DeepSeek-V3 技术报告

January 5, 2025

DeepSeek-AI
research@deepseek.com

Abstract

我们介绍了 DeepSeek-V3，这是一个强大的专家混合 (MoE) 语言模型，总参数量为 671B，每 token 激活 37B 参数。为了实现高效的推理和成本效益的训练，DeepSeek-V3 采用了多头潜在注意力 (MLA) 和 DeepSeekMoE 架构，这些架构在 DeepSeek-V2 中得到了彻底验证。此外，DeepSeek-V3 首次采用无辅助损失策略进行负载均衡，并设定多 tokens 预测训练目标以实现更强的性能。我们在 14.8 万亿个多样化和高质量的 tokens 上预训练了 DeepSeek-V3，随后进行监督微调和强化学习阶段以充分发挥其能力。全面的评估表明，DeepSeek-V3 在性能上超越了其他开源模型，并且达到了与领先闭源模型相当的性能。尽管性能卓越，DeepSeek-V3 的完整训练仅需 2.788M H800 GPU 小时。此外，其训练过程非常稳定。在整个训练过程中，我们没有遇到任何不可恢复的损失峰值，也没有进行任何回滚。模型检查点可在 https://github.com/deepseek-ai/DeepSeek-V3 中获得。

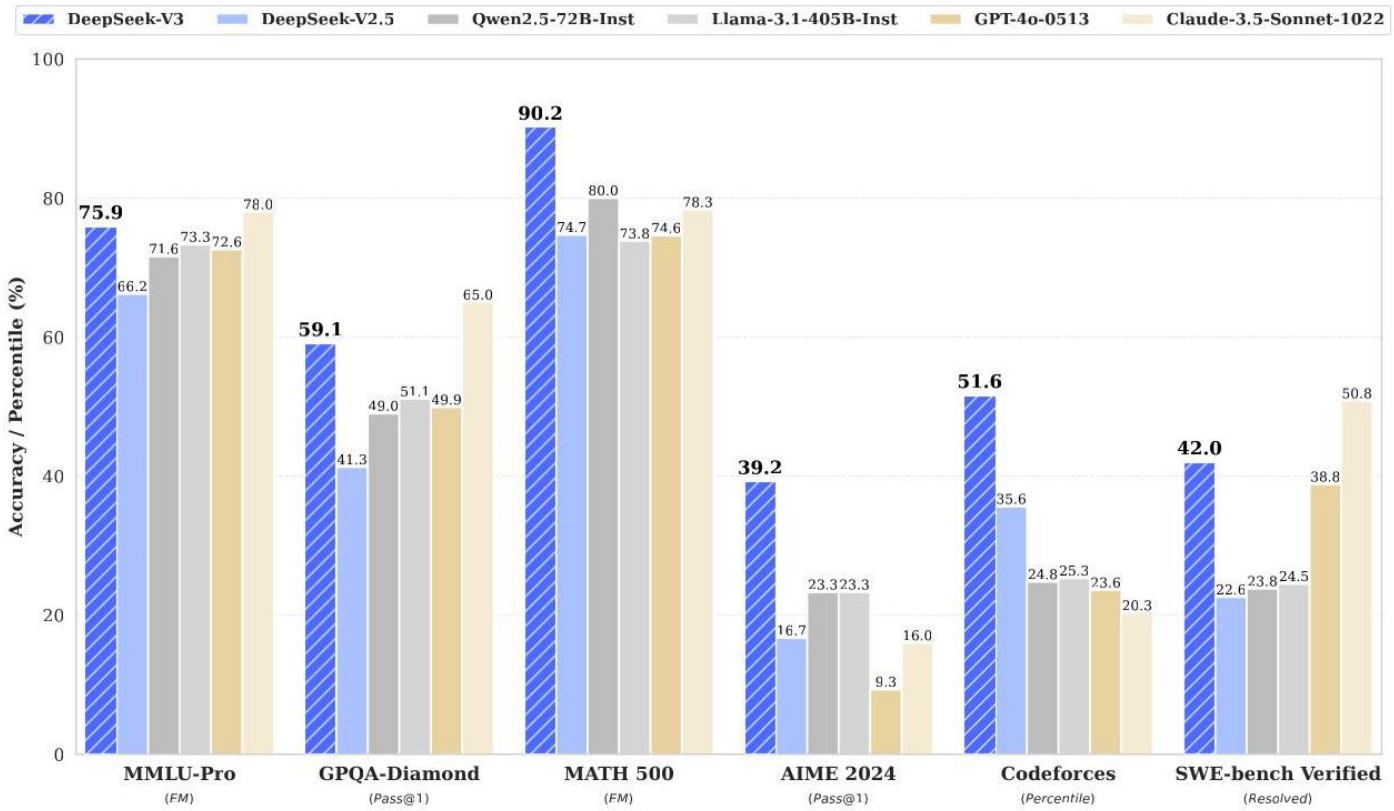


图 1 | DeepSeek-V3 及其同类模型的基准性能。

Contents

1 Introduction 引言

近年来，大型语言模型 (LLMs) 经历了快速的迭代和进化 (Anthropic, 2024; Google, 2024; OpenAI, 2024a)，逐步缩小了与通用人工智能 (AGI) 的差距。除了闭源、开源模型，包括 DeepSeek 系列 (DeepSeek-AI, 2024a|b|c; Guo 等人, 2024)，LLaMA 系列 (AI@Meta, 2024a b, Touvron 等人, 2023a b)，Qwen 系列 (Qwen, 2023, 2024a b)，以及 Mistral 系列 (Jiang 等人, 2023; Mistral, 2024)，也在取得显著进展，努力缩小与闭源模型的差距。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，并引入了 DeepSeek-V3，这是一个拥有 671B 参数的大型专家混合 (MoE) 模型，其中每 token 激活了 37B 参数。

以前瞻性的视角，我们始终致力于实现强大的模型性能和经济的成本。因此，在架构方面，DeepSeek-V3 仍然采用多头潜在注意力 (MLA)(DeepSeek-AI，2024c) 进行高效的推理，并采用 DeepSeekMoE(Dai 等人，2024) 进行成本效益的训练。这两种架构已经在 DeepSeek-V2(DeepSeek-AI，2024c) 中得到了验证，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。除了基本架构之外，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 首次采用无辅助损失策略 (Wang 等人，2024a) 进行负载均衡，旨在最小化为鼓励负载均衡而对模型性能产生的不利影响。其次，DeepSeek-V3 采用多 tokens 预测训练目标，我们观察到这可以提高在评估基准上的整体性能。

为了实现高效的训练，我们支持 FP8 混合精度训练，并对训练框架进行了全面优化。低精度训练作为一种高效的训练解决方案正在兴起 (Dettmers 等人，2022；Kalamkar 等人，2019；Narang 等人，2017；Peng 等人，2023b)，其发展与硬件能力的进步紧密相关 (Luo 等人，2024；Micikevicius 等人，2022；Rouhani 等人，2023a)。在这项工作中，我们引入了一个 FP8 混合精度训练框架，并首次验证了其在极大规模模型上的有效性。通过支持 FP8 计算和存储，我们实现了加速训练和减少 GPU 内存使用。对于训练框架，我们设计了 DualPipe 算法以实现高效的流水线并行，该算法具有较少的流水线气泡，并通过计算与通信的重叠隐藏了训练期间的大部分通信。这种重叠确保了，随着模型规模的进一步扩大，只要我们保持计算与通信的比例恒定，我们仍然可以在节点之间使用细粒度的专家，同时实现几乎为零的全对全通信开销。此外，我们还开发了高效的跨节点全对全通信内核，以充分利用 InfiniBand (IB) 和 NVLink 带宽。此外，我们还精心优化了内存占用，使得在不使用昂贵的张量并行的情况下也能训练 DeepSeek-V3。结合这些努力，我们实现了高训练效率。

在预训练过程中，我们在 14.8T 高质量和多样化的 token 上训练了 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 32 K ，在第二阶段，进一步扩展到 128 K 。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)，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 系列模型中提炼出推理能力，同时仔细保持模型准确性和生成长度之间的平衡。

训练成本	预训练	上下文扩展	后训练	总计
以 H800 GPU 小时数	2664K	119K	5K	2788K
以美元计	$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 GPU 小时 2 美元。

我们在一系列全面的基准测试上评估了 DeepSeek-V3。尽管其训练成本较低，但全面的评估显示，DeepSeek-V3-Base 已经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性基准测试中实现了与领先闭源模型 (包括 GPT-4o 和 Claude-3.5-Sonnet) 相当的性能。

最后，我们再次强调 DeepSeek-V3 的经济训练成本，这些成本总结在表 1 中，通过我们优化的算法、框架和硬件协同设计实现。在预训练阶段，每训练一万亿个标记需要 180K H800 GPU 小时，即在我们拥有 2048 个 H800 GPU 的集群上只需 3.7 天。因此，我们的预训练阶段在不到两个月内完成，耗时 2664K GPU 小时。结合上下文长度扩展所需的 119K GPU 小时和后训练所需的 5K GPU 小时，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 557.6 万美元。请注意，上述成本仅包括 DeepSeek-V3 的正式训练成本，不包括架构、算法或数据的前期研究和消融实验相关的成本。

我们的主要贡献包括:

1.1 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，该策略最大限度地减少了由于鼓励负载均衡而产生的性能下降。
- 我们研究了多标记预测 (MTP) 目标，并证明了其对模型性能的益处。它也可以用于推测性解码以加速推理。

1.2 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、框架和硬件的协同设计，我们克服了跨节点 MoE 训练中的通信瓶颈，实现了接近完全的计算-通信重叠。这显著提高了我们的训练效率，降低了训练成本，使我们能够在不增加额外开销的情况下进一步扩展模型规模。
- 仅以 2.664M H800 GPU 小时的经济成本，我们在 14.8T 个标记上完成了 DeepSeek-V3 的预训练，产生了当前最强的开源基础模型。预训练之后的后续训练阶段仅需 0.1M GPU 小时。

1.3 后训练: 从 DeepSeek-R1 中的知识蒸馏

- 我们引入了一种创新的方法，从长链思维 (CoT) 模型，特别是 DeepSeek R1 系列模型之一，将推理能力蒸馏到标准的大型语言模型中，特别是 DeepSeek-V3。我们的管道巧妙地将 R1 的验证和反思模式融入 DeepSeek-V3 中，并显著提高了其推理性能。同时，我们还控制了 DeepSeek-V3 的输出风格和长度。

1.4 核心评估结果摘要

- 知识:(1) 在 MMLU、MMLU-Pro 和 GPQA 等教育基准测试中，DeepSeek-V3 超越了所有其他开源模型，分别在 MMLU 上达到 88.5，在 MMLU-Pro 上达到 75.9，在 GPQA 上达到 59.1。其性能与 GPT-4o 和 Claude-Sonnet-3.5 等领先的闭源模型相当，缩小了开源和闭源模型在这一领域的差距。(2) 对于事实基准测试，DeepSeek-V3 在 SimpleQA 和中文 SimpleQA 上表现出优于其他开源模型的性能。虽然在英文事实知识 (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 在所有非长链推理开源和闭源模型中，于数学相关基准测试中达到最先进的性能。值得注意的是，它甚至在特定的基准测试中超过了 o1-preview，例如 MATH-500，展示了其强大的数学推理能力。(2) 在与编码相关的任务上，DeepSeek-V3 成为 LiveCodeBench 等编码竞赛基准测试中的最佳模型，巩固了其在该领域的领先地位。对于工程相关任务，尽管 DeepSeek-V3 的表现略低于 Claude-Sonnet-3.5，但它仍然显著超越了所有其他模型，展示了其在各种技术基准测试中的竞争力。

在本文的其余部分，我们首先详细介绍了我们的 DeepSeek-V3 模型架构 (第 2 节)。随后，我们介绍了我们的基础设施，包括我们的计算集群、训练框架、FP8 训练的支持、推理部署策略以及对未来硬件设计的建议。接下来，我们描述了预训练过程，包括训练数据的构建、超参数设置、长上下文扩展技术、相关的评估以及一些讨论 (第 4 节)。之后，我们讨论了我们在训练后的努力，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，我们总结了这项工作，讨论了 DeepSeek-V3 的现有局限性，并提出了未来研究的潜在方向 (第 6 节)。

2 Architecture 架构

我们首先介绍了 DeepSeek-V3 的基本架构，该架构具有 Multi-head Latent Attention (MLA) (DeepSeek-AI, 2024c) 以实现高效的推理和 DeepSeekMoE (Dai et al., 2024) 以实现经济的训练。然后，我们提出了一种 Multi-Token Prediction (MTP) 训练目标，我们观察到这可以提高在评估基准上的整体性能。对于未明确提及的其他次要细节，DeepSeek-V3 遵循 DeepSeek-V2 (DeepSeek-AI, 2024c) 的设置。

2.1 Basic Architecture 基本架构

DeepSeek-V3 的基本架构仍然在 Transformer (Vaswani et al., 2017) 框架内。为了实现高效的推理和经济的训练，DeepSeek-V3 也采用了 MLA 和 DeepSeekMoE，这些已经在 DeepSeek-V2 中得到了充分验证。与 DeepSeek-V2 相比，一个例外是我们还为 DeepSeekMoE 引入了一种无辅助损失的负载均衡策略 (Wang et al. 2024a)，以减轻为确保负载均衡而付出的努力所导致的性能下降。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。

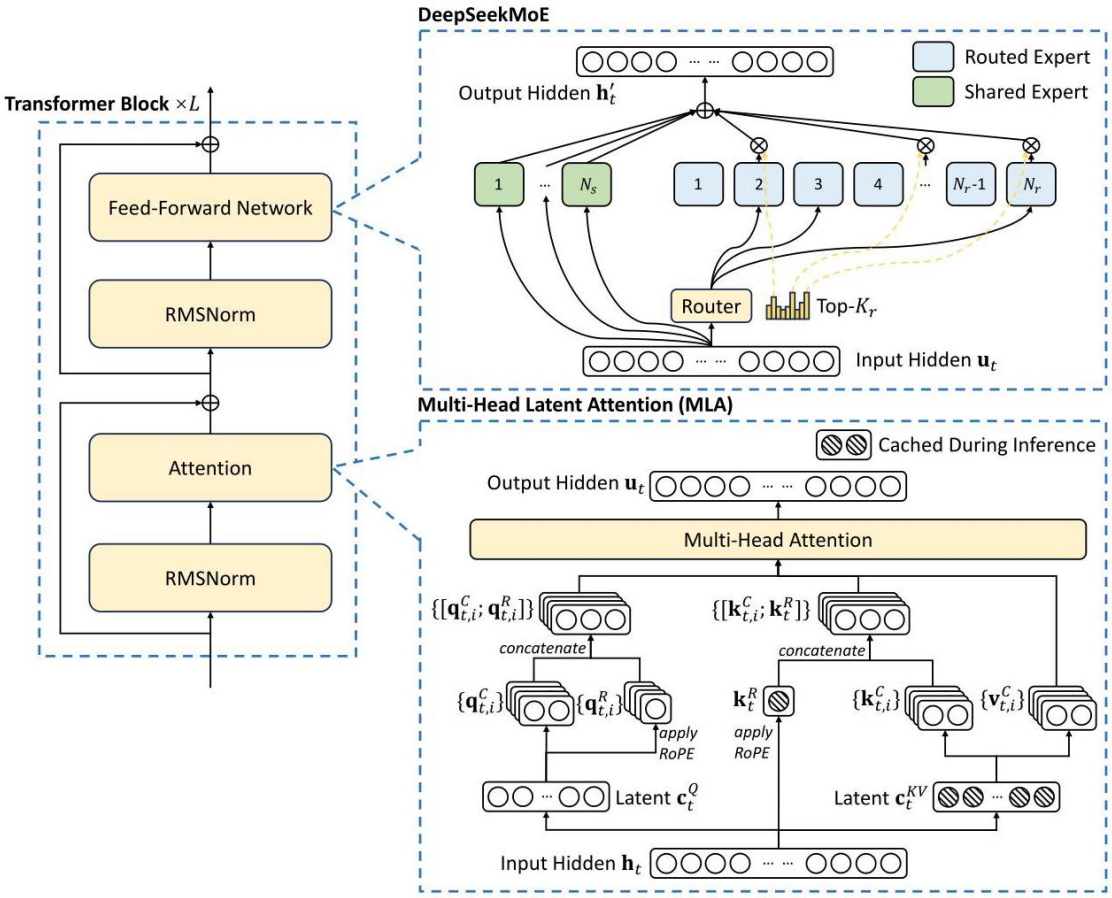


图 2 | DeepSeek-V3 的基本架构示意图。继 DeepSeek-V2 之后，我们采用 MLA 和 DeepSeekMoE 以实现高效的推理和经济的训练。

2.1.1 Multi-Head Latent Attention

对于注意力机制，DeepSeek-V3 采用了 MLA 架构。令 d 表示嵌入维度，n_h 表示注意力头的数量，d_h 表示每个头的维度，$\mathrm{h}_t \in \mathbb{R}^d$ 表示在给定注意力层中第 t 个标记的注意力输入。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 Key-Value (KV) 缓存:

$$\mathrm{c}_t^{KV} = W^{DKV}\mathrm{h}_t \tag{1}$$

$$\left[\mathrm{k}_{t,1}^C; \mathrm{k}_{t,2}^C; \ldots; \mathrm{k}_{t,n_h}^C\right] = \mathrm{k}_t^C = W^{UK}\mathrm{c}_t^{KV}, \tag{2}$$

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

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

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

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；$W^{DKV} \in \mathbb{R}^{d_c \times d}$ 表示降维矩阵；$W^{UK}, W^{UV} \in \mathbb{R}^{d_h n_h \times d_c}$ 分别表示键和值的升维矩阵；$W^{KR} \in \mathbb{R}^{d_h^R \times d}$ 是用于生成携带 Rotary Positional Embedding (RoPE) 的解耦键的矩阵 (Su et al., 2024)；$\mathrm{RoPE}(\cdot)$ 表示应用 RoPE 矩阵的操作；$[\cdot;\cdot]$ 表示连接操作。注意，对于 MLA，仅需在生成过程中缓存蓝色框中的向量 (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 et al., 2017) 相当的性能。

对于注意力查询，我们也进行了低秩压缩，这可以减少训练过程中的激活内存:

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

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

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

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

其中 $\mathrm{c}_t^Q\in\mathbb{R}^{d_c'}$ 是查询的压缩潜在向量；$d_c'\,(\ll d_hn_h)$ 表示查询压缩维度；$W^{DQ}\in\mathbb{R}^{d_c'\times d},W^{UQ}\in\mathbb{R}^{d_hn_h\times d_c'}$ 分别表示查询的降维和升维矩阵；$W^{QR}\in\mathbb{R}^{d_h^Rn_h\times d_c'}$ 是用于生成携带 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 $\left(\mathrm{v}_{j,i}^C\right)$ 结合生成最终的注意力输出 u_t ：

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

$$\mathrm{u}_t=W^O\left[\mathrm{o}_{t,1};\mathrm{o}_{t,2};\ldots;\mathrm{o}_{t,n_h}\right], \tag{11}$$

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

2.1.2 DeepSeekMoE with Auxiliary-Loss-Free Load Balancing **无辅助损失负载均衡：** DeepSeekMoE

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

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

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

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

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

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；$s_{i,t}$ 是标记到专家的亲和力；e_i 是第 i 个路由专家的中心向量；$\mathrm{Topk}\,(\cdot,K)$ 表示由为第 t 个标记和所有路由专家计算的亲和力分数中的 K 个最高分数组成的集合。与 DeepSeek-V2 稍有不同，DeepSeek-V3 使用 sigmoid 函数计算亲和力分数，并对所有选定的亲和力分数进行归一化以产生门控值。

无辅助损失的负载均衡。对于 MoE 模型而言，不平衡的专家负载将导致路由崩溃 (Shazeer 等人，2017 年) 并降低专家并行场景下的计算效率。传统解决方案通常依赖辅助损失 (Fedus 等人，2021 年；Lepikhin 等人，2021 年) 来避免负载不平衡。然而，过大的辅助损失会损害模型性能 (Wang 等人，2024a)。为了在负载均衡和模型性能之间实现更好的权衡，我们开创了一种无辅助损失的负载均衡策略 (Wang 等人，2024a) 以确保负载均衡。具体来说，我们为每个专家引入一个偏差项 b_i 并将其添加到相应的亲和力分数 $s_{i,t}$ 中以确定 top-K 路由:

$$g_{i,t}'=\begin{cases}s_{i,t}, & s_{i,t}+b_i\in\mathrm{Topk}\left(\left\{s_{j,t}+b_j\mid 1\le j\le N_r\right\},K_r\right),\\ 0, & \text{otherwise.}\end{cases} \tag{16}$$

请注意，偏差项仅用于路由。门控值，即将与 FFN 输出相乘的值，仍然从原始亲和力分数 $s_{i,t}$ 中得出。在训练过程中，我们持续监控每个训练步骤的整个批次的专家负载。在每一步结束时，如果其对应的专家负载过重，我们将减少偏差项 γ ；如果其对应的专家负载不足，我们将增加偏差项 γ ，其中 γ 是一个称为偏差更新速度的超参数。通过动态调整，DeepSeek-V3 在训练过程中保持了平衡的专家负载，并且比通过纯辅助损失鼓励负载均衡的模型表现更好。

互补的序列级辅助损失。尽管 DeepSeek-V3 主要依赖无辅助损失策略来实现负载均衡，为了防止任何单个序列内的极端不平衡，我们还采用了互补的序列级平衡损失:

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

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

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

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

其中平衡因子 α 是一个超参数，在 DeepSeek-V3 中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均衡。

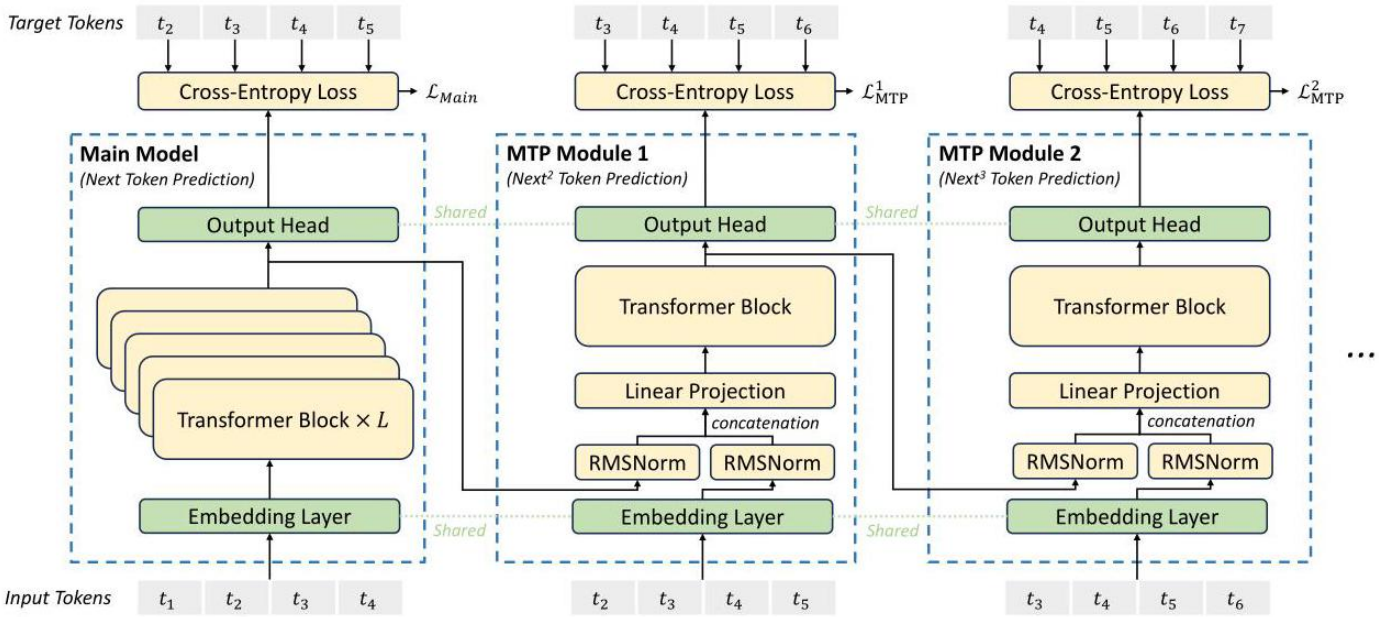


图 3 | 我们的多标记预测 (MTP) 实现示意图。我们在每个深度为每个标记的预测保留完整的因果链。

节点限制路由。类似于 DeepSeek-V2 使用的设备限制路由，DeepSeek-V3 也使用一种受限路由机制来限制训练期间的通信成本。简而言之，我们确保每个标记最多会被发送到 M 个节点，这些节点是根据每个节点上分布的专家的最高 $\frac{K_r}{M}$ 个亲和力分数的总和来选择的。在这一约束下，我们的 MoE 训练框架可以几乎实现完全的计算-通信重叠。

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

2.2 Multi-Token Prediction 多标记预测

受 Gloeckle 等 (2024) 的启发，我们对 DeepSeek-V3 设置了多 tokens 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 tokens。一方面，MTP 目标密集了训练信号，可能提高数据效率。另一方面，MTP 可能使模型能够预规划其表示，以更好地预测未来的 tokens。图 3 展示了我们的 MTP 实现。与 Gloeckle 等 (2024) 不同，后者使用独立的输出头并行预测 D 个额外的 tokens，我们顺序预测额外的 tokens，并在每个预测深度保持完整的因果链。我们在此部分介绍了 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 tokens。第 k 个 MTP 模块包括一个共享嵌入层 $\text{Emb}(\cdot)$、一个共享输出头 $\text{OutHead}(\cdot)$、一个 Transformer 块 $\text{TRM}_k(\cdot)$ 和一个投影矩阵 $M_k \in \mathbb{R}^{d \times 2d}$。对于第 i 个输入 tokens t_i，在第 k 个预测深度，我们首先将第 i 个 tokens 在 (k-1) 深度的表示 $\mathrm{h}_i^{k-1} \in \mathbb{R}^d$ 和第 $(i+k)$ 个 tokens 的嵌入 $\text{Emb}(t_{i+k}) \in \mathbb{R}^d$ 通过线性投影结合起来:

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

其中 $[\cdot;\cdot]$ 表示连接。特别地，当 $k=1, \mathrm{h}_i^{k-1}$ 指主模型给出的表示时。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 个深度的 Transformer 块的输入，以生成当前深度的输出表示 h_i^k：

$$\mathrm{h}_{1:T-k}^{k}=\mathrm{TRM}_{k}\left(\mathrm{h}_{1:T-k}^{\prime k}\right), \tag{22}$$

其中 T 表示输入序列的长度，$i:j$ 表示切片操作 (包括左右边界)。最后，以 h_{i}^{k} 作为输入，共享输出头将计算第 k 个附加预测标记 $P_{i+1+k}^{k} \in \mathbb{R}^{V}$ 的概率分布，其中 V 是词汇表的大小:

$$P_{i+k+1}^{k}=\mathrm{OutHead}\left(\mathrm{h}_{i}^{k}\right). \tag{23}$$

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

MTP 训练目标。对于每个预测深度，我们计算一个交叉熵损失 $\mathcal{L}_{\mathrm{MTP}}^{k}$ ：

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

其中 T 表示输入序列的长度，t_i 表示第 i 个位置的真实标记，$P_{i}^{k}\left[t_{i}\right]$ 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，并乘以一个加权因子 λ 以获得整体 MTP 损失 $\mathcal{L}_{\mathrm{MTP}}$ ，这作为 DeepSeek-V3 的附加训练目标:

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

MTP 在推理中的应用。我们的 MTP 策略主要旨在提高主模型的性能，因此在推理期间，我们可以直接丢弃 MTP 模块，主模型可以独立正常工作。此外，我们还可以重新利用这些 MTP 模块进行推测性解码以进一步提高生成延迟。

3 Infrastructures 基础设施

3.1 Compute Clusters 计算集群

DeepSeek-V3 在配备有 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 GPU，这些 GPU 通过 NVLink 和 NVSwitch 在节点内部连接。在不同节点之间，使用 InfiniBand (IB) 互连来促进通信。

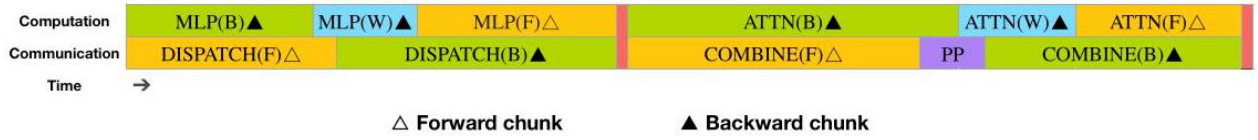


图 4 | 一对单独的前向和后向块的重叠策略 (变压器块的边界未对齐)。橙色表示前向，绿色表示“输入的后向”，蓝色表示“权重的后向”，紫色表示 PP 通信，红色表示障碍。全对全通信和 PP 通信都可以完全隐藏。

3.2 Training Framework 训练框架

DeepSeek-V3 的训练由 HAI-LLM 框架支持，这是一个由我们的工程师从头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 Pipeline 并行 (PP)(Qi 等人，2023a)，跨越 8 个节点的 64 路专家并行 (EP)(Lepikhin 等人，2021)，以及 ZeRO-1 数据并行 (DP)(Rajb-handari 等人，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 有更少的管道气泡。更重要的是，它在前向和后向过程之间重叠计算和通信阶段，从而解决了由跨节点专家并行引入的大量通信开销的挑战。其次，我们开发了高效的跨节点全对全通信内核，以充分利用 IB 和 NVLink 带宽，并节省用于通信的流式多处理器 (SM)。最后，我们仔细优化了训练期间的内存占用，从而使我们能够在不使用昂贵的 Tensor 并行 (TP) 的情况下训练 DeepSeek-V3。

3.2.1 DualPipe and Computation-Communication Overlap 和计算-通信重叠

对于 DeepSeek-V3，跨节点专家并行引入的通信开销导致了大约 1:1 的低效计算与通信比率。为了解决这一挑战，我们设计了一种创新的流水线并行算法，称为 DualPipe，该算法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，而且还减少了流水线气泡。

DualPipe 的核心思想是在一对单独的前向和后向块内重叠计算和通信。具体来说，我们将每个块分为四个部分: 注意力、all-to-all 分发、MLP 和 all-to-all 组合。特别地，对于一个后向块，注意力和 MLP 都进一步分为两部分，输入的后

向和权重的后向，类似于 ZeroBubble (Qi et al., 2023b)。此外，我们有一个 PP 通信组件。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 all-to-all 和 PP 通信在执行期间完全被隐藏。鉴于这种高效的重叠策略，完整的 DualPipe 调度如图 5 所示。它采用双向流水线调度，同时从流水线的两端输入微批次，并且通信的很大一部分可以完全重叠。这种重叠还确保了，随着模型进一步扩展，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专家，同时实现接近零的 all-to-all 通信开销。

图 5 | 8 个 PP 级别和两个方向上的 20 个微批次的 DualPipe 调度示例。反向方向的微批次与正向方向的微批次对称，为了简化说明，我们省略了它们的批次 ID。由共享黑色边框包围的两个单元格具有相互重叠的计算和通信。

方法	泡泡	参数	激活
1F1B	$(PP-1)(F+B)$	1×	PP
ZB1P	$(PP-1)(F+B-2W)$	1×	PP
双管道 (我们的方法)	$\left(\frac{PP}{2}-1\right)(F\&B+B-3W)$	2×	$PP+1$

表 2 | 不同流水线并行方法之间的流水线气泡和内存使用比较。F 表示前向块的执行时间，B 表示完整后向块的执行时间，W 表示" 权重反向" 块的执行时间，$F\&B$ 表示两个相互重叠的前向和后向块的执行时间。

此外，即使在没有大量通信负担的更一般场景中，Du-alPipe 仍表现出效率优势。在表 2 中，我们总结了不同 PP 方法的流水线气泡和内存使用情况。如表所示，与 ZB1P (Qi et al. 2023b) 和 1F1B (Harlap et al. 2018) 相比，DualPipe 显著减少了流水线气泡，同时仅将峰值激活内存增加了 $\frac{1}{PP}$ 倍。尽管 DualPipe 需要保留两份模型参数的副本，但这不会显著增加内存消耗，因为我们训练时使用了较大的 EP 大小。与 Chimera (Li 和 Hoefler, 2021) 相比，DualPipe 只要求流水线阶段和微批次可被 2 整除，而不要求微批次可被流水线阶段整除。此外，对于 DualPipe，气泡和激活内存不会随着微批次数量的增加而增加。

3.2.2 Efficient Implementation of Cross-Node All-to-All Communication **跨节点** All-to-All **通信实现**

为了确保 DualPipe 具有足够的计算性能，我们定制了高效的跨节点全对全通信内核 (包括调度和组合)，以减少专门用于通信的 SM 数量。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 完全互连，而节点内的通信则通过 NVLink 处理。NVLink 提供 160 GB/s 的带宽，大约是 IB(50 GB/s) 的 3.2 倍。为了有效利用 IB 和 NVLink 的不同带宽，我们将每个 token 的调度限制在最多 4 个节点，从而减少 IB 流量。对于每个 token，一旦其路由决策确定，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，我们将尽力确保它通过 NVLink 立即转发到承载其目标专家的特定 GPU，而不被随后到达的 token 阻塞。这样，IB 和 NVLink 的通信可以完全重叠，每个 token 可以高效地选择每个节点平均 3.2 个专家，而不会因 NVLink 引起额外开销。这意味着，尽管 DeepSeek-V3 实际上只选择了 8 个路由专家，但它可以将这个数量扩展到最多 13 个专家 (4 个节点 × 3.2 个专家/节点)，同时保持相同的通信成本。总体而言，在这种通信策略下，仅需 20 个 SM 即可充分利用 IB 和 NVLink 的带宽。

具体来说，我们采用了 warp 专业化技术 (Bauer 等人，2014 年)，并将 20 个 SM 分成 10 个通信通道。在分发过程中，(1)IB 发送，(2)IB 到 NVLink 转发，以及 (3)NVLink 接收分别由各自的 warp 处理。分配给每个通信任务的 warp 数量会根据所有 SM 的实际工作负载动态调整。同样，在组合过程中，(1)NVLink 发送，(2)NVLink 到 IB 转发和累积，以及 (3)IB 接收和累积也由动态调整的 warp 处理。此外，分发和组合内核与计算流重叠，因此我们也考虑了它们对其他 SM 计算内核的影响。具体来说，我们使用了定制的 PTX(并行线程执行) 指令，并自动调整通信块大小，这显著减少了 L2 缓存的使用，并减少了对其他 SM 的干扰。

3.2.3 Extremely Memory Saving with Minimal Overhead **极其节省内存且开销极小**

为了减少训练期间的内存占用，我们采用了以下技术。

RMSNorm 和 MLA 上投影的重新计算。我们在反向传播过程中重新计算所有 RMSNorm 操作和 MLA 上投影，从而消除了持久存储其输出激活的需要。通过引入轻微的开销，该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，我们保存模型参数的指数移动平均 (EMA)，以便在学习率衰减后早期估计模型性能。EMA 参数存储在 CPU 内存中，并在每次训练步骤后异步更新。这种方法允许我们在不增加额外内存或时间开销的情况下维持 EMA 参数。

多标记预测的共享嵌入和输出头。采用 DualPipe 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) 部署在相同的 PP rank 上。这种安排使得 MTP 模块和主模型之间可以共享嵌入和输出头的参数和梯度。这种物理共享机制进一步提高了我们的内存效率。

3.3 FP8-Training FP8 **训练**

受最近低精度训练进展的启发 (Dettmers 等人，2022；Noune 等人，2022；Peng 等人，2023b)，我们提出了一种利用 FP8 数据格式进行 DeepSeek-V3 训练的细粒度混合精度框架。尽管低精度训练前景广阔，但它往往受限于激活值、权重和梯度中的异常值 (Fishman 等人，2024；He 等人，Sun 等人，2024)。尽管在推理量化方面已经取得了显著进展 (Frantar 等人，2022；Xiao 等人，2023)，但在大规模语言模型预训练中成功应用低精度技术的研究相对较少 (Fishman 等人，2024)。为应对这一挑战并有效扩展 FP8 格式的动态范围，我们引入了一种细粒度量化策略: 以 $1 \times N_c$ 个元素为单位的分块分组或以 $N_c \times N_c$ 个元素为单位的块分组。在我们的高精度累加过程中，相关的去量化开销得到了很大程度的缓解，这是实现准确的 FP8 通用矩阵乘法 (GEMM) 的关键方面。此外，为了进一步减少 MoE 训练中的内存和通信开销，我们将激活值以 FP8 格式缓存和分发，同时将低精度优化器状态存储在 BF16 中。我们在两个与 DeepSeek-V2-Lite 和 DeepSeek-V2 类似的模型规模上验证了所提出的 FP8 混合精度框架，训练了大约 1 万亿个标记 (详见附录 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低于 0.25%，这一水平在训练随机性可接受范围内。

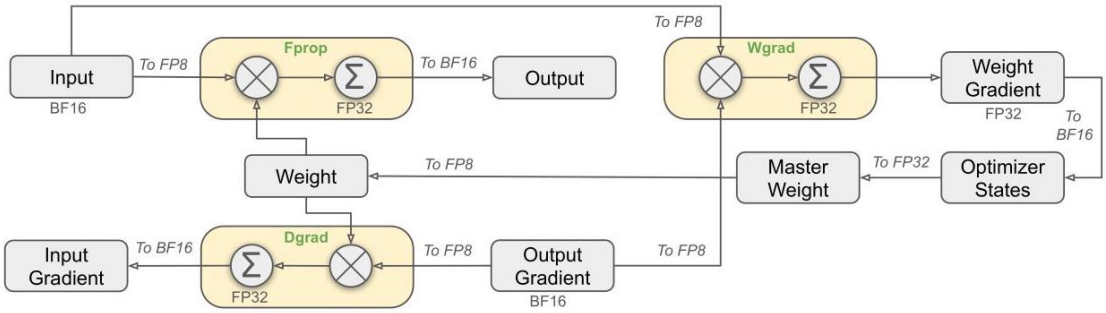


图 6 | 使用 FP8 数据格式的整体混合精度框架。为便于理解，仅展示了线性算子。

3.3.1 Mixed Precision Framework **混合精度框架**

基于广泛采用的低精度训练技术 (Kalamkar 等人，2019；Narang 等人，2017)，我们提出了一种用于 FP8 训练的混合精度框架。在这个框架中，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它们原有的数据格式中，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，以 FP8 精度实现。这些 GEMM 操作接受 FP8 张量作为输入，并以 BF16 或 FP32 生成输出。如图 6 所示，与线性操作符相关的所有三个 GEMM，即 Fprop(前向传递)、Dgrad(激活反向传递) 和 Wgrad(权重反向传递)，都在 FP8 中执行。这种设计理论上将计算速度提高了一倍，与原始的 BF16 方法相比。此外，FP8 Wgrad GEMM 允许激活以 FP8 格式存储，用于反向传递。这显著减少了内存消耗。

尽管 FP8 格式具有效率优势，但由于某些操作对低精度计算的敏感性，它们仍然需要更高的精度。此外，一些低成本操作也可以在整体训练成本几乎没有增加的情况下使用更高的精度。因此，经过仔细调查，我们为以下组件保留了原有的精度 (例如 BF16 或 FP32): 嵌入模块、输出头、MoE 门控模块、归一化操作和注意力操作。这些高精度的针对性保留确保了 DeepSeek-V3 的稳定训练动态。为了进一步保证数值稳定性，我们将主权重、权重梯度和优化器状态存储在更高的精度中。虽然这些高精度组件会带来一些内存开销，但通过在我们的分布式训练系统中跨多个 DP 级别进行高效的分片，可以将这些影响最小化。

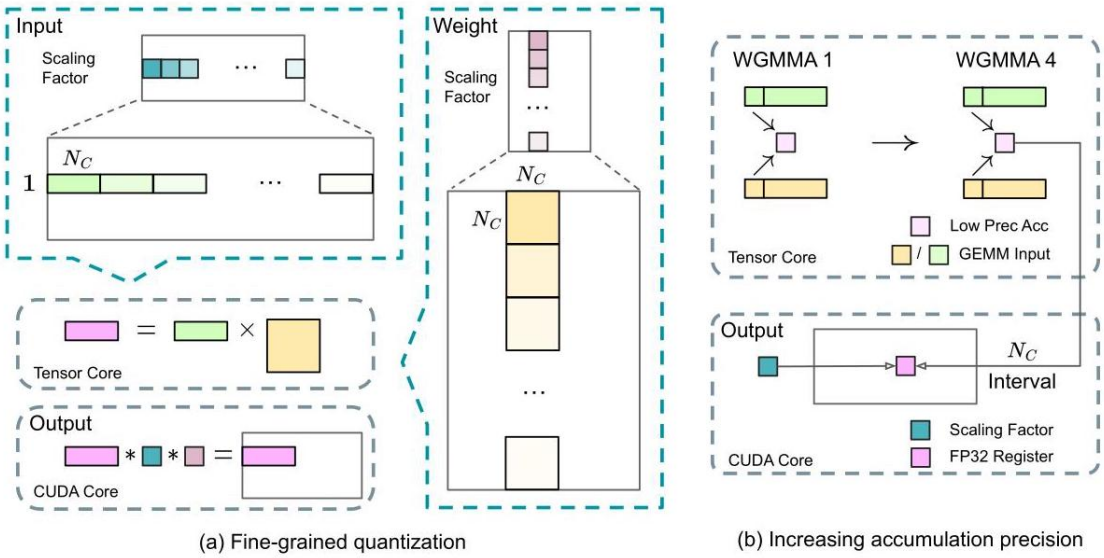


图 7 | (a) 我们提出了一种细粒度量化方法来减轻由特征异常值引起的量化误差；为了简化说明，仅展示了 Fprop。(b) 结合我们的量化策略，我们通过在 $N_C = 128$ 元素 MMA 间隔内提升到 CUDA 内核来提高 FP8 GEMM 的精度，以实现高精度累加。

3.3.2 Improved Precision from Quantization and Multiplication **量化和乘法改进的精度**

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，重点在于量化方法和乘法过程。

细粒度量化。在低精度训练框架中，由于 FP8 格式的动态范围有限 (受限于其减少的指数位数)，溢出和下溢是常见的挑战。作为标准做法，通过将输入张量的最大绝对值缩放到 FP8 的最大可表示值来对齐输入分布 (Narang 等人，2017)。这种方法使得低精度训练对激活异常值非常敏感，这可能会严重降低量化精度。为了解决这个问题，我们提出了一种细粒度量化方法，该方法在更细粒度的级别上应用缩放。如图 7 (a) 所示，(1) 对于激活，我们基于 1x128 瓷砖 (即每个标记每 128 个通道) 对元素进行分组和缩放；(2) 对于权重，我们基于 128x128 块 (即每 128 个输入通道每 128 个输出通道) 对元素进行分组和缩放。这种方法确保量化过程可以根据较小的元素组调整缩放比例，更好地适应异常值。在附录 B.2 中，我们进一步讨论了当我们将激活分组和缩放与权重量化相同的方式进行时的训练不稳定性。

我们方法的一个关键修改是在 GEMM 操作的内部维度上引入了每组缩放因子。这种功能在标准的 FP8 GEMM 中并不直接支持。然而，结合我们的精确 FP32 累加策略，它可以高效地实现。

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

提高累积精度。低精度 GEMM 操作通常会遇到下溢问题，其准确性很大程度上依赖于高精度累积，这通常在 FP32 精度下进行 (Kalamkar et al., 2019; Narang et al., 2017)。然而，我们观察到 NVIDIA H800 GPU 上的 FP8 GEMM 累积精度仅保留约 14 位，这远低于 FP32 累积精度。当内部维度 K 较大时，这个问题将更加明显 (Wortsman et al. 2023)，这在大规模模型训练中是一个典型场景，其中批量大小和模型宽度增加。以两个随机矩阵的 GEMM 操作 K = 4096 为例，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某些 FP8 框架中的默认选项 (NVIDIA, 2024b)，严重限制了训练精度。

为了解决这一问题，我们采用了将中间结果提升到 CUDA 内核以实现更高精度的策略 (Thakkar 等人，2023)。该过程在图 7(b) 中进行了说明。具体来说，在 Tensor 内核上的 MMA(矩阵乘累加) 执行过程中，中间结果使用有限的位宽进行累加。一旦达到 N_C 区间，这些部分结果将被复制到 CUDA 内核上的 FP32 寄存器中，在那里进行全精度 FP32 累加。如前所述，我们的细粒度量化沿内部维度应用每组缩放因子 K 。这些缩放因子可以在 CUDA 内核上作为去量化过程高效地进行乘法运算，而不会增加额外的计算成本。

值得注意的是，这种修改降低了单个 warpgroup 的 WGMMA(warpgroup 级矩阵乘累加) 指令的发布率。然而，在 H800 架构上，通常两个 WGMMA 可以并行存在: 当一个 warpgroup 执行提升操作时，另一个可以执行 MMA 操作。这种设计使两个操作能够重叠，保持 Tensor 内核的高利用率。根据我们的实验，设置 $N_C = 128$ 个元素，相当于 4 个 WGMMAs，代表了可以显著提高精度而不会引入显著开销的最小累加区间。

尾数超过指数。与先前工作 (NVIDIA, 2024b; Peng et al., 2023b; Sun et al., 2019b) 采用的混合 FP8 格式不同，后者在 Fprop 中使用 E4M3(4 位指数和 3 位尾数)，在 Dgrad 和 Wgrad 中使用 E5M2(5 位指数和 2 位尾数)，我们对所有张量采用 E4M3 格式以获得更高的精度。我们将其归因于我们的细粒度量化策略，即按瓷砖和块进行缩放。通过操作较小的元素组，我们的方法有效地在这些分组元素之间共享指数位，减轻了有限动态范围的影响。

在线量化。在张量级量化框架 (NVIDIA, 2024b; Peng et al., 2023b) 中采用延迟量化，该框架保留了先前迭代中最大绝对值的历史记录以推断当前值。为了确保准确的缩放因子并简化框架，我们为每个 1x128 激活瓷砖或 128x128 权重块在线计算最大绝对值。基于此，我们推导出缩放因子，然后在线将激活或权重量化为 FP8 格式。

3.3.3 Low-Precision Storage and Communication **低精度存储和通信**

结合我们的 FP8 训练框架，我们通过将缓存的激活和优化器状态压缩为低精度格式，进一步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矩，而不会导致可观察到的性能下降。然而，主权重 (由优化器存储) 和梯度 (用于批量大小累积) 仍保持在 FP32 以确保训练过程中的数值稳定性。

低精度激活。如图 6 所示，Wgrad 操作在 FP8 中执行。为了减少内存消耗，将激活缓存为 FP8 格式以供 Linear 算子的反向传播使用是一个自然的选择。然而，对于低成本高精度训练，对几个算子进行了特别考虑:

(1) 注意力算子之后 Linear 算子的输入。这些激活也用于注意力算子的反向传播，这使得它们对精度敏感。我们为这些激活采用了一种专门的 E5M6 数据格式。此外，这些激活将在反向传播中从 1x128 量化块转换为 128x1 块。为了避免引入额外的量化误差，所有缩放因子都是整数幂 2。

(2) MoE 中 SwiGLU 算子的输入。为了进一步减少内存成本，我们缓存 SwiGLU 算子的输入并在反向传播中重新计算其输出。这些激活也使用我们的细粒度量化方法存储为 FP8，以平衡内存效率和计算精度。

低精度通信。通信带宽是训练 MoE 模型的关键瓶颈。为了解决这一挑战，我们将 MoE 上投影前的激活量化为 FP8，然后应用分发组件，这与 MoE 上投影中的 FP8 前向传播兼容。像注意力算子之后 Linear 算子的输入一样，这些激活的缩放因子也是 2 的整数幂。类似的策略应用于 MoE 下投影前的激活梯度。对于前向和反向组合组件，我们保留它们为 BF16，以在训练管道的关键部分保持训练精度。

3.4 Inference and Deployment **推理与部署**

我们在 H800 集群上部署了 DeepSeek-V3，其中每个节点内的 GPU 通过 NVLink 相互连接，而集群中的所有 GPU 则通过 IB 完全互连。为了同时确保在线服务的服务级别目标 (SLO) 和高吞吐量，我们采用了一种将预填充和解码阶段分开的部署策略。

3.4.1 **预填充**

预填充阶段的最小部署单元由 4 个节点和 32 个 GPU 组成。注意力部分采用 4 路张量并行 (TP4) 与序列并行 (SP) 结合，同时采用 8 路数据并行 (DP8)。其较小的 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大小，从而提高计算效率。对于 MoE 的 all-to-all 通信，我们使用与训练中相同的方法: 首先通过 IB 在节点间传输 token，然后通过 NVLink 在节点内的 GPU 之间转发。特别地，我们为浅层中的密集 MLP 使用 1 路张量并行，以节省 TP 通信。

为了在 MoE 部分的不同专家之间实现负载均衡，我们需要确保每个 GPU 处理大约相同数量的 token。为此，我们引入了冗余专家部署策略，该策略复制高负载专家并冗余部署它们。高负载专家是基于在线部署期间收集的统计数据检测到的，并且会定期调整 (例如，每 10 分钟)。确定冗余专家集后，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，力求在不增加跨节点全对全通信开销的情况下尽可能平衡 GPU 之间的负载。对于 DeepSeek-V3 的部署，我们在预填充阶段设置了 32 个冗余专家。对于每个 GPU，除了它原本托管的 8 个专家外，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对全和 TP 通信的开销，我们同时处理两个计算工作量相似的微批次，将一个微批次的注意力和 MoE 与另一个微批次的分发和组合重叠。

最后，我们正在探索专家的动态冗余策略，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只有 9 个会被激活。在每一层的全对全操作开始之前，我们实时计算全局最优路由方案。鉴于预填充阶段涉及大量的计算，计算此路由方案的开销几乎可以忽略不计。

3.4.2 **解码**

在解码过程中，我们将共享专家视为路由选择的一个。从这个角度来看，每个标记在路由时将选择 9 个专家，其中共享专家被视为一个始终被选中的高负载专家。解码阶段的最小部署单元由 40 个节点和 320 个 GPU 组成。注意力部分采用 TP4 和 SP 结合 DP80，而 MoE 部分使用 EP320。对于 MoE 部分，每个 GPU 只托管一个专家，64 个 GPU 负责托管冗余专家和共享专家。分发和组合部分的 All-to-all 通信通过 IB 上的直接点对点传输实现低延迟。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，我们根据在线服务的统计专家负载，在一定区间内定期确定冗余专家的集合。然而，由于每个 GPU 只托管一个专家，我们不需要重新安排专家。我们还在探索解码的动态冗余策略。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，并与分发内核融合以减少开销。

此外，为了提高吞吐量并隐藏全对全通信的开销，我们还在解码阶段探索同时处理两个具有相似计算工作负载的微批次。与预填充不同，注意力在解码阶段消耗了更多的时间。因此，我们将一个微批次的注意力与另一个微批次的调度 +MoE+ 组合重叠。在解码阶段，每个专家的批处理大小相对较小 (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，内存访问的开销很小，因此使用较少的 SM 不会显著影响整体性能。因此，为了避免影响注意力部分的计算速度，我们可以只分配一小部分 SM 用于调度 +MoE+ 组合。

3.5 Suggestions on Hardware Design **硬件设计建议**

基于我们对全对全通信和 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 **通信硬件**

在 DeepSeek-V3 中，我们实现了计算和通信之间的重叠，以隐藏计算期间的通信延迟。这显著减少了对通信带宽的依赖，相比串行计算和通信。然而，当前的通信实现依赖于昂贵的 SM(例如，我们为这个目的分配了 H800 GPU 可用的 132 个 SM 中的 20 个)，这将限制计算吞吐量。此外，使用 SM 进行通信导致显著的效率低下，因为张量核心完全未被充分利用。

目前，SM 主要执行以下任务以实现全对全通信:

- 在 IB(InfiniBand) 和 NVLink 域之间转发数据，同时将同一节点内多个 GPU 的目标 IB 流量从单个 GPU 聚合。
- 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。
- 执行所有节点间的组合操作的 reduce 操作。
- 在通过 IB 和 NVLink 域向多个专家传输分块数据期间管理细粒度的内存布局。

我们希望看到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 中卸载出来，作为 GPU 协处理器或像 NVIDIA SHARP Graham et al. (2016) 这样的网络协处理器。此外，为了减少应用编程的复杂性，我们希望这种硬件能够从计算单元的角度统一 IB(扩展) 和 NVLink(向上扩展) 网络。借助这种统一的接口，计算单元可以通过提交基于简单原语的通信请求，轻松地在整个 IB-NVLink 统一域中完成读取、写入、多播和 reduce 等操作。

3.5.2 Compute Hardware **计算硬件**

更高的 FP8 GEMM 累加精度在 Tensor Core 中。在当前 NVIDIA Hopper 架构的 Tensor Core 实现中，FP8 GEMM(通用矩阵乘法) 使用定点累加，通过基于最大指数的右移来对齐乘积的尾数。我们的实验表明，它仅使用每个尾数乘积右移填充符号后的最高 14 位，并截断超出此范围的位。然而，例如，为了从 32 个 FP8 × FP8 乘法的累加中获得精确的 FP32 结果，至少需要 34 位精度。因此，我们建议未来的芯片设计增加 Tensor Core 中的累加精度，以支持全精度累加，或者根据训练和推理算法的精度要求选择适当的累加位宽。这种方法确保误差保持在可接受范围内，同时保持计算效率。

对 Tile-和 Block-量化提供支持。当前的 GPU 仅支持张量级量化，缺乏对我们这种细粒度量化 (如 Tile-和 Block-量化) 的原生支持。在当前的实现中，当 N_C 区间达到时，部分结果将从 Tensor Cores 复制到 CUDA cores，乘以缩放因子，并加到 CUDA cores 上的 FP32 寄存器中。尽管结合我们精确的 FP32 累加策略显著减轻了去量化开销，但 Tensor Cores 和 CUDA cores 之间的频繁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允许 Tensor Cores 接收缩放因子并实现带有组缩放的 MMA 来支持细粒度量化。这样，整个部分和累加和去量化可以直接在 Tensor Cores 内完成，直到生成最终结果，从而避免频繁的数据移动。

对在线量化提供支持。当前的实现难以有效地支持在线量化，尽管我们的研究证明了其有效性。在现有过程中，我们需要从 HBM(高带宽内存) 读取 128 个 BF16 激活值 (前一计算的输出) 进行量化，然后将量化后的 FP8 值写回 HBM，以便再次读取用于 MMA。为了解决这种低效问题，我们建议未来的芯片将 FP8 转换和 TMA(张量内存加速器) 访问集成到单一融合操作中，从而在激活值从全局内存传输到共享内存时完成量化，避免频繁的内存读写。我们还建议支持一种 warp 级别的转换指令以提高速度，这进一步促进了层归一化和 FP8 转换的更好融合。或者，可以采用近内存计算方法，即将计算逻辑放置在 HBM 附近。在这种情况下，BF16 元素可以直接在从 HBM 读取到 GPU 时转换为 FP8，减少大约 50% 的片外内存访问。

对转置 GEMM 操作提供支持。当前架构使得矩阵转置与 GEMM 操作的融合变得繁琐。在我们的工作流程中，前向传递期间的激活值被量化为 1x128 FP8 块并存储。在后向传递期间，矩阵需要被读出、去量化、转置、重新量化为 128×1 块，并存储在 HBM 中。为了减少内存操作，我们建议未来的芯片能够在 MMA 操作之前直接从共享内存读取矩阵的转置，适用于训练和推理所需的精度。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 **预训练**

4.1 Data Construction **数据构建**

与 DeepSeek-V2 相比，我们通过增加数学和编程样本的比例来优化预训练语料库，同时扩展了多语言覆盖范围，超出了英语和中文。我们的数据处理管道也经过优化，以减少冗余并保持语料库的多样性。受 Ding 等人 (2024) 的启发，我们实现了文档打包方法以确保数据完整性，但在训练过程中并未采用跨样本注意力掩码。最后，DeepSeek-V3 的训练语料库由 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 Fill-in-Middle(FIM) 策略不会损害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 保持一致，我们在 DeepSeek-V3 的预训练中也采用了 FIM 策略。具体来说，我们使用前缀-后缀-中间 (PSM) 框架来构建数据，如下所示:

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

该结构在文档级别作为预打包过程的一部分应用。FIM 策略以 0.1 的比例应用，与 PSM 框架保持一致。

DeepSeek-V3 的分词器采用 Byte-level BPE(Shibata 等人，1999)，其扩展词汇表包含 128 K 个标记。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标记。然而，当模型处理没有终端换行符的多行提示时，特别是对于少量样本评估提示，这种技巧可能会引入标记边界偏差 (Lundberg，2023)。为了解决这个问题，我们在训练过程中随机拆分一定比例的这种组合标记，使模型接触到更广泛的特殊情况，从而减轻这种偏差。

4.2 Hyper-Parameters **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。所有可学习参数的初始化标准差为 0.006。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。键值压缩维度 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。除了前三层之外，我们将所有前馈网络替换为 MoE 层。每个 MoE 层包含 1 个共享专家和 256 个路由专家，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，并且每个 token 将确保发送到最多 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 之外，每个 token 将预测一个额外的 token。与 DeepSeek-V2 类似，DeepSeek-V3 在压缩的潜在向量之后也使用额外的 RMSNorm 层，并在宽度瓶颈处乘以额外的缩放因子。在这种配置下，DeepSeek-V3 包含 671B 总参数，其中每个 token 激活 37B 参数。

训练超参数。我们采用 AdamW 优化器 (Loshchilov 和 Hutter, 2017)，超参数设置为 $\beta_1 = 0.9, \beta_2 = 0.95$ ，weight_decay 设置为 $= 0.1$ 。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。关于学习率调度，我们首先在前 2 K 步线性增加学习率从 0 到 2.2×10^{-4} 。然后，我们保持 2.2×10^{-4} 的恒定学习率，直到模型消耗了 10T 训练标记。随后，我们按照余弦衰减曲线，在 4.3T 标记上逐渐将学习率衰减到 2.2×10^{-5} 。在训练最后 500B 标记期间，我们首先在前 333B 标记上保持 2.2×10^{-5} 的恒定学习率，然后在剩余的 167 B 标记上切换到另一个恒定学习率 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，其中在前 469B 标记的训练中，批量大小从 3072 逐渐增加到 15360，然后在剩余的训练中保持 15360。我们利用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，路由专家将均匀部署在 8 个节点的 64 个 GPU 上。对于节点限制路由，每个标记最多发送到 4 个节点 (即 $M = 4$)。对于无辅助损失的负载均衡，我们设置前 14.3T 标记的偏置更新速度 γ 为 0.001，剩余 500B 标记的偏置更新速度为 0.0。对于平衡损失，我们设置 α 为 0.0001，仅为了防止任何单个序列内的极端不平衡。MTP 损失权重 λ 在前 10T 标记上设置为 0.3，在剩余的 4.8T 标记上设置为 0.1。

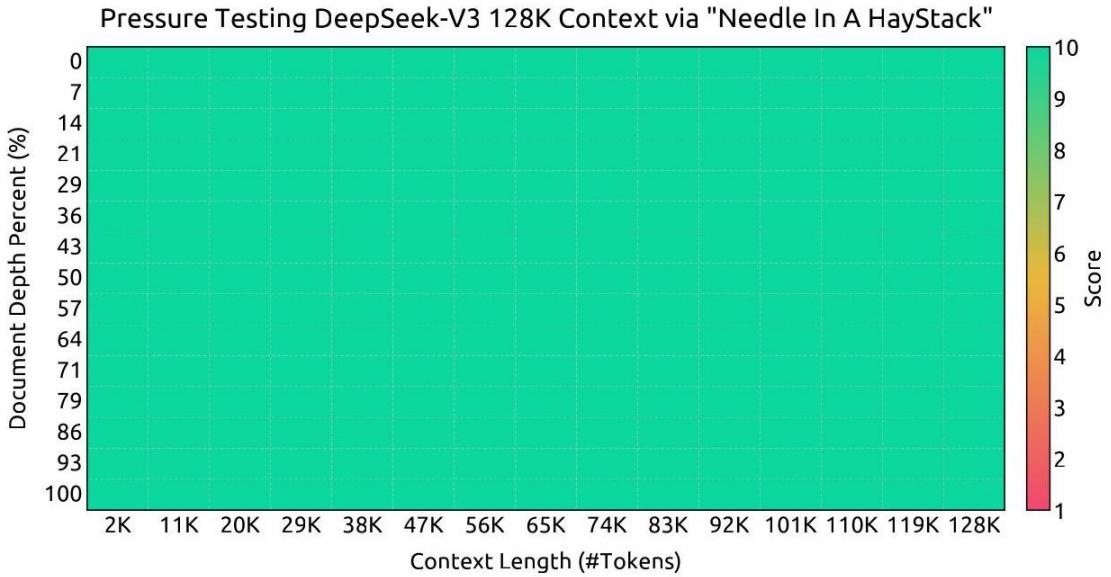


图 8 | 在 “大海捞针” (NIAH) 测试中的评估结果。DeepSeek-V3 在所有上下文窗口长度上均表现出色，直到 128 K 。

4.3 Long Context Extension **长上下文扩展**

我们采用与 DeepSeek-V2 (DeepSeek-AI, 2024c) 类似的方法来在 DeepSeek-V3 中启用长上下文功能。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，逐步将上下文窗口从 4 K 扩展到 32 K ，然后再扩展到 128 K 。YaRN 的配置与在 DeepSeek-V2 中使用的配置一致，仅应用于解耦的共享键 k_t^R 。两个阶段中的超参数保持相同，比例 $s = 40, \alpha = 1, \beta = 32$ ，以及缩放因子 $\sqrt{t} = 0.1 \ln s + 1$ 。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 128 K ，批量大小减少到 480。两个阶段的学习率均设置为 7.3×10^{-6} ，与预训练阶段的最终学习率匹配。

通过这种两阶段扩展训练，DeepSeek-V3 能够处理长达 128K 的输入，同时保持强大的性能。图 8 显示，经过监督微调后，DeepSeek-V3 在“大海捞针” (NIAH) 测试中表现出色，展示了在上下文窗口长度达到 128K 时的一致鲁棒性。

4.4 Evaluations **评估**

4.4.1 Evaluation Benchmarks **评估基准**

DeepSeek-V3 的基础模型是在包含英语和中文为主的多语言语料库上进行预训练的，因此我们在一系列主要使用英语和中文的基准测试上评估其性能，同时也包括一个多语言基准测试。我们的评估基于集成在 HAI-LLM 框架中的内部评估框架。考虑的基准测试按类别列出如下，其中带下划线的基准测试为中文的，带双下划线的基准测试为多语言的:

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

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

闭卷问答数据集包括 TriviaQA (Joshi 等人，2017) 和 NaturalQuestions (Kwiatkowski 等人，2019)。

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

词义消歧数据集包括 CLUEWSC (Xu 等人，2020) 和 WinoGrande Sakaguchi 等人 (2019)。

语言模型数据集包括 Pile (Gao 等人，2020)。

中国理解和文化数据集包括 CCPM (Li 等人, 2021)。

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

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

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

根据我们之前的工作 (DeepSeek-AI, 2024b c)，我们采用困惑度为基础的评估方法，适用于包括 HellaSwag, PIQA, WinoGrande, RACE-Middle, RACE-High, MMLU, MMLU-Redux, MMLU-Pro, MMMLU, ARC-Easy, ARC-Challenge, C-Eval, CMMLU, C3, 和 CCPM 的数据集，并采用生成为基础的评估方法，适用于 TriviaQA, NaturalQuestions, DROP, MATH, GSM8K, MGSM, HumanEval, MBPP, LiveCodeBench-Base, CRUXEval, BBH, AGIEval, CLUEWSC, CMRC, 和 CMath。此外，我们对 Pile-test 进行语言模型为基础的评估，并使用每字节比特数 (BPB) 作为度量标准，以确保使用不同分词器的模型之间公平比较。

	基准测试 (指标)	# Shots	DeepSeek-V2 基础版	Qwen2.5 72B 基础版	LLaMA-3.1 405B 基础版	DeepSeek-V3 Base
	架构	-	MoE	密集型	密集型	MoE
	# 激活参数	-	21B	72B	405B	37B
	# 总参数	-	236B	72B	405B	671B
英语	堆载测试 (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-简单 (EM)	25-shot	97.6	98.4	98.4	98.9
	ARC-挑战 (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-中学 (EM)	5-shot	73.1	68.1	74.2	67.1
	RACE-高中 (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
代码	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
数学	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
中文	CLUEWSC (EM)	5-shot	82.0	82.5	83.0	82.7
	C-Eval (EM)	5-shot	81.4	89.2	72.5	90.1
	CMMLU (EM)	5-shot	84.0	89.5	73.7	88.8
	CMRC (EM)	1-shot	77.4	75.8	76.0	76.3
	C3 (EM)	0-shot	77.4	76.7	79.7	78.6
	CCPM (EM)	0-shot	93.0	88.5	78.6	92.0
多语言	MMMLU-非英语 (EM)	5-shot	64.0	74.8	73.8	79.4

表 3 | DeepSeek-V3-Base 与其他代表性开源基础模型的比较。所有模型都在我们的内部框架中进行评估，并且共享相同的评估设置。得分差距不超过 0.3 的模型被认为处于同一水平。DeepSeek-V3-Base 在大多数基准测试中表现出最佳性能，特别是在数学和代码任务上。

4.4.2 Evaluation Results **评估结果**

在表 3 中，我们将 DeepSeek-V3 的基础模型与最先进的开源基础模型进行了比较，包括 DeepSeek-V2-Base(DeepSeek-AI，2024c)(我们的上一个版本)、Qwen 2.5 72B Base(Qwen，2024b) 和 LLaMA-3.1 405B Base(AI@Meta，2024b)。我们使用内部评估框架评估了所有这些模型，并确保它们共享相同的评估设置。请注意，由于过去几个月我们的评估框架发生了变化，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 在各个方面都优于 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。

从更详细的视角来看，我们将 DeepSeek-V3-Base 与其他开源基础模型进行了单独比较。(1) 与 DeepSeek-V2-Base 相比，由于我们的模型架构改进、模型规模和训练标记的扩展以及数据质量的提升，DeepSeek-V3-Base 达到了预期的显著更好的性能。(2) 与最先进的中文开源模型 Qwen2.5 72B Base 相比，DeepSeek-V3-Base 仅使用一半的激活参数就展示了显著的优势，特别是在英文、多语言、代码和数学基准测试中。至于中文基准测试，除了 CMMLU(一个中文多学科多项选择任务) 之外，DeepSeek-V3-Base 的表现也优于 Qwen2.5 72B。(3) 与拥有 11 倍激活参数的最大开源模型 LLaMA-3.1 405B Base 相比，DeepSeek-V3-Base 在多语言、代码和数学基准测试中也表现出更好的性能。至于英文和中文语言基准测试，DeepSeek-V3-Base 展现了具有竞争力或更好的性能，特别是在 BBH、MMLU 系列、DROP、C-Eval、CMMLU 和 CCPM 上。

由于我们的高效架构和全面的工程优化，DeepSeek-V3 达到了极高的训练效率。在我们的训练框架和基础设施下，每万亿标记的训练只需 180K H800 GPU 小时，这比训练 72B 或 405B 密集模型要便宜得多。

基准测试 (指标)	# 示例数量	小型 MoE 基线	小型 MoE 配备 MTP	LargeMoE 基线	LargeMoE w/MTP
# 激活参数 (推理)	-	2.4B	2.4B	20.9B	20.9B
# 总参数 (推理)	-	15.7B	15.7B	228.7B	228.7B
# 训练标记	-	1.33T	1.33T	540B	540B
Pile 测试 (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
$\text{MBPP}_{(\text{Pass@1})}$	3-shot	35.8	36.8	61.6	62.2
GSM8K (EM)	8-shot	25.4	31.4	72.3	74.0
数学 (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测试中一致提升了模型性能。

4.5 Discussion **讨论**

4.5.1 Ablation Studies for Multi-Token Prediction **多标记预测的消融研究**

在表 4 中，我们展示了 MTP 策略的消融结果。具体来说，我们在两种基线模型上验证了 MTP 策略，这些模型在不同的规模上进行了测试。在小规模上，我们训练了一个包含 157 亿总参数的基线 MoE 模型，使用了 1.33 万亿个 token。在大规模上，我们训练了一个包含 2287 亿总参数的基线 MoE 模型，使用了 5400 亿个 token。在这些模型之上，保持训练数据和其他架构相同，我们添加了一个 1 层的 MTP 模块，并使用 MTP 策略训练了两个模型以进行比较。需要注意的是，在推理过程中，我们直接丢弃了 MTP 模块，因此比较模型的推理成本完全相同。从表中可以看出，MTP 策略在大多数评估基准上始终提高了模型性能。

基准测试 (指标)	# 示例数	小型 MoE 辅助损失基础	小型 MoE 无辅助损失	LargeMoE 辅助损失基础	LargeMoE 无辅助损失
# 激活参数	-	2.4B	2.4B	20.9B	20.9B
# 总参数	-	15.7B	15.7B	228.7B	228.7B
# 训练标记	-	1.33T	1.33T	578B	578B
堆测试 (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 | 辅助损失无关平衡策略的消融结果。与纯粹基于辅助损失的方法相比，辅助损失无关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 **辅助损失无关平衡策略的消融研究**

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该策略，这些模型在不同规模下进行训练。在小规模上，我们在 1.33T 个标记上训练了一个包含 15.7B 个总参数的基线 MoE 模型。在大规模上，我们在 578B 个标记上训练了一个包含 228.7B 个总参数的基线 MoE 模型。这两个基线模型纯粹使用辅助损失来鼓励负载平衡，并使用带有 top-K 亲和力归一化的 sigmoid 门控函数。它们控制辅助损失强度的超参数分别与 DeepSeek-V2-Lite 和 DeepSeek-V2 相同。在这些两个基线模型之上，保持训练数据和其他架构相同，我们移除了所有辅助损失并引入无辅助损失平衡策略进行比较。从表中可以看出，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 **批处理负载平衡** VS. **序列负载平衡**

无辅助损失平衡与序列辅助损失之间的关键区别在于它们的平衡范围: 批处理与序列。与序列辅助损失相比，批处理平衡施加了一个更灵活的约束，因为它不对每个序列强制执行域内平衡。这种灵活性允许专家更好地专业化于不同的领域。为了验证这一点，我们记录并分析了在 Pile 测试集的不同域上，一个基于辅助损失的 16B 基线模型和一个无辅助损失的 16B 模型的专家负载。如图 9 所示，我们观察到无辅助损失模型展示了预期的专家专业化模式。

为了进一步研究这种灵活性与模型性能优势之间的相关性，我们还设计并验证了一种按批次的辅助损失，它鼓励在每个训练批次上而不是在每个序列上实现负载均衡。实验结果表明，当达到相似的按批次负载均衡水平时，按批次的辅助

损失也能达到与无辅助损失方法相似的模型性能。具体来说，在我们使用 10 亿参数 MoE 模型的实验中，验证损失分别为:2.258(使用按序列的辅助损失)，2.253(使用无辅助损失方法)，以及 2.253(使用按批次的辅助损失)。我们在 30 亿参数 MoE 模型上也观察到了类似的结果: 使用按序列的辅助损失的模型验证损失为 2.085，而使用无辅助损失方法或按批次的辅助损失的模型验证损失均为 2.080。

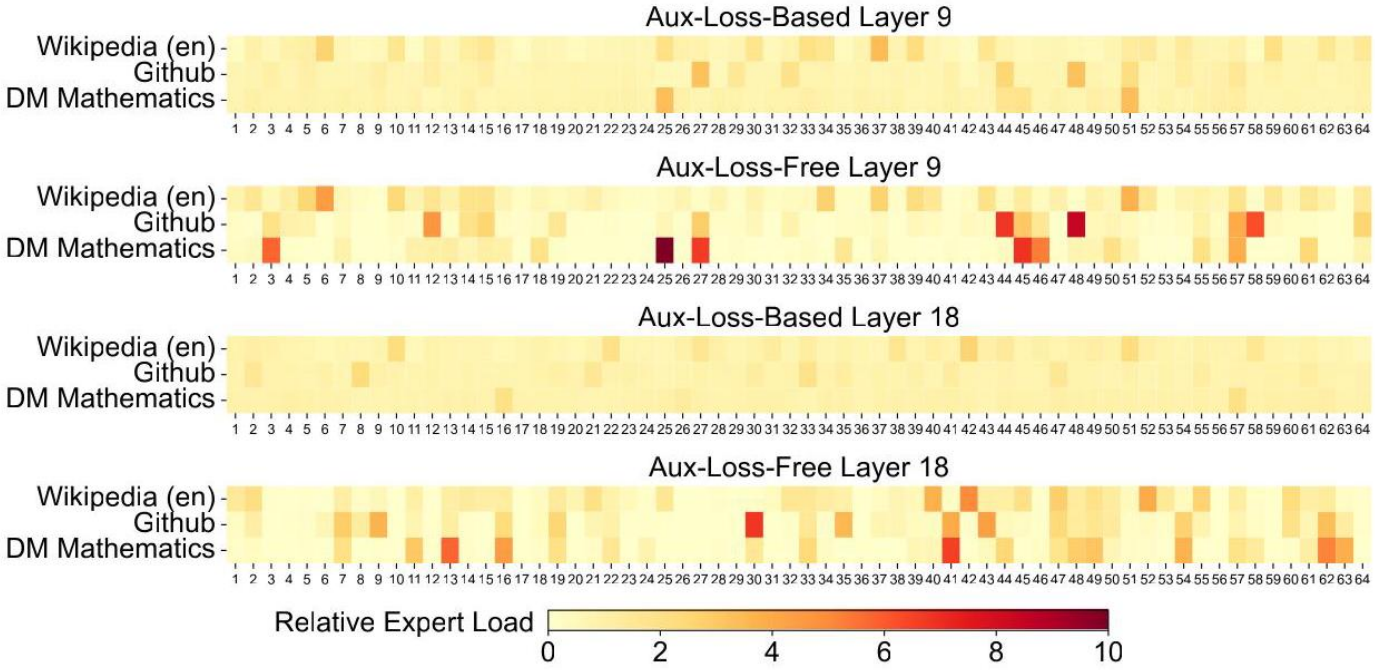


图 9 | Pile 测试集三个领域中无辅助损失和基于辅助损失模型的专家负载。无辅助损失模型显示出比基于辅助损失模型更明显的专家专业化模式。相对专家负载表示实际专家负载与理论上平衡的专家负载之间的比率。由于篇幅限制，我们仅展示了两层的结果作为示例，所有层的结果详见附录 C。

此外，尽管按批次的负载均衡方法显示出一致的性能优势，它们也面临着两个潜在的效率挑战:(1) 某些序列或小批次内的负载不平衡，以及 (2) 推理过程中由领域迁移引起的负载不平衡。第一个挑战自然地通过我们的训练框架解决了，该框架使用大规模专家并行和数据并行，保证了每个微批次的较大规模。对于第二个挑战，我们也设计并实现了一个高效的推理框架，通过冗余专家部署来克服它，如第 3.4 节所述。

5 Post-Training **训练后处理**

5.1 Supervised Fine-Tuning **监督微调**

我们整理了包含 150 万实例的指令微调数据集，这些实例跨越多个领域，每个领域都采用了与其特定需求相匹配的数据创建方法。

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

为了建立我们的方法论，我们首先为特定领域 (如编程、数学或通用推理) 开发一个专家模型，使用结合了监督微调 (SFT) 和强化学习 (RL) 的训练管道。该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 >，第二种则在问题和 R1 响应旁边加入系统提示，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含反思和验证机制的响应的指令。在 RL 阶段，模型利用高温采样生成响应，即使没有明确的系统提示，也能整合 R1 生成的数据和原始数据中的模式。经过数百次 RL 步骤后，中间 RL 模型学会了整合 R1 模式，从而战略性地提升整体性能。

完成 RL 训练阶段后，我们实施拒绝采样以整理高质量的 SFT 数据供最终模型使用，其中专家模型被用作数据生成源。该方法确保最终训练数据保留 DeepSeek-R1 的优点，同时产生简洁有效的响应。

非推理数据。对于非推理数据，如创意写作、角色扮演和简单的问答，我们使用 DeepSeek-V2.5 生成响应，并聘请人类注释员验证数据的准确性和正确性。

SFT 设置。我们使用 SFT 数据集对 DeepSeek-V3-Base 进行微调，微调两个周期，使用余弦衰减学习率调度，从 5×10^{-6} 逐渐减少到 1×10^{-6} 。在训练过程中，每个单一序列由多个样本打包而成。然而，我们采用样本掩码策略以确保这些示例保持隔离且彼此不可见。

5.2 Reinforcement Learning 强化学习

5.2.1 Reward Model 奖励模型

在我们的 RL 过程中，我们采用基于规则的奖励模型 (RM) 和基于模型的 RM。

基于规则的 RM。对于可以使用特定规则验证的问题，我们采用基于规则的奖励系统来确定反馈。例如，某些数学问题有确定的结果，我们要求模型在指定格式 (例如，在一个框内) 提供最终答案，使我们能够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。通过尽可能地利用基于规则的验证，我们确保了更高的可靠性，因为这种方法不易被操纵或利用。

基于模型的 RM。对于具有自由形式真实答案的问题，我们依赖奖励模型来确定响应是否符合预期的真实答案。相反，对于没有确定真实答案的问题，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作为输入提供反馈。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，这些数据不仅提供最终奖励，还包括导致奖励的思维链。这种方法有助于减轻特定任务中奖励操纵的风险。

5.2.2 Group Relative Policy Optimization 组相对策略优化

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

$$\mathcal{J}_{GRPO}(\theta) = \mathbb{E}\left[q \sim P(Q), \{o_i\}_{i=1}^{G} \sim \pi_{\theta_{\text{old}}}(O \mid q)\right]$$

$$\frac{1}{G}\sum_{i=1}^{G}\left(\min\left(\frac{\pi_\theta(o_i \mid q)}{\pi_{\theta_{old}}(o_i \mid q)}A_i, \text{clip}\left(\frac{\pi_\theta(o_i \mid q)}{\pi_{\theta_{old}}(o_i \mid q)}, 1-\varepsilon, 1+\varepsilon\right)A_i\right) - \beta\mathbb{D}_{KL}(\pi_\theta \| \pi_{ref})\right), \tag{26}$$

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

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

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

我们在 RL 过程中纳入了来自不同领域的提示，例如编码、数学、写作、角色扮演和问题回答。这种方法不仅使模型更紧密地与人类偏好对齐，还提高了在基准测试中的性能，特别是在可用 SFT 数据有限的情况下。

5.3 Evaluation 评估

5.3.1 Evaluation Settings 评估设置

评估基准。除了我们用于基础模型测试的基准外，我们还在 IFEval(周等人，2023)，FRAMES(克里希纳等人，2024)，LongBench v2(白等人，2024)，GPQA(莱因等人，2023)，SimpleQA(OpenAI，2024c)，C-SimpleQA(何等人，2024)，SWE-Bench Verified(OpenAI，2024d)，Aider[1]，LiveCodeBench(贾因等人，2024)(2024 年 8 月至 11 月的问题)，Codeforces[2]，中国全国高中数学奥林匹克 (CNMO 2024) [3] ，以及 2024 年美国数学邀请赛 (AIME 2024)(MAA，2024) 上进一步评估了指令模型。

对比基线。我们对我们的聊天模型进行了全面的评估，对比了几种强大的基线模型，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，通过各自的 API 进行评估。

详细评估配置。对于包括 MMLU、DROP、GPQA 和 SimpleQA 在内的标准基准测试，我们采用 simple-evals 框架 [4] 中的评估提示。我们使用 Zero-Eval 提示格式 (Lin, 2024) 在零样本设置下对 MMLU-Redux 进行评估。对于其他数据集，我们遵循数据集创建者提供的原始评估协议和默认提示。对于代码和数学基准测试，HumanEval-Mul 数据集总共包含 8 种主流编程语言 (Python、Java、Cpp、C#、JavaScript、TypeScript、PHP 和 Bash)。我们使用 CoT 和非 CoT 方法评估模型在 LiveCodeBench 上的表现，其中数据收集时间为 2024 年 8 月至 2024 年 11 月。Codeforces 数据集使用

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

竞争对手的百分比进行衡量。SWE-Bench 验证使用无代理框架 (Xia et al., 2024) 进行评估。我们使用"diff" 格式评估与 Aider 相关的基准测试。对于数学评估，AIME 和 CNMO 2024 使用 0.7 的温度进行评估，并在 16 次运行中取平均值，而 MATH-500 使用贪婪解码。我们允许所有模型为每个基准测试输出最多 8192 个 token。

	基准测试 (指标)	DeepSeek V2-0506	DeepSeek V2.5-0905	Qwen2.5 72B-Inst.	LLaMA-3.1 405B-Inst.	Claude-3.5- Sonnet-1022	GPT-40 0513	DeepSeek V3
	架构	MoE	MoE	密集	密集	-	-	MoE
	# 激活参数	21B	21B	72B	405B	-	-	37B
	# 总参数	236B	236B	72B	405B	-	-	671B
英语	MMLU (EM)	78.2	80.6	85.3	88.6	88.3	87.2	88.5
	MMLU-Redux (EM)	77.9	80.3	85.6	86.2	88.9	88.0	89.1
	MMLU-Pro (EM)	58.5	66.2	71.6	73.3	78.0	72.6	75.9
	DROP(3-shot F1)	83.0	87.8	76.7	88.7	88.3	83.7	91.6
	IF-Eval(Prompt Strict)	57.7	80.6	84.1	86.0	86.5	84.3	86.1
	GPQA-Diamond(Pass@1)	35.3	41.3	49.0	51.1	65.0	49.9	59.1
	SimpleQA(正确)	9.0	10.2	9.1	17.1	28.4	38.2	24.9
	FRAMES(准确率)	66.9	65.4	69.8	70.0	72.5	80.5	73.3
	LongBench v2(准确率)	31.6	35.4	39.4	36.1	41.0	48.1	48.7
代码	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(百分位)	17.5	35.6	24.8	25.3	20.3	23.6	51.6
	经过验证 (已解决)	-	22.6	23.8	24.5	50.8	38.8	42.0
	Aider-Edit(准确)	60.3	71.6	65.4	63.9	84.2	72.9	79.7
	Aider-Polyglot(准确)	-	18.2	7.6	5.8	45.3	16.0	49.6
数学	AIME 2024(首次通过)	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(首次通过)	2.8	10.8	15.9	6.8	13.1	10.8	43.2
中文	CLUEWSC(EM)	89.9	90.4	91.4	84.7	85.4	87.9	90.9
	C-Eval(EM)	78.6	79.5	86.1	61.5	76.7	76.0	86.5
	C-SimpleQA(正确)	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。所有模型都在限制输出长度为 8 K 的配置下进行评估。包含少于 1000 个样本的基准测试使用不同的温度设置进行多次测试，以得出稳健的最终结果。DeepSeek-V3 作为表现最佳的开源模型，同时也表现出与前沿闭源模型相竞争的性能。

5.3.2 Standard Evaluation **标准评估**

表 6 展示了评估结果，表明 DeepSeek-V3 是表现最佳的开源模型。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。

英语基准测试。MMLU 是一个广泛认可的基准测试，旨在评估大型语言模型在不同知识领域和任务上的性能。DeepSeek-V3 表现出竞争力，与 LLaMA-3.1-405B、GPT-40 和 Claude-Sonnet 3.5 等顶级模型处于同一水平，同时显著优于 Qwen2.5 72B。此外，在更具挑战性的教育知识基准 MMLU-Pro 上，DeepSeek-V3 表现优异，紧随 Claude-Sonnet 3.5 之后。在经过修正标签的 MMLU-Redux 版本上，DeepSeek-V3 也超越了其同行。此外，在博士水平的评估测试平台 GPQA-Diamond 上，DeepSeek-V3 取得了显著成果，仅次于 Claude 3.5 Sonnet，并大幅领先于其他所有竞争对手。

在长上下文理解基准测试如 DROP、LongBench v2 和 FRAMES 中，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，超越了该类别中的所有其他模型。在 FRAMES 上，这是一个需要在 100k tokens 上下文中进行问答的基准测试，DeepSeek-V3 紧随 GPT-4o 之后，同时大幅领先于其他所有模型。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 的长上下文能力还通过其在 LongBench v2 上的最佳表现得到了进一步验证，该数据集在 DeepSeek V3 发布前几周才发布。在事实知识基准测试 SimpleQA 上，DeepSeek-V3 落后于 GPT-4o 和 Claude-Sonnet，这主要是由于其设计重点和资源分配。DeepSeek-V3 将更多的训练 tokens 分配给学习中文知识，从而在 C-SimpleQA 上表现出色。在指令跟随基准测试中，DeepSeek-V3 显著超越了其前身 DeepSeek-V2 系列，突显了其在理解和遵守用户定义的格式约束方面的能力得到了提升。

代码和数学基准测试。编码是大语言模型面临的具有挑战性和实用性的任务，包括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 预期将推动与编码相关的工程任务的发展。通过提供其强大的功能，DeepSeek-V3 可以在软件工程和算法开发等领域推动创新和改进，使开发者和研究人员能够拓展开源模型在编码任务中的能力边界。在算法任务中，DeepSeek-V3 展现了卓越的性能，在 HumanEval-Mul 和 LiveCodeBench 等基准测试中超越了所有基线模型。这一成功归功于其先进的知识蒸馏技术，该技术有效地增强了其在算法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，显著超越了基线模型，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 在绝对分数上比第二好的模型 Qwen2.5 72B 高出约 10%，这是一个对于如此具有挑战性的基准测试而言相当大的差距。这一显著的能力突显了从 DeepSeek-R1 蒸馏技术的有效性，该技术已被证明对非 o1 类模型非常有益。

模型	Arena-Hard	AlpacaEval 2.0
DeepSeek-V2.5-0905	76.2	50.5
Qwen2.5-72B-指令	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 | 英语开放式对话评估。对于 AlpacaEval 2.0，我们使用长度控制胜率作为指标。

中文基准。Qwen 和 DeepSeek 是两个具有强大中英文支持的代表性模型系列。在事实基准 Chinese SimpleQA 上，尽管 Qwen2.5 在包含 18T 个 tokens 的更大语料库上进行了训练，这些 tokens 比 DeepSeek-V3 预训练的 14.8T 个 tokens 多 20%，DeepSeek-V3 仍比 Qwen2.5-72B 高出 16.4 分。

在 C-Eval(中文教育知识评估的代表性基准) 和 CLUEWSC(中文 Winograd 架构挑战) 上，DeepSeek-V3 和 Qwen2.5-72B 展现了相似的性能水平，表明这两个模型都很好地优化了中文语言推理和教育任务。

5.3.3 Open-Ended Evaluation **开放性评估**

除了标准基准，我们还使用 LLM 作为裁判对我们的模型进行开放性生成任务的评估，结果见表 7。具体来说，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 作为成对比较的裁判。在 Arena-Hard 上，DeepSeek-V3 达到了超过 86% 的胜率，与 Claude-Sonnet-3.5-1022 等顶级模型持平。这突显了 DeepSeek-V3 的强大能力，特别是在处理复杂提示方面，包括编码和调试任务。此外，DeepSeek-V3 达到了一个开创性的里程碑，作为第一个在 Arena-Hard 基准上超过 85% 的开源模型。这一成就显著缩小了开源和闭源模型之间的性能差距，为开源模型在具有挑战性的领域中所能达到的标准设定了新标准。

同样，DeepSeek-V3 在 AlpacaEval 2.0 上展示了卓越的性能，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中的出色能力。值得注意的是，它比 DeepSeek-V2.5-0905 高出 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 **作为生成奖励模型**

我们比较了 DeepSeek-V3 与最先进的模型 (即 GPT-4o 和 Claude-3.5) 的判断能力。表 8 展示了这些模型在 RewardBench (Lambert et al. 2024) 上的表现。DeepSeek-V3 达到了与 GPT-40-0806 和 Claude-3.5-Sonnet-1022 最佳版本相当的性能，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，我们使用 DeepSeek-V3 结合投票来提供开放式问题的自我反馈，从而提高对齐过程的有效性和鲁棒性。

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

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

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

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

5.4 Discussion **讨论**

5.4.1 Distillation from DeepSeek-R1 **来自** DeepSeek-R1 **的蒸馏**

我们基于 DeepSeek-V2.5 研究了来自 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，显示在 LiveCodeBench 和 MATH-500 基准测试中均有显著提升。我们的实验揭示了一个有趣的权衡: 蒸馏带来了更好的性能，但也大幅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 在蒸馏过程中的最优设置。

我们的研究表明，从推理模型中进行知识蒸馏是训练后优化的一个有前景的方向。虽然我们当前的工作集中在从数学和编程领域提取数据，但这种方法显示出在各种任务领域中具有广泛的应用潜力。在这些特定领域中展示的有效性表明，长-CoT 蒸馏可能对提高其他需要复杂推理的认知任务的模型性能有价值。在不同领域进一步探索这种方法仍然是未来研究的重要方向。

5.4.2 Self-Rewarding **自我奖励**

奖励在强化学习中扮演着关键角色，引导优化过程。在通过外部工具进行验证较为直接的领域，如某些编程或数学场景中，强化学习表现出色。然而，在更一般的场景中，通过硬编码构建反馈机制是不切实际的。在 DeepSeek-V3 的开发过程中，对于这些更广泛的场景，我们采用宪法 AI 方法 (Bai 等人，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，大幅提升了 DeepSeek-V3 在主观评估中的表现。通过整合额外的宪法输入，DeepSeek-V3 可以优化向宪法方向发展。我们相信，这种结合补充信息与大语言模型作为反馈来源的范式至关重要。大语言模型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，最终促进大语言模型的自我改进。除了自我奖励之外，我们还致力于发现其他通用且可扩展的奖励方法，以持续提升模型在一般场景中的能力。

5.4.3 Multi-Token Prediction Evaluation **多标记预测评估**

通过 MTP 技术，DeepSeek-V3 预测的不是下一个单一标记，而是接下来的 2 个标记。结合推测解码框架 (Leviathan 等人，2023；Xia 等人，2023)，可以显著加速模型的解码速度。一个自然的问题是关于额外预测标记的接受率。根据我们的评估，第二个标记预测的接受率在各种生成主题中均在 85% 至 90% 之间，显示出一致的可靠性。这种高接受率使 DeepSeek-V3 能够实现显著提升的解码速度，达到 1.8 倍的 TPS(每秒标记数)。

6 Conclusion, Limitations, and Future Directions **结论、局限与未来方向**

在本文中，我们介绍了 DeepSeek-V3，这是一个拥有 671B 总参数和 37B 激活参数的大规模 MoE 语言模型，训练数据量为 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均衡策略，并设定了多标记预测的训练目标以实现更强的性能。由于支持 FP8 训练和精心的工程优化，DeepSeek-V3 的训练成本效益显著。训练后也成功地提炼了来自 DeepSeek-R1 系列模型的推理能力。全面的评估表明，DeepSeek-V3 已经成为目前最强的开源模型，并且其性能与 GPT-4o 和 Claude-3.5-Sonnet 等领先的闭源模型相当。尽管性能强大，但它也保持了经济的训练成本。其完整的训练，包括预训练、上下文长度扩展和训练后处理，仅需 2.788M H800 GPU 小时。

尽管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，尤其是在部署方面。首先，为了确保高效的推理，DeepSeek-V3 的推荐部署单元相对较大，这可能会给小型团队带来负担。其次，尽管我们的 DeepSeek-V3 部署策略已经实现了比 DeepSeek-V2 高出两倍以上的端到端生成速度，但仍存在进一步提升的潜力。幸运的是，这些局限性有望随着更先进硬件的发展而自然得到解决。

DeepSeek 一直坚持长期主义的开源模型路线，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的研究投资。

- 我们将持续研究和优化我们的模型架构，旨在进一步提高训练和推理效率，努力实现对无限上下文长度的有效支持。此外，我们将尝试突破 Transformer 的架构限制，从而推动其建模能力的边界。
- 我们将持续迭代我们的训练数据的数量和质量，并探索整合额外的训练信号来源，旨在推动数据在更广泛维度上的扩展。
- 我们将持续探索和迭代我们模型的深度思考能力，旨在通过扩展其推理长度和深度来增强其智能和解决问题的能力。
- 我们将探索更全面和多维度的模型评估方法，以防止研究中优化固定一组基准的趋势，这可能会对模型能力产生误导，并影响我们对其基础评估。

参考文献

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

AI@Meta.Llama 3.1 model card, 2024b. URL https://github.com/meta-llama/llama-m odels/blob/main/models/l-lama3_1/MODEL_CARD.md

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

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

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

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

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

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

M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. de Oliveira Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, A. Ray, R. Puri, G. Krueger, M. Petrov, H. Khlaaf, G. Sastry, P. Mishkin,

B. Chan, S. Gray, N. Ryder, M. Pavlov, A. Power, L. Kaiser, M. Bavarian, C. Winter, P. Tillet, F. P. Such, D. Cummings, M. Plappert, F. Chantzis, E. Barnes, A. Herbert-Voss, W. H. Guss, A. Nichol, A. Paino, N. Tezak, J. Tang, I. Babuschkin, S. Balaji, S. Jain, W. Saunders, C. Hesse, 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/D19-1 600

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.2406.11 931

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

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

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

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

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

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

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.or g/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/techno logy/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/10.485 50/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.or g/abs/1806.03377

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

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

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

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. 1. 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/10.485 50/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/d1 7–1082.

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.

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/leviathan23 a.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: 10.1145/345881 7.3476145.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/ZeroE

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-bound aries-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.

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 NVSH-MEM and GPUDirect Async. https://developer.nvidia.com/blog/improving-net work-performance-of-hpc-systems-using-nvidia-magnum-io-nvshmem-and-g pudirect-async, 2022.

NVIDIA. Blackwell architecture. https://www.nvidia.com/en-us/data-center/tech nologies/blackwell-architecture/ 2024a.

NVIDIA. TransformerEngine, 2024b. URL https://github.com/NVIDIA/TransformerE ngine. 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/

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/b log/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. 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?i d=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/cutlas S

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,

R. Hou, H. Inan, M. Kardas, V. Kerkez, M. Khabsa, I. Kloumann, A. Korenev, P. S. Koura, M. Lachaux, T. Lavril, J. Lee, D. Liskovich, Y. Lu, Y. Mao, X. Martinet, T. Mihaylov, P. Mishra, I. Molybog, Y. Nie, A. Poulton, J. Reizenstein, R. Rungta, K. Saladi, A. Schelten, R. Silva, E. M. Smith, R. Subramanian, X. E. Tan, B. Tang, R. Taylor, A. Williams, J. X. Kuan, P. Xu, Z. Yan, I. Zarov, Y. Zhang, A. Fan, M. Kambadur, S. Narang, A. Rodriguez, R. Stojnic, S. Edunov, and T. Scialom. Llama 2: Open foundation and fine-tuned chat models. CoRR, abs/2307.09288, 2023b. doi: 10.48550/arXiv.2307.09288. URL https://doi.org/10.48550/arXiv.2307 09288

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polo-sukhin. 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/arX iv.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. 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/p1 9–1472

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.

Appendix 附录

A. 贡献与致谢

在每个角色中，作者按首名字母顺序列出。标记有 * 的名字表示已离开我们的团队的个人。

B. 低精度训练的消融研究

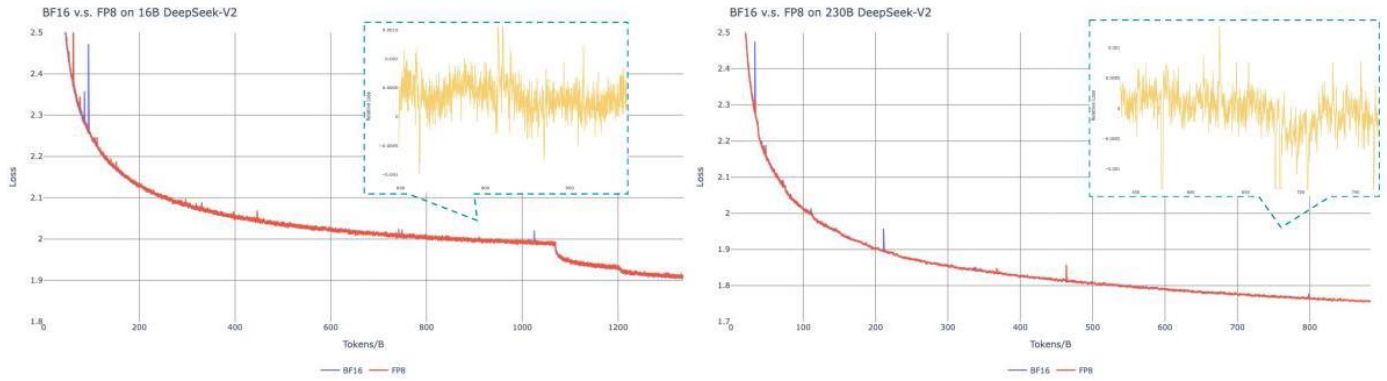


图 10 | BF16 与 FP8 训练的损失曲线对比。结果通过指数移动平均 (EMA) 平滑，系数为 0.9。

B.1. FP8 与 BF16 训练

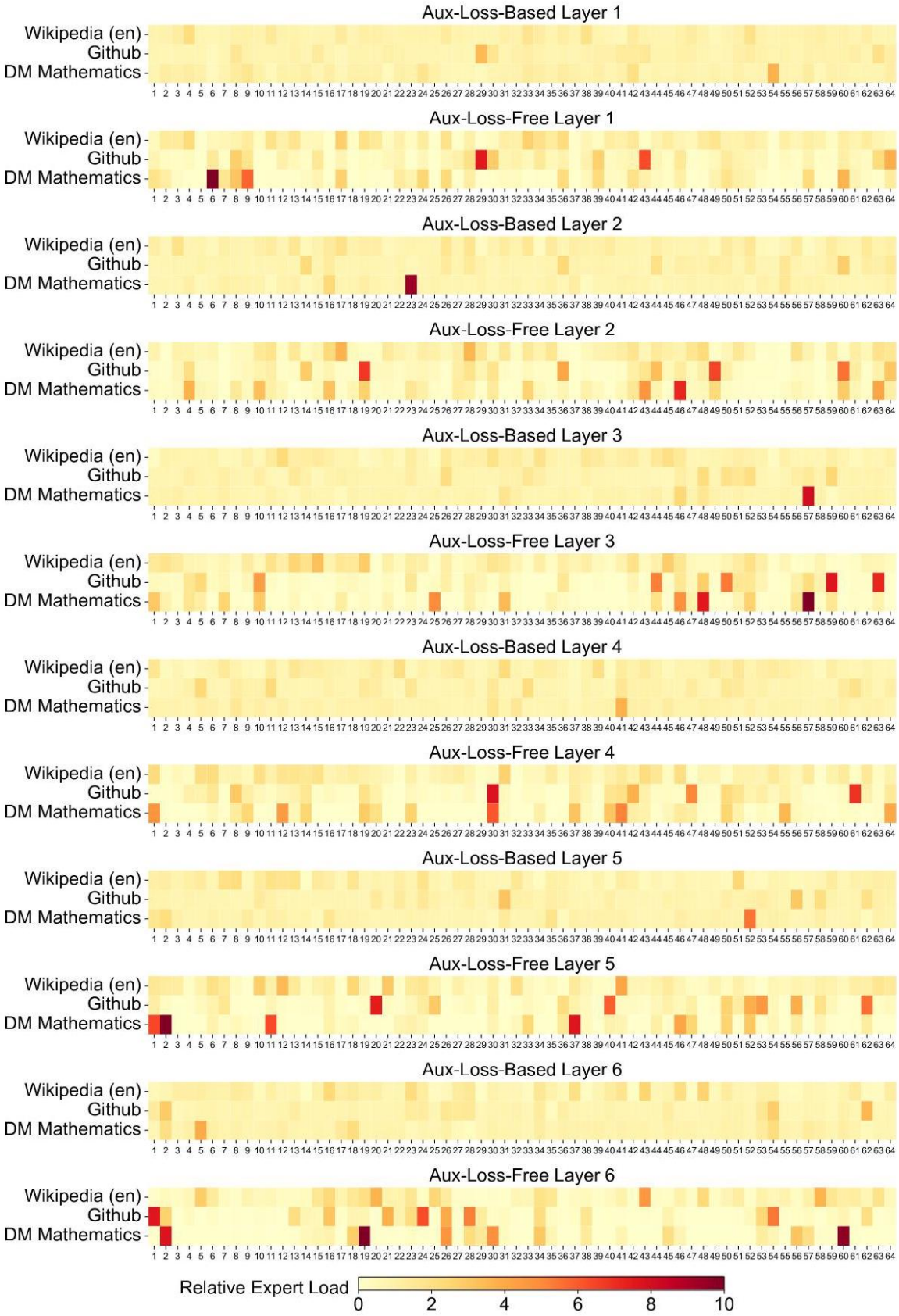
我们通过与 BF16 训练在两个不同规模的基线模型上的比较来验证我们的 FP8 混合精度框架。在小规模上，我们在 1.33T 标记上训练一个包含大约 16B 总参数的基线 MoE 模型。在大规模上，我们在大约 0.9T 标记上训练一个包含大约 230B 总参数的基线 MoE 模型。我们在图 10 中展示了训练曲线，并证明了相对误差在我们的高精度累积和细粒度量化策略下保持在 0.25% 以下。

B.2. 关于块量化讨论

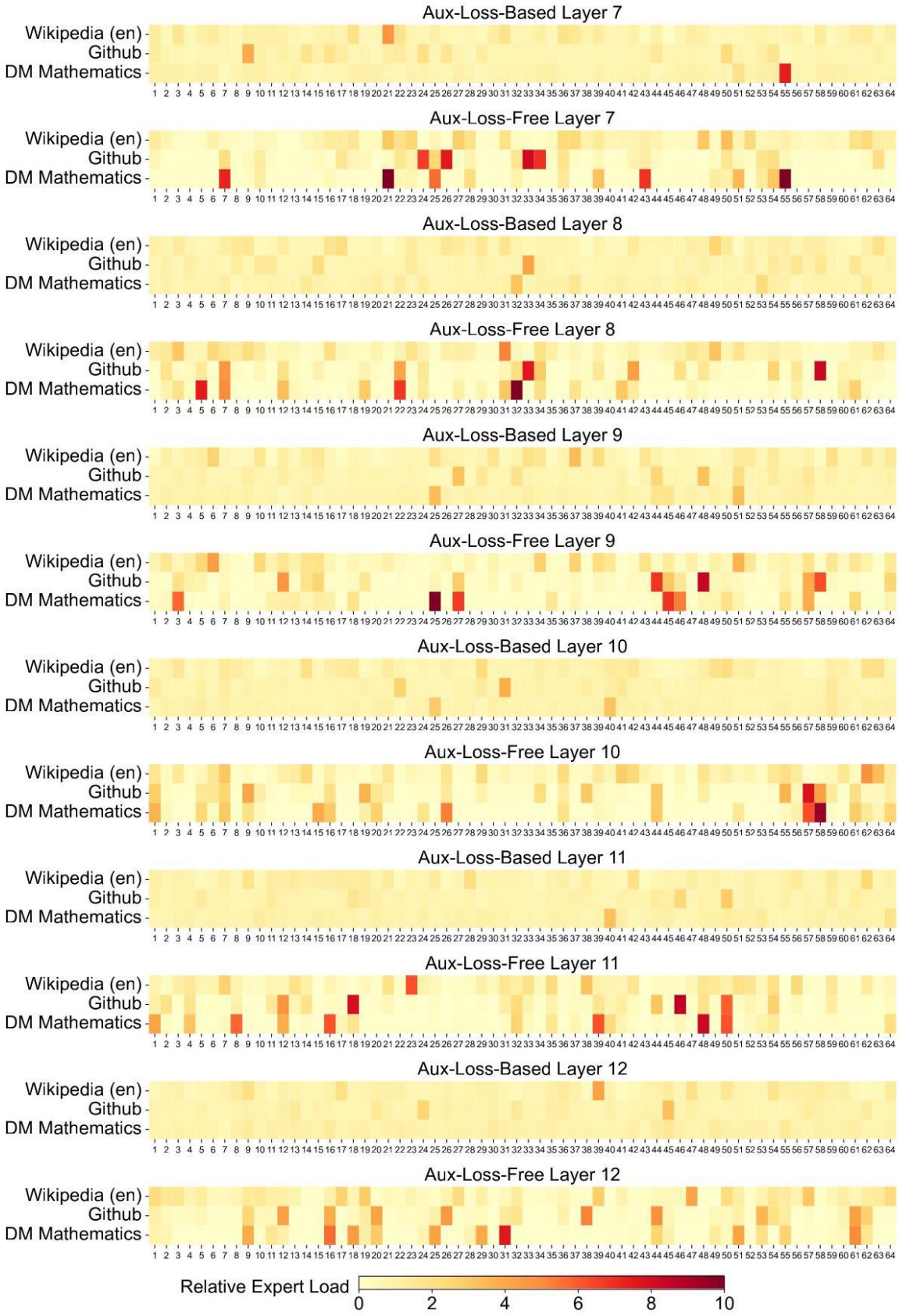
尽管我们的基于瓷砖的细粒度量化有效地减轻了特征异常值引入的误差，但它需要为激活量化使用不同的分组，即前向传递中的 1x128 和后向传递中的 128x1。激活梯度也需要类似的处理过程。一种直接的策略是按照我们量化模型权重的方式，对每 128x128 个元素进行块量化。这样，只需要进行转置操作即可用于后向传递。因此，我们进行了一项实验，其中与 Dgrad 相关的所有张量都基于块进行量化。结果表明，计算激活梯度并以链式方式反向传播到浅层的 Dgrad 操作对精度非常敏感。具体来说，激活梯度的块量化导致了一个包含大约 16B 总参数的 MoE 模型在训练了大约 300 B 个标记时模型发散。我们假设这种敏感性源于激活梯度在标记之间高度不平衡，导致了与标记相关的异常值 (Xi 等，2023)。这些异常值无法通过块量化方法有效管理。

C. 16B 辅助损失基和无辅助损失模型的专家专业化模式

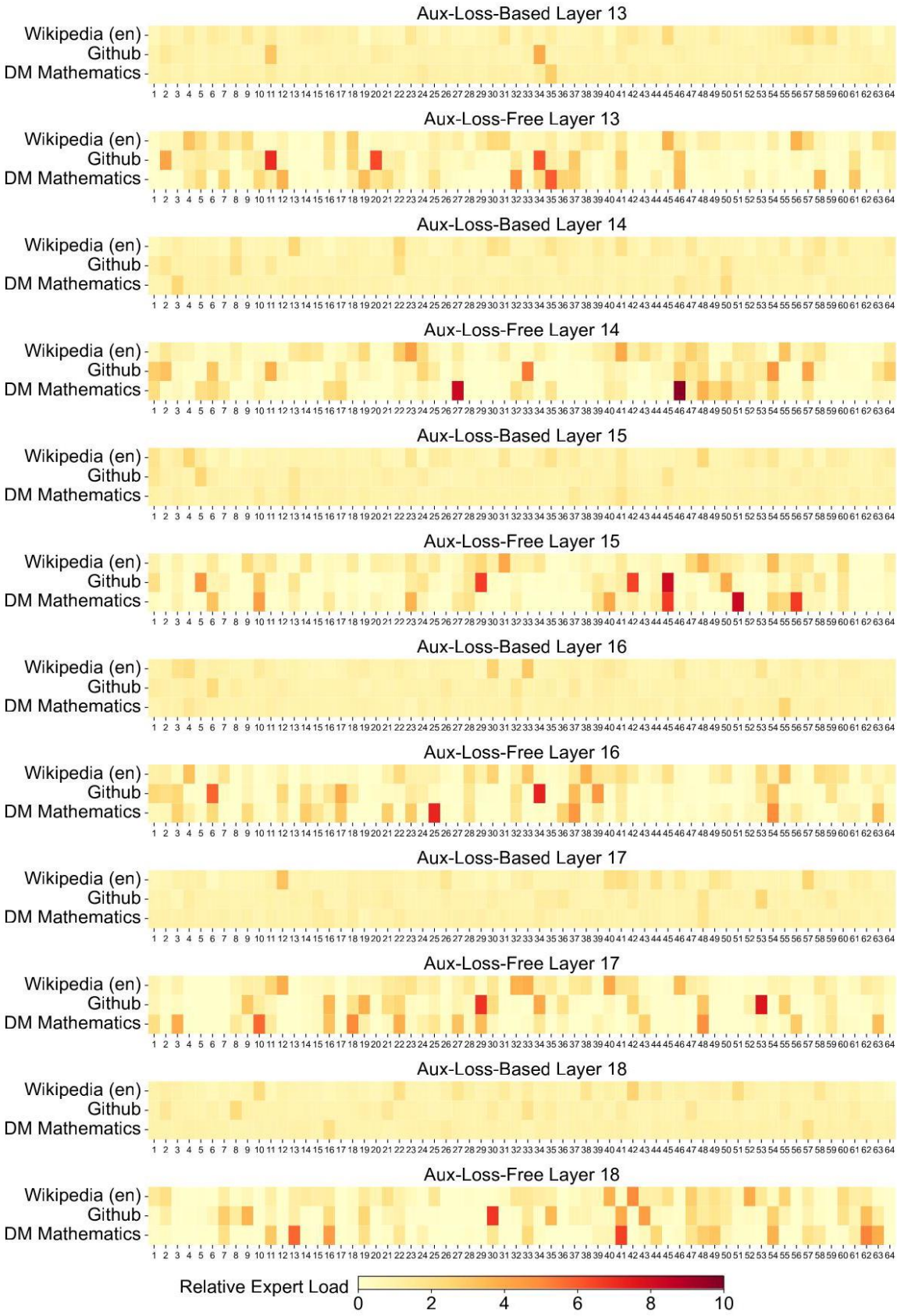
我们记录了 16B 辅助损失基线模型和无辅助损失模型在 Pile 测试集上的专家负载。如图 10 所示，无辅助损失模型在所有层中表现出更大的专家专业化。


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

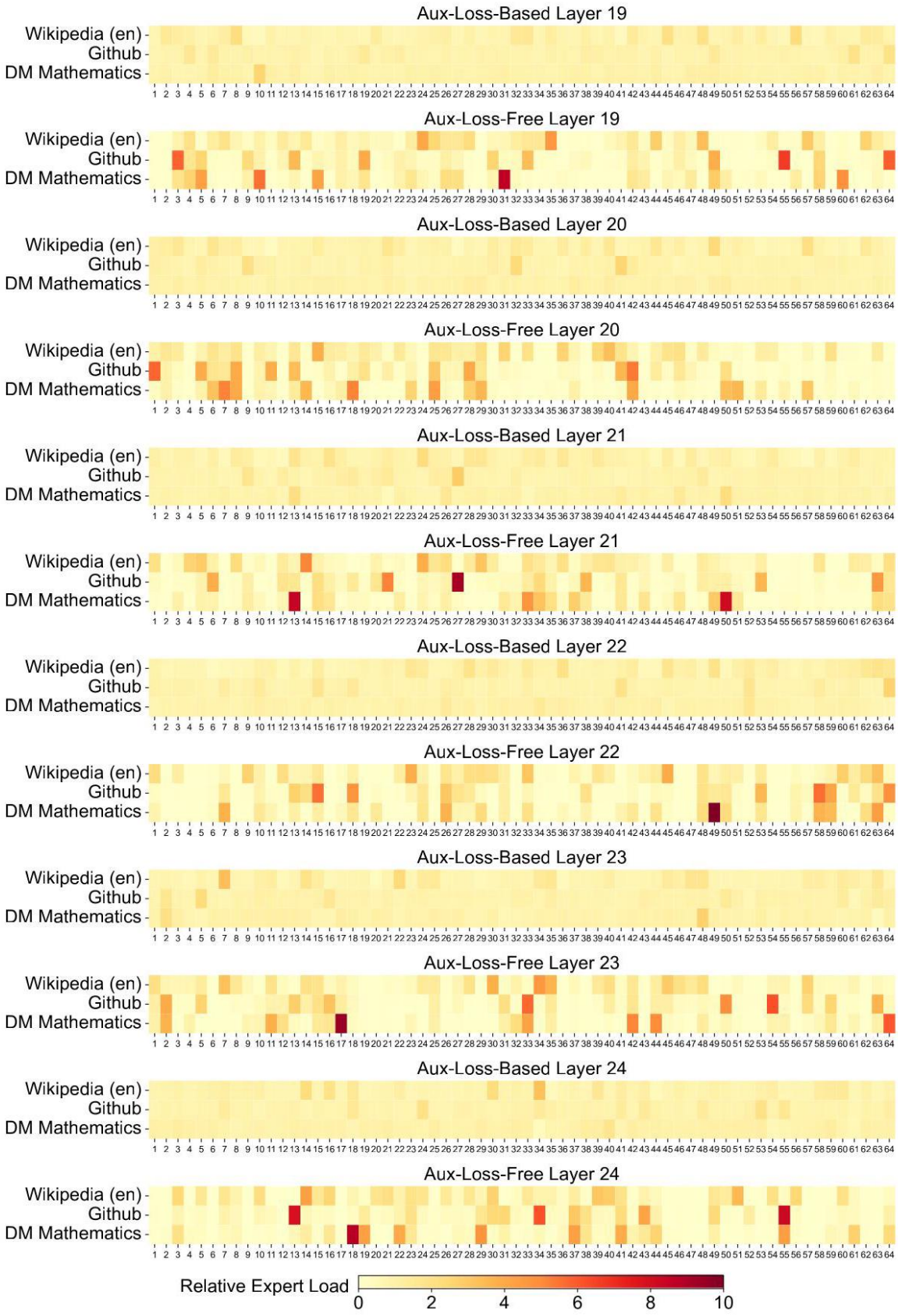
(a) Layers 1-7


Aux-Loss-Based Layer 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
Relative Expert Load
0
2
4
6
8
10

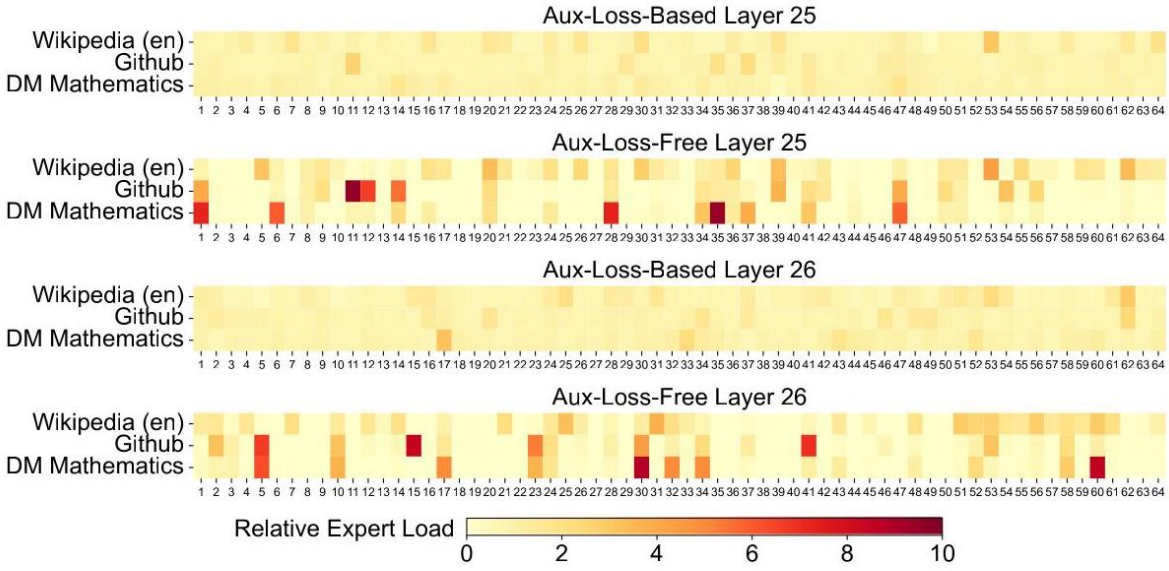
(b) Layers 7-13



(c) Layers 13-19



(d) Layers 19-25



(e) Layers 25-27

图 10 | 无辅助损失和辅助损失模型在 Pile 测试集三个领域中的专家负载。无辅助损失模型比辅助损失模型显示出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上平衡的专家负载之间的比率。