

DeepSeek-V3 技术报告

深度搜索-AI

research@deepseek.com

抽象

我们介绍了 DeepSeek-V3，这是一个强大的专家混合（MoE）语言模型，总共有 671B 个参数，每个令牌激活了 37B。为了实现高效的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 中得到了全面验证。此外，DeepSeek-V3 开创了一种用于负载均衡的辅助无损策略，并设置了多标记预测训练目标以获得更强的性能。我们在 14.8 万亿个多样化和高质量的代币上对 DeepSeek-V3 进行预训练，然后是监督微调和强化学习阶段，以充分利用其功能。综合评估表明，DeepSeek-V3 的性能优于其他开源模型，并实现了与领先的闭源模型相当的性能。尽管性能出色，但 DeepSeek-V3 只需要 2.788M H800 GPU 小时即可进行完整训练。此外，它的训练过程非常稳定。在整个训练过程中，我们没有遇到任何无法恢复的损失峰值或执行任何回滚。

模型检查点可在 <https://github.com/deepseek-ai/DeepSeek-V3> 获取。

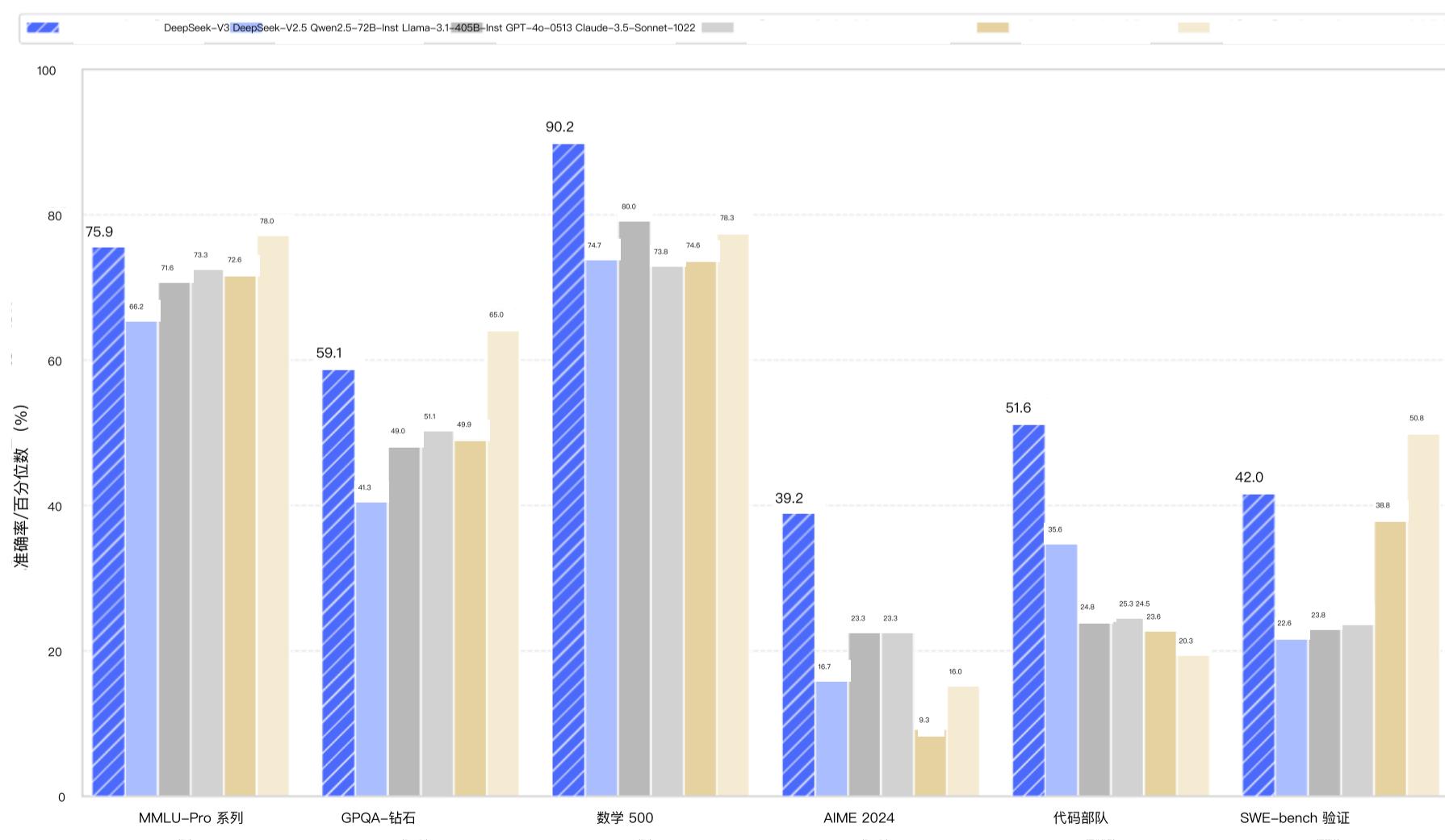


图 1 |DeepSeek-V3 及其对应产品的基准测试性能。

内容

1 介绍	4
2 建筑	6
2.1 基本架构	6
2.1.1 多头潜在注意力	7
2.1.2 具有辅助无损负载均衡的 DeepSeekMoE	8
2.2 多 Token 预测	10
3 基础设施	11
3.1 计算集群	11
3.2 培训框架	12
3.2.1 DualPipe 和 Computation–Communication 重叠	12
3.2.2 高效实现跨节点 All-to-All 通信	13
3.2.3 以最小的开销节省大量内存	14
3.3 FP8 训练	14
3.3.1 混合精度框架	15
3.3.2 提高量化和乘法的精度	16
3.3.3 低精度存储和通信	18
3.4 推理和部署	18
3.4.1 预填充	19
3.4.2 译码	19
3.5 硬件设计建议	20
3.5.1 通信硬件	20
3.5.2 计算硬件	20
4 训练前	22
4.1 数据构建	22
4.2 超参数	22
4.3 长上下文扩展	23
4.4 评估	24
4.4.1 评估基准	24
4.4.2 评估结果	25
4.5 讨论	26
4.5.1 用于多标记预测的消融研究	26
4.5.2 辅助无损平衡策略的消融研究	27

4.5.3 分批负载均衡 VS.顺序负载均衡	27
5 培训后	28
5.1 监督微调	28
5.2 强化学习	29
5.2.1 奖励模式	29
5.2.2 组相对策略优化	30
5.3 评估	30
5.3.1 评估设置	30
5.3.2 标准评估	32
5.3.3 开放式评估	33
5.3.4 DeepSeek–V3 作为生成奖励模型	33
5.4 讨论	34
5.4.1 从 DeepSeek–R1 蒸馏	34
5.4.2 自我奖励	34
5.4.3 多 Token 预测评估	35
6 结论、局限性和未来方向	35
A 贡献和鸣谢	45
B 用于低精度训练的消融研究	47
B.1 FP8 vs. BF16 训练	47
B.2 关于 Block–Wise 量化的讨论	47
16B Aux–Loss–Based 和 Aux–Loss–Free 型号的 C 专家专业模式	48

1. 引言

近年来，大型语言模型（LLMs）一直在经历快速迭代和演变（Anthropic, 2024 年;谷歌, 2024 年;OpenAI, 2024a），逐步缩小与通用人工智能（AGI）的差距。除了闭源模型之外，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（江 et al., 2023;Mistral, 2024 年）也取得了重大进展，努力缩小与闭源同行的差距。为了进一步突破开源模型能力的界限，我们扩展了我们的模型并引入了 DeepSeek-V3，这是一个大型专家混合（MoE）模型，具有 671B 参数，其中 37B 为每个令牌激活。

凭借前瞻性的视角，我们始终如一地努力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）进行高效推理，并采用 DeepSeekMoE（Dai et al., 2024）进行经济高效的训练。这两种架构已在 DeepSeekV2

（DeepSeek-AI, 2024c）中得到验证，证明了它们在实现高效训练和推理的同时保持稳健模型性能的能力。除了基本架构之外，我们还实施了两个额外的策略来进一步增强模型功能。首先，DeepSeek-V3 开创了一种用于负载均衡的辅助无损策略（Wang et al., 2024a），目的是最大限度地减少鼓励负载均衡对模型性能的不利影响。其次，DeepSeek-V3 采用了多标记预测训练目标，我们观察到该目标可以提高评估基准的整体性能。

为了实现高效的训练，我们支持 FP8 混合精度训练，并对训练框架进行全面优化。低精度训练已成为一种很有前途的高效训练解决方案（Dettmers et al., 2022;Kalamkar等人, 2019 年;Narang等人, 2017 年;Peng et al., 2023b），其发展与硬件能力的进步密切相关（Luo et al., 2024;Micikevicius 等人, 2022 年;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，并首次在超大规模模型上验证了其有效性。通过支持 FP8 计算和存储，我们实现了加速训练和减少 GPU 内存使用。对于训练框架，我们设计了 DualPipe 算法以实现高效的管道并行性，该算法具有更少的管道气泡，并通过计算–通信重叠隐藏了训练期间的大部分通信。这种重叠确保了，随着模型的进一步扩展，只要我们保持恒定的计算与通信比率，我们仍然可以跨节点雇用细粒度的专家，同时实现接近零的全对全通信开销。此外，我们还开发了高效的跨节点 all-to-all 通信内核，以充分利用 InfiniBand（IB）和 NVLink 带宽。此外，我们精心优化了内存占用，从而可以在不使用昂贵的张量并行性的情况下训练 DeepSeek-V3。

结合这些努力，我们实现了高培训效率。

在预训练期间，我们在 14.8T 高质量和多样化的 Token 上训练 DeepSeek-V3。预训练过程非常稳定。在整个训练过程中，我们没有遇到任何无法挽回的 loss suttage 或不得不回滚。接下来，我们对 DeepSeek-V3 进行两阶段上下文长度扩展。在第一阶段，最大上下文长度扩展到 32K，在第二阶段，它进一步扩展到 128K。在此之后，我们对 DeepSeek-V3 的基础模型进行后训练，包括监督微调（SFT）和强化学习（RL），以使其与人类偏好保持一致并进一步释放其潜力。在后期训练阶段，我们从 DeepSeekR1 系列模型中提炼出推理能力，同时小心翼翼地保持模型准确性之间的平衡

培训费用	训练前	上下文扩展	培训后	总
在 H800 GPU 小时数	2664K	119K	5K	\$5.328M \$0.238M 美元 \$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。

和世代长度。

我们根据一系列全面的基准测试来评估 DeepSeek-V3。尽管训练成本经济，但综合评估表明，DeepSeek-V3-Base 已成为目前可用的最强大的开源基础模型，尤其是在代码和数学方面。它的聊天版本还优于其他开源模型，并在一系列标准和开放式基准测试中实现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，通过我们对算法、框架和硬件的优化协同设计实现。在预训练阶段，在每万亿个令牌上训练 DeepSeek-V3 只需要 180K H800 GPU 小时，即在具有 2048 个 H800 GPU 的集群上训练 3.7 天。因此，我们的预训练阶段在不到两个月的时间里就完成了，花费了 2664K GPU 小时。结合 119K GPU 小时（用于上下文长度扩展）和 5K GPU 小时（用于后期训练），DeepSeek-V3 的完整训练成本仅为 2.788M GPU 小时。假设 H800 GPU 的租赁价格为每 GPU 小时 2 美元，我们的总训练成本仅为 5.576M 美元。请注意，上述费用仅包括 DeepSeek-V3 的官方训练费用，不包括与先前对架构、算法或数据进行研究和消融实验相关的费用。

我们的主要贡献包括：

架构：创新的负载均衡策略和训练目标

- 除了 DeepSeek-V2 的高效架构之外，我们还开创了一种用于负载均衡的辅助无损策略，该策略可以最大限度地减少因鼓励负载均衡而引起的性能下降。
- 我们研究了多标记预测（MTP）目标，并证明它对模型性能有益。它还可用于推理加速的推测解码。

训练前：迈向终极训练效率

- 我们设计了一个 FP8 混合精度训练框架，并首次在超大规模模型上验证了 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，实现了近乎全的计算通信重叠。这显著提高了我们的训练效率并降低了训练成本，使我们能够在不增加开销的情况下进一步扩大模型大小。
 - 我们以仅 2.664M H800 GPU 小时的经济成本，在 14.8T 令牌上完成了 DeepSeek-V3 的预训练，生成了目前最强的开源基础模型。
- 预训练后的后续训练阶段只需要 0.1M GPU 小时。

培训后：DeepSeek-R1 的知识提炼

- 我们引入了一种创新的方法，将 longChain-of-Thought (CoT) 模型的推理能力提炼出来，特别是从 DeepSeek R1 系列模型之一提炼到标准LLMs模型中，特别是 DeepSeek-V3。我们的管道优雅地整合了

将 R1 的验证和反射模式引入 DeepSeek-V3，并显著提高其推理性能。同时，我们还保持对 DeepSeek-V3 的输出样式和长度的控制。

核心评估结果摘要

- 知识：(1) 在 MMLU、MMLU-Pro 和 GPQA 等教育基准测试中，DeepSeek-V3 的表现优于所有其他开源模型，在 MMLU 上达到 88.5，在 MMLU-Pro 上达到 75.9，在 GPQA 上达到 59.1。它的性能可与 GPT-4o 和 Claude-Sonnet-3.5 等领先的闭源模型相媲美，缩小了该领域开源和闭源模型之间的差距。(2) 对于事实性基准测试，DeepSeek-V3 在 SimpleQA 和中文 SimpleQA 上的开源模型中都表现出卓越的性能。虽然它在英文事实知识 (SimpleQA) 方面落后于 GPT-4o 和 Claude-Sonnet-3.5，但它在中国事实知识 (Chinese SimpleQA) 方面超过了这些模型，凸显了它在中国事实知识方面的优势。
- 代码、数学和推理：(1) 在所有非 long-CoT 开源和闭源模型中，DeepSeek-V3 在数学相关基准测试中实现了最先进的性能。值得注意的是，它在特定基准测试（如 MATH-500）上的表现甚至优于 o1-preview，展示了其强大的数学推理能力。(2) 在编码相关任务方面，DeepSeek-V3 成为编码竞赛基准测试（如 LiveCodeBench）中表现最好的模型，巩固了其在该领域的领先模型地位。对于与工程相关的任务，虽然 DeepSeek-V3 的性能略低于 Claude-Sonnet-3.5，但它仍然大大超过所有其他模型，展示了其在各种技术基准上的竞争力。

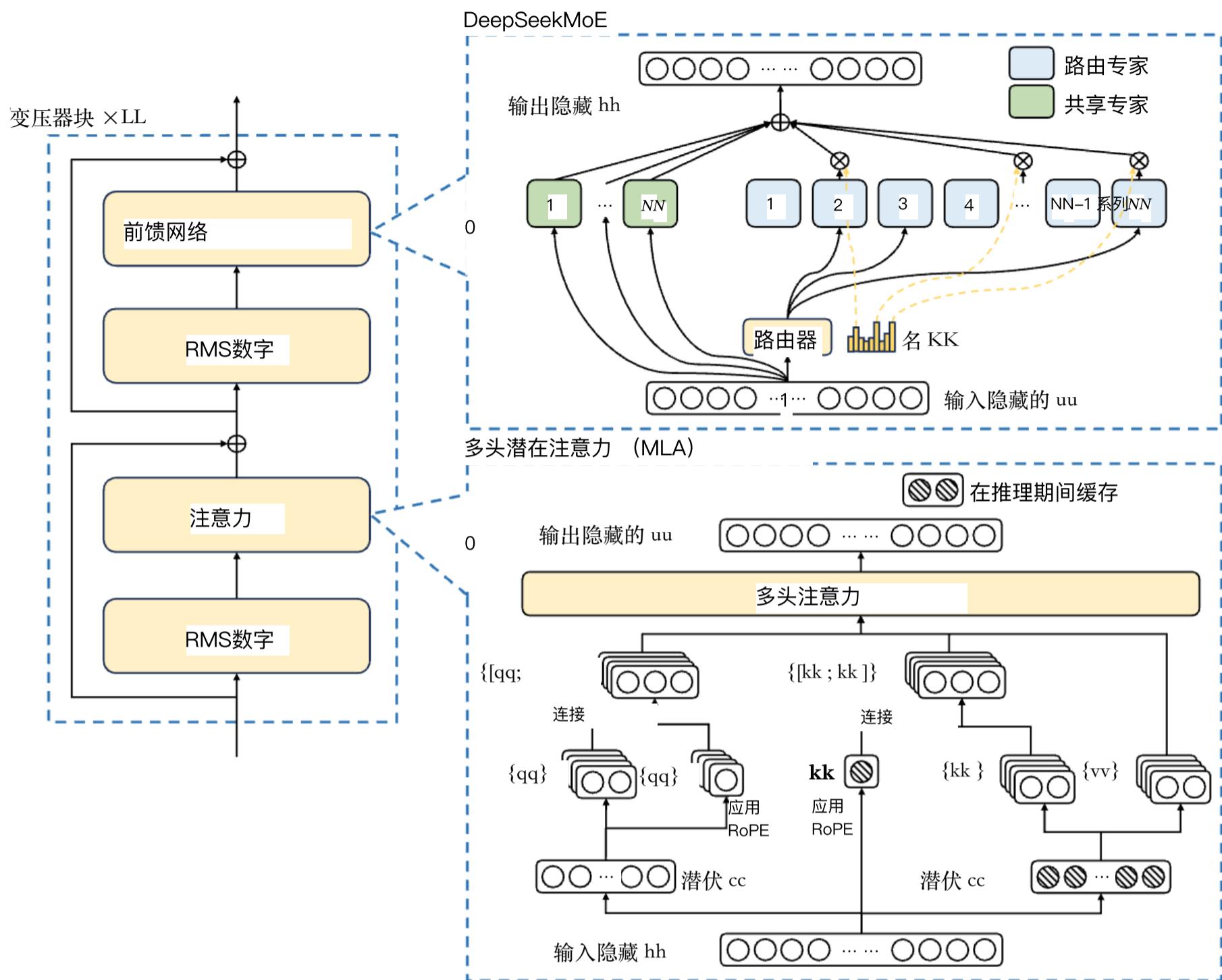
在本文的其余部分，我们首先详细阐述了我们的 DeepSeek-V3 模型架构（第 2 节）。随后，我们介绍了我们的基础设施，包括我们的计算集群、训练框架、对 FP8 训练的支持、推理部署策略以及我们对未来硬件设计的建议。接下来，我们描述了我们的预训练过程，包括训练数据的构建、超参数设置、longcontext 扩展技术、相关的评估以及一些讨论（第 4 节）。此后，我们讨论了我们在培训后所做的工作，包括监督微调 (SFT)、强化学习 (RL)、相应的评估和讨论（第 5 节）。最后，我们总结了这项工作，讨论了 DeepSeek-V3 的现有局限性，并提出了未来研究的潜在方向（第 6 节）。

2. 架构

我们首先介绍了 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，DeepSeekMoE (Dai et al., 2024) 用于经济训练。然后，我们提出了一个多标记预测 (MTP) 训练目标，我们观察到该目标可以提高评估基准的整体性能。对于其他未明确提及的小细节，DeepSeek-V3 遵循 DeepSeekV2 (DeepSeek-AI, 2024c) 的设置。

2.1. 基本架构

DeepSeek-V3 的基本架构仍在 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 还采用了 MLA 和 DeepSeekMoE，它们已经被 DeepSeek-V2 彻底验证。与 DeepSeek-V2 相比，一个例外是我们还引入了辅助无损失负载均衡



与 DeepSeek-V2 相比 图 2 |DeepSeek-V3 的基本架构图示。在 DeepSeek-V2 之后，我们采用 MLA 和 DeepSeekMoE 进行高效推理和经济训练。

策略 (Wang et al., 2024a) 来减轻因确保负载平衡而引起的性能下降。图 2 说明了 DeepSeek-V3 的基本架构，我们将在本节中简要回顾 MLA 和 DeepSeekMoE 的详细信息。

2.1.1. 多头潜在注意力

为了引起注意，DeepSeek-V3 采用了 MLA 架构。设 d 表示嵌入维度， n 表示注意力头的数量， d 表示每个头的维度， $h \in \mathbb{R}$ 表示给定注意力层的第 t 个标记的注意力输入。MLA 的核心是对注意力键和值的低秩联合压缩，以减少推理过程中的键值 (KV) 缓存：

$$c = W h, \quad (1)$$

$$[k; k; \dots; k] = k = W c, \quad (2)$$

$$k = \text{RoPE}(W h), \quad (3)$$

$$k = [k; k], \quad (4)$$

$$[v; v; \dots; v] = v = W c, \quad (5)$$

其中 $c \in R$ 是键和值的压缩潜在向量; $d (\ll dn)$ 表示 KV 压缩维度; $W \in R$ 表示下投影矩阵; $W, W \in R$ 分别是键和值的上投影矩阵; $W \in R$ 是用于生成带有旋转位置嵌入 (RoPE) 的解耦键的矩阵 (Su et al., 2024); $\text{RoPE}(\cdot)$ 表示应用 RoPE 矩阵的操作; 和 $[\cdot; \cdot]$ 表示连接。请注意, 对于 MLA, 在生成过程中只需要缓存蓝框向量 (即 c 和 k), 这会导致 KV 缓存显着减少, 同时保持与标准多头注意力 (MHA) 相当的性能 (Vaswani et al., 2017)。

对于注意力查询, 我们还执行了低秩压缩, 这可以减少训练期间的激活记忆:

$$c = W h, \quad (6)$$

$$[q; q; \dots; q] = q = W c, \quad (7)$$

$$[q; q; \dots; q] = q = \text{RoPE}(W c), \quad (8)$$

$$q = [q; q], \quad (9)$$

其中 $c \in R$ 是查询的压缩潜在向量; $d (\ll dn)$ 表示查询压缩维度; $W \in R, W \in R$ 分别是用于查询的下投影和上投影矩阵; $W \in R$ 是生成携带 RoPE 的解耦查询的矩阵。

最终, 注意力查询 (q)、键 (k) 和值 (v) 组合在一起, 产生最终的注意力输出 u :

$$o = \sum_{j=1}^t \text{Softmax} \left(\frac{qk}{\sqrt{d+d}} \right) v, \quad (10)$$

$$u = W[o; o; \dots; o], \quad (11)$$

其中 $W \in R$ 表示输出投影矩阵。

2.1.2. 具有辅助无损负载均衡的 DeepSeekMoE

DeepSeekMoE 的基本架构。对于前馈网络 (FFN), DeepSeek-V3 采用了 DeepSeekMoE 架构 (Dai et al., 2024)。与 GShard 等传统 MoE 架构 (Lepikhin et al., 2021) 相比, DeepSeekMoE 使用更细粒度的专家, 并将一些专家隔离为共享专家。让我们表示第 t 个标记的 FFN 输入, 我们计算 FFN 输出如下:

$$h = u + \sum_{i=1}^N \text{FFN}_i(u) + g_i \text{FFN}_i(u), \quad (12)$$

$$g = \begin{cases} \frac{g}{g}, & \text{if } j=1 \\ 0, & \text{otherwise} \end{cases}, \quad (13)$$

$$g_i = \begin{cases} s, & s \in \text{Topk}(\{s | 1 \leq j \leq N\}, K), \\ 0, & \text{otherwise} \end{cases}, \quad (14)$$

$$s = \text{乙状结肠 ue}, \quad (15)$$

其中 N 和 N 分别表示共享专家和路由专家的数量; $\text{FFN}(\cdot)$ 和 $\text{FFN}(\cdot)$ 分别表示第 i 个共享专家和第 i 个路由专家; K 表示激活的路由专家的数量; g_i 是第 i 个专家的门控值; s_{ij} 令牌到专家的关联性; e 是第 i 个路由专家的质心向量; $\text{Topk}(\cdot, K)$ 表示为第 t 个令牌和所有路由专家计算的关联性分数中包含 K 个最高分的集合。与 DeepSeek-V2 略有不同, DeepSeek-V3 使用 sigmoid 函数计算关联分数, 并在所有选定的关联分数之间应用归一化以生成门控值。

辅助无损负载均衡。对于 MoE 模型, 不平衡的专家负载将导致路由崩溃 (Shazeer et al., 2017), 并在专家并行的场景中降低计算效率。传统的解决方案通常依靠辅助损失 (Fedus et al., 2021; Lepikhin et al., 2021) 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 et al., 2024a)。为了在负载均衡和模型性能之间实现更好的权衡, 我们开创了一种辅助无损负载均衡策略 (Wang et al., 2024a) 来确保负载均衡。具体来说, 我们为每个专家引入了一个偏差项 b , 并将其添加到相应的亲和力分数 s 中, 以确定前 K 个路由:

$$g = \begin{cases} s, & s + b \in \text{Topk}(\{s + b | 1 \leq j \leq N\}, K), \\ 0, & \text{否则。} \end{cases} \quad (16)$$

请注意, 偏差项仅用于路由。门控值 (将乘以 FFN 输出) 仍然来自原始亲和力分数 s 。在训练期间, 我们会持续监控每个训练步骤的整个批次的 EA 负载。在每个步骤结束时, 如果相应的 EA 过载, 我们将把偏差项减少 γ , 如果其对应的 EA 过载, 则将其增加

γ 其对应的 EA 是否负载不足, 其中 γ 是一个称为 bias update speed 的超参数。通过动态调整, DeepSeek-V3 在训练期间保持平衡的 expert 负载, 并比通过纯辅助损失鼓励负载均衡的模型获得更好的性能。

互补序列辅助损失。虽然 DeepSeek-V3 主要依靠辅助无损策略进行负载均衡, 但为了防止任何单个序列内的极端不平衡, 我们还采用了互补序列平衡损失:

$$L = \alpha \sum_{i=1}^N f_i P_i, \quad (17)$$

$$f_i = \frac{N}{KT} - \sum_{t=1}^T \mathbf{1}_{s_i \in \text{Topk}(\{s_i | 1 \leq j \leq N\}, K)}, \quad (18)$$

$$s_i = \left| \frac{\sum_{j=1}^N s_{ij}}{N} \right|, \quad (19)$$

$$P_i = \frac{1}{T} \sum_{t=1}^T s_{it}, \quad (20)$$

其中余额因子 α 是一个超参数, 将为 DeepSeek-V3 分配一个非常小的值; $\mathbf{1}(\cdot)$ 表示指标函数; T 表示序列中的代币数量。序列平衡损失鼓励每个序列上的智能交易负载平衡。

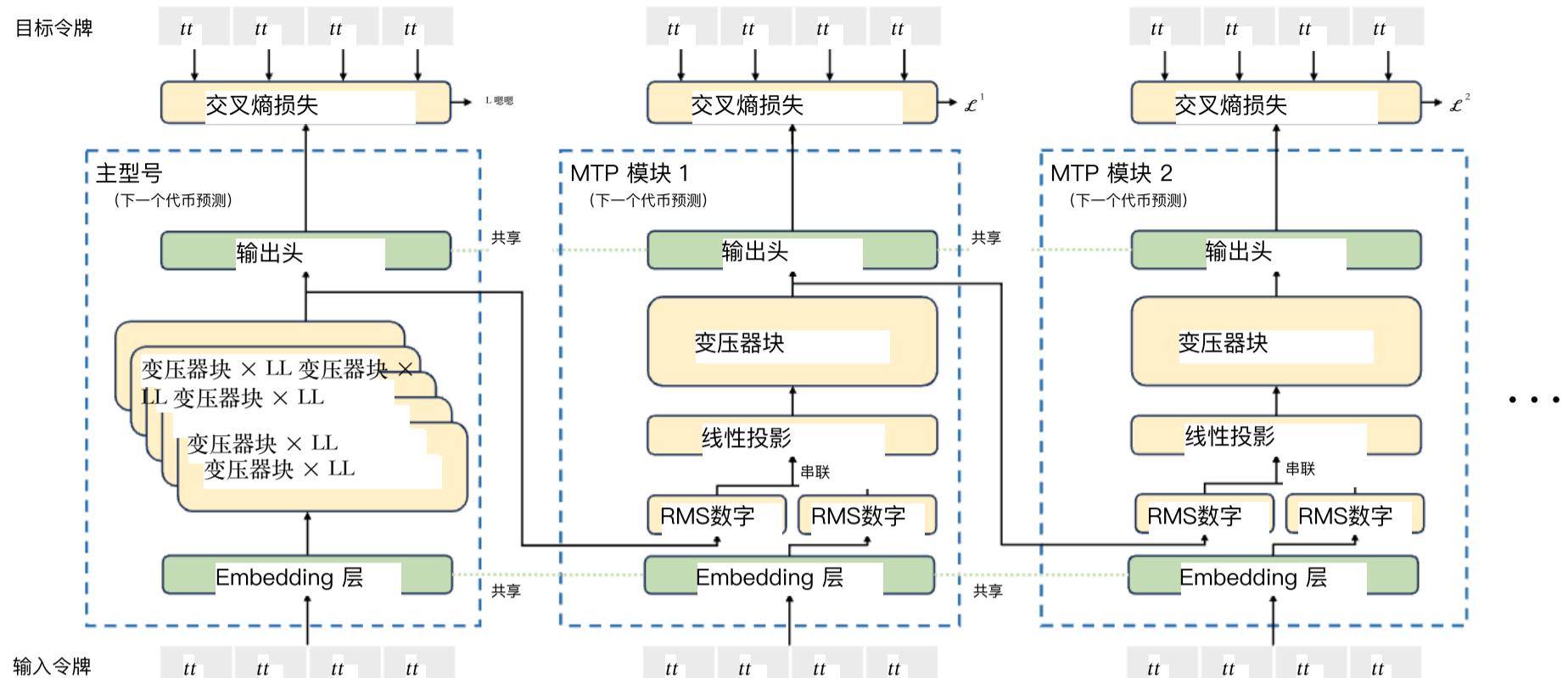


图 3 |我们的多代币预测（MTP）实施图示。我们保留了完整的因果链，用于预测每个深度的每个代币。

节点限制路由。与 DeepSeek-V2 使用的设备限制路由一样，DeepSeek-V3 也使用限制路由机制来限制训练期间的通信成本。简而言之，我们确保每个令牌最多发送到 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亲和力分数之和选择的。在这种约束下，我们的 MoE 训练框架几乎可以实现完全的计算-通信重叠。

没有 Token-Dropping。由于有效的负载均衡策略，DeepSeek-V3 在全程训练期间保持了良好的负载均衡。因此，DeepSeek-V3 在训练过程中不会丢弃任何 Token。此外，我们还实施了特定的部署策略来确保推理负载均衡，因此 DeepSeek-V3 在推理过程中也不会丢弃 Token。

2.2. 多 Token 预测

受 Gloeckle 等人（2024 年）的启发，我们研究并为 DeepSeek-V3 设定了一个多标记预测（MTP）目标，它将预测范围扩展到每个位置的多个未来标记。一方面，MTP 目标使训练信号致密，并可能提高数据效率。另一方面，MTP 可能使模型能够预先规划其表示，以便更好地预测未来标记。图 3 说明了我们对 MTP 的实现。与 Gloeckle 等人（2024 年）使用独立输出头并行预测 D 个额外标记不同，我们按顺序预测额外的标记，并在每个预测深度保留完整的因果链。我们在本节中介绍了 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 模块由一个共享嵌入层 $\text{Emb}(\cdot)$ 、一个共享输出头 $\text{OutHead}(\cdot)$ 、一个 Transformer 块 $\text{TRM}(\cdot)$ 和一个投影矩阵 $M \in \mathbb{R}$ 组成。对于第 i 个输入标记 t ，在第 k 个预测深度，我们首先将第 i 个标记在第 $(k-1)$ 个深度 h 的表示组合起来

$$_i \in \text{Rand} \text{ 第 } (i+k) \text{ 个标记 } \text{Emb}(t) \text{ 的嵌入} \in \mathbb{R}$$

使用线性投影:

$$h = M[RMSNorm(h_{\dots_i}); RMSNorm(Emb(t))], \quad (21)$$

其中模型给表的表示形式注意, k 对于每个 MTP 模块, 其嵌入层与主模型共享。组合的 h 作为第 k 个深度处 Transformer 模块的输入, 以生成当前深度 h 处的输出表示形式:

$$h = TRM(h), \quad (22)$$

其中 T 表示输入序列长度, 表示切片操作 (包括左边界和右边界)。最后, 获取输入, 共享输出头将计算第 k 个附加预测标记 $P \in R$ 的概率分布, 其中

V 是词汇表大小:

$$P = \text{OutHead}(h). \quad (23)$$

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

MTP 训练目标。

对于每个预测深度, 我们计算交叉熵损失 L :

$$L = \text{交叉熵}(P, t) = -\frac{1}{T} \sum_{i=2+k}^{\Sigma+1} \text{对数 } P[t], \quad (24)$$

其中 T 表示输入序列长度, t 表示第 i 个位置的真实标记, $P[t]$ 表示 t 的相应预测概率, 由第 k 个 MTP 模块给出。最后, 我们计算所有深度的 MTP 损失的平均值, 并将其乘以加权因子 λ 得到总体 MTP 损失 L , 作为 DeepSeek-V3 的附加训练目标:

$$L = \frac{\lambda}{D} \sum_{k=1}^D L_k. \quad (25)$$

推理中的 MTP。我们的 MTP 策略主要是为了提高主模型的性能, 所以在推理过程中, 我们可以直接丢弃 MTP 模块, 主模型可以独立正常地运行。此外, 我们还可以将这些 MTP 模块重新用于推测解码, 以进一步改善生成延迟。

3. 基础设施

3.1. 计算集群

DeepSeek-V3 在配备 2048 个 NVIDIA H800 GPU 的集群上进行训练。H800 集群中的每个节点都包含 8 个 GPU, 这些 GPU 在节点内通过 NVLink 和 NVSwitch 连接。在不同的节点之间, 利用 InfiniBand (IB) 互连来促进通信。

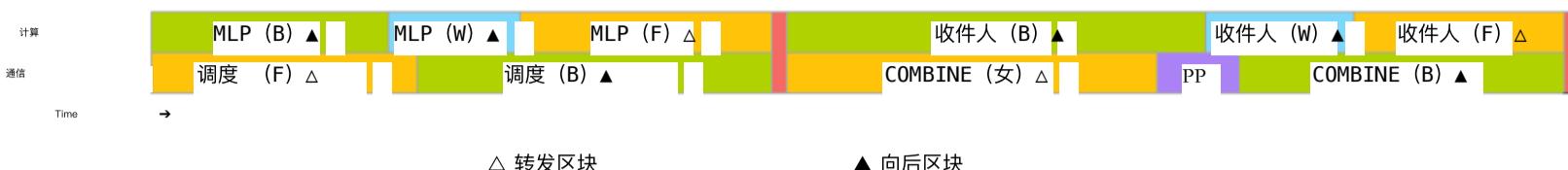


图 4 | 一对单独的向前和向后块的重叠策略 (transformer 块的边界未对齐)。橙色表示向前，绿色表示“向后输入”，蓝色表示“向后权重”，紫色表示 PP 通信，红色表示障碍。all-to-all 和 PP 通信都可以完全隐藏。

3.2. 培训框架

DeepSeek-V3 的训练由 HAI-LLM 框架提供支持，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 应用了跨越 8 个节点的 16 路管道并行 (PP) (Qi et al., 2023a)、64 路专家并行 (EP) (Lepikhin et al., 2021) 和 ZeRO-1 数据并行 (DP) (Rajbhandari et al., 2020)。

为了促进 DeepSeek-V3 的高效训练，我们实施了细致的工程优化。首先，我们设计了 DualPipe 算法以实现高效的管道并行性。与现有的 PP 方法相比，DualPipe 的管道气泡更少。更重要的是，它重叠了前向和后向过程中的计算和通信阶段，从而解决了跨节点专家并行性带来的沉重通信开销的挑战。其次，我们开发了高效的跨节点多对多通信内核，以充分利用 IB 和 NVLink 带宽，并节省专用于通信的流式多处理器 (SM)。最后，我们在训练过程中精心优化了内存占用，从而使我们能够在不使用昂贵的张量并行 (TP) 的情况下训练 DeepSeek-V3。

3.2.1. DualPipe 和 Computation–Communication 重叠

对于 DeepSeek-V3，跨节点专家并行引入的通信开销导致计算与通信的比率低下，约为 1: 1。为了应对这一挑战，我们设计了一种名为 DualPipe 的创新管道并行算法，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，而且还减少了管道气泡。

DualPipe 的关键思想是在一对单独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合。

nents: attention、all-to-all dispatch、MLP 和 all-to-all 组合。特别地，用于一个向后块，注意力和 MLP 都进一步分为两部分，向后用于输入，向后用于权重，就像在 ZeroBubble 中一样 (Qi 等人, 2023b)。此外，我们还有一个 PP 通信组件。如图 4 所示，对于一对向前和向后块，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 的比率。在这种重叠策略中，我们可以确保在执行过程中可以完全隐藏多对全和 PP 通信。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，同时从管道两端馈送微批次，并且很大一部分通信可以完全重叠。这种重叠还确保，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，我们仍然可以跨节点雇用细粒度的专家，同时实现接近零的全对全通信开销。

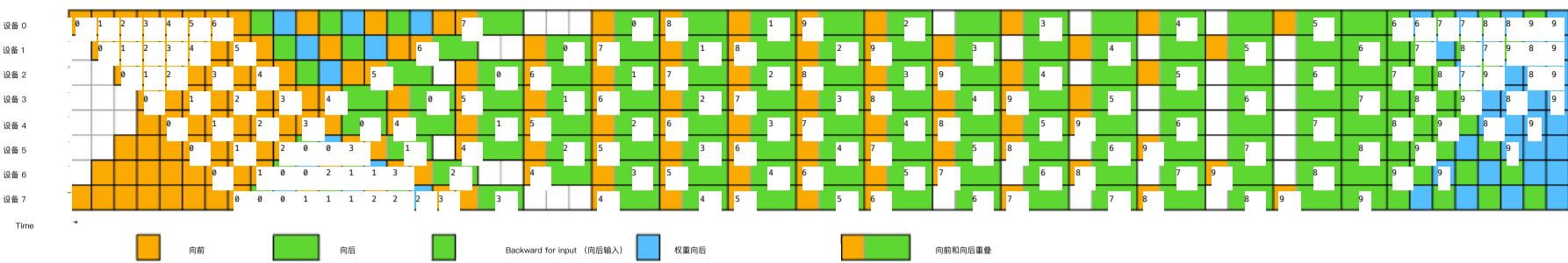


图 5 |20 个 PP 等级和 20 个两个方向的微批次调度示例 DualPipe。相反方向的微批次与正向的微批次对称，因此为了简单说明，我们省略了它们的批次 ID。由共享黑色边框包围的两个单元格在计算和通信方面相互重叠。

方法	泡沫	参数	激活
1F1B	$(PP - 1)(F + B)$	$1\times$	PP
ZB1P	$(PP - 1)(F + B - 2W)$	$1\times$	PP
DualPipe (我们的)	$(-1)(F\&B + B - 3W)$	$2\times$	$PP + 1$

表 2 |不同管道并行方法的管道气泡和内存使用情况的比较。F 表示正向数据块的执行时间，B 表示完整向后数据块的执行时间，W 表示“向后权重”数据块的执行时间，F&B 表示两个相互重叠的正向和向后数据块的执行时间。

此外，即使在没有沉重通信负担的更通用的场景中，DualPipe 仍然表现出效率优势。在表 2 中，我们总结了不同 PP 方法下的管道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 显着减少了管道气泡，同时仅将峰值激活内存增加了数倍。虽然 DualPipe 需要保留两个模型参数副本，但这并没有显着增加内存消耗，因为我们在训练过程中使用了较大的 EP 大小。与 Chimera (Li and Hoefler, 2021) 相比，DualPipe 只要求流水线阶段和微批次能被 2 整除，而不要求微批次能被流水线阶段整除。此外，对于 DualPipe，气泡和激活记忆都不会随着微批次数量的增加而增加。

3.2.2. 跨节点 All-to-All 通信的高效实现

为了保证 DualPipe 有足够的计算性能，我们定制了高效的跨节点 all-to-all 通信内核（包括调度和组合），以节省专用于通信的 SM 数量。内核的实现与我们集群的 MoE 门控算法和网络拓扑协同设计。具体来说，在我们的集群中，跨节点 GPU 与 IB 完全互连，节点内通信通过 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，我们将每个代币限制为最多 4 个节点，从而减少 IB 流量。对于每个代币，当做出路由决策时，它将首先通过 IB 传输到其目标节点上具有相同节点内索引的 GPU。一旦它到达目标节点，我们将努力确保它通过 NVLink 立即转发到托管目标专家的特定 GPU，而不会被随后到达的代币阻塞。通过这种方式，通过 IB 和 NVLink 的通信是完全重叠的，每个代币可以有效地为每个节点选择平均 3.2 个专家，而不会产生 NVLink 的额外开销。这意味着，尽管 DeepSeek-V3

虽然 DeepSeek-V3 在实践中只选择了 8 个路由专家，但它可以在保持相同的通信成本的情况下，将这个数字扩展到最多 13 个专家（4 个节点× 3.2 个专家/节点）。总体来看，在这样的通信策略下，只需 20 个 SM 就足以充分利用 IB 和 NVLink 的带宽。

详细地说，我们采用了 warp 专业化技术（Bauer et al., 2014）并将 20 个 SM 划分为 10 个通信通道。在调度过程中，(1) IB 发送，(2) IB 到 NVLink 转发和 (3) NVLink 接收由相应的 warp 处理。分配给每个通信任务的 warp 数量会根据所有 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 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，调度和组合内核都与计算流重叠，因此我们还考虑了它们对其他 SM 计算内核的影响。具体来说，我们采用定制的 PTX（并行线程执行）指令并自动调整通信块大小，这显着减少了 L2 缓存的使用和对其他 SM 的干扰。

3.2.3. 以最小的开销节省大量内存

为了减少训练期间的内存占用，我们采用了以下技术。

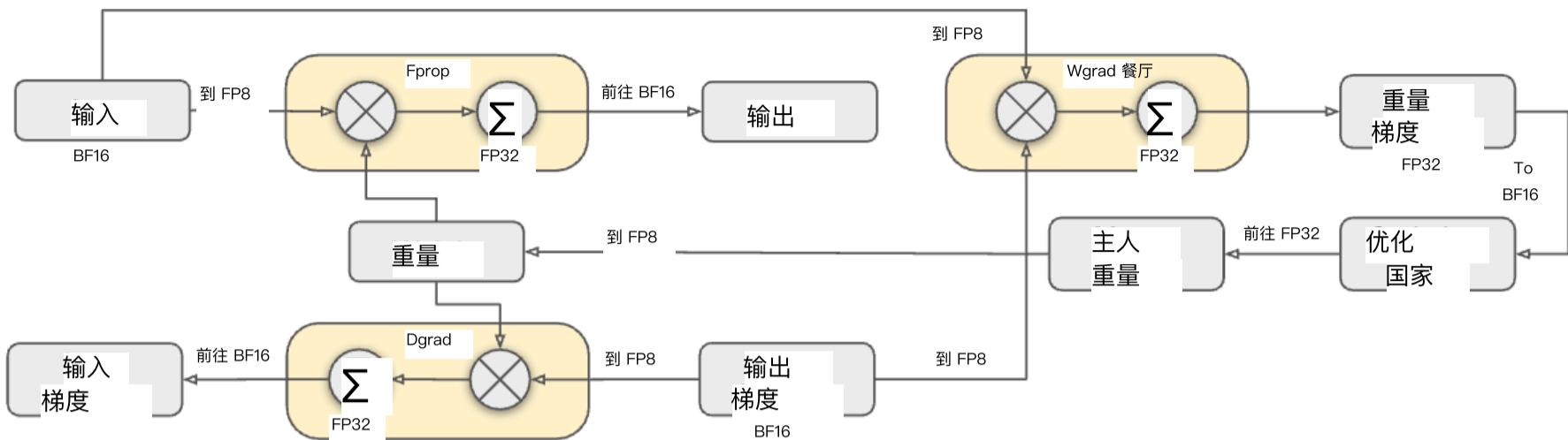
重新计算 RMSNorm 和 MLA 向上投影。我们重新计算所有 RMSNorm 运算 erations 和 MLA up-projections，因此无需持久存储其输出激活。此策略的开销很小，可显著降低存储激活的内存需求。

CPU 中的指数移动平均线。在训练过程中，我们保留模型参数的指数移动平均线（EMA），以便在学习率衰减后及早估计模型性能。EMA 参数存储在 CPU 内存中，并在每个训练步骤后异步更新。这种方法允许我们维护 EMA 参数，而不会产生额外的内存或时间开销。

用于多标记预测的共享嵌入和输出头。使用 DualPipe 策略，我们将模型最浅的层（包括嵌入层）和最深的层（包括输出头）部署在相同的 PP 等级上。这种安排实现了在 MTP 模块和主模型之间共享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 训练

受低精度训练最新进展的启发（Dettmers et al., 2022;Noune et al., 2022;Peng et al., 2023b），我们提出了一个利用 FP8 数据格式训练 DeepSeek-V3 的细粒度混合精度框架。虽然低精度训练前景广阔，但它往往受到激活、权重和梯度中存在异常值的限制（Fishman et al., 2024;He et al.;Sun et al., 2024）。尽管推理量化已经取得了重大进展（Frantar et al., 2022;Xiao et al., 2023），但证明低精度技术在大规模语言模型中成功应用的研究相对较少。



证明低精度技术在大规模语言模型中成功应用的研究相对较少 图 6 |具有 FP8 数据格式的整体混合精度框架。为澄清起见，仅说明了线性运算符。

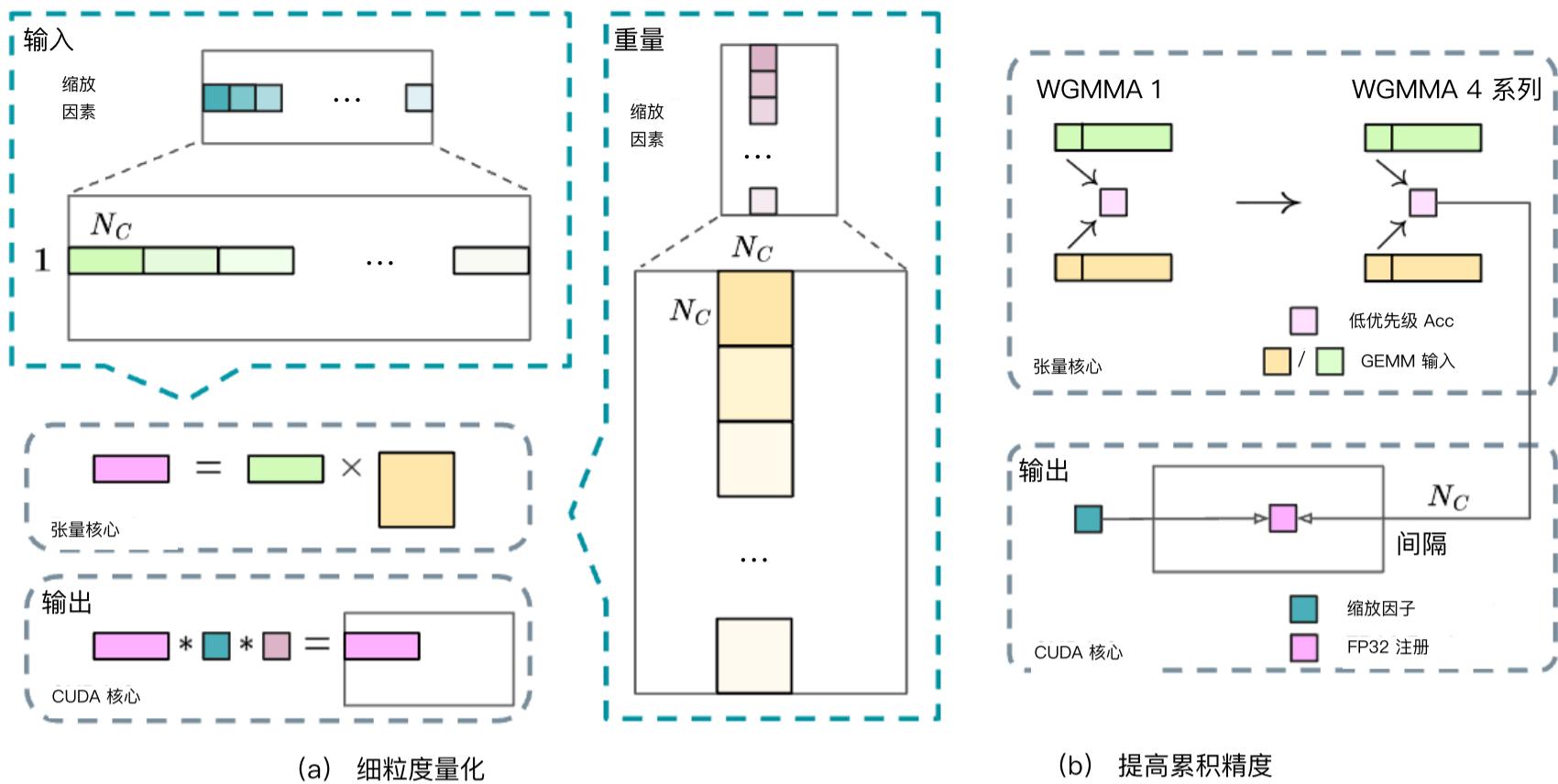
预训练 (Fishman et al., 2024)。为了应对这一挑战并有效扩展 FP8 格式的动态范围，我们引入了一种细粒度量化策略：使用 $1 \times$ 个 Nelements 进行分块分组，或使用 $N \times N$ 元素进行分块分组。在我们的提高精度累积过程中，相关的反量化开销在很大程度上得到了缓解。输出是激活向量矩阵乘法 (GEMM) 的关键方面。此外，为了进一步减少 MoE 训练中的内存和通信开销，我们在 FP8 中缓存和调度激活，同时在 BF16 中存储低精度优化器状态。我们在类似于 DeepSeek-V2-Lite 和 DeepSeekV2 的两个模型尺度上验证了所提出的 FP8 混合精度框架，训练了大约 1 万亿个代币（参见附录 B.1 中的更多详细信息）。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%，这一水平远在训练随机性的可接受范围内。

3.3.1. 混合精度框架

基于低精度训练中广泛采用的技术 (Kalamkar et al., 2019; Narang et al., 2017)，我们提出了一个用于 FP8 训练的混合精度框架。在这个框架中，大多数计算密度操作都是在 FP8 中进行的，而一些关键操作则战略性地保持其原始数据格式，以平衡训练效率和数值稳定性。整个框架如图 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，都以 FP8 精度实现。这些 GEMM 操作接受 FP8 张量作为输入，并在 BF16 或 FP32 中产生输出。如图 6 所示，与线性运算符相关的所有三个 GEMM，即 Fprop (前向传递)、Dgrad (激活向后传递) 和 Wgrad (权重向后传递)，都在 FP8 中执行。理论上，与原始的 BF16 方法相比，这种设计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，以便在向后传递中使用。这显着降低了内存消耗。

尽管 FP8 格式具有效率优势，但由于某些算子对低精度计算敏感，因此仍然需要更高的精度。此外，一些低成本算子还可以利用更高的精度，而总体训练成本的开销可以忽略不计。出于这个原因，经过仔细调查，我们保持了以下组件的原始精度 (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。这些有针对性的高精度保留确保了 DeepSeek-V3 的稳定训练动态。为了进一步保证数值稳定性，我们以更高的精度存储主权重、权重梯度和优化器状态。而



虽然图 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;为简单起见, 仅说明了 Fprop 。(b) 结合我们的量化策略, 我们通过以 $N=128$ 个元素 MMA 的间隔提升到 CUDA 核心来实现高精度累积, 从而提高了 FP8 GEMM 精度。

这些高精度组件会产生一些内存开销，在我们的分布式训练系统中，可以通过跨多个 DP 等级的高效分片来最大限度地减少它们的影响。

3.3.2. 通过量化和乘法提高精度

基于我们的混合精度 FP8 框架，我们引入了几种提高低精度训练精度的策略，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，而 FP8 格式的动态范围受到限制，因此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，输入分布与 FP8 格式的可表示范围保持一致 (Narang et al., 2017)。这种方法使低精度训练对激活异常值高度敏感，这会严重降低量化精度。为了解决这个问题，我们提出了一种细粒度量化方法，该方法在更精细的级别上应用缩放。如图 7 (a)、(1) 所示，对于激活，我们在 1×128 个平铺的基础上对元素进行分组和缩放（即，每 128 个通道的每个令牌）；(2) 对于权重，我们以 128×128 块为基础（即每 128 个输出通道每 128 个输入通道）对元素进行分组和缩放。这种方法确保量化过程可以通过根据更小的元素组调整比例来更好地容纳异常值。在附录 B.2 中，我们进一步讨论了当我们以与权重量化相同的方式在块的基础上对激活进行分组和缩放时，训练的不稳定性。

我们方法中的一个关键修改是沿 GEMM 操作的内部维度引入每组缩放因子。标准 FP8 GEMM 不直接支持此功能。但是，结合我们精确的 FP32 累积策略，它可以

它可以有效地实施。

值得注意的是，我们的细粒度量化策略与微缩放格式的思想高度一致 (Rouhani et al., 2023b)，而 NVIDIA 下一代 GPU (Blackwell 系列) 的 Tensor Core 已宣布支持具有较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 操作经常存在下溢问题，其准确性在很大程度上取决于高精度累积，这通常以 FP32 精度执行 (Kalamkar 等人, 2019 年; Narang 等人, 2017 年)。然而，我们观察到 FP8 GEMM 在 NVIDIA H800 GPU 上的累积精度仅限于保留 14 位左右，明显低于 FP32 累积精度。当内部维度 K 较大时 (Wortsman et al., 2023)，这个问题会变得更加明显，这是大规模模型训练中的典型场景，其中批量大小和模型宽度增加。以 $K = 4096$ 的两个随机矩阵的 GEMM 运算为例，在我们的初步测试中，Tensor Core 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少数 FP8 框架 (NVIDIA, 2024b) 的默认选项，严重限制了训练精度。

为了解决这个问题，我们采用了升级到 CUDA 核心以获得更高精度的策略 (Thakkar 等人, 2023 年)。该过程如图 7 (b) 所示。具体来说，在张量核上执行 MMA (矩阵乘法累加) 期间，使用有限的位宽累积中间结果。一旦达到 N_{is} 的间隔，这些部分结果将被复制到 CUDA 核心上的 FP32 寄存器，在那里执行全精度 FP32 累加。如前所述，我们的细粒度量化沿内部维度 K 应用每组缩放因子。这些缩放因子可以在 CUDA 核心上有效地乘以作为反量化过程，而额外的计算成本最小。

值得注意的是，此修改降低了单个 warpgroup 的 WGMMA (Warpgroup 级矩阵乘法累加) 指令发出率。但是，在 H800 架构上，两个 WGMMA 并发存在是典型的：当一个 warpgroup 执行提升操作时，另一个 warpgroup 能够执行 MMA 操作。这种设计使两个操作能够重叠，保持 Tensor Core 的高利用率。根据我们的实验，设置 $N=128$ 个元素，相当于 4 个 WGMMA，代表最小累积区间，可以显着提高精度，而不会引入大量开销。

指数上的尾数。与先前工作采用的混合 FP8 格式 (NVIDIA, 2024b; Peng 等人, 2023b; Sun 等人, 2019b) 相比，它在 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的细粒度量化策略，即平铺和块级缩放。通过对较小的元素组进行操作，我们的方法在这些分组元素之间有效地共享指数位，从而减轻了有限动态范围的影响。

在线量化。延迟量化用于张量量化框架 (NVIDIA, 2024b; Peng 等人, 2023b)，它保持了最大绝对值的历史

它维护了之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并简化框架，我们在线计算每个 1×128 激活图块或 128×128 权重块的最大绝对值。基于它，我们得出缩放因子，然后在线将激活或权重量化为 FP8 格式。

3.3.3. 低精度存储和通信

结合我们的 FP8 训练框架，我们通过将缓存的激活和优化器状态压缩为精度较低的格式，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 and 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，自然选择以 FP8 格式缓存激活，以便线性算子向后传递。但是，对于低成本、高精度的训练，需要特别考虑几个算子：

- (1) 注意力运算符之后的 Linear 输入。这些激活也用于注意力运算符的向后传递，这使得它对精度很敏感。我们采用专门用于这些激活的自定义 E5M6 数据格式。此外，这些激活将在向后传递中从 1×128 量化图块转换为 128×1 图块。为避免引入额外的量化误差，所有缩放因子都是四舍五入的，即 2 的积分幂。
- (2) SwiGLU 算子的 MoE 输入。为了进一步降低内存成本，我们缓存了 SwiGLU 算子的输入，并在向后传递中重新计算其输出。这些激活也通过我们的细粒度量化方法存储在 FP8 中，在内存效率和计算准确性之间取得平衡。

低精度通信。通信带宽是 MoE 模型训练中的关键瓶颈。为了缓解这一挑战，我们将 MoE 上投影之前的激活量化为 FP8，然后应用调度组件，这与 MoE 上投影中的 FP8 Fprop 兼容。就像注意力运算符之后的线性输入一样，这种激活的缩放因子是 2 的整数幂。类似的策略应用于 MoE 下投影之前的激活梯度。对于前向和后向组合组件，我们将它们保留在 BF16 中，以保持训练管道关键部分的训练精度。

3.4. 推理和部署

我们在 H800 集群上部署了 DeepSeek-V3，其中每个节点内的 GPU 都使用 NVLink 互连，集群中的所有 GPU 都通过 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，我们采用了以下部署策略，将预填充和解码阶段分开。

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) 和序列并行 (SP)，结合 8 路数据并行 (DP8)。其 4 的小 TP 大小限制了 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 全对全通信，我们使用与训练中相同的方法：首先通过 IB 跨节点传输令牌，然后通过 NVLink 在节点内 GPU 之间转发。特别是，我们对浅层中的密集 MLP 使用 1 路张量并行，以节省 TP 通信。

为了实现 MoE 部分不同专家之间的负载均衡，我们需要确保每个 GPU 处理大致相同数量的令牌。为此，我们引入了冗余专家部署策略，即复制高负载专家并冗余部署它们。高负载专家是根据在线部署期间收集的统计数据检测出来的，并定期调整（例如，每 10 分钟一次）。在确定了冗余专家的集合后，我们根据观察到的负载仔细地重新排列了节点内 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，尽可能地平衡 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，除了它托管的原始 8 个专家外，它还将额外托管 1 个冗余专家。

此外，在预填充阶段，为了提高吞吐量并隐藏 all-to-all 和 TP 通信的开销，我们同时处理两个具有相似计算工作负载的微批次，将一个微批次的注意力和 MoE 与

dispatch 和 combine 的 another。

最后，我们正在探索一种面向 expert 的动态冗余策略，其中每个 GPU 托管更多的 expert（例如，16 个 expert），但在每个推理步骤中只有 9 个会被激活。在每一层的 all-to-all 操作开始之前，我们动态计算全局最优路由方案。鉴于预填充阶段涉及大量计算，计算此路由方案的开销几乎可以忽略不计。

3.4.2. 解码

在解码过程中，我们将共享专家视为路由专家。从这个角度来看，每个代币在路由过程中会选择 9 个专家，其中共享专家被视为重负载专家，将始终被选中。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 只托管一名专家，64 个 GPU 负责托管冗余专家和共享专家。调度和组合部分的多对全通信是通过 IB 上的直接点对点传输进行的，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。

与预填充类似，我们根据来自我们在线服务的统计专家负载，定期确定特定时间间隔内的冗余专家集。但是，我们不需要重新排列专家，因为每个 GPU 只托管一名专家。我们还在探索解码的动态冗余策略。但是，这需要更仔细地优化算法，以计算全局最佳路由方案并与调度内核融合以减少开销。

此外，为了提高吞吐量并隐藏 all-to-all 通信的开销，我们还在探索在解码阶段同时处理两个计算工作负载相似的微 batch。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 batch 的注意力与另一个微 batch 的 dispatch+MoE+combine 重叠。在解码阶段，每个 expert 的 batch 大小相对较小（通常在 256 个 token 以内），瓶颈是内存访问而不是计算。由于 MoE 部分只需要加载一个 Expert 的参数，因此内存访问开销很小，因此使用较少的 SM 不会对整体性能产生显著影响。因此，为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 进行 dispatch+MoE+combine。

3.5. 硬件设计建议

基于我们对 all-to-all communication 和 FP8 训练方案的实施，我们向 AI 硬件厂商提出以下芯片设计建议。

3.5.1. 通信硬件

在 DeepSeek-V3 中，我们实现了计算和通信之间的重叠，以隐藏计算过程中的通信延迟。与串行计算和通信相比，这大大减少了对通信带宽的依赖。然而，当前的通信实现依赖于昂贵的 SM（例如，我们为此目的在 H800 GPU 中可用的 132 个 SM 中分配了 20 个），这将限制计算吞吐量。此外，使用 SM 进行通信会导致效率严重低下，因为张量核仍然完全未得到充分利用。

目前，SM 主要执行以下任务以进行 all-to-all 通信：

- 在 IB (InfiniBand) 和 NVLink 域之间转发数据，同时从单个 GPU 聚合发往同一节点内多个 GPU 的 IB 流量。
- 在 RDMA 缓冲区（已注册的 GPU 内存区域）和输入/输出缓冲区之间传输数据。
- 执行 all 对 all 组合的 reduce 操作。
- 在向 IB 和 NVLink 域中的多个专家传输分块数据期间管理精细的内存布局。

我们渴望看到未来的供应商开发硬件，从有价值的计算单元 SM 中卸载这些通信任务，作为 GPU 协处理器或网络协处理器，如 NVIDIA SHARP Graham 等人（2016 年）。此外，为了降低应用程序编程的复杂性，我们的目标是让这些硬件从计算单元的角度统一 IB（横向扩展）和 NVLink（纵向扩展）网络。借助这个统一的接口，计算单元可以通过提交基于简单基元的通信请求，在整个 IB-NVLink 统一域中轻松完成读、写、多播和缩减等操作。

3.5.2. 计算硬件

在 Tensor Core 中具有更高的 FP8 GEMM 累积精度。在当前的 Tensor Core 中 FP8 GEMM（General Matrix Multiply）采用 NVIDIA Hopper 架构的实现，采用定点累加，通过在加法前根据最大指数右移来对齐尾数乘积。我们的实验表明，它只使用最高的 14

我们的实验表明，在符号填充右移后，它只使用每个尾数乘积的最高 14 位，并截断超出此范围的位。但是，例如，要从 32 次 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提高 Tensor Core 中的累积精度以支持全精度累积，或者根据训练和推理算法的精度要求选择合适的累积位宽。这种方法可确保误差保持在可接受的范围内，同时保持计算效率。

支持 Tile-Wise 和 Block-Wise 量化。当前 GPU 仅支持每张量量化，缺乏对细粒度量化的原生支持，如我们的平铺和块级量化。在当前的实现中，当达到 Ninterval 时，部分结果将从 Tensor Core 复制到 CUDA 核心，乘以缩放因子，并添加到 CUDA 核心上的 FP32 寄存器中。尽管结合我们精确的 FP32 累积策略，反量化开销得到了显着缓解，但 Tensor Core 和 CUDA 核心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支持细粒度量化，使 Tensor Core 能够接收缩放因子并通过组缩放实现 MMA。这样，整个部分和累加和反量化可以直接在 Tensor Core 内部完成，直到产生最终结果，避免频繁的数据移动。

支持在线量化。尽管我们的研究证明了在线量化的有效性，但目前的实现难以有效支持在线量化。在现有流程中，我们需要从 HBM（高带宽内存）读取 128 个 BF16 激活值（之前计算的输出）进行量化，然后将量化的 FP8 值写回 HBM，然后再读取以进行 MMA。为了解决这种低效率问题，我们建议未来的芯片将 FP8 转换和 TMA（张量内存加速器）访问集成到单个融合操作中，以便在激活从全局内存到共享内存的传输过程中完成量化，避免频繁的内存读取和写入。我们还建议支持 warp 级转换指令以加速，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方法，将计算逻辑放置在 HBM 附近。在这种情况下，BF16 元件在从 HBM 读取到 GPU 时可以直接转换为 FP8，从而减少大约 50% 的片外内存访问。

支持转置 GEMM 操作。当前的架构使得矩阵转置与 GEMM 操作融合变得很麻烦。在我们的工作流程中，前向传递期间的激活被量化为 1×128 FP8 瓦片并存储。在向后传递期间，需要读出、去量化、转置、重新量化为 128×1 瓦片，并存储在 HBM 中。为了减少内存操作，我们建议未来的芯片能够在 MMA 操作之前从共享内存中直接转置读取矩阵，以实现训练和推理所需的精度。结合 FP8 格式转换和 TMA 访问的融合，这一增强功能将显著简化量化工作流程。

4. 训练前

4.1. 数据构建

与 DeepSeek-V2 相比，我们通过提高数学和编程样本的比例来优化预训练语料，同时将多语言覆盖范围扩展到英文和中文之外。此外，我们的数据处理管道经过改进，在保持语料多样性的同时最大限度地减少冗余。受 Ding 等人（2024 年）的启发，我们实现了数据完整性的文档打包方法，但在训练过程中没有加入跨样本注意力掩码。最后，DeepSeek-V3 的训练语料由我们的分词器中的 14.8T 高质量和多样化的词元组成。

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，我们还在 DeepSeek-V3 的预训练中纳入了 FIM 策略。具体来说，我们采用前缀-后缀-中间 (PSM) 框架来构建数据，如下所示：

```
<|fim_begin|> f<|fim_hole|> f<|fim_end|> f<|eos_token|>.
```

此结构作为预打包过程的一部分应用于文档级别。FIM 策略的应用速率为 0.1，与 PSM 框架一致。

DeepSeek-V3 的分词器采用字节级 BPE (Shibata et al., 1999)，具有 128K 词元的扩展词汇量。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词元。然而，当模型处理没有终端换行符的多行提示时，特别是对于小样本评估提示时，这个技巧可能会引入词元边界偏差 (Lundberg, 2023 年)。为了解决这个问题，我们在训练过程中随机拆分了一定比例的此类组合词元，这使模型暴露在更广泛的特殊情况下，并减轻了这种偏差。

4.2. 超参数

模型超参数。我们将 Transformer 层数设置为 61，隐藏维度设置为 7168。所有可学习的参数都以 0.006 的标准差随机初始化。在 MLA 中，我们将注意力头的数量设置为 n128，每个头的维度 d 设置为 128。KV 压缩维度 dis 设置为 512，查询压缩维度 d 设置为 1536。对于解耦的查询和 key，我们将每个头维度 d 设置为 64。我们将除前三层之外的所有 FFN 替换为 MoE 层。每个 MoE 层由 1 个共享专家和 256 个路由专家组成，其中每个专家的中间隐藏维度为 2048。在路由专家中，每个 Token 将激活 8 个 Expert，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 也在压缩的潜在向量之后使用了额外的 RMSNorm 层，并在宽度瓶颈处增加了额外的缩放因子。在这种配置下，DeepSeek-V3 总共包含 671B 个参数，其中每个 Token 激活了 37B。

训练超参数。我们采用 AdamW 优化器 (Loshchilov 和 Hutter, 2017 年)，超参数设置为 $\beta_1 = 0.9$ 、 $\beta_2 = 0.95$ 和 $\text{weight_decay} = 0.1$ 。我们在预训练期间将最大序列长度设置为 4K，并在 14.8T 令牌上预训练 DeepSeek-V3。至于

至于学习率调度，我们首先在前 2K 步中从 0 到 2.2×10 线性增加。然后，我们保持 2.2×10 的恒定学习率，直到模型消耗 10T 训练令牌。随后，我们按照余弦衰减曲线，将学习率逐渐衰减到 $\sqrt{10}$ 个 4.3T 令牌。在训练最后的 500B 令牌期间，我们在前 333B 令牌中保持 2.2×10 的恒定学习率，在剩余的 167B 令牌中切换到另一个 7.3×10 的恒定学习率。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 令牌的训练中，批量大小从 3072 逐渐增加到 15360，然后在剩余的训练中保持 15360。我们利用流水线并行性将模型的不同层部署在不同的 GPU 上，对于每一层，路由的专家将统一部署在属于 8 个节点的 64 个 GPU 上。至于节点限制的路由，每个 Token 最多会被发送到 4 个节点（即 $M = 4$ ）。为了实现辅助无损负载均衡，我们将前 14.3T Token 的偏差更新速度 γ 设置为 0.001，其余 500B Token 的偏差更新速度设置为 0.0。对于余额损失，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。前 10T Token 的 MTP 损失权重 λ 设置为 0.3，其余 4.8T Token 的 MTP 损失权重 λ 设置为 0.1。

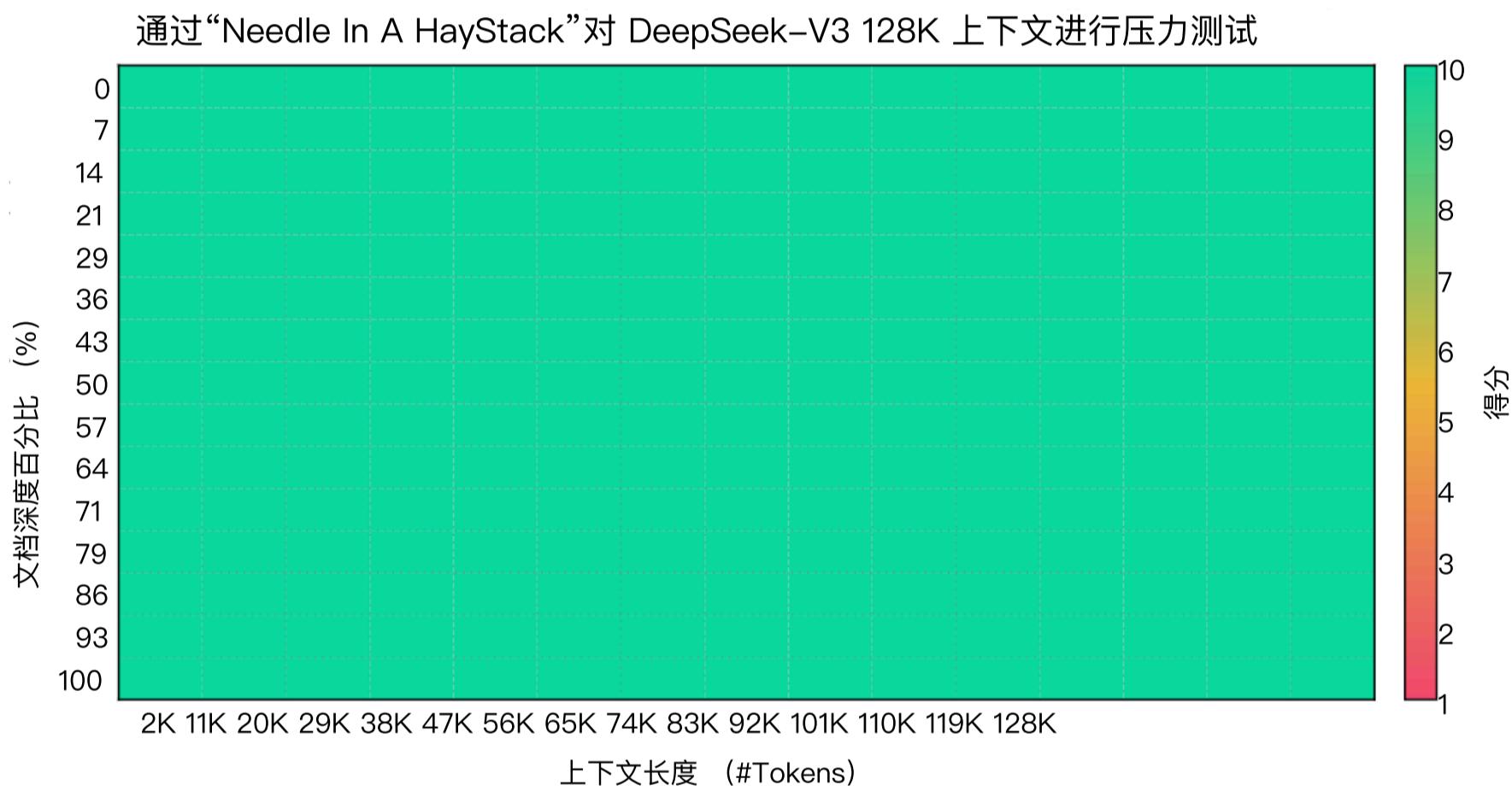


图 8 |“Needle In A Haystack” (NIAH) 测试的评估结果。DeepSeek-V3 在高达 128K 的所有上下文窗口长度上都表现良好。

4.3. 长上下文扩展

我们采用与 DeepSeek-V2 (DeepSeek-AI, 2024c) 类似的方法，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，并执行两个额外的训练阶段，每个阶段包括 1000 个步骤，以逐步将上下文窗口从 4K 扩展到 32K，然后扩展到 128K。YaRN 配置与 DeepSeek-V2 中使用的配置一致，专门用于解耦的共享密钥 k 。超参数在两个阶段保持相同，尺度 $s = 40$ ， $\alpha = 1$ ， $\beta = 32$ ，缩放因子

$0.1 \ln s + 1$ 。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。两个阶段的学习率都设置为 7.3×10 ，与预训练阶段的最终学习率相匹配。

通过这个两阶段的扩展训练，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 表明，DeepSeek-V3 在监督微调后，在“大海捞针”（NIAH）测试中取得了显着的性能，在高达 128K 的上下文窗口长度上表现出一致的鲁棒性。

4.4. 评估

4.4.1. 评估基准

DeepSeek-V3 的基础模型是在多语言语料库上进行预训练的，其中英语和中文占大多数，因此我们在一系列主要以英语和中文进行的基准测试以及多语言基准测试中评估其性能。我们的评估基于我们 HAI-LLM 框架中集成的内部评估框架。考虑的基准测试分类和列出如下，其中带下划线的基准测试为中文，双下划线的基准测试为多语言基准测试：

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

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

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

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

参考消歧数据集包括 CLUEWSC (Xu et al., 2020) 和 WinoGrande Sakaguchi et al. (2019)。

语言建模数据集包括 Pile (Gao et al., 2020)。

中国理解和文化数据集包括 CCPM (Li et al., 2021)。

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

代码数据集包括 HumanEval (Chen 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 et al., 2023)。请注意，AGIEval 包括英文和中文子集。

根据我们之前的工作 (DeepSeek-AI, 2024b, c)，我们对数据集采用基于困惑度的评估，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，并对 TriviaQA、NaturalQuestions、DROP、MATH、GSM8K、MGSM、HumanEval、MBPP、LiveCodeBench-Base、CRUXEval、BBH、AGIEval、CLUEWSC、CMRC 和 CMath 等数据集采用基于生成的评估。此外，我们对桩测试进行基于语言建模的评估，并使用每字节位数 (BPB) 作为指标，以保证使用不同分词器的模型之间的公平比较。

基准	# 镜头	DeepSeek–V2		Qwen2.5	LLaMA–3.1	DeepSeek–V3	
		Base	72B 底座	72B	405B 底座	405B	
架构 – MoE 密集	# 激活的参数	21B	72B	405B	37B	# 总参数 –	236B
英语	桩测试	–	0.606	0.638	0.542	0.548	72B
	BBH	3 镜头	78.8	79.8	82.9	87.5	405B
	MMLU	5 镜头	78.4	85.0	84.4	87.1	底座
	MMLU–Redux	5 镜头	75.6	83.2	81.3	86.2	671B
	MMLU–Pro 系列	5 镜头	51.4	58.3	52.8	64.4	
	DROP	3 镜头	80.4	80.6	86.0	89.0	
	ARC–易	25 发	97.6	98.4	98.4	98.9	
	ARC 挑战赛	25 发	92.2	94.5	95.3	95.3	
	HellaSwag 贽物	10 次拍摄	87.1	84.8	89.2	88.9	
	PIQA	0 次射击	83.9	82.6	85.9	84.7	
	维诺格兰德	5 镜头	86.3	82.3	85.2	84.9	
	RACE–中级	5 镜头	73.1	68.1	74.2	67.1	
	RACE–高	5 镜头	52.6	50.3	56.8	51.3	
Code	花絮 QA	5 镜头	80.0	71.9	82.7	82.9	
	自然问题	5 镜头	38.6	33.2	41.5	40.0	
	AGIEval	0 次射击	57.5	75.8	60.6	79.6	
	HumanEval	0 次射击	43.3	53.0	54.9	65.2	
	MBPP	3 镜头	65.0	72.6	68.4	75.4	
Math	LiveCodeBench–Base	3 镜头	11.6	12.9	15.5	19.4	
	CRUXEval–I	2 次射击	52.5	59.1	58.5	67.3	
	CRUXEval–O	2 次射击	49.8	59.9	59.9	69.8	
	GSM8K 型	8 镜头	81.6	88.3	83.5	89.3	
中文	MATH	4 发	43.4	54.4	49.0	61.6	
	MGSM	8 镜头	63.6	76.2	69.9	79.8	
	CMath	3 镜头	78.7	84.5	77.3	90.7	
	CLUEWSC 公司	5 镜头	82.0	82.5	83.0	82.7	
	C–评估	5 镜头	81.4	89.2	72.5	90.1	
	CMMU 公司	5 镜头	84.0	89.5	73.7	88.8	
多种语言	CMRC	1 次	77.4	75.8	76.0	76.3	
	C3	0 次射击	77.4	76.7	79.7	78.6	
	CCPM	0 次射击	93.0	88.5	78.6	92.0	
MMMLU–非英语	5 镜头	64.0	74.8	73.8	79.4		

表 3 |DeepSeek–V3–Base 与其他具有代表性的开源基础模型之间的比较。所有模型都在我们的内部框架中进行评估，并共享相同的评估设置。差距不超过 0.3 的分数被视为处于同一水平。DeepSeekV3–Base 在大多数基准测试中都取得了最佳性能，尤其是在数学和代码任务上。

4.4.2. 评估结果

在表 3 中，我们将 DeepSeek–V3 的基础模型与最先进的开源基础模型进行了比较，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评估所有这些模型，并确保它们共享相同的评估设置。请注意，由于过去几个月我们评估框架的变化，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 全面优于 DeepSeek–V2–Base 和 Qwen2.5 72B Base，在大多数基准测试中都超过了 LLaMA–3.1 405B Base，基本上成为最强的开源模型。

从更详细的角度来看，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，由于我们模型架构的改进、模型大小和训练令牌的扩展以及数据质量的增强，DeepSeek-V3-Base 实现了预期的明显更好的性能。(2) 与最先进的中国开源模型 Qwen2.5 72B Base 相比，DeepSeek-V3-Base 仅激活了一半的参数，也表现出显着的优势，尤其是在英语、多语言、代码和数学基准测试方面。在中文基准测试方面，除了中文多学科选择任务 CMMLU 外，DeepSeek-V3-Base 也表现出优于 Qwen2.5 72B 的性能。(3) 与最大的开源模型 LLaMA-3.1 405B Base 相比，DeepSeek-V3-Base 在多语言、代码和数学基准测试上也表现出更好的性能。在中英文基准测试方面，DeepSeek-V3-Base 表现出具有竞争力或更好的性能，在 BBH、MMLU 系列、DROP、C-Eval、CMMLU 和 CCPM 上表现尤好。

由于我们高效的架构和全面的工程优化，DeepSeekV3 实现了极高的训练效率。在我们的训练框架和基础设施下，在每万亿个令牌上训练 DeepSeek-V3 只需要 180K H800 GPU 小时，这比训练 72B 或 405B 密集模型便宜得多。

基准 (度量)	# 镜头	小 MoE	小 MoE	大 MoE	大 MoE	基线 w/ MTP	基线 w/ MTP			
		# 激活的参数 – 2.4B	2.4B	20.9B	20.9B	# 总参数 – 15.7B	15.7B	228.7B	228.7B	# 训练代币 – 1.33T
桩测试	–	0.729	0.729	0.658	0.657					
BBH	3 镜头	39.0	41.4	70.0	70.7					
MMLU	5 镜头	50.0	53.3	67.5	66.6					
DROP	1 次	39.2	41.3	68.5	70.6					
花絮 QA	5 镜头	56.9	57.7	67.0	67.3					
自然问题	5 镜头	22.7	22.3	27.2	28.5					
HumanEval	0 次射击	20.7	26.8	44.5	53.7					
MBPP	3 镜头	35.8	36.8	61.6	62.2					
GSM8K 型	8 镜头	25.4	31.4	72.3	74.0					
MATH	4 发	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如一地提高了模型性能。

4.5. 讨论

4.5.1. 用于多标记预测的消融研究

在表 4 中，我们显示了 MTP 策略的消融结果。具体来说，我们在不同规模的两个基线模型之上验证了 MTP 策略。在小规模上，我们在 1.33T 代币上训练了一个基线 MoE 模型，该模型包含 15.7B 总参数。在大规模上，我们训练了一个基线 MoE 模型，该模型在 540B 代币上包含 228.7B 总参数。最重要的是，在保持训练数据和其他架构相同的情况下，我们在它们上面附加了一个 1 深度的 MTP 模块，并使用 MTP 策略训练两个模型进行比较。请注意，在推理过程中，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。从表中可以观察到，MTP 策略在大多数评估基准上始终如一地提高了模型性能。

基准 (度量)	# 镜头	小 MoE Aux–Loss–Free	小 MoE Aux–Loss–Free	大 MoE Aux–Loss–Free	大 MoE Aux–Loss–Based	基于 Aux–Loss–Based	Aux–Loss–Free
# 激活的参数 – 2.4B 578B	2.4B 578B	20.9B # 总参数 – 15.7B 228.7B	20.9B 15.7B 228.7B	228.7B # 训练代币 – 1.33T 1.33T			
桩测试	–	0.727	0.724	0.656	0.652		
BBH	3 镜头	37.3	39.3	66.7	67.9		
MMLU	5 镜头	51.0	51.8	68.3	67.2		
DROP	1 次	38.1	39.0	67.1	67.1		
花絮QA	5 镜头	58.3	58.5	66.7	67.7		
自然问题	5 镜头	23.2	23.4	27.1	28.1		
HumanEval	0 次射击	22.0	22.6	40.2	46.3		
MBPP	3 镜头	36.6	35.8	59.2	61.2		
GSM8K 型	8 镜头	27.1	29.6	70.7	74.5		
MATH	4 发	10.9	11.1	37.2	39.6		

表 5 | 辅助无损平衡策略的消融结果。与纯粹基于辅助损失的方法相比，辅助无损策略在大多数评估基准上始终保持更好的模型性能。

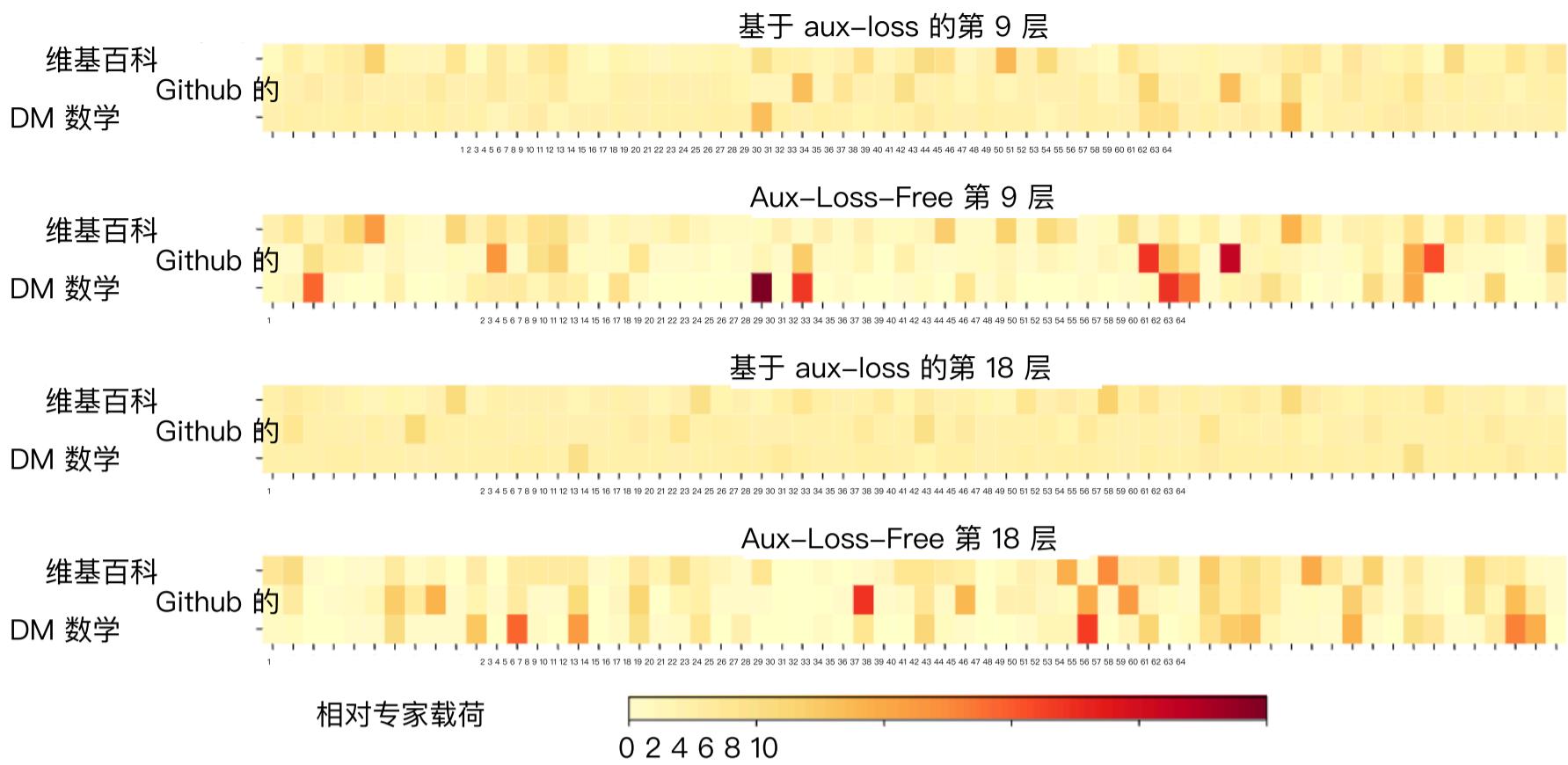
4.5.2. 辅助无损平衡策略的消融研究

在表 5 中，我们显示了辅助无损失平衡策略的消融结果。我们在不同尺度的两个基线模型之上验证了这一策略。在小规模上，我们训练了一个基线 MoE 模型，该模型在 1.33T 代币上包含 15.7B 总参数。在大规模上，我们训练了一个基线 MoE 模型，该模型在 578B 代币上包含 228.7B 总参数。这两个基线模型都纯粹使用辅助损失来鼓励负载均衡，并使用具有 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。在这两个基线模型之上，保持训练数据和其他架构相同，我们去除了所有辅助损失，并引入了辅助无损失平衡策略进行比较。从表中可以观察到，辅助无损失策略在大多数评估基准上始终如一地实现了更好的模型性能。

4.5.3. 分批负载均衡 VS. 顺序负载均衡

辅助无损失平衡和序列辅助损失之间的主要区别在于它们的平衡范围：批次与序列。与序列辅助损失相比，批次平衡施加了更灵活的约束，因为它不会在每个序列上强制执行域内平衡。这种灵活性使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 Pile 测试集中不同域上基于 16B 辅助损失的基线和 16B 辅助无损模型的专家负载。如图 9 所示，我们观察到辅助无损失模型如预期的那样表现出更大的专家专业化模式。

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批次级辅助损失，它鼓励在每个训练批次上而不是每个序列上进行负载均衡。实验结果表明，当达到相似水平的批次级负载均衡时，批次级辅助损失也可以实现与辅助无损方法相似的模型性能。具体来说，在我们用 1B MoE 模型的实验中，验证损失是：2.258（使用序列式辅助损失）、2.253（使用辅助无损方法）和 2.253（使用批次级



253 (使用批次 图 9 |辅助无损和基于辅助损耗模型的专家载荷在桩测试集中的三个域上。辅助无损模型比基于辅助损失的模型显示出更大的专家专业化模式。相对专家载荷表示实际专家载荷与理论上平衡的专家载荷之间的比率。由于空间限制，我们只提供两层的结果作为示例，所有层的结果在附录 C 中提供。

辅助损失)。我们还在 3B MoE 模型上观察到类似的结果：使用序列辅助损失的模型实现了 2.085 的验证损失，而使用辅助无损方法或批量辅助损失的模型实现了相同的 2.080 验证损失。

此外，尽管分批负载均衡方法显示出一致的性能优势，但它们在效率方面也面临两个潜在的挑战：(1) 某些序列或小批量内的负载不平衡，以及 (2) 推理过程中域偏移引起的负载不平衡。第一个挑战自然是由我们的训练框架解决的，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大规模。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，如第 3.4 节所述，具有冗余专家部署，以克服它。

5. 培训后

5.1. 监督微调

我们管理我们的指令调优数据集，包括跨多个域的 1.5M 实例，每个域都采用根据其特定要求量身定制的不同数据创建方法。

推理数据。对于与推理相关的数据集，包括专注于数学、代码竞争问题和逻辑谜题的数据集，我们利用内部 DeepSeek-R1 模型生成数据。具体来说，虽然 R1 生成的数据表现出很高的准确性，但它存在思考过度、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精度和常规格式推理数据的清晰度和简洁性。

为了建立我们的方法，我们首先使用组合的监督式微调（SFT）和强化学习（RL）训练管道，为特定领域（例如代码、数学或一般推理）开发一个量身定制的专家模型。该专家模型用作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：第一种以 <问题，原始响应> 的格式将问题与其原始响应耦合，而第二种将系统提示与问题和 R1 响应<系统提示、问题、R1 响应>格式的 R1 响应相结合。

系统提示经过精心设计，包括指导模型生成具有丰富反射和验证机制的响应的指令。在 RL 阶段，该模型利用高温采样生成响应，这些响应集成了来自 R1 生成数据和原始数据的模式，即使没有明确的系统提示也是如此。经过数百个 RL 步骤后，中间 RL 模型学习合并 R1 模式，从而从战略上提高整体性能。

完成 RL 训练阶段后，我们实施拒绝抽样，为最终模型策划高质量的 SFT 数据，其中专家模型用作数据生成源。这种方法可确保最终训练数据保留 DeepSeek–R1 的优势，同时产生简洁有效的响应。

非推理数据。对于非推理数据，例如创意写作、角色扮演和简单的问答，我们利用 DeepSeek–V2.5 来生成响应，并招募人类注释者来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 进行了两个 epoch 的微调，使用余弦衰减学习率调度，从 5×10 开始，逐渐减少到 1×10 。在训练期间，每个序列都是从多个样本打包的。但是，我们采用样本掩码策略来确保这些样本保持隔离和相互不可见。

5.2. 强化学习

5.2.1. 奖励模型

我们在 RL 流程中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，某些数学问题具有确定性的结果，我们要求模型以指定格式（例如，在框中）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，我们确保了更高水平的可靠性，因为这种方法可以抵抗操纵或利用。

基于模型的 RM。对于具有自由格式真实答案的问题，我们依靠奖励模型来确定响应是否与预期的真实数据匹配。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。

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 检查点训练的。为了提高其可靠性，我们构建了偏好数据，该数据不仅提供最终奖励，还包括导致奖励的思维链。这种方法有助于降低特定任务中奖励黑客攻击的风险。

5.2.2. 组相对策略优化

与 DeepSeek–V2 (DeepSeek–AI, 2024c) 类似，我们采用组相对策略优化 (GRPO) (Shao et al., 2024)，它放弃了通常与策略模型大小相同的批评模型，而是从组分数中估计基线。具体来说，对于每个问题 q ，GRPO 从旧的策略模型中抽取一组输出 $\{o_1, o_2, \dots, o_n\}$

然后优化策略模型 π 通过最大化以下目标：

$$\text{JGRPO}(\theta) = E[q \sim P(Q), \{o_i\}_{i=1}^G \sim \pi(o|q)] \min_{\pi} \frac{\pi(o|q)}{\pi_{\text{老}}(o|q)} A, \text{ 夹子 } \frac{\pi(o|q)}{\pi_{\text{老}}(o|q)}, 1 - \varepsilon, 1 + \varepsilon A - \beta D \quad \pi \parallel \pi, \quad (26)$$

$$D = \frac{\pi(o|q)}{\pi_{\text{老}}(o|q)} - \text{对数} \frac{\pi(o|q)}{\pi_{\text{老}}(o|q)} - 1, \quad (27)$$

其中 ε 和 β 是超参数； π 是参考模型； A 是优势，从对应于每组内输出的奖励 $\{r_1, r_2, \dots, r_n\}$ 得出：

$$A = \frac{r - \text{mean}(\{r_1, r_2, \dots, r_n\})}{\text{std}(\{r_1, r_2, \dots, r_n\})}, \quad (28)$$

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), CSimpleQA (He et al., 2024), SWE–Bench Verified (OpenAI, 2024d), Aider¹, LiveCodeBench (Jain et al., 2024) (questions from August 2024 to November 2024), Codeforces², Chinese National High School Mathematics Olympiad (CNMO 2024), 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>

详细的评估配置。对于包括 MMLU、DROP、GPQA 和 SimpleQA 在内的标准基准测试，我们采用 simple-evals 框架中的评估提示。我们在零镜头设置中对 MMLU–Redux 使用零评估提示格式 (Lin, 2024)。对于其他数据集，我们遵循其原始评估协议和数据集创建者提供的默认提示。对于代码和数学基准测试，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 LiveCodeBench 上的模型性能，该数据是在 2024 年 8 月至 2024 年 11 月期间收集的。Codeforces 数据集是使用竞争对手的百分比来衡量的。SWE–Bench 验证使用无代理框架进行评估 (Xia et al., 2024)。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 在 0.7 的温度下进行评估，结果是 16 次运行的平均结果，而 MATH–500 采用贪婪解码。我们允许所有模型为每个基准输出最多 8192 个令牌。

基准		DeepSeek	DeepSeek	Qwen2.5	LLaMA–3.1	Claude–3.5–	GPT–4o	DeepSeek V2–0506
		V2.5–0905	72B–Inst.405B–Inst.	十四行诗–1022	0513	V3		
架构 MoE 密集 稀疏 – – MoE # 激活参数 21B 21B 72B 405B – – 37B # 总参数 236B 236B 72B 405B – – 671B								
英语	MMLU	78.2	80.6	85.3	88.6	88.3	87.2	88.5
	MMLU–Redux	77.9	80.3	85.6	86.2	88.9	88.0	89.1
	MMLU–Pro 系列	58.5	66.2	71.6	73.3	78.0	72.6	75.9
	DROP	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–钻石	35.3	41.3	49.0	51.1	65.0	49.9	59.1
	简单质量保证	9.0	10.2	9.1	17.1	28.4	38.2	24.9
	框架	66.9	65.4	69.8	70.0	72.5	80.5	73.3
Code	LongBench 版本 2	31.6	35.4	39.4	36.1	41.0	48.1	48.7
	HumanEval–Mul	69.3	77.4	77.3	77.2	81.7	80.5	82.6
	LiveCodeBench 函数	18.8	29.2	31.1	28.4	36.3	33.4	40.5
	LiveCodeBench 函数	20.3	28.4	28.7	30.1	32.8	34.2	37.6
	代码部队	17.5	35.6	24.8	25.3	20.3	23.6	51.6
	SWE 验证	–	22.6	23.8	24.5	50.8	38.8	42.0
	Aider–编辑	60.3	71.6	65.4	63.9	84.2	72.9	79.7
Math	Aider–Polyglot (艾德多语者)	–	18.2	7.6	5.8	45.3	16.0	49.6
	AIME 2024	4.6	16.7	23.3	23.3	16.0	9.3	39.2
	数学 500	56.3	74.7	80.0	73.8	78.3	74.6	90.2
	CNMO 2024	2.8	10.8	15.9	6.8	13.1	10.8	43.2
中文	CLUEWSC 公司	89.9	90.4	91.4	84.7	85.4	87.9	90.9
	C–评估	78.6	79.5	86.1	61.5	76.7	76.0	86.5
	C–SimpleQA	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>

5.3.2. 标准评估

表 6 显示了评估结果，表明 DeepSeek-V3 是性能最佳的开源模型。此外，它与 GPT-4o 和 Claude-3.5-Sonnet 等前沿闭源模型具有竞争力。

英语基准测试。MMLU 是公认的基准测试，旨在评估大型语言模型在不同知识领域和任务中的性能。DeepSeek-V3 表现出有竞争力的性能，与 LLaMA3.1-405B、GPT-4o 和 Claude-Sonnet 3.5 等顶级模型相当，同时性能明显优于 Qwen2.5 72B。此外，DeepSeek-V3 在更具挑战性的教育知识基准测试 MMLU-Pro 中表现出色，紧随 Claude-Sonnet 3.5 之后。在 MMLU-Redux（带有校正标签的 MMLU 的改进版本）上，DeepSeek-V3 超越了同行。此外，在博士级评估台 GPQA-Diamond 上，DeepSeek-V3 取得了显著的成绩，仅次于 Claude 3.5 Sonnet，表现明显优于所有其他竞争对手。

在 DROP、LongBench v2 和 FRAMES 等长上下文理解基准测试中，DeepSeek-V3 继续展示其作为顶级模型的地位。它在 DROP 的 3 次设置中取得了令人印象深刻的 91.6 F1 分数，优于该类别中的所有其他模型。在 FRAMES（一个需要超过 100k 个令牌上下文的问答的基准测试）上，DeepSeek-V3 紧随 GPT-4o 之后，同时性能明显优于所有其他模型。这表明 DeepSeek-V3 在处理超长上下文任务方面的强大能力。DeepSeek-V3 在处理超长上下文任务方面的出色性能进一步验证了其在 LongBench v2 上的最佳性能，该数据集在 DeepSeek V3 发布前几周发布。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 分配了更多的训练令牌来学习中文知识，从而在 C-SimpleQA 上实现了卓越的性能。在指令跟踪基准测试中，DeepSeek-V3 的性能明显优于其前身 DeepSeek-V2 系列，突出了其理解和遵守用户定义的格式约束的能力的改进。

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.

[C 重试](#) [i 错误原因](#)

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.

[返回顶部](#)

型	竞技场-困难	羊驼评估 2.0
深度搜索-V2.5-0905 81.2 49.1 LLaMA-3.1 405B	76.2 50.5 Qwen2.5-72B-指令 69.3 40.5	
GPT-4o-0513 52.0	80.4 51.1 克劳德-索内特-3.5-1022	85.2
深度搜索-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 lengthcontrolled win rate 作为指标。中文基准测试。Qwen 和 DeepSeek 是两个具有代表性的模型系列，对中文和英文都有强大的支持。在事实基准测试中文 SimpleQA 中，DeepSeekV3 比 Qwen2.5-72B 高出 16.4 分，尽管 Qwen2.5 是在更大的语料库上训练的，该语料库泄露了 18T 令牌，这比 DeepSeek-V3 预训练的 14.8T 令牌多了 20%。

在中文教育知识评估的代表性基准 C-Eval 和 CLUEWSC（中文 Winograd 模式挑战赛）上，DeepSeek-V3 和 Qwen2.5-72B 表现出相似的性能水平，表明这两个模型都很好地优化了具有挑战性的中文推理和教育任务。

5.3.3. 开放式评估

除了标准基准外，我们还使用LLMs作为裁判来评估我们在开放式生成任务上的模型，结果如表 7 所示。具体来说，我们遵循 AlpacaEval 2.0 (Dubois 等人, 2024 年) 和 Arena-Hard (Li 等人, 2024a) 的原始配置，它们利用 GPT-4-Turbo-1106 作为评判进行成对比较。在 Arena-Hard 上，DeepSeek-V3 与基线 GPT-4-0314 相比取得了超过 86% 的令人印象深刻的胜率，与 Claude-Sonnet-3.5-1022 等顶级模型相当。这凸显了 DeepSeek-V3 的强大功能，尤其是在处理复杂的提示（包括编码和调试任务）方面。此外，DeepSeek-V3 实现了一个开创性的里程碑，成为第一个在 Arena-Hard 基准测试中超过 85% 的开源模型。这一成就显著弥合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可以完成的工作设定了新标准。

同样，DeepSeek-V3 在 AlpacaEval 2.0 上展示了卓越的性能，性能优于闭源和开源模型。这表明它在编写任务和处理简单的问答场景方面的出色能力。值得注意的是，它以 20% 的显着优势超越了 DeepSeek-V2.5-0905，突出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 作为生成式奖励模型

我们将 DeepSeek-V3 的判断能力与最先进的模型（即 GPT-4o 和 Claude-3.5）进行了比较。表 8 显示了这些模型在 RewardBench 中的性能 (Lambert et al., 2024)。DeepSeek-V3 的性能与最佳版本的 GPT-4o-0806 和 Claude-3.5-Sonnet-1022 相当，同时超越了其他版本。此外，DeepSeek-V3 的判断能力也可以通过投票技术来增强。因此，我们将 DeepSeekV3 与投票结合使用，以提供对开放式问题的自我反馈，从而提高

型	Chat	Chat-Hard	(蒸馏)	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
克劳德-3.5-十四行诗-0620	96.4	74.0	81.6	84.7	84.2
克劳德-3.5-十四行诗-1022	96.4	79.7	91.1	87.6	88.7
深度搜索-V3	96.9	79.8	87.0	84.3	87.0
深度搜索-V3 (maj@6)	96.9	82.6	89.5	89.2	89.6

从而改善了表 8 |GPT-4o、Claude-3.5-sonnet 和 DeepSeek-V3 在 RewardBench 上的表现。

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

[C 重试](#) [① 错误原因](#)

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

[C 重试](#) [① 错误原因](#)

effectiveness and robustness of the alignment process.

[C 重试](#) [① 错误原因](#)

5.4. Discussion

[C 重试](#) [① 错误原因](#)

5.4.1. Distillation from DeepSeek-R1

[C 重试](#) [① 错误原因](#)

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.

表 9 展示了蒸馏数据的有效性，显示 LiveCodeBench 和 MATH-500 基准测试都有了显著的改进。我们的实验揭示了一个有趣的权衡：蒸馏可以带来更好的性能，但也大大增加了平均响应长度。为了在模型精度和计算效率之间保持平衡，我们精心选择了 DeepSeek-V3 在蒸馏中的最佳设置。

我们的研究表明，从推理模型中提炼知识为训练后优化提供了一个有前途的方向。虽然我们目前的工作重点是从数学和编码领域提炼数据，但这种方法显示出在各种任务领域中更广泛应用的潜力。在这些特定领域证明的有效性表明，long-CoT 蒸馏对于提高模型在其他需要复杂推理的认知任务中的性能可能很有价值。跨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. 自我奖励

奖励在 RL 中起着关键作用，指导优化过程。在通过外部工具进行验证非常简单的领域中，例如某些编码或数学场景，RL 表现出非凡的功效。但是，在更一般的场景中，构建反馈

通过硬编码构建反馈机制是不切实际的。在 DeepSeek-V3 的开发过程中，针对这些更广泛的上下文，我们采用了宪法 AI 方法 (Bai et al., 2022)，利用 DeepSeek-V3 本身的投票评估结果作为反馈源。这种方法产生了显着的对齐效果，显着提高了 DeepSeek-V3 在主观评估中的性能。通过集成额外的宪法输入，DeepSeek-V3 可以朝着宪法方向进行优化。我们认为，这种将补充信息相结合作为LLMs反馈源的范式至关重要。它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的自我提升。除了自我奖励之外，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. 多 Token 预测评估

DeepSeek-V3 不是只预测下一个单个令牌，而是通过 MTP 技术预测接下来的 2 个令牌。结合推测解码框架 (Leviathan等人, 2023 年;Xia 等人, 2023 年)，它可以显着加快模型的解码速度。自然而然地出现了一个关于额外预测的令牌的接受率的问题。根据我们的评估，第二个令牌预测的接受率在各个生成主题中在 85% 到 90% 之间，显示出一致的可靠性。这种高接受率使 DeepSeek-V3 能够实现显著提高的解码速度，提供 1.8 倍的 TPS (每秒令牌数)。

6. 结论、局限性和未来方向

在本文中，我们介绍了 DeepSeek-V3，这是一个大型 MoE 语言模型，具有 671B 总参数和 37B 激活参数，在 14.8T 令牌上进行训练。除了 MLA 和 DeepSeekMoE 架构外，它还开创了一种用于负载均衡的辅助无损策略，并设置了多令牌预测训练目标以获得更强的性能。DeepSeek-V3 的训练具有成本效益，因为支持 FP8 训练和细致的工程优化。后期训练也成功地从 DeepSeek-R1 系列模型中提炼出推理能力。综合评估表明，DeepSeek-V3 已成为目前可用的最强开源模型，并实现了与 GPT-4o 和 Claude-3.5-Sonnet 等领先的闭源模型相媲美的性能。尽管性能强劲，但它也保持了经济的训练成本。它只需要 2.788M H800 GPU 小时即可进行完整训练，包括预训练、上下文长度扩展和后训练。

在承认其强大的性能和成本效益的同时，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 DeepSeekV3 部署策略已经实现了 DeepSeek-V2 的两倍多的端到端生成速度，但仍有进一步增强的潜力。幸运的是，随着更先进硬件的开发，这些限制有望自然而然地得到解决。

DeepSeek 始终坚持开源模式的路线，具有长期主义精神，旨在稳步接近 AGI（通用人工智能）的最终目标。未来，我们计划在以下方向的战略性投资研究。

- 我们将不断研究和完善我们的模型架构，以进一步改进

训练和推理效率，力争实现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，从而突破其建模能力的界限。

- 我们将不断迭代训练数据的数量和质量，并探索整合额外的训练信号源，旨在推动更全面维度的数据扩展。
- 我们将不断探索和迭代模型的深度思考能力，旨在通过扩大模型的推理长度和深度来提高模型的智能和解决问题的能力。
- 我们将探索更全面、多维的模型评价方法，以防止在研究过程中倾向于优化一套固定的基准，这可能会对模型能力产生误导性印象，影响我们的基础评价。

引用

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

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

人类的。克劳德 3.5 十四行诗, 2024 年。网址 <https://www.anthropic.com/news/clause-3-5-十四行诗>。

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, 和 W. Zaremba. 评估在代码上训练的大型语言模型。评估
URL <https://arxiv.org/abs/2107.03374>。2021 年

P. Clark、I. Cowhey、O. Etzioni、T. Khot、A. Sabharwal、C. Schoenick 和 O. Tafjord。认为您已经解决了问答问题？尝试 ARC，AI2 推理挑战。CoRR, abs/1803.05457, 英文, 2018. 网址 <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单词问题。arXiv 预印本 arXiv: 2110.14168, 2021 年。

Y. Cui, T. Liu, W. Che, L. Xiao, Z. Chen, W. 马, S. Wang, 和 G. 胡. 用于中文机器阅读理解的跨域提取数据集。K. Inui, J. 江, V. Ng, and X. Wan, editors, 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP) 论文集, 第 5883–5889 页, 中国香港, 2019 年 11 月。计算语言学协会。doi: 10.18653/v1/D19-1600。网址 <https://aclanthology.org/D19-1600>。

600.

戴大斌, 邓斌, 赵斌, 徐锐斌, 高浩, 陈大斌, 李建军, 曾伟斌, 于昱, 吴宇, 谢志强, Y. K. Li, P. Huang, F. Luo, C. Ruan, Z. Sui, 和 W. Liang. Deepseekmoe: 迈向专家混合语言模型的终极专家专业化。CoRR, abs/2401.06066, 2024 年。网址 <https://doi.org/10.48550/arXiv.2401.06066>。

DeepSeek-AI 的。Deepseek-coder-v2: 打破代码智能中闭源模型的障碍。CoRR, abs/2406.11931, 2024a。URL <https://doi.org/10.48550/arXiv.2406.11931>。

深度搜索-人工智能。Deepseek LLM: 以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 的。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。URL <https://doi.org/10.48550/arXiv.2405.04434>。

T. Dettmers、M. Lewis、Y. Belkada 和 L. Zettlemoyer. Gpt3. int8 ()：大规模变压器的 8 位矩阵乘法。神经信息处理系统进展, 35: 30318–30332, 2022 年。

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

D. Dua、Y. Wang、P. Dasigi、G. Stanovsky、S. Singh 和 M. Gardner. DROP: 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《计算语言学协会北美分会 2019 年会议论文集：人类语言技术》中, NAACL-HLT 2019, 美国明尼苏达州明尼阿波利斯, 2019 年 6 月 2 日至 7 日, 第 1 卷 (长篇和短篇论文), 第 2368–2378 页。计算语言学协会, 2019 年。doi: 10.18653/V1/N19-1246。网址

<https://doi.org/10.18653/v1/n19-1246>.

Y. Dubois、B. Galambosi、P. Liang 和 TB桥本。长度控制的羊驼：一种消除自动评估器偏差的简单方法。arXiv 预印本 arXiv: 2404.04475,2024 年。

W. Fedus、B. Zoph 和 N. Shazeer。开关变压器：以简单高效的稀疏性扩展到万亿参数模型。CoRR, abs/2101.03961,2021 年。网址 <https://arxiv.org/abs/2101.03961> 的。

M. Fishman、B. Chmiel、R. Banner 和 D. Soudry。将 FP8 训练扩展到万亿个代币 lms。arXiv 预印本 arXiv: 2409.12517,2024 年。

E. Frantar、S. Ashkboos、T. Hoefler 和 D. Alistarh。Gptq：生成式预训练转换器的准确训练后量化。arXiv 预印本 arXiv: 2210.17323,2022 年。

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

AP Gema、JO Leang、G. Hong、A. Devoto、ACM Mancino、R. Saxena、X. He、Y. Zhao、X. Du、MRG Madani、C. Barale、R. McHardy、J. Harris、J. Kaddour、E. van Krieken 和 P. Minervini。我们完成了 mmlu 吗？CoRR, abs/2406.04127,2024 年。网址 <https://doi.org/10.48550/arXiv.2406.04127>。

F. Gloeckle, B. Y. Idrissi, B. Rozière, D. Lopez-Paz, 和 G. Synnaeve。通过多标记预测更好、更快的大型语言模型。在第 41 届机器学习国际会议, ICML 2024, 奥地利维也纳, 2024 年 7 月 21 日至 27 日。OpenReview.net, 2024 年。网址 <https://openreview.net/forum?id=pEWAcjejiU2>.

谷歌。我们的下一代型号：Gemini 1.5,2024 年。网址 <https://blog.google/technology/ai/google-gemini-next-generation-model-2024> 年 2 月。

RL Graham、D. Bureddy、P. Lui、H. Rosenstock、G. Shainer、G. Bloch、D. Goldenberg、M. Dubman、S. Kotchubievsky、V. Koushnir 等人。可扩展的分层聚合协议 (SHArP)：一种用于高效数据缩减的硬件架构。2016 年第一届 HPC 通信优化国际研讨会 (COMHPC)，第 1–10 页。IEEE, 2016 年。

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

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时 – 代码智能的兴起。CoRR, abs/2401.14196, 2024. 网址 <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 年。网址 <https://arxiv.org/abs/1806.03377>。

B. He、L. Noci、D. Paliotta、I. Schlag 和 T. Hofmann。了解和最小化转换器训练中的异常值特征。在第 38 届神经信息处理系统年会上。

Y. He, S. Li, J. Liu, Y. Tan, W. Wang, H. Huang, X. Bu, H. Guo, C. 胡, B. Zheng, et al. 中文 simpleqa：大型语言模型的中文事实性评估。arXiv 预印本 arXiv: 2411.07140, 2024。

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

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 年。

Y. Huang, Y. Bai, Z. Zhu, J. Zhang, J. Zhang, T. Su, J. Liu, C. Lv, Y. Zhang, J. Lei, et al. C-Eval: 基础模型的多级多学科中文评估套件。arXiv 预印本 arXiv: 2305.08322, 2023.

N. Jain、K. Han、A. Gu、W. Li、F. Yan、T. Zhang、S. Wang、A. Solar-Lezama、K. Sen 和 I. Stoica。

Livecodebench: 对代码的大型语言模型进行整体和无污染的评估。

CoRR, abs/2403.07974,2024 年。网址 <https://doi.org/10.48550/arXiv.2403.07974>。

A. Q. 江, A. 萨布莱罗勒斯, A. 门施, C. 班福德, D. S. 查普洛特, D. d. L. 卡萨斯, F. 布雷桑德, G. 伦耶尔, G. 兰普尔, L. 索尔尼尔, et al. Mistral 7b. arXiv 预印本 arXiv: 2310.06825, 2023.

M. Joshi、E. Choi、D. Weld 和 L. Zettlemoyer。琐事QA: 用于阅读理解的大规模远程监督挑战数据集。在 R. Barzilay 和 M.-Y. Kan 编辑, 计算语言学协会第 55 届年会论文集 (第 1 卷: 长篇论文), 第 1601–1611 页, 加拿大温哥华, 2017 年 7 月。计算语言学协会。doi: 10.18653/v1/P17-1147。网址 <https://aclanthology.org/P17-1147>。

D. Kalamkar, D. Mudigere, N. Mellemundi, D. Das, K. Banerjee, S. Avancha, D. T. Vooturi, N. Jammalamadaka, J. Huang, H. Yuen, et al. 用于深度学习训练的 bfloat16 研究。

arXiv 预印本 arXiv: 1905.12322,2019 年。

S. Krishna、K. Krishna、A. Mohananey、S. Schwarcz、A. Stambler、S. Upadhyay 和 M. Faruqui。事实、获取和原因: 检索增强生成的统一评估。CoRR, abs/2409.12941,2024 年。doi: 10.48550/ARXIV.2409.12941。网址 <https://doi.org/10.48550/arXiv.2409.12941> 的。

T. Kwiatkowski、J. Palomaki、O. Redfield、M. Collins、AP Parikh、C. Alberti、D. Epstein、I. Polosukhin、J. Devlin、K. Lee、K. Toutanova、L. Jones、M. Kelcey、M. Chang、AM Dai、J. Uszkoreit、Q. Le 和 S. Petrov。自然问题: 问答研究的基准。翻译 Assoc. Comput. 语言学, 7: 452–466,2019 年。doi: 10.1162/tacl_a_00276。

URL https://doi.org/10.1162/tacl_a_00276。

G. Lai、Q. Xie、H. Liu、Y. Yang 和 EH 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 主编的 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 日至 11 日, 第 785–794 页。计算语言学协会, 2017 年。doi: 10.18653/V1/D17-1082。网址 <https://doi.org/10.18653/v1/d17-1082>。

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: 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扩展巨型模型。在第 9 届学习表征国际会议中, ICLR 2021。OpenReview.net, 2021 年。

URL <https://openreview.net/forum?id=qrwe7XHTmYb>。

Y. Leviathan、M. Kalman 和 Y. Matias。通过推测解码从变压器进行快速推理。在机器学习国际会议中, ICML 2023,2023 年 7 月 23 日至 29 日, 美国夏威夷檀香山, 机器学习研究论文集第 202 卷, 页面

19274–19286. PMLR, 2023 年。网址
https://proceedings.mlr.press/v202/leviathan23_a.html。

H. Li、Y. Zhang、F. Koto、Y. Yang、H. Zhao、Y. Gong、N. Duan 和 T. Baldwin。CMMLU: 测量中文中的大量多任务语言理解。arXiv 预印本 arXiv: 2306.09212,2023 年。

S. Li 和 T. Hoefer。Chimera: 使用双向管道高效训练大规模神经网络。在高性能计算、网络、存储和分析国际会议论文集中, SC '21, 第 1–14 页。ACM, 2021 年 11 月。doi: 10.1145/345881

7.3476145. URL <http://dx.doi.org/10.1145/3458817.3476145>。

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

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

Y. Li、F. Wei、C. Zhang 和 H. Zhang。EAGLE: 推测采样需要重新考虑特征不确定性。在第 41 届机器学习国际会议中, ICML 2024, 奥地利维也纳, 2024 年 7 月 21 日至 27 日。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.

B. Noune、P. Jones、D. Justus、D. Masters 和 C. Luschi。用于深度神经网络的 8 位数字格式。arXiv 预印本 arXiv: 2206.02915, 2022 年。

英伟达。使用 NVIDIA Magnum IO NVSH 提高 HPC 系统的网络性能

MEM 和 GPUDirect 异步。<https://developer.nvidia.com/blog/improving-network-performance-of-hpc-systems-using-nvidia-magnum-io-nvshmem-and-gpudirect-async>, 2022 年。

英伟达。布莱克威尔架构。<https://www.nvidia.com/en-us/data-center/technologies/blackwell-architecture/>, 2024a。

英伟达。TransformerEngine, 2024b。URL

<https://nvidia.github.io/TransformerEngine/>

打开人工智能。你好 GPT-4o, 2024a. 网址 <https://openai.com/index/hello-gpt-4o/>。

开放人工智能。多语言大规模多任务语言理解 (mmmlu), 2024b。网址

<https://huggingface.co/datasets/openai/MMMLU>.

OpenAI 的。介绍 SimpleQA, 2024c。URL <https://openai.com/index/introducing-simpleqa/> 中。

OpenAI 的 SWE-Bench 验证。介绍 SWE-bench 验证我们将发布经过人工验证的 swe- 子集
bench that more, 2024d. URL <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. 胡, et al. 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。arXiv 预印本 arXiv: 2401.10241, 2023a.

零气泡管道并行性, 2023b. URL <https://arxiv.org/abs/2401.10241>。

Qwen (英语)。Qwen 技术报告。arXiv 预印本 arXiv: 2309.16609, 2023 年。

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

Qwen 的。Qwen2.5: 基础模型的聚会, 2024b。网址 <https://qwenlm.github.io/blog/qwen2.5> 的

S. Rajbhandari、J. Rasley、O. Ruwase 和 Y. He。零: 针对训练万亿个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中, 第 1–16 页。IEEE, 2020 年。

D. Rein、BL Hou、AC Stickland、J. Petty、RY Pang、J. Dirani、J. Michael 和 SR Bowman。

GPQA: 研究生水平的 google proof-q&a 基准测试。arXiv 预印本 arXiv: 2311.12022, 2023。

BD Rouhani、R. Zhao、A. More、M. Hall、A. Khodamoradi、S. 邓、D. Choudhary、M. Cornea、E. Dellinger、K. Denolf 等人。用于深度学习的微标度数据格式。arXiv 预印本 arXiv: 2310.10537, 2023a。

BD Rouhani、R. Zhao、A. More、M. Hall、A. Khodamoradi、S. 邓、D. Choudhary、M. Cornea、E. Dellinger、K. Denolf 等人。用于深度学习的微标度数据格式。arXiv 预印本 arXiv: 2310.10537,2023b。

K. Sakaguchi、RL Bras、C. Bhagavatula 和 Y. Choi。Winogrande：大规模对抗性 winograd 模式挑战，2019 年。

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

N. Shazeer、A. Mirhoseini、K. Maziarz、A. Davis、Q. V. Le、GE Hinton 和 J. Dean。超大型神经网络：稀疏门控专家混合层。第 5 届国际

学习表征会议，ICLR 2017。OpenReview.net, 2017 年。网址 <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. 周、D. Das 和 J. Wei。语言模型是多语言的思维链推理器。

在第十一届学习表征国际会议中，ICLR 2023，卢旺达基加利，2023 年 5 月 1 日至 5 日。

OpenReview.net, 2023 年。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：具有旋转位置嵌入的增强型变压器。神经计算, 568: 127063,2024。

K. Sun、D. Yu、D. Yu 和 C. Cardie。调查挑战中文机器阅读理解的先验知识, 2019a。

M. Sun、X. Chen、JZ 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 年。

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

M. Suzgun、N. Scales、N. Schärlí、S. Gehrmann、Y. Tay、HW Chung、A. Chowdhery、Q. V. Le、E. H. Chi、D. 周 等人。具有挑战性的大工作台任务以及思维链是否可以解决它们。arXiv 预印本 arXiv: 2210.09261,2022 年。

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。卡特拉斯, 2023 年 1 月。网址 <https://github.com/NVIDIA/cutlas>

S.

H. Touvron, T. Lavril, G. Izacard, X. Martinet, MALachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. LLaMA：开放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. 侯, H. 伊南, M. 卡尔达斯, V. 科尔克兹, M. 哈布萨, I. 克鲁曼, A. 科列涅夫, P. S. 库拉, M. 拉肖, T. 拉夫里尔, J. 李, D. 利斯科维奇, Y. 卢, Y. 毛, X. 马丁内特, T. 米哈洛夫, P. 米什拉, I. 莫利伯格, Y. 聂, A. 波尔顿, J. 赖森斯坦, R. 朗塔, K. 萨拉迪, A. 谢尔滕, R. 席尔瓦, E. M. 史密斯, R. 苏布拉马尼安, X. E. 谭, B. 唐, R. 泰勒, A. 威廉姆斯, J. X. 宽, P. 徐, Z. 燕, I. 扎罗夫, Y. 张, A. 范, M. 坎巴杜尔, S. 纳朗, A. 罗德里格斯, R. 斯托伊尼克, S. 埃杜诺夫, 和 T. 西亚洛姆。骆驼 2: 开放基础和微调的聊天模型。CoRR, abs/2307.09288, 2023b。
doi: 10.48550/arXiv.2307.09288。网址 <https://doi.org/10.48550/arXiv.2307.09288>。

09288.

A. Vaswani、N. Shazeer、N. Parmar、J. Uszkoreit、L. Jones、AN Gomez、L.Kaiser 和 I. Polosukhin。注意力就是你所需要的。神经信息处理系统的进展, 30, 2017。

L. Wang、H. Gao、C. Zhao、X. Sun 和 D. Dai。混合专家的辅助无损负载均衡策略。CoRR, abs/2408.15664, 2024a. 网址 <https://doi.org/10.48550/arXiv.2408.15664>.

Y. Wang, X. 马, G. Zhang, Y. Ni, A. Chandra, S. Guo, W. 任, A. Arulraj, X. He, Z. 江, T. Li, M. Ku, K. Wang, A. Zhuang, R. Fan, X. Yue, 和 W. Chen. Mmlu-pro: 一种更强大且更具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: 您的语言模型能否通过中国小学数学考试? , 2023 年。

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 年。

H. 习、C. Li、J. Chen 和 J. Zhu。用 4 位整数训练转换器。神经信息处理系统进展, 36: 49146–49168, 2023 年。

CS Xia、Y. 邓、S. Dunn 和 L. Zhang。无代理: 揭开基于软件工程代理的神秘面纱! lm。arXiv 预印本, 2024 年。

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。计算语言学协会的调查结果: EMNLP 2023, 新加坡, 2023 年 12 月 6 日至 10 日, 第 3909–3925 页。

计算语言学协会, 2023. 网址 <https://doi.org/10.18653/v1-2023.调查结果-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的训练后量化。在机器学习国际会议上, 第 38087–38099 页。PMLR, 2023 年。

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 和 Y. Choi。HellaSwag：机器真的能说完你的话吗？在 A. Korhonen、DR Traum 和 L. Màrquez 主编的《计算语言学协会第 57 届会议论文集》中，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，第 4791–4800 页。计算语言学协会，2019 年。doi: 10.18653/v1/p19–1472。网址 <https://doi.org/10.18653/v1/p19–1472>。

9–1472.

W. Zhong、R. Cui、Y. Guo、Y. Liang、S. Lu、Y. Wang、A. Saied、W. Chen 和 N. Duan。AGIEval：评估基础模型的以人为本的基准。中国联合无线电报，abs/2304.06364,2023 年。

doi: 10.48550/arXiv.2304.06364。网址 <https://doi.org/10.48550/arXiv.2304.06364>。

J. 周、T. Lu、S. Mishra、S. Brahma、S. Basu、Y. Luan、D. 周和 L. Hou。大型语言模型的指令跟踪评估。arXiv 预印本 arXiv: 2311.07911,2023 年。

附录

A. 贡献和鸣谢

研究与工程

刘爱欣 Lecong Zhang ◉
薛冰 Liang Zhao ◉
王秉轩 Litong Wang ◉
吴博超 Liyue Zhang ◉
陆成达 Mingchuan Zhang ◉
赵成刚 Minghua Zhang ◉
邓成琪 Minghui Tang ◉
张晨宇* Panpan Huang ◉
阮崇 Peiyi Wang ◉
Peace Dai Qiancheng Wang ◉
郭大亚 Qihaoy Zhu ◉
杨德健 Qinyu Chen ◉
陈德利 Qiushi Du ◉
李尔杭 Ruiqi Ge ◉
林方云 Ruisong Zhang ◉
戴福聪 Ruizhe Pan ◉
罗福丽* Runji Wang ◉
郝广波 Runxin Xu ◉
陈冠婷 Ruoyu Zhang ◉
李国伟 Shanghao Lu ◉
张海 Shangyan Zhou ◉
韩宝* Shanhua Chen ◉
徐汉伟 Shengfeng Ye ◉
王浩成* Shirong Ma ◉
张浩伟 Shiyu Wang ◉
宏辉物语 Shuiping Yu ◉
辛华剑* Shunfeng Zhou ◉
高华作 Shuting Pan ◉
曲慧 Tao Yun ◉
郭建中 Tian Pei ◉
李佳石 Wangding Zeng ◉
王佳伟* Wanjia Zhao* ◉
陈景昌 Wen Liu ◉
Jingyang Yuan ◉ Wenfeng Liang ◉
Junjie Qiu ◉ Wenjun Gao ◉
Junlong Li ◉ Wenqin Yu ◉
Junxiao Song ◉ Wentao Zhang ◉
Kai Dong ◉ Xiao Bi ◉
Kai Hu* ◉ Xiaodong Liu ◉
Kaige Gao ◉ Xiaohan Wang ◉
Kang Guan ◉ Xiaokang Chen ◉
Kexin Huang ◉ Xiaokang Zhang ◉
Kuai Yu ◉ Xiaotao Nie ◉
Lean Wang ◉ Xin Cheng ◉
Xin Liu ◉

谢欣	Zhigang Yan
刘兴超	Zihong Shao
俞兴凯	Zhiyu Wu
杨欣宇	Zhuoshu Li
李欣元	Zihui Gu
苏雪成	Zijia Zhu
林旭恒	Zijun Liu*
李耀强	Zilin Li
王耀清	Ziwei Xie
魏 Y.X.	Ziyang Song
张洋	Ziyi Gao
徐彦宏	Zizheng Pan
李瑶	Data Annotation
赵瑶	Bei Feng
孙耀峰	Hui Li
王耀辉	J.L. Cai
余毅	Jiaqi Ni
张毅超	Lei Xu
史一凡	Meng Li
熊一亮	Ning Tian
何英	R.J. Chen
彪石	R.L. Jin
王一松	Ruyi Chen
谭逸轩	S.S. Li
马一阳*	Shuang Zhou
刘怡源	Tianyu Sun
郭永强	X.Q. Li
吴宇	Xiangyue Jin
元或	Xiaojin Shen
王玉端	Xiaosha Chen
龚岳	Xiaowen Sun
邹玉恒	Xiaozi Wang
何玉佳	Xinnan Song
熊云帆	Xinyi Zhou
Yuxiang Luo	Y.X. Zhu
Yuxiang You	徐彦宏
Yuxuan Liu	Yanping Huang
Yuyang Zhou	Yaohui Li
Z.F. Wu	Yi Zheng
Z.Z. Ren	Yuchen Zhu
Zehui Ren	Yunxian Ma
Zhangli Sha	Zhen Huang
Zhe Fu	Zhipeng Xu
Zhean Xu	Zhongyu Zhang
Zhenda Xie	Business & Compliance
Zhengyan Zhang	Dongjie Ji
Zhewen Hao	
Zhibin Gou	
Zhicheng Ma	

梁健
陈瑾
夏乐怡
王淼军
李明明
张鹏
吴绍青
叶胜峰
王 T.

W.L. 肖
安伟
王贤祖
单欣霞
唐英
查玉坤
严玉婷
张震

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

B. 用于低精度训练的消融研究

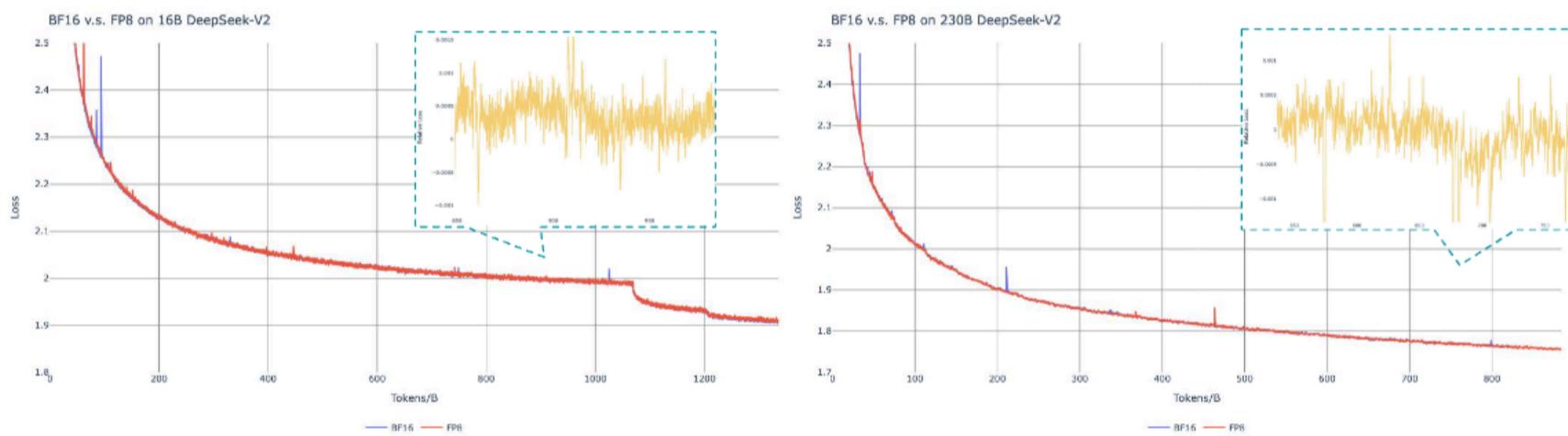


图 10 |BF16 和 FP8 训练之间的损失曲线比较。结果由系数为 0.9 的指数移动平均线（EMA）平滑处理。

B.1. FP8 与 BF16 训练

我们通过与不同规模的两个基线模型之上的 BF16 训练进行比较来验证我们的 FP8 混合精度框架。在小规模上，我们在 1.33T 代币上训练一个基线 MoE 模型，该模型包含大约 16B 的总参数。在大规模上，我们在大约 0.9T 代币上训练了一个基线 MoE 模型，该模型包含大约 230B 的总参数。我们展示了图 10 中的训练曲线，并证明使用我们的高精度累积和细粒度量化策略，相对误差保持在 0.25% 以下。

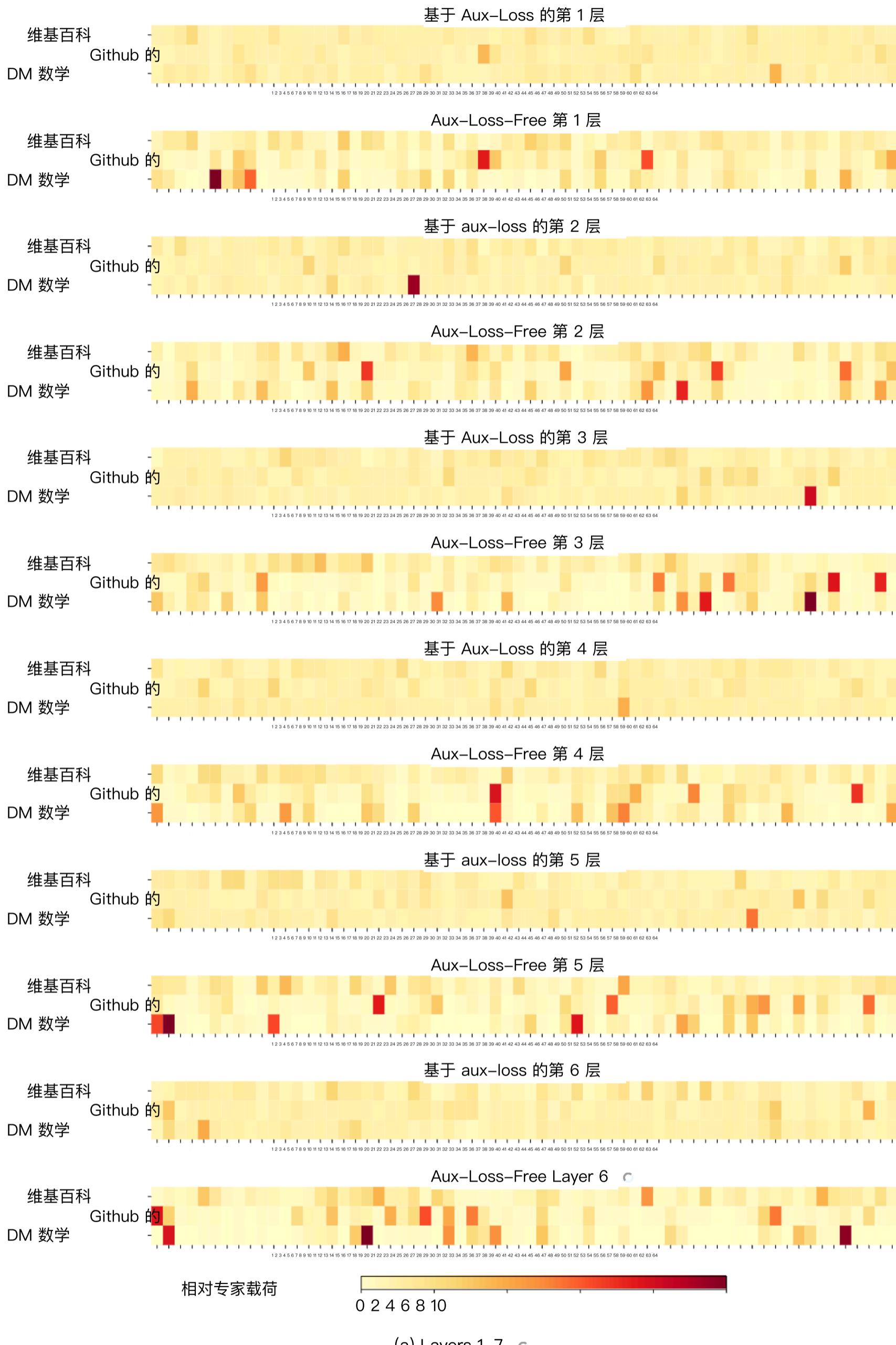
B.2. 关于逐块量化的讨论

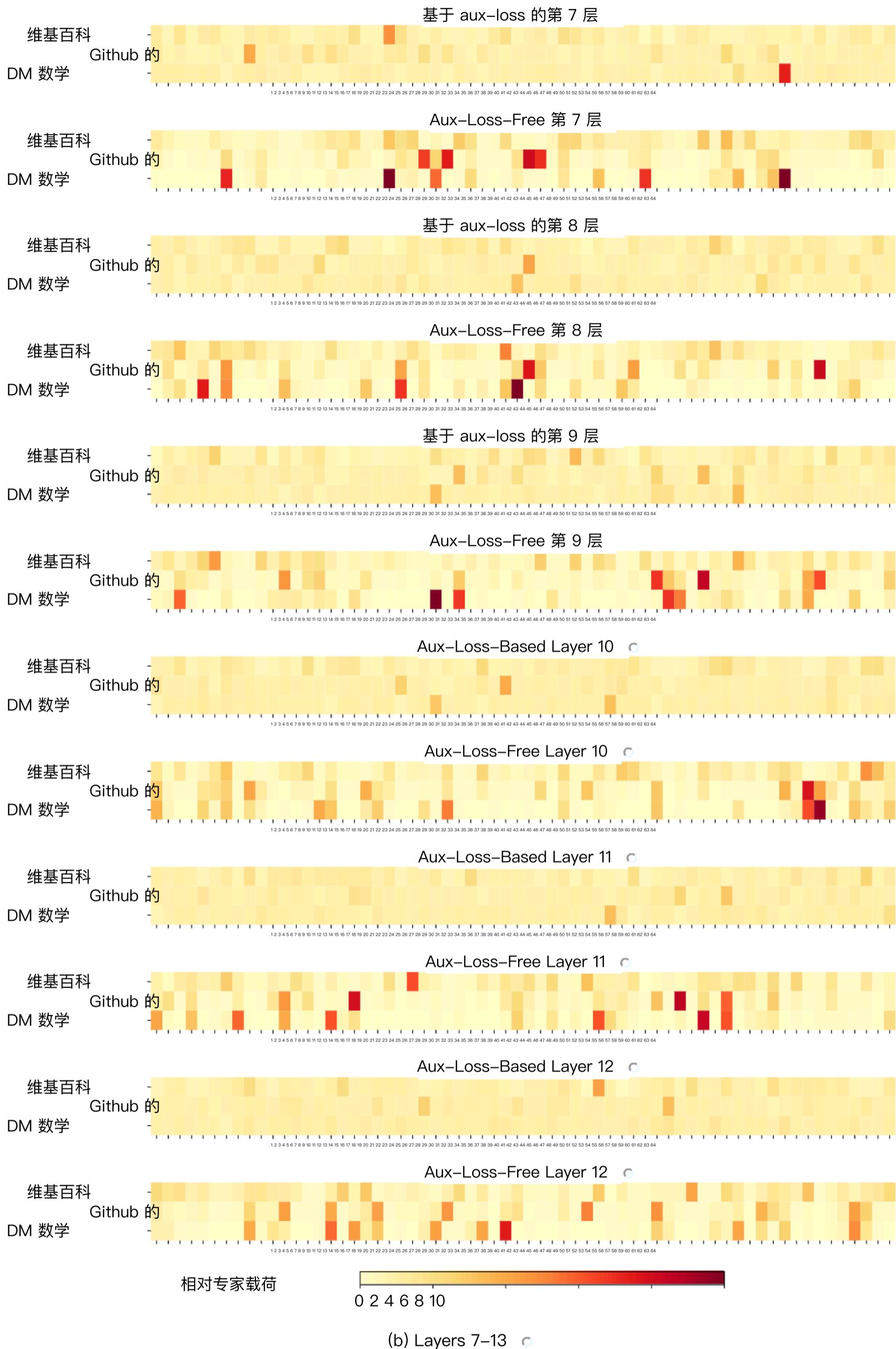
Although our tile-wise fine-grained quantization effectively mitigates the error introduced by feature outliers, it requires different groupings for activation quantization, i.e., 1x128 in forward pass and 128x1 for backward pass. A similar process is also required for the activation gradient. A straightforward 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

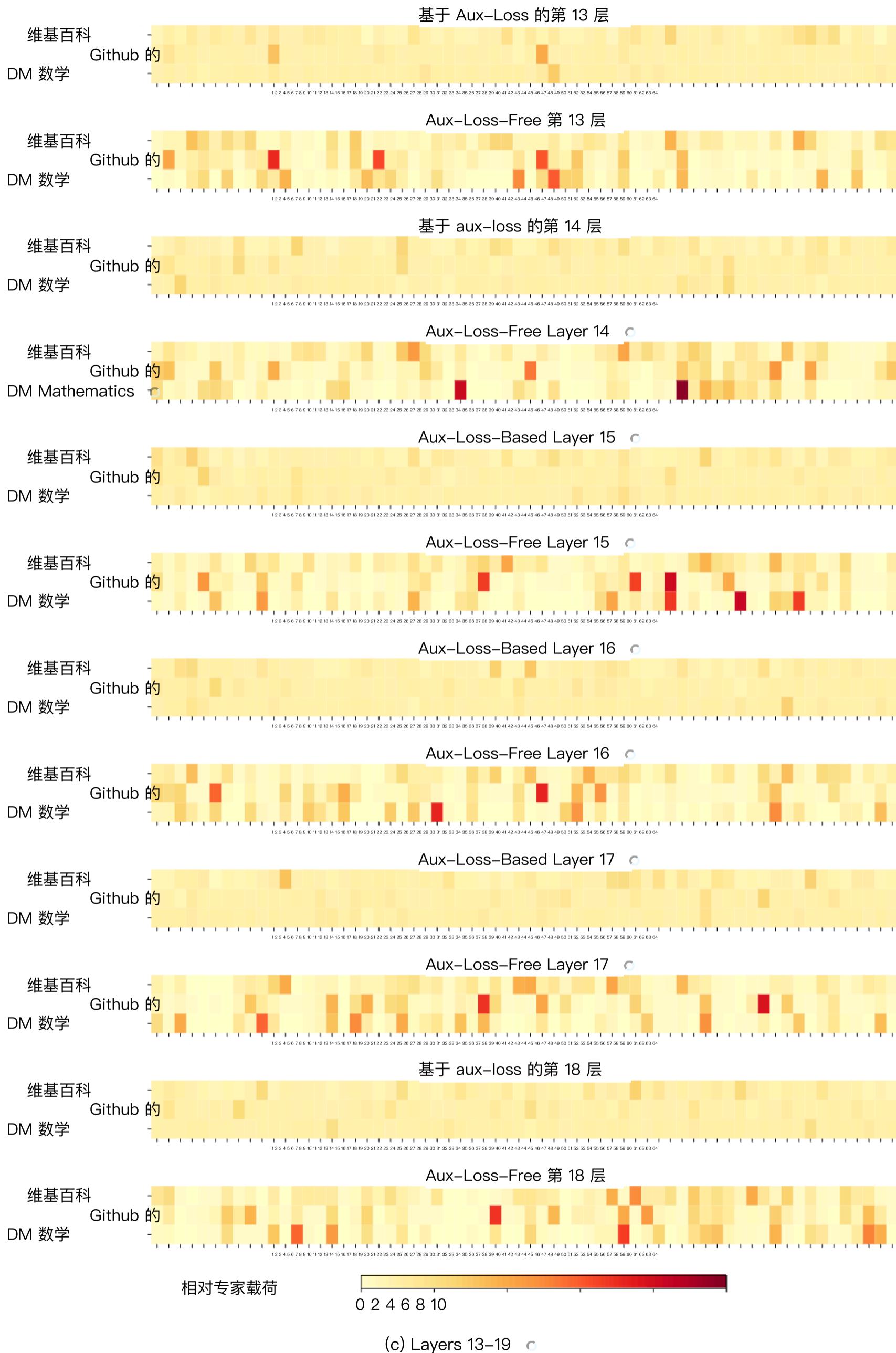
激活梯度的块级量化导致模型在包含大约 16B 总参数的 MoE 模型上出现分歧，训练了大约 300B 令牌。我们假设这种敏感性的出现是因为令牌之间的激活梯度高度不平衡，导致令牌相关的异常值 (习 et al., 2023)。这些异常值无法通过块级量化方法有效管理。

C. 16B Aux–Loss–Based 和 Aux–LossFree 型号的专家专业化模式

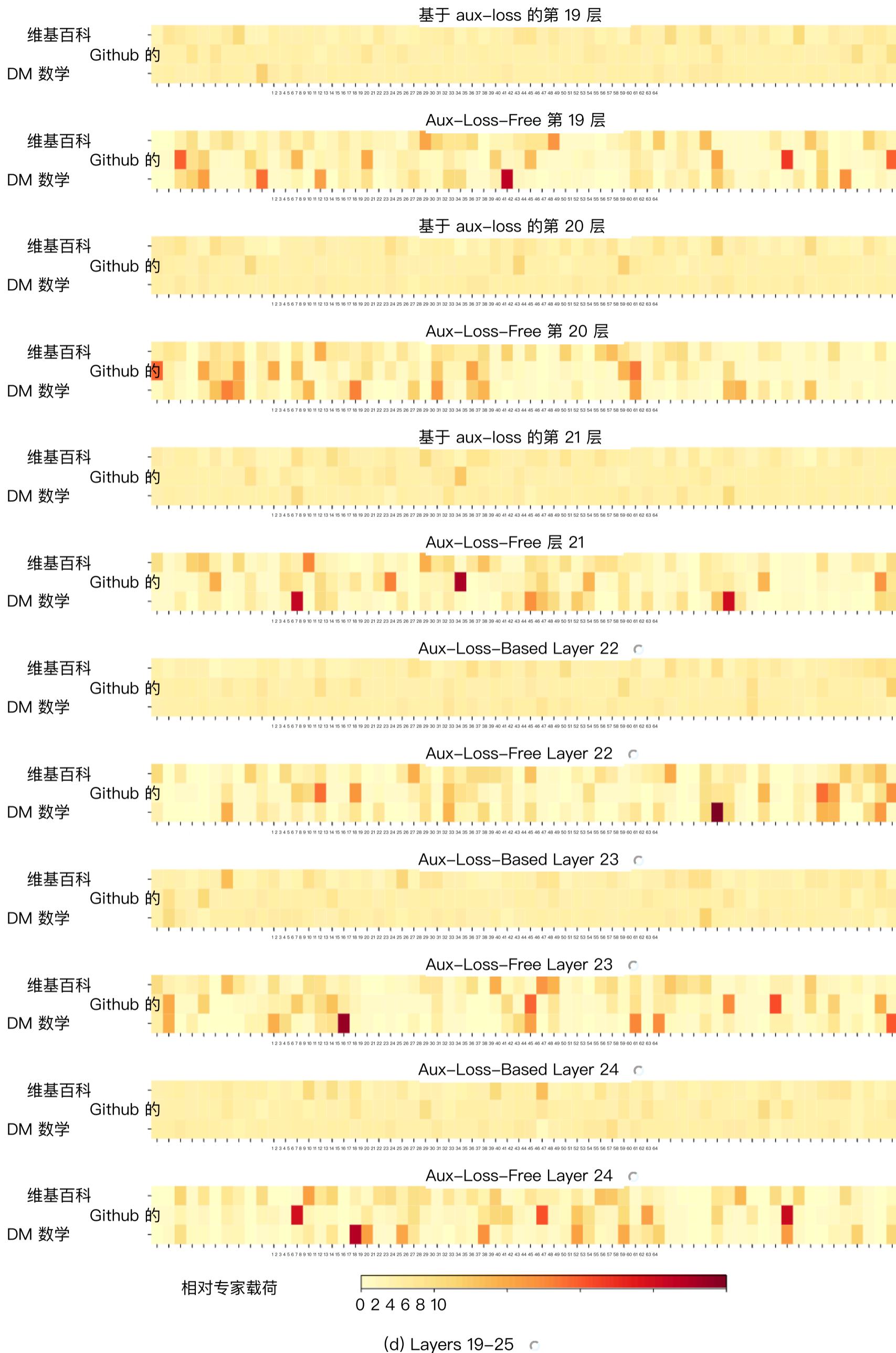
我们在 Pile 测试集上记录了 16B 基于辅助损耗的基线和辅助无损模型的专家负载。辅助无损模型往往在所有层中具有更大的专家专业化，如图 10 所示。







(c) Layers 13–19



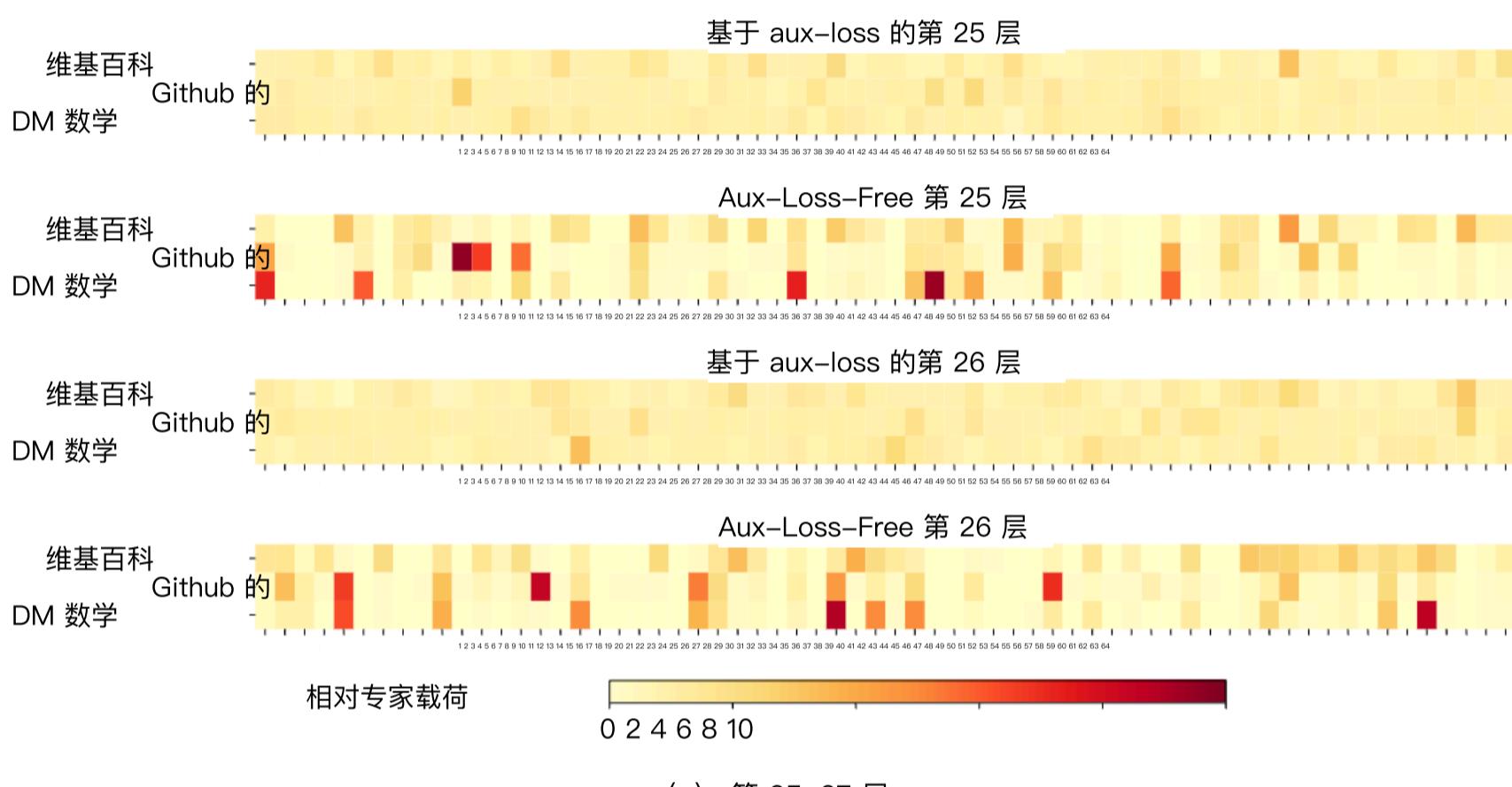


图 10 | 辅助无损和基于辅助损耗模型在 Pile 测试集中三个域上的专家载荷。辅助无损模型比基于辅助损失的模型显示出更大的专家专业化模式。相对专家载荷表示实际专家载荷与理论上平衡的专家载荷之间的比率。