24}$$

其中 T 表示输入序列长度，t_i 表示第 i-个位置的ground-truth token， $P_i^k[t_i]$ 表示 t_i 对应的预测概率，由第 k-个MTP模块给出。最后，我们计算所有深度的MTP损失的均值，并乘以加权因子 λ 得到整体MTP损失 LMTP，它作为DeepSeek-V3的附加训练目标：

$$\mathcal{L}_{\text{MTP}} = \frac{\lambda}{D}\sum_{k=1}^{D} \mathcal{L}_{\text{MTP}}^k. \tag{25}$$

推理中的MTP。 我们的MTP策略主要旨在提高主模型的性能，因此在推理时，我们可以直接丢弃MTP模块，主模型可以独立且正常运行。此外，我们还可以将这些MTP模块用于推测解码，以进一步提高生成延迟。

3. 基础设施

3.1. 计算集群

DeepSeek-V3 在一个配备 2048 块 NVIDIA H800 GPU 的集群上进行训练。H800 集群中的每个节点包含 8 块通过节点内的 NVLink 和 NVSwitch 连接的 GPU。在不同的节点之间，使用 InfiniBand (IB) 互连来促进通信。

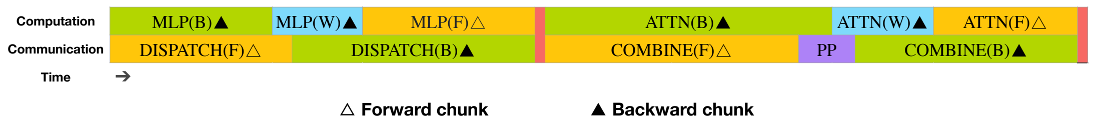


Figure 4 | Overlapping strategy for a pair of individual forward and backward chunks (the boundaries of the transformer blocks are not aligned). Orange denotes forward, green denotes "backward for input", blue denotes "backward for weights", purple denotes PP communication, and red denotes barriers. Both all-to-all and PP communication can be fully hidden.

3.2. Training Framework

The training of DeepSeek-V3 is supported by the HAI-LLM framework, an efficient and lightweight training framework crafted by our engineers from the ground up. On the whole, DeepSeek-V3 applies 16-way Pipeline Parallelism (PP) (Qi et al., 2023a), 64-way Expert Parallelism (EP) (Lepikhin et al., 2021) spanning 8 nodes, and ZeRO-1 Data Parallelism (DP) (Rajbhandari et al., 2020).

In order to facilitate efficient training of DeepSeek-V3, we implement meticulous engineering optimizations. Firstly, we design the DualPipe algorithm for efficient pipeline parallelism. Compared with existing PP methods, DualPipe has fewer pipeline bubbles. More importantly, it overlaps the computation and communication phases across forward and backward processes, thereby addressing the challenge of heavy communication overhead introduced by cross-node expert parallelism. Secondly, we develop efficient cross-node all-to-all communication kernels to fully utilize IB and NVLink bandwidths and conserve Streaming Multiprocessors (SMs) dedicated to communication. Finally, we meticulously optimize the memory footprint during training, thereby enabling us to train DeepSeek-V3 without using costly Tensor Parallelism (TP).

3.2.1. DualPipe and Computation-Communication Overlap

For DeepSeek-V3, the communication overhead introduced by cross-node expert parallelism results in an inefficient computation-to-communication ratio of approximately 1:1. To tackle this challenge, we design an innovative pipeline parallelism algorithm called DualPipe, which not only accelerates model training by effectively overlapping forward and backward computation-communication phases, but also reduces the pipeline bubbles.

The key idea of DualPipe is to overlap the computation and communication within a pair of individual forward and backward chunks. To be specific, we divide each chunk into four components: `attention`, `all-to-all dispatch`, `MLP`, and `all-to-all combine`. Specially, for a backward chunk, both `attention` and `MLP` are further split into two parts, `backward for input` and `backward for weights`, like in ZeroBubble (Qi et al., 2023b). In addition, we have a `PP communication` component. As illustrated in Figure 4, for a pair of forward and backward chunks, we rearrange these components and manually adjust the ratio of GPU SMs dedicated to communication versus computation. In this overlapping strategy, we can ensure that both all-to-all and PP communication can be fully hidden during execution. Given the efficient overlapping strategy, the full DualPipe scheduling is illustrated in Figure 5. It employs a bidirectional pipeline scheduling, which feeds micro-batches from both ends of the pipeline simultaneously and a significant portion of communications can be fully overlapped. This overlap also ensures that, as the model further scales up, as long as we maintain a constant computation-to-communication ratio, we can still employ fine-grained experts across nodes while achieving a near-zero all-to-all communication overhead.

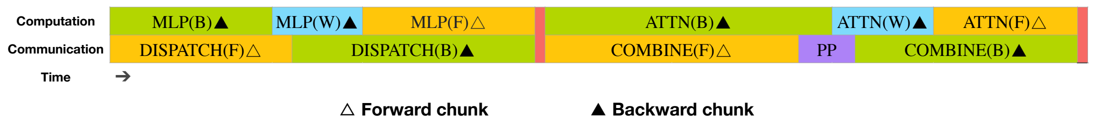


图 4 | 一对单独的前向和后向块的覆盖策略（Transformer块的边界未对齐）。橙色表示前向，绿色表示“输入的后向”，蓝色表示“权重的后向”，紫色表示PP通信，红色表示障碍。所有到所有和PP通信都可以完全隐藏。

3.2. 训练框架

DeepSeek-V3的训练由HAI-LLM框架支持，该框架是由我们的工程师从零开始精心设计的，高效且轻量级。总的来说，DeepSeek-V3应用了16路流水线并行(PP)(Qiet al., 2023a)，64路专家并行(EP) (Lepikhin et al., 2021)，跨越8个节点，以及ZeRO-1数据并行(DP) (Rajbhandari et al., 2020)。

为了促进DeepSeek-V3的高效训练，我们实施了细致的工程优化。首先，我们设计了用于高效流水线并行的DualPipe算法。与现有的PP方法相比，DualPipe的流水线气泡更少。更重要的是，它在前向和后向过程中重叠计算和通信阶段，从而解决了跨节点专家并行引入的通信开销过大的挑战。其次，我们开发了高效跨节点的所有到所有通信内核，以充分利用IB和NVLink带宽，并节省用于通信的流式多处理器(SMs)。最后，我们细致地优化了训练期间的内存占用，从而使我们能够在不使用昂贵的张量并行(TP)的情况下训练DeepSeek-V3。

*3.2.1. DualPipe*和计算-通信重叠

对于DeepSeek-V3，跨节点专家并行性引入的通信开销导致计算-通信比率约为1:1，效率低下。为了应对这一挑战，我们设计了一种名为DualPipe的创新流水线并行算法，该算法不仅通过有效重叠前向和后向计算-通信阶段加速模型训练，还减少了流水线气泡。

DualPipe的关键思想是在一对单独的前向和后向块内重叠计算和通信。具体来说，我们将每个块分为四个组件： `attention`, `all-to-all dispatch`, MLP和 `all-to-all combine`。特别地，对于后向块， `attention` 和 MLP 都被进一步分为两部分， `backward forinput` 和 `backward for weights`，类似于ZeroBubble（Qi等人，2023b）。此外，我们还有一个 `PP communication` 组件。如图4所示，对于一对前向和后向块，我们重新排列这些组件，并手动调整用于通信和计算的GPU SMs的比率。在这种重叠策略中，我们可以确保在执行期间所有对全和PP通信都能完全隐藏。鉴于高效的重叠策略，完整的DualPipe调度如图5所示。它采用双向流水线调度，同时从流水线的两端输入微批次，并且大部分通信可以完全重叠。这种重叠还确保，随着模型进一步扩展，只要我们保持计算-通信比率恒定，我们仍然可以在节点上采用细粒度专家，同时实现近零的所有对全通信开销。

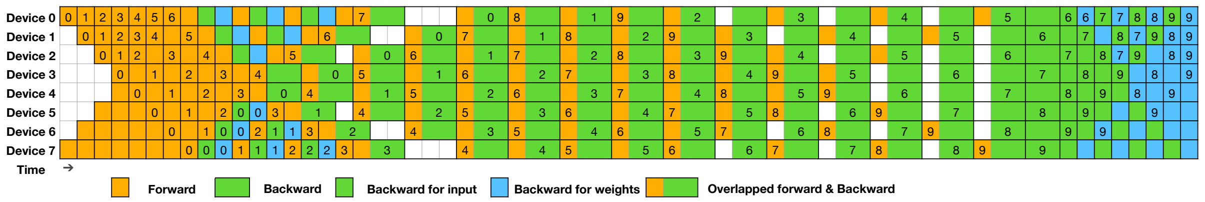


Figure 5 | Example DualPipe scheduling for 8 PP ranks and 20 micro-batches in two directions. The micro-batches in the reverse direction are symmetric to those in the forward direction, so we omit their batch ID for illustration simplicity. Two cells enclosed by a shared black border have mutually overlapped computation and communication.

Method	Bubble	Parameter	Activation
1F1B	$(PP-1)(F+B)$	1×	PP
ZB1P	$(PP-1)(F+B-2W)$	1×	PP
DualPipe (Ours)	$(\frac{PP}{2}-1)(F\&B+B-3W)$	2×	$PP+1$

Table 2 | Comparison of pipeline bubbles and memory usage across different pipeline parallel methods. F denotes the execution time of a forward chunk, B denotes the execution time of a full backward chunk, W denotes the execution time of a "backward for weights" chunk, and $F\&B$ denotes the execution time of two mutually overlapped forward and backward chunks.

In addition, even in more general scenarios without a heavy communication burden, DualPipe still exhibits efficiency advantages. In Table 2, we summarize the pipeline bubbles and memory usage across different PP methods. As shown in the table, compared with ZB1P (Qi et al., 2023b) and 1F1B (Harlap et al., 2018), DualPipe significantly reduces the pipeline bubbles while only increasing the peak activation memory by $\frac{1}{PP}$ times. Although DualPipe requires keeping two copies of the model parameters, this does not significantly increase the memory consumption since we use a large EP size during training. Compared with Chimera (Li and Hoefler, 2021), DualPipe only requires that the pipeline stages and micro-batches be divisible by 2, without requiring micro-batches to be divisible by pipeline stages. In addition, for DualPipe, neither the bubbles nor activation memory will increase as the number of micro-batches grows.

3.2.2. Efficient Implementation of Cross-Node All-to-All Communication

In order to ensure sufficient computational performance for DualPipe, we customize efficient cross-node all-to-all communication kernels (including dispatching and combining) to conserve the number of SMs dedicated to communication. The implementation of the kernels is co-designed with the MoE gating algorithm and the network topology of our cluster. To be specific, in our cluster, cross-node GPUs are fully interconnected with IB, and intra-node communications are handled via NVLink. NVLink offers a bandwidth of 160 GB/s, roughly 3.2 times that of IB (50 GB/s). To effectively leverage the different bandwidths of IB and NVLink, we limit each token to be dispatched to at most 4 nodes, thereby reducing IB traffic. For each token, when its routing decision is made, it will first be transmitted via IB to the GPUs with the same in-node index on its target nodes. Once it reaches the target nodes, we will endeavor to ensure that it is instantaneously forwarded via NVLink to specific GPUs that host their target experts, without being blocked by subsequently arriving tokens. In this way, communications via IB and NVLink are fully overlapped, and each token can efficiently select an average of 3.2 experts per node without incurring additional overhead from NVLink. This implies that, although DeepSeek-V3

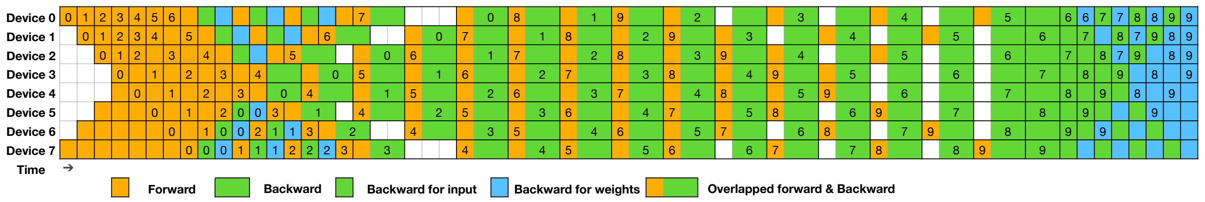


Figure 5 | Example DualPipe scheduling for 8 PP ranks and 20 micro-batches in two directions. The micro-batches in the reverse direction are symmetric to those in the forward direction, so we omit their batch ID for illustration simplicity. Two cells enclosed by a shared black border have mutually overlapped computation and communication.

方法	气泡	参数	激活
1F1B	$(PP-1)\ (F+B)$	1×	PP
ZB1P	$(PP-1)\ (F+B-2W)$	1×	PP
DualPipe (我们的)	$(\frac{PP}{2}-1)\ (F\&B+B-3W)$	2×	$PP+1$

表 2 | 不同流水线并行方法的流水线气泡和内存使用比较。 F 表示一个前向块的执行时间， B 表示一个完整反向块的执行时间， W 表示一个 “ 反向用于权重 ” 块的执行时间， $F\&B$ 表示两个相互重叠的前向和反向块的执行时间。

此外，即使在通信负担较轻的更一般场景中，DualPipe 仍然表现出效率优势。在表 2 中，我们总结了不同 PP 方法之间的流水线气泡和内存使用情况。如表所示，与 ZB1P（Qi 等人，2023b）和 1F1B（Harlap 等人，2018）相比，DualPipe 显着减少了流水线气泡，同时仅将峰值激活内存增加了 $\frac{1}{PP}$ 倍。尽管 DualPipe 需要保留模型参数的两个副本，但由于我们在训练期间使用较大的 EP 大小，这并不会显着增加内存消耗。与 Chimera（Li 和 Hoefler，2021）相比，DualPipe 仅要求流水线阶段和微批次是 2 的倍数，而无需微批次是流水线阶段的倍数。此外，对于 DualPipe，随着微批次数量的增加，气泡和激活内存都不会增加。

3.2.2. 高效实现跨节点全对全通信

为了确保 DualPipe 具备足够的计算性能，我们定制了高效的跨节点全对全通信内核（包括调度和合并），以节省用于通信的 SM 数量。内核的实现与 MoE 门控算法和集群的网络拓扑协同设计。具体来说，在我们的集群中，跨节点 GPU 通过 IB 完全互连，而节点内通信则通过 NVLink 处理。NVLink 提供 160 GB/s 的带宽，大约是 IB（50 GB/s）的 3.2 倍。为了有效利用 IB 和 NVLink 的不同带宽，我们将每个 token 限制最多调度到 4 个节点，从而减少 IB 流量。对于每个 token，当其路由决策做出时，它将首先通过 IB 传输到目标节点上具有相同节点内索引的 GPU。一旦到达目标节点，我们将努力确保它通过 NVLink 立即转发到承载其目标专家的特定 GPU，而不会被随后到达的 token 阻塞。通过这种方式，IB 和 NVLink 的通信完全重叠，每个 token 可以高效地选择每个节点平均 3.2 个专家，而不会产生额外的 NVLink 开销。这意味着，尽管 DeepSeek-V3 selects only 8 routed experts in practice, it can scale up this number to a maximum of 13 experts (4 nodes × 3.2 experts/node) while preserving the same communication cost. Overall, under such a communication strategy, only 20 SMs are sufficient to fully utilize the bandwidths of IB and NVLink.

In detail, we employ the warp specialization technique (Bauer et al., 2014) and partition 20 SMs into 10 communication channels. During the dispatching process, (1) IB sending, (2) IB-to-NVLink forwarding, and (3) NVLink receiving are handled by respective warps. The number of warps allocated to each communication task is dynamically adjusted according to the actual workload across all SMs. Similarly, during the combining process, (1) NVLink sending, (2) NVLink-to-IB forwarding and accumulation, and (3) IB receiving and accumulation are also handled by dynamically adjusted warps. In addition, both dispatching and combining kernels overlap with the computation stream, so we also consider their impact on other SM computation kernels. Specifically, we employ customized PTX (Parallel Thread Execution) instructions and auto-tune the communication chunk size, which significantly reduces the use of the L2 cache and the interference to other SMs.

3.2.3. Extremely Memory Saving with Minimal Overhead

In order to reduce the memory footprint during training, we employ the following techniques.

Recomputation of RMSNorm and MLA Up-Projection. We recompute all RMSNorm operations and MLA up-projections during back-propagation, thereby eliminating the need to persistently store their output activations. With a minor overhead, this strategy significantly reduces memory requirements for storing activations.

Exponential Moving Average in CPU. During training, we preserve the Exponential Moving Average (EMA) of the model parameters for early estimation of the model performance after learning rate decay. The EMA parameters are stored in CPU memory and are updated asynchronously after each training step. This method allows us to maintain EMA parameters without incurring additional memory or time overhead.

Shared Embedding and Output Head for Multi-Token Prediction. With the DualPipe strategy, we deploy the shallowest layers (including the embedding layer) and deepest layers (including the output head) of the model on the same PP rank. This arrangement enables the physical sharing of parameters and gradients, of the shared embedding and output head, between the MTP module and the main model. This physical sharing mechanism further enhances our memory efficiency.

3.3. FP8 Training

Inspired by recent advances in low-precision training (Dettmers et al., 2022; Noune et al., 2022; Peng et al., 2023b), we propose a fine-grained mixed precision framework utilizing the FP8 data format for training DeepSeek-V3. While low-precision training holds great promise, it is often limited by the presence of outliers in activations, weights, and gradients (Fishman et al., 2024; He et al.; Sun et al., 2024). Although significant progress has been made in inference quantization (Frantar et al., 2022; Xiao et al., 2023), there are relatively few studies demonstrating successful application of low-precision techniques in large-scale language model

在实践中选择仅8个路由专家，它可以扩展这个数字到最多13个专家（4个节点 ×3.2个专家/节点），同时保持相同的通信成本。总体而言，在这种通信策略下，仅需要20个SM即可充分利用IB和NVLink的带宽。

详细来说，我们采用warp专业化技术（Bauer等人，2014年）并将20个SM划分为10个通信通道。在调度过程中，（1）IB发送，（2）IB到NVLink转发和（3）NVLink接收由相应的warp处理。分配给每个通信任务的warp数量根据所有SM的实际工作负载动态调整。类似地，在合并过程中，（1）NVLink发送，（2）NVLink到IB转发和累积以及（3）IB接收和累积也由动态调整的warp处理。此外，调度和合并内核与计算流重叠，因此我们也考虑它们对其他SM计算内核的影响。具体来说，我们采用定制的PTX（并行线程执行）指令并自动调整通信块大小，这显著减少了L2缓存的使用和对其他SM的干扰。

3.2.3. 极致内存节省与极小开销

为了在训练过程中减少内存占用，我们采用了以下技术。

重新计算 RMSNorm 和 MLA 上投影。 我们在反向传播过程中重新计算所有 RMSNorm 操作和 MLA 上投影，从而无需持久存储它们的输出激活。以轻微的开销，这种策略显著降低了存储激活所需的内存需求。

CPU中的指数移动平均。在训练过程中，我们保存模型的指数移动平均（EMA）参数，以便在学习率衰减后对模型性能进行早期估计。EMA参数存储在CPU内存中，并在每次训练步骤后异步更新。这种方法使我们能够在不产生额外内存或时间开销的情况下保持EMA参数。

共享嵌入和输出头用于多标记预测。 通过 DualPipe 策略，我们将模型的浅层（包括嵌入层）和深层（包括输出头）部署在同一个 PP 排名上。这种安排使得共享嵌入和输出头的参数和梯度可以在 MTP 模块和主模型之间进行物理共享，这种物理共享机制进一步提高了我们的内存效率。

3.3. FP8 训练

受近期低精度训练的进展启发（Dettmers 等人，2022；Noune 等人，2022；Peng 等人，2023b），我们提出了一种利用 FP8 数据格式进行 DeepSeek-V3 训练的细粒度混合精度框架。虽然低精度训练具有巨大潜力，但它在激活值、权重和梯度中存在异常值时常常受到限制（ Fishman 等人，2024；He 等人；Sun 等人，2024）。尽管在推理量化方面取得了显著进展（Frantar 等人，2022；Xiao 等人，2023），但相对较少的研究展示了低精度技术在大型语言模型中的成功应用

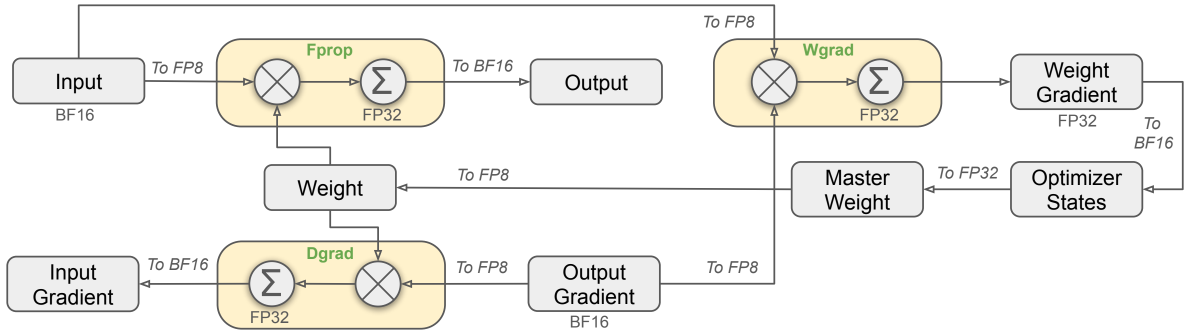


Figure 6 | The overall mixed precision framework with FP8 data format. For clarification, only the `Linear` operator is illustrated.

pre-training (Fishman et al., 2024). To address this challenge and effectively extend the dynamic range of the FP8 format, we introduce a fine-grained quantization strategy: tile-wise grouping with $1 \times N_c$ elements or block-wise grouping with $N_c \times N_c$ elements. The associated dequantization overhead is largely mitigated under our increased-precision accumulation process, a critical aspect for achieving accurate FP8 General Matrix Multiplication (GEMM). Moreover, to further reduce memory and communication overhead in MoE training, we cache and dispatch activations in FP8, while storing low-precision optimizer states in BF16. We validate the proposed FP8 mixed precision framework on two model scales similar to DeepSeek-V2-Lite and DeepSeek-V2, training for approximately 1 trillion tokens (see more details in Appendix B.1). Notably, compared with the BF16 baseline, the relative loss error of our FP8-training model remains consistently below 0.25%, a level well within the acceptable range of training randomness.

3.3.1. Mixed Precision Framework

Building upon widely adopted techniques in low-precision training (Kalamkar et al., 2019; Narang et al., 2017), we propose a mixed precision framework for FP8 training. In this framework, most compute-density operations are conducted in FP8, while a few key operations are strategically maintained in their original data formats to balance training efficiency and numerical stability. The overall framework is illustrated in Figure 6.

Firstly, in order to accelerate model training, the majority of core computation kernels, i.e., GEMM operations, are implemented in FP8 precision. These GEMM operations accept FP8 tensors as inputs and produce outputs in BF16 or FP32. As depicted in Figure 6, all three GEMMs associated with the `Linear` operator, namely `Fprop` (forward pass), `Dgrad` (activation backward pass), and `Wgrad` (weight backward pass), are executed in FP8. This design theoretically doubles the computational speed compared with the original BF16 method. Additionally, the FP8 `Wgrad` GEMM allows activations to be stored in FP8 for use in the backward pass. This significantly reduces memory consumption.

Despite the efficiency advantage of the FP8 format, certain operators still require a higher precision due to their sensitivity to low-precision computations. Besides, some low-cost operators can also utilize a higher precision with a negligible overhead to the overall training cost. For this reason, after careful investigations, we maintain the original precision (e.g., BF16 or FP32) for the following components: the embedding module, the output head, MoE gating modules, normalization operators, and attention operators. These targeted retentions of high precision ensure stable training dynamics for DeepSeek-V3. To further guarantee numerical stability, we store the master weights, weight gradients, and optimizer states in higher precision. While

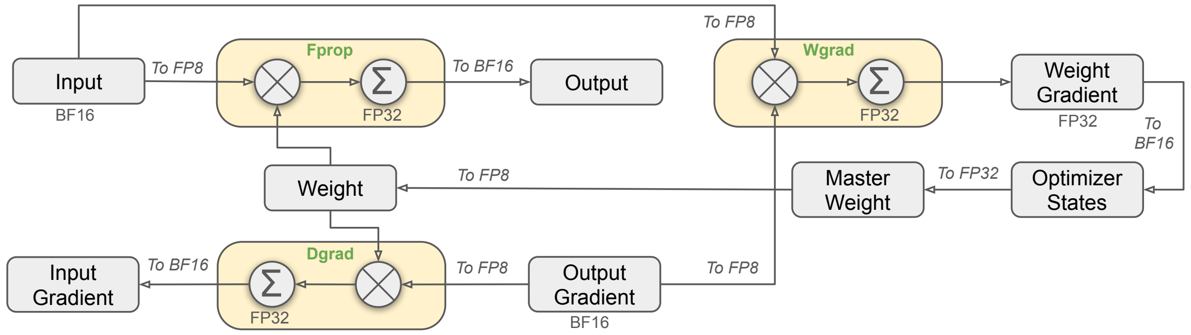


图 6 | FP8数据格式的整体混合精度框架。为清晰起见，仅绘制了 `Linear`算子。

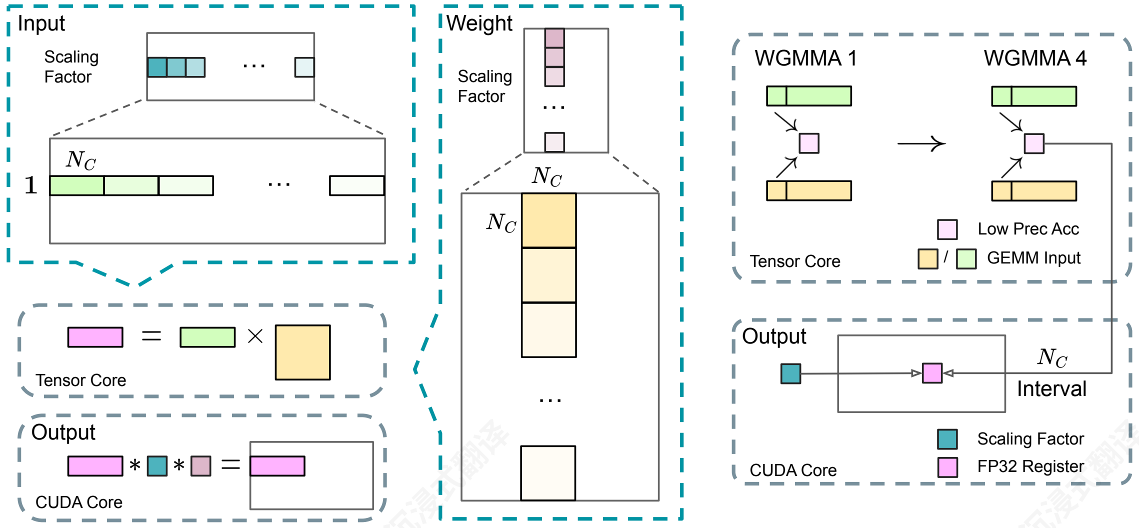
预训练（Fishman等人，2024年）。为了应对这一挑战并有效扩展FP8格式的动态范围，我们引入了一种细粒度量化策略：以 $1 \times N_c$ 个元素为单位的tiling分组或以 $N_c \times N_c$ 个元素为单位的block分组。在增加精度的累积过程中，相关的解量化开销得到了显著缓解，这是实现精确FP8通用矩阵乘法（GEMM）的关键方面。此外，为了进一步减少MoE训练中的内存和通信开销，我们在FP8中缓存和调度激活值，同时将低精度优化器状态存储在BF16中。我们在两个与DeepSeek-V2-Lite和DeepSeek-V2相似的模型规模上验证了所提出的FP8混合精度框架，训练了大约1万亿个token（更多细节见附录B.1）。值得注意的是，与BF16基线相比，我们FP8训练模型的相对损失误差始终保持在0.25%以下，这一水平在训练随机性的可接受范围内。

3.3.1. 混合精度框架

基于在低精度训练中广泛采用的技巧（Kalamkar 等人，2019；Narang 等人，2017），我们提出了一种用于FP8训练的混合精度框架。在该框架中，大多数计算密集型操作在FP8中执行，而少数关键操作则策略性地保持在其原始数据格式中，以平衡训练效率和数值稳定性。整体框架如图6所示。

首先，为了加速模型训练，大多数核心计算内核，即GEMM操作，均以FP8精度实现。这些GEMM操作接受FP8张量作为输入，并产生BF16或FP32格式的输出。如图6所示，与 `Linear` 操作相关的三个GEMM，即 `Fprop` （前向传递）、 `Dgrad` （激活反向传递）和 `Wgrad` （权重反向传递），均以FP8执行。理论上，这种设计比原始BF16方法将计算速度翻倍。此外，FP8 `Wgrad`GEMM允许激活以FP8格式存储，用于反向传递。这显著减少了内存消耗。

尽管FP8格式具有效率优势，但由于某些算子对低精度计算敏感，仍然需要更高的精度。此外，一些低成本算子也可以利用更高的精度，而整体训练成本的开销可以忽略不计。因此，经过仔细研究，我们为以下组件保留原始精度（例如，BF16或FP32）：嵌入模块、输出头、MoE门控模块、归一化算子和注意力算子。这些有针对性的高精度保留确保了DeepSeek-V3的稳定训练动态。为了进一步保证数值稳定性，我们将主权重、权重梯度和优化器状态存储在更高的精度中。而



(a) Fine-grained quantization

(b) Increasing accumulation precision

Figure 7 | (a) We propose a fine-grained quantization method to mitigate quantization errors caused by feature outliers; for illustration simplicity, only Fprop is illustrated. (b) In conjunction with our quantization strategy, we improve the FP8 GEMM precision by promoting to CUDA Cores at an interval of N_C = 128 elements MMA for the high-precision accumulation.

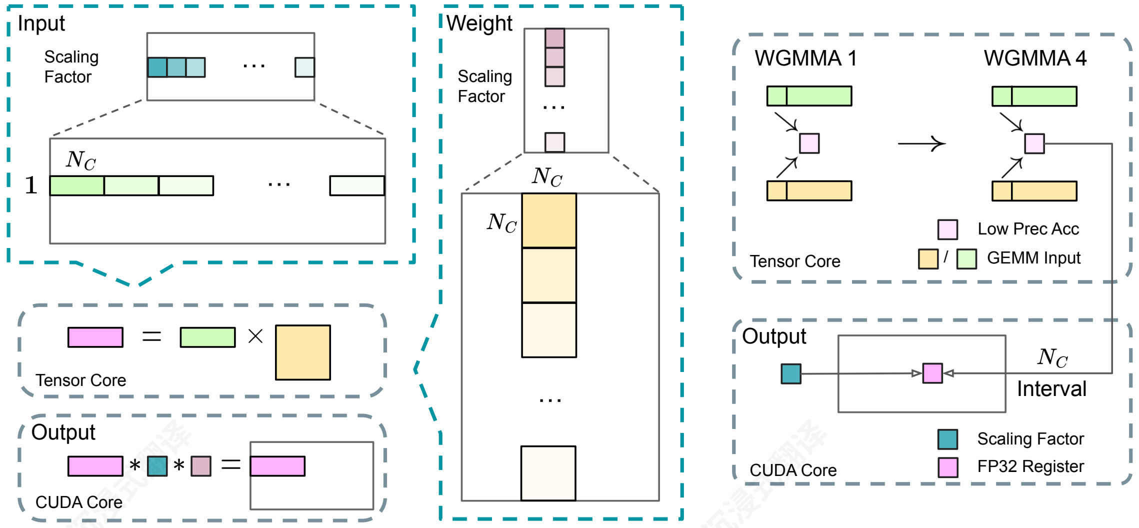
these high-precision components incur some memory overheads, their impact can be minimized through efficient sharding across multiple DP ranks in our distributed training system.

3.3.2. Improved Precision from Quantization and Multiplication

Based on our mixed precision FP8 framework, we introduce several strategies to enhance low-precision training accuracy, focusing on both the quantization method and the multiplication process.

Fine-Grained Quantization. In low-precision training frameworks, overflows and underflows are common challenges due to the limited dynamic range of the FP8 format, which is constrained by its reduced exponent bits. As a standard practice, the input distribution is aligned to the representable range of the FP8 format by scaling the maximum absolute value of the input tensor to the maximum representable value of FP8 (Narang et al., 2017). This method makes low-precision training highly sensitive to activation outliers, which can heavily degrade quantization accuracy. To solve this, we propose a fine-grained quantization method that applies scaling at a more granular level. As illustrated in Figure 7 (a), (1) for activations, we group and scale elements on a 1x128 tile basis (i.e., per token per 128 channels); and (2) for weights, we group and scale elements on a 128x128 block basis (i.e., per 128 input channels per 128 output channels). This approach ensures that the quantization process can better accommodate outliers by adapting the scale according to smaller groups of elements. In Appendix B.2, we further discuss the training instability when we group and scale activations on a block basis in the same way as weights quantization.

One key modification in our method is the introduction of per-group scaling factors along the inner dimension of GEMM operations. This functionality is not directly supported in the standard FP8 GEMM. However, combined with our precise FP32 accumulation strategy, it can



(a) Fine-grained quantization

(b) 提高累积精度

图 7 | (a) 我们提出一种细粒度量化方法来减轻由特征异常值引起的量化误差；为简单起见，仅说明了 Fprop。 (b) 在我们的量化策略的配合下，我们通过将 CUDA 核心以 N_C = 128 个元素间隔的 MMA 进行高精度累积，提高了 FP8 GEMM 精度。

这些高精度组件会带来一些内存开销，但它们的影响可以通过在我们的分布式训练系统中在多个 DP 排名之间进行高效的分片来最小化。

3.3.2. 量化与乘法带来的精度提升

基于我们的混合精度 FP8 框架，我们引入了几种策略来提高低精度训练的精度，重点关注量化方法和乘法过程。

细粒度量化。 在低精度训练框架中，由于 FP8 格式的动态范围有限（受指数位减少的限制），溢出和下溢是常见的挑战。作为一种标准做法，通过将输入张量的最大绝对值缩放到 FP8 的最大表示值（Narang 等人，2017 年），将输入分布对齐到 FP8 的表示范围。这种方法使低精度训练对激活异常值非常敏感，这会严重降低量化精度。为了解决这个问题，我们提出了一种细粒度量化方法，该方法在更细粒度的级别应用缩放。如图 7 (a)，(1) 对于激活值，我们在 1x128 的基础上对元素进行分组和缩放（即每个标记每 128 个通道）；(2) 对于权重，我们在 128x128 的基础上对元素进行分组和缩放（即每个 128 个输入通道每 128 个输出通道）。这种方法确保量化过程可以通过根据较小的元素组调整缩放来更好地适应异常值。在附录 B.2 中，我们进一步讨论了当我们以与权重量化相同的方式在块级别对激活值进行分组和缩放时训练的不稳定性。

我们方法中的一个关键修改是在GEMM操作的内部维度上引入了按组的缩放因子。这项功能在标准的FP8 GEMM中并不直接支持。然而，结合我们精确的FP32累加策略，它可以

be efficiently implemented.

Notably, our fine-grained quantization strategy is highly consistent with the idea of microscaling formats (Rouhani et al., 2023b), while the Tensor Cores of NVIDIA next-generation GPUs (Blackwell series) have announced the support for microscaling formats with smaller quantization granularity (NVIDIA, 2024a). We hope our design can serve as a reference for future work to keep pace with the latest GPU architectures.

Increasing Accumulation Precision. Low-precision GEMM operations often suffer from underflow issues, and their accuracy largely depends on high-precision accumulation, which is commonly performed in an FP32 precision (Kalamkar et al., 2019; Narang et al., 2017). However, we observe that the accumulation precision of FP8 GEMM on NVIDIA H800 GPUs is limited to retaining around 14 bits, which is significantly lower than FP32 accumulation precision. This problem will become more pronounced when the inner dimension `K` is large (Wortsman et al., 2023), a typical scenario in large-scale model training where the batch size and model width are increased. Taking GEMM operations of two random matrices with `K` = 4096 for example, in our preliminary test, the limited accumulation precision in Tensor Cores results in a maximum relative error of nearly 2%. Despite these problems, the limited accumulation precision is still the default option in a few FP8 frameworks (NVIDIA, 2024b), severely constraining the training accuracy.

In order to address this issue, we adopt the strategy of promotion to CUDA Cores for higher precision (Thakkar et al., 2023). The process is illustrated in Figure 7 (b). To be specific, during MMA (Matrix Multiply-Accumulate) execution on Tensor Cores, intermediate results are accumulated using the limited bit width. Once an interval of N_C is reached, these partial results will be copied to FP32 registers on CUDA Cores, where full-precision FP32 accumulation is performed. As mentioned before, our fine-grained quantization applies per-group scaling factors along the inner dimension `K`. These scaling factors can be efficiently multiplied on the CUDA Cores as the dequantization process with minimal additional computational cost.

It is worth noting that this modification reduces the WGMMA (Warpgroup-level Matrix Multiply-Accumulate) instruction issue rate for a single warpgroup. However, on the H800 architecture, it is typical for two WGMMA to persist concurrently: while one warpgroup performs the promotion operation, the other is able to execute the MMA operation. This design enables overlapping of the two operations, maintaining high utilization of Tensor Cores. Based on our experiments, setting N_C = 128 elements, equivalent to 4 WGMMAs, represents the minimal accumulation interval that can significantly improve precision without introducing substantial overhead.

Mantissa over Exponents. In contrast to the hybrid FP8 format adopted by prior work (NVIDIA, 2024b; Peng et al., 2023b; Sun et al., 2019b), which uses `E4M3` (4-bit exponent and 3-bit mantissa) in `Fprop` and `E5M2` (5-bit exponent and 2-bit mantissa) in `Dgrad` and `Wgrad`, we adopt the `E4M3` format on all tensors for higher precision. We attribute the feasibility of this approach to our fine-grained quantization strategy, i.e., tile and block-wise scaling. By operating on smaller element groups, our methodology effectively shares exponent bits among these grouped elements, mitigating the impact of the limited dynamic range.

Online Quantization. Delayed quantization is employed in tensor-wise quantization frameworks (NVIDIA, 2024b; Peng et al., 2023b), which maintains a history of the maximum absolute

可以被高效地实现。

值得注意的是，我们的细粒度量化策略与微缩格式（Rouhani et al., 2023b）的理念高度一致，而NVIDIA下一代GPU（Blackwell系列）的Tensor Cores已宣布支持具有较小量化粒度的微缩格式（NVIDIA, 2024a）。我们希望我们的设计能为未来工作提供参考，以跟上最新的GPU架构。

提高累积精度。 低精度GEMM操作常常出现下溢问题，其精度很大程度上取决于高精度累积，这通常以FP32精度进行（Kalamkar et al., 2019; Narang et al., 2017）。然而，我们观察到NVIDIA H800 GPU上FP8 GEMM的累积精度仅能保留约14位，这显著低于FP32累积精度。当内维度 `K` 较大时（Wortsman et al., 2023），这个问题会更加明显，这在大型模型训练中很常见，其中批大小和模型宽度都会增加。以两个随机矩阵的GEMM操作为例，其中 `K` = 4096，在我们的初步测试中，Tensor Cores的有限累积精度导致最大相对误差接近2%。尽管存在这些问题，但在少数FP8框架（NVIDIA, 2024b）中，有限的累积精度仍然是默认选项，严重限制了训练精度。

为了解决这个问题，我们采用提升至CUDA核心以提高精度的策略（Thakkar等人，2023年）。该过程如图7（b）所示。具体来说，在Tensor核心上执行MMA（矩阵乘累加）时，中间结果使用有限的位宽进行累加。一旦达到 N_C 的间隔，这些部分结果将被复制到CUDA核心上的FP32寄存器中，在那里执行全精度FP32累加。如前所述，我们的细粒度量化沿内维度 `K`应用每组的缩放因子。这些缩放因子可以在CUDA核心上高效地乘以去量化过程，且附加计算成本极小。

值得注意的是，这种修改降低了单个warp组（warpgroup）的WGMMA（Warpgroup级矩阵乘累加）指令发射率。然而，在H800架构上，两个WGMMA通常可以同时存在：当一个warp组执行提升操作时，另一个能够执行MMA操作。这种设计使两种操作可以重叠，从而保持Tensor核心的高利用率。根据我们的实验，设置 N_C = 128 个元素，相当于4个WGMMAs，代表了能够显著提高精度且不引入大量开销的最小累加间隔。

尾数大于指数。 与先前工作（NVIDIA, 2024b; Peng et al., 2023b; Sun et al., 2019b）采用的混合FP8格式（使用 `E4M3` (4位指数和3位尾数)在 `Fprop` 和 `E5M2` (5位指数和2位尾数)在 `Dgrad` 和 `Wgrad`）不同，我们采用 `E4M3` 格式对所有张量进行更高精度的量化。我们将这种方法的可行性归因于我们的细粒度量化策略，即瓦片和块级缩放。通过操作较小的元素组，我们的方法有效地在这些分组元素之间共享指数位，减轻了动态范围有限的 影响。

在线量化。 在张量级量化框架（NVIDIA, 2024b; Peng et al., 2023b）中采用延迟量化，该量化方法维护了最大绝对值的 历史记录

values across prior iterations to infer the current value. In order to ensure accurate scales and simplify the framework, we calculate the maximum absolute value online for each `1x128` activation tile or `128x128` weight block. Based on it, we derive the scaling factor and then quantize the activation or weight online into the FP8 format.

3.3.3. Low-Precision Storage and Communication

In conjunction with our FP8 training framework, we further reduce the memory consumption and communication overhead by compressing cached activations and optimizer states into lower-precision formats.

Low-Precision Optimizer States. We adopt the BF16 data format instead of FP32 to track the first and second moments in the AdamW (Loshchilov and Hutter, 2017) optimizer, without incurring observable performance degradation. However, the master weights (stored by the optimizer) and gradients (used for batch size accumulation) are still retained in FP32 to ensure numerical stability throughout training.

Low-Precision Activation. As illustrated in Figure 6, the `Wgrad` operation is performed in FP8. To reduce the memory consumption, it is a natural choice to cache activations in FP8 format for the backward pass of the `Linear` operator. However, special considerations are taken on several operators for low-cost high-precision training:

> **(1) Inputs of the `Linear` after the attention operator.** These activations are also used in the backward pass of the attention operator, which makes it sensitive to precision. We adopt a customized `E5M6` data format exclusively for these activations. Additionally, these activations will be converted from an `1x128` quantization tile to an `128x1` tile in the backward pass. To avoid introducing extra quantization error, all the scaling factors are round scaled, i.e., integral power of 2.
>
> **(2) Inputs of the SwiGLU operator in MoE.** To further reduce the memory cost, we cache the inputs of the SwiGLU operator and recompute its output in the backward pass. These activations are also stored in FP8 with our fine-grained quantization method, striking a balance between memory efficiency and computational accuracy.

Low-Precision Communication. Communication bandwidth is a critical bottleneck in the training of MoE models. To alleviate this challenge, we quantize the activation before MoE up-projections into FP8 and then apply `dispatch` components, which is compatible with FP8 `Fprop` in MoE up-projections. Like the inputs of the `Linear` after the attention operator, scaling factors for this activation are integral power of 2. A similar strategy is applied to the activation gradient before MoE down-projections. For both the forward and backward `combine` components, we retain them in BF16 to preserve training precision in critical parts of the training pipeline.

3.4. Inference and Deployment

We deploy DeepSeek-V3 on the H800 cluster, where GPUs within each node are interconnected using NVLink, and all GPUs across the cluster are fully interconnected via IB. To simultaneously ensure both the Service-Level Objective (SLO) for online services and high throughput, we employ the following deployment strategy that separates the *prefilling* and *decoding* stages.

通过跨先前迭代的值来推断当前值。为了确保精确的缩放并简化框架，我们为每个 `1x128` 激活瓦片或 `128x128` 权重块在线计算最大绝对值。基于此，我们推导出缩放因子，然后将激活或权重在线量化为FP8格式。

3.3.3. 低精度存储和通信

结合我们的FP8训练框架，我们通过将缓存的激活和优化器状态压缩为低精度格式，进一步降低内存消耗和通信开销。

低精度优化器状态。 我们采用BF16数据格式而不是FP32来跟踪AdamW（Loshchilov和Hutter，2017）优化器中的第一和第二矩，而不会造成可观察的性能下降。然而，主权重（由优化器存储）和梯度（用于批大小累积）仍然保留在FP32中，以确保整个训练过程中的数值稳定性。

低精度激活。 如图6所示，`Wgrad` 操作在FP8中执行。为了减少内存消耗，将激活缓存为FP8格式以供 `Linear` 算子的反向传播使用是一种自然的选择。然而，对于几个算子，在低成本高精度训练中进行特殊考虑：

> **(1) 注意力算子后的 `Linear` 输入。** 这些激活也在注意力算子的反向传播中使用，这使得它对精度敏感。我们为这些激活采用定制的 `E5M6` 数据格式。此外，这些激活将在反向传播中从一个 `1x128` 量化块转换为 `128x1` 块。为了避免引入额外的量化误差，所有缩放因子都进行舍入缩放，即2的整数幂。
>
> **(2) MoE中的SwiGLU算子的输入。** 为了进一步减少内存成本，我们将SwiGLU算子的输入缓存起来，并在反向传播中重新计算其输出。这些激活也使用我们的细粒度量化方法以FP8格式存储，在内存效率和计算精度之间取得平衡。

低精度通信。 通信带宽是MoE模型训练中的关键瓶颈。为了缓解这一挑战，我们在MoE上投影到FP8之前对激活值进行量化，然后应用与MoE上投影中的FP8 `Fprop` 兼容的 `dispatch`组件。类似于注意力操作符后的 `Linear` 输入，此激活值的缩放因子是2的整数幂。对于MoE下投影前的激活梯度，也应用了类似的策略。对于正向和反向的 `combine`组件，我们保留它们在BF16中以在训练管道的关键部分保留训练精度。

3.4. 推理和部署

我们在H800集群上部署DeepSeek-V3，其中每个节点的GPU通过NVLink互连，集群中的所有GPU通过IB完全互连。为了同时确保在线服务的服务等级目标（SLO）和高吞吐量，我们采用以下部署策略，该策略将预填充 和 解码 阶段分开。

3.4.1. *Prefilling*

The minimum deployment unit of the prefilling stage consists of 4 nodes with 32 GPUs. The `attention` part employs 4-way Tensor Parallelism (TP4) with Sequence Parallelism (SP), combined with 8-way Data Parallelism (DP8). Its small TP size of 4 limits the overhead of TP communication. For the `MoE` part, we use 32-way Expert Parallelism (EP32), which ensures that each expert processes a sufficiently large batch size, thereby enhancing computational efficiency. For the `MoE` all-to-all communication, we use the same method as in training: first transferring tokens across nodes via IB, and then forwarding among the intra-node GPUs via NVLink. In particular, we use 1-way Tensor Parallelism for the dense MLPs in shallow layers to save TP communication.

To achieve load balancing among different experts in the `MoE` part, we need to ensure that each GPU processes approximately the same number of tokens. To this end, we introduce a deployment strategy of *redundant experts*, which duplicates high-load experts and deploys them redundantly. The high-load experts are detected based on statistics collected during the online deployment and are adjusted periodically (e.g., every 10 minutes). After determining the set of redundant experts, we carefully rearrange experts among GPUs within a node based on the observed loads, striving to balance the load across GPUs as much as possible without increasing the cross-node all-to-all communication overhead. For the deployment of DeepSeek-V3, we set 32 redundant experts for the prefilling stage. For each GPU, besides the original 8 experts it hosts, it will also host one additional redundant expert.

Furthermore, in the prefilling stage, to improve the throughput and hide the overhead of all-to-all and TP communication, we simultaneously process two micro-batches with similar computational workloads, overlapping the `attention` and `MoE` of one micro-batch with the `dispatch` and `combine` of another.

Finally, we are exploring a *dynamic redundancy* strategy for experts, where each GPU hosts more experts (e.g., 16 experts), but only 9 will be activated during each inference step. Before the all-to-all operation at each layer begins, we compute the globally optimal routing scheme on the fly. Given the substantial computation involved in the prefilling stage, the overhead of computing this routing scheme is almost negligible.

3.4.2. *Decoding*

During decoding, we treat the shared expert as a routed one. From this perspective, each token will select 9 experts during routing, where the shared expert is regarded as a heavy-load one that will always be selected. The minimum deployment unit of the decoding stage consists of 40 nodes with 320 GPUs. The `attention` part employs TP4 with SP, combined with DP80, while the `MoE` part uses EP320. For the `MoE` part, each GPU hosts only one expert, and 64 GPUs are responsible for hosting redundant experts and shared experts. All-to-all communication of the `dispatch` and `combine` parts is performed via direct point-to-point transfers over IB to achieve low latency. Additionally, we leverage the IBGDA (NVIDIA, 2022) technology to further minimize latency and enhance communication efficiency.

Similar to prefilling, we periodically determine the set of redundant experts in a certain interval, based on the statistical expert load from our online service. However, we do not need to rearrange experts since each GPU only hosts one expert. We are also exploring the *dynamic redundancy* strategy for decoding. However, this requires more careful optimization of the algorithm that computes the globally optimal routing scheme and the fusion with the `dispatch` kernel to reduce overhead.

3.4.1. 预填充

预填充阶段的最低部署单元由4个节点和32个GPU组成。The`attention` 部分采用4路张量并行（TP4）与序列并行（SP）相结合的8路数据并行（DP8）。其4的小TP规模限制了TP通信的开销。对于 The `MoE` 部分，我们使用32路专家并行（EP32），这确保了每个专家处理足够大的batchsize，从而提高了计算效率。对于 The `MoE` 全对全通信，我们使用与训练相同的方法：首先通过IB在节点之间传输token，然后在节点内GPU之间通过NVLink进行转发。特别是，对于浅层中的密集MLP，我们使用1路张量并行来节省TP通信。

为了在 The `MoE` 部分的不同专家之间实现负载均衡，我们需要确保每个GPU处理大约相同数量的token。为此，我们引入了一种 冗余专家 部署策略，该策略复制高负载专家并在冗余地部署它们。高负载专家是根据在线部署期间收集的统计数据检测到的，并定期调整（例如，每10分钟）。在确定冗余专家集之后，我们根据观察到的负载仔细在节点内的GPU之间重新安排专家，力求在不增加跨节点全对全通信开销的情况下尽可能平衡GPU之间的负载。对于DeepSeek-V3的部署，我们在预填充阶段设置了32个冗余专家。对于每个GPU，除了它原本托管的原8个专家外，它还将托管一个额外的冗余专家。

此外，在预填充阶段，为了提高所有对全和TP通信的吞吐量并隐藏开销，我们同时处理两个具有相似计算工作负载的微批次，将一个微批次的 `attention` 和 `MoE` 与另一个微批次的 `dispatch`和 `combine` 重叠。

最后，我们正在探索一种动态冗余 专家策略，其中每个GPU托管更多专家（例如，16个专家），但在每次推理步骤中只有9个将被激活。在每一层的所有对操作开始之前，我们在运行时计算全局最优路由方案。鉴于预填充阶段涉及大量计算，计算此路由方案的开销几乎可以忽略不计。

3.4.2. 解码

在解码期间，我们将共享专家视为已路由的一个。从这个角度来看，每个标记在路由期间将选择9个专家，其中共享专家被视为一个高负载的专家，它总是会被选择。解码阶段的最小部署单元由40个节点和320个GPU组成。 `attention` 部分采用TP4与SP，结合DP80，而 `MoE` 部分使用EP320。对于 `MoE` 部分，每个GPU只托管一个专家，而64个GPU负责托管冗余专家和共享专家。 `dispatch` 和 `combine` 部分的所有对通信通过IB直接点对点传输执行以实现低延迟。此外，我们利用IBGDA（NVIDIA，2022）技术进一步最小化延迟并提高通信效率。

与预填充类似，我们定期在一个特定的时间间隔内根据我们在线服务的统计专家负载确定冗余专家的集合。然而，我们不需要重新排列专家，因为每个GPU只托管一个专家。我们还正在探索用于解码的 动态冗余 策略。但是，这需要对计算全局最优路由方案和与 `dispatch` 内核融合的算法进行更仔细的优化，以减少开销。

Additionally, to enhance throughput and hide the overhead of all-to-all communication, we are also exploring processing two micro-batches with similar computational workloads simultaneously in the decoding stage. Unlike prefilling, `attention` consumes a larger portion of time in the decoding stage. Therefore, we overlap the `attention` of one micro-batch with the `dispatch+MoE+combine` of another. In the decoding stage, the batch size per expert is relatively small (usually within 256 tokens), and the bottleneck is memory access rather than computation. Since the `MoE` part only needs to load the parameters of one expert, the memory access overhead is minimal, so using fewer SMs will not significantly affect the overall performance. Therefore, to avoid impacting the computation speed of the `attention` part, we can allocate only a small portion of SMs to `dispatch+MoE+combine`.

3.5. Suggestions on Hardware Design

Based on our implementation of the all-to-all communication and FP8 training scheme, we propose the following suggestions on chip design to AI hardware vendors.

3.5.1. Communication Hardware

In DeepSeek-V3, we implement the overlap between computation and communication to hide the communication latency during computation. This significantly reduces the dependency on communication bandwidth compared to serial computation and communication. However, the current communication implementation relies on expensive SMs (e.g., we allocate 20 out of the 132 SMs available in the H800 GPU for this purpose), which will limit the computational throughput. Moreover, using SMs for communication results in significant inefficiencies, as tensor cores remain entirely under-utilized.

Currently, the SMs primarily perform the following tasks for all-to-all communication:

- **Forwarding data** between the IB (InfiniBand) and NVLink domain while aggregating IB traffic destined for multiple GPUs within the same node from a single GPU.
- **Transporting data** between RDMA buffers (registered GPU memory regions) and input/output buffers.
- **Executing `reduce` operations** for `all-to-all combine`.
- **Managing fine-grained memory layout** during chunked data transferring to multiple experts across the IB and NVLink domain.

We aspire to see future vendors developing hardware that offloads these communication tasks from the valuable computation unit SM, serving as a GPU co-processor or a network co-processor like NVIDIA SHARP Graham et al. (2016). Furthermore, to reduce application programming complexity, we aim for this hardware to unify the IB (scale-out) and NVLink (scale-up) networks from the perspective of the computation units. With this unified interface, computation units can easily accomplish operations such as `read`, `write`, `multicast`, and `reduce` across the entire IB-NVLink-unified domain via submitting communication requests based on simple primitives.

3.5.2. Compute Hardware

Higher FP8 GEMM Accumulation Precision in Tensor Cores. In the current Tensor Core implementation of the NVIDIA Hopper architecture, FP8 GEMM suffers from limited accumulation precision. After aligning 32 mantissa products by right-shifting based on the maximum exponent, the Tensor Core only uses the highest 14 bits of each mantissa product for addition,

此外，为了提高吞吐量并隐藏所有对全通信的开销，我们还在解码阶段探索同时处理两个具有相似计算工作负载的微批次。与预填充不同， `attention` 在解码阶段消耗了更多的时间。因此，我们将一个微批次的 `attention` 与另一个的 `dispatch+MoE+combine`重叠。在解码阶段，每个专家的批次大小相对较小（通常在256个token以内），瓶颈是内存访问而不是计算。由于 `MoE` 部分只需要加载一个专家的参数，内存访问开销很小，因此使用较少的SM不会显著影响整体性能。因此，为了避免影响 `attention` 部分的计算速度，我们可以仅将一小部分SM分配给 `dispatch+MoE+combine`。

3.5. 硬件设计建议

基于我们对所有对全通信和FP8训练方案的实施，我们向AI硬件供应商提出以下芯片设计建议。

3.5.1. 通信硬件

在 DeepSeek-V3 中，我们实现了计算与通信的重叠，以在计算过程中隐藏通信延迟。这显著降低了与串行计算和通信相比对通信带宽的依赖。然而，当前的通信实现依赖于昂贵的 SM（例如，我们为这个目的分配了 H800 GPU 中可用的 132 个 SM 中的 20 个），这将限制计算吞吐量。此外，使用 SM 进行通信会导致显著的低效性，因为张量核心完全未被充分利用。

Currently, the SM 主要为 all-to- all communication:

- **在 IB（InfiniBand）和 NVLink 域之间转发数据**，同时从单个 GPU 汇总 destined for 多个 GPU within the same node 的 IB 流量。
- **在 RDMA 缓冲区（注册的 GPU 内存区域）和输入/输出缓冲区之间传输数据**。
- **执行 `reduce` 操作** 为 `all-to-all combine`。
- **管理细粒度内存布局** 在将分块数据传输到 IB 和 NVLink 域的多个专家时。

我们期望未来的供应商开发能够从宝贵的计算单元 SM 卸载这些通信任务的硬件，作为 GPU 协处理器或像 NVIDIA SHARP Graham et al. (2016) 这样的网络协处理器。此外，为了降低应用程序编程复杂性，我们希望这种硬件能够从计算单元的角度统一 IB（扩展）和 NVLink（扩展）网络。通过这个统一的接口，计算单元可以通过基于简单原语提交通信请求，在整个 IB-NVLink 统一域内轻松完成 `read`, `write`, `multicast` 和`reduce` 等操作。

3.5.2. 计算硬件

张量核中的更高 FP8 GEMM 累加精度。 在 NVIDIA Hopper 架构的张量核当前实现中，FP8 GEMM 受限于有限的累加精度。在基于最大指数通过右移对 32 个尾数乘积进行对齐后，张量核仅使用每个尾数乘积的最高 14 位进行加法，

and truncates bits exceeding this range. The accumulation of addition results into registers also employs 14-bit precision. Our implementation partially mitigates the limitation by accumulating the addition results of 128 FP8×FP8 multiplications into registers with FP32 precision in the CUDA core. Although helpful in achieving successful FP8 training, it is merely a compromise due to the Hopper architecture's hardware deficiency in FP8 GEMM accumulation precision. Future chips need to adopt higher precision.

Support for Tile- and Block-Wise Quantization. Current GPUs only support per-tensor quantization, lacking the native support for fine-grained quantization like our tile- and block-wise quantization. In the current implementation, when the N_C interval is reached, the partial results will be copied from Tensor Cores to CUDA cores, multiplied by the scaling factors, and added to FP32 registers on CUDA cores. Although the dequantization overhead is significantly mitigated combined with our precise FP32 accumulation strategy, the frequent data movements between Tensor Cores and CUDA cores still limit the computational efficiency. Therefore, we recommend future chips to support fine-grained quantization by enabling Tensor Cores to receive scaling factors and implement MMA with group scaling. In this way, the whole partial sum accumulation and dequantization can be completed directly inside Tensor Cores until the final result is produced, avoiding frequent data movements.

Support for Online Quantization. The current implementations struggle to effectively support online quantization, despite its effectiveness demonstrated in our research. In the existing process, we need to read 128 BF16 activation values (the output of the previous computation) from HBM (High Bandwidth Memory) for quantization, and the quantized FP8 values are then written back to HBM, only to be read again for MMA. To address this inefficiency, we recommend that future chips integrate FP8 cast and TMA (Tensor Memory Accelerator) access into a single fused operation, so quantization can be completed during the transfer of activations from global memory to shared memory, avoiding frequent memory reads and writes. We also recommend supporting a warp-level cast instruction for speedup, which further facilitates the better fusion of layer normalization and FP8 cast. Alternatively, a near-memory computing approach can be adopted, where compute logic is placed near the HBM. In this case, BF16 elements can be cast to FP8 directly as they are read from HBM into the GPU, reducing off-chip memory access by roughly 50%.

Support for Transposed GEMM Operations. The current architecture makes it cumbersome to fuse matrix transposition with GEMM operations. In our workflow, activations during the forward pass are quantized into 1x128 FP8 tiles and stored. During the backward pass, the matrix needs to be read out, dequantized, transposed, re-quantized into 128x1 tiles, and stored in HBM. To reduce memory operations, we recommend future chips to enable direct transposed reads of matrices from shared memory before MMA operation, for those precisions required in both training and inference. Combined with the fusion of FP8 format conversion and TMA access, this enhancement will significantly streamline the quantization workflow.

4. Pre-Training

4.1. Data Construction

Compared with DeepSeek-V2, we optimize the pre-training corpus by enhancing the ratio of mathematical and programming samples, while expanding multilingual coverage beyond

并在该范围之外的位进行截断。将加法结果累积到寄存器中也采用14位精度。我们的实现通过将128个FP8×FP8乘法结果的加法结果以FP32精度累积到CUDA核心的寄存器中，部分缓解了这一限制。尽管这在实现成功的FP8训练中很有帮助，但由于Hopper架构在FP8 GEMM累积精度方面的硬件缺陷，这只是一个妥协。未来的芯片需要采用更高的精度。

支持块状和区块量化。 当前的GPU仅支持逐张量量化，缺乏对细粒度量化的原生支持，例如我们的块状和区块量化。在当前实现中，当达到 N_C 间隔时，部分结果将从张量核心复制到CUDA核心，乘以缩放因子，并加到CUDA核心上的FP32寄存器中。尽管结合我们精确的FP32累积策略，去量化开销显著降低，但张量核心和CUDA核心之间频繁的数据移动仍然限制了计算效率。因此，我们建议未来的芯片通过使张量核心能够接收缩放因子并实现带组缩放的MMA来支持细粒度量化。这样，整个部分和累积和去量化可以直接在张量核心内完成，直到最终结果产生，避免频繁的数据移动。

支持在线量化。 当前的实现难以有效支持在线量化，尽管我们在研究中证明了其有效性。在现有流程中，我们需要从 HBM（高带宽内存）读取 128 个 BF16 激活值（先前计算的结果）进行量化，然后将量化的 FP8 值写回 HBM，之后又需要再次读取这些值用于 MMA。为了解决这种低效问题，我们建议未来芯片将 FP8 转换和 TMA（张量内存加速器）访问整合为单个融合操作，以便在激活从全局内存传输到共享内存的过程中完成量化，避免频繁的内存读写。我们还建议支持 warp 级别的转换指令以加速处理，这进一步促进了层归一化和 FP8 转换的更好融合。或者，可以采用近内存计算方法，将计算逻辑放置在 HBM 附近。在这种情况下，BF16 元素可以在从 HBM 读取到 GPU 时直接转换为 FP8，减少片外内存访问约 50%。

支持转置GEMM操作。 当前的架构使得矩阵转置与GEMM操作的融合变得复杂。在我们的工作流程中，前向传播期间的激活值被量化为 1x128 FP8瓦片并存储。在反向传播期间，矩阵需要被读取、去量化、转置、重新量化为 128x1 瓦片，并存储在HBM中。为了减少内存操作，我们建议未来的芯片在MMA操作之前启用直接从共享内存读取转置矩阵的功能，这对于训练和推理中所需的精度都是必要的。结合FP8格式转换和TMA访问的融合，这项增强将显著简化量化工作流程。

4. 预训练

4.1. 数据构建

与DeepSeek-V2相比，我们通过提高数学和编程样本的比例，同时扩展多语言覆盖范围来优化预训练语料库，

English and Chinese. Also, our data processing pipeline is refined to minimize redundancy while maintaining corpus diversity. Inspired by Ding et al. (2024), we implement the document packing method for data integrity but do not incorporate cross-sample attention masking during training. Finally, the training corpus for DeepSeek-V3 consists of 14.8T high-quality and diverse tokens in our tokenizer.

In the training process of DeepSeekCoder-V2 (DeepSeek-AI, 2024a), we observe that the Fill-in-Middle (FIM) strategy does not compromise the next-token prediction capability while enabling the model to accurately predict middle text based on contextual cues. In alignment with DeepSeekCoder-V2, we also incorporate the FIM strategy in the pre-training of DeepSeek-V3. To be specific, we employ the Prefix-Suffix-Middle (PSM) framework to structure data as follows:

$$\texttt{<|fim\_begin|>}f_{\text{pre}}\texttt{<|fim\_hole|>}f_{\text{suf}}\texttt{<|fim\_end|>}f_{\text{middle}}\texttt{<|eos\_token|>}.$$

This structure is applied at the document level as a part of the pre-packing process. The FIM strategy is applied at a rate of 0.1, consistent with the PSM framework.

The tokenizer for DeepSeek-V3 employs Byte-level BPE (Shibata et al., 1999) with an extended vocabulary of 128K tokens. The pretokenizer and training data for our tokenizer are modified to optimize multilingual compression efficiency. In addition, compared with DeepSeek-V2, the new pretokenizer introduces tokens that combine punctuations and line breaks. However, this trick may introduce the token boundary bias (Lundberg, 2023) when the model processes multi-line prompts without terminal line breaks, particularly for few-shot evaluation prompts. To address this issue, we randomly split a certain proportion of such combined tokens during training, which exposes the model to a wider array of special cases and mitigates this bias.

4.2. Hyper-Parameters

Model Hyper-Parameters. We set the number of Transformer layers to 61 and the hidden dimension to 7168. All learnable parameters are randomly initialized with a standard deviation of 0.006. In MLA, we set the number of attention heads n_h to 128 and the per-head dimension d_h to 128. The KV compression dimension d_c is set to 512, and the query compression dimension d'_c is set to 1536. For the decoupled queries and key, we set the per-head dimension d_h^R to 64. We substitute all FFNs except for the first three layers with MoE layers. Each MoE layer consists of 1 shared expert and 256 routed experts, where the intermediate hidden dimension of each expert is 2048. Among the routed experts, 8 experts will be activated for each token, and each token will be ensured to be sent to at most 4 nodes. The multi-token prediction depth D is set to 1, i.e., besides the exact next token, each token will predict one additional token. As DeepSeek-V2, DeepSeek-V3 also employs additional RMSNorm layers after the compressed latent vectors, and multiplies additional scaling factors at the width bottlenecks. Under this configuration, DeepSeek-V3 comprises 671B total parameters, of which 37B are activated for each token.

Training Hyper-Parameters. We employ the AdamW optimizer (Loshchilov and Hutter, 2017) with hyper-parameters set to $\beta_1 = 0.9$, $\beta_2 = 0.95$, and weight_decay = 0.1. We set the maximum sequence length to 4K during pre-training, and pre-train DeepSeek-V3 on 14.8T tokens. As for the learning rate scheduling, we first linearly increase it from 0 to 2.2×10^{-4} during the first 2K steps. Then, we keep a constant learning rate of 2.2×10^{-4} until the model consumes 10T training tokens. Subsequently, we gradually decay the learning rate to 2.2×10^{-5} in 4.3T tokens, following a cosine decay curve. During the training of the final 500B tokens, we keep a constant learning rate of 2.2×10^{-5} in the first 333B tokens, and switch to another constant learning rate

英语和中文。此外，我们的数据处理流程经过优化，以最小化冗余同时保持语料库多样性。受 Ding 等人 (2024) 启发，我们实现了文档打包方法以确保数据完整性，但在训练过程中不包含跨样本注意力掩码。最后，DeepSeek-V3 的训练语料库包含我们分词器中的 14.8T 高质量和多样化的 token。

在 DeepSeekCoder-V2（DeepSeek-AI，2024a）的训练过程中，我们观察到填充中间（FIM）策略在启用模型能够基于上下文线索准确预测中间文本的同时，不会损害下一个 token 的预测能力。与 DeepSeekCoder-V2 一致，我们在 DeepSeek-V3 的预训练中也包含了 FIM 策略。具体来说，我们采用前缀-后缀-中间（PSM）框架来构建数据，如下所示：

$$\texttt{<|fim\_begin|>}f_{\text{pre}}\texttt{<|fim\_hole|>}f_{\text{suf}}\texttt{<|fim\_end|>}f_{\text{middle}}\texttt{<|eos\_token|>}.$$

这种结构在文档级别作为预打包过程的一部分应用。FIM 策略的应用率为 0.1，与 PSM 框架一致。

DeepSeek-V3 的分词器采用字节级 BPE（Shibata 等人，1999 年）并扩展了包含 128K 个词元的词汇表。我们的分词器的预分词器和训练数据已修改以优化多语言压缩效率。此外，与 DeepSeek-V2 相比，新的预分词器引入了结合标点和换行符的词元。然而，当模型处理没有终止换行符的多行提示时，这种技巧可能会引入词边界偏差（Lundberg，2023 年），尤其是在少样本评估提示中。为了解决这个问题，我们在训练期间随机拆分了一定比例的这些组合词元，这使模型接触到了更广泛的各种特殊情况并减轻了这种偏差。

4.2. 超参数

模型超参数。 我们将 Transformer 层数设置为 61，隐藏维度设置为 7168。所有可学习参数都使用标准差为 0.006 的随机初始化。在 MLA 中，我们将注意力头的数量 n_h 设置为 128，每个头的维度 d_h 设置为 128。KV 压缩维度 d_c 设置为 512，查询压缩维度 $d_{\prime c}$ 设置为 1536。对于解耦的查询和键，我们将每个头的维度 $d^R h$ 设置为 64。我们将除前三层以外的所有 FFN 替换为 MoE 层。每个 MoE 层由 1 个共享专家和 256 个路由专家组成，其中每个专家的中间隐藏维度为 2048。在路由专家中，每个词元将激活 8 个专家，并且每个词元将确保最多被发送到 4 个节点。多词元预测深度 D 设置为 1，即除了精确的下一个词元外，每个词元将预测一个额外的词元。与 DeepSeek-V2 一样，DeepSeek-V3 在压缩后的隐向量之后也采用了额外的 RMSNorm 层，并在宽度瓶颈处乘以额外的缩放因子。在此配置下，DeepSeek-V3 总参数量为 671B，其中每个词元激活 37B 参数。

训练超参数。 我们采用 AdamW 优化器（Loshchilov 和 Hutter，2017），其超参数设置为 $\beta_1 = 0.9$, $\beta_2 = 0.95$，以及权重_衰减 = 0.1。我们在预训练期间将最大序列长度设置为 4K，并在 14.8T token 上预训练 DeepSeek-V3。关于学习率调度，我们首先在最初的 2K 步内从 0 线性增加到 2.2×10^{-4}。然后，我们保持 2.2×10^{-4} 的恒定学习率，直到模型消耗 10T 训练 token。随后，我们逐渐将学习率衰减到 2.2×10^{-5}，在 4.3T token 内遵循余弦衰减曲线。在训练最后 500B token 期间，我们在最初的 333B token 内保持 2.2×10^{-5} 的恒定学习率，然后切换到另一个恒定学习率

of 7.3×10^{-6} in the remaining 167B tokens. The gradient clipping norm is set to 1.0. We employ a batch size scheduling strategy, where the batch size is gradually increased from 3072 to 15360 in the training of the first 469B tokens, and then keeps 15360 in the remaining training. We leverage pipeline parallelism to deploy different layers of a model on different GPUs, and for each layer, the routed experts will be uniformly deployed on 64 GPUs belonging to 8 nodes. As for the node-limited routing, each token will be sent to at most 4 nodes (i.e., $M = 4$). For auxiliary-loss-free load balancing, we set the bias update speed γ to 0.001 for the first 14.3T tokens, and to 0.0 for the remaining 500B tokens. For the balance loss, we set α to 0.0001, just to avoid extreme imbalance within any single sequence. The MTP loss weight λ is set to 0.3 for the first 10T tokens, and to 0.1 for the remaining 4.8T tokens.

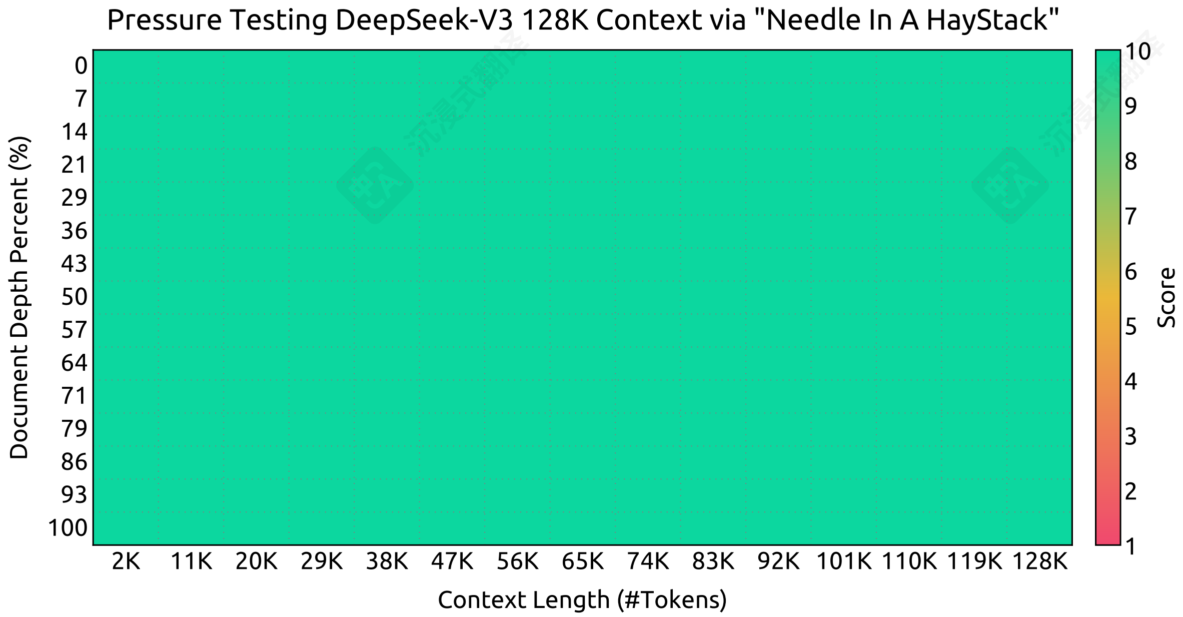


Figure 8 | Evaluation results on the "Needle In A Haystack" (NIAH) tests. DeepSeek-V3 performs well across all context window lengths up to 128K.

4.3. Long Context Extension

We adopt a similar approach to DeepSeek-V2 (DeepSeek-AI, 2024c) to enable long context capabilities in DeepSeek-V3. After the pre-training stage, we apply YaRN (Peng et al., 2023a) for context extension and perform two additional training phases, each comprising 1000 steps, to progressively expand the context window from 4K to 32K and then to 128K. The YaRN configuration is consistent with that used in DeepSeek-V2, being applied exclusively to the decoupled shared key $\mathbf{k}_t^R$. The hyper-parameters remain identical across both phases, with the scale $s = 40$, $\alpha = 1$, $\beta = 32$, and the scaling factor $\sqrt{t} = 0.1 \ln s + 1$. In the first phase, the sequence length is set to 32K, and the batch size is 1920. During the second phase, the sequence length is increased to 128K, and the batch size is reduced to 480. The learning rate for both phases is set to 7.3×10^{-6}, matching the final learning rate from the pre-training stage.

Through this two-phase extension training, DeepSeek-V3 is capable of handling inputs up to 128K in length while maintaining strong performance. Figure 8 illustrates that DeepSeek-V3, following supervised fine-tuning, achieves notable performance on the "Needle In A Haystack" (NIAH) test, demonstrating consistent robustness across context window lengths up to 128K.

在剩余的167B个token中，为7.3×10^{-6}。梯度裁剪范数设置为1.0。我们采用批量大小调度策略，在训练前469B个token时，批量大小从3072逐渐增加到15360，然后在剩余的训练中保持15360。我们利用流水线并行性将模型的的不同层部署在不同的GPU上，对于每一层，路由到的专家将均匀分布在属于8个节点的64个GPU上。对于节点限制路由，每个token最多将被发送到4个节点（即 $M = 4$）。对于无辅助损失负载均衡，我们首先为前14.3T个token将偏差更新速度γ 设置为0.001，然后在剩余的500B个token中设置为0.0。对于平衡损失，我们设置α 为0.0001，以避免任何单个序列内的极端不平衡。MTP损失权重 λ 在前10T个token中设置为0.3，在剩余的4.8T个token中设置为0.1。

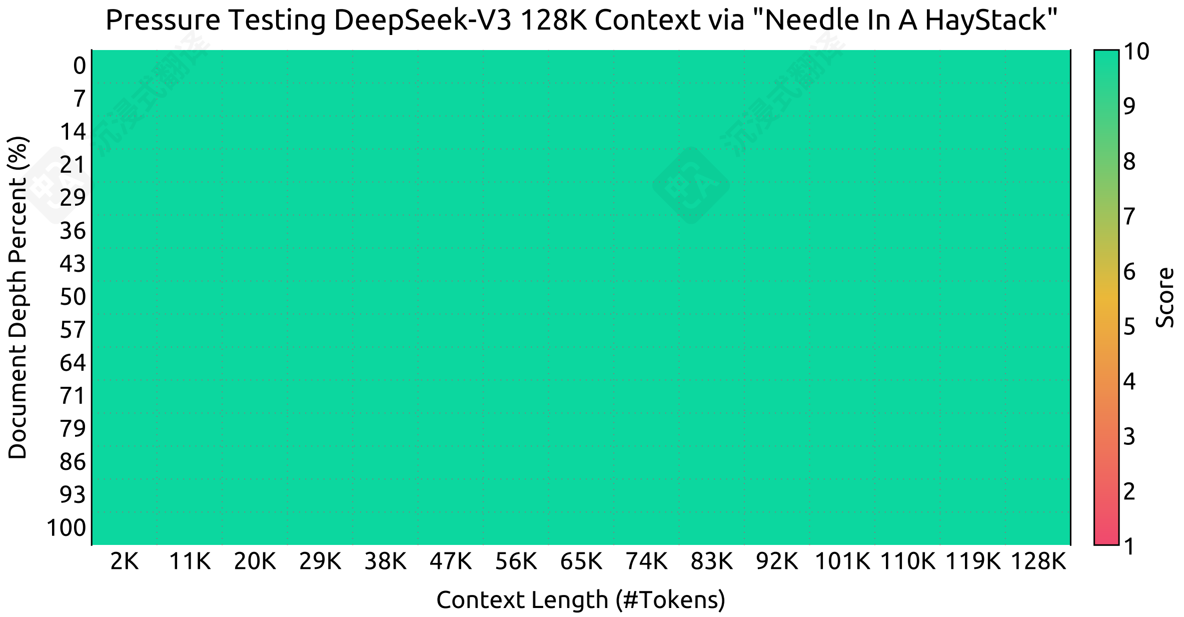


图 8 | 在“大海捞针”（NIAH）测试上的评估结果。DeepSeek-V3 在所有上下文窗口长度至 128K 的测试中都表现良好。

4.3. 长上下文扩展

我们采用与 DeepSeek-V2（DeepSeek-AI, 2024c）类似的方法，以在 DeepSeek-V3 中实现长上下文能力。在预训练阶段后，我们应用 YaRN（Peng et al., 2023a）进行上下文扩展，并执行两个额外的训练阶段，每个阶段包含 1000 步，以逐步将上下文窗口从 4K 扩展到 32K，然后扩展到 128K。YaRN 的配置与 DeepSeek-V2 中使用的配置一致，仅应用于解耦共享键 $\mathbf{k}_t^R$。两个阶段的超参数保持相同，包括缩放 $s = 40, \alpha = 1, \beta = 32$ 和缩放因子 $\sqrt{t} = 0$。1 ln $s + 1$。在第一阶段，序列长度设置为 32K，批大小为 1920。在第二阶段，序列长度增加到 128K，批大小减少到 480。两个阶段的 learning rate 设置为 7.3×10^{-6}，与预训练阶段的最终 learning rate 匹配。

通过这个两阶段扩展训练，DeepSeek-V3 能够处理长达 128K 的输入，同时保持优异的性能。图 8 显示，在经过监督微调后，DeepSeek-V3 在 "Needle In A Haystack" (NIAH) 测试中表现出色，并在上下文窗口长度高达 128K 的范围内展现出一致的鲁棒性。

4.4. Evaluations

4.4.1. Evaluation Benchmarks

The base model of DeepSeek-V3 is pretrained on a multilingual corpus with English and Chinese constituting the majority, so we evaluate its performance on a series of benchmarks primarily in English and Chinese, as well as on a multilingual benchmark. Our evaluation is based on our internal evaluation framework integrated in our HAI-LLM framework. Considered benchmarks are categorized and listed as follows, where underlined benchmarks are in Chinese and double-underlined benchmarks are multilingual ones:

Multi-subject multiple-choice datasets include MMLU (Hendrycks et al., 2020), MMLU-Redux (Gema et al., 2024), MMLU-Pro (Wang et al., 2024b), MMMLU (OpenAI, 2024b), C-Eval (Huang et al., 2023), and CMMLU (Li et al., 2023).

Language understanding and reasoning datasets include HellaSwag (Zellers et al., 2019), PIQA (Bisk et al., 2020), ARC (Clark et al., 2018), and BigBench Hard (BBH) (Suzgun et al., 2022).

Closed-book question answering datasets include TriviaQA (Joshi et al., 2017) and NaturalQuestions (Kwiatkowski et al., 2019).

Reading comprehension datasets include RACE Lai et al. (2017), DROP (Dua et al., 2019), C3 (Sun et al., 2019a), and CMRC (Cui et al., 2019).

Reference disambiguation datasets include CLUEWSC (Xu et al., 2020) and WinoGrande Sakaguchi et al. (2019).

Language modeling datasets include Pile (Gao et al., 2020).

Chinese understanding and culture datasets include CCPM (Li et al., 2021).

Math datasets include GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021), MGSM (Shi et al., 2023), and CMath (Wei et al., 2023).

Code datasets include HumanEval (Chen et al., 2021), LiveCodeBench-Base (0801-1101) (Jain et al., 2024), MBPP (Austin et al., 2021), and CRUXEval (Gu et al., 2024).

Standardized exams include AGIEval (Zhong et al., 2023). Note that AGIEval includes both English and Chinese subsets.

Following our previous work (DeepSeek-AI, 2024b,c), we adopt perplexity-based evaluation for datasets including HellaSwag, PIQA, WinoGrande, RACE-Middle, RACE-High, MMLU, MMLU-Redux, MMLU-Pro, MMMLU, ARC-Easy, ARC-Challenge, C-Eval, CMMLU, C3, and CCPM, and adopt generation-based evaluation for TriviaQA, NaturalQuestions, DROP, MATH, GSM8K, MGSM, HumanEval, MBPP, LiveCodeBench-Base, CRUXEval, BBH, AGIEval, CLUEWSC, CMRC, and CMath. In addition, we perform language-modeling-based evaluation for Pile-test and use Bits-Per-Byte (BPB) as the metric to guarantee fair comparison among models using different tokenizers.

4.4.2. Evaluation Results

In Table 3, we compare the base model of DeepSeek-V3 with the state-of-the-art open-source base models, including DeepSeek-V2-Base (DeepSeek-AI, 2024c) (our previous release), Qwen2.5 72B Base (Qwen, 2024b), and LLaMA-3.1 405B Base (AI@Meta, 2024b). We evaluate all these models with our internal evaluation framework, and ensure that they share the same evaluation setting. Note that due to the changes in our evaluation framework over the past months, the performance

4.4. 评估

4.4.1. 评估基准

DeepSeek-V3 的基础模型在包含英语和中文的多语言语料库上进行了预训练，因此我们在一系列以英语和中文为主的基准测试以及多语言基准测试上评估其性能。我们的评估基于我们集成在 HAI-LLM 框架中的内部评估框架。考虑的基准测试按类别列出如下，其中下划线表示的基准测试为中文，双下划线表示的多语言基准测试如下：

多主题多选题 数据集包括 MMLU (Hendrycks et al., 2020), MMLU- Redux (Gema et al., 2024), MMLU-Pro (Wang et al., 2024b), MMMLU (OpenAI, 2024b), C-Eval (Huang et al., 2023), 和 CMMLU(Li et al., 2023)。

语言理解和推理 数据集包括 HellaSwag (Zellers et al., 2019), PIQA (Bisk etal., 2020), ARC (Clark et al., 2018), 和 BigBench Hard (BBH) (Suzgun et al., 2022)。

闭卷问答数据集包括 TriviaQA (Joshi etal., 2017)和 NaturalQuestions (Kwiatkowski et al., 2019).

阅读理解数据集包括 RACE Lai et al.(2017),DROP (Duaetal., 2019), C3 (Sun et al., 2019a), 和 CMRC (Cui et al., 2019).

指代消解数据集包括 CLUEWSC (Xu et al., 2020) 和 WinoGrande Sakaguchi etal. (2019).

语言建模数据集包括 Pile (Gao et al., 2020).

中文理解和文化数据集包括 CCPM (Li et al., 2021) .

数学数据集包括 GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021), MGSM (Shi et al., 2023), 和 CMath (Wei et al., 2023).

代码 数据集包括 HumanEval (Chen 等人，2021年)、LiveCodeBench-Base (0801-1101) (Jain 等人，2024年)、MBPP (Austin 等人，2021年) 和 CRUXEval (Gu 等人，2024年)。

标准化考试包括 AGIEval (Zhong 等人，2023年)。请注意，AGIEval 包括英语和中文子集。

根据我们之前的工作 (DeepSeek-AI, 2024b,c)，我们采用基于困惑度的评估方法对包括 HellaSwag、PIQA、WinoGrande、RACE-Middle、RACE-High、MMLU、MMLU-Redux、MMLU-Pro、MMMLU、ARC-Easy、ARC-Challenge、C-Eval、CMMLU、C3 和 CCPM 的数据集进行评估，并采用基于生成的评估方法对 TriviaQA、NaturalQuestions、DROP、MATH、GSM8K、MGSM、HumanEval、MBPP、LiveCodeBench-Base、CRUXEval、BBH、AGIEval、CLUEWSC、CMRC 和 CMath 进行评估。此外，我们对 Pile-test 进行基于语言模型的评估，并使用每字节比特数 (BPB) 作为指标，以确保使用不同分词器的模型之间进行公平比较。

4.4.2. 评估结果

在表3中，我们将DeepSeek-V3的基础模型与最先进的开源基础模型进行比较，包括 DeepSeek-V2-Base（DeepSeek-AI，2024c）（我们之前的发布版本）、Qwen2.5 72B Base（Qwen，2024b）和LLaMA-3.1 405B Base（AI@Meta，2024b）。我们使用内部评估框架评估所有这些模型，并确保它们共享相同的评估设置。请注意，由于我们评估框架在过去几个月中的变化，性能

	Benchmark (Metric)	# Shots	DeepSeek-V2 Base	Qwen2.5 72B Base	LLaMA-3.1 405B Base	DeepSeek-V3 Base
	Architecture	-	MoE	Dense	Dense	MoE
	# Activated Params	-	21B	72B	405B	37B
	# Total Params	-	236B	72B	405B	671B
English	Pile-test (BPB)	-	0.606	0.638	**0.542**	0.548
	BBH (EM)	3-shot	78.8	79.8	82.9	**87.5**
	MMLU (EM)	5-shot	78.4	85.0	84.4	**87.1**
	MMLU-Redux (EM)	5-shot	75.6	83.2	81.3	**86.2**
	MMLU-Pro (EM)	5-shot	51.4	58.3	52.8	**64.4**
	DROP (F1)	3-shot	80.4	80.6	86.0	**89.0**
	ARC-Easy (EM)	25-shot	97.6	98.4	98.4	**98.9**
	ARC-Challenge (EM)	25-shot	92.2	94.5	**95.3**	**95.3**
	HellaSwag (EM)	10-shot	87.1	84.8	**89.2**	**88.9**
	PIQA (EM)	0-shot	83.9	82.6	**85.9**	84.7
	WinoGrande (EM)	5-shot	**86.3**	82.3	85.2	84.9
	RACE-Middle (EM)	5-shot	73.1	68.1	**74.2**	67.1
	RACE-High (EM)	5-shot	52.6	50.3	**56.8**	51.3
	TriviaQA (EM)	5-shot	80.0	71.9	**82.7**	**82.9**
	NaturalQuestions (EM)	5-shot	38.6	33.2	**41.5**	40.0
	AGIEval (EM)	0-shot	57.5	75.8	60.6	**79.6**
Code	HumanEval (Pass@1)	0-shot	43.3	53.0	54.9	**65.2**
	MBPP (Pass@1)	3-shot	65.0	72.6	68.4	**75.4**
	LiveCodeBench-Base (Pass@1)	3-shot	11.6	12.9	15.5	**19.4**
	CRUXEval-I (EM)	2-shot	52.5	59.1	58.5	**67.3**
	CRUXEval-O (EM)	2-shot	49.8	59.9	59.9	**69.8**
Math	GSM8K (EM)	8-shot	81.6	88.3	83.5	**89.3**
	MATH (EM)	4-shot	43.4	54.4	49.0	**61.6**
	MGSM (EM)	8-shot	63.6	76.2	69.9	**79.8**
	CMath (EM)	3-shot	78.7	84.5	77.3	**90.7**
Chinese	CLUEWSC (EM)	5-shot	82.0	82.5	**83.0**	**82.7**
	C-Eval (EM)	5-shot	81.4	89.2	72.5	**90.1**
	CMMLU (EM)	5-shot	84.0	**89.5**	73.7	88.8
	CMRC (EM)	1-shot	**77.4**	75.8	76.0	76.3
	C3 (EM)	0-shot	77.4	76.7	**79.7**	78.6
	CCPM (EM)	0-shot	**93.0**	88.5	78.6	92.0
Multilingual	MMMLU-non-English (EM)	5-shot	64.0	74.8	73.8	**79.4**

Table 3 | Comparison among DeepSeek-V3-Base and other representative open-source base models. All models are evaluated in our internal framework and share the same evaluation setting. Scores with a gap not exceeding 0.3 are considered to be at the same level. DeepSeek-V3-Base achieves the best performance on most benchmarks, especially on math and code tasks.

of DeepSeek-V2-Base exhibits a slight difference from our previously reported results. Overall, DeepSeek-V3-Base comprehensively outperforms DeepSeek-V2-Base and Qwen2.5 72B Base, and surpasses LLaMA-3.1 405B Base in the majority of benchmarks, essentially becoming the strongest open-source model.

From a more detailed perspective, we compare DeepSeek-V3-Base with the other open-source base models individually. (1) Compared with DeepSeek-V2-Base, due to the improvements in our model architecture, the scale-up of the model size and training tokens, and the enhancement of data quality, DeepSeek-V3-Base achieves significantly better performance as expected. (2) Compared with Qwen2.5 72B Base, the state-of-the-art Chinese open-source model, with only half of the activated parameters, DeepSeek-V3-Base also demonstrates remarkable advantages,

	基准测试 (Metric)	# 射击次数	DeepSeek-V2 Base	Qwen2.5 72B Base	LLaMA-3.1 405B Base	DeepSeek-V3 Base
	架构	-	MoE	密集	密集	MoE
	# 激活参数	-	21B	72B	405B	37B
	# 总参数量	-	236B	72B	405B	671B
English	Pile-test (BPB)	-	0.606	0.638	**0.542**	0.548
	BBH (EM)	3-shot	78.8	79.8	82.9	**87.5**
	MMLU (EM)	5-shot	78.4	85.0	84.4	**87.1**
	MMLU-Redux (EM)	5-shot	75.6	83.2	81.3	**86.2**
	MMLU-Pro (EM)	5-shot	51.4	58.3	52.8	**64.4**
	DROP (F1)	3-shot	80.4	80.6	86.0	**89.0**
	ARC-Easy (EM)	25-shot	97.6	98.4	98.4	**98.9**
	ARC-Challenge (EM)	25-shot	92.2	94.5	**95.3**	**95.3**
	HellaSwag (EM)	10-shot	87.1	84.8	**89.2**	**88.9**
	PIQA (EM)	0-shot	83.9	82.6	**85.9**	84.7
	WinoGrande (EM)	5-shot	**86.3**	82.3	85.2	84.9
	RACE-Middle (EM)	5-shot	73.1	68.1	**74.2**	67.1
	RACE-High (EM)	5-shot	52.6	50.3	**56.8**	51.3
	TriviaQA (EM)	5-shot	80.0	71.9	**82.7**	**82.9**
	NaturalQuestions (EM)	5-shot	38.6	33.2	**41.5**	40.0
	AGIEval (EM)	0-shot	57.5	75.8	60.6	**79.6**
Code	HumanEval (Pass@1)	0-shot	43.3	53.0	54.9	**65.2**
	MBPP(Pass@1	3-shot	65.0	72.6	68.4	**75.4**
	LiveCodeBench-Base (P ass@1)	3-shot	11.6	12.9	15.5	**19.4**
	CRUXEval-I (EM)	2-shot	52.5	59.1	58.5	**67.3**
	CRUXEval-O (EM)	2-shot	49.8	59.9	59.9	**69.8**
Math	GSM8K (EM)	8-shot	81.6	88.3	83.5	**89.3**
	MATH (EM)	4-shot	43.4	54.4	49.0	**61.6**
	MGSM (EM)	8-shot	63.6	76.2	69.9	**79.8**
	CMath (EM)	3-shot	78.7	84.5	77.3	**90.7**
Chinese	CLUEWSC (EM)	5-shot	82.0	82.5	**83.0**	**82.7**
	C-Eval (EM)	5-shot	81.4	89.2	72.5	**90.1**
	CMMLU (EM)	5-shot	84.0	**89.5**	73.7	88.8
	CMRC (EM)	1-shot	**77.4**	75.8	76.0	76.3
	C3 (EM)	0-shot	77.4	76.7	**79.7**	78.6
	CCPM (EM)	0-shot	**93.0**	88.5	78.6	92.0
多语言MMMLU-非英语(EM)		5-shot	64.0	74.8	73.8	**79.4**

Table 3 | DeepSeek-V3-Base与其他代表性开源基础模型的对比。所有模型均在我们的内部框架中进行评估，并共享相同的评估设置。分数差距不超过0.3的被视为同一水平。DeepSeek-V3-Base在大多数基准测试中表现最佳，尤其是在数学和代码任务上。

DeepSeek-V2-Base的结果与我们的先前报告结果略有差异。总体而言，DeepSeek-V3-Base全面优于DeepSeek-V2-Base和Qwen2.5 72B Base，并在大多数基准测试中超越了LLaMA-3.1 405B Base，实质上成为最强的开源模型。

从更详细的角度来看，我们将DeepSeek-V3-Base与其他开源基础模型单独进行比较。(1) 与DeepSeek-V2-Base相比，由于我们模型架构的改进、模型规模和训练token的扩大以及数据质量的提升，DeepSeek-V3-Base达到了预期的显著更好的性能。(2) 与Qwen2.5 72B Base相比，作为最先进的中文开源模型，其激活参数仅为其一半，DeepSeek-V3-Base也展现了显著的优势，

especially on English, multilingual, code, and math benchmarks. As for Chinese benchmarks, except for CMMLU, a Chinese multi-subject multiple-choice task, DeepSeek-V3-Base also shows better performance than Qwen2.5 72B. (3) Compared with LLaMA-3.1 405B Base, the largest open-source model with 11 times the activated parameters, DeepSeek-V3-Base also exhibits much better performance on multilingual, code, and math benchmarks. As for English and Chinese language benchmarks, DeepSeek-V3-Base shows competitive or better performance, and is especially good on BBH, MMLU-series, DROP, C-Eval, CMMLU, and CCPM.

Due to our efficient architectures and comprehensive engineering optimizations, DeepSeek-V3 achieves extremely high training efficiency. Under our training framework and infrastructures, training DeepSeek-V3 on each trillion tokens requires only 180K H800 GPU hours, which is much cheaper than training 72B or 405B dense models.

Benchmark (Metric)	# Shots	Small MoE Baseline	Small MoE w/ MTP	Large MoE Baseline	Large MoE w/ MTP
# Activated Params (Inference)	-	2.4B	2.4B	20.9B	20.9B
# Total Params (Inference)	-	15.7B	15.7B	228.7B	228.7B
# Training Tokens	-	1.33T	1.33T	540B	540B
Pile-test (BPB)	-	**0.729**	**0.729**	0.658	**0.657**
BBH (EM)	3-shot	39.0	**41.4**	70.0	**70.7**
MMLU (EM)	5-shot	50.0	**53.3**	**67.5**	66.6
DROP (F1)	1-shot	39.2	**41.3**	68.5	**70.6**
TriviaQA (EM)	5-shot	56.9	**57.7**	**67.0**	**67.3**
NaturalQuestions (EM)	5-shot	**22.7**	22.3	27.2	**28.5**
HumanEval (Pass@1)	0-shot	20.7	**26.8**	44.5	**53.7**
MBPP (Pass@1)	3-shot	35.8	**36.8**	61.6	**62.2**
GSM8K (EM)	8-shot	25.4	**31.4**	72.3	**74.0**
MATH (EM)	4-shot	10.7	**12.6**	38.6	**39.8**

Table 4 | Ablation results for the MTP strategy. The MTP strategy consistently enhances the model performance on most of the evaluation benchmarks.

4.5. Discussion

4.5.1. Ablation Studies for Multi-Token Prediction

In Table 4, we show the ablation results for the MTP strategy. To be specific, we validate the MTP strategy on top of two baseline models across different scales. At the small scale, we train a baseline MoE model comprising 15.7B total parameters on 1.33T tokens. At the large scale, we train a baseline MoE model comprising 228.7B total parameters on 540B tokens. On top of them, keeping the training data and the other architectures the same, we append a 1-depth MTP module onto them and train two models with the MTP strategy for comparison. Note that during inference, we directly discard the MTP module, so the inference costs of the compared models are exactly the same. From the table, we can observe that the MTP strategy consistently enhances the model performance on most of the evaluation benchmarks.

4.5.2. Ablation Studies for the Auxiliary-Loss-Free Balancing Strategy

In Table 5, we show the ablation results for the auxiliary-loss-free balancing strategy. We validate this strategy on top of two baseline models across different scales. At the small scale, we train a baseline MoE model comprising 15.7B total parameters on 1.33T tokens. At the large scale, we train a baseline MoE model comprising 228.7B total parameters on 578B tokens.

尤其是在英语、多语言、代码和数学基准测试上。对于中文基准测试，除了CMMLU（一个中文多主题选择题任务），DeepSeek-V3-Base的表现也优于Qwen2.5 72B。（3）与拥有11倍激活参数的最大的开源模型LLaMA-3.1 405B Base相比，DeepSeek-V3-Base在多语言、代码和数学基准测试上表现也明显更好。对于英语和中文语言基准测试，DeepSeek-V3-Base表现出具有竞争力的或更好的性能，尤其是在BBH、MMLU系列、DROP、C-Eval、CMMLU和CCPM上。

由于我们高效的建筑和全面的工程优化，DeepSeek-V3实现了极高的训练效率。在我们的训练框架和基础设施下，每个万亿token的DeepSeek-V3训练只需要180K H800 GPU小时，这比训练72B或405B密集模型要便宜得多。

基准 (指标)	# 射击次数	Small MoE 基线	Small MoE w/ MTP	Large MoE 基线	Large MoE w/ MTP
# 激活参数(Inference)	-	2.4B	2.4B	20.9B	20.9B
# 总参数(Inference)	-	15.7B	15.7B	228.7B	228.7B
# 训练Token	-	1.33T	1.33T	540B	540B
Pile-test (BPB)	-	**0.729**	**0.729**	0.658	**0.657**
BBH (EM)	3-shot	39.0	**41.4**	70.0	**70.7**
MMLU (EM)	5-shot	50.0	**53.3**	**67.5**	66.6
DROP (F1)	1-shot	39.2	**41.3**	68.5	**70.6**
TriviaQA (EM)	5-shot	56.9	**57.7**	**67.0**	**67.3**
NaturalQuestions (EM)	5-shot	**22.7**	22.3	27.2	**28.5**
HumanEval (Pass@1)	0-shot	20.7	**26.8**	44.5	**53.7**
MBPP (Pass@1)	3-shot	35.8	**36.8**	61.6	**62.2**
GSM8K (EM)	8-shot	25.4	**31.4**	72.3	**74.0**
MATH (EM)	4-shot	10.7	**12.6**	38.6	**39.8**

Table 4 | Ablation results for the MTP strategy. The MTP strategy consistently enhances the model performance on most of the evaluation benchmarks.

4.5. 讨论

4.5.1. 多*token*预测的消融研究

在表4中，我们展示了MTP策略的消融结果。具体来说，我们在不同规模上基于两个基线模型验证了MTP策略。在小规模上，我们训练了一个包含157B总参数的基线MoE模型，使用1.33T tokens进行训练。在大规模上，我们训练了一个包含2287B总参数的基线MoE模型，使用540B tokens进行训练。在此基础上，保持训练数据和其它架构不变，我们向它们附加一个1-depth MTP模块，并使用MTP策略训练两个模型进行比较。请注意，在推理时，我们直接丢弃了MTP模块，因此比较模型的推理成本完全相同。从表格中，我们可以观察到MTP策略在大多数评估基准上始终提升了模型性能。

4.5.2. 辅助损失无平衡策略的消融研究

在表5中，我们展示了辅助损失无平衡策略的消融结果。我们在不同规模上基于两个基线模型验证该策略。在小规模上，我们训练了一个包含157亿总参数的基线MoE模型，使用1.33T tokens进行训练。在大规模上，我们训练了一个包含2287亿总参数的基线MoE模型，使用578B tokens进行训练。

Benchmark (Metric)	# Shots	Small MoE Aux-Loss-Based	Small MoE Aux-Loss-Free	Large MoE Aux-Loss-Based	Large MoE Aux-Loss-Free
# Activated Params	-	2.4B	2.4B	20.9B	20.9B
# Total Params	-	15.7B	15.7B	228.7B	228.7B
# Training Tokens	-	1.33T	1.33T	578B	578B
Pile-test (BPB)	-	0.727	**0.724**	0.656	**0.652**
BBH (EM)	3-shot	37.3	**39.3**	66.7	**67.9**
MMLU (EM)	5-shot	51.0	**51.8**	**68.3**	67.2
DROP (F1)	1-shot	38.1	**39.0**	**67.1**	**67.1**
TriviaQA (EM)	5-shot	**58.3**	**58.5**	66.7	**67.7**
NaturalQuestions (EM)	5-shot	**23.2**	**23.4**	27.1	**28.1**
HumanEval (Pass@1)	0-shot	22.0	**22.6**	40.2	**46.3**
MBPP (Pass@1)	3-shot	**36.6**	35.8	59.2	**61.2**
GSM8K (EM)	8-shot	27.1	**29.6**	70.7	**74.5**
MATH (EM)	4-shot	**10.9**	**11.1**	37.2	**39.6**

Table 5 | Ablation results for the auxiliary-loss-free balancing strategy. Compared with the purely auxiliary-loss-based method, the auxiliary-loss-free strategy consistently achieves better model performance on most of the evaluation benchmarks.

Both of the baseline models purely use auxiliary losses to encourage load balance, and use the sigmoid gating function with top-K affinity normalization. Their hyper-parameters to control the strength of auxiliary losses are the same as DeepSeek-V2-Lite and DeepSeek-V2, respectively. On top of these two baseline models, keeping the training data and the other architectures the same, we remove all auxiliary losses and introduce the auxiliary-loss-free balancing strategy for comparison. From the table, we can observe that the auxiliary-loss-free strategy consistently achieves better model performance on most of the evaluation benchmarks.

4.5.3. Batch-Wise Load Balance VS. Sequence-Wise Load Balance

The key distinction between auxiliary-loss-free balancing and sequence-wise auxiliary loss lies in their balancing scope: batch-wise versus sequence-wise. Compared with the sequence-wise auxiliary loss, batch-wise balancing imposes a more flexible constraint, as it does not enforce in-domain balance on each sequence. This flexibility allows experts to better specialize in different domains. To validate this, we record and analyze the expert load of a 16B auxiliary-loss-based baseline and a 16B auxiliary-loss-free model on different domains in the Pile test set. As illustrated in Figure 9, we observe that the auxiliary-loss-free model demonstrates greater expert specialization patterns as expected.

To further investigate the correlation between this flexibility and the advantage in model performance, we additionally design and validate a batch-wise auxiliary loss that encourages load balance on each training batch instead of on each sequence. The experimental results show that, when achieving a similar level of batch-wise load balance, the batch-wise auxiliary loss can also achieve similar model performance to the auxiliary-loss-free method. To be specific, in our experiments with 1B MoE models, the validation losses are: 2.258 (using a sequence-wise auxiliary loss), 2.253 (using the auxiliary-loss-free method), and 2.253 (using a batch-wise auxiliary loss). We also observe similar results on 3B MoE models: the model using a sequence-wise auxiliary loss achieves a validation loss of 2.085, and the models using the auxiliary-loss-free method or a batch-wise auxiliary loss achieve the same validation loss of 2.080.

In addition, although the batch-wise load balancing methods show consistent performance advantages, they also face two potential challenges in efficiency: (1) load imbalance within

基准（指标）	# 射击次数	Small MoE 基于辅助损失的	Small MoE 无辅助损失的	Large MoE 基于辅助损失的	Large MoE 无辅助损失的
# 激活参数	-	2.4B	2.4B	20.9B	20.9B
# Total Params	-	15.7B	15.7B	228.7B	228.7B
# 训练Token	-	1.33T	1.33T	578B	578B
Pile-test (BPB)	-	0.727	**0.724**	0.656	**0.652**
BBH (EM)	3-shot	37.3	**39.3**	66.7	**67.9**
MMLU (EM)	5-shot	51.0	**51.8**	**68.3**	67.2
DROP (F1)	1-shot	38.1	**39.0**	**67.1**	**67.1**
TriviaQA (EM)	5-shot	**58.3**	**58.5**	66.7	**67.7**
NaturalQuestions (EM)	5-shot	**23.2**	**23.4**	27.1	**28.1**
HumanEval (Pass@1)	0-shot	22.0	**22.6**	40.2	**46.3**
MBPP (Pass@1)	3-shot	**36.6**	35.8	59.2	**61.2**
GSM8K (EM)	8-shot	27.1	**29.6**	70.7	**74.5**
MATH (EM)	4-shot	**10.9**	**11.1**	37.2	**39.6**

Table 5 | Ablation results for the auxiliary-loss-free balancing strategy. Compared with the purely auxiliary-loss-based method, the auxiliary-loss-free strategy consistently achieves better model performance on most of the evaluation benchmarks.

所有的基线模型纯粹使用辅助损失来鼓励负载均衡，并使用具有top-K亲和度归一化的sigmoid门控函数。它们控制辅助损失强度的超参数分别与DeepSeek-V2-Lite和DeepSeek-V2相同。在保持这两个基线模型的训练数据和其它架构相同的基础上，我们移除所有辅助损失，并引入无辅助损失的均衡策略进行比较。从表格中，我们可以观察到无辅助损失策略在大多数评估基准上始终实现了更好的模型性能。

4.5.3. 批次式负载均衡 *VS.* 序列式负载均衡

无辅助损失均衡与序列式辅助损失之间的关键区别在于它们的均衡范围：批次式与序列式。与序列式辅助损失相比，批次式均衡施加了更灵活的约束，因为它不会对每个序列强制执行领域内均衡。这种灵活性允许专家更好地专注于不同的领域。为了验证这一点，我们记录并分析了在Pile测试集的不同领域中，一个16B基于辅助损失的基线和16B无辅助损失模型的专业负载。如图9所示，我们观察到无辅助损失模型如预期展示了更强的专家专业化模式。

为了进一步研究这种灵活性与模型性能优势之间的相关性，我们额外设计并验证了一种批处理方式的辅助损失，该损失鼓励在每个训练批次上而不是每个序列上实现负载平衡。实验结果表明，当实现相似的批处理方式负载平衡水平时，批处理方式的辅助损失也能达到与无辅助损失方法相似的模型性能。具体来说，在我们的1B MoE模型实验中，验证损失分别为：2.258（使用序列方式辅助损失）、2.253（使用无辅助损失方法）和2.253（使用批处理方式辅助损失）。我们在3B MoE模型上也观察到相似的结果：使用序列方式辅助损失的模型实现了2.085的验证损失，而使用无辅助损失方法或批处理方式辅助损失的模型则达到了相同的2.080验证损失。

此外，尽管批处理方式负载平衡方法表现出一致的性能优势，但它们在效率方面也面临两个潜在挑战：(1) 负载不平衡存在于

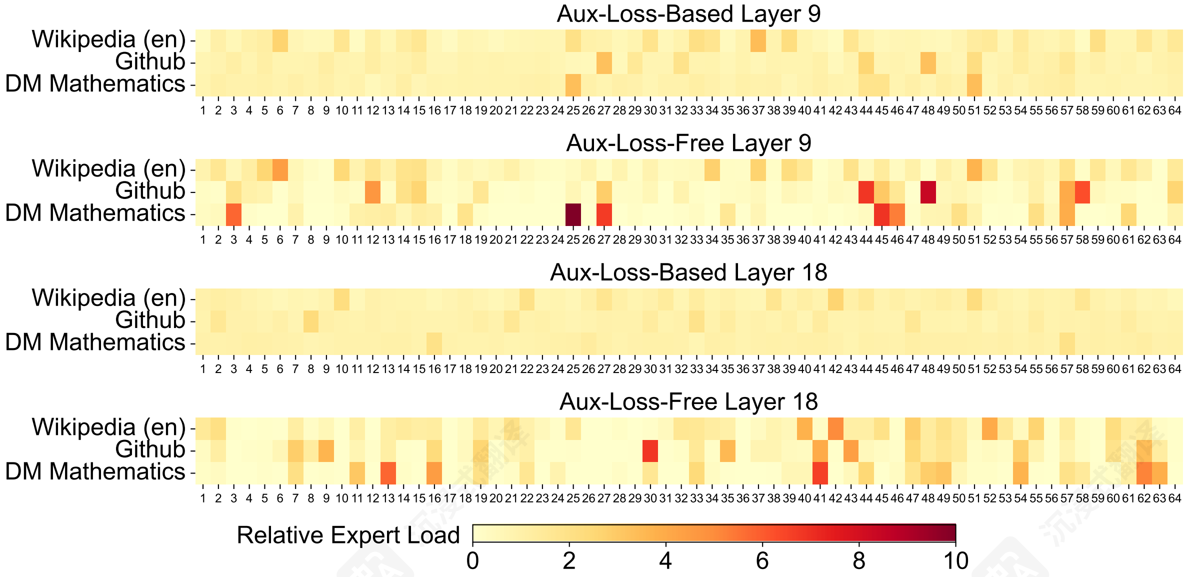


Figure 9 | Expert load of auxiliary-loss-free and auxiliary-loss-based models on three domains in the Pile test set. The auxiliary-loss-free model shows greater expert specialization patterns than the auxiliary-loss-based one. The relative expert load denotes the ratio between the actual expert load and the theoretically balanced expert load. Due to space constraints, we only present the results of two layers as an example, with the results of all layers provided in Appendix C.

certain sequences or small batches, and (2) domain-shift-induced load imbalance during inference. The first challenge is naturally addressed by our training framework that uses large-scale expert parallelism and data parallelism, which guarantees a large size of each micro-batch. For the second challenge, we also design and implement an efficient inference framework with redundant expert deployment, as described in Section 3.4, to overcome it.

5. Post-Training

5.1. Supervised Fine-Tuning

We curate our instruction-tuning datasets to include 1.5M instances spanning multiple domains, with each domain employing distinct data creation methods tailored to its specific requirements.

Reasoning Data. For reasoning-related datasets, including those focused on mathematics, code competition problems, and logic puzzles, we generate the data by leveraging an internal DeepSeek-R1 model. Specifically, while the R1-generated data demonstrates strong accuracy, it suffers from issues such as overthinking, poor formatting, and excessive length. Our objective is to balance the high accuracy of R1-generated reasoning data and the clarity and conciseness of regularly formatted reasoning data.

To establish our methodology, we begin by developing an expert model tailored to a specific domain, such as code, mathematics, or general reasoning, using a combined Supervised Fine-Tuning (SFT) and Reinforcement Learning (RL) training pipeline. This expert model serves as a data generator for the final model. The training process involves generating two distinct types of SFT samples for each instance: the first couples the problem with its original response in the format of <problem, original response>, while the second incorporates a system prompt

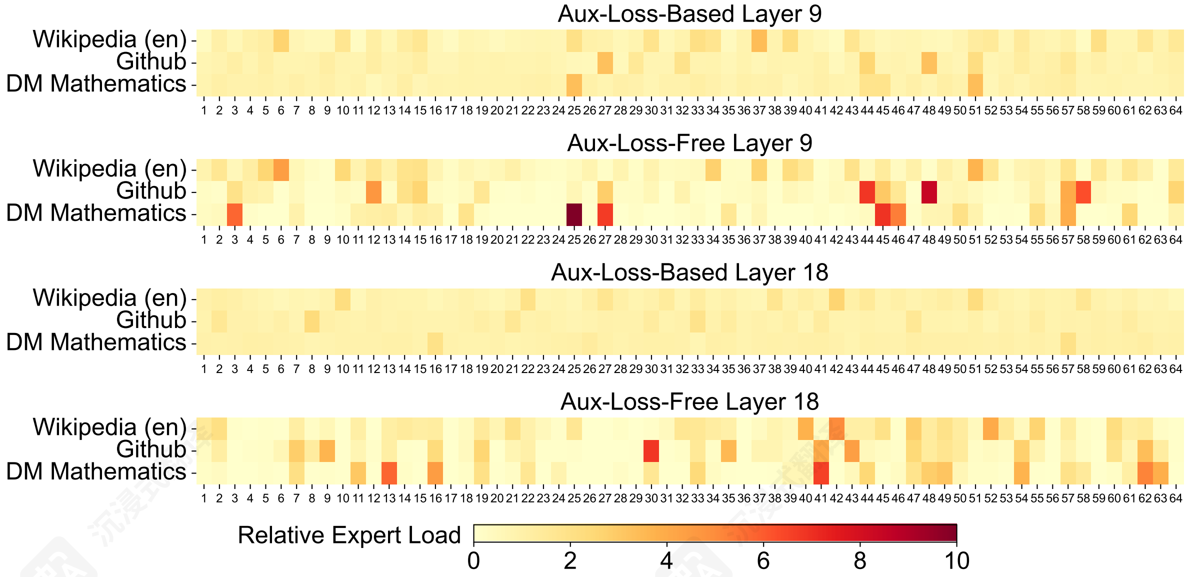


图 9 | 在 Pile 测试集上，无辅助损失和基于辅助损失的模型在三个领域的专家负载。无辅助损失的模型比基于辅助损失的模型表现出更强的专家专业化模式。相对专家负载表示实际专家负载与理论上平衡的专家负载之间的比率。由于空间限制，我们仅以两层的结果为例，所有层的结果在附录 C 中提供。

对于某些序列或小批次，以及 (2) 推理过程中的领域偏移引起的负载不平衡。第一个挑战自然地由我们的训练框架解决，该框架使用大规模专家并行和数据并行，确保每个微批次的较大大小。对于第二个挑战，我们还设计并实现了一个具有冗余专家部署的高效推理框架，如第 3.4 节所述，以克服它。

5. 训练后

5.1. 有监督微调

我们整理我们的指令微调数据集，包含跨越多个领域的 1.5M 个实例，每个领域采用针对其特定需求量身定制的数据创建方法。

推理数据。 对于推理相关数据集，包括那些专注于数学、代码竞赛问题和逻辑谜题的数据集，我们通过利用内部的 DeepSeek-R1 模型来生成数据。具体来说，虽然 R1 生成的数据表现出很强的准确性，但它存在过度思考、格式不佳和长度过长等问题。我们的目标是平衡 R1 生成的推理数据的高准确性和常规格式推理数据的清晰性和简洁性。

为了建立我们的方法论，我们首先使用结合了监督微调（SFT）和强化学习（RL）训练管道，开发一个针对特定领域的专家模型，例如代码、数学或一般推理。这个专家模型作为最终模型的数据生成器。训练过程涉及为每个实例生成两种不同类型的 SFT 样本：第一种将问题与其原始响应以 <问题，原始响应> 的格式配对，第二种则包含系统提示

alongside the problem and the R1 response in the format of <system prompt, problem, R1 response>.

The system prompt is meticulously designed to include instructions that guide the model toward producing responses enriched with mechanisms for reflection and verification. During the RL phase, the model leverages high-temperature sampling to generate responses that integrate patterns from both the R1-generated and original data, even in the absence of explicit system prompts. After hundreds of RL steps, the intermediate RL model learns to incorporate R1 patterns, thereby enhancing overall performance strategically.

Upon completing the RL training phase, we implement rejection sampling to curate high-quality SFT data for the final model, where the expert models are used as data generation sources. This method ensures that the final training data retains the strengths of DeepSeek-R1 while producing responses that are concise and effective.

Non-Reasoning Data. For non-reasoning data, such as creative writing, role-play, and simple question answering, we utilize DeepSeek-V2.5 to generate responses and enlist human annotators to verify the accuracy and correctness of the data.

SFT Settings. We fine-tune DeepSeek-V3-Base for two epochs using the SFT dataset, using the cosine decay learning rate scheduling that starts at 5×10^{-6} and gradually decreases to 1×10^{-6}. During training, each single sequence is packed from multiple samples. However, we adopt a sample masking strategy to ensure that these examples remain isolated and mutually invisible.

5.2. Reinforcement Learning

5.2.1. Reward Model

We employ a rule-based Reward Model (RM) and a model-based RM in our RL process.

Rule-Based RM. For questions that can be validated using specific rules, we adopt a rule-based reward system to determine the feedback. For instance, certain math problems have deterministic results, and we require the model to provide the final answer within a designated format (e.g., in a box), allowing us to apply rules to verify the correctness. Similarly, for LeetCode problems, we can utilize a compiler to generate feedback based on test cases. By leveraging rule-based validation wherever possible, we ensure a higher level of reliability, as this approach is resistant to manipulation or exploitation.

Model-Based RM. For questions with free-form ground-truth answers, we rely on the reward model to determine whether the response matches the expected ground-truth. Conversely, for questions without a definitive ground-truth, such as those involving creative writing, the reward model is tasked with providing feedback based on the question and the corresponding answer as inputs. The reward model is trained from the DeepSeek-V3 SFT checkpoints. To enhance its reliability, we construct preference data that not only provides the final reward but also includes the chain-of-thought leading to the reward. This approach helps mitigate the risk of reward hacking in specific tasks.

与问题及 R1 响应在 <system prompt, problem, R1 response> 格式下一起提供。

系统提示被精心设计，以包含指导模型生成包含反思和验证机制的响应的指令。在 RL 阶段，模型利用高温采样生成响应，这些响应结合了 R1 生成的和原始数据中的模式，即使在缺乏明确系统提示的情况下也是如此。经过数百次 RL 步骤后，中间 RL 模型学会结合 R1 模式，从而战略性地提升整体性能。

完成 RL 训练阶段后，我们实施拒绝采样，为最终模型筛选高质量 SFT 数据，其中专家模型被用作数据生成源。这种方法确保最终训练数据保留了 DeepSeek-R1 的优势，同时生成的响应简洁有效。

非推理数据。 对于非推理数据，例如创意写作、角色扮演和简单问答，我们利用 DeepSeek-V2.5 生成响应，并聘请人工标注员验证数据的准确性和正确性。

SFT 设置。 我们使用 SFT 数据集对 DeepSeek-V3-Base 进行两个 epoch 的微调，采用从 5×10^{-6} 开始并逐渐衰减到 1×10^{-6} 的余弦学习率调度。在训练过程中，每个单序列由多个样本打包而成。然而，我们采用样本掩码策略，以确保这些示例保持隔离且相互不可见。

5.2. 强化学习

5.2.1. 奖励模型

We employ a rul基于 e 的奖励模型 (RM) 和基于模型的 RM in our RL process.

基于规则的 RM。 对于可以使用特定规则进行验证的问题，我们采用基于规则的奖励系统来确定反馈。例如，某些数学问题具有确定性结果，我们要求模型以指定格式（例如，在方框内）提供最终答案，从而允许我们应用规则来验证正确性。类似地，对于 LeetCode 问题，我们可以利用编译器根据测试用例生成反馈。通过尽可能利用基于规则的验证，我们确保了更高的可靠性，因为这种方法对操纵或利用具有抵抗力。

基于模型的RM。 对于具有自由形式ground-truth答案的问题，我们依赖奖励模型来确定响应是否与预期的ground-truth相匹配。相反，对于没有明确ground-truth的问题，例如涉及创意写作的问题，奖励模型的任务是根据问题和相应的答案作为输入来提供反馈。奖励模型是从DeepSeek-V3 SFT检查点训练的。为了提高其可靠性，我们构建了偏好数据，这些数据不仅提供最终奖励，还包括导致奖励的推理链。这种方法有助于降低特定任务中奖励攻击的风险。

5.2.2. Group Relative Policy Optimization

Similar to DeepSeek-V2 (DeepSeek-AI, 2024c), we adopt Group Relative Policy Optimization (GRPO) (Shao et al., 2024), which foregoes the critic model that is typically with the same size as the policy model, and estimates the baseline from group scores instead. Specifically, for each question q, GRPO samples a group of outputs $\{o_1, o_2, \cdots, o_G\}$ from the old policy model $\pi_{\theta_{old}}$ and then optimizes the policy model π_θ by maximizing the following objective:

$$\mathcal{J}_{GRPO}(\theta) = \mathbb{E}[q \sim P(Q), \{o_i\}_{i=1}^{G} \sim \pi_{\theta_{old}}(O|q)]$$
$$\frac{1}{G}\sum_{i=1}^{G}\left(\min\left(\frac{\pi_\theta(o_i|q)}{\pi_{\theta_{old}}(o_i|q)}A_i, \text{clip}\left(\frac{\pi_\theta(o_i|q)}{\pi_{\theta_{old}}(o_i|q)}, 1-\varepsilon, 1+\varepsilon\right)A_i\right) - \beta\mathbb{D}_{KL}\left(\pi_\theta||\pi_{ref}\right)\right), \tag{26}$$

$$\mathbb{D}_{KL}\left(\pi_\theta||\pi_{ref}\right) = \frac{\pi_{ref}(o_i|q)}{\pi_\theta(o_i|q)} - \log\frac{\pi_{ref}(o_i|q)}{\pi_\theta(o_i|q)} - 1, \tag{27}$$

where ε and β are hyper-parameters; π_{ref} is the reference model; and A_i is the advantage, derived from the rewards $\{r_1, r_2, \ldots, r_G\}$ corresponding to the outputs within each group:

$$A_i = \frac{r_i - \text{mean}(\{r_1, r_2, \cdots, r_G\})}{\text{std}(\{r_1, r_2, \cdots, r_G\})}. \tag{28}$$

We incorporate prompts from diverse domains, such as coding, math, writing, role-playing, and question answering, during the RL process. This approach not only aligns the model more closely with human preferences but also enhances performance on benchmarks, especially in scenarios where available SFT data are limited.

5.3. Evaluations

5.3.1. Evaluation Settings

Evaluation Benchmarks. Apart from the benchmark we used for base model testing, we further evaluate instructed models on IFEval (Zhou et al., 2023), FRAMES (Krishna et al., 2024), LongBench v2 (Bai et al., 2024), GPQA (Rein et al., 2023), SimpleQA (OpenAI, 2024c), C-SimpleQA (He et al., 2024), SWE-Bench Verified (OpenAI, 2024d), Aider [1], LiveCodeBench (Jain et al., 2024) (questions from August 2024 to November 2024), Codeforces [2], Chinese National High School Mathematics Olympiad (CNMO 2024)[3], and American Invitational Mathematics Examination 2024 (AIME 2024) (MAA, 2024).

Compared Baselines. We conduct comprehensive evaluations of our chat model against several strong baselines, including DeepSeek-V2-0506, DeepSeek-V2.5-0905, Qwen2.5 72B Instruct, LLaMA-3.1 405B Instruct, Claude-Sonnet-3.5-1022, and GPT-4o-0513. For the DeepSeek-V2 model series, we select the most representative variants for comparison. For closed-source models, evaluations are performed through their respective APIs.

Detailed Evaluation Configurations. For standard benchmarks including MMLU, DROP, GPQA, and SimpleQA, we adopt the evaluation prompts from the simple-evals framework[4].

[1] https://aider.chat
[2] https://codeforces.com
[3] https://www.cms.org.cn/Home/comp/comp/cid/12.html
[4] https://github.com/openai/simple-evals

5.2.2. 组相对策略优化

与 DeepSeek-V2 (DeepSeek-AI, 2024c) 类似，我们采用组相对策略优化 (GRPO) (Shao et al., 2024)，该优化方法放弃了与策略模型大小相同的评价模型，而是从组分数中估计基线。具体来说，对于每个问题 q，GRPO 从旧策略模型$\pi_{\theta_{old}}$ 中采样一组输出 $\{o_1, o_2, \cdots, o_G\}$，然后通过最大化以下目标来优化策略模型$\pi_\theta$：

$$\mathcal{J}_{GRPO}(\theta) = \mathbb{E}[q \sim P(Q), \{o_i\}_{i=1}^{G} \sim \pi_{\theta_{old}}(O|q)]$$
$$\frac{1}{G}\sum_{i=1}^{G}\left(\min\left(\frac{\pi_\theta(o_i|q)}{\pi_{\theta_{old}}(o_i|q)}A_i, \text{clip}\left(\frac{\pi_\theta(o_i|q)}{\pi_{\theta_{old}}(o_i|q)}, 1-\varepsilon, 1+\varepsilon\right)A_i\right) - \beta\mathbb{D}_{KL}\left(\pi_\theta||\pi_{ref}\right)\right), \tag{26}$$

$$\mathbb{D}_{KL}\left(\pi_\theta||\pi_{ref}\right) = \frac{\pi_{ref}(o_i|q)}{\pi_\theta(o_i|q)} - \log\frac{\pi_{ref}(o_i|q)}{\pi_\theta(o_i|q)} - 1, \tag{27}$$

其中 ε 和 β 是超参数；π_{ref} 是参考模型；而 A_i 是优势，它来自与每个组内的输出对应的奖励 $\{r_1, r_2, \ldots, r_G\}$:

$$A_i = \frac{r_i - \text{mean}(\{r_1, r_2, \cdots, r_G\})}{\text{std}(\{r_1, r_2, \cdots, r_G\})}. \tag{28}$$

我们在 RL 过程中结合了来自不同领域的提示，例如编码、数学、写作、角色扮演和问答。这种方法不仅使模型更紧密地与人类偏好保持一致，还提高了基准测试的性能，尤其是在可用 SFT 数据有限的情况下。

5.3. 评估

5.3.1. 评估设置

评估基准。除了我们用于基础模型测试的基准外，我们还进一步在 IFEval (Zhou et al., 2023)、FRAMES (Krishna et al., 2024)、LongBench v2 (Bai et al., 2024)、GPQA (Rein et al., 2023)、SimpleQA (OpenAI, 2024c)、C- SimpleQA (He et al., 2024)、SWE-Bench Verified (OpenAI, 2024d)、Aider 1、LiveCodeBench (Jain et al., 2024)（问题来自2024年8月至11月）、Codeforces 2、中国高中数学奥林匹克 (CNMO 2024)3，以及美国邀请数学考试 2024 (AIME 2024) (MAA, 2024) 上评估指令模型。

与基线比较。我们将我们的聊天模型与几个强大的基线进行了全面评估，包括 DeepSeek-V2-0506、DeepSeek-V2.5-0905、Qwen2.5 72B Instruct、LLaMA-3.1 405B Instruct、Claude-Sonnet-3.5-1022 和 GPT-4o-0513。对于 DeepSeek-V2 模型系列，我们选择了最具代表性的变体进行比较。对于闭源模型，评估通过其各自的 API 进行。

详细评估配置。对于包括 MMLU、DROP、GPQA 和 SimpleQA 在内的标准基准测试，我们采用来自 thesimple-evals 框架4 的评估提示。

[1] https://aider.chat
[2] https://codeforces.com
[3] https://www.cms.org.cn/Home/comp/comp/cid/12.html
[4] https://github.com/openai/simple-evals

We utilize the Zero-Eval prompt format (Lin, 2024) for MMLU-Redux in a zero-shot setting. For other datasets, we follow their original evaluation protocols with default prompts as provided by the dataset creators. For code and math benchmarks, the HumanEval-Mul dataset includes 8 mainstream programming languages (Python, Java, Cpp, C#, JavaScript, TypeScript, PHP, and Bash) in total. We use CoT and non-CoT methods to evaluate model performance on LiveCodeBench, where the data are collected from August 2024 to November 2024. The Codeforces dataset is measured using the percentage of competitors. SWE-Bench verified is evaluated using the agentless framework (Xia et al., 2024). We use the "diff" format to evaluate the Aider-related benchmarks. For mathematical assessments, AIME and CNMO 2024 are evaluated with a temperature of 0.7, and the results are averaged over 16 runs, while MATH-500 employs greedy decoding. We allow all models to output a maximum of 8192 tokens for each benchmark.

	Benchmark (Metric)	**DeepSeek V2-0506**	**DeepSeek V2.5-0905**	**Qwen2.5 72B-Inst.**	**LLaMA-3.1 405B-Inst.**	**Claude-3.5-Sonnet-1022**	**GPT-4o 0513**	**DeepSeek V3**
	Architecture	MoE	MoE	Dense	Dense	-	-	MoE
	# Activated Params	21B	21B	72B	405B	-	-	37B
	# Total Params	236B	236B	72B	405B	-	-	671B
English	MMLU (EM)	78.2	80.6	85.3	**88.6**	**88.3**	87.2	**88.5**
	MMLU-Redux (EM)	77.9	80.3	85.6	86.2	**88.9**	88.0	**89.1**
	MMLU-Pro (EM)	58.5	66.2	71.6	73.3	**78.0**	72.6	75.9
	DROP (3-shot F1)	83.0	87.8	76.7	88.7	88.3	83.7	**91.6**
	IF-Eval (Prompt Strict)	57.7	80.6	84.1	86.0	**86.5**	84.3	86.1
	GPQA-Diamond (Pass@1)	35.3	41.3	49.0	51.1	**65.0**	49.9	59.1
	SimpleQA (Correct)	9.0	10.2	9.1	17.1	28.4	**38.2**	24.9
	FRAMES (Acc.)	66.9	65.4	69.8	70.0	72.5	**80.5**	73.3
	LongBench v2 (Acc.)	31.6	35.4	39.4	36.1	41.0	48.1	**48.7**
Code	HumanEval-Mul (Pass@1)	69.3	77.4	77.3	77.2	81.7	80.5	**82.6**
	LiveCodeBench (Pass@1-COT)	18.8	29.2	31.1	28.4	36.3	33.4	**40.5**
	LiveCodeBench (Pass@1)	20.3	28.4	28.7	30.1	32.8	34.2	**37.6**
	Codeforces (Percentile)	17.5	35.6	24.8	25.3	20.3	23.6	**51.6**
	SWE Verified (Resolved)	-	22.6	23.8	24.5	**50.8**	38.8	42.0
	Aider-Edit (Acc.)	60.3	71.6	65.4	63.9	**84.2**	72.9	79.7
	Aider-Polyglot (Acc.)	-	18.2	7.6	5.8	45.3	16.0	**49.6**
Math	AIME 2024 (Pass@1)	4.6	16.7	23.3	23.3	16.0	9.3	**39.2**
	MATH-500 (EM)	56.3	74.7	80.0	73.8	78.3	74.6	**90.2**
	CNMO 2024 (Pass@1)	2.8	10.8	15.9	6.8	13.1	10.8	**43.2**
Chinese	CLUEWSC (EM)	89.9	90.4	**91.4**	84.7	85.4	87.9	90.9
	C-Eval (EM)	78.6	79.5	86.1	61.5	76.7	76.0	**86.5**
	C-SimpleQA (Correct)	48.5	54.1	48.4	50.4	51.3	59.3	**64.8**

Table 6 | Comparison between DeepSeek-V3 and other representative chat models. All models are evaluated in a configuration that limits the output length to 8K. Benchmarks containing fewer than 1000 samples are tested multiple times using varying temperature settings to derive robust final results. DeepSeek-V3 stands as the best-performing open-source model, and also exhibits competitive performance against frontier closed-source models.

5.3.2. Standard Evaluation

Table 6 presents the evaluation results, showcasing that DeepSeek-V3 stands as the best-performing open-source model. Additionally, it is competitive against frontier closed-source models like GPT-4o and Claude-3.5-Sonnet.

我们使用 Zero-Eval 提示格式（Lin, 2024）在零样本设置下对 MMLU-Redux 进行评估。对于其他数据集，我们遵循其原始评估协议，并使用数据集创建者提供的默认提示。对于代码和数学基准，HumanEval-Mul 数据集总共包含 8 种主流编程语言（Python、Java、Cpp、C#、JavaScript、TypeScript、PHP 和 Bash）。我们使用 CoT 和非 CoT 方法在 LiveCodeBench 上评估模型性能，其中数据收集时间从 2024 年 8 月到 2024 年 11 月。Codeforces 数据集使用参赛者的百分比进行衡量。SWE-Bench verified 使用无代理框架（Xia 等人，2024）进行评估。我们使用“diff”格式评估与 Aider 相关的基准。对于数学评估，AIME 和 CNMO 2024 使用 0.7 的温度进行评估，结果在 16 次运行中取平均值，而 MATH-500 采用贪婪解码。我们允许所有模型在每个基准下输出最多 8192 个 token。

	基准 (Metric)	**DeepSeek V2-0506**	**DeepSeek V2.5-0905**	**Qwen2.5 72B-Inst.**	**LLaMA-3.1 405B-Inst.**	**Claude-3.5- Sonnet-1022**	**GPT-4o 0513**	**DeepSeek V3**
	架构	MoE	MoE	密集	密集	-	-	MoE
	# 激活参数	21B	21B	72B	405B	-	-	37B
	# 总参数	236B	236B	72B	405B	-	-	671B
英语	MMLU (EM)	78.2	80.6	85.3	**88.6**	**88.3**	87.2	**88.5**
	MMLU-Redux (EM)	77.9	80.3	85.6	86.2	**88.9**	88.0	**89.1**
	MMLU-Pro (EM)	58.5	66.2	71.6	73.3	**78.0**	72.6	75.9
	DROP (3-shot F1)	83.0	87.8	76.7	88.7	88.3	83.7	**91.6**
	IF-Eval (Prompt Strict)	57.7	80.6	84.1	86.0	**86.5**	84.3	86.1
	GPQA-Diamond (Pass@1)	35.3	41.3	49.0	51.1	**65.0**	49.9	59.1
	SimpleQA (Correct)	9.0	10.2	9.1	17.1	28.4	**38.2**	24.9
	FRAMES (Acc.)	66.9	65.4	69.8	70.0	72.5	**80.5**	73.3
	LongBench v2 (Acc.)	31.6	35.4	39.4	36.1	41.0	48.1	**48.7**
Code	HumanEval-Mul (Pass@1)	69.3	77.4	77.3	77.2	81.7	80.5	**82.6**
	LiveCodeBench (Pass@1-COT)	18.8	29.2	31.1	28.4	36.3	33.4	**40.5**
	LiveCodeBench (Pass@1)	20.3	28.4	28.7	30.1	32.8	34.2	**37.6**
	Codeforces (Percentile)	17.5	35.6	24.8	25.3	20.3	23.6	**51.6**
	SWE Verified (Resolved)	-	22.6	23.8	24.5	**50.8**	38.8	42.0
	Aider-Edit (Acc.)	60.3	71.6	65.4	63.9	**84.2**	72.9	79.7
	Aider-Polyglot (Acc.)	-	18.2	7.6	5.8	45.3	16.0	**49.6**
Math	AIME 2024 (Pass@1)	4.6	16.7	23.3	23.3	16.0	9.3	**39.2**
	MATH-500 (EM)	56.3	74.7	80.0	73.8	78.3	74.6	**90.2**
	CNMO 2024 (Pass@1)	2.8	10.8	15.9	6.8	13.1	10.8	**43.2**
中文	CLUEWSC (EM)	89.9	90.4	**91.4**	84.7	85.4	87.9	90.9
	C-Eval (EM)	78.6	79.5	86.1	61.5	76.7	76.0	**86.5**
	C-SimpleQA (Correct)	48.5	54.1	48.4	50.4	51.3	59.3	**64.8**

表 6 | DeepSeek-V3 与其他代表性聊天模型的对比。所有模型均在输出长度限制为 8K 的配置下进行评估。包含少于 1000 个样本的基准测试使用不同的温度设置进行多次测试，以得出稳健的最终结果。DeepSeek-V3 是表现最佳的开源模型，同时在性能上也与前沿闭源模型具有竞争力。

5.3.2. 标准评估

表6展示了评估结果，表明DeepSeek-V3是表现最佳的开放源模型。此外，它在与GPT-4o和Claude-3.5-Sonnet等前沿闭源模型相比中具有竞争力。

English Benchmarks. MMLU is a widely recognized benchmark designed to assess the performance of large language models, across diverse knowledge domains and tasks. DeepSeek-V3 demonstrates competitive performance, standing on par with top-tier models such as LLaMA-3.1-405B, GPT-4o, and Claude-Sonnet 3.5, while significantly outperforming Qwen2.5 72B. Moreover, DeepSeek-V3 excels in MMLU-Pro, a more challenging educational knowledge benchmark, where it closely trails Claude-Sonnet 3.5. On MMLU-Redux, a refined version of MMLU with corrected labels, DeepSeek-V3 surpasses its peers. In addition, on GPQA-Diamond, a PhD-level evaluation testbed, DeepSeek-V3 achieves remarkable results, ranking just behind Claude 3.5 Sonnet and outperforming all other competitors by a substantial margin.

In long-context understanding benchmarks such as DROP, LongBench v2, and FRAMES, DeepSeek-V3 continues to demonstrate its position as a top-tier model. It achieves an impressive 91.6 F1 score in the 3-shot setting on DROP, outperforming all other models in this category. On FRAMES, a benchmark requiring question-answering over 100k token contexts, DeepSeek-V3 closely trails GPT-4o while outperforming all other models by a significant margin. This demonstrates the strong capability of DeepSeek-V3 in handling extremely long-context tasks. The long-context capability of DeepSeek-V3 is further validated by its best-in-class performance on LongBench v2, a dataset that was released just a few weeks before the launch of DeepSeek V3. On the factual knowledge benchmark, SimpleQA, DeepSeek-V3 falls behind GPT-4o and Claude-Sonnet, primarily due to its design focus and resource allocation. DeepSeek-V3 assigns more training tokens to learn Chinese knowledge, leading to exceptional performance on the C-SimpleQA. On the instruction-following benchmark, DeepSeek-V3 significantly outperforms its predecessor, DeepSeek-V2-series, highlighting its improved ability to understand and adhere to user-defined format constraints.

Code and Math Benchmarks. Coding is a challenging and practical task for LLMs, encompassing engineering-focused tasks like SWE-Bench-Verified and Aider, as well as algorithmic tasks such as HumanEval and LiveCodeBench. In engineering tasks, DeepSeek-V3 trails behind Claude-Sonnet-3.5-1022 but significantly outperforms open-source models. The open-source DeepSeek-V3 is expected to foster advancements in coding-related engineering tasks. By providing access to its robust capabilities, DeepSeek-V3 can drive innovation and improvement in areas such as software engineering and algorithm development, empowering developers and researchers to push the boundaries of what open-source models can achieve in coding tasks. In algorithmic tasks, DeepSeek-V3 demonstrates superior performance, outperforming all baselines on benchmarks like HumanEval-Mul and LiveCodeBench. This success can be attributed to its advanced knowledge distillation technique, which effectively enhances its code generation and problem-solving capabilities in algorithm-focused tasks.

On math benchmarks, DeepSeek-V3 demonstrates exceptional performance, significantly surpassing baselines and setting a new state-of-the-art for non-o1-like models. Specifically, on AIME, MATH-500, and CNMO 2024, DeepSeek-V3 outperforms the second-best model, Qwen2.5 72B, by approximately 10% in absolute scores, which is a substantial margin for such challenging benchmarks. This remarkable capability highlights the effectiveness of the distillation technique from DeepSeek-R1, which has been proven highly beneficial for non-o1-like models.

Chinese Benchmarks. Qwen and DeepSeek are two representative model series with robust support for both Chinese and English. On the factual benchmark Chinese SimpleQA, DeepSeek-V3 surpasses Qwen2.5-72B by 16.4 points, despite Qwen2.5 being trained on a larger corpus compromising 18T tokens, which are 20% more than the 14.8T tokens that DeepSeek-V3 is

英文基准测试。 MMLU 是一个被广泛认可的基准测试，旨在评估大型语言模型在多样化知识领域和任务中的表现。DeepSeek-V3 表现出具有竞争力的性能，与 LLaMA-3.1-405B、GPT-4o 和 Claude-Sonnet 3.5 等顶级模型相当，同时显著优于 Qwen2.5 72B。此外，DeepSeek-V3 在更具挑战性的教育知识基准测试 MMLU-Pro 中表现出色，仅次于 Claude-Sonnet 3.5。在 MMLU-Redux（一个标签已修正的 MMLU 精炼版本）中，DeepSeek-V3 超过了其他同行。此外，在 GPQA-Diamond（一个博士级别的评估测试平台）中，DeepSeek-V3 取得了卓越的成果，仅次于 Claude 3.5 Sonnet，并大幅领先所有其他竞争对手。

在 DROP、LongBench v2 和 FRAMES 等长文本理解基准测试中，DeepSeek-V3 继续展现出其作为顶级模型的地位。在 DROP 的 3-shot 设置中，DeepSeek-V3 实现了令人印象深刻的 91.6 F1 分数，优于该类别中的所有其他模型。在 FRAMES（一个要求在 100k 个 token 的上下文中进行问答的基准测试）中，DeepSeek-V3 仅次于 GPT-4o，但大幅优于其他所有模型。这展示了 DeepSeek-V3 在处理极长文本任务方面的强大能力。DeepSeek-V3 的长文本能力通过其在 LongBench v2 上的最佳表现得到进一步验证，该数据集是在 DeepSeek V3 发布前几周发布的。在事实知识基准测试 SimpleQA 中，DeepSeek-V3 仅次于 GPT-4o 和 Claude-Sonnet，这主要是由于其设计重点和资源分配。DeepSeek-V3 将更多训练 token 分配给学习中文知识，导致在 C-SimpleQA 上表现优异。在指令遵循基准测试中，DeepSeek-V3 显著优于其前身 DeepSeek-V2-series，突显了其理解和遵循用户定义格式约束能力的提升。

代码和数学基准。 编程对大型语言模型（LLM）来说是一项具有挑战性和实践性的任务，涵盖了以工程为导向的任务（如SWE-Bench-Verified和Aider）以及算法任务（如HumanEval和LiveCodeBench）。在工程任务中，DeepSeek-V3落后于Claude-Sonnet-3.5-1022，但显著优于开源模型。开源的DeepSeek-V3有望推动与编程相关的工程任务的进步。通过提供对其强大能力的访问，DeepSeek-V3可以在软件工程和算法开发等领域推动创新和改进，使开发者和研究人员能够拓展开源模型在编程任务中能够达到的边界。在算法任务中，DeepSeek-V3表现出卓越的性能，在HumanEval-Mul和LiveCodeBench等基准测试中优于所有基线。这一成功归因于其先进的知识蒸馏技术，该技术有效地增强了其在算法导向任务中的代码生成和问题解决能力。

在数学基准测试中，DeepSeek-V3表现出色，显著超越了基线，为非o1-like模型设定了新的SOTA。具体来说，在AIME、MATH-500和CNMO 2024上，DeepSeek-V3在绝对分数上比第二好的模型Qwen2.5 72B高出约10%，对于如此具有挑战性的基准测试来说，这是一个相当大的差距。这种卓越的能力突出了DeepSeek-R1的蒸馏技术的有效性，该技术已被证明对非o1-like模型非常有益。

中文基准测试。 Qwen和DeepSeek是两个具有强大中文和英文支持的代表模型系列。在事实基准测试Chinese SimpleQA上，尽管Qwen2.5在包含18T tokens（比DeepSeek-V3的14.8T tokens多20%）的大型语料库上进行了训练，但DeepSeek-V3仍然比Qwen2.5-72B高出16.4分。

Model	Arena-Hard	AlpacaEval 2.0
DeepSeek-V2.5-0905	76.2	50.5
Qwen2.5-72B-Instruct	81.2	49.1
LLaMA-3.1 405B	69.3	40.5
GPT-4o-0513	80.4	51.1
Claude-Sonnet-3.5-1022	85.2	52.0
DeepSeek-V3	**85.5**	**70.0**

Table 7 | English open-ended conversation evaluations. For AlpacaEval 2.0, we use the length-controlled win rate as the metric.

pre-trained on.

On C-Eval, a representative benchmark for Chinese educational knowledge evaluation, and CLUEWSC (Chinese Winograd Schema Challenge), DeepSeek-V3 and Qwen2.5-72B exhibit similar performance levels, indicating that both models are well-optimized for challenging Chinese-language reasoning and educational tasks.

5.3.3. *Open-Ended Evaluation*

In addition to standard benchmarks, we also evaluate our models on open-ended generation tasks using LLMs as judges, with the results shown in Table 7. Specifically, we adhere to the original configurations of AlpacaEval 2.0 (Dubois et al., 2024) and Arena-Hard (Li et al., 2024a), which leverage GPT-4-Turbo-1106 as judges for pairwise comparisons. On Arena-Hard, DeepSeek-V3 achieves an impressive win rate of over 86% against the baseline GPT-4-0314, performing on par with top-tier models like Claude-Sonnet-3.5-1022. This underscores the robust capabilities of DeepSeek-V3, especially in dealing with complex prompts, including coding and debugging tasks. Furthermore, DeepSeek-V3 achieves a groundbreaking milestone as the first open-source model to surpass 85% on the Arena-Hard benchmark. This achievement significantly bridges the performance gap between open-source and closed-source models, setting a new standard for what open-source models can accomplish in challenging domains.

Similarly, DeepSeek-V3 showcases exceptional performance on AlpacaEval 2.0, outperforming both closed-source and open-source models. This demonstrates its outstanding proficiency in writing tasks and handling straightforward question-answering scenarios. Notably, it surpasses DeepSeek-V2.5-0905 by a significant margin of 20%, highlighting substantial improvements in tackling simple tasks and showcasing the effectiveness of its advancements.

5.3.4. *DeepSeek-V3 as a Generative Reward Model*

We compare the judgment ability of DeepSeek-V3 with state-of-the-art models, namely GPT-4o and Claude-3.5. Table 8 presents the performance of these models in RewardBench (Lambert et al., 2024). DeepSeek-V3 achieves performance on par with the best versions of GPT-4o-0806 and Claude-3.5-Sonnet-1022, while surpassing other versions. Additionally, the judgment ability of DeepSeek-V3 can also be enhanced by the voting technique. Therefore, we employ DeepSeek-V3 along with voting to offer self-feedback on open-ended questions, thereby improving the effectiveness and robustness of the alignment process.

模型	Arena-Hard	AlpacaEval 2.0
DeepSeek-V2.5-0905	76.2	50.5
Qwen2.5-72B-Instruct	81.2	49.1
LLaMA-3.1 405B	69.3	40.5
GPT-4o-0513	80.4	51.1
Claude-Sonnet-3.5-1022	85.2	52.0
DeepSeek-V3	**85.5**	**70.0**

Table 7 | English open-ended conversation evaluations. For AlpacaEval 2.0, we use the length-controlled win rate as the metric.

pre-trained on.

在 C-Eval（中文教育知识评估的代表性基准）和 CLUEWSC（中文 Winograd Sc hema 挑战赛）上，DeepSeek-V3 和 Qwen2.5-72B 表现相似，表明这两个模型都针对具有挑战性的中文语言推理和教育任务进行了优化。

5.3.3. 开放式评估

除了标准基准外，我们还使用 LLM 作为裁判器，在开放式生成任务上评估我们的模型，结果如表 7 所示。具体来说，我们遵循 AlpacaEval 2.0（Dubois 等人，2024）和 Arena-Hard（Li 等人，2024a）的原始配置，后者使用 GPT-4-Turbo-1106 作为成对比较的裁判器。在 Arena-Hard 上，DeepSeek-V3 对基线 GPT-4-0314 的胜率超过 86%，表现与顶级模型如 Claude-Sonnet-3.5-1022 相当。这突显了 DeepSeek-V3 的强大能力，尤其是在处理复杂提示（包括编码和调试任务）方面。此外，DeepSeek-V3 成为首个在 Arena-Hard 基准上胜率超过 85% 的开源模型，这一成就显著缩小了开源模型与闭源模型之间的性能差距，为开源模型在具有挑战性领域的表现设定了新标准。

同样地，DeepSeek-V3 在 AlpacaEval 2.0 上表现出色，超越了闭源和开源模型。这展示了其在写作任务和处理简单问答场景方面的卓越能力。值得注意的是，它比 DeepSeek-V2.5-0905 提高了 20% 的显著差距，突出了其在处理简单任务方面的巨大改进及其进步的有效性。

5.3.4. DeepSeek-V3 作为生成式奖励模型

我们将 DeepSeek-V3 与最先进的模型（即 GPT-4o 和 Claude-3.5）的判断能力进行比较。表 8 展示了这些模型在 RewardBench（Lambert 等人，2024 年）中的性能。DeepSeek-V3 的性能与 GPT-4o-0806 和 Claude-3.5-Sonnet-1022 的最佳版本相当，同时超越了其他版本。此外，DeepSeek-V3 的判断能力也可以通过投票技术来增强。因此，我们采用 DeepSeek-V3 和投票技术来对开放式问题进行自我反馈，从而提高对齐过程的有效性和鲁棒性。

Model	Chat	Chat-Hard	Safety	Reasoning	Average
GPT-4o-0513	96.6	70.4	86.7	84.9	84.7
GPT-4o-0806	96.1	76.1	88.1	86.6	86.7
GPT-4o-1120	95.8	71.3	86.2	85.2	84.6
Claude-3.5-sonnet-0620	96.4	74.0	81.6	84.7	84.2
Claude-3.5-sonnet-1022	96.4	79.7	91.1	87.6	88.7
DeepSeek-V3	96.9	79.8	87.0	84.3	87.0
DeepSeek-V3 (maj@6)	96.9	82.6	89.5	89.2	89.6

Table 8 | Performances of GPT-4o, Claude-3.5-sonnet and DeepSeek-V3 on RewardBench.

Model	LiveCodeBench-CoT		MATH-500	
	Pass@1	Length	Pass@1	Length
DeepSeek-V2.5 Baseline	31.1	718	74.6	769
DeepSeek-V2.5 +R1 Distill	37.4	783	83.2	1510

Table 9 | The contribution of distillation from DeepSeek-R1. The evaluation settings of LiveCodeBench and MATH-500 are the same as in Table 6.

5.4. Discussion

5.4.1. Distillation from DeepSeek-R1

We ablate the contribution of distillation from DeepSeek-R1 based on DeepSeek-V2.5. The baseline is trained on short CoT data, whereas its competitor uses data generated by the expert checkpoints described above.

Table 9 demonstrates the effectiveness of the distillation data, showing significant improvements in both LiveCodeBench and MATH-500 benchmarks. Our experiments reveal an interesting trade-off: the distillation leads to better performance but also substantially increases the average response length. To maintain a balance between model accuracy and computational efficiency, we carefully selected optimal settings for DeepSeek-V3 in distillation.

Our research suggests that knowledge distillation from reasoning models presents a promising direction for post-training optimization. While our current work focuses on distilling data from mathematics and coding domains, this approach shows potential for broader applications across various task domains. The effectiveness demonstrated in these specific areas indicates that long-CoT distillation could be valuable for enhancing model performance in other cognitive tasks requiring complex reasoning. Further exploration of this approach across different domains remains an important direction for future research.

5.4.2. Self-Rewarding

Rewards play a pivotal role in RL, steering the optimization process. In domains where verification through external tools is straightforward, such as some coding or mathematics scenarios, RL demonstrates exceptional efficacy. However, in more general scenarios, constructing a feedback mechanism through hard coding is impractical. During the development of DeepSeek-V3, for these broader contexts, we employ the constitutional AI approach (Bai et al., 2022), leveraging the voting evaluation results of DeepSeek-V3 itself as a feedback source. This method has

模型	Chat	Chat-Hard	安全	推理	平均
GPT-4o-0513	96.6	70.4	86.7	84.9	84.7
GPT-4o-0806	96.1	76.1	88.1	86.6	86.7
GPT-4o-1120	95.8	71.3	86.2	85.2	84.6
Claude-3.5-sonnet-0620	96.4	74.0	81.6	84.7	84.2
Claude-3.5-sonnet-1022	96.4	79.7	91.1	87.6	88.7
DeepSeek-V3	96.9	79.8	87.0	84.3	87.0
DeepSeek-V3 (maj@6)	96.9	82.6	89.5	89.2	89.6

表 8 | GPT-4o、Claude-3.5-sonnet 和 DeepSeek-V3 在 RewardBench 上的性能表现。

模型	LiveCodeBench-CoT		MATH-500	
	Pass@1	长度	Pass@1	长度
DeepSeek-V2.5 基线	31.1	718	74.6	769
DeepSeek-V2.5 +R1 蒸馏	37.4	783	83.2	1510

Table 9 | The contribution of distillation from DeepSeek-R1. The evaluation settings of LiveCodeBench and MATH-500 are the same as in Table 6.

5.4. 讨论

5.4.1. 从 ***DeepSeek-R1*** 蒸馏

我们基于 DeepSeek-V2.5 剔除了 DeepSeek-R1 蒸馏的贡献。基线模型在短 CoT 数据上进行训练，而其竞争对手则使用上述专家检查点生成数据。

表 9 展示了蒸馏数据的有效性，在 LiveCodeBench 和 MATH-500 基准测试中均显著提升了性能。我们的实验揭示了一个有趣的权衡：蒸馏虽然带来了更好的性能，但也显著增加了平均响应长度。为了在模型准确性和计算效率之间保持平衡，我们为 DeepSeek-V3 在蒸馏过程中精心选择了最优设置。

我们的研究表明，从推理模型中进行知识蒸馏为后训练优化提供了一个有前景的方向。尽管我们当前的工作专注于从数学和编码领域蒸馏数据，但这种方法显示出在各个任务领域进行更广泛应用的潜力。在这些特定领域所展示的有效性表明，长 CoT 蒸馏可能对增强需要复杂推理的其他认知任务的模型性能具有重要价值。未来研究的一个重要方向是在不同领域进一步探索这种方法。

5.4.2. 自我奖励

奖励在强化学习中起着关键作用，引导优化过程。在外部工具验证简单的领域，例如某些编码或数学场景，强化学习表现出色。然而，在更一般的情况下，通过硬编码构建反馈机制是不切实际的。在DeepSeek-V3的开发过程中，针对这些更广泛的背景，我们采用了宪法人工智能方法（Bai等人，2022年），利用DeepSeek-V3自身的投票评估结果作为反馈来源。这种方法有

produced notable alignment effects, significantly enhancing the performance of DeepSeek-V3 in subjective evaluations. By integrating additional constitutional inputs, DeepSeek-V3 can optimize towards the constitutional direction. We believe that this paradigm, which combines supplementary information with LLMs as a feedback source, is of paramount importance. The LLM serves as a versatile processor capable of transforming unstructured information from diverse scenarios into rewards, ultimately facilitating the self-improvement of LLMs. Beyond self-rewarding, we are also dedicated to uncovering other general and scalable rewarding methods to consistently advance the model capabilities in general scenarios.

5.4.3. Multi-Token Prediction Evaluation

Instead of predicting just the next single token, DeepSeek-V3 predicts the next 2 tokens through the MTP technique. Combined with the framework of speculative decoding (Leviathan et al., 2023; Xia et al., 2023), it can significantly accelerate the decoding speed of the model. A natural question arises concerning the acceptance rate of the additionally predicted token. Based on our evaluation, the acceptance rate of the second token prediction ranges between 85% and 90% across various generation topics, demonstrating consistent reliability. This high acceptance rate enables DeepSeek-V3 to achieve a significantly improved decoding speed, delivering 1.8 times TPS (Tokens Per Second).

6. Conclusion, Limitations, and Future Directions

In this paper, we introduce DeepSeek-V3, a large MoE language model with 671B total parameters and 37B activated parameters, trained on 14.8T tokens. In addition to the MLA and DeepSeekMoE architectures, it also pioneers an auxiliary-loss-free strategy for load balancing and sets a multi-token prediction training objective for stronger performance. The training of DeepSeek-V3 is cost-effective due to the support of FP8 training and meticulous engineering optimizations. The post-training also makes a success in distilling the reasoning capability from the DeepSeek-R1 series of models. Comprehensive evaluations demonstrate that DeepSeek-V3 has emerged as the strongest open-source model currently available, and achieves performance comparable to leading closed-source models like GPT-4o and Claude-3.5-Sonnet. Despite its strong performance, it also maintains economical training costs. It requires only 2.788M H800 GPU hours for its full training, including pre-training, context length extension, and post-training.

While acknowledging its strong performance and cost-effectiveness, we also recognize that DeepSeek-V3 has some limitations, especially on the deployment. Firstly, to ensure efficient inference, the recommended deployment unit for DeepSeek-V3 is relatively large, which might pose a burden for small-sized teams. Secondly, although our deployment strategy for DeepSeek-V3 has achieved an end-to-end generation speed of more than two times that of DeepSeek-V2, there still remains potential for further enhancement. Fortunately, these limitations are expected to be naturally addressed with the development of more advanced hardware.

DeepSeek consistently adheres to the route of open-source models with longtermism, aiming to steadily approach the ultimate goal of AGI (Artificial General Intelligence). In the future, we plan to strategically invest in research across the following directions.

- We will consistently study and refine our model architectures, aiming to further improve both the training and inference efficiency, striving to approach efficient support for infinite context length. Additionally, we will try to break through the architectural limitations of Transformer, thereby pushing the boundaries of its modeling capabilities.

产生了显著的对齐效果，显著提升了 DeepSeek-V3 在主观评估中的性能。通过整合额外的宪法输入，DeepSeek-V3 可以优化向宪法方向。我们相信，这种将补充信息与 LLM 结合作为反馈源的范式至关重要。LLM 是一个多功能的处理器，能够将来自不同场景的非结构化信息转化为奖励，最终促进 LLM 的自我改进。除了自我奖励之外，我们还致力于发现其他通用且可扩展的奖励方法，以持续提升模型在一般场景中的能力。

5.4.3. 多 *token* 预测评估

DeepSeek-V3 不是仅预测下一个单个 token，而是通过 MTP 技术预测下一个 2 个 token。结合推测解码框架（Leviathan 等人，2023；Xia 等人，2023），它可以显著加速模型的解码速度。一个自然的问题是关于额外预测 token 的接受率。根据我们的评估，第二 token 预测的接受率在各种生成主题中范围为 85% 到 90%，显示出一致的可靠性。这种高接受率使 DeepSeek-V3 能够实现显著提升的解码速度，提供 1.8 倍 TPS（每秒 token 数）。

6. Conclusion, Limitations, and Future Directions

在本文中，我们介绍了 DeepSeek-V3，一个拥有 671B 总参数量和 37B 激活参数的大型 MoE 语言模型，该模型在 14.8T token 上进行训练。除了 MLA 和 DeepSeekMoE 架构外，它还开创了一种无辅助损失的负载均衡策略，并设定了多 token 预测训练目标以实现更强的性能。由于支持 FP8 训练和细致的工程优化，DeepSeek-V3 的训练具有成本效益。在训练后，它还成功地将推理能力从 DeepSeek-R1 系列模型中蒸馏出来。综合评估表明，DeepSeek-V3 已经成为目前最强的开源模型，其性能与 GPT-4o 和 Claude-3.5-Sonnet 等领先的闭源模型相当。尽管性能强大，它仍然保持了经济的训练成本。它的完整训练（包括预训练、上下文长度扩展和训练后）仅需 2.788M H800 GPU 小时。

虽然承认其性能强大且成本效益高，我们也认识到 DeepSeek-V3 存在一些局限性，尤其是在部署方面。首先，为了确保高效的推理，DeepSeek-V3 推荐的部署单元相对较大，这可能给小型团队带来负担。其次，尽管我们的 DeepSeek-V3 部署策略已实现比 DeepSeek-V2 高两倍以上的端到端生成速度，但仍存在进一步改进的潜力。幸运的是，随着更先进硬件的发展，这些局限性预计将自然得到解决。

DeepSeek 始终坚持具有长远主义的开源模型路线，旨在稳步接近 AGI（通用人工智能）的最终目标。未来，我们计划在以下方向进行战略投资。

- 我们将持续研究和优化我们的模型架构，旨在进一步提高训练和推理效率，力求实现对无限上下文长度的有效支持。此外，我们将尝试突破 Transformer 的架构局限性，从而推动其建模能力的边界。

- We will continuously iterate on the quantity and quality of our training data, and explore the incorporation of additional training signal sources, aiming to drive data scaling across a more comprehensive range of dimensions.
- We will consistently explore and iterate on the deep thinking capabilities of our models, aiming to enhance their intelligence and problem-solving abilities by expanding their reasoning length and depth.
- We will explore more comprehensive and multi-dimensional model evaluation methods to prevent the tendency towards optimizing a fixed set of benchmarks during research, which may create a misleading impression of the model capabilities and affect our foundational assessment.

References

AI@Meta. Llama 3 model card, 2024a. URL https://github.com/meta-llama/llama3/blob/main/MODEL_CARD.md.

AI@Meta. Llama 3.1 model card, 2024b. URL https://github.com/meta-llama/llama-models/blob/main/models/llama3_1/MODEL_CARD.md.

Anthropic. Claude 3.5 sonnet, 2024. URL https://www.anthropic.com/news/claude-3-5-sonnet.

J. Austin, A. Odena, M. Nye, M. Bosma, H. Michalewski, D. Dohan, E. Jiang, C. Cai, M. Terry, Q. Le, et al. Program synthesis with large language models. arXiv preprint arXiv:2108.07732, 2021.

Y. Bai, S. Kadavath, S. Kundu, A. Askell, J. Kernion, A. Jones, A. Chen, A. Goldie, A. Mirhoseini, C. McKinnon, et al. Constitutional AI: Harmlessness from AI feedback. arXiv preprint arXiv:2212.08073, 2022.

Y. Bai, S. Tu, J. Zhang, H. Peng, X. Wang, X. Lv, S. Cao, J. Xu, L. Hou, Y. Dong, J. Tang, and J. Li. LongBench v2: Towards deeper understanding and reasoning on realistic long-context multitasks. arXiv preprint arXiv:2412.15204, 2024.

M. Bauer, S. Treichler, and A. Aiken. Singe: leveraging warp specialization for high performance on GPUs. In Proceedings of the 19th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, PPoPP '14, page 119–130, New York, NY, USA, 2014. Association for Computing Machinery. ISBN 9781450326568. doi: 10.1145/2555243.2555258. URL https://doi.org/10.1145/2555243.2555258.

Y. Bisk, R. Zellers, R. L. Bras, J. Gao, and Y. Choi. PIQA: reasoning about physical commonsense in natural language. In The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020, pages 7432–7439. AAAI Press, 2020. doi: 10.1609/aaai.v34i05.6239. URL https://doi.org/10.1609/aaai.v34i05.6239.

M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. de Oliveira Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, A. Ray, R. Puri, G. Krueger, M. Petrov, H. Khlaaf, G. Sastry, P. Mishkin, B. Chan, S. Gray, N. Ryder, M. Pavlov, A. Power, L. Kaiser, M. Bavarian, C. Winter, P. Tillet, F. P. Such, D. Cummings, M. Plappert, F. Chantzis, E. Barnes, A. Herbert-Voss, W. H. Guss, A. Nichol, A. Paino, N. Tezak, J. Tang, I. Babuschkin, S. Balaji, S. Jain, W. Saunders, C. Hesse,

- 我们将持续迭代训练数据的数量和质量，并探索整合额外的训练信号源，旨在实现跨更广泛维度的数据扩展。
- 我们将持续探索和迭代模型的深度思考能力，通过扩展其推理长度和深度，提升其智能和问题解决能力。
- 我们将探索更全面、多维度的模型评估方法，以防止在研究过程中过度优化固定的一组基准，这可能造成对模型能力的误导印象，并影响我们的基础评估。

参考文献

AI@Meta. Llama 3模型卡，2024a。URL https://github.com/meta-llama/llama3/blob/main/MODEL CARD.md._

AI@Meta. Llama 3.1模型卡，2024b。URL https://github.com/meta-llama/llama-models/blob/main/models/llama3 1/MODEL CARD.md._ _

Anthropic. Claude 3.5 sonnet, 2024. URL https://www.anthropic.com/news/claude-3-5-sonnet.

J. Austin, A. Odena, M. Nye, M. Bosma, H. Michalewski, D. Dohan, E. Jiang, C. Cai, M. Terry, Q. Le, et al. 大型语言模型程序合成. arXiv 预印本 arXiv:2108.07732, 2021.

Y. Bai, S. Kadavath, S. Kundu, A. Askell, J. Kernion, A. Jones, A. Chen, A. Goldie, A.Mirhoseini, C. McKinnon, et al. 宪法式人工智能：从AI反馈中实现无害性. arXiv 预印本 arXiv:2212.08073, 2022.

Y. Bai, S. Tu, J. Zhang, H. Peng, X. Wang, X. Lv, S. Cao, J. Xu, L. Hou, Y. Dong, J. Tang, 和 J. Li. LongBench v2：迈向对现实长上下文多任务更深入的理解和推理. arXiv 预印本 arXiv:2412.15204, 2024.

M. Bauer, S. Treichler, 和 A. Aiken. Singe: 利用 warp 专业化实现 GPU 上的高性能. 在第 19 届 ACM SIGPLAN 并行编程原理与实践研讨会论文集，PPoPP’14，第 119–130 页，纽约，纽约州，美国，2014 年. 计算机协会. ISBN 9781450326568. doi: 10.1145/2555243.2555258. URLhttps://doi.org/10.1145/2555243.2555258.

Y. Bisk, R. Zellers, R. L. Bras, J. Gao, 和 Y. Choi. PIQA: 在自然语言中推理物理常识. 在第 34 届 AAAI 人工智能大会，AAAI 2020，第 32 届人工智能创新应用大会，IAAI 2020，第 10 届 AAAI 人工智能教育进展研讨会，EAAI 2020，纽约，纽约州，美国，2020 年 2 月 7-12 日，第 7432–7439 页. AAAI 出版社，2020 年. doi: 10.1609/aaai.v34i05.6239. URL https://doi.org/10.1609/aaai.v34i05.6239.

M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. de Oliveira Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, A. Ray, R. Puri, G.Krueger, M.Petrov, H. Khlaaf, G. Sastry, P. Mishkin, B. Chan, S. Gray, N. Ryder, M. Pavlov, A. Power, L. Kaiser, M. Bavarian, C. Winter, P. Tillet, F. P. Such, D. Cummings, M. Plappert, F. Chantzis, E. Barnes, A. Herbert-Voss, W. H. Guss, A. Nichol, A. Paino, N. Tezak, J. Tang, I. Babuschkin, S. Balaji, S. Jain, W. Saunders, C. Hesse

A. N. Carr, J. Leike, J. Achiam, V. Misra, E. Morikawa, A. Radford, M. Knight, M. Brundage, M. Murati, K. Mayer, P. Welinder, B. McGrew, D. Amodei, S. McCandlish, I. Sutskever, and W. Zaremba. Evaluating large language models trained on code. CoRR, abs/2107.03374, 2021. URL https://arxiv.org/abs/2107.03374.

P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, and O. Tafjord. Think you have solved question answering? try arc, the AI2 reasoning challenge. CoRR, abs/1803.05457, 2018. URL http://arxiv.org/abs/1803.05457.

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

Y. Cui, T. Liu, W. Che, L. Xiao, Z. Chen, W. Ma, S. Wang, and G. Hu. A span-extraction dataset for Chinese machine reading comprehension. In K. Inui, J. Jiang, V. Ng, and X. Wan, editors, Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 5883–5889, Hong Kong, China, Nov. 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1600. URL https://aclanthology.org/D19-1600.

D. Dai, C. Deng, C. Zhao, R. X. Xu, H. Gao, D. Chen, J. Li, W. Zeng, X. Yu, Y. Wu, Z. Xie, Y. K. Li, P. Huang, F. Luo, C. Ruan, Z. Sui, and W. Liang. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. CoRR, abs/2401.06066, 2024. URL https://doi.org/10.48550/arXiv.2401.06066.

DeepSeek-AI. Deepseek-coder-v2: Breaking the barrier of closed-source models in code intelligence. CoRR, abs/2406.11931, 2024a. URL https://doi.org/10.48550/arXiv.2406.11931.

DeepSeek-AI. Deepseek LLM: scaling open-source language models with longtermism. CoRR, abs/2401.02954, 2024b. URL https://doi.org/10.48550/arXiv.2401.02954.

DeepSeek-AI. Deepseek-v2: A strong, economical, and efficient mixture-of-experts language model. CoRR, abs/2405.04434, 2024c. URL https://doi.org/10.48550/arXiv.2405.04434.

T. Dettmers, M. Lewis, Y. Belkada, and L. Zettlemoyer. Gpt3. int8 (): 8-bit matrix multiplication for transformers at scale. Advances in Neural Information Processing Systems, 35:30318–30332, 2022.

H. Ding, Z. Wang, G. Paolini, V. Kumar, A. Deoras, D. Roth, and S. Soatto. Fewer truncations improve language modeling. arXiv preprint arXiv:2404.10830, 2024.

D. Dua, Y. Wang, P. Dasigi, G. Stanovsky, S. Singh, and M. Gardner. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In J. Burstein, C. Doran, and T. Solorio, editors, Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers), pages 2368–2378. Association for Computational Linguistics, 2019. doi: 10.18653/V1/N19-1246. URL https://doi.org/10.18653/v1/n19-1246.

Y. Dubois, B. Galambosi, P. Liang, and T. B. Hashimoto. Length-controlled alpacaeval: A simple way to debias automatic evaluators. arXiv preprint arXiv:2404.04475, 2024.

A. N. Carr, J. Leike, J. Achiam, V. Misra, E. Morikawa, A. Radford, M. Knight, M. Brundage, M. Murati, K. Mayer, P. Welinder, B. McGrew, D. Amodei, S. McCandlish, I. Sutskever, 和 W. Zaremba. 评估在代码上训练的大型语言模型。CoRR, abs/2107.03374, 2021. URL https://arxiv.org/abs/2107.03374.

P. Clark, I. Cowhey, O. Etzioni, T. Khot, A. Sabharwal, C. Schoenick, 和 O. Tafjord. 你认为已经解决了问答问题？试试 arc，AI2 推理挑战。CoRR, abs/1803.05457, 2018. URL http://arxiv.org/abs/1803.05457.

K. Cobbe, V. Kosaraju, M. Bavarian, M. Chen, H. Jun, L. Kaiser, M. Plappert, J. Tworek, J. Hilton, R. Nakano, 等. 训练验证器解决数学应用题. arXiv 预印本 arXiv:2110.14168, 2021.

Y. Cui, T. Liu, W. Che, L. Xiao, Z. Chen, W. Ma, S. Wang, 和 G. Hu. 一个用于中文机器阅读理解的抽取式数据集. 收录于 K. Inui, J. Jiang, V. Ng, 和 X. Wan 编, 第 2019 年自然语言处理经验方法会议暨第九届国际自然语言处理联合会议论文集 (EMNLP-IJCNLP), 第 5883–5889 页, 中国香港, 2019 年 11 月. 计算语言学协会. doi: 10.18653/v1/D19-1600. URL https://aclanthology.org/D19-1600.

D. 戴, C. 邓恩, C. 赵晓, R. 许, H. 高, D. 陈, J. 李, W. 曾, X. 余, Y. 吴, Z. 谢晓, Y. 李, P. 黄, F. 罗欧, C. 阮, Z. 隋, 和 W. 梁撰. Deepseekmoe: 迈向专家专精的混合专家语言模型. CoRR, abs/2401.06066, 2024. URLhttps://doi.org/10.48550/arXiv.2401.06066.

DeepSeek-AI. Deepseek-coder-v2: 打破闭源模型在代码智能中的障碍. CoRR, abs/2406.11931, 2024a. URL https://doi.org/10.48550/arXiv.2406.11931.

DeepSeek-AI. Deepseek LLM: 以长期主义扩展开源语言模型. CoRR, abs/2401.02954, 2024b. URL https://doi.org/10.48550/arXiv.2401.02954.

DeepSeek-AI. Deepseek-v2: 一个强大、经济且高效的专家混合语言模型。CoRR, abs/2405.04434, 2024c. URL https://doi.org/10.48550/arXiv.2405.04434.

T. Dettmers, M. Lewis, Y. Belkada, 和 L. Zettlemoyer. Gpt3. int8 (): 用于大规模 Transformer 的 8 位矩阵乘法。神经信息处理系统进展, 35:30318– 30332, 2022。

H. Ding, Z. Wang, G. Paolini, V. Kumar, A. Deoras, D. Roth, 和 S. Soatto. 更少的截断改进语言建模。arXiv 预印本 arXiv:2404.10830, 2024。

D. Dua, Y. Wang, P. Dasigi, G. Stanovsky, S. Singh, 和 M. Gardner. DROP: 一个需要段落离散推理的阅读理解基准。在 J. Burstein, C. Doran, 和 T. Solorio 编辑的 2019 年北美计算语言学协会分会会议论文集：人机语言技术，NAACL-HLT 2019，明尼苏达州明尼阿波利斯，美国，2019 年 6 月 2-7 日，第 1 卷（长篇和短篇论文），第 2368– 2378 页。计算语言学协会，2019 年。doi: 10.18653/V1/N19-1246。URLhttps://doi.org/10.18653/v1/n19-1246。

Y. Dubois, B. Galambosi, P. Liang, 和 T. B. Hashimoto. 长度控制的 alpacaeval：一种简单的方法来消除自动评估器的偏差。arXiv 预印本 arXiv:2404.04475, 2024。

W. Fedus, B. Zoph, and N. Shazeer. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. CoRR, abs/2101.03961, 2021. URL https://arxiv.org/abs/2101.03961.

M. Fishman, B. Chmiel, R. Banner, and D. Soudry. Scaling FP8 training to trillion-token llms. arXiv preprint arXiv:2409.12517, 2024.

E. Frantar, S. Ashkboos, T. Hoefler, and D. Alistarh. Gptq: Accurate post-training quantization for generative pre-trained transformers. arXiv preprint arXiv:2210.17323, 2022.

L. Gao, S. Biderman, S. Black, L. Golding, T. Hoppe, C. Foster, J. Phang, H. He, A. Thite, N. Nabeshima, et al. The Pile: An 800GB dataset of diverse text for language modeling. arXiv preprint arXiv:2101.00027, 2020.

A. P. Gema, J. O. J. Leang, G. Hong, A. Devoto, A. C. M. Mancino, R. Saxena, X. He, Y. Zhao, X. Du, M. R. G. Madani, C. Barale, R. McHardy, J. Harris, J. Kaddour, E. van Krieken, and P. Minervini. Are we done with mmlu? CoRR, abs/2406.04127, 2024. URL https://doi.org/10.48550/arXiv.2406.04127.

F. Gloeckle, B. Y. Idrissi, B. Rozière, D. Lopez-Paz, and G. Synnaeve. Better & faster large language models via multi-token prediction. In Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024. OpenReview.net, 2024. URL https://openreview.net/forum?id=pEWAcejiU2.

Google. Our next-generation model: Gemini 1.5, 2024. URL https://blog.google/technology/ai/google-gemini-next-generation-model-february-2024.

R. L. Graham, D. Bureddy, P. Lui, H. Rosenstock, G. Shainer, G. Bloch, D. Goldenerg, M. Dubman, S. Kotchubievsky, V. Koushnir, et al. Scalable hierarchical aggregation protocol (SHArP): A hardware architecture for efficient data reduction. In 2016 First International Workshop on Communication Optimizations in HPC (COMHPC), pages 1–10. IEEE, 2016.

A. Gu, B. Rozière, H. Leather, A. Solar-Lezama, G. Synnaeve, and S. I. Wang. Cruxeval: A benchmark for code reasoning, understanding and execution, 2024.

D. Guo, Q. Zhu, D. Yang, Z. Xie, K. Dong, W. Zhang, G. Chen, X. Bi, Y. Wu, Y. K. Li, F. Luo, Y. Xiong, and W. Liang. Deepseek-coder: When the large language model meets programming - the rise of code intelligence. CoRR, abs/2401.14196, 2024. URL https://doi.org/10.48550/arXiv.2401.14196.

A. Harlap, D. Narayanan, A. Phanishayee, V. Seshadri, N. Devanur, G. Ganger, and P. Gibbons. Pipedream: Fast and efficient pipeline parallel dnn training, 2018. URL https://arxiv.org/abs/1806.03377.

B. He, L. Noci, D. Paliotta, I. Schlag, and T. Hofmann. Understanding and minimising outlier features in transformer training. In The Thirty-eighth Annual Conference on Neural Information Processing Systems.

Y. He, S. Li, J. Liu, Y. Tan, W. Wang, H. Huang, X. Bu, H. Guo, C. Hu, B. Zheng, et al. Chinese simpleqa: A chinese factuality evaluation for large language models. arXiv preprint arXiv:2411.07140, 2024.

D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, and J. Steinhardt. Measuring massive multitask language understanding. arXiv preprint arXiv:2009.03300, 2020.

W. Fedus, B. Zoph, 和 N. Shazeer. 开关Transformer：使用简单高效的稀疏性扩展到万亿参数模型。CoRR, abs/2101.03961, 2021. URL https://arxiv.org/abs/2101.03961.

M. Fishman, B. Chmiel, R. Banner, 和 D. Soudry. 将FP8训练扩展到万亿token的LLM。arXiv预印本 arXiv:2409.12517, 2024。

E. Frantar, S. Ashkboos, T. Hoefler, 和 D. Alistarh. Gptq：为生成式预训练Transformer的精确训练后量化。arXiv预印本 arXiv:2210.17323, 2022。

L. Gao, S. Biderman, S. Black, L. Golding, T. Hoppe, C. Foster, J. Phang, H. He, A. Thite, N. Nabeshima, 等。Pile：一个800GB的多样化文本数据集用于语言建模。arXiv预印本 arXiv:2101.00027, 2020。

A. P. Gema, J. O. J. Leang, G. Hong, A. Devoto, A. C. M. Mancino, R. Saxena, X. He, Y. Zhao, X. Du, M. R. G. Madani, C. Barale, R. McHardy, J. Harris, J. Kaddour, E. van Krieken, 和 P. Minervini. 我们完成 mmlu 吗？CoRR, abs/2406.04127, 2024. URL https://doi.org/10.48550/arXiv.2406.04127.

F. Gloeckle, B. Y. Idrissi, B. Rozière, D. Lopez-Paz, 和 G. Synnaeve. 通过多标记预测实现更好、更快的大语言模型。在第四十一届国际机器学习会议，ICML 2024，奥地利维也纳，2024年7月21-27日。OpenReview.net, 2024. URLhttps://openreview.net/forum?id=pEWAcejiU2.

Google. 我们的下一代模型：Gemini 1.5，2024。URL https://blog.google/technology/ai/google-gemini-next-generation-model-february-2024。

R. L. Graham, D. Bureddy, P. Lui, H. Rosenstock, G. Shainer, G. Bloch, D. Goldenerg, M. Dubman, S. Kotchubievsky, V. Koushnir, 等。可扩展分层聚合协议（SHArP）：一种用于高效数据缩减的硬件架构。在2016年第一届高性能计算通信优化研讨会（COMHPC）中，第1-10页。IEEE, 2016。

A. Gu, B. Rozière, H. Leather, A. Solar-Lezama, G. Synnaeve, 和 S. I. Wang。Cruxeval：一个用于代码推理、理解和执行的基准测试，2024。

郭丹, 朱启, 杨东, 谢哲, 董凯, 张伟, 陈刚, 毕翔, 吴宇, 李奕坤, 罗峰, 邢宇, 和梁伟. Deepseek-coder: 当大语言模型遇见编程 - 代码智能的崛起. CoRR, abs/2401.14196, 2024. URL https://doi.org/10.48550/arXiv.2401.14196.

哈拉普·阿南德, 纳拉扬·达斯, 菲尼沙耶·阿南德, 舍哈德里·维什瓦南, 德瓦纳尔·纳特, 加nger·格安格, 和吉本斯·帕特里克. Pipedream: 快速高效的管道并行DNN训练, 2018. URL https://arxiv.org/abs/1806.03377.

何博, 诺西·莱昂, 帕利奥塔·达米安, 施拉格·伊尔, 和霍夫曼·托马斯. 理解和最小化Transformer训练中的异常特征. 在第38届神经信息处理系统年会上.

Y. He, S. Li, J. Liu, Y. Tan, W. Wang, H. Huang, X. Bu, H. Guo, C. Hu, B. Zheng, 等. Chinese simpleqa: 一种针对大型语言模型的中文事实性评估. arXiv 预印本 arXiv:2411.07140, 2024.

D. Hendrycks, C. Burns, S. Basart, A. Zou, M. Mazeika, D. Song, 和 J. Steinhardt. 测量大规模多任务语言理解. arXiv 预印本 arXiv:2009.03300, 2020.

D. Hendrycks, C. Burns, S. Kadavath, A. Arora, S. Basart, E. Tang, D. Song, and J. Steinhardt. Measuring mathematical problem solving with the math dataset. arXiv preprint arXiv:2103.03874, 2021.

Y. Huang, Y. Bai, Z. Zhu, J. Zhang, J. Zhang, T. Su, J. Liu, C. Lv, Y. Zhang, J. Lei, et al. C-Eval: A multi-level multi-discipline chinese evaluation suite for foundation models. arXiv preprint arXiv:2305.08322, 2023.

N. Jain, K. Han, A. Gu, W. Li, F. Yan, T. Zhang, S. Wang, A. Solar-Lezama, K. Sen, and I. Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. CoRR, abs/2403.07974, 2024. URL `https://doi.org/10.48550/arXiv.2403.07974`.

A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. d. l. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, et al. Mistral 7b. arXiv preprint arXiv:2310.06825, 2023.

M. Joshi, E. Choi, D. Weld, and L. Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In R. Barzilay and M.-Y. Kan, editors, Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1601–1611, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1147. URL `https://aclanthology.org/P17-1147`.

D. Kalamkar, D. Mudigere, N. Mellempudi, D. Das, K. Banerjee, S. Avancha, D. T. Vooturi, N. Jammalamadaka, J. Huang, H. Yuen, et al. A study of bfloat16 for deep learning training. arXiv preprint arXiv:1905.12322, 2019.

S. Krishna, K. Krishna, A. Mohananey, S. Schwarcz, A. Stambler, S. Upadhyay, and M. Faruqui. Fact, fetch, and reason: A unified evaluation of retrieval-augmented generation. CoRR, abs/2409.12941, 2024. doi: 10.48550/ARXIV.2409.12941. URL `https://doi.org/10.48550/arXiv.2409.12941`.

T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, A. P. Parikh, C. Alberti, D. Epstein, I. Polosukhin, J. Devlin, K. Lee, K. Toutanova, L. Jones, M. Kelcey, M. Chang, A. M. Dai, J. Uszkoreit, Q. Le, and S. Petrov. Natural questions: a benchmark for question answering research. Trans. Assoc. Comput. Linguistics, 7:452–466, 2019. doi: 10.1162/tacl_a_00276. URL `https://doi.org/10.1162/tacl_a_00276`.

G. Lai, Q. Xie, H. Liu, Y. Yang, and E. H. Hovy. RACE: large-scale reading comprehension dataset from examinations. In M. Palmer, R. Hwa, and S. Riedel, editors, Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017, pages 785–794. Association for Computational Linguistics, 2017. doi: 10.18653/V1/D17-1082. URL `https://doi.org/10.18653/v1/d17-1082`.

N. Lambert, V. Pyatkin, J. Morrison, L. Miranda, B. Y. Lin, K. Chandu, N. Dziri, S. Kumar, T. Zick, Y. Choi, et al. Rewardbench: Evaluating reward models for language modeling. arXiv preprint arXiv:2403.13787, 2024.

D. Lepikhin, H. Lee, Y. Xu, D. Chen, O. Firat, Y. Huang, M. Krikun, N. Shazeer, and Z. Chen. Gshard: Scaling giant models with conditional computation and automatic sharding. In 9th International Conference on Learning Representations, ICLR 2021. OpenReview.net, 2021. URL `https://openreview.net/forum?id=qrwe7XHTmYb`.

D. Hendrycks, C. Burns, S. Kadavath, A. Arora, S. Basart, E. Tang, D. Song, 和 J. Steinhardt. 使用数学数据集测量数学问题解决能力。arXiv 预印本 arXiv:2103.03874, 2021。

Y. Huang, Y. Bai, Z. Zhu, J. Zhang, J. Zhang, T. Su, J. Liu, C. Lv, Y. Zhang, J. Lei, 等。C-Eval：面向基础模型的跨层次跨学科中文评估套件。arXiv 预印本 arXiv:2305.08322, 2023。

N. Jain, K. Han, A. Gu, W. Li, F. Yan, T. Zhang, S. Wang, A. Solar-Lezama, K. Sen, 和 I. Stoica。Livecodebench：对代码的大型语言模型进行全面且无污染的评估。CoRR, abs/2403.07974, 2024。URL `https://doi.org/10.48550/arXiv.2403.07974`。

A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. d. l. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, 等. Mistral 7b. arXiv 预印本 arXiv:2310.06825, 2023.

M. Joshi, E. Choi, D. Weld, 和 L. Zettlemoyer. TriviaQA: 一个大规模的远程监督阅读理解挑战数据集. 在 R. Barzilay 和 M.-Y. Kan 编的《计算语言学协会第 55 届年会论文集（第一卷：长篇论文）》，第 1601–1611 页，加拿大温哥华，2017 年 7 月. 计算语言学协会. doi: 10.18653/v1/P17-1147. URL `https://aclanthology.org/P17-1147`.

D. Kalamkar, D. Mudigere, N. Mellempudi, D. Das, K. Banerjee, S. Avancha, D. T. Vooturi, N. Jammalamadaka, J. Huang, H. Yuen, 等. 深度学习训练中 bfloat16 的研究. arXiv 预印本 arXiv:1905.12322, 2019.

S. Krishna, K. Krishna, A. Mohananey, S. Schwarcz, A. Stambler, S. Upadhyay, 和 M. Faruqui. 事实、获取与推理：检索增强生成统一评估. CoRR, abs/2409.12941, 2024. doi: 10.48550/ARXIV.2409.12941. URL `https://doi.org/10.48550/arXiv.2409.12941`.

T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, A. P. Parikh, C. Alberti, D. Epstein, I. Polosukhin, J. Devlin, K. Lee, K. Toutanova, L. Jones, M. Kelcey, M. Chang, A. M. Dai, J. Uszkoreit, Q. Le, 和 S. Petrov. 自然问题：问答研究基准. 计算语言学协会汇刊, 7:452–466, 2019. doi: 10.1162/tacl_a_00276. URL `https://doi.org/10.1162/tacl_a_00276`.

G. Lai, Q. Xie, H. Liu, Y. Yang, 和 E. H. Hovy. RACE: 从考试中来的大规模阅读理解数据集。在 M. Palmer, R. Hwa, 和 S. Riedel 编辑的《2017 年自然语言处理经验方法会议论文集》，EMNLP 2017，哥本哈根，丹麦，2017 年 9 月 9-11 日，第 785-794 页。计算语言学协会，2017 年。doi: 10.18653/V1/D17-1082。URL `https://doi.org/10.18653/v1/d17-1082`。

N. Lambert, V. Pyatkin, J. Morrison, L. Miranda, B. Y. Lin, K. Chandu, N. Dziri, S. Kumar, T. Zick, Y. Choi, 等。Rewardbench: 评估语言模型的奖励模型。arXiv 预印本 arXiv:2403.13787，2024 年。

D. Lepikhin, H. Lee, Y. Xu, D. Chen, O. Firat, Y. Huang, M. Krikun, N. Shazeer, 和 Z. Chen. Gshard: 使用条件计算和自动分片扩展巨型模型。在第九届学习表示国际会议，ICLR 2021。OpenReview.net, 2021。URL `https://openreview.net/forum?id=qrwe7XHTmYb`.

Y. Leviathan, M. Kalman, and Y. Matias. Fast inference from transformers via speculative decoding. In International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA, volume 202 of Proceedings of Machine Learning Research, pages 19274–19286. PMLR, 2023. URL https://proceedings.mlr.press/v202/leviathan23a.html.

H. Li, Y. Zhang, F. Koto, Y. Yang, H. Zhao, Y. Gong, N. Duan, and T. Baldwin. CMMLU: Measuring massive multitask language understanding in Chinese. arXiv preprint arXiv:2306.09212, 2023.

S. Li and T. Hoefler. Chimera: efficiently training large-scale neural networks with bidirectional pipelines. In Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC '21, page 1–14. ACM, Nov. 2021. doi: 10.1145/3458817.3476145. URL http://dx.doi.org/10.1145/3458817.3476145.

T. Li, W.-L. Chiang, E. Frick, L. Dunlap, T. Wu, B. Zhu, J. E. Gonzalez, and I. Stoica. From crowdsourced data to high-quality benchmarks: Arena-hard and benchbuilder pipeline. arXiv preprint arXiv:2406.11939, 2024a.

W. Li, F. Qi, M. Sun, X. Yi, and J. Zhang. Ccpm: A chinese classical poetry matching dataset, 2021.

Y. Li, F. Wei, C. Zhang, and H. Zhang. EAGLE: speculative sampling requires rethinking feature uncertainty. In Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024. OpenReview.net, 2024b. URL https://openreview.net/forum?id=1NdN7eXyb4.

B. Y. Lin. ZeroEval: A Unified Framework for Evaluating Language Models, July 2024. URL https://github.com/WildEval/ZeroEval.

I. Loshchilov and F. Hutter. Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101, 2017.

S. Lundberg. The art of prompt design: Prompt boundaries and token healing, 2023. URL https://towardsdatascience.com/the-art-of-prompt-design-prompt-boundaries-and-token-healing-3b2448b0be38.

Y. Luo, Z. Zhang, R. Wu, H. Liu, Y. Jin, K. Zheng, M. Wang, Z. He, G. Hu, L. Chen, et al. Ascend HiFloat8 format for deep learning. arXiv preprint arXiv:2409.16626, 2024.

MAA. American invitational mathematics examination - aime. In American Invitational Mathematics Examination - AIME 2024, February 2024. URL https://maa.org/math-competitions/american-invitational-mathematics-examination-aime.

P. Micikevicius, D. Stosic, N. Burgess, M. Cornea, P. Dubey, R. Grisenthwaite, S. Ha, A. Heinecke, P. Judd, J. Kamalu, et al. FP8 formats for deep learning. arXiv preprint arXiv:2209.05433, 2022.

Mistral. Cheaper, better, faster, stronger: Continuing to push the frontier of ai and making it accessible to all, 2024. URL https://mistral.ai/news/mixtral-8x22b.

S. Narang, G. Diamos, E. Elsen, P. Micikevicius, J. Alben, D. Garcia, B. Ginsburg, M. Houston, O. Kuchaiev, G. Venkatesh, et al. Mixed precision training. In Int. Conf. on Learning Representation, 2017.

Y. Leviathan, M. Kalman, 和 Y. Matias. 基于推测解码的 Transformer 快速推理. 在机器学习国际会议, ICML 2023, 2023年7月23日至29日, 夏威夷檀香山, 美国夏威夷州, 第202卷《机器学习研究论文集》, 第19274–19286页. PMLR, 2023. URL https://proceedings.mlr.press/v202/leviathan23a.html.

H. Li, Y. Zhang, F. Koto, Y. Yang, H. Zhao, Y. Gong, N. Duan, 和 T. Baldwin. CMMLU: 测量中文大规模多任务语言理解. arXiv 预印本 arXiv:2306.09212, 2023.

S. Li 和 T. Hoefler. Chimera: 通过双向管道高效训练大规模神经网络. 在高性能计算、网络、存储和分析国际会议论文集, SC ' 21, 第1–14页. ACM, 2021年11月. doi: 10.1145/3458817.3476145. URL http://dx.doi.org/10.1145/3458817.3476145.

T. Li, W.-L. Chiang, E. Frick, L. Dunlap, T. Wu, B. Zhu, J. E. Gonzalez, and I. Stoica. 从众包数据到高质量基准：Arena-hardand benchbuilder 管道. arXiv 预印本 arXiv:2406.11939, 2024a.

W. Li, F. Qi, M. Sun, X. Yi, and J. Zhang. Ccpm: 一个中国古典诗歌匹配数据集, 2021.

Y. Li, F. Wei, C. Zhang, and H. Zhang. EAGLE: 推测采样需要重新思考特征不确定性. 在第41届国际机器学习会议, ICML 2024, 维也纳, 奥地利, 2024年7月21-27日. OpenReview.net, 2024b. URL https://openreview.net/forum?id=1NdN7eXyb4.

B. Y. Lin. ZeroEval: 一个用于评估语言模型的统一框架, 2024年7月. URL https://github.com/WildEval/ZeroEval.

I. Loshchilov 和 F. Hutter. 解耦权重衰减正则化. arXiv 预印本 arXiv:1711.05101, 2017.

S. Lundberg. 提示设计的艺术：提示边界和标记修复, 2023. URL https://towardsdatascience.com/the-art-of-prompt-design-prompt-boundaries-and-token-healing-3b2448b0be38.

Y. Luo, Z. Zhang, R. Wu, H. Liu, Y. Jin, K. Zheng, M. Wang, Z. He, G. Hu, L. Chen, 等. Ascend HiFloat8 格式用于深度学习. arXiv 预印本 arXiv:2409.16626, 2024.

MAA. 美国邀请数学考试 - aime. 在美国邀请数学考试 - AIME 2024, 2024 年 2 月. URL https://maa.org/math-competitions/american-invitational-mathematics-examination-aime.

P. Micikevicius, D. Stosic, N. Burgess, M. Cornea, P. Dubey, R. Grisenthwaite, S. Ha, A. Heinecke, P. Judd, J. Kamalu, 等. 用于深度学习的FP8格式. arXiv预印本 arXiv:2209.05433, 2022.

Mistral. 更便宜、更好、更快、更强：继续推动人工智能的边界并使其人人可及, 2024. URL https://mistral.ai/news/mixtral-8x22b.

S. Narang, G. Diamos, E. Elsen, P. Micikevicius, J. Alben, D. Garcia, B. Ginsburg, M. Houston, O. Kuchaiev, G. Venkatesh, 等. 混合精度训练. 在国际学习表示会议, 2017.

B. Noune, P. Jones, D. Justus, D. Masters, and C. Luschi. 8-bit numerical formats for deep neural networks. arXiv preprint arXiv:2206.02915, 2022.

NVIDIA. Improving network performance of HPC systems using NVIDIA Magnum IO NVSHMEM and GPUDirect Async. `https://developer.nvidia.com/blog/improving-network-performance-of-hpc-systems-using-nvidia-magnum-io-nvshmem-and-gpudirect-async`, 2022.

NVIDIA. Blackwell architecture. `https://www.nvidia.com/en-us/data-center/technologies/blackwell-architecture/`, 2024a.

NVIDIA. TransformerEngine, 2024b. URL `https://github.com/NVIDIA/TransformerEngine`. Accessed: 2024-11-19.

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

OpenAI. Multilingual massive multitask language understanding (mmmlu), 2024b. URL `https://huggingface.co/datasets/openai/MMMLU`.

OpenAI. Introducing SimpleQA, 2024c. URL `https://openai.com/index/introducing-simpleqa/`.

OpenAI. Introducing SWE-bench verified we're releasing a human-validated subset of swe-bench that more, 2024d. URL `https://openai.com/index/introducing-swe-bench-verified/`.

B. Peng, J. Quesnelle, H. Fan, and E. Shippole. Yarn: Efficient context window extension of large language models. arXiv preprint arXiv:2309.00071, 2023a.

H. Peng, K. Wu, Y. Wei, G. Zhao, Y. Yang, Z. Liu, Y. Xiong, Z. Yang, B. Ni, J. Hu, et al. FP8-LM: Training FP8 large language models. arXiv preprint arXiv:2310.18313, 2023b.

P. Qi, X. Wan, G. Huang, and M. Lin. Zero bubble pipeline parallelism. arXiv preprint arXiv:2401.10241, 2023a.

P. Qi, X. Wan, G. Huang, and M. Lin. Zero bubble pipeline parallelism, 2023b. URL `https://arxiv.org/abs/2401.10241`.

Qwen. Qwen technical report. arXiv preprint arXiv:2309.16609, 2023.

Qwen. Introducing Qwen1.5, 2024a. URL `https://qwenlm.github.io/blog/qwen1.5`.

Qwen. Qwen2.5: A party of foundation models, 2024b. URL `https://qwenlm.github.io/blog/qwen2.5`.

S. Rajbhandari, J. Rasley, O. Ruwase, and Y. He. Zero: Memory optimizations toward training trillion parameter models. In SC20: International Conference for High Performance Computing, Networking, Storage and Analysis, pages 1–16. IEEE, 2020.

D. Rein, B. L. Hou, A. C. Stickland, J. Petty, R. Y. Pang, J. Dirani, J. Michael, and S. R. Bowman. GPQA: A graduate-level google-proof q&a benchmark. arXiv preprint arXiv:2311.12022, 2023.

B. D. Rouhani, R. Zhao, A. More, M. Hall, A. Khodamoradi, S. Deng, D. Choudhary, M. Cornea, E. Dellinger, K. Denolf, et al. Microscaling data formats for deep learning. arXiv preprint arXiv:2310.10537, 2023a.

B. Noune, P. Jones, D. Justus, D. Masters, 和 C. Luschi. 8位数值格式用于深度神经网络. arXiv 预印本 arXiv:2206.02915, 2022.

NVIDIA. 使用 NVIDIA Magnum IO NVSH-MEM 和 GPUDirect Async 提高 HPC 系统的网络性能. `https://developer.nvidia.com/blog/improving-network-performance-of-hpc-systems-using-nvidia-magnum-io-nvshmem-and-gpudirect-async`, 2022.

NVIDIA. Blackwell 架构. `https://www.nvidia.com/en-us/data-center/technologies/blackwell-architecture/`, 2024a.

NVIDIA. TransformerEngine, 2024b. URL `https://github.com/NVIDIA/TransformerEngine`. 访问时间: 2024-11-19.

OpenAI。你好 GPT-4o，2024a。URL `https://openai.com/index/hello-gpt-4o/`

OpenAI. 多语言大规模多任务语言理解 (mmmlu), 2024b. URL `https://huggingface.co/datasets/openai/MMMLU`.

OpenAI. 介绍 SimpleQA, 2024c. URL `https://openai.com/index/introducing-simpleqa/`.

OpenAI. 介绍 SWE-bench 验证我们发布了一个人类验证的 swe-bench 子集，2024d. URL `https://openai.com/index/introducing-swe-bench-verified/`.

B. Peng, J. Quesnelle, H. Fan, 和 E. Shippole. Yarn: 高效的大语言模型上下文窗口扩展. arXiv 预印本 arXiv:2309.00071, 2023a.

H. Peng, K. Wu, Y. Wei, G. Zhao, Y. Yang, Z. Liu, Y. Xiong, Z. Yang, B. Ni, J. Hu, 等人. FP8-LM: 训练 FP8 大语言模型. arXiv 预印本 arXiv:2310.18313, 2023b.

P. Qi, X. Wan, G. Huang, 和 M. Lin. 零气泡管道并行性. arXiv 预印本 arXiv:2401.10241, 2023a.

P. Qi, X. Wan, G. Huang, 和 M. Lin. 零气泡管道并行性, 2023b. URL `https://arxiv.org/abs/2401.10241`.

Qwen. Qwen 技术报告. arXiv 预印本 arXiv:2309.16609, 2023.

Qwen. 介绍 Qwen1.5, 2024a. URL `https://qwenlm.github.io/blog/qwen1.5`.

Qwen. Qwen2.5: 一组基础模型, 2024b. URL `https://qwenlm.github.io/blog/qwen2.5`.

S.Rajbhandari, J.Rasley, O. Ruwase, 和 Y.He. Zero: 内存优化面向训练万亿参数模型. 在 SC20: 高性能计算、网络、存储和分析国际会议, 页面 1–16. IEEE, 2020.

D. Rein, B. L. Hou, A. C. Stickland, J. Petty, R. Y. Pang, J. Dirani, J. Michael, 和 S. R. Bowman. GPQA: 一个研究生级别的谷歌验证问答基准. arXiv 预印本 arXiv:2311.12022, 2023.

B. D. Rouhani, R. Zhao, A. More, M. Hall, A. Khodamoradi, S. Deng, D. Choudhary, M. Cornea, E. Dellinger, K. Denolf, 等人. 深度学习的微缩数据格式. arXiv 预印本 arXiv:2310.10537, 2023a.

B. D. Rouhani, R. Zhao, A. More, M. Hall, A. Khodamoradi, S. Deng, D. Choudhary, M. Cornea, E. Dellinger, K. Denolf, et al. Microscaling data formats for deep learning. arXiv preprint arXiv:2310.10537, 2023b.

K. Sakaguchi, R. L. Bras, C. Bhagavatula, and Y. Choi. Winogrande: An adversarial winograd schema challenge at scale, 2019.

Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, M. Zhang, Y. Li, Y. Wu, and D. Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. arXiv preprint arXiv:2402.03300, 2024.

N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. V. Le, G. E. Hinton, and J. Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In 5th International Conference on Learning Representations, ICLR 2017. OpenReview.net, 2017. URL https://openreview.net/forum?id=B1ckMDqlg.

F. Shi, M. Suzgun, M. Freitag, X. Wang, S. Srivats, S. Vosoughi, H. W. Chung, Y. Tay, S. Ruder, D. Zhou, D. Das, and J. Wei. Language models are multilingual chain-of-thought reasoners. In The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023. OpenReview.net, 2023. URL https://openreview.net/forum?id=fR3wGCk-IXp.

Y. Shibata, T. Kida, S. Fukamachi, M. Takeda, A. Shinohara, T. Shinohara, and S. Arikawa. Byte pair encoding: A text compression scheme that accelerates pattern matching. 1999.

J. Su, M. Ahmed, Y. Lu, S. Pan, W. Bo, and Y. Liu. Roformer: Enhanced transformer with rotary position embedding. Neurocomputing, 568:127063, 2024.

K. Sun, D. Yu, D. Yu, and C. Cardie. Investigating prior knowledge for challenging chinese machine reading comprehension, 2019a.

M. Sun, X. Chen, J. Z. Kolter, and Z. Liu. Massive activations in large language models. arXiv preprint arXiv:2402.17762, 2024.

X. Sun, J. Choi, C.-Y. Chen, N. Wang, S. Venkataramani, V. V. Srinivasan, X. Cui, W. Zhang, and K. Gopalakrishnan. Hybrid 8-bit floating point (HFP8) training and inference for deep neural networks. Advances in neural information processing systems, 32, 2019b.

M. Suzgun, N. Scales, N. Schärli, S. Gehrmann, Y. Tay, H. W. Chung, A. Chowdhery, Q. V. Le, E. H. Chi, D. Zhou, et al. Challenging big-bench tasks and whether chain-of-thought can solve them. arXiv preprint arXiv:2210.09261, 2022.

V. Thakkar, P. Ramani, C. Cecka, A. Shivam, H. Lu, E. Yan, J. Kosaian, M. Hoemmen, H. Wu, A. Kerr, M. Nicely, D. Merrill, D. Blasig, F. Qiao, P. Majcher, P. Springer, M. Hohnerbach, J. Wang, and M. Gupta. CUTLASS, Jan. 2023. URL https://github.com/NVIDIA/cutlass.

H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. LLaMA: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971, 2023a.

H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. Canton-Ferrer, M. Chen, G. Cucurull, D. Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini,

B. D. Rouhani, R. 赵亚飞, A. More, M. Hall, A. Khodamoradi, S. Deng, D. Choudhary, M. Cornea, E. Dellinger, K. Denolf, et al. 微观尺度数据格式用于深度学习。arXiv 预印本 arXiv:2310.10537, 2023b。

K. Sakaguchi, R. L. Bras, C. Bhagavatula, 和 Y. Choi. Winogrande: 大规模对抗性 Winograd 方案挑战, 2019。

Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, M. Zhang, Y. Li, Y. Wu, 和 D. Guo. Deepseekmath: 推动开放语言模型中数学推理的极限。arXiv 预印本 arXiv:2402.03300, 2024。

N. Shazeer, A. Mirhoseini, K. Maziarz, A. Davis, Q. V. Le, G. E. Hinton, 和 J. Dean. 极其庞大的神经网络：稀疏门控专家混合层。在 5th International Conference on Learning Representations, ICLR 2017。OpenReview.net, 2017。URL https://openreview.net/forum?id=B1ckMDqlg。

F. Shi, M. Suzgun, M. Freitag, X. Wang, S. Srivats, S. Vosoughi, H. W. Chung, Y. Tay, S. Ruder, D. Zhou, D. Das, 和 J. Wei. 语言模型是多语言思维链推理器. 在第11届学习表示国际会议，ICLR 2023，卢旺达基加利，2023年5月1日至5日. OpenReview.net, 2023. URL https://openreview.net/forum?id=fR3wGCk-IXp.

Y. Shibata, T. Kida, S. Fukamachi, M. Takeda, A. Shinohara, T. Shinohara, 和 S. Arikawa. 字节对编码：一种加速模式匹配的文本压缩方案. 1999.

J. Su, M. Ahmed, Y. Lu, S. Pan, W. Bo, 和 Y. Liu. Roformer：增强型Transformer与旋转位置嵌入. Neurocomputing, 568:127063, 2024.

K. Sun, D. Yu, D. Yu, 和 C. Cardie. 研究先验知识对具有挑战性的中文机器阅读理解的影响, 2019a.

M. Sun, X. Chen, J. Z. Kolter, 和 Z. Liu. 大型语言模型中的大规模激活。arXiv 预印本 arXiv:2402.17762, 2024。

X. Sun, J. Choi, C.-Y. Chen, N. Wang, S. Venkataramani, V. V. Srinivasan, X. Cui, W. Zhang, 和 K. Gopalakrishnan. 深度神经网络的混合 8 位浮点 (HFP8) 训练和推理。神经信息处理系统进展, 32, 2019b。

M. Suzgun, N. Scales, N. Schärli, S. Gehrmann, Y. Tay, H. W. Chung, A. Chowdhery, Q. V. Le, E. H. Chi, D. Zhou, 等。具有挑战性的 big-bench 任务以及思维链是否可以解决它们。arXiv 预印本 arXiv:2210.09261, 2022。

V. Thakkar, P. Ramani, C. Cecka, A. Shivam, H. Lu, E. Yan, J. Kosaian, M. Hoemmen, H. Wu, A. Kerr, M. Nicely, D. Merrill, D. Blasig, F. Qiao, P. Majcher, P. Springer, M. Hohnerbach, J. Wang, 和M. Gupta。CUTLASS, 2023年1月。URL https://github.com/NVIDIA/cutlass。

H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, 等. LLaMA：开放且高效的基础语言模型. arXiv 预印本 arXiv:2302.13971, 2023a.

H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale, D. Bikel, L. Blecher, C. Canton-Ferrer, M. Chen, G. Cucurull, D.Esiobu, J. Fernandes, J. Fu, W. Fu, B. Fuller, C. Gao, V. Goswami, N. Goyal, A. Hartshorn, S. Hosseini,

R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P. S. Koura, M. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith, R. Subramanian, X. E. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan, I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, and T. Scialom. Llama 2: Open foundation and fine-tuned chat models. CoRR, abs/2307.09288, 2023b. doi: 10.48550/arXiv.2307.09288. URL `https://doi.org/10.48550/arXiv.2307.09288`.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. Advances in neural information processing systems, 30, 2017.

L. Wang, H. Gao, C. Zhao, X. Sun, and D. Dai. Auxiliary-loss-free load balancing strategy for mixture-of-experts. CoRR, abs/2408.15664, 2024a. URL `https://doi.org/10.48550/arXiv.2408.15664`.

Y. Wang, X. Ma, G. Zhang, Y. Ni, A. Chandra, S. Guo, W. Ren, A. Arulraj, X. He, Z. Jiang, T. Li, M. Ku, K. Wang, A. Zhuang, R. Fan, X. Yue, and W. Chen. Mmlu-pro: A more robust and challenging multi-task language understanding benchmark. CoRR, abs/2406.01574, 2024b. URL `https://doi.org/10.48550/arXiv.2406.01574`.

T. Wei, J. Luan, W. Liu, S. Dong, and B. Wang. Cmath: Can your language model pass chinese elementary school math test?, 2023.

M. Wortsman, T. Dettmers, L. Zettlemoyer, A. Morcos, A. Farhadi, and L. Schmidt. Stable and low-precision training for large-scale vision-language models. Advances in Neural Information Processing Systems, 36:10271–10298, 2023.

H. Xi, C. Li, J. Chen, and J. Zhu. Training transformers with 4-bit integers. Advances in Neural Information Processing Systems, 36:49146–49168, 2023.

C. S. Xia, Y. Deng, S. Dunn, and L. Zhang. Agentless: Demystifying llm-based software engineering agents. arXiv preprint, 2024.

H. Xia, T. Ge, P. Wang, S. Chen, F. Wei, and Z. Sui. Speculative decoding: Exploiting speculative execution for accelerating seq2seq generation. In Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, December 6-10, 2023, pages 3909–3925. Association for Computational Linguistics, 2023. URL `https://doi.org/10.18653/v1/2023.findings-emnlp.257`.

G. Xiao, J. Lin, M. Seznec, H. Wu, J. Demouth, and S. Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In International Conference on Machine Learning, pages 38087–38099. PMLR, 2023.

L. Xu, H. Hu, X. Zhang, L. Li, C. Cao, Y. Li, Y. Xu, K. Sun, D. Yu, C. Yu, Y. Tian, Q. Dong, W. Liu, B. Shi, Y. Cui, J. Li, J. Zeng, R. Wang, W. Xie, Y. Li, Y. Patterson, Z. Tian, Y. Zhang, H. Zhou, S. Liu, Z. Zhao, Q. Zhao, C. Yue, X. Zhang, Z. Yang, K. Richardson, and Z. Lan. CLUE: A chinese language understanding evaluation benchmark. In D. Scott, N. Bel, and C. Zong, editors, Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Barcelona, Spain (Online), December 8-13, 2020, pages 4762–4772. International Committee on Computational Linguistics, 2020. doi: 10.18653/V1/2020.COLING-MAIN.419. URL `https://doi.org/10.18653/v1/2020.coling-main.419`.

R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P. S. Koura, M. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith, R. Subramanian, X. E. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan, I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, and T. Scialom. Llama 2: Open foundation and fine-tuned chat models. CoRR, abs/2307.09288, 2023b. doi: 10.48550/arXiv.2307.09288. URL `https://doi.org/10.48550/arXiv.2307.09288`.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polo-suukhin. 注意力机制足以应对一切。神经信息处理系统进展，30，2017.

L. Wang, H. Gao, C. Zhao, X. Sun, 和 D. Dai. 无辅助损失的专家混合负载均衡策略。CoRR, abs/2408.15664, 2024a. URL `https://doi.org/10.48550/arXiv.2408.15664`

王宇，马晓，张刚，倪宇，Chandra阿南，郭思，任伟，Arulraj阿尼尔，何翔，蒋子，李涛，Ku马库，王凯，庄阿，范瑞，岳晓，陈伟。Mmlu-pro：一个更健壮和具有挑战性的多任务语言理解基准。CoRR，abs/2406.01574，2024b。URL `https://doi.org/10.48550/arXiv.2406.01574`

T. Wei, J. Luan, W. Liu, S. Dong, 和 B. Wang. Cmath: 你的语言模型能通过中国小学数学测试吗?, 2023。

M. Wortsman, T. Dettmers, L. Zettlemoyer, A. Morcos, A. Farhadi, 和 L. Schmidt. 大规模视觉语言模型的稳定和低精度训练. Neural Information Processing Systems 的进展, 36:10271–10298, 2023。

H. Xi, C. Li, J. Chen, 和 J. Zhu. 使用 4 位整数的 Transformer 训练. Neural Infor mation Processing Systems 的进展, 36:49146–49168, 2023。

C. S. Xia, Y. Deng, S. Dunn, 和 L. Zhang. Agentless: 揭开基于 llm 的软件工程代理的神秘面纱. arXiv 预印本, 2024。

H. Xia, T. Ge, P. Wang, S. Chen, F. Wei, 和 Z. Sui. 推测解码：利用推测执行加速 seq2seq 生成. 在计算语言学协会发现：EMNLP 2023, 新加坡, 2023 年 12 月 6-10 日, 第 3909–3925 页. 计算语言学协会 , 2023. URL `https://doi.org/10.18653/v1/2023.findings-emnlp.257`。

Xiao G, Lin J, Seznec M, Wu H, Demouth J, and Han S. Smoothquant: 大型语言模型的高精度高效后训练量化方法. In International Conference on Machine Learning, pages 38087–38099. PMLR, 2023.

Xu L, Hu H, Zhang X, Li L, Cao C, Li Y, Xu Y, Sun K, Yu D, Yu C, Tian Y, Dong Q, Liu W, Shi B, Cui Y, Li J, Zeng J, Wang R, Xie W, Li Y, Patterson Y, Tian Z, Zhang Y, Zhou H, Liu S, Zhao Z, Zhao Q, Yue C, Zhang X, Yang Z, Richardson K, and Lan Z. CLUE: 一个中文语言理解评估基准. In D. Scott, N. Bel, and C. Zong, editors, Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020, Spain (Online), December 8-13, 2020, pages 4762–4772. International Committee on Computational Linguistics, 2020. doi: 10.18653/V1/2020.COLING-MAIN.419. URL `https://doi.org/10.18653/v1/2020.coling-main.419`.

R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, and Y. Choi. HellaSwag: Can a machine really finish your sentence? In A. Korhonen, D. R. Traum, and L. Màrquez, editors, Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers, pages 4791–4800. Association for Computational Linguistics, 2019. doi: 10.18653/v1/p19-1472. URL `https://doi.org/10.18653/v1/p19-1472`.

W. Zhong, R. Cui, Y. Guo, Y. Liang, S. Lu, Y. Wang, A. Saied, W. Chen, and N. Duan. AGIEval: A human-centric benchmark for evaluating foundation models. CoRR, abs/2304.06364, 2023. doi: 10.48550/arXiv.2304.06364. URL `https://doi.org/10.48550/arXiv.2304.06364`.

J. Zhou, T. Lu, S. Mishra, S. Brahma, S. Basu, Y. Luan, D. Zhou, and L. Hou. Instruction-following evaluation for large language models. arXiv preprint arXiv:2311.07911, 2023.

R. Zellers, A. Holtzman, Y. Bisk, A. Farhadi, 和 Y. Choi. HellaSwag: 机器真的能帮你完成句子吗？在 A. Korhonen, D. R. Traum, 和 L. Màrquez 编的《第 57 届计算语言学协会会议论文集》，ACL 2019，意大利佛罗伦萨，2019 年 7 月 28 日-8 月 2 日，第一卷：长论文，第 4791–4800 页。计算语言学协会，2019 年。doi: 10.18653/v1/p19-1472. URL `https://doi.org/10.18653/v1/p19-1472`.

W. Zhong, R. Cui, Y. Guo, Y. Liang, S. Lu, Y. Wang, A. Saied, W. Chen, 和 N. Duan. AGIEval: 一个以人为中心的基准，用于评估基础模型。CoRR, abs/2304.06364, 2023. doi: 10.48550/arXiv.2304.06364. URL `https://doi.org/10.48550/arXiv.2304.06364`.

J. Zhou, T. Lu, S. Mishra, S. Brahma, S. Basu, Y. Luan, D. Zhou, and L. Hou. 大型语言模型的指令遵循评估。arXiv 预印本 arXiv:2311.07911, 2023。

Appendix

A. Contributions and Acknowledgments

Research & Engineering
Aixin Liu
Bing Xue
Bingxuan Wang
Bochao Wu
Chengda Lu
Chenggang Zhao
Chengqi Deng
Chenyu Zhang*
Chong Ruan
Damai Dai
Daya Guo
Dejian Yang
Deli Chen
Erhang Li
Fangyun Lin
Fucong Dai
Fuli Luo*
Guangbo Hao
Guanting Chen
Guowei Li
H. Zhang
Han Bao*
Hanwei Xu
Haocheng Wang*
Haowei Zhang
Honghui Ding
Huajian Xin*
Huazuo Gao
Hui Qu
Jianzhong Guo
Jiashi Li
Jiawei Wang*
Jingchang Chen
Jingyang Yuan
Junjie Qiu
Junlong Li
Junxiao Song
Kai Dong
Kai Hu*
Kaige Gao
Kang Guan
Kexin Huang
Kuai Yu
Lean Wang
Lecong Zhang
Liang Zhao
Litong Wang
Liyue Zhang
Mingchuan Zhang
Minghua Zhang
Minghui Tang
Panpan Huang
Peiyi Wang
Qiancheng Wang
Qihao Zhu
Qinyu Chen
Qiushi Du
Ruiqi Ge
Ruisong Zhang
Ruizhe Pan
Runji Wang
Runxin Xu
Ruoyu Zhang
Shanghao Lu
Shangyan Zhou
Shanhuang Chen
Shengfeng Ye
Shirong Ma
Shiyu Wang
Shuiping Yu
Shunfeng Zhou
Shuting Pan
Tao Yun
Tian Pei
Wangding Zeng
Wanjia Zhao*
Wen Liu
Wenfeng Liang
Wenjun Gao
Wenqin Yu
Wentao Zhang
Xiao Bi
Xiaodong Liu
Xiaohan Wang
Xiaokang Chen
Xiaokang Zhang
Xiaotao Nie
Xin Cheng
Xin Liu

附录

A. 贡献与致谢

研究与工程
艾欣 刘冰 学兵
王博超 吴成达 陆
成刚 赵成奇 邓晨
宇 张* 钟崇 车大
卫 戴大雅 郭德建
杨德力 陈尔航 李
方云 林福丛 戴福
立 罗* 郝广波 陈
冠霆 陈国伟 李
H. 张 韩宝* 韩伟
徐浩成 王浩 张
浩伟 丁宏伟 辛华
健* 高华祖 赵会
仇建中 国嘉石 李
嘉伟 王* 陈景长
陈景阳 袁俊杰 邱
俊龙 李俊晓 宋凯
东 宋凯胡* 高凯
歌 韩关 康 王克
新 黄快宇 王
Lean

张乐丛 张亮 赵立
通 王立悦 张明川
张明华 张明辉 唐
攀攀 黄沛怡 王千
程 王启浩 朱启宇
陈奇淑 杜瑞琪 诸
葛瑞松 张瑞哲 潘
润芝 王润欣 徐若
宇 张双昊 陆上好
周上岩 周山黄 陈
胜峰 叶诗荣 马时
雨 王淑萍 余顺峰
周淑婷 潘涛 云天
魏王定 曾万嘉 赵
文* 刘文 文峰梁 文
俊高 余文钦 张文
涛 张晓北 刘晓东
王晓寒 王晓康 陈
晓康 张晓康 韩晓
涛 那欣 成欣 刘欣

Xin Xie
Xingchao Liu
Xingkai Yu
Xinyu Yang
Xinyuan Li
Xuecheng Su
Xuheng Lin
Y.K. Li
Y.Q. Wang
Y.X. Wei
Yang Zhang
Yanhong Xu
Yao Li
Yao Zhao
Yaofeng Sun
Yaohui Wang
Yi Yu
Yichao Zhang
Yifan Shi
Yiliang Xiong
Ying He
Yishi Piao
Yisong Wang
Yixuan Tan
Yiyang Ma*
Yiyuan Liu
Yongqiang Guo
Yu Wu
Yuan Ou
Yuduan Wang
Yue Gong
Yuheng Zou
Yujia He
Yunfan Xiong
Yuxiang Luo
Yuxiang You
Yuxuan Liu
Yuyang Zhou
Z.F. Wu
Z.Z. Ren
Zehui Ren
Zhangli Sha
Zhe Fu
Zhean Xu
Zhenda Xie
Zhengyan Zhang
Zhewen Hao
Zhibin Gou
Zhicheng Ma
Zhigang Yan
Zhihong Shao
Zhiyu Wu
Zhuoshu Li
Zihui Gu
Zijia Zhu
Zijun Liu*
Zilin Li
Ziwei Xie
Ziyang Song
Ziyi Gao
Zizheng Pan

Data Annotation

Bei Feng
Hui Li
J.L. Cai
Jiaqi Ni
Lei Xu
Meng Li
Ning Tian
R.J. Chen
R.L. Jin
Ruyi Chen
S.S. Li
Shuang Zhou
Tianyu Sun
X.Q. Li
Xiangyue Jin
Xiaojin Shen
Xiaosha Chen
Xiaowen Sun
Xiaoxiang Wang
Xinnan Song
Xinyi Zhou
Y.X. Zhu
Yanhong Xu
Yanping Huang
Yaohui Li
Yi Zheng
Yuchen Zhu
Yunxian Ma
Zhen Huang
Zhipeng Xu
Zhongyu Zhang

Business & Compliance

Dongjie Ji

徐晓行超 刘行开
余新宇 杨新元 李
学成 苏旭恒 林
Y.K. 李Y.Q. 王
Y.X. 魏阳 张彦红
徐瑶 李瑶 赵亚峰
孙亚辉 王毅 余一
超 张一帆 石一良
邢英 何一石 普一
松 王一轩 谭一阳
马 * 一元 刘永强
郭宇 吴源 欧宇段
王越 龚宇恒 左宇
嘉 何云帆 邢宇翔
罗宇翔 尤宇轩 刘
宇阳 周Z.F. 吴
Z.Z. 任泽辉 任张
立 沙哲 冯 赵富
赵安 许振达 谢正
炎 张哲文 郝志斌
骆志成 马
严志刚 严志
宏 邵志宇 吴
卓舒 李子辉
古子嘉 朱子
君 刘* 子林
李子伟 谢子
阳 宋子一 高
子正 潘

数据标注北风会
李 J.L. 蔡嘉琪倪
蕾徐萌李宁天
R.J. 陈 R.L. 金瑞
怡陈 S.S. 李双周
天宇孙 X.Q. 李向
岳金晓金沈晓莎
陈晓文孙晓翔王
欣南宋新怡周
Y.X. 朱艳红徐艳
平黄耀辉李艺郑
宇朱云仙马振黄
志鹏徐钟宇张

业务与合规东杰姬

Jian Liang
Jin Chen
Leyi Xia
Miaojun Wang
Mingming Li
Peng Zhang
Shaoqing Wu
Shengfeng Ye
T. Wang
W.L. Xiao
Wei An
Xianzu Wang
Xinxia Shan
Ying Tang
Yukun Zha
Yuting Yan
Zhen Zhang

Within each role, authors are listed alphabetically by the first name. Names marked with * denote individuals who have departed from our team.

B. Ablation Studies for Low-Precision Training

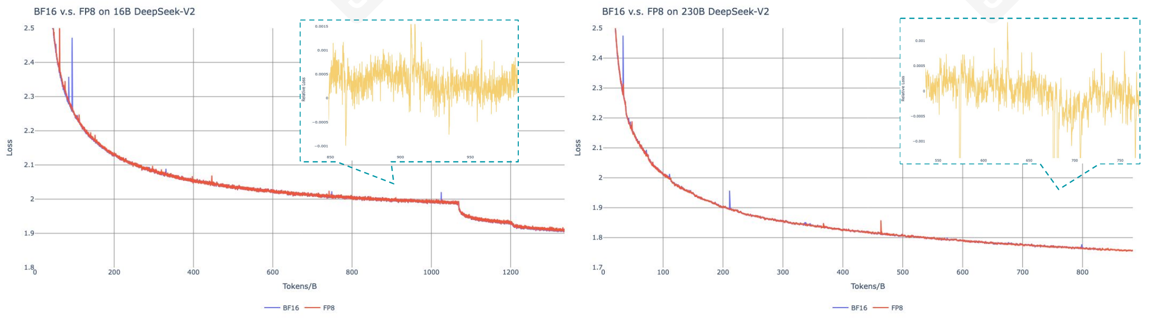


Figure 10 | Loss curves comparison between BF16 and FP8 training. Results are smoothed by Exponential Moving Average (EMA) with a coefficient of 0.9.

B.1. FP8 v.s. BF16 Training

We validate our FP8 mixed precision framework with a comparison to BF16 training on top of two baseline models across different scales. At the small scale, we train a baseline MoE model comprising approximately 16B total parameters on 1.33T tokens. At the large scale, we train a baseline MoE model comprising approximately 230B total parameters on around 0.9T tokens. We show the training curves in Figure 10 and demonstrate that the relative error remains below 0.25% with our high-precision accumulation and fine-grained quantization strategies.

B.2. Discussion About Block-Wise Quantization

Although our tile-wise fine-grained quantization effectively mitigates the error introduced by feature outliers, it requires different groupings for activation quantization, i.e., `1x128` in forward pass and `128x1` for backward pass. A similar process is also required for the activation gradient. A straightforward strategy is to apply block-wise quantization per `128x128` elements like the way we quantize the model weights. In this way, only transposition is required for backward. Therefore, we conduct an experiment where all tensors associated with `Dgrad` are quantized on a block-wise basis. The results reveal that the `Dgrad` operation which computes the activation gradients and back-propagates to shallow layers in a chain-like manner, is highly sensitive to precision. Specifically, block-wise quantization of activation gradients leads to

Jian Liang
Jin Chen
Leyi Xia
Miaojun Wang
Mingming Li
Peng Zhang
Shaoqing Wu
Shengfeng Ye
T. Wang
W.L. Xiao
Wei An
Xianzu Wang
Xinxia Shan
Ying Tang
Yukun Zha
Yuting Yan
Zhen Zhang

在每个角色中，作者按名字首字母排序列出。带*标记的名字表示已离开我们团队的成员。

B. 低精度训练的消融研究

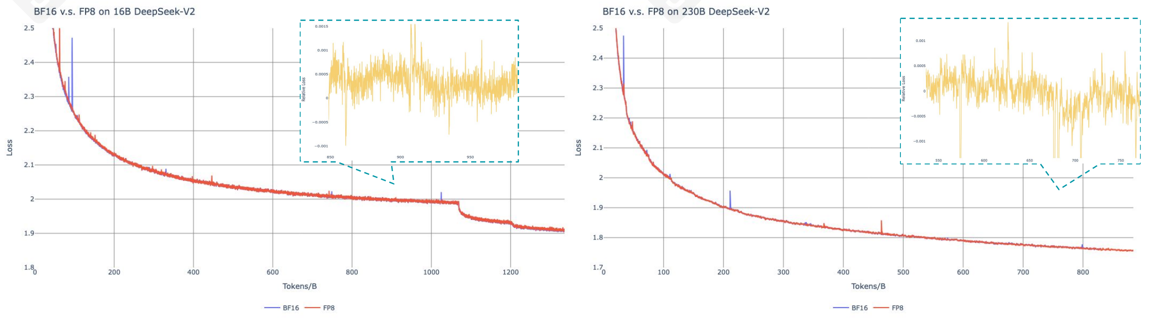


图 10 | BF16 和 FP8 训练的损失曲线对比。结果通过系数为 0.9 的指数移动平均 (EMA) 平滑处理。

B.1. FP8 v.s. BF16 训练

我们在不同规模上使用 BF16 训练对两个基线模型验证了我们的 FP8 混合精度框架。在小规模上，我们在 1.33T tokens 上训练一个包含约 16B 总参数的基线 MoE 模型。在大规模上，我们在约 0.9T tokens 上训练一个包含约 230B 总参数的基线 MoE 模型。我们在图 10 中展示了训练曲线，并证明通过我们的高精度累积和细粒度量化策略，相对误差保持在 0.25% 以下。

B.2. 关于分块量化的讨论

尽管我们的逐瓦片细粒度量化有效地减轻了特征异常值引入的错误，但它需要不同的激活量化分组，即在前向传递中使用 `1x128` ，在后向传递中使用 `128x1` 。激活梯度的处理也需要类似的过程。一种直接策略是像我们量化模型权重一样，对 `128x128` 个元素进行分块量化。这样，后向传递只需要转置。因此，我们进行了一项实验，对与 `Dgrad` 相关的所有张量进行分块量化。结果表明，计算激活梯度并在链式方式下反向传播到浅层层的 `Dgrad` 操作对精度高度敏感。具体来说，激活梯度的分块量化会导致

model divergence on an MoE model comprising approximately 16B total parameters, trained for around 300B tokens. We hypothesize that this sensitivity arises because activation gradients are highly imbalanced among tokens, resulting in token-correlated outliers (Xi et al., 2023). These outliers cannot be effectively managed by a block-wise quantization approach.

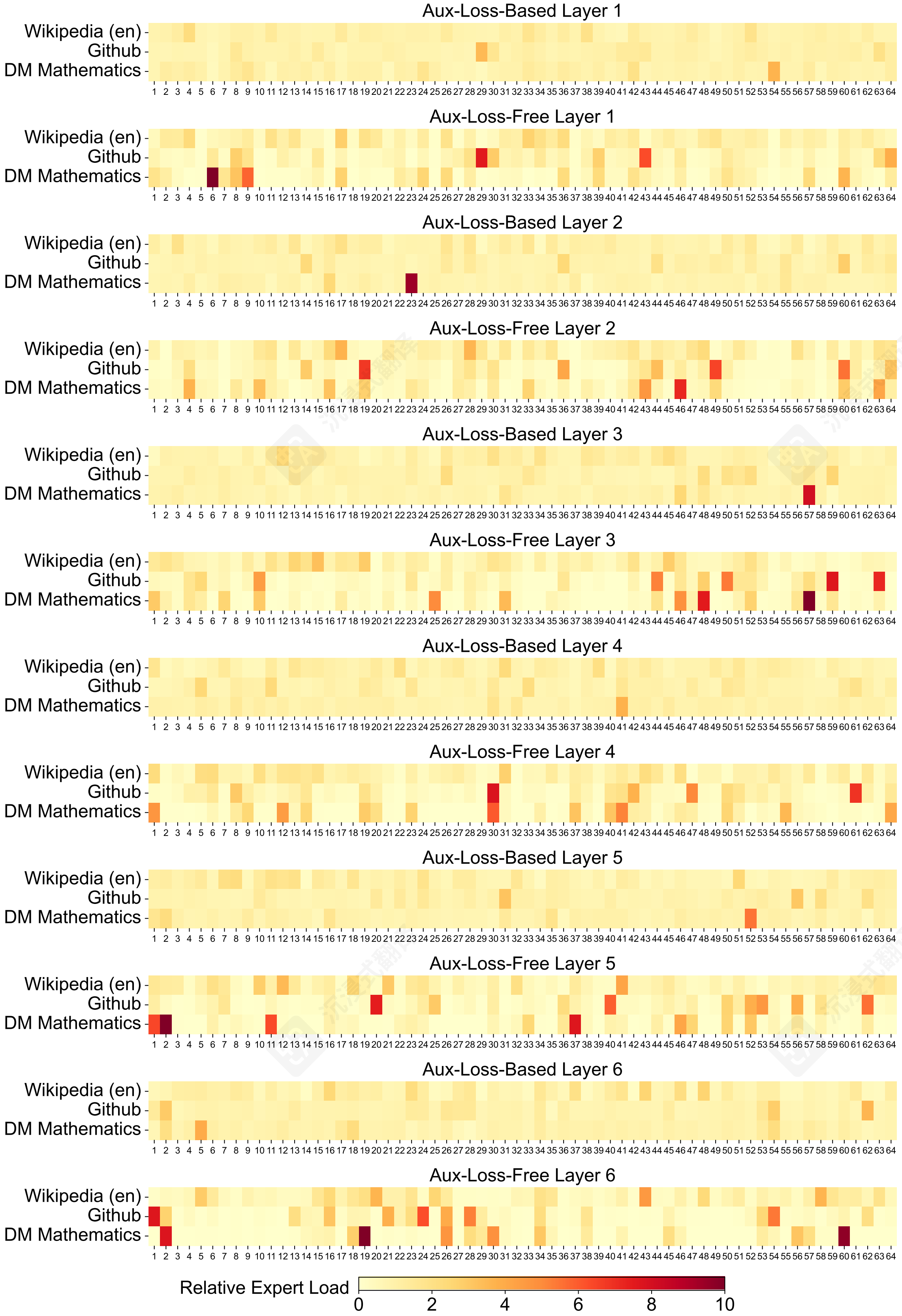
C. Expert Specialization Patterns of the 16B Aux-Loss-Based and Aux-Loss-Free Models

We record the expert load of the 16B auxiliary-loss-based baseline and the auxiliary-loss-free model on the Pile test set. The auxiliary-loss-free model tends to have greater expert specialization across all layers, as demonstrated in Figure 10.

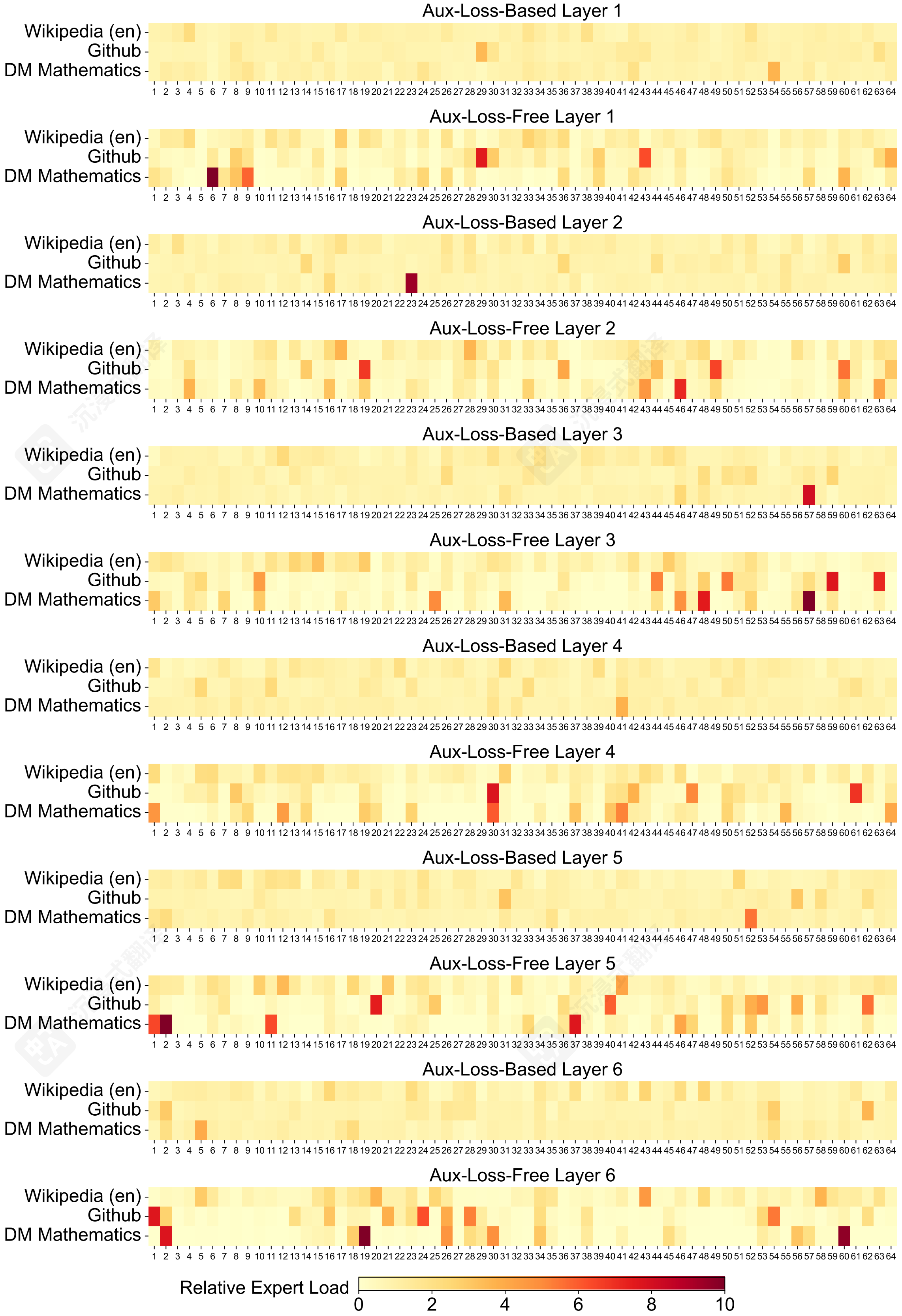
一个包含约 16B 总参数的 MoE 模型上的模型发散，训练了大约 300B 个 token。我们假设这种敏感性产生的原因是激活梯度在 token 之间高度不平衡，导致 token 相关的异常值（Xi 等人，2023 年）。这些异常值无法通过块状量化方法有效管理。

C. 基于 16B 辅助损失和无辅助损失的模型的专家专业化模式

我们在 Pile 测试集上记录了 16B 辅助损失基线模型和无辅助损失模型的专家负载。无辅助损失模型在所有层上倾向于具有更大的专家专业化，如图 10 所示。

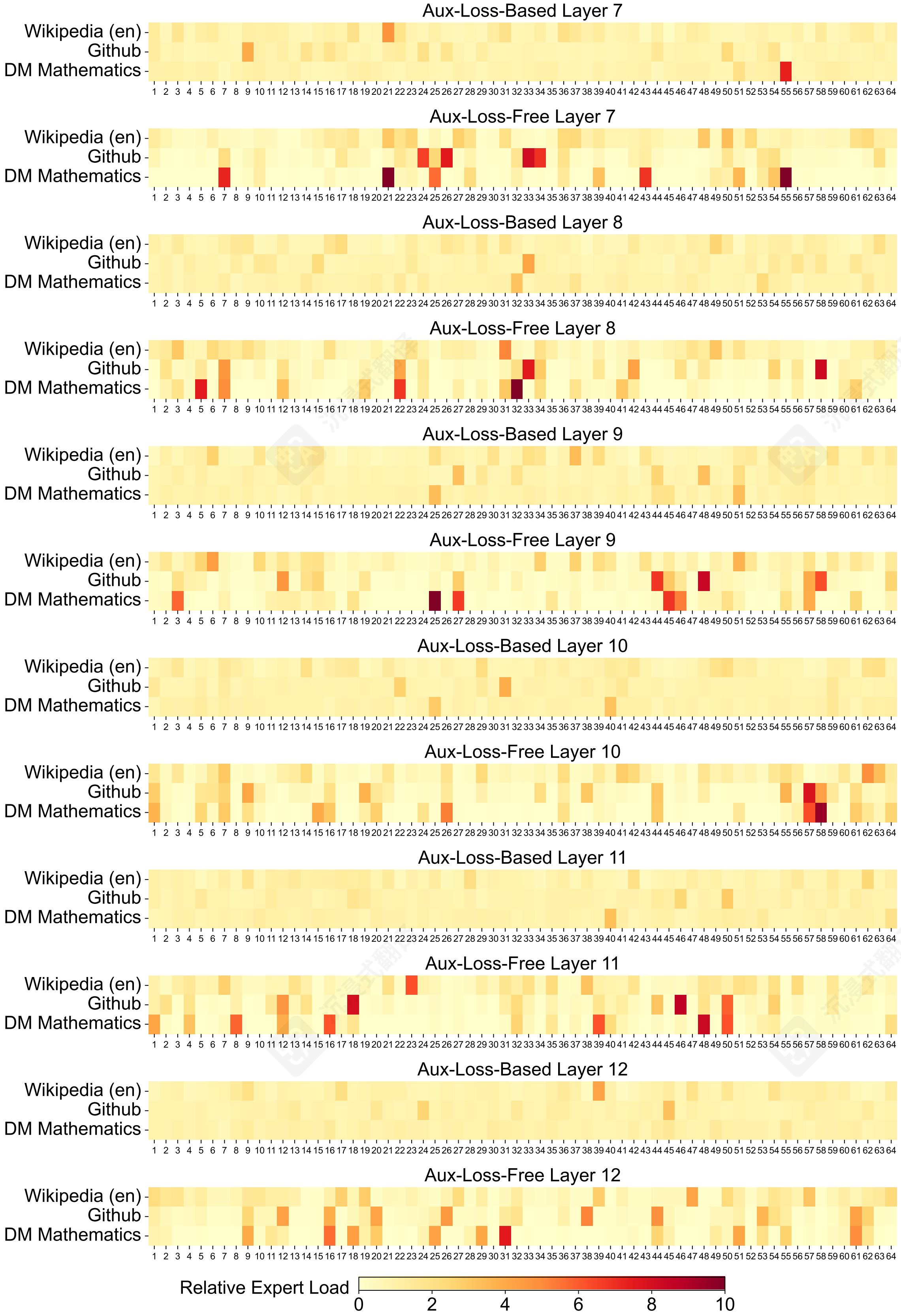


(a) Layers 1-7

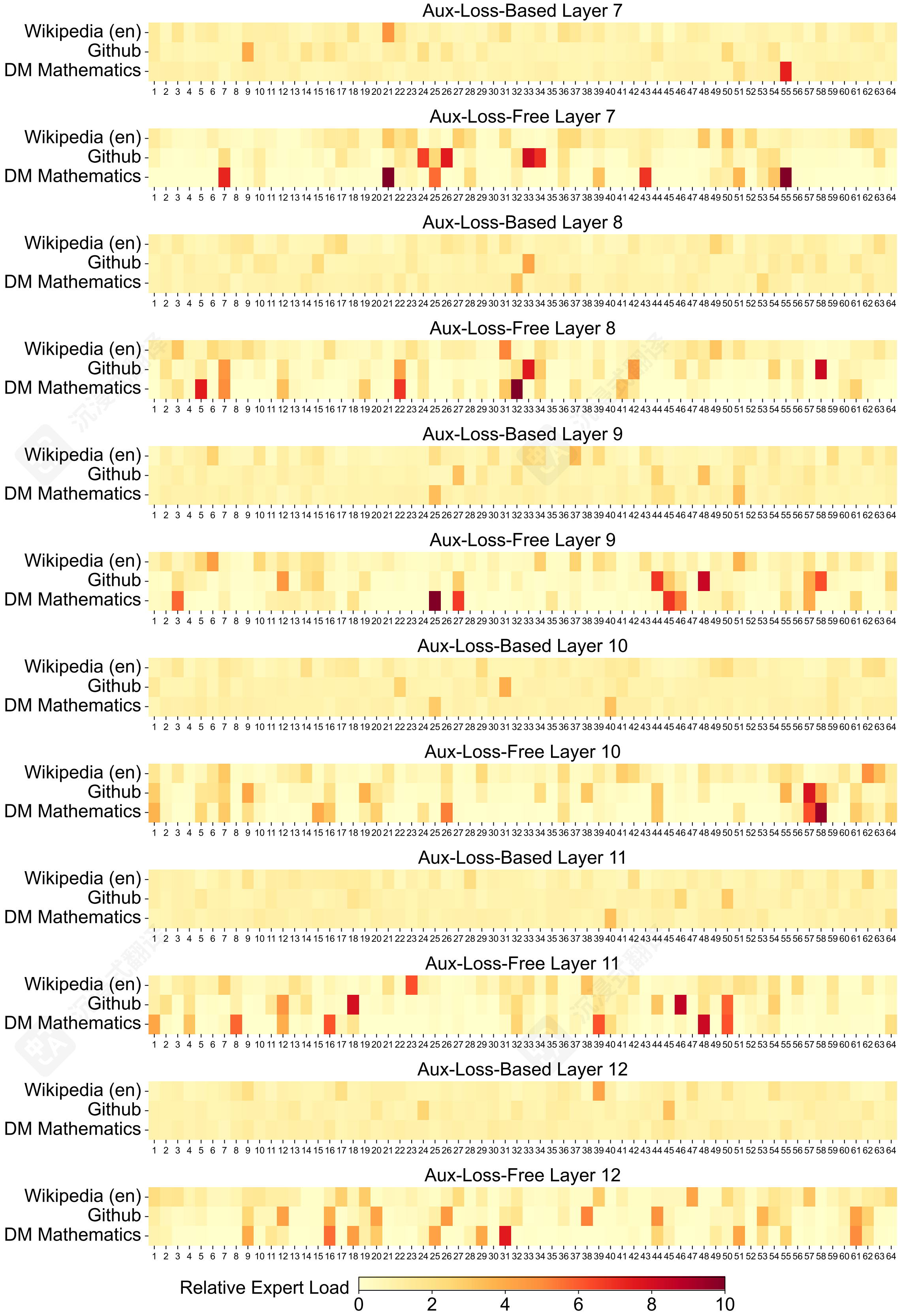


(a) Layers 1-7

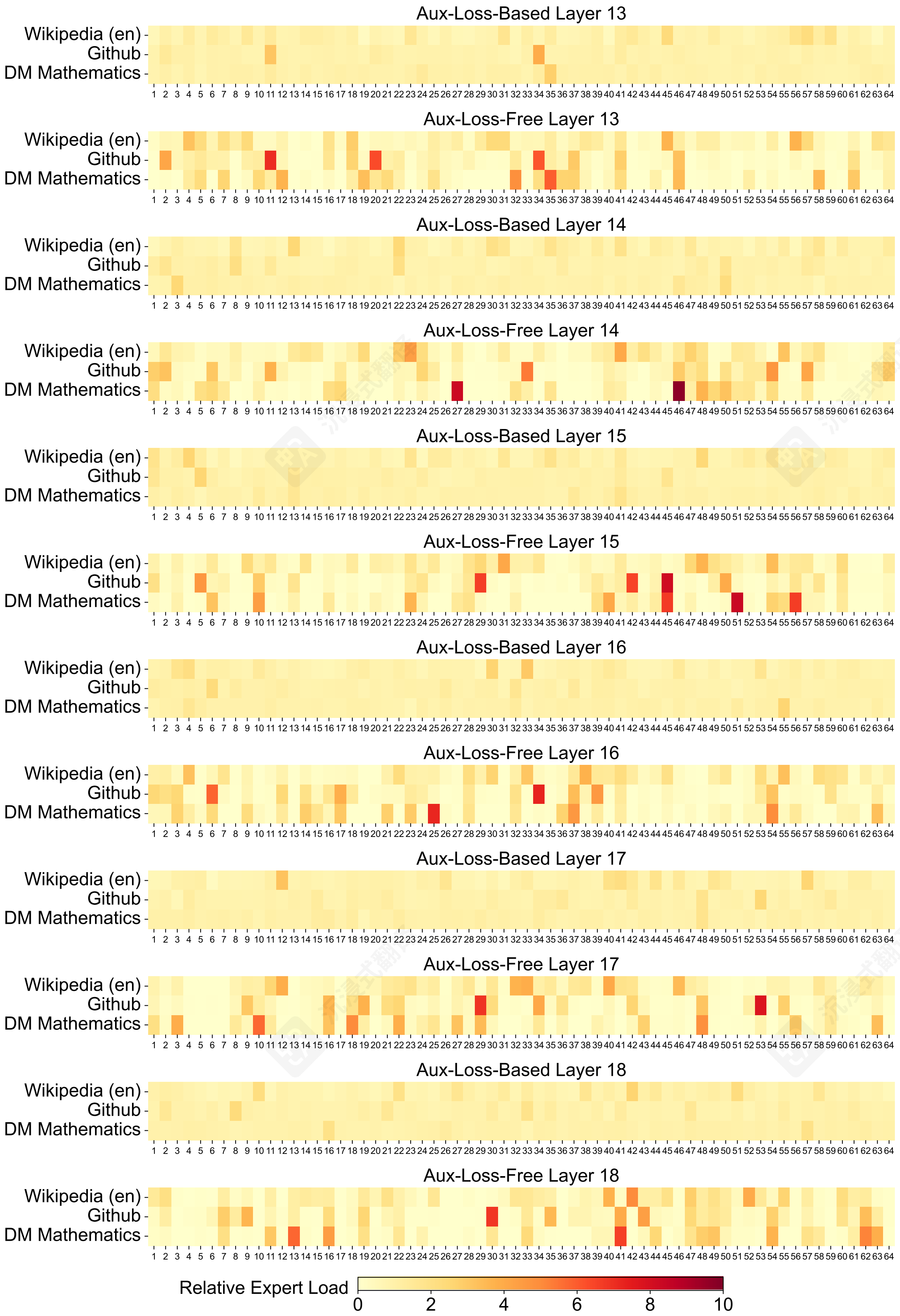
49



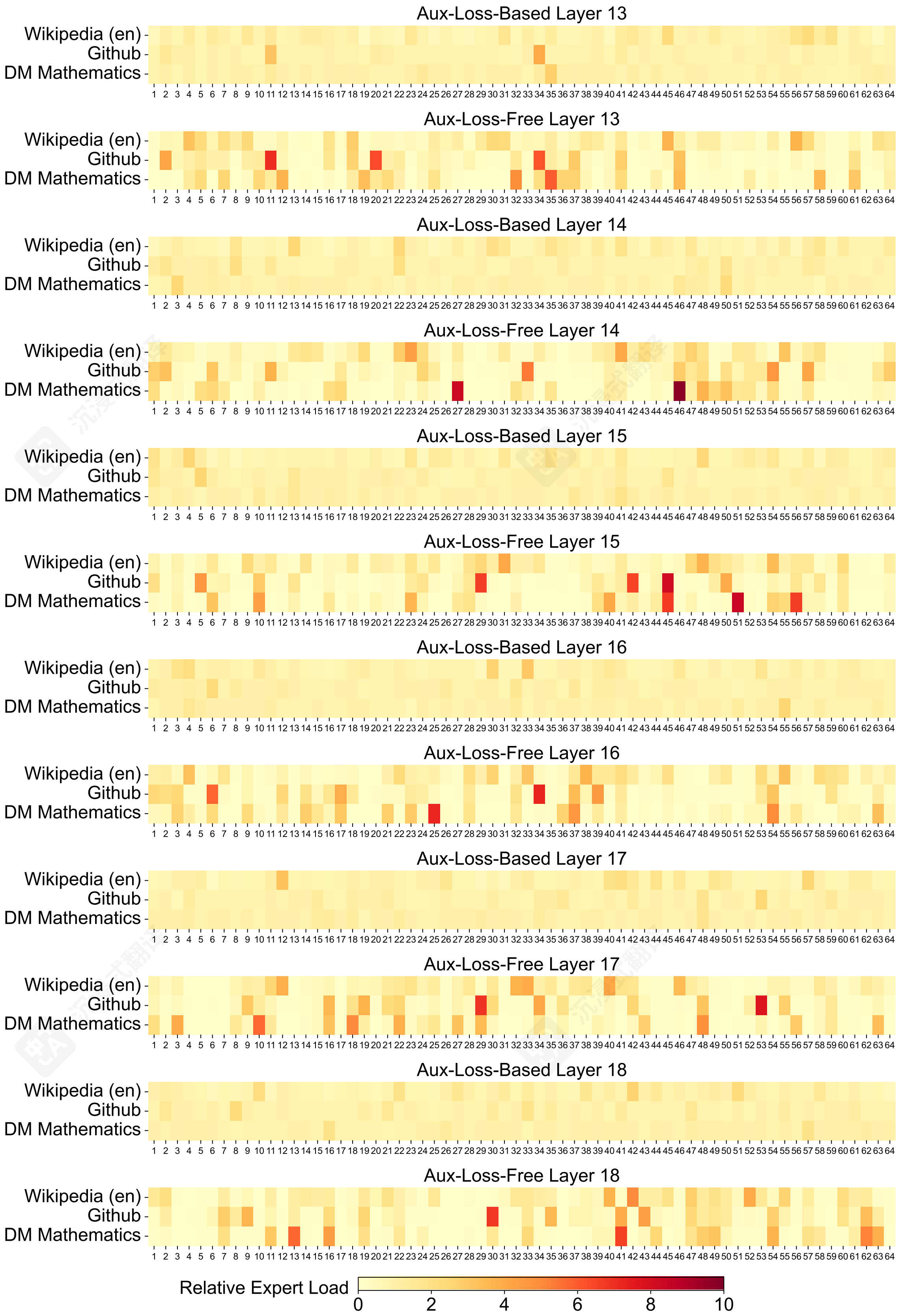
(b) Layers 7-13



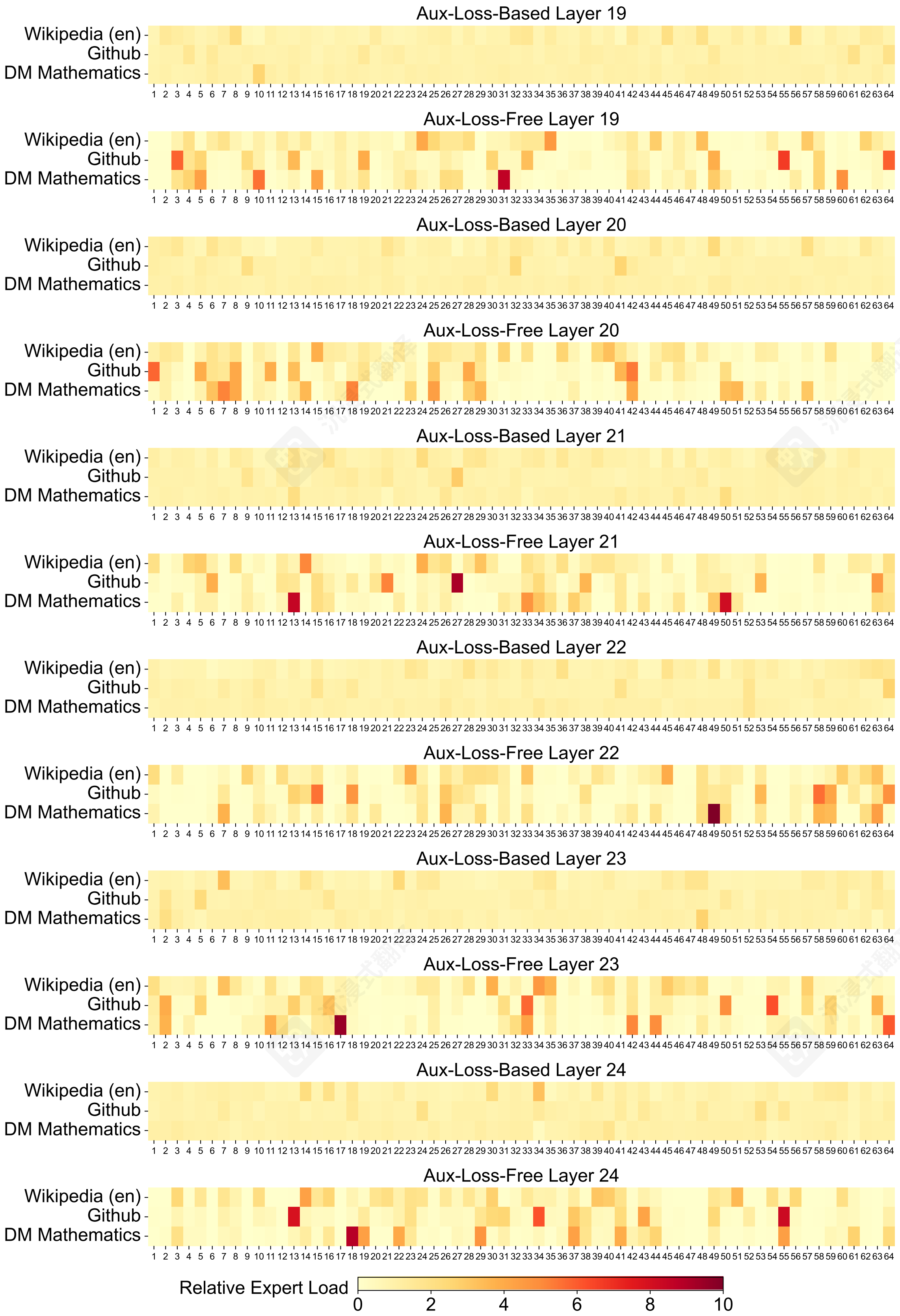
(b) Layers 7-13



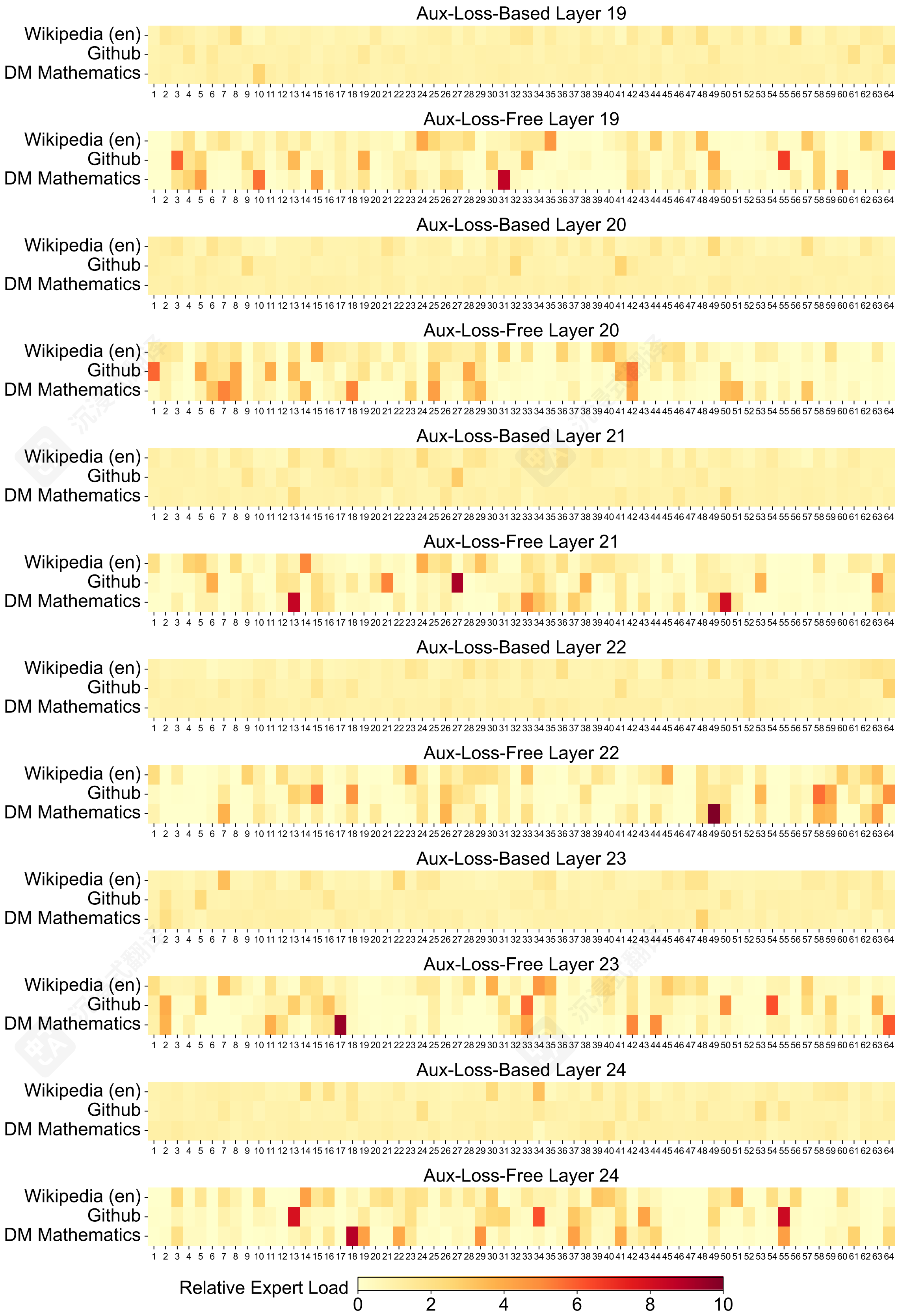
(c) Layers 13-19



(c) Layers 13-19

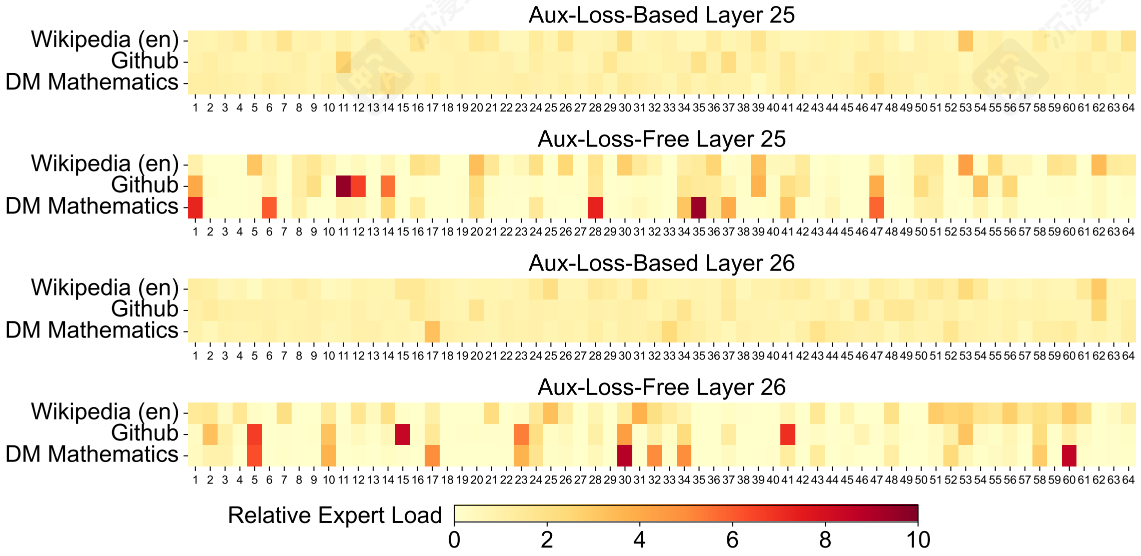


(d) Layers 19-25



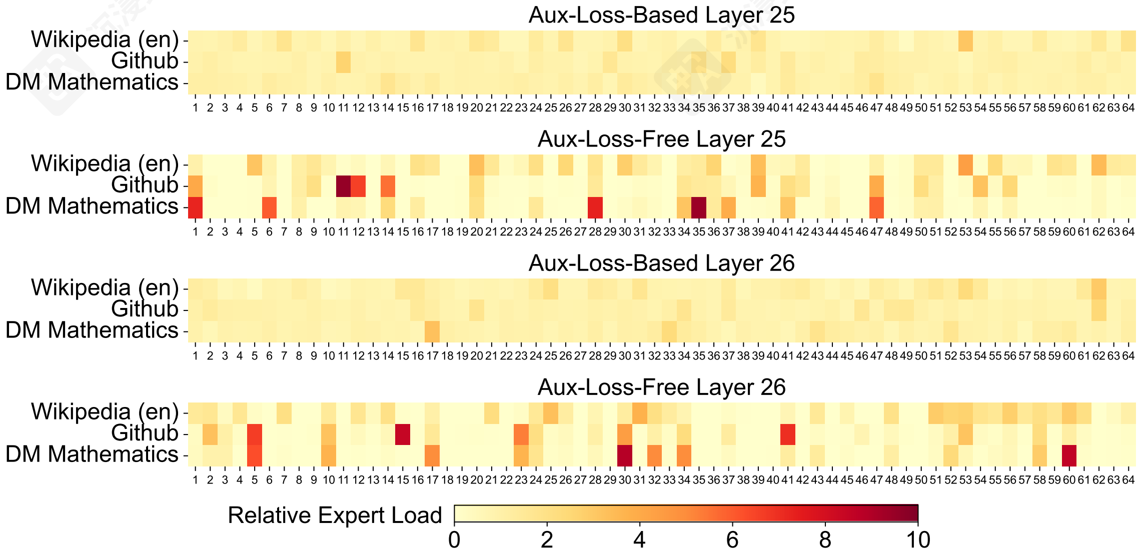
(d) Layers 19-25

52



(e) Layers 25-27

Figure 10 | Expert load of auxiliary-loss-free and auxiliary-loss-based models on three domains in the Pile test set. The auxiliary-loss-free model shows greater expert specialization patterns than the auxiliary-loss-based one. The relative expert load denotes the ratio between the actual expert load and the theoretically balanced expert load.



(e) Layers 25-27

图 10 | 在Pile测试集上，辅助损失模型和无辅助损失模型的专家负载在三个领域上的表现。无辅助损失模型比辅助损失模型表现出更强的专家专业化模式。相对专家负载表示实际专家负载与理论上平衡的专家负载之间的比率。