本文由 AINLP 公众号整理翻译，更多 LLM 资源请扫码关注!

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.

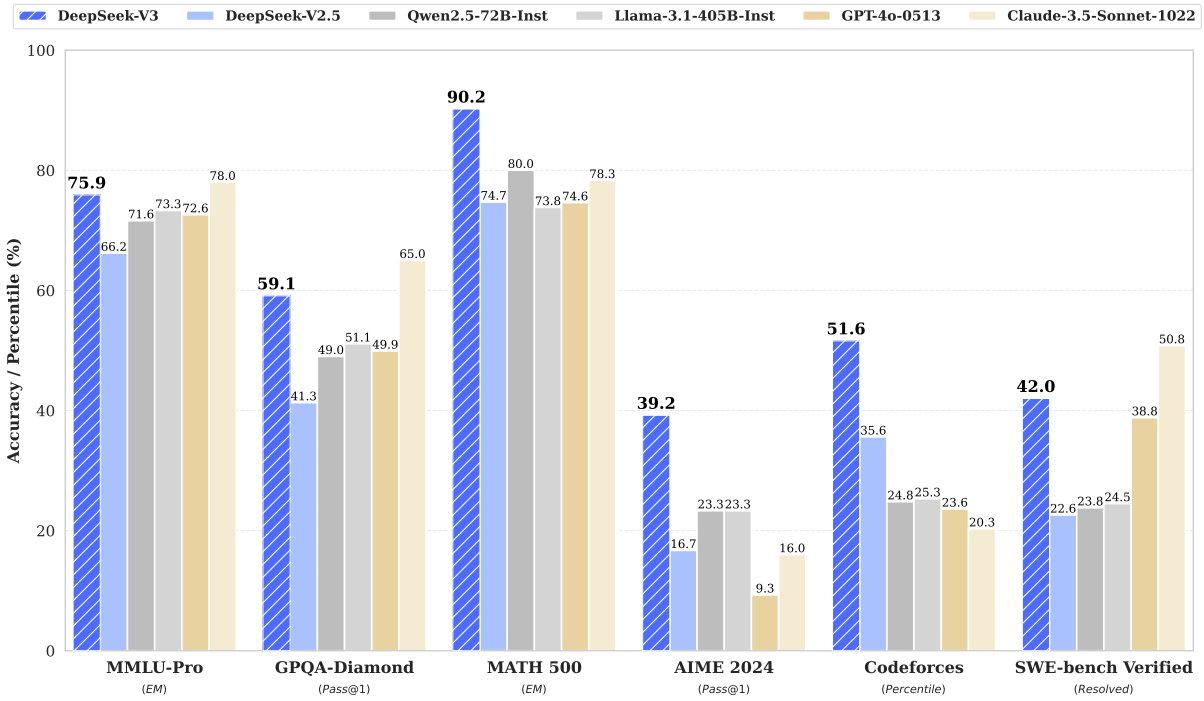


Figure 1 | Benchmark performance of DeepSeek-V3 and its counterparts.

DeepSeek-V3 技术报告

深度搜索-AI

research@deepseek.com

摘要

我们介绍DeepSeek-V3，这是一个强大的混合专家（MoE）语言模型，总参数量为671B，每个token激活37B。为了实现高效推理和具有成本效益的训练，DeepSeek-V3采用了多头潜在注意力（MLA）和DeepSeekMoE架构，这些架构在DeepSeek-V2中得到了充分验证。此外，DeepSeek-V3开创了一种无辅助损失的负载平衡策略，并设定了多token预测训练目标，以实现更强的性能。我们在148万亿个多样化和高质量的token上对DeepSeek-V3进行了预训练，随后进行了监督微调和强化学习阶段，以充分发挥其能力。全面评估表明，DeepSeek-V3的表现优于其他开源模型，并且达到了与领先的闭源模型相当的性能。尽管表现出色，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，我们没有经历任何不可恢复的损失峰值，也没有进行任何回滚。模型检查点可在 https://github.com/deepseek-ai/DeepSeek-V3获取。

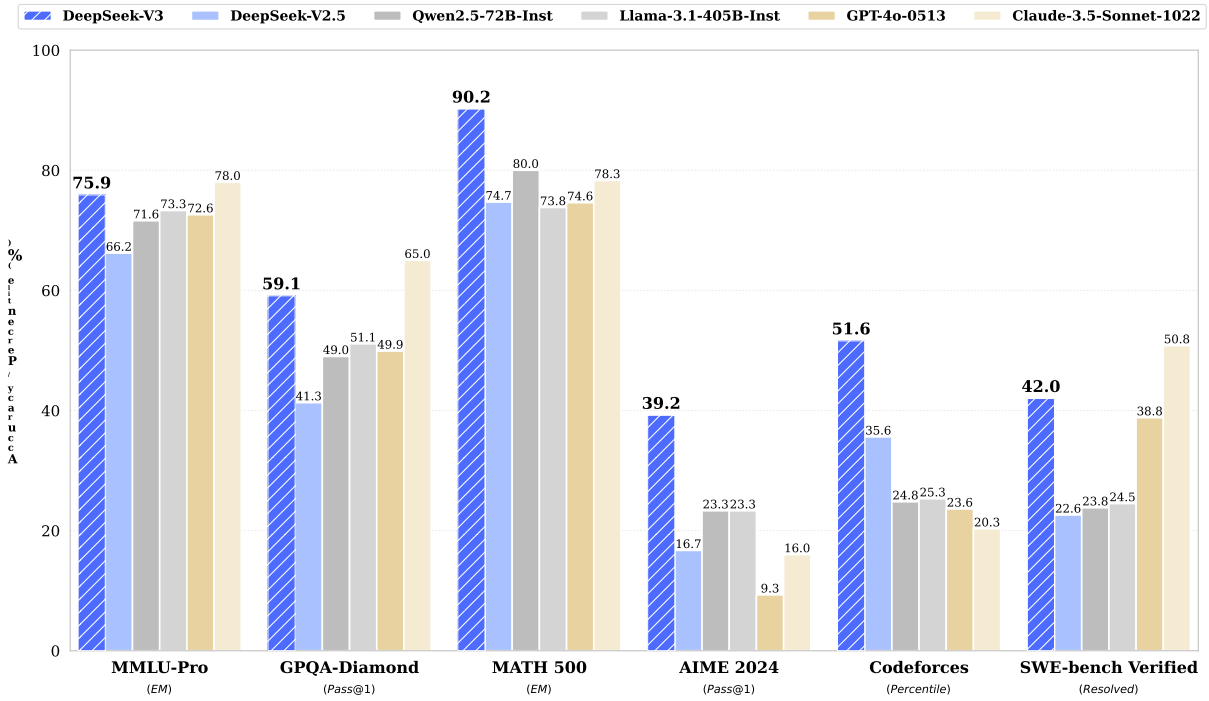


图1 | DeepSeek-V3及其对应模型的基准性能。

Contents

内容

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. 引言

近年来，大型语言模型（LLMs）正在经历快速的迭代和演变（Anthropic, 2024；Google, 2024；OpenAI, 2024a），逐渐缩小与人工通用智能（AGI）之间的差距。除了闭源模型，开源模型，包括DeepSeek系列（DeepSeek-AI, 2024a,b,c；Guo et al., 2024）、LLaMA系列（AI@Meta, 2024a, b；Touvron et al., 2023a,b）、Qwen系列（Qwen, 2023, 2024a,b）和Mistral系列（Jiang et al., 2023；Mistral, 2024），也在取得显著进展，努力缩小与其闭源同行的差距。为了进一步推动开源模型能力的边界，我们扩大了模型规模，并推出了DeepSeek-V3，这是一个具有671B参数的大型专家混合模型（MoE），其中每个token激活37B参数。

从前瞻性的角度来看，我们始终努力追求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）以实现高效推理，并采用 DeepSeekMoE（Dai 等，2024）以实现具有成本效益的训练。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训练和推理的同时保持强大的模型性能。除了基本架构外，我们还实施了两种额外策略，以进一步增强模型能力。首先，DeepSeek-V3 首创了一种无辅助损失策略（Wang 等，2024a）用于负载均衡，旨在最小化因鼓励负载均衡而对模型性能产生的不利影响。其次，DeepSeek-V3 采用了多标记预测训练目标，我们观察到这增强了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，并对训练框架实施全面优化。低精度训练已成为高效训练的一个有前景的解决方案（Dettmers et al., 2022; Kalamkar et al., 2019; Narang et al., 2017; Peng et al., 2023b），其演变与硬件能力的进步密切相关（Luo et al., 2024; Micikevicius et al., 2022; Rouhani et al., 2023a）。在这项工作中，我们引入了FP8混合精度训练框架，并首次验证其在极大规模模型上的有效性。通过对FP8计算和存储的支持，我们实现了加速训练和减少GPU内存使用。至于训练框架，我们设计了DualPipe算法以实现高效的管道并行性，该算法具有更少的管道气泡，并通过计算-通信重叠隐藏了大部分训练过程中的通信。这种重叠确保了，随着模型的进一步扩展，只要我们保持恒定的计算与通信比率，我们仍然可以在节点之间使用细粒度的专家，同时实现近乎零的全到全通信开销。此外，我们还开发了高效的跨节点全到全通信内核，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，使得在不使用昂贵的张量并行性的情况下训练DeepSeek-V3成为可能。结合这些努力，我们实现了高训练效率。

在预训练期间，我们在14.8T高质量和多样化的标记上训练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，我们进行后训练，包括对DeepSeek-V3基础模型的监督微调（SFT）和强化学习（RL），以使其与人类偏好对齐，并进一步释放其潜力。在后训练阶段，我们从DeepSeek-R1系列模型中提炼推理能力，同时仔细保持模型准确性之间的平衡。

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

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

表1 | DeepSeek-V3的训练成本，假设H800的租赁价格为每个GPU小时2美元。

和生成长度。

我们在一系列综合基准上评估了DeepSeek-V3。尽管其训练成本经济，但全面评估显示，DeepSeek-V3-Base已成为目前可用的最强开源基础模型，特别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的闭源模型（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，这得益于我们对算法、框架和硬件的优化共同设计。在预训练阶段，训练DeepSeek-V3每万亿个标记仅需180K H800 GPU小时，即在我们拥有2048个H800 GPU的集群上仅需3.7天。因此，我们的预训练阶段在不到两个月的时间内完成，耗时2664K GPU小时。结合119K GPU小时用于上下文长度扩展和5K GPU小时用于后训练，DeepSeek-V3的完整训练仅需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，不包括与先前研究和架构、算法或数据的消融实验相关的成本。

我们的主要贡献包括：

A架构：创新的负载均衡策略和训练目标

- 在DeepSeek-V2高效架构的基础上，我们开创了一种无辅助损失的负载均衡策略，最小化因促进负载均衡而导致的性能下降。
- 我们研究了多标记预测（MTP）目标，并证明它对模型性能是有益的。它还可以用于推测解码以加速推理。

预训练：迈向终极训练效率

- 我们设计了一个FP8混合精度训练框架，并首次验证了在极大规模模型上进行FP8训练的可行性和有效性。
- 通过算法、框架和硬件的共同设计，我们克服了跨节点 MoE 训练中的通信瓶颈，实现了近乎完全的计算-通信重叠。这显著提高了我们的训练效率并降低了训练成本，使我们能够在没有额外开销的情况下进一步扩大模型规模。
- 以仅需2.664M H800 GPU小时的经济成本，我们在14.8T标记上完成了DeepSeek-V3的预训练，产生了目前最强的开源基础模型。预训练后的后续训练阶段仅需0.1M GPU小时。

后训练：来自DeepSeek-R1的知识蒸馏

- 我们引入了一种创新的方法，将推理能力从长链思维（CoT）模型中提炼出来，特别是从DeepSeek R1系列模型中，转移到标准的LLM中，尤其是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的验证和反思模式引入DeepSeek-V3，并显著提高其推理性能。同时，我们还保持对DeepSeek-V3的输出风格和长度的控制。

核心评估结果摘要

- 知识：（1）在教育基准测试如MMLU、MMLU-Pro和GPQA上，DeepSeek-V3的表现优于所有其他开源模型，在MMLU上获得88.5分，在MMLU-Pro上获得75.9分，在GPQA上获得59.1分。其性能与领先的闭源模型如GPT-4o和Claude-Sonnet-3.5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和中文SimpleQA上表现出色，领先于其他开源模型。尽管在英语事实知识（SimpleQA）上落后于GPT-4o和Claude-Sonnet-3.5，但在中文事实知识（中文SimpleQA）上超越了这些模型，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 在所有非长链思维的开源和闭源模型中，在与数学相关的基准测试中达到了最先进的性能。值得注意的是，它在特定基准测试（如 MATH-500）上甚至超越了 o1-preview，展示了其强大的数学推理能力。（2）在与编码相关的任务中，DeepSeek-V3 成为编码竞赛基准测试（如 LiveCodeBench）的最佳表现模型，巩固了其在该领域的领先地位。在与工程相关的任务中，尽管 DeepSeek-V3 的表现略低于 Claude-Sonnet-3.5，但它仍然以显著的优势超越了所有其他模型，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，我们首先详细介绍我们的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、对FP8训练的支持、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，包括训练数据的构建、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）。之后，我们讨论我们在后训练方面的努力，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的现有局限性，并提出未来研究的潜在方向（第6节）。

2. 架构

我们首先介绍DeepSeek-V3的基本架构，其特点是多头潜在注意力（MLA）（DeepSeek-AI，2024c）用于高效推理，以及DeepSeekMoE（Dai等，2024）用于经济训练。然后，我们提出了一种多标记预测（MTP）训练目标，我们观察到它能增强在评估基准上的整体性能。对于其他未明确提及的细节，DeepSeek-V3遵循DeepSeek-V2（DeepSeek-AI，2024c）的设置。

2.1. 基本架构

DeepSeek-V3的基本架构仍然在Transformer（Vaswani等，2017）框架内。为了高效推理和经济训练，DeepSeek-V3还采用了MLA和DeepSeekMoE，这些在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的负载平衡。

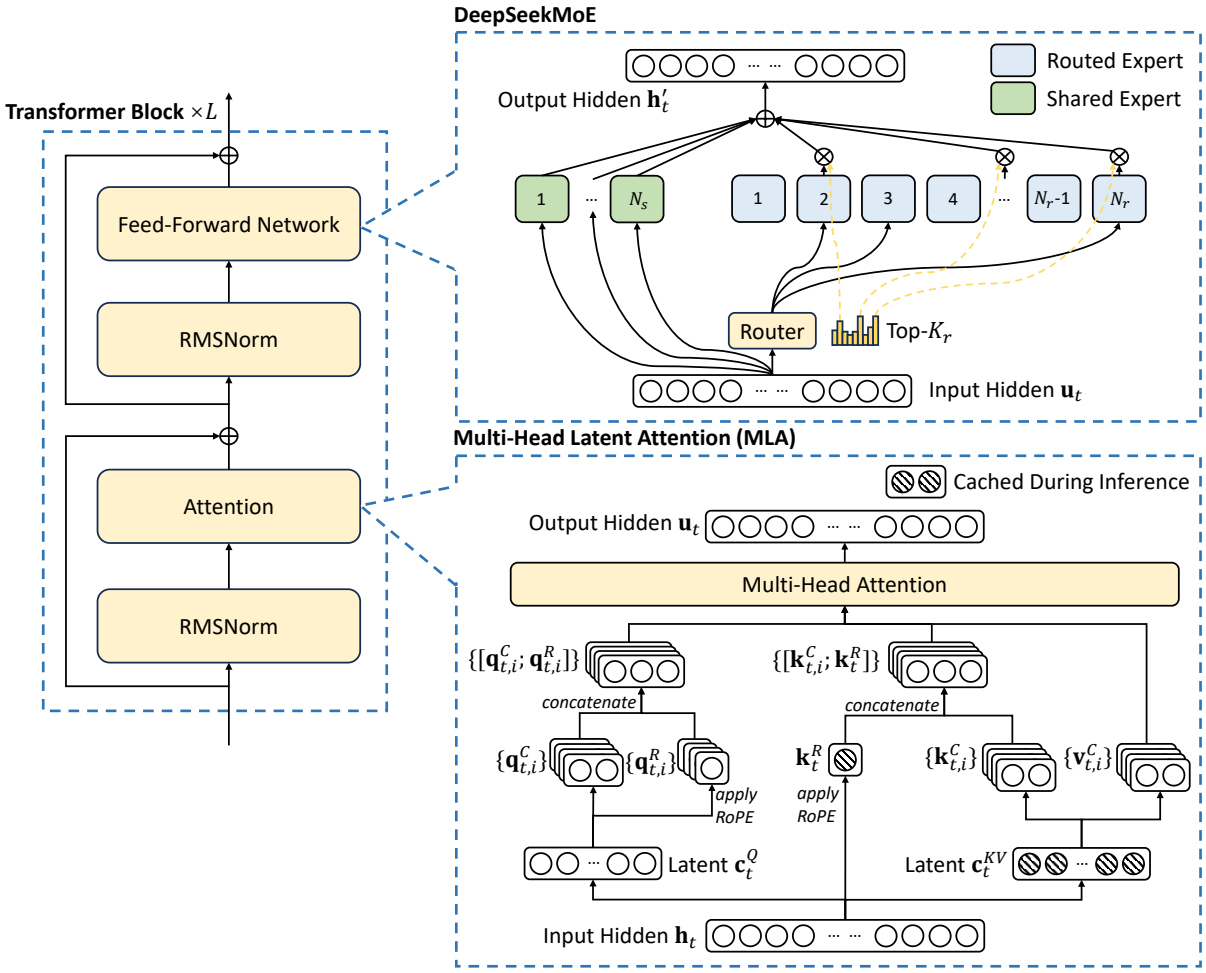


Figure 2 | Illustration of the basic architecture of DeepSeek-V3. Following DeepSeek-V2, we adopt MLA and DeepSeekMoE for efficient inference and economical training.

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:

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

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

$$\boxed{\mathbf{k}_t^R} = \mathrm{RoPE}(W^{KR}\mathbf{h}_t), \tag{3}$$

$$\mathbf{k}_{t,i} = [\mathbf{k}_{t,i}^C; \mathbf{k}_t^R], \tag{4}$$

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

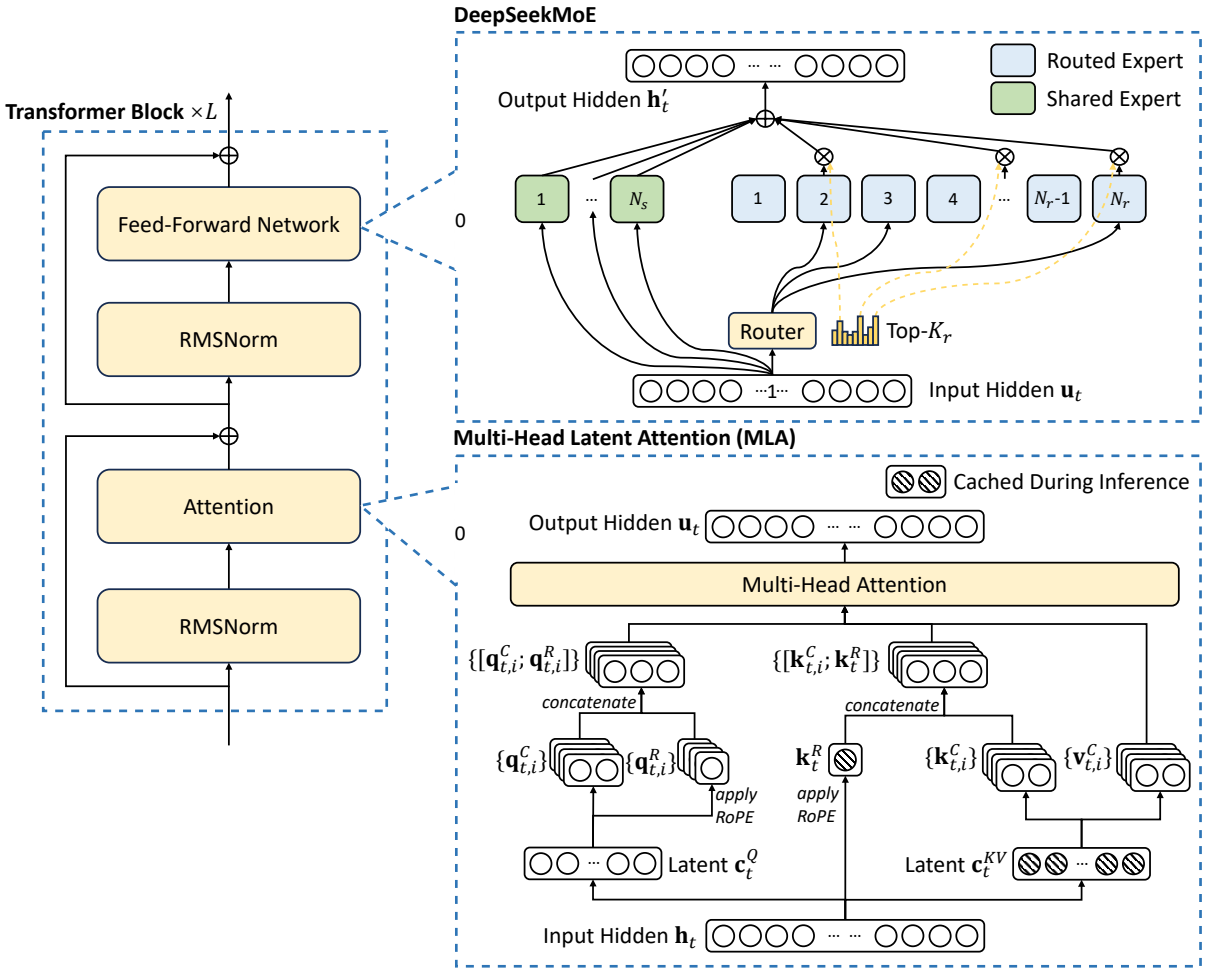


图2 | DeepSeek-V3基本架构的示意图。继DeepSeek-V2之后，我们采用MLA和DeepSeekMoE以实现高效推理和经济训练。

策略（Wang et al., 2024a）用于DeepSeekMoE，以减轻因确保负载平衡而导致的性能下降。图2展示了DeepSeek-V3的基本架构，我们将在本节中简要回顾MLA和DeepSeekMoE的细节。

2.1.1. Multi-Head Latent Attention

对于注意力，DeepSeek-V3 采用 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：

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

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

$$\boxed{\mathbf{k}_t^R} = \mathrm{RoPE}(W^{KR}\mathbf{h}_t), \tag{3}$$

$$\mathbf{k}_{t,i} = [\mathbf{k}_{t,i}^C; \mathbf{k}_t^R], \tag{4}$$

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

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); 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:

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

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

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

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

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 $\mathbf{u}_t$:

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

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

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:

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

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

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

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

其中 $\mathrm{c}_t^{KV} \in \mathbb{R}^{d_c}$ 是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} 和 $W^{UV} \in \mathbb{R}^{d_h n_h \times d_c}$ 分别是键和值的上投影矩阵；$W^{KR} \in \mathbb{R}^{d_h^R \times d}$ 是用于生成携带旋转位置嵌入（RoPE）的解耦键的矩阵（Su 等，2024）；RoPE(·) 表示应用 RoPE 矩阵的操作；[·；·] 表示连接。请注意，对于 MLA，仅在生成过程中需要缓存蓝框中的向量（即 c_t^{KV} 和 k_t^R），这导致 KV 缓存显著减少，同时保持与标准多头注意力（MHA）（Vaswani 等，2017）相当的性能。

对于注意力查询，我们还执行低秩压缩，这可以在训练期间减少激活内存：

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

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

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

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

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

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

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

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

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 with Auxiliary-Loss-Free Load Balancing

DeepSeekMoE的基本架构。对于前馈网络（FFNs），DeepSeek-V3采用了DeepSeekMoE架构（Dai等，2024）。与传统的MoE架构如GShard（Lepikhin等，2021）相比，DeepSeekMoE使用了更细粒度的专家，并将一些专家隔离为共享专家。设u_t表示第t个标记的FFN输入，我们计算FFN输出h_t'如下：

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

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

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

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

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；$\mathrm{FFN}_i^{(s)}(\cdot)$ 和 $\mathrm{FFN}_i^{(r)}(\cdot)$ 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数量；$g_{i,t}$ 是第 i 个专家的门控值；$s_{i,t}$ 是令牌与专家的亲和力；e_i 是第 i 个路由专家的中心向量；$\mathrm{Topk}(\cdot$ 和 $K)$ 表示包含第 K 高分的亲和力分数的集合，这些分数是为第 t 个令牌和所有路由专家计算的。与 DeepSeek-V2 略有不同，DeepSeek-V3 使用 sigmoid 函数来计算亲和力分数，并在所有选定的亲和力分数之间应用归一化，以生成门控值。

辅助损失自由的负载均衡。对于 MoE 模型，不平衡的专家负载将导致路由崩溃（Shazeer 等，2017）并降低在专家并行场景中的计算效率。传统解决方案通常依赖于辅助损失（Fedus 等，2021；Lepikhin 等，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，2024a）。为了在负载平衡和模型性能之间实现更好的权衡，我们开创了一种辅助损失自由的负载均衡策略（Wang 等，2024a）以确保负载平衡。具体来说，我们为每个专家引入一个偏置项 b_i 并将其添加到相应的亲和分数 $s_{i,t}$ 中，以确定前 K 个路由：

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

请注意，偏置项仅用于路由。门控值将与FFN输出相乘，仍然源自原始亲和力得分 $s_{i,t}$。在训练过程中，我们持续监控每个训练步骤整个批次的专家负载。在每个步骤结束时，如果其对应的专家过载，我们将减少偏置项 γ，如果其对应的专家负载不足，我们将增加偏置项 γ，其中 γ 是一个称为偏置更新速度的超参数。通过动态调整，DeepSeek-V3 在训练过程中保持专家负载平衡，并且比通过纯辅助损失鼓励负载平衡的模型实现更好的性能。

互补序列辅助损失。尽管DeepSeek-V3主要依赖于无辅助损失策略来实现负载平衡，为了防止任何单一序列内的极端不平衡，我们还采用了互补序列平衡损失：

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

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

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

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

其中平衡因子 α 是一个超参数，对于 DeepSeek-V3，将被赋予一个极小的值；$1(\cdot)$ 表示指示函数；T 表示序列中的标记数量。序列级平衡损失鼓励每个序列上的专家负载保持平衡。

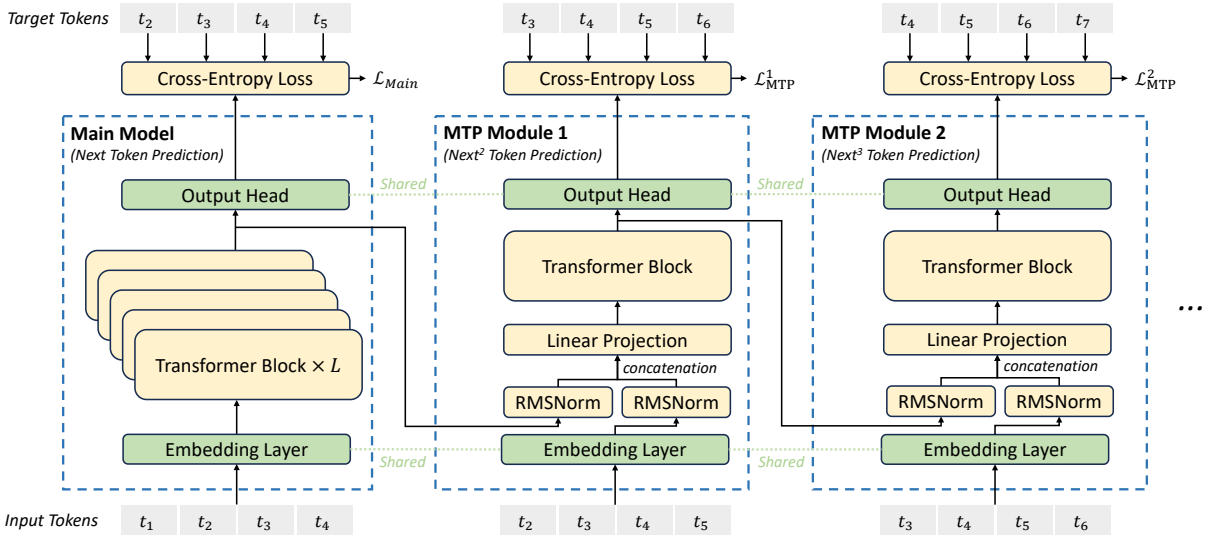


Figure 3 | Illustration of our Multi-Token Prediction (MTP) implementation. We keep the complete causal chain for the prediction of each token at each depth.

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 $\mathrm{Emb}(\cdot)$, a shared output head $\mathrm{OutHead}(\cdot)$, a Transformer block $\mathrm{TRM}_k(\cdot)$, 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 $Emb(t_{i+k}) \in \mathbb{R}^d$

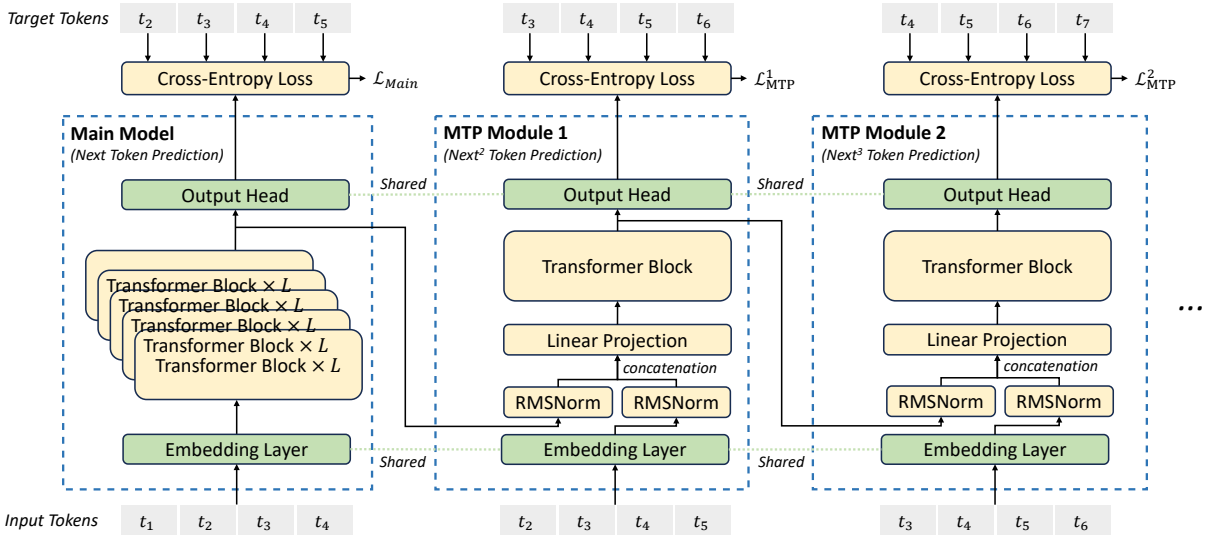


图3 | 我们的多标记预测（MTP）实现的示意图。我们保持每个深度上每个标记预测的完整因果链。

节点限制路由。与 DeepSeek-V2 使用的设备限制路由类似，DeepSeek-V3 也使用了一种受限路由机制，以限制训练期间的通信成本。简而言之，我们确保每个令牌最多会发送到 M 个节点，这些节点是根据分布在每个节点上的专家的最高 $\frac{K_r}{M}$ 亲和力得分之和进行选择的。在这一约束下，我们的 MoE 训练框架几乎可以实现完全的计算-通信重叠。

无令牌丢失。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平衡。因此，DeepSeek-V3 在训练期间不会丢弃任何令牌。此外，我们还实施了特定的部署策略，以确保推理负载平衡，因此 DeepSeek-V3 在推理期间也不会丢弃令牌。

2.2. 多标记预测

受到Gloeckle等人（2024）的启发，我们研究并为DeepSeek-V3设定了多标记预测（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目标密集化了训练信号，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）不同，他们使用独立的输出头并行预测D个额外标记，我们则顺序预测额外标记，并在每个预测深度保持完整的因果链。在本节中，我们介绍了MTP实现的细节。

MTP模块。具体来说，我们的MTP实现使用D个顺序模块来预测D个额外的标记。第k个MTP模块由一个共享的嵌入层Emb(·)、一个共享的输出头OutHead(·)、一个Transformer块$\mathrm{TRM}_k(\cdot)$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，我们首先结合第$i$个标记在$(k-1)$个深度$\mathrm{h}_i^{k-1} \in \mathbb{R}^d$的表示和第$(i+k)$个标记$Emb(t_{i+k}) \in \mathbb{R}^d$的嵌入。

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 $\text{OutHead}(\cdot)$ linearly maps the representation to logits and subsequently applies the $\text{Softmax}(\cdot)$ 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.

通过线性投影：

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

其中 $[\cdot;\cdot]$ 表示连接。特别地，当 k = 为 1 时，h_i^{k-1} 指的是主模型给出的表示。请注意，对于每个 MTP 模块，其嵌入层与主模型共享。组合后的 $\mathrm{h}_i'^k$ 作为第 k 层深度的 Transformer 块的输入，以生成当前深度的输出表示 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（包括左边界和右边界）。最后，以 h_i^k 作为输入，共享输出头将计算 k-th 额外预测标记 $P_{i+1+k}^k \in \mathbb{R}^V$ 的概率分布，其中 V 是词汇表大小：

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

输出头 OutHead(·) 线性映射表示到 logits，并随后应用 Softmax(·) 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 来改善训练。

MTP训练目标。对于每个预测深度，我们计算交叉熵损失 $\mathcal{L}_{\text{MTP}}^k$：

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

其中 T 表示输入序列长度，t_i 表示第 i 个位置的真实标记，$P_i^k[t_i]$ 表示由第 k 个 MTP 模块给出的 t_i 的相应预测概率。最后，我们计算所有深度的 MTP 损失的平均值，并乘以权重因子 λ，以获得整体 MTP 损失 $\mathcal{L}_{\text{MTP}}$，这作为 DeepSeek-V3 的额外训练目标：

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

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

3. 基础设施

3.1. 计算聚类

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

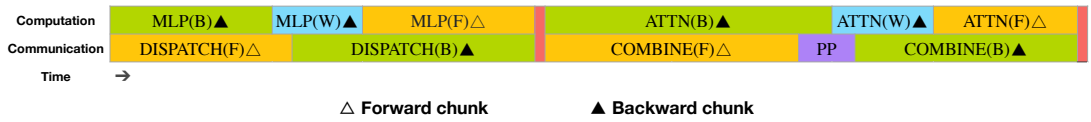


Figure 4 | Overlapping strategy for a pair of individual forward and backward chunks (the boundaries of the transformer blocks are not aligned). Orange denotes forward, green denotes "backward for input", blue denotes "backward for weights", purple denotes PP communication, and red denotes barriers.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

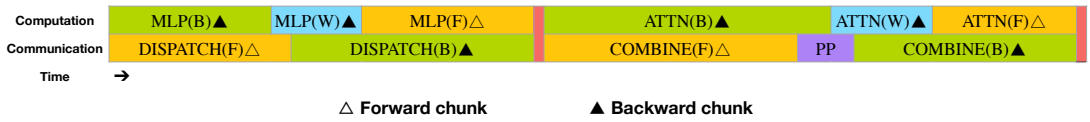


图4 | 一对个体前向和后向块的重叠策略（变换器块的边界未对齐）。橙色表示前向，绿色表示“输入的后向”，蓝色表示“权重的后向”，紫色表示PP通信，红色表示障碍。全到全和PP通信都可以完全隐藏。

3.2. 训练框架

DeepSeek-V3的训练得益于HAI-LLM框架，这是一个由我们的工程师从零开始打造的高效轻量级训练框架。总体而言，DeepSeek-V3应用了16路管道并行（PP）（Qi等，2023a）、64路专家并行（EP）（Lepikhin等，2021），跨越8个节点，以及ZeRO-1数据并行（DP）（Rajbhandari等，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DualPipe算法以实现高效的管道并行。与现有的PP方法相比，DualPipe具有更少的管道气泡。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高通信开销的挑战。其次，我们开发了高效的跨节点全到全通信内核，以充分利用IB和NVLink带宽，并节省专用于通信的流处理器（SM）。最后，我们在训练过程中精心优化内存占用，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 and Computation-Communication Overlap

对于DeepSeek-V3，由于跨节点专家并行性引入的通信开销，导致计算与通信的比率约为1:1。为了解决这个挑战，我们设计了一种创新的管道并行算法，称为DualPipe，它不仅通过有效重叠前向和后向计算-通信阶段来加速模型训练，还减少了管道气泡。

DualPipe的关键思想是在一对独立的前向和后向块中重叠计算和通信。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，对于一个后向块，`attention`和`MLP`进一步分为两部分，`backward for input`和`backward for weights`，如在ZeroBubble中所示（Qi et al., 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，并手动调整专用于通信与计算的GPU SM的比例。在这种重叠策略中，我们可以确保在执行期间，所有到所有的通信和PP通信都可以完全隐藏。鉴于高效的重叠策略，完整的DualPipe调度如图5所示。它采用双向管道调度，同时从管道的两端输入微批次，并且大量的通信可以完全重叠。这种重叠还确保了，随着模型的进一步扩展，只要我们保持恒定的计算与通信比例，我们仍然可以在节点之间使用细粒度的专家，同时实现近乎零的全到全通信开销。

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	$(PP-1)(F+B)$	$1\times$	PP
ZB1P	$(PP-1)(F+B-2W)$	$1\times$	PP
DualPipe (Ours)	$(\frac{PP}{2}-1)(F\&B+B-3W)$	$2\times$	$PP+1$

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

图5 | 示例双管道调度，适用于8个PP等级和20个微批次，分为两个方向。反向的微批次与正向的微批次是对称的，因此我们为了简化说明省略了它们的批次ID。两个被共享黑色边框包围的单元具有相互重叠的计算和通信。

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×	$PP+1$

表 2 | 不同管道并行方法的管道气泡和内存使用情况比较。F 表示前向块的执行时间，B 表示完整反向块的执行时间，W 表示“反向权重”块的执行时间，F 和 B 表示两个相互重叠的前向和反向块的执行时间。

此外，即使在没有重通信负担的更一般场景中，DualPipe 仍然表现出效率优势。在表 2 中，我们总结了不同 PP 方法的管道气泡和内存使用情况。如表中所示，与 ZB1P (Qi et al., 2023b) 和 1F1B (Harlap et al., 2018) 相比，DualPipe 显著减少了管道气泡，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了较大的 EP 大小，这并不会显著增加内存消耗。与 Chimera (Li 和 Hoefler, 2021) 相比，DualPipe 只要求管道阶段和微批次可被 2 整除，而不要求微批次可被管道阶段整除。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. Efficient Implementation of Cross-Node All-to-All Communication

为了确保DualPipe的计算性能足够，我们定制了高效的跨节点全到全通信内核（包括调度和合并），以节省用于通信的SM数量。这些内核的实现与MoE门控算法和我们集群的网络拓扑共同设计。具体来说，在我们的集群中，跨节点的GPU通过IB完全互联，而节点内通信则通过NVLink处理。NVLink提供160 GB/s的带宽，约为IB（50 GB/s）的3.2倍。为了有效利用IB和NVLink的不同带宽，我们限制每个令牌最多分发到4个节点，从而减少IB流量。对于每个令牌，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瞬时转发到承载其目标专家的特定GPU，而不被随后到达的令牌阻塞。通过这种方式，IB和NVLink之间的通信完全重叠，每个令牌可以高效地选择每个节点平均3.2个专家，而不会产生来自NVLink的额外开销。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

仅选择8名实际路由专家，它可以将此数字扩展到最多13名专家（4个节点 × 3.2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20个SM即可充分利用IB和NVLink的带宽。

详细来说，我们采用了扭曲专门化技术（Bauer et al., 2014），并将20个SM分成10个通信通道。在调度过程中，（1）IB发送，（2）IB到NVLink转发，以及（3）NVLink接收由各自的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。同样，在合并过程中，（1）NVLink发送，（2）NVLink到IB的转发和累积，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，因此我们还考虑它们对其他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，并自动调优通信块大小，这显著减少了对L2缓存的使用和对其他SM的干扰。

3.2.3. Extremely Memory Saving with Minimal Overhead

为了在训练过程中减少内存占用，我们采用以下技术。

重新计算 RMSNorm 和 MLA 上投影。在反向传播过程中，我们重新计算所有 RMSNorm 操作和 MLA 上投影，从而消除持续存储其输出激活的需要。通过少量的开销，这一策略显著减少了存储激活所需的内存。

CPU中的指数移动平均。在训练过程中，我们保留模型参数的指数移动平均（EMA），以便在学习率衰减后对模型性能进行早期评估。EMA参数存储在CPU内存中，并在每个训练步骤后异步更新。这种方法使我们能够在不增加额外内存或时间开销的情况下维护EMA参数。

共享嵌入和输出头用于多标记预测。通过双管道策略，我们将模型的最浅层（包括嵌入层）和最深层（包括输出头）部署在同一PP等级上。这种安排使得MTP模块和主模型之间的共享嵌入和输出头的参数和梯度能够物理共享。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

受到近期低精度训练的进展启发（Dettmers et al., 2022; Noune et al., 2022; Peng et al., 2023b），我们提出了一种细粒度混合精度框架，利用 FP8 数据格式来训练 DeepSeek-V3。尽管低精度训练前景广阔，但通常受到激活、权重和梯度中异常值的限制（Fishman et al., 2024; He et al.; Sun et al., 2024）。尽管在推理量化方面取得了显著进展（Frantar et al., 2022; Xiao et al., 2023），但成功应用低精度技术于大规模语言模型的研究相对较少。

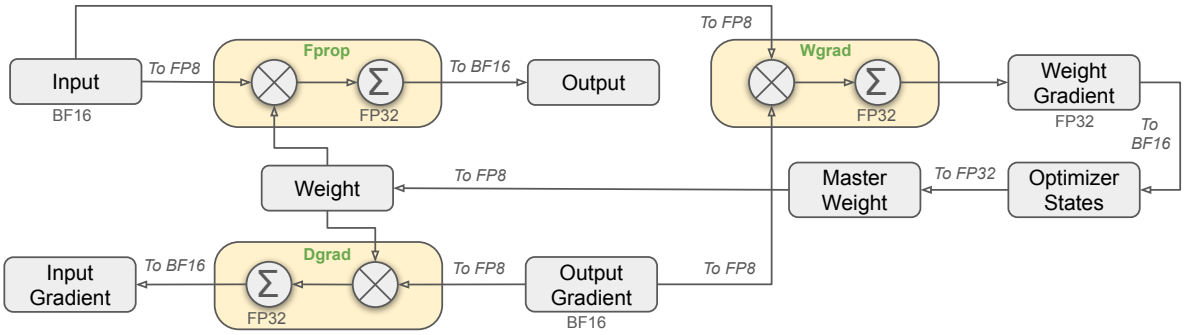


Figure 6 | The overall mixed precision framework with FP8 data format. For clarification, only the `Linear` operator is illustrated.

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

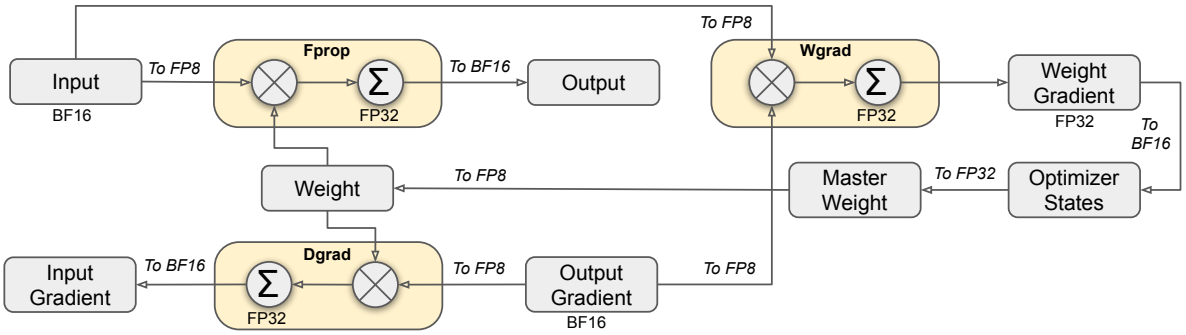


图6 | 使用FP8数据格式的整体混合精度框架。为清晰起见，仅展示了Linear运算符。

预训练（Fishman 等，2024）。为了解决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 $1 \times N_c$ 元素进行块状分组或以 $N_c \times N_c$ 元素进行块状分组。在我们的高精度累积过程中，相关的去量化开销在很大程度上得到了缓解，这是实现准确的 FP8 一般矩阵乘法（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，同时在 BF16 中存储低精度优化器状态。我们在与 DeepSeek-V2-Lite 和 DeepSeek-V2 类似的两个模型规模上验证了所提出的 FP8 混合精度框架，训练大约 1 万亿个标记（更多细节见附录 B.1）。值得注意的是，与 BF16 基线相比，我们的 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可接受范围内。

3.3.1. Mixed Precision Framework

基于广泛采用的低精度训练技术（Kalamkar et al., 2019; Narang et al., 2017），我们提出了一种用于FP8训练的混合精度框架。在这个框架中，大多数计算密集型操作在FP8中进行，而少数关键操作则战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，采用FP8精度实现。这些GEMM操作接受FP8张量作为输入，并以BF16或FP32格式生成输出。如图6所示，与Linear操作相关的所有三个GEMM，即Fprop (前向传播)、Dgrad (激活反向传播)和Wgrad (权重反向传播)，均在FP8中执行。与原始的BF16方法相比，这种设计理论上将计算速度提高了一倍。此外，FP8 Wgrad GEMM允许将激活存储在FP8中，以便在反向传播中使用。这显著减少了内存消耗。

尽管FP8格式具有效率优势，但某些运算符仍然由于对低精度计算的敏感性而需要更高的精度。此外，一些低成本运算符也可以在对整体训练成本几乎没有影响的情况下利用更高的精度。因此，在经过仔细调查后，我们为以下组件保持原始精度（例如，BF16或FP32）：嵌入模块、输出头、MoE门控模块、归一化运算符和注意力运算符。这些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，我们以更高的精度存储主权重、权重梯度和优化器状态。

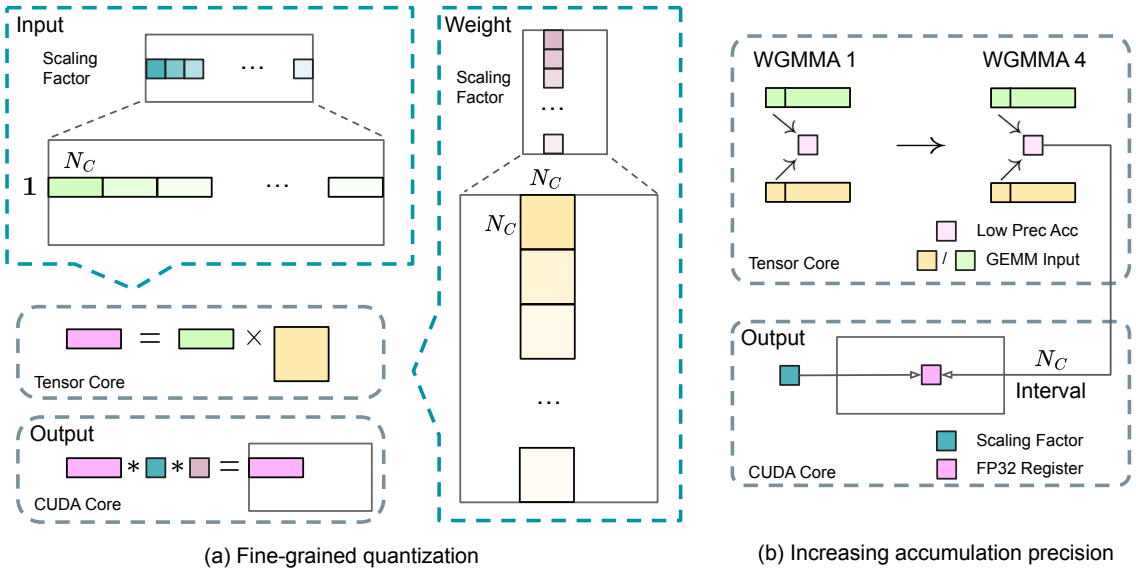


Figure 7 | (a) We propose a fine-grained quantization method to mitigate quantization errors caused by feature outliers; for illustration simplicity, only `Fprop` is illustrated. (b) In conjunction with our quantization strategy, we improve the FP8 GEMM precision by promoting to CUDA Cores at an interval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

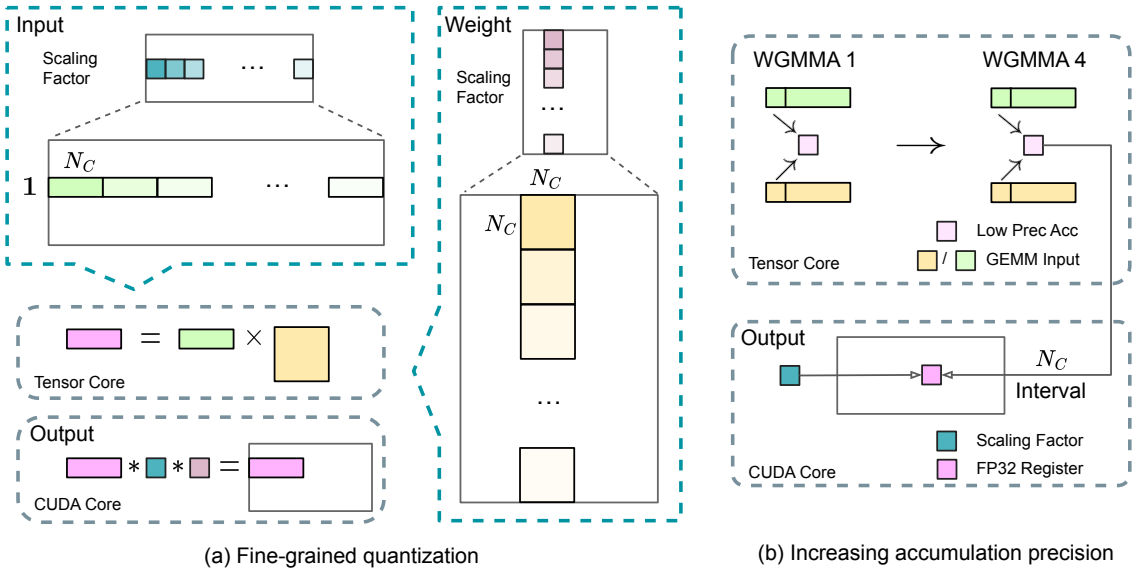
these high-precision components incur some memory overheads, their impact can be minimized through efficient sharding across multiple DP ranks in our distributed training system.

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) Increasing accumulation precision

图7 | (a) 我们提出了一种细粒度量化方法，以减轻特征异常值引起的量化误差；为简化说明，仅展示 Fprop。(b) 结合我们的量化策略，我们通过在 N_C = 128 个元素的 MMA 间隔内提升到 CUDA 核心，来提高 FP8 GEMM 的精度，以实现高精度累积。

这些高精度组件会产生一些内存开销，但通过在我们的分布式训练系统中对多个DP等级进行有效分片，可以最小化它们的影响。

3.3.2. Improved Precision from Quantization and Multiplication

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准确性，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，溢出和下溢是常见的挑战，这受到其减少的指数位数的限制。作为标准做法，输入分布通过将输入张量的最大绝对值缩放到FP8的最大可表示值来与FP8格式的可表示范围对齐（Narang等，2017）。这种方法使得低精度训练对激活异常值高度敏感，这可能严重降低量化精度。为了解决这个问题，我们提出了一种细粒度量化方法，在更细的层面上应用缩放。如图7（a）所示，（1）对于激活，我们在1x128瓦片基础上对元素进行分组和缩放（即每个token每128个通道）；（2）对于权重，我们在128x128块基础上对元素进行分组和缩放（即每128个输入通道每128个输出通道）。这种方法确保量化过程能够更好地适应异常值，通过根据较小的元素组调整缩放。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基础上对激活进行分组和缩放时的训练不稳定性。

我们方法中的一个关键修改是引入了沿GEMM操作内维度的每组缩放因子。这个功能在标准的FP8 GEMM中并不直接支持。然而，结合我们精确的FP32累积策略，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

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的理念高度一致（Rouhani et al., 2023b），而NVIDIA下一代GPU（Blackwell系列）的Tensor Cores已宣布支持具有更小量化粒度的微缩格式（NVIDIA, 2024a）。我们希望我们的设计能够为未来的工作提供参考，以跟上最新的GPU架构。

提高累积精度。低精度GEMM操作常常遭受下溢问题，其准确性在很大程度上依赖于高精度累积，这通常在FP32精度下进行（Kalamkar等，2019；Narang等，2017）。然而，我们观察到在NVIDIA H800 GPU上，FP8 GEMM的累积精度仅限于保留约14位，这显著低于FP32的累积精度。当内维度K较大时（Wortsman等，2023），这一问题将更加明显，这是在大规模模型训练中，批量大小和模型宽度增加的典型场景。以两个随机矩阵的GEMM操作为例，K =为4096，在我们的初步测试中，Tensor Cores中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，有限的累积精度仍然是一些FP8框架（NVIDIA，2024b）中的默认选项，严重限制了训练准确性。

为了应对这个问题，我们采用将计算推广到CUDA核心以提高精度的策略（Thakkar等，2023）。该过程在图7（b）中进行了说明。具体来说，在Tensor核心上执行MMA（矩阵乘法-累加）时，使用有限的位宽累积中间结果。一旦达到N_C的区间，这些部分结果将被复制到CUDA核心上的FP32寄存器，在那里进行全精度FP32累积。如前所述，我们的细粒度量化在内维度K上应用每组缩放因子。这些缩放因子可以在CUDA核心上高效地进行乘法运算，作为去量化过程，几乎没有额外的计算成本。

值得注意的是，这一修改降低了单个warpgroup的WGMMA（Warpgroup级矩阵乘加）指令发射率。然而，在H800架构上，两个WGMMA通常会同时存在：当一个warpgroup执行提升操作时，另一个可以执行MMA操作。这一设计使得两种操作能够重叠，从而保持Tensor Cores的高利用率。根据我们的实验，将N_C =设置为128个元素，相当于4个WGMMAs，代表了可以显著提高精度而不引入大量开销的最小累积间隔。

尾数与指数。与之前的工作（NVIDIA, 2024b; Peng et al., 2023b; Sun et al., 2019b）采用的混合 FP8 格式相比，该格式使用 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，我们在所有张量上采用 E4M3 格式以获得更高的精度。我们将这种方法的可行性归因于我们细粒度的量化策略，即瓷砖和块级缩放。通过对较小的元素组进行操作，我们的方法有效地在这些分组元素之间共享指数位，从而减轻了有限动态范围的影响。

在线量化。延迟量化在张量级量化框架中被采用（NVIDIA, 2024b; Peng et al., 2023b），它保持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.

通过先前迭代中的值来推断当前值。为了确保准确的比例并简化框架，我们在线计算每个 1x128 激活块或 128x128 权重块的最大绝对值。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. Low-Precision Storage and Communication

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，进一步减少了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时刻，而不会导致可观察的性能下降。然而，主权重（由优化器存储）和梯度（用于批量大小累积）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格式以用于Linear操作符的反向传播是一个自然的选择。然而，对于低成本高精度训练，几个操作符需要特别考虑:

(1) 注意力操作符后的 **Linear** 输入。这些激活值也用于注意力操作符的反向传播，这使得它对精度敏感。我们为这些激活值采用了专门定制的 E5M6 数据格式。此外，这些激活值将在反向传播中从 1x128 量化块转换为 128x1 块。为了避免引入额外的量化误差，所有的缩放因子都是四舍五入缩放的，即为 2 的整数次幂。

(2) MoE 中 SwiGLU 操作符的输入。为了进一步降低内存成本，我们缓存 SwiGLU 操作符的输入，并在反向传播中重新计算其输出。这些激活值也使用我们的细粒度量化方法以 FP8 存储，达到内存效率和计算准确性之间的平衡。

低精度通信。通信带宽是MoE模型训练中的一个关键瓶颈。为了缓解这一挑战，我们在MoE上投影之前将激活量化为FP8，然后应用与FP8 Fprop兼容的dispatch组件。在注意力操作符后的Linear输入中，这个激活的缩放因子是2的整数次幂。对MoE下投影之前的激活梯度也采用类似策略。对于前向和后向combine组件，我们将其保留为BF16，以保持训练管道关键部分的训练精度。

3.4. 推理与部署

我们在 H800 集群上部署 DeepSeek-V3，其中每个节点内的 GPU 通过 NVLink 互连，集群内的所有 GPU 通过 IB 完全互连。为了同时确保在线服务的服务级目标 (SLO) 和高吞吐量，我们采用以下部署策略，将 *prefilling* 和 *decoding* 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. Prefilling

预填充阶段的最小部署单元由4个节点和32个GPU组成。attention部分采用4路张量并行（TP4）与序列并行（SP）相结合，并结合8路数据并行（DP8）。其小的TP大小为4，限制了TP通信的开销。对于MoE部分，我们使用32路专家并行（EP32），确保每个专家处理足够大的批量，从而提高计算效率。对于MoE的全到全通信，我们使用与训练相同的方法：首先通过IB在节点之间传输令牌，然后通过NVLink在节点内的GPU之间转发。特别地，我们对浅层中的密集MLP使用1路张量并行，以节省TP通信。

为了在 MoE 部分实现不同专家之间的负载均衡，我们需要确保每个 GPU 处理大约相同数量的标记。为此，我们引入了一种 *redundant experts* 的部署策略，该策略复制高负载专家并进行冗余部署。高负载专家是根据在线部署期间收集的统计数据进行检测的，并定期进行调整（例如，每 10 分钟）。在确定冗余专家的集合后，我们根据观察到的负载仔细重新安排节点内 GPU 之间的专家，努力在不增加跨节点全到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，我们为预填充阶段设置了 32 个冗余专家。对于每个 GPU，除了它所托管的原始 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，为了提高吞吐量并隐藏全到全和TP通信的开销，我们同时处理两个具有相似计算负载的微批次，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，我们正在探索一种针对专家的 *dynamic redundancy* 策略，其中每个 GPU 托管更多的专家（例如，16 个专家），但在每个推理步骤中仅激活 9 个。在每层的全到全操作开始之前，我们会即时计算全局最优路由方案。考虑到预填充阶段涉及的巨大计算，计算该路由方案的开销几乎可以忽略不计。

3.4.2. Decoding

在解码过程中，我们将共享专家视为路由专家。从这个角度来看，每个令牌在路由时将选择9个专家，其中共享专家被视为一个重负载专家，始终会被选择。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相结合，结合DP80，而MoE部分使用EP320。对于MoE部分，每个GPU仅托管一个专家，64个GPU负责托管冗余专家和共享专家。dispatch和combine部分的全到全通信通过直接点对点传输在IB上进行，以实现低延迟。此外，我们利用IBGDA（NVIDIA，2022）技术进一步减少延迟并提高通信效率。

类似于预填充，我们定期根据我们在线服务的统计专家负载，在某个时间间隔内确定冗余专家的集合。然而，我们不需要重新安排专家，因为每个GPU只托管一个专家。我们还在探索用于解码的 *dynamic redundancy* 策略。然而，这需要对计算全局最优路由方案的算法进行更仔细的优化，并与 dispatch 内核进行融合以减少开销。

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 (General Matrix Multiply) employs fixed-point accumulation, aligning the mantissa products by right-shifting based on the maximum exponent before addition. Our experiments reveal that it only uses the highest 14

此外，为了提高吞吐量并隐藏全到全通信的开销，我们还在解码阶段探索同时处理两个具有相似计算负载的微批次。与预填充不同，`attention` 在解码阶段消耗了更大一部分时间。因此，我们将一个微批次的 `attention` 与另一个微批次的 `dispatch+MoE+combine` 重叠。在解码阶段，每个专家的批量大小相对较小（通常在 256 个标记以内），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，因此使用更少的 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，我们可以仅分配少量的 SM 给 `dispatch+MoE+combine`。

3.5. 硬件设计建议

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

3.5.1. Communication Hardware

在DeepSeek-V3中，我们实现了计算与通信之间的重叠，以隐藏计算过程中的通信延迟。这显著减少了与串行计算和通信相比对通信带宽的依赖。然而，目前的通信实现依赖于昂贵的SM（例如，我们为此目的在H800 GPU中分配了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，因为张量核心完全未被充分利用。

目前，SMs 主要执行以下任务以实现全到全的通信:

- 在同一节点内，从单个GPU聚合目标为多个GPU的IB（InfiniBand）流量，同时在IB（InfiniBand）和NVLink域之间转发数据。
- 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` 操作以 `all-to-all combine`。
- 在通过IB和NVLink域将分块数据传输到多个专家时，管理细粒度内存布局。

我们期望未来的供应商开发硬件，将这些通信任务从宝贵的计算单元 SM 中卸载，作为 GPU 协处理器或类似 NVIDIA SHARP 的网络协处理器（Graham 等，2016）。此外，为了减少应用程序编程的复杂性，我们希望这些硬件能够从计算单元的角度统一 IB（横向扩展）和 NVLink（纵向扩展）网络。通过这个统一的接口，计算单元可以轻松地通过基于简单原语提交通信请求，在整个 IB-NVLink-统一域中完成 `read`、`write`、`multicast` 和 `reduce` 等操作。

3.5.2. Compute Hardware

在当前NVIDIA Hopper架构的Tensor Core实现中，FP8 GEMM（通用矩阵乘法）采用定点累加，通过根据最大指数右移来对齐尾数乘积，然后进行加法。我们的实验表明，它仅使用最高的14

bits of each mantissa product after sign-fill right shifting, and truncates bits exceeding this range. However, for example, to achieve precise FP32 results from the accumulation of 32 FP8×FP8 multiplications, at least 34-bit precision is required. Thus, we recommend that future chip designs increase accumulation precision in Tensor Cores to support full-precision accumulation, or select an appropriate accumulation bit-width according to the accuracy requirements of training and inference algorithms. This approach ensures that errors remain within acceptable bounds while maintaining computational efficiency.

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.

每个尾数乘积在符号填充右移后的位数，并截断超出此范围的位数。然而，例如，为了从32个FP8×FP8乘法的累积中获得精确的FP32结果，至少需要34位精度。因此，我们建议未来的芯片设计在张量核心中增加累积精度，以支持全精度累积，或根据训练和推理算法的准确性要求选择适当的累积位宽。这种方法确保错误保持在可接受的范围内，同时保持计算效率。

对瓷砖和块状量化的支持。目前的GPU仅支持每个张量的量化，缺乏对我们瓷砖和块状量化等细粒度量化的原生支持。在当前的实现中，当达到N_C区间时，部分结果将从张量核心复制到CUDA核心，乘以缩放因子，并添加到CUDA核心的FP32寄存器中。尽管结合我们精确的FP32累积策略，去量化的开销显著减轻，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接收缩放因子并实现带组缩放的MMA来支持细粒度量化。通过这种方式，整个部分和累积和去量化可以直接在张量核心内部完成，直到产生最终结果，从而避免频繁的数据移动。

支持在线量化。目前的实现难以有效支持在线量化，尽管我们的研究已证明其有效性。在现有过程中，我们需要从 HBM（高带宽内存）中读取 128 个 BF16 激活值（前一次计算的输出）进行量化，然后将量化后的 FP8 值写回 HBM，再次读取以进行 MMA。为了解决这一低效问题，我们建议未来的芯片将 FP8 转换和 TMA（张量内存加速器）访问集成到一个单一的融合操作中，以便在将激活从全局内存转移到共享内存时完成量化，避免频繁的内存读写。我们还建议支持一个 warp 级别的转换指令以加速，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的方法，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，从而将离芯片内存访问减少约 50%。

对转置GEMM操作的支持。目前的架构使得将矩阵转置与GEMM操作结合变得繁琐。在我们的工作流程中，前向传播期间的激活被量化为`1x128` FP8块并存储。在反向传播期间，需要读取矩阵，进行去量化、转置、重新量化为`128x1`块，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操作之前启用从共享内存直接读取矩阵的转置，以满足训练和推理中所需的精度。结合FP8格式转换和TMA访问的融合，这一增强将显著简化量化工作流程。

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 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^R_h 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

4. 预训练

4.1. 数据构建

与DeepSeek-V2相比，我们通过增强数学和编程样本的比例来优化预训练语料库，同时将多语言覆盖范围扩展到英语和中文以外。此外，我们的数据处理流程经过精细化，以最小化冗余，同时保持语料库的多样性。受到Ding等人（2024）的启发，我们实施了文档打包方法以确保数据完整性，但在训练过程中不采用跨样本注意力掩蔽。最后，DeepSeek-V3的训练语料库由我们分词器中的14.8T高质量和多样化的标记组成。

在DeepSeekCoder-V2（DeepSeek-AI，2024a）的训练过程中，我们观察到填充中间（FIM）策略并未妨碍下一个标记的预测能力，同时使模型能够根据上下文线索准确预测中间文本。与DeepSeekCoder-V2一致，我们在DeepSeek-V3的预训练中也采用了FIM策略。具体而言，我们使用前缀-后缀-中间（PSM）框架来构建数据，如下所示：

$$\texttt{<|fim\_begin|>}f_{\text{前}}\texttt{<|fim\_hole|>}f_{\text{后}}\texttt{<|fim\_end|>}f_{\text{中间}}\texttt{<|eos\_token|>}。$$

该结构在文档级别应用，作为预打包过程的一部分。FIM策略的应用比例为0.1，与PSM框架一致。

DeepSeek-V3 的分词器采用字节级 BPE（Shibata 等，1999），具有 128K 令牌的扩展词汇表。我们的分词器的预分词器和训练数据经过修改，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符号和换行符的令牌。然而，当模型处理没有终端换行符的多行提示时，这种技巧可能会引入令牌边界偏差（Lundberg，2023），特别是在少量示例评估提示中。为了解决这个问题，我们在训练期间随机拆分一定比例的此类组合令牌，这使模型接触到更广泛的特殊情况，并减轻了这种偏差。

4.2. 超参数

模型超参数。我们将Transformer层的数量设置为61，隐藏维度设置为7168。所有可学习参数均以标准差0.006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'_c设置为1536。对于解耦的查询和键，我们将每个头的维度d_h^R设置为64。我们用MoE层替换除了前三层之外的所有FFN。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token将激活8个专家，并确保每个token最多发送到4个节点。多token预测深度D设置为1，即除了确切的下一个token，每个token将预测一个额外的token。与DeepSeek-V2一样，DeepSeek-V3在压缩的潜在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总共有671B参数，其中每个token激活37B。

训练超参数。我们使用AdamW优化器（Loshchilov和Hutter，2017），超参数设置为$\beta_1 = 0.9$，$\beta_2 = 0.95$，以及权重衰减= 0.1。在预训练期间，我们将最大序列长度设置为4K，并在14.8T个标记上预训练DeepSeek-V3。至于

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 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.

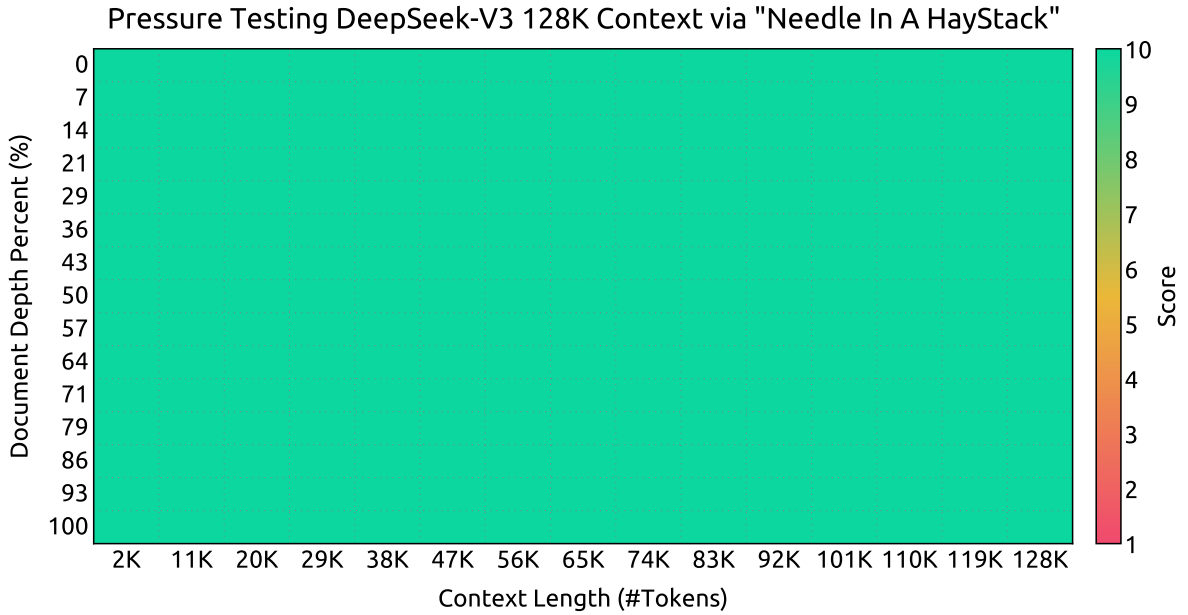


Figure 8 | Evaluation results on the "Needle In A Haystack" (NIAH) tests. DeepSeek-V3 performs well across all context window lengths up to 128K.

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.

学习率调度，我们首先在前2K步内将其线性增加从0到2.2×10^{-4}。然后，我们保持一个恒定的学习率为2.2×10^{-4}，直到模型消耗10T训练标记。随后，我们在4.3T标记中逐渐将学习率衰减到2.2×10^{-5}，遵循余弦衰减曲线。在最后500B标记的训练中，我们在前333B标记中保持恒定的学习率为2.2×10^{-5}，并在剩余的167B标记中切换到另一个恒定学习率为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，在前469B标记的训练中，批量大小逐渐从3072增加到15360，然后在剩余的训练中保持15360。我们利用管道并行性将模型的不同层部署在不同的GPU上，对于每一层，路由的专家将均匀部署在属于8个节点的64个GPU上。至于节点限制路由，每个标记最多将发送到4个节点（即，$M = 4$）。对于无辅助损失的负载平衡，我们将偏置更新速度γ设置为0.001，前14.3T标记为0.0，剩余500B标记。对于平衡损失，我们将α设置为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标记中设置为0.3，在剩余4.8T标记中设置为0.1。

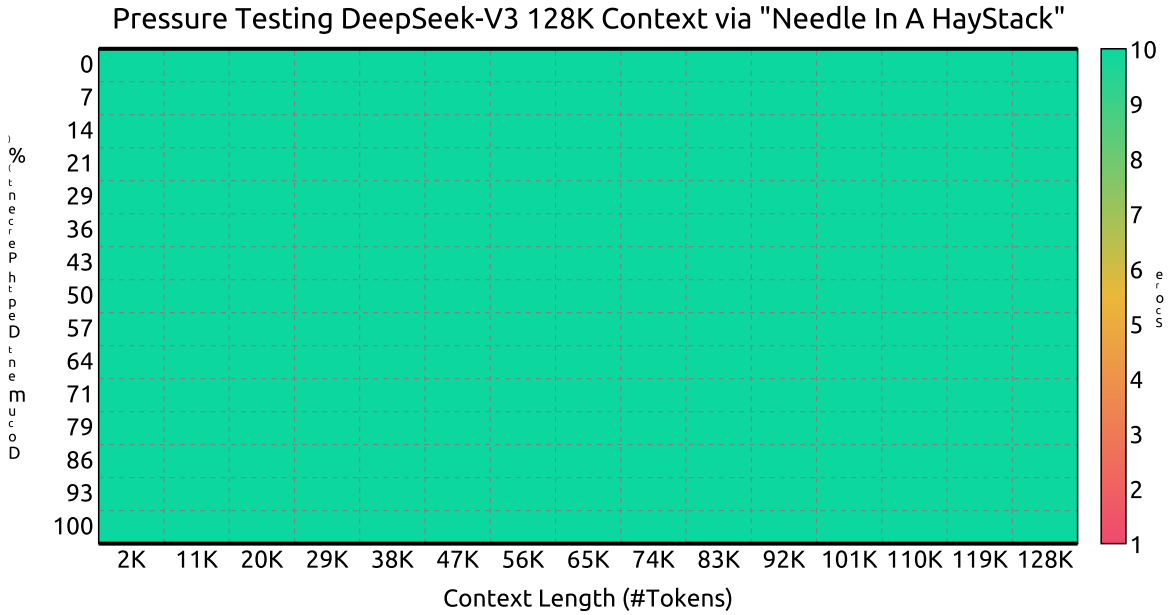


图8 | 在“干草堆中的针”（NIAH）测试上的评估结果。DeepSeek-V3 在所有上下文窗口长度（最高可达128K）上表现良好。

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。在预训练阶段之后，我们应用 YaRN (Peng et al., 2023a) 进行上下文扩展，并进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，然后到 128K。YaRN 配置与 DeepSeek-V2 中使用的配置一致，仅应用于解耦的共享密钥 k_t^R。两个阶段的超参数保持一致，规模 $s = 40$，$\alpha = 1$，$\beta = 32$，缩放因子 $\sqrt{t} = 0.1 \ln s + 1$。在第一阶段，序列长度设置为 32K，批量大小为 1920。在第二阶段，序列长度增加到 128K，批量大小减少到 480。两个阶段的学习率设置为 7.3×10^{-6}，与预训练阶段的最终学习率相匹配。

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.

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.

通过这两阶段的扩展训练，DeepSeek-V3能够处理长度高达128K的输入，同时保持强大的性能。图8显示，经过监督微调的DeepSeek-V3在“干草堆中的针”（NIAH）测试中取得了显著的表现，展示了在上下文窗口长度高达128K时的一致鲁棒性。

4.4. 评估

4.4.1. Evaluation Benchmarks

DeepSeek-V3的基础模型是在一个多语言语料库上进行预训练的，其中英语和中文占据了大多数，因此我们在一系列主要以英语和中文为主的基准上评估其性能，以及在一个多语言基准上。我们的评估基于我们在HAI-LLM框架中集成的内部评估框架。考虑的基准被分类并列出如下，其中下划线基准为中文，双下划线基准为多语言基准：

多学科多项选择数据集包括 MMLU（Hendrycks 等，2020），MMLU-Redux（Gema 等，2024），MMLU-Pro（Wang 等，2024b），MMMLU（OpenAI，2024b），C-Eval（Huang 等，2023）和 CMMLU（Li 等，2023）。

语言理解和推理数据集包括 HellaSwag (Zellers et al., 2019)、PIQA (Bisk et al., 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等，2017）和NaturalQuestions（Kwiatkowski等，2019）。

阅读理解数据集包括 RACE Lai 等（2017）、DROP（Dua 等，2019）、C3（Sun 等，2019a）和 CMRC（Cui 等，2019）。

参考消歧义数据集包括 CLUEWSC（Xu 等，2020）和 WinoGrande（Sakaguchi 等，2019）。

语言建模数据集包括 Pile（Gao 等，2020）。

中文理解和文化数据集包括CCPM（Li等，2021）。

数学数据集包括 GSM8K（Cobbe 等，2021）、MATH（Hendrycks 等，2021）、MGSM（Shi 等，2023）和 CMath（Wei 等，2023）。

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

标准化考试包括 AGIEval（Zhong 等，2023）。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 在内的数据集采用基于困惑度的评估，并对 TriviaQA、NaturalQuestions、DROP、MATH、GSM8K、MGSM、HumanEval、MBPP、LiveCodeBench-Base、CRUXEval、BBH、AGIEval、CLUEWSC、CMRC 和 CMath 采用基于生成的评估。此外，我们对 Pile-test 进行基于语言建模的评估，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的公平比较。

	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.

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 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.

	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**

表3 | DeepSeek-V3-Base与其他代表性开源基础模型的比较。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。得分差距不超过0.3的被视为处于同一水平。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。

4.4.2. Evaluation Results

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对所有这些模型进行评估，并确保它们共享相同的评估设置。请注意，由于我们在过去几个月中对评估框架的更改，DeepSeek-V2-Base的性能与我们之前报告的结果略有不同。总体而言，DeepSeek-V3-Base在各项基准测试中全面超越了DeepSeek-V2-Base和Qwen2.5 72B Base，并在大多数基准测试中超过了LLaMA-3.1 405B Base，基本上成为最强的开源模型。

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, 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.

从更详细的角度来看，我们分别将 DeepSeek-V3-Base 与其他开源基础模型进行比较。 (1) 与 DeepSeek-V2-Base 相比，由于我们模型架构的改进、模型规模和训练标记的扩大以及数据质量的提升，DeepSeek-V3-Base 的性能显著优于预期。 (2) 与 Qwen2.5 72B Base 这一最先进的中文开源模型相比，DeepSeek-V3-Base 在激活参数仅为其一半的情况下，仍展现出显著的优势，特别是在英语、多语言、代码和数学基准测试上。至于中文基准测试，除了 CMMLU（一个中文多学科选择题任务）外，DeepSeek-V3-Base 的表现也优于 Qwen2.5 72B。 (3) 与 LLaMA-3.1 405 B Base 这一激活参数多达 11 倍的最大开源模型相比，DeepSeek-V3-Base 在多语言、代码和数学基准测试上也表现得更好。至于英语和中文语言基准测试，DeepSeek-V3-Base 显示出具有竞争力或更好的表现，特别是在 BBH、MMLU 系列、DROP、C-Eval、CMMLU 和 CCPM 上表现尤为出色。

由于我们高效的架构和全面的工程优化，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，训练 DeepSeek-V3 每万亿个标记仅需 180K H800 GPU 小时，这比训练 72 B 或 405B 稠密模型便宜得多。

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**

表4 | MTP策略的消融结果。MTP策略在大多数评估基准上始终提升模型性能。

4.5. 讨论

4.5.1. Ablation Studies for Multi-Token Prediction

在表4中，我们展示了MTP策略的消融结果。具体来说，我们在两个基线模型上验证了MTP策略，涵盖不同的规模。在小规模下，我们在1.33T个标记上训练了一个包含15.7B总参数的基线MoE模型。在大规模下，我们在540B个标记上训练了一个包含228.7B总参数的基线MoE模型。在此基础上，保持训练数据和其他架构不变，我们在它们上附加了一个1深度的MTP模块，并训练了两个使用MTP策略的模型进行比较。请注意，在推理过程中，我们直接丢弃MTP模块，因此被比较模型的推理成本完全相同。从表中可以观察到，MTP策略在大多数评估基准上始终增强了模型性能。

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.

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. 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

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**

表5 | 无辅助损失平衡策略的消融结果。与纯粹基于辅助损失的方法相比，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。

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

在表5中，我们展示了无辅助损失平衡策略的消融结果。我们在两个基线模型上验证了这一策略，涵盖不同的规模。在小规模下，我们训练了一个基线MoE模型，包含15.7B总参数，使用1.33T标记。在大规模下，我们训练了一个基线MoE模型，包含228.7B总参数，使用578B标记。这两个基线模型纯粹使用辅助损失来促进负载平衡，并使用带有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与DeepSeek-V2-Lite和DeepSeek-V2相同。在这两个基线模型的基础上，保持训练数据和其他架构不变，我们去除了所有辅助损失，并引入了无辅助损失平衡策略进行比较。从表中可以观察到，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. Batch-Wise Load Balance VS. Sequence-Wise Load Balance

辅助损失自由平衡与序列级辅助损失之间的关键区别在于它们的平衡范围：批次级与序列级。与序列级辅助损失相比，批次级平衡施加了更灵活的约束，因为它不强制每个序列在领域内保持平衡。这种灵活性使得专家能够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测试集上，基于16B辅助损失的基线模型和16B辅助损失自由模型在不同领域的专家负载。如图9所示，我们观察到辅助损失自由模型表现出更大的专家专业化模式，正如预期的那样。

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次辅助损失，该损失鼓励每个训练批次的负载平衡，而不是每个序列的负载平衡。实验结果表明，当实现类似水平的批次负载平衡时，批次辅助损失也可以实现与无辅助损失方法相似的模型性能。具体来说，在我们对1B MoE模型的实验中，验证损失为：2.258（使用序列辅助损失），2.253（使用无辅助损失方法），以及2.253（使用批次辅助损失）。

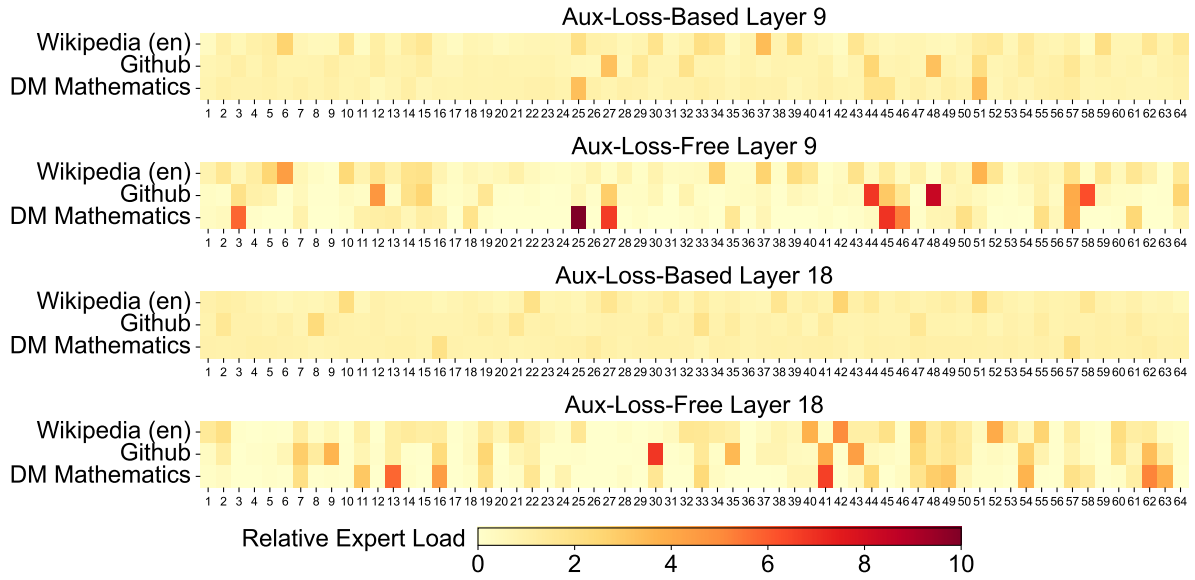


Figure 9 | Expert load of auxiliary-loss-free and auxiliary-loss-based models on three domains in the Pile test set. The auxiliary-loss-free model shows greater expert specialization patterns than the auxiliary-loss-based one. The relative expert load denotes the ratio between the actual expert load 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.

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 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.

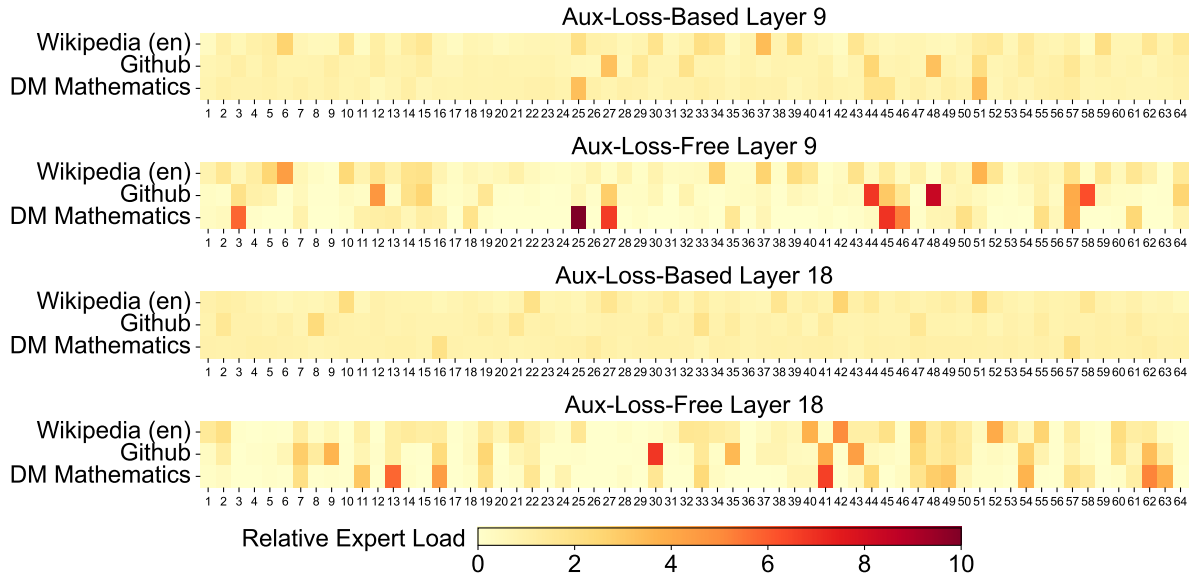


图9 | 在Pile测试集上，辅助无损和基于辅助损失模型在三个领域的专家负载。辅助无损模型显示出比基于辅助损失模型更强的专家专业化模式。相对专家负载表示实际专家负载与理论平衡专家负载之间的比率。由于空间限制，我们仅以两个层的结果作为示例，所有层的结果见附录C。

辅助损失）。我们还观察到在3B MoE模型上有类似的结果：使用序列级辅助损失的模型达到了2.085的验证损失，而使用无辅助损失方法或批量级辅助损失的模型达到了相同的验证损失2.080。

此外，尽管批量负载均衡方法显示出一致的性能优势，但它们在效率上也面临两个潜在挑战：（1）某些序列或小批次内的负载不平衡，以及（2）推理过程中由领域转移引起的负载不平衡。第一个挑战自然通过我们的训练框架得到解决，该框架使用大规模专家并行和数据并行，确保每个微批次的大小较大。对于第二个挑战，我们还设计并实现了一个高效的推理框架，采用冗余专家部署，如第3.4节所述，以克服这一挑战。

5. 训练后

5.1. 监督微调

我们精心策划我们的指令调优数据集，包括150万个实例，涵盖多个领域，每个领域采用不同的数据创建方法，以满足其特定需求。

推理数据。对于与推理相关的数据集，包括那些专注于数学、代码竞赛问题和逻辑难题的数据集，我们通过利用内部的DeepSeek-R1模型生成数据。具体而言，虽然R1生成的数据表现出强大的准确性，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与常规格式化推理数据的清晰性和简洁性。

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 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

为了建立我们的方法论，我们首先开发一个针对特定领域（如代码、数学或一般推理）的专家模型，使用结合监督微调（SFT）和强化学习（RL）的训练流程。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配对，格式为 <问题，原始响应>，而第二种则在问题和R1响应的基础上加入系统提示，格式为 <系统提示，问题，R1响应>。

系统提示经过精心设计，包含指导模型生成富有反思和验证机制的响应的指令。在强化学习阶段，模型利用高温采样生成响应，这些响应整合了来自R1生成的数据和原始数据的模式，即使在没有明确系统提示的情况下也是如此。在数百个强化学习步骤之后，中间强化学习模型学会了整合R1模式，从而战略性地提升整体性能。

在完成RL训练阶段后，我们实施拒绝采样，以策划高质量的SFT数据用于最终模型，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DeepSeek-R1的优势，同时生成简洁有效的响应。

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，我们使用 DeepSeek-V2.5 生成响应，并招募人工注释者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 进行两轮微调，采用从 5×10^{-6} 开始并逐渐降低到 1×10^{-6} 的余弦衰减学习率调度。在训练过程中，每个单独的序列是由多个样本打包而成。然而，我们采用了一种样本掩蔽策略，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. Reward Model

我们在我们的强化学习过程中采用基于规则的奖励模型（RM）和基于模型的奖励模型。

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的结果，我们要求模型在指定格式内（例如，在一个框中）提供最终答案，从而允许我们应用规则来验证正确性。同样，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，我们确保更高的可靠性，因为这种方法抵抗操控或利用。

基于模型的奖励机制。对于具有自由形式真实答案的问题，我们依赖奖励模型来判断响应是否与预期的真实答案相匹配。相反，对于没有明确真实答案的问题，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提供反馈。

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.

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.

[1] https://aider.chat
[2] https://codeforces.com
[3] https://www.cms.org.cn/Home/comp/comp/cid/12.html

作为输入。奖励模型是从DeepSeek-V3 SFT检查点训练而来的。为了增强其可靠性，我们构建了偏好数据，不仅提供最终奖励，还包括导致该奖励的思维链。这种方法有助于降低特定任务中奖励黑客攻击的风险。

5.2.2. Group Relative Policy Optimization

类似于 DeepSeek-V2 (DeepSeek-AI, 2024c)，我们采用了群体相对策略优化 (GRPO) (Shao et al., 2024)，该方法放弃了通常与策略模型大小相同的评论模型，而是从群体得分中估计基线。具体而言，对于每个问题 q，GRPO 从旧的策略模型 $\pi_{\theta_{old}}$ 中抽取一组输出 $\{o_1、o_2、\cdots、o_G\}$，然后通过最大化以下目标来优化策略模型 π_θ：

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

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

其中 ε 和 β 是超参数；π_{ref} 是参考模型；A_i 是优势，源自于对应于每个组内输出的奖励 $\{r_1、r_2、\ldots、r_G\}$：

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

我们在强化学习过程中融入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贴近人类偏好，还提高了基准测试的性能，特别是在可用的监督微调数据有限的情况下。

5.3. 评估

5.3.1. Evaluation Settings

评估基准。除了我们用于基础模型测试的基准外，我们还在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）（2024年8月至2024年11月的问题）、Codeforces [2]、中国国家高中数学奥林匹克（CNMO 2024）[3]以及2024年美国邀请数学考试（AIME 2024）（MAA, 2024）上进一步评估了指令模型。

比较基准。我们对我们的聊天模型进行了全面评估，针对几个强大的基准，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，评估是通过各自的 API 进行的。

[1] https://aider.chat
[2] https://codeforces.com
[3] https://www.cms.org.cn/Home/comp/comp/cid/12.html

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

[4] https://github.com/openai/simple-evals

详细评估配置。对于包括 MMLU、DROP、GPQA 和 SimpleQA 在内的标准基准，我们采用来自 simple-evals 框架的评估提示[4]。我们在零样本设置中对 MMLU-Redux 使用 Zero-Eval 提示格式（Lin, 2024）。对于其他数据集，我们遵循数据集创建者提供的默认提示的原始评估协议。对于代码和数学基准，HumanEval-Mul 数据集总共包括 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 方法来评估模型在 LiveCodeBench 上的表现，数据收集时间为 2024 年 8 月至 2024 年 11 月。Codeforces 数据集的评估使用竞争者的百分比。SWE-Bench 验证使用无代理框架（Xia et al., 2024）进行评估。我们使用“diff”格式来评估与 Aider 相关的基准。对于数学评估，AIME 和 CNMO 2024 的评估温度为 0.7，结果在 16 次运行中取平均，而 MATH-500 则采用贪婪解码。我们允许所有模型在每个基准上输出最多 8192 个标记。

	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**

表6 | DeepSeek-V3与其他代表性聊天模型的比较。所有模型在限制输出长度为8K的配置下进行评估。包含少于1000个样本的基准测试多次使用不同的温度设置进行测试，以得出稳健的最终结果。DeepSeek-V3作为表现最佳的开源模型，同时在与前沿闭源模型的竞争中也展现出竞争力。

[4] https://github.com/openai/simple-evals

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.

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.

5.3.2. Standard Evaluation

表6展示了评估结果，显示DeepSeek-V3是表现最好的开源模型。此外，它在与前沿的闭源模型如GPT-4o和Claude-3.5-Sonnet的竞争中也表现出色。

英语基准。MMLU 是一个广泛认可的基准，旨在评估大型语言模型在不同知识领域和任务中的表现。DeepSeek-V3 展现出竞争力的表现，与顶级模型如 LLaMA-3.1-405B、GPT-4o 和 Claude-Sonnet 3.5 不相上下，同时显著超越 Qwen2.5 72B。此外，DeepSeek-V3 在 MMLU-Pro 中表现出色，这是一个更具挑战性的教育知识基准，在该基准中，它与 Claude-Sonnet 3.5 紧密相随。在 MMLU-Redux 上，MMLU 的一个经过修正标签的精炼版本，DeepSeek-V3 超越了其同行。此外，在 GPQA-Diamond 这个博士级评估测试平台上，DeepSeek-V3 取得了显著的成绩，仅次于 Claude 3.5 Sonnet，并且大幅超越了所有其他竞争对手。

在长文本理解基准测试中，如DROP、LongBench v2和FRAMES，DeepSeek-V3继续展示其作为顶级模型的地位。在DROP的3-shot设置中，它取得了令人印象深刻的91.6 F1分数，超越了该类别的所有其他模型。在FRAMES中，这是一项需要在超过10万标记上下文中进行问答的基准，DeepSeek-V3紧随GPT-4o之后，同时显著超越了所有其他模型。这展示了DeepSeek-V3在处理极长上下文任务方面的强大能力。DeepSeek-V3的长文本能力在LongBench v2上的最佳表现进一步得到了验证，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落后于GPT-4o和Claude-Sonnet，主要是由于其设计重点和资源分配。DeepSeek-V3分配更多的训练标记来学习中文知识，从而在C-SimpleQA上表现出色。在遵循指令的基准测试中，DeepSeek-V3显著超越了其前身DeepSeek-V2系列，突显了其理解和遵循用户定义格式约束的能力的提升。

代码和数学基准。编码是一个具有挑战性和实用性的任务，对于 LLM 来说，涵盖了以工程为中心的任务，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 预计将促进与编码相关的工程任务的进步。通过提供其强大的能力，DeepSeek-V3 可以推动软件工程和算法开发等领域的创新和改进，使开发者和研究人员能够突破开源模型在编码任务中可以实现的界限。在算法任务中，DeepSeek-V3 展现出卓越的性能，在 HumanEval-Mul 和 LiveCodeBench 等基准测试中超越了所有基线。这一成功可归因于其先进的知识蒸馏技术，有效增强了其在算法集中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 展现了卓越的性能，显著超越了基线，并为非 o1 类模型设定了新的最先进水平。具体而言，在 AIME、MATH-500 和 CNMO 2024 上，DeepSeek-V3 的绝对得分比第二名模型 Qwen2.5 72B 高出约 10%，这对于如此具有挑战性的基准测试来说是一个相当大的差距。这一显著能力突显了 DeepSeek-R1 中蒸馏技术的有效性，该技术已被证明对非 o1 类模型极为有利。

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.

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 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

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**

表 7 | 英文开放式对话评估。对于 AlpacaEval 2.0，我们使用长度控制的胜率作为指标。

中文基准。Qwen 和 DeepSeek 是两个在中文和英文方面都有强大支持的代表性模型系列。在事实基准中文 SimpleQA 上，DeepSeek-V3 超过了 Qwen2.5-72B 16.4 分，尽管 Qwen2.5 是在一个更大的语料库上训练的，包含 18T 令牌，比 DeepSeek-V3 预训练的 14.8T 令牌多出 20%。

在C-Eval，一个代表性的中文教育知识评估基准，以及CLUEWSC（中文Winograd模式挑战），DeepSeek-V3和Qwen2.5-72B表现出相似的性能水平，表明这两个模型在具有挑战性的中文推理和教育任务上都经过了良好的优化。

5.3.3. Open-Ended Evaluation

除了标准基准测试，我们还使用大型语言模型（LLMs）作为评判者，在开放式生成任务上评估我们的模型，结果如表7所示。具体而言，我们遵循AlpacaEval 2.0（Dubois等，2024）和Arena-Hard（Li等，2024a）的原始配置，这些配置利用GPT-4-Turbo-1106作为成对比较的评判者。在Arena-Hard上，DeepSeek-V3以超过86%的令人印象深刻的胜率击败基线GPT-4-0314，表现与Claude-Sonnet-3.5-1022等顶级模型相当。这突显了DeepSeek-V3的强大能力，尤其是在处理复杂提示时，包括编码和调试任务。此外，DeepSeek-V3作为第一个在Arena-Hard基准上超过85%的开源模型，达成了一个突破性的里程碑。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，为开源模型在挑战性领域的成就设定了新的标准。

同样，DeepSeek-V3 在 AlpacaEval 2.0 上展现了卓越的性能，超越了闭源和开源模型。这证明了它在写作任务和处理简单问答场景方面的出色能力。值得注意的是，它比 DeepSeek-V2.5-0905 超出了 20% 的显著差距，突显了在处理简单任务方面的重大改进，并展示了其进步的有效性。

5.3.4. DeepSeek-V3 as a Generative Reward Model

我们将DeepSeek-V3的判断能力与最先进的模型进行比较，即GPT-4o和Claude-3.5。表8展示了这些模型在RewardBench中的表现（Lambert等，2024）。DeepSeek-V3的表现与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的判断能力还可以通过投票技术得到增强。因此，我们采用DeepSeek-V3结合投票技术，对开放式问题提供自我反馈，从而提高{v*}。

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.

effectiveness and robustness of the alignment process.

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

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

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

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

表9 | DeepSeek-R1的蒸馏贡献。Live-CodeBench和MATH-500的评估设置与表6相同。

对齐过程的有效性和稳健性。

5.4. 讨论

5.4.1. Distillation from DeepSeek-R1

我们基于 DeepSeek-V2.5 消除了来自 DeepSeek-R1 的蒸馏贡献。基线是在短 CoT 数据上训练的，而其竞争对手使用的是上述专家检查点生成的数据。

表9展示了蒸馏数据的有效性，在LiveCodeBench和MATH-500基准测试中均显示出显著的改进。我们的实验揭示了一个有趣的权衡：蒸馏导致了更好的性能，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，我们仔细选择了DeepSeek-V3在蒸馏中的最佳设置。

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。虽然我们目前的工作集中在从数学和编码领域提取数据，但这种方法在各种任务领域中显示出更广泛应用的潜力。在这些特定领域中展示的有效性表明，长链推理（long-CoT）蒸馏可能对提高其他需要复杂推理的认知任务中的模型性能具有重要价值。进一步探索这种方法在不同领域的应用仍然是未来研究的重要方向。

5.4.2. Self-Rewarding

奖励在强化学习中发挥着关键作用，指导优化过程。在通过外部工具进行验证相对简单的领域，例如某些编码或数学场景中，强化学习表现出卓越的有效性。然而，在更一般的场景中，构建反馈

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 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

通过硬编码的机制是不切实际的。在DeepSeek-V3的开发过程中，对于这些更广泛的背景，我们采用了宪法AI方法（Bai et al., 2022），利用DeepSeek-V3自身的投票评估结果作为反馈来源。这种方法产生了显著的对齐效果，显著提升了DeepSeek-V3在主观评估中的表现。通过整合额外的宪法输入，DeepSeek-V3可以朝着宪法方向进行优化。我们相信，这种将补充信息与LLMs结合作为反馈来源的范式至关重要。LLM作为一个多功能处理器，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LLMs的自我改进。除了自我奖励，我们还致力于发现其他通用和可扩展的奖励方法，以持续提升模型在一般场景中的能力。

5.4.3. Multi-Token Prediction Evaluation

DeepSeek-V3不仅仅预测下一个单一的标记，而是通过MTP技术预测下两个标记。结合推测解码的框架（Leviathan等，2023；Xia等，2023），它可以显著加快模型的解码速度。一个自然的问题是关于额外预测标记的接受率。根据我们的评估，第二个标记预测的接受率在85%到90%之间，涵盖了各种生成主题，显示出一致的可靠性。这一高接受率使DeepSeek-V3能够实现显著提高的解码速度，提供1.8倍的TPS（每秒标记数）。

6. 结论、局限性和未来方向

在本文中，我们介绍了DeepSeek-V3，这是一种具有671B总参数和37B激活参数的大型MoE语言模型，训练于14.8T个标记。除了MLA和DeepSeekMoE架构外，它还开创了一种无辅助损失的负载平衡策略，并设定了多标记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。后训练阶段也成功地从DeepSeek-R1系列模型中提炼了推理能力。全面评估表明，DeepSeek-V3已成为当前最强的开源模型，其性能可与领先的闭源模型如GPT-4o和Claude-3.5-Sonnet相媲美。尽管性能强劲，但它仍保持经济的训练成本。其完整训练仅需2.788M H800 GPU小时，包括预训练、上下文长度扩展和后训练。

虽然承认其强大的性能和成本效益，我们也认识到DeepSeek-V3存在一些局限性，特别是在部署方面。首先，为了确保高效的推理，DeepSeek-V3的推荐部署单元相对较大，这可能对小型团队构成负担。其次，尽管我们针对DeepSeek-V3的部署策略已实现了超过DeepSeek-V2两倍的端到端生成速度，但仍然存在进一步提升的潜力。幸运的是，随着更先进硬件的发展，这些局限性预计会自然得到解决。

DeepSeek 始终坚持开放源代码模型的长期主义路线，旨在稳步接近 AGI（人工通用智能）的最终目标。未来，我们计划在以下方向上进行战略性研究投资。

- 我们将持续研究和完善我们的模型架构，旨在进一步提高 {v*}。

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.
- 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,

在训练和推理效率方面，努力接近对无限上下文长度的高效支持。此外，我们将尝试突破Transformer的架构限制，从而推动其建模能力的边界。

- 我们将不断迭代我们的训练数据的数量和质量，并探索纳入额外训练信号源，旨在推动数据在更全面的维度范围内的扩展。
- 我们将持续探索和迭代我们模型的深度思维能力，旨在通过扩展它们的推理长度和深度来增强它们的智能和解决问题的能力。
- 我们将探索更全面和多维的模型评估方法，以防止在研究过程中倾向于优化一组固定的基准，这可能会对模型能力产生误导性的印象，并影响我们的基础评估。

参考文献

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

AI@Meta。Llama 3.1 模型卡，2024b。网址 https://github.com/meta-llama/llama-models/blob/main/models/llama3_1/MODEL_CARD.md。

人类学。克劳德 3.5 颂，2024。网址 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 等人。使用大型语言模型进行程序合成。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, 等人。宪法人工智能：来自人工智能反馈的无害性。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: 朝着对现实长上下文多任务的更深入理解和推理. arXiv 预印本 arXiv:2412.15204, 2024.

M. Bauer, S. Treichler 和 A. Aiken. Singe: 利用 warp 专门化在 GPU 上实现高性能. 在第 19 届 ACM SIGPLAN 并行编程原理与实践研讨会论文集，PPoPP ’14，页 119–130，纽约，NY，美国，2014. 计算机协会. ISBN 9781450326568. doi: 10.1145/2555243.2555258. URL https://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。在第三十四届人工智能AAAI会议，AAAI 2020，第三十二届人工智能创新应用会议，IAAI 2020，第十届人工智能教育进展AAAI研讨会，EAAI 2020，美国纽约，2020年2月7-12日，页码7432–7439。AAAI出版社，2020。doi: 10.1609/aaai.v34i05.6239。网址 https://doi.org/10.1609/aaai.v34i05.6239。

M. 陈, 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.

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, 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. 认为你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 等人。训练验证者解决数学文字问题。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跨度提取数据集。在 K. Inui, J. Jiang, V. Ng, 和 X. Wan 编辑的《2019年自然语言处理实证方法会议暨第九届国际联合自然语言处理会议（EMNLP-IJCNLP）论文集》，第 5883–5889 页，香港，中国，2019年11月。计算语言学协会。doi: 10.18653/v1/D19-1600。网址 https://aclanthology.org/D19-1600。

D. 戴, C. 邓, C. 赵, R. X. 许, H. 高, D. 陈, J. 李, W. 曾, X. 余, Y. 吴, Z. 谢, Y. K. 李, P. 黄, F. 罗, C. 阮, Z. 隋, 和 W. 梁. Deepseekmoe: 朝着混合专家语言模型的终极专家专业化迈进. CoRR, abs/2401.06066, 2024. URL https://doi.org/10.48550/arXiv.2401.06066.

DeepSeek-AI。Deepseek-coder-v2：打破代码智能中闭源模型的壁垒。CoRR， abs/2406.11931， 2024a。网址 https://doi.org/10.48550/arXiv.2406.11931。

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

DeepSeek-AI。Deepseek-v2：一种强大、经济且高效的专家混合语言模型。CoRR， abs/2405.04434， 2024c。网址 https://doi.org/10.48550/arXiv.2405.04434。

T. Dettmers, M. Lewis, Y. Belkada 和 L. Zettlemoyer. Gpt3. int8 (): 大规模变换器的 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. 更少的截断改善语言建模. arXiv 预印本 arXiv:2404.10830, 2024.

D. Dua, Y. Wang, P. Dasigi, G. Stanovsky, S. Singh 和 M. Gardner. DROP：一个需要对段落进行离散推理的阅读理解基准。在 J. Burstein, C. Doran 和 T. Solorio 主编的《2019年北美计算语言学协会会议论文集：人类语言技术，NAACL-HLT 2019》，明尼阿波利斯，MN，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。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.

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.

Y. Dubois, B. Galambosi, P. Liang, 和 T. B. Hashimoto. 长度控制的 alpacaeval：一种简单的去偏自动评估方法。arXiv 预印本 arXiv:2404.04475, 2024. W. Fedus, B. Zoph, 和 N. Shazeer. Switch transformers：以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 训练扩展到万亿标记的 llms。arXiv 预印本 arXiv:2409.12517, 2024. E. Frantar, S. Ashkboos, T. Hoefler, 和 D. Alistarh. Gptq：针对生成预训练变换器的准确后训练量化。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 等人。《The Pile：一个用于语言建模的800GB多样文本数据集》。arXiv预印本 arXiv:2101.00027，2020年。

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 了吗？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型语言模型。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日至27日。OpenReview.net，2024。网址 https://openreview.net/forum?id=pEWAcejiU2。

谷歌。我们的下一代模型：Gemini 1.5，2024。网址 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. 可扩展的分层聚合协议 (SHArP)：一种高效数据压缩的硬件架构. 在 2016 年第一次国际高性能计算通信优化研讨会 (COMHPC) 上，第 1–10 页. IEEE, 2016.

A. Gu, B. Rozière, H. Leather, A. Solar-Lezama, G. Synnaeve, 和 S. I. Wang. Cruxeval: 一项用于代码推理、理解和执行的基准，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, 和 W. Liang. Deepseek-coder: 当大型语言模型遇到编程 - 代码智能的崛起. 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, 和 P. Gibbons. Pipedream: 快速高效的管道并行 DNN 训练, 2018. URL https://arxiv.org/abs/1806.03377.

B. He, L. Noci, D. Paliotta, I. Schlag 和 T. Hofmann. 理解和最小化变压器训练中的离群特征. 在第三十八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. 中文简单问答：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, and J. Steinhardt. Measuring massive multitask language understanding. arXiv preprint 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. 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, 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: 一个多层次多学科的中文评估套件，用于基础模型. 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: 大型语言模型代码的整体和无污染评估. 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 等人。Mistral 7b。arXiv 预印本 arXiv:2310.06825，2023。

M. Joshi, E. Choi, D. Weld 和 L. Zettlemoyer. TriviaQA：一个大规模远程监督的挑战数据集，用于阅读理解。在 R. Barzilay 和 M.-Y. Kan 主编的《计算语言学协会第55届年会论文集（第1卷：长篇论文）》中，页面 1601–1611，加拿大温哥华，2017年7月。计算语言学协会。doi: 10.18653/v1/P17-1147。网址 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 等人。关于 bfloat16 在深度学习训练中的研究。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、获取和推理：检索增强生成的统一评估。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：一个用于问答研究的基准。计算语言学协会会刊，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：来自考试的大规模阅读理解数据集。在 M. Palmer, R. Hwa, 和 S. Riedel 编辑的《2017年自然语言处理实证方法会议论文集》，EMNLP 2017，丹麦哥本哈根，2017年9月9-11日，页码 785–794。计算语言学协会，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 等人。Rewardbench：评估语言建模的奖励模型。arXiv 预印本 arXiv:2403.13787，2024。

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来扩展巨型模型. 在第九届国际学习表征会议，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. 通过推测解码从变换器快速推断. 在国际机器学习会议，ICML 2023，2023年7月23日至29日，夏威夷檀香山，美国，机器学习研究会议论文集第202卷，页码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, 和 T. Baldwin. CMMLU：在中文中测量大规模多任务语言理解。arXiv 预印本 arXiv:2306.09212, 2023.

S. Li 和 T. Hoefler. Chimera：高效训练大规模神经网络的双向管道。在国际高性能计算、网络、存储与分析会议论文集，SC '21，第 1–14 页。ACM，2021 年 11 月。doi: 10.1145/3458817.3476145。网址 http://dx.doi.org/10.1145/3458817.3476145。

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

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

Y. Li, F. Wei, C. Zhang, 和 H. Zhang. EAGLE: 推测采样需要重新思考特征不确定性. 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, 2024b. URL https://openreview.net/forum?id=1NdN7eXyb4.

B. Y. Lin. ZeroEval：评估语言模型的统一框架，2024年7月。网址 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：提示边界和令牌修复，2023。网址 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. 深度学习的 Ascend HiFloat8 格式. arXiv 预印本 arXiv:2409.16626, 2024.

MAA。美国邀请数学考试 - AIME。在2024年2月的美国邀请数学考试 - AIME 2024。网址 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 等人。深度学习的 FP8 格式。arXiv 预印本 arXiv:2209.05433, 2022。

米斯特拉尔。更便宜、更好、更快、更强：继续推动人工智能的前沿，使其对所有人可及，2024年。网址 https://mistral.ai/news/mixtral-8x22b。

S. Narang, G. Diamos, E. Elsen, P. Micikevicius, J. Alben, D. Garcia, B. Ginsburg, M. Hous- ton, O. Kuchaiev, G. Venkatesh 等人。混合精度训练。在国际学习表征会议，2017年。

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. 深度神经网络的 8 位数值格式. 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。网址 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。网址 https://openai.com/index/introducing-simpleqa/。

OpenAI。介绍SWE-bench验证，我们发布了一个经过人工验证的swe-bench子集，更多信息见2024d。网址 https://openai.com/index/introducing-swe-bench-verified/。

B. Peng, J. Quesnelle, H. Fan, 和 E. Shippole. Yarn: 大型语言模型的高效上下文窗口扩展. 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. FP8-LM：训练 FP8 大型语言模型。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。网址 https://qwenlm.github.io/blog/qwen1.5。

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

S. Rajbhandari, J. Rasley, O. Ruwase, 和 Y. He. Zero: 针对训练万亿参数模型的内存优化. 在 SC20: 高性能计算、网络、存储和分析国际会议, 页码 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防护问答基准。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. 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, 2023b。

K. Sakaguchi, R. L. Bras, C. Bhagavatula 和 Y. Choi. Winogrande：一个大规模的对抗性 Winograd 语法挑战，2019。

Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, M. Zhang, Y. Li, Y. Wu, 和 D. Guo. Deepseekmath: 在开放语言模型中推动数学推理的极限. 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. 在第五届国际学习表征会议，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. 语言模型是多语言的链式思维推理者。在第十一届国际学习表征会议，ICLR 2023，卢旺达基加利，2023年5月1-5日。OpenReview.net，2023。网址 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: 带有旋转位置嵌入的增强型变换器. Neurocomputing, 568:127063, 2024.

K. Sun, D. Yu, D. Yu, 和 C. Cardie. 研究挑战性中文机器阅读理解的先验知识, 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. 混合8位浮点数（HFP8）训练和推理用于深度神经网络. 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 等人。挑战大基准任务以及思维链是否能解决这些任务。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 等人。LLaMA：开放和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, 和 T. Scialom. Llama 2：开放基础和微调聊天模型。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, . Kaiser, 和 I. Polo-sukhin. 注意力是你所需要的一切. 神经信息处理系统进展, 30, 2017.

L. Wang, H. Gao, C. Zhao, X. Sun 和 D. Dai. 无辅助损失的混合专家负载平衡策略. 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, 和 W. Chen. Mmlu-pro: 一个更强大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. 神经信息处理系统进展, 36:10271–10298, 2023.

H. Xi, C. Li, J. Chen 和 J. Zhu. 用 4 位整数训练变换器. 神经信息处理系统进展, 36:49146–49168, 2023.

C. S. Xia, Y. Deng, S. Dunn, 和 L. Zhang. 无代理: 揭示基于 llm 的软件工程代理的奥秘. 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. 在计算语言学协会的发现：EMNLP 2023，新加坡，2023年12月6-10日，页面3909–3925. 计算语言学协会，2023. URL https://doi.org/10.18653/v1/2023.findings-emnlp.257.

G. Xiao, J. Lin, M. Seznec, H. Wu, J. Demouth, 和 S. Han. Smoothquant: 大型语言模型的准确高效后训练量化. 在国际机器学习会议, 页码 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, 和 Z. Lan. CLUE：中文理解评估基准。在 D. Scott, N. Bel, 和 C. Zong 编辑的《第28届国际计算语言学会议论文集》，COLING 2020，西班牙巴塞罗那（在线），2020年12月8-13日，页4762–4772。国际计算语言学委员会，2020。doi: 10.18653/V1/2020.COLING-MAIN.419。网址 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: 机器真的能完成你的句子吗? 在 A. Korhonen, D. R. Traum 和 L. Màrquez 编辑的《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日至8月2日，第1卷：长篇论文，页码 4791–4800. 计算语言学协会，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, 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, 和 L. Hou. 大型语言模型的指令跟随评估. 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
爱心 刘 冰 雪 冰
轩 王 博超 吴 成
达 陆 成刚 赵 成
奇 邓 晨宇 张* 冲
阮 达麦 戴 达雅
郭 德建 杨 德利
陈 二航 李 方云
林 复聪 戴 复利
罗* 光博 郝 观婷
陈 国伟 李 H. 张
汉 宝* 汉伟 许 浩
城 王* 浩伟 张 洪
辉 丁 华建 辛* 华
作 高 辉 曲 建中
郭 家士 李 家伟
王* 京畅 陈 京阳
袁 军杰 邱 军龙
李 军晓 宋 凯 董
凯 胡* 凯歌 高 康
关 可欣 黄 快宇 L
ean 王
张乐聪 张亮 赵丽
彤 王丽月 张明川
张明华 张明辉 唐
盼盼 黄佩怡 王千
城 王启浩 朱沁宇
陈秋实 杜瑞琪 葛
瑞松 张瑞哲 潘润
基 王润欣 许若宇
张尚浩 陆尚妍 周
山煌 陈胜峰 叶世
荣 马诗雨 王水平
于顺风 周淑婷 潘
涛 云天 佩王丁 曾
万佳 赵* 文 刘文峰
梁文俊 高文琴 于
文涛 张小碧 夏东
刘晓寒 王晓康 陈
晓康 张晓涛 聂欣
程欣 刘

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

辛谢星潮 刘星凯 余新宇 杨新源 李雪城 苏旭恒 林Y.K. 李Y.Q. 王Y.X. 魏阳 张艳红 许瑶李瑶 赵耀峰 孙耀辉 王毅 余义超 张逸凡 石义良 熊颖何怡诗 票怡松 王逸轩 谭怡扬 马*逸源 刘永强 郭宇吴元欧 余端 王跃龚宇恒 邹宇佳 何云帆 熊宇翔 罗宇翔 游宇轩 刘宇扬 周Z.F. 吴Z.Z. 任泽辉 任张丽 沙哲傅哲安 许振达 谢正言 张哲文 郝志斌 龚志成 马

严志刚 邵志宏 吴志宇 李卓书 资慧 顾自佳 朱子俊 刘子林 李子维 谢子扬 宋子怡 高子正 潘

数据标注 贝风 惠利 J.L. 蔡家琪 倪雷 许梦 李宁 天 R.J. 陈 R.L. 金如意 陈 S.S. 李双 周天宇 孙 X.Q. 李向月 金小金 沈晓莎 陈晓文 孙小湘 王新南 宋新怡 周 Y.X. 朱艳红 许燕平 黄耀辉 李毅 郑宇辰 朱云仙 马震 黄志鹏 许忠宇 张

商业与合规 纪东杰

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

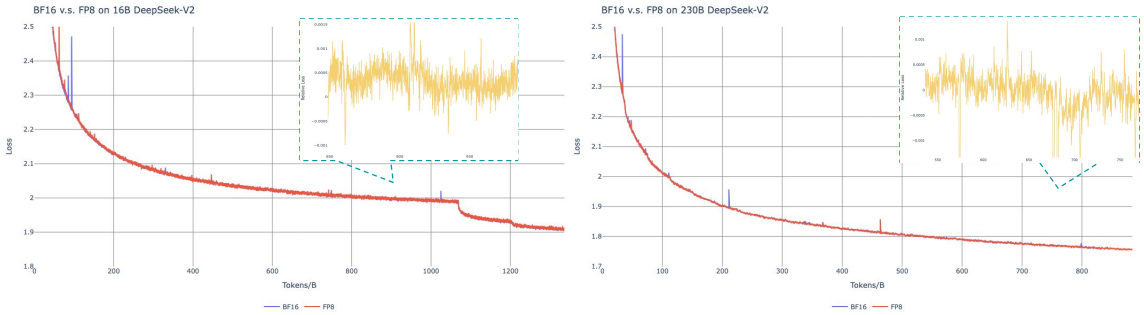


Figure 10 | Loss curves comparison between BF16 and FP8 training. Results are smoothed by Exponential Moving Average (EMA) with a coefficient of 0.9.

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

简良金晨乐怡
夏妙君王明明
李鹏张少青吴
胜峰叶T.王

肖伟 安 仙祖
王新霞 单颖
唐玉坤 赵玉
婷 闫真 张

在每个角色中，作者按名字的字母顺序列出。标有 * 的名字表示已离开我们团队的个人。

B. 低精度训练的消融研究

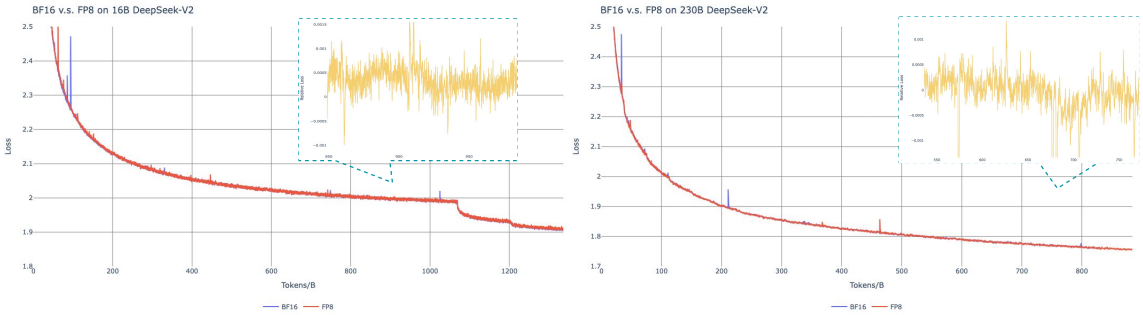


图10 | BF16和FP8训练的损失曲线比较。结果通过指数移动平均（EMA）平滑，系数为0.9。

B.1. FP8 与 BF16 训练

我们通过与两种基线模型在不同规模上的BF16训练进行比较，验证了我们的FP8混合精度框架。在小规模上，我们在1.33T的token上训练了一个基线MoE模型，包含大约16B的总参数。在大规模上，我们在大约0.9T的token上训练了一个基线MoE模型，包含大约230B的总参数。我们在图10中展示了训练曲线，并证明我们的高精度累积和细粒度量化策略使相对误差保持在0.25%以下。

B.2. 关于块级量化的讨论

尽管我们的瓷砖级细粒度量化有效地减轻了特征异常值引入的误差，但它需要对激活量化进行不同的分组，即在前向传播中使用 `1x128`，在反向传播中使用 `128x1`。激活梯度也需要类似的过程。一种简单的策略是对每 `128x128` 元素应用块级量化，就像我们量化模型权重的方式一样。通过这种方式，反向传播只需要转置。因此，我们进行了一项实验，其中与 `Dgrad` 相关的所有张量都以块级方式进行量化。结果表明，计算激活梯度并以链式方式反向传播到浅层的 `Dgrad` 操作对精度非常敏感。具体来说，激活梯度的块级量化导致了

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.

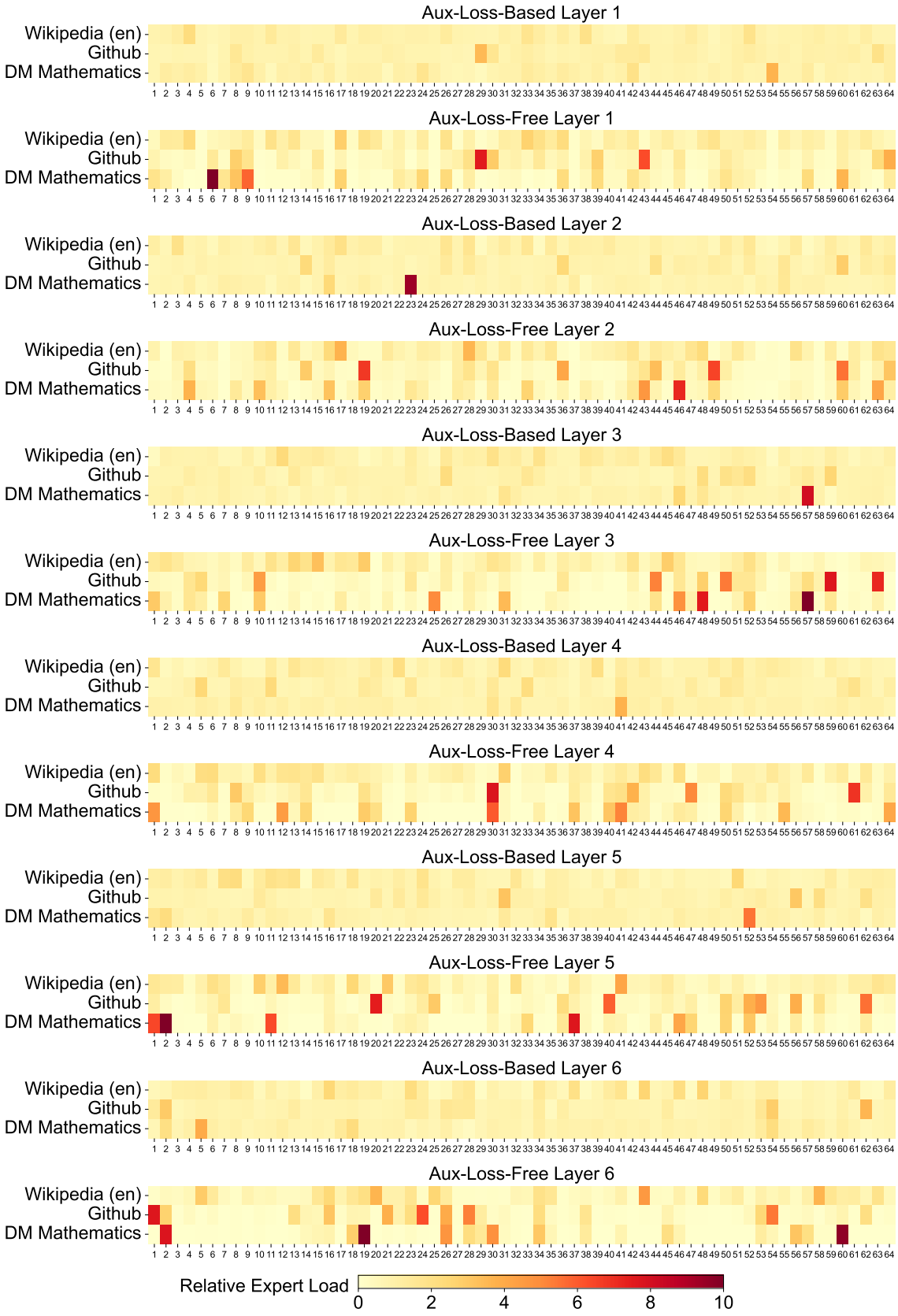
C. Expert Specialization Patterns of the 16B Aux-Loss-Based and Aux-Loss-Free Models

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.

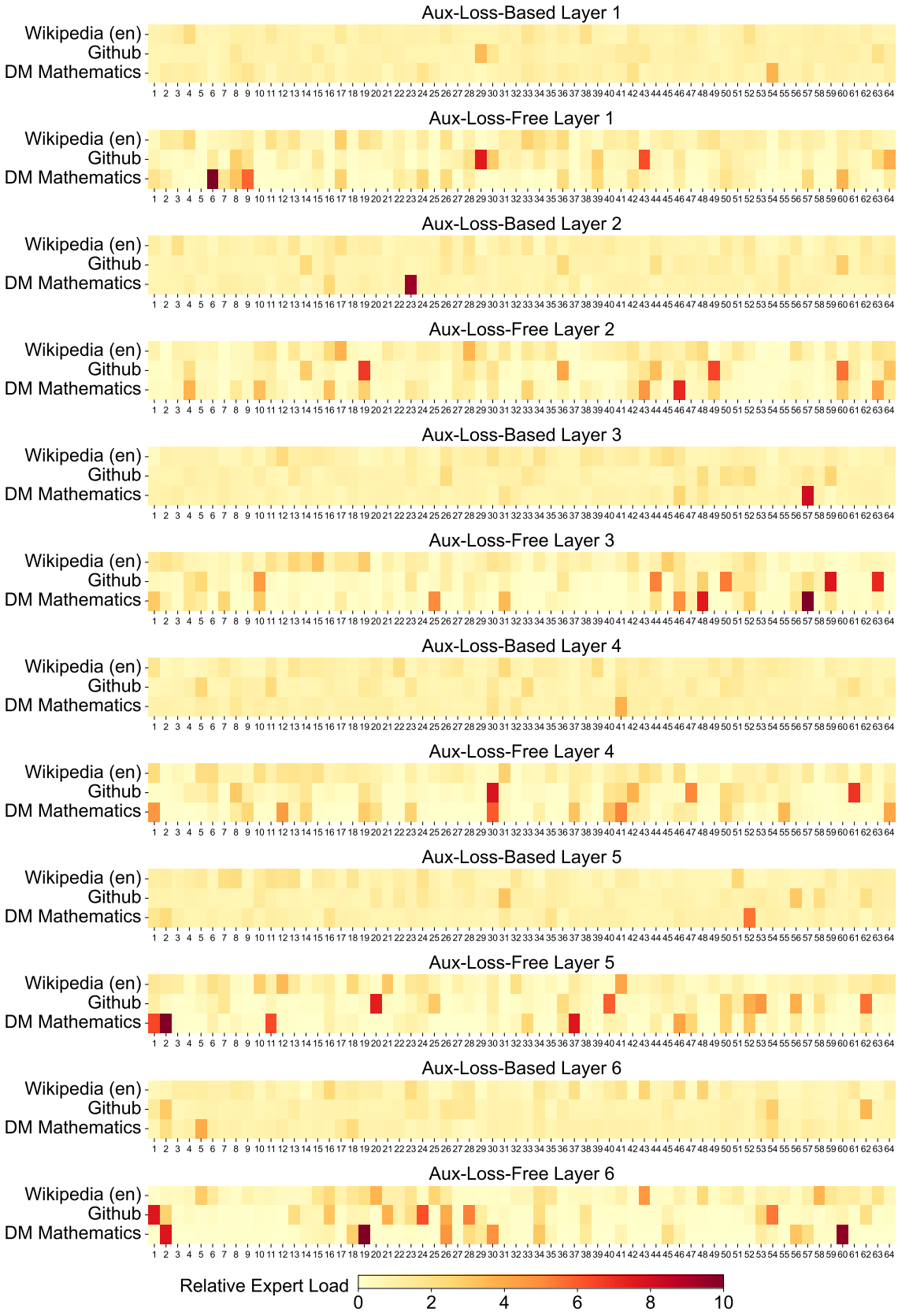
在一个包含大约160亿个总参数的MoE模型上，模型发散，训练了大约3000亿个标记。我们假设这种敏感性产生的原因是激活梯度在标记之间高度不平衡，导致与标记相关的异常值（Xi等，2023）。这些异常值无法通过块级量化方法有效管理。

C. 16B 辅助损失基础模型和无辅助损失模型的专家专业化模式

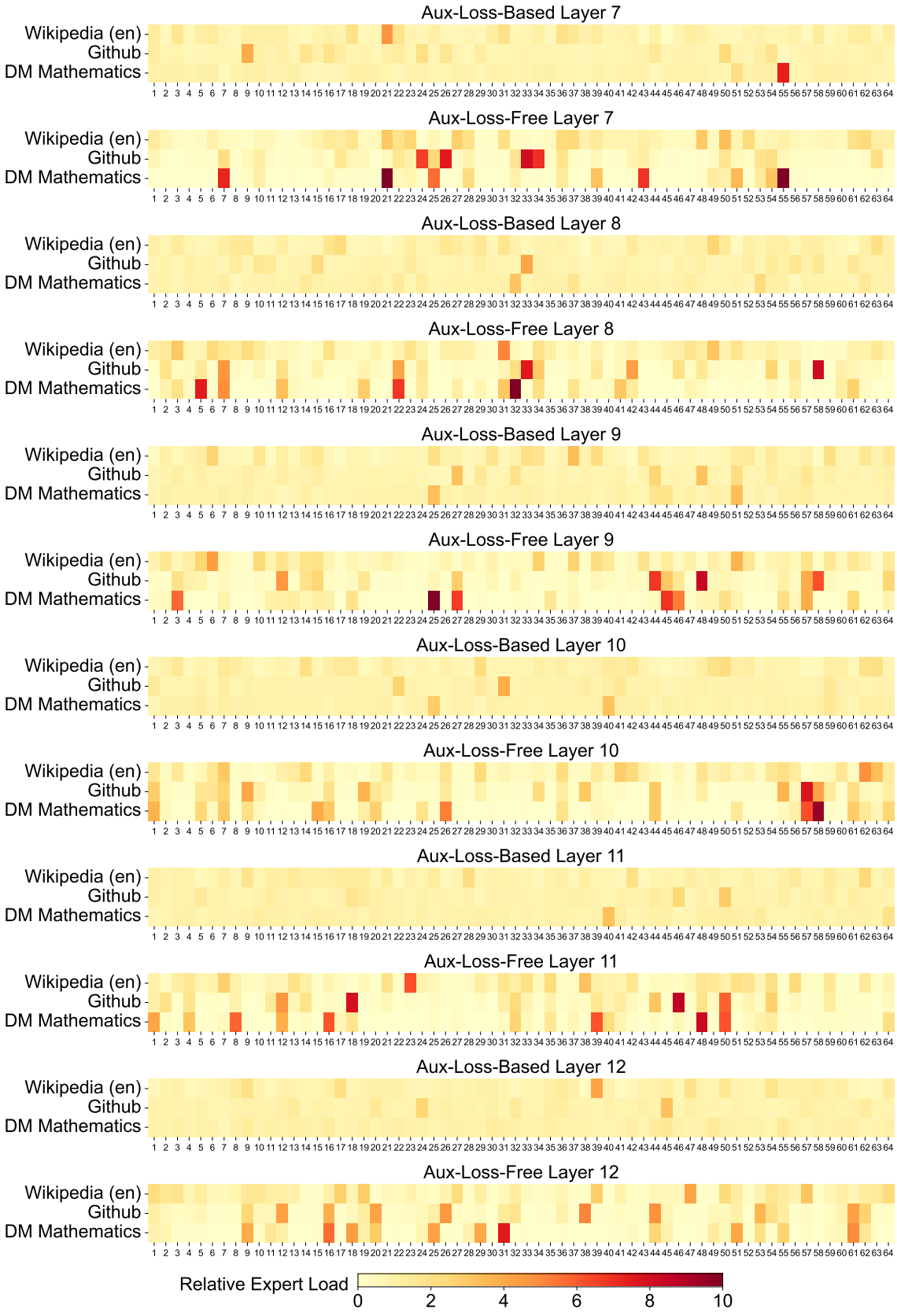
我们记录了16B辅助损失基线和无辅助损失模型在Pile测试集上的专家负载。无辅助损失模型在所有层中往往具有更大的专家专业化，如图10所示。


Aux-Loss-Based Layer 1
Aux-Loss-Free Layer 1
Aux-Loss-Based Layer 2
Aux-Loss-Free Layer 2
Aux-Loss-Based Layer 3
Aux-Loss-Free Layer 3
Aux-Loss-Based Layer 4
Aux-Loss-Free Layer 4
Aux-Loss-Based Layer 5
Aux-Loss-Free Layer 5
Aux-Loss-Based Layer 6
Aux-Loss-Free Layer 6
Wikipedia (en)
Github
DM Mathematics
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64
Relative Expert Load
0
2
4
6
8
10

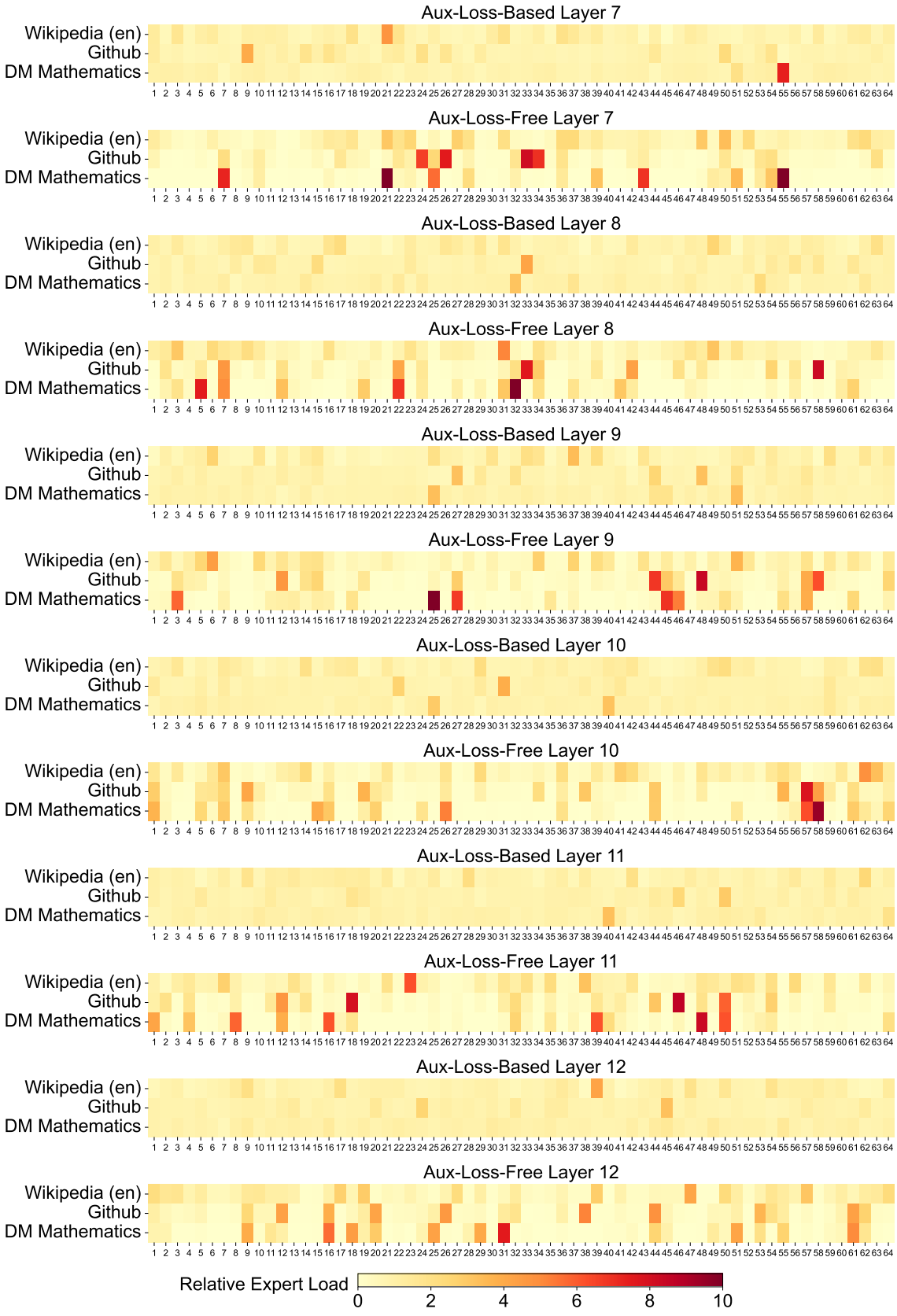
(a) Layers 1-7


Aux-Loss-Based Layer 1
Aux-Loss-Free Layer 1
Aux-Loss-Based Layer 2
Aux-Loss-Free Layer 2
Aux-Loss-Based Layer 3
Aux-Loss-Free Layer 3
Aux-Loss-Based Layer 4
Aux-Loss-Free Layer 4
Aux-Loss-Based Layer 5
Aux-Loss-Free Layer 5
Aux-Loss-Based Layer 6
Aux-Loss-Free Layer 6
Wikipedia (en)
Github
DM Mathematics
Relative Expert Load
0
2
4
6
8
10

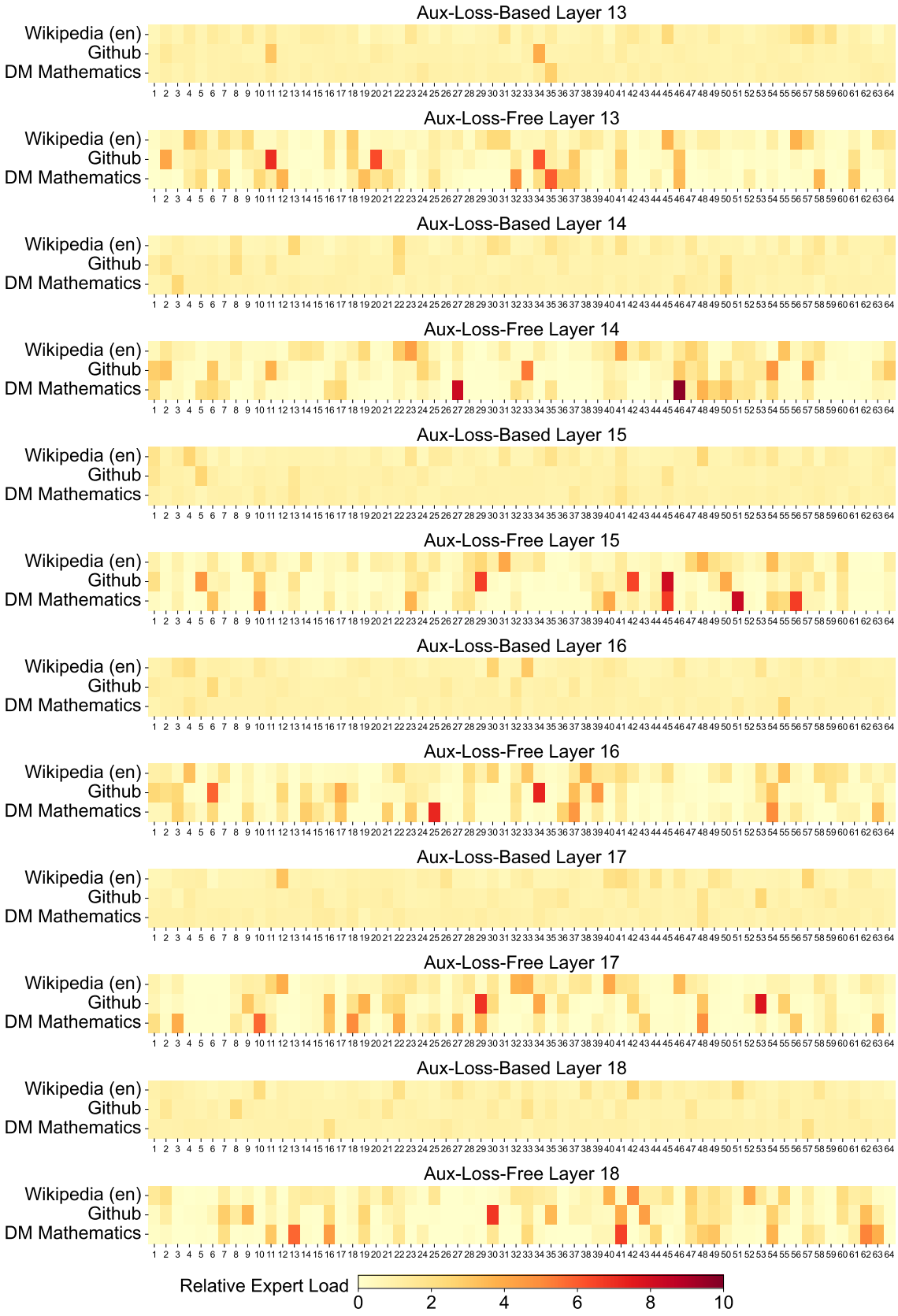
(a) Layers 1-7


Aux-Loss-Based Layer 7
Aux-Loss-Free Layer 7
Aux-Loss-Based Layer 8
Aux-Loss-Free Layer 8
Aux-Loss-Based Layer 9
Aux-Loss-Free Layer 9
Aux-Loss-Based Layer 10
Aux-Loss-Free Layer 10
Aux-Loss-Based Layer 11
Aux-Loss-Free Layer 11
Aux-Loss-Based Layer 12
Aux-Loss-Free Layer 12
Wikipedia (en)
Github
DM Mathematics
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64
Relative Expert Load
0
2
4
6
8
10

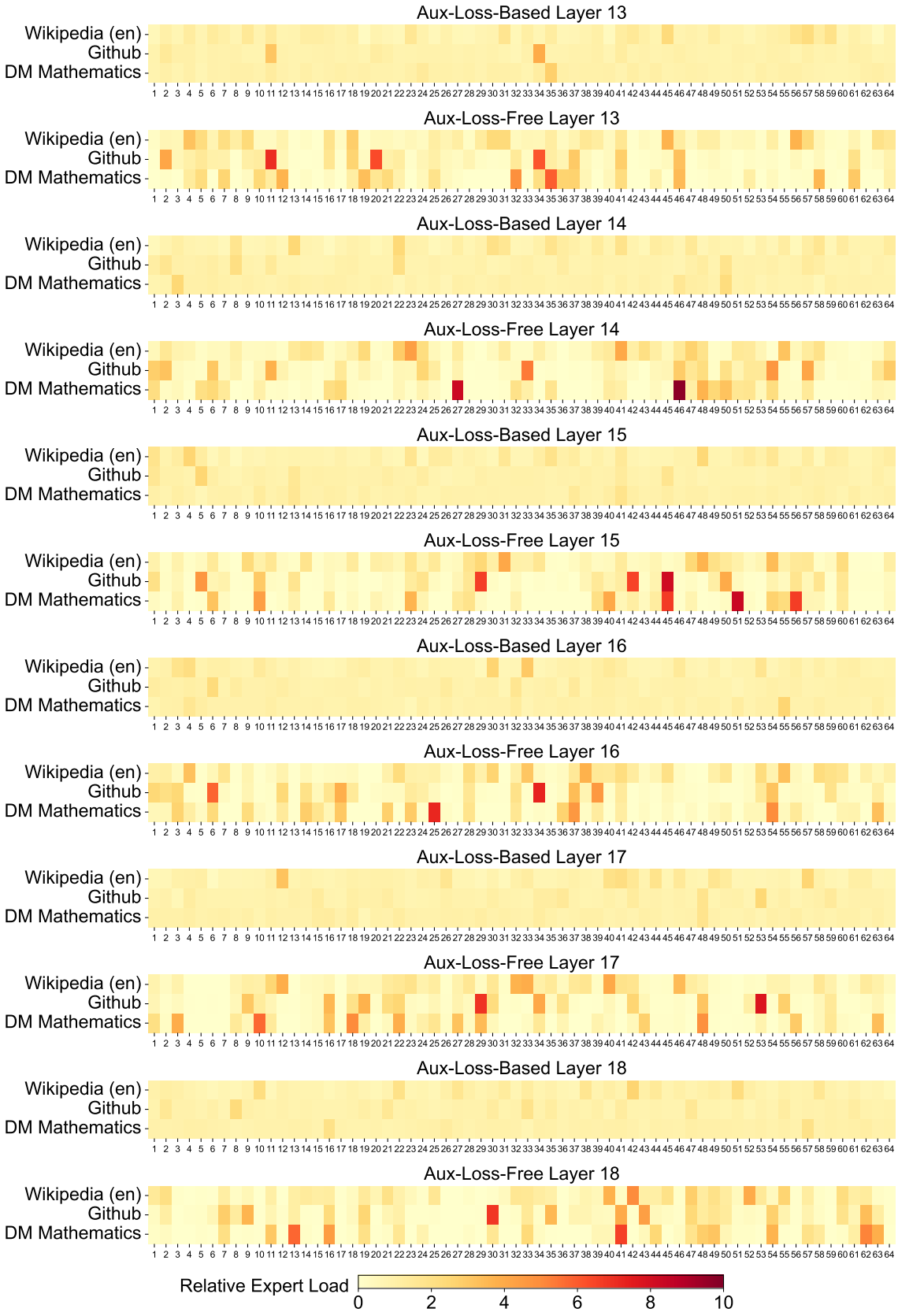
(b) Layers 7-13


Aux-Loss-Based Layer 7
Aux-Loss-Free Layer 7
Aux-Loss-Based Layer 8
Aux-Loss-Free Layer 8
Aux-Loss-Based Layer 9
Aux-Loss-Free Layer 9
Aux-Loss-Based Layer 10
Aux-Loss-Free Layer 10
Aux-Loss-Based Layer 11
Aux-Loss-Free Layer 11
Aux-Loss-Based Layer 12
Aux-Loss-Free Layer 12
Wikipedia (en)
Github
DM Mathematics
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64
Relative Expert Load
0
2
4
6
8
10

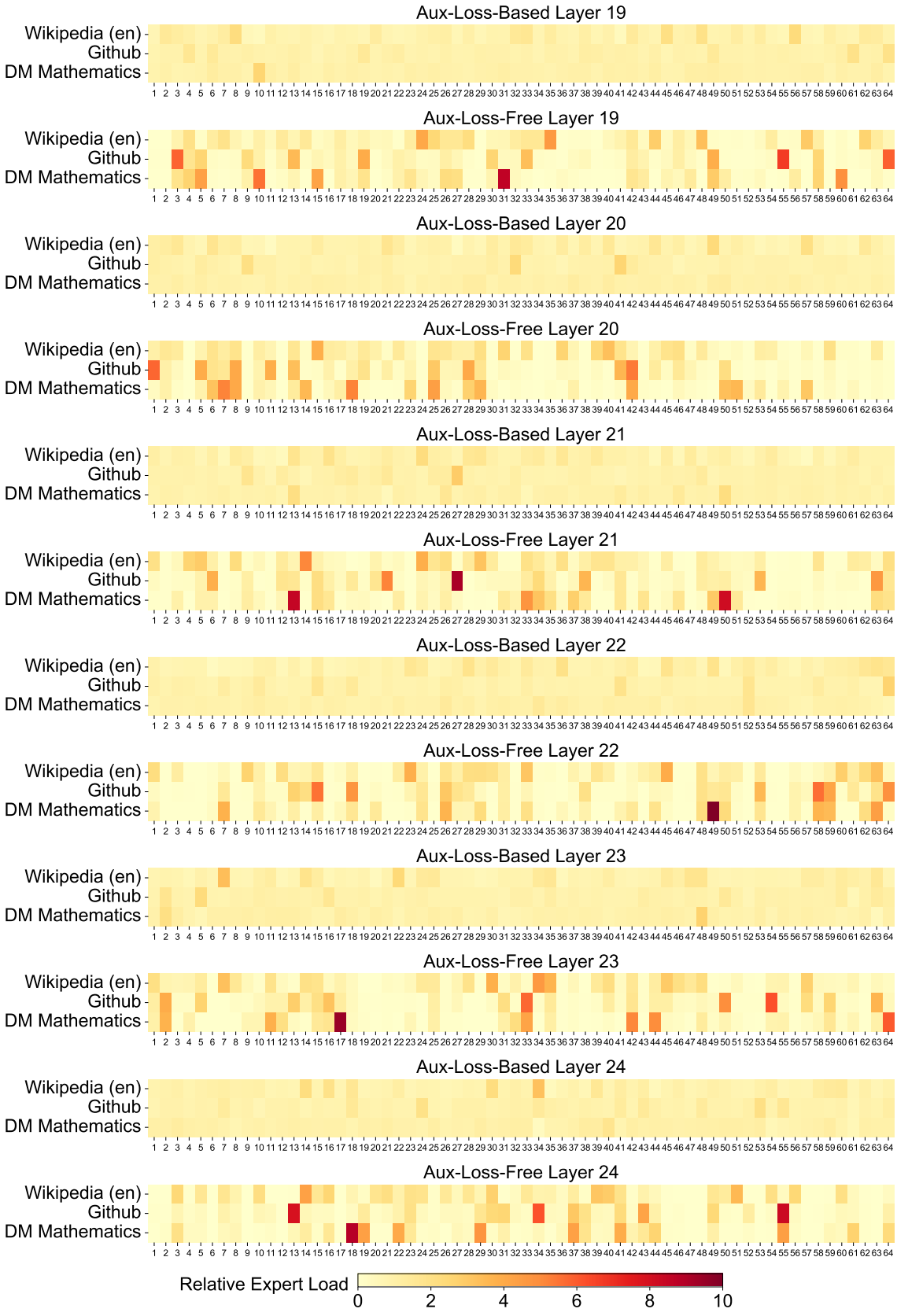
(b) Layers 7-13



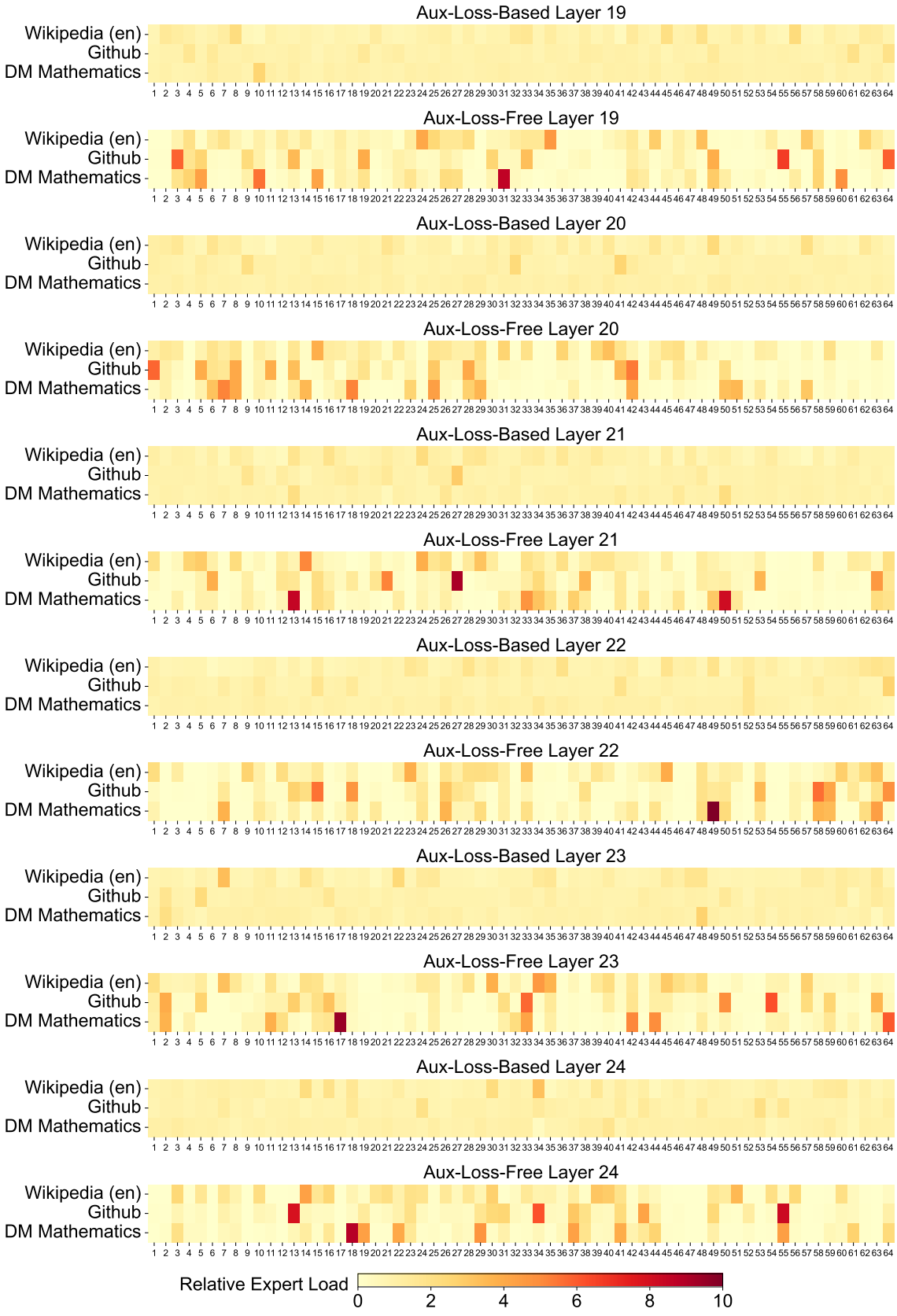
(c) Layers 13-19



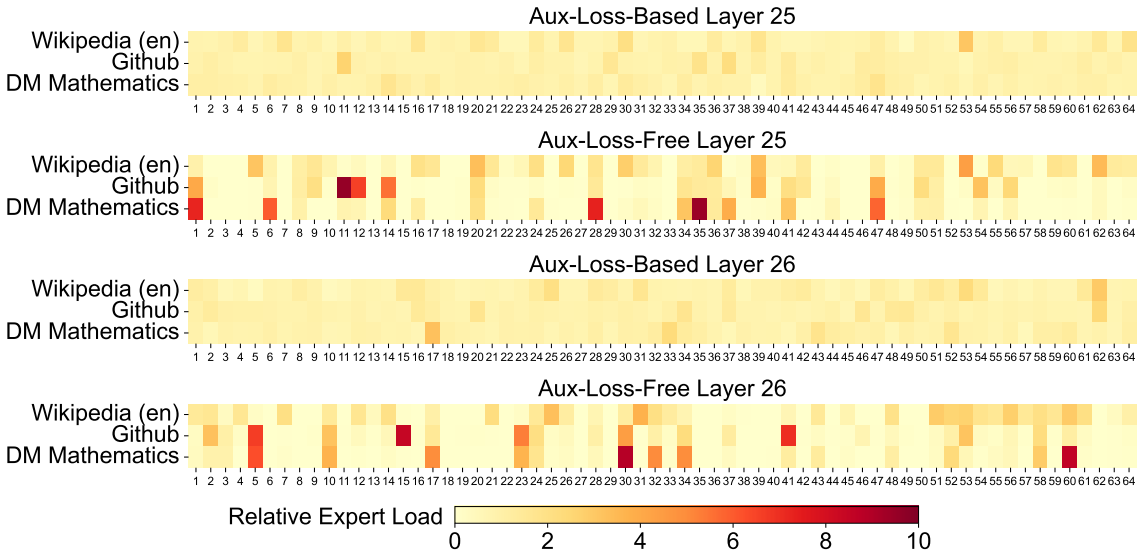
(c) Layers 13-19



(d) Layers 19-25

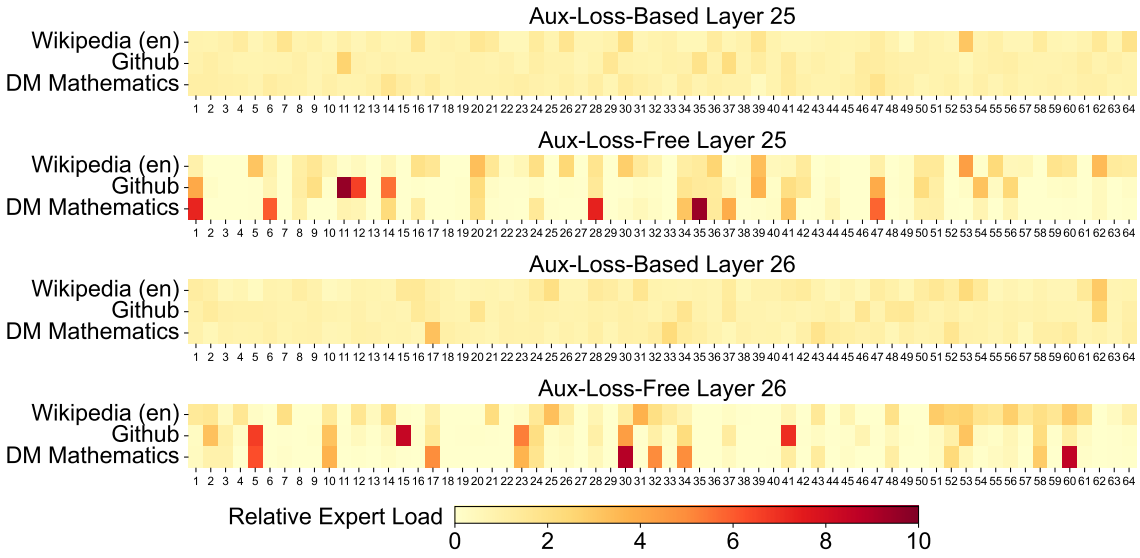


(d) Layers 19-25



(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测试集的三个领域中，辅助无损模型和基于辅助损失模型的专家负载。辅助无损模型显示出比基于辅助损失模型更大的专家专业化模式。相对专家负载表示实际专家负载与理论平衡专家负载之间的比率。