

# DeepSeek 是否有国运级的创新？

——2 万字解读与硬核分析 DeepSeek V3/R1 的架构

（官方版 V1.31/2025 年 2 月 7 日）

知乎专栏：<https://www.zhihu.com/people/chenweiphd>

微信公众号：陈巍 博士

# 目录

1 V3 与 R1 的主要特征 .....	1
1.1 V3/R1 架构特征 .....	1
1.2 R1 在 CoT 的进化 .....	3
1.2.1 DeepSeek-R1-Zero .....	3
1.2.2 DeepSeek-R1 .....	3
2 V3/R1 的架构提升 .....	4
2.1 多头潜注意力 (MLA) .....	4
2.1.1 从 KV Cache (KV 缓存) 说起 .....	4
2.1.2 MLA 的原理与优势 .....	5
2.1.4 MLA 是颠覆性创新吗? .....	6
2.2 MoE 架构与辅助无损负载均衡 .....	7
2.2.1 MoE 与 Dense 模型的混战 .....	7
2.2.2 无辅助损耗负载均衡 .....	7
2.2.3 MoE 会是大模型的终局吗? .....	8
3 V3/R1 训练架构的独特优势 .....	10
3.1 HAI-LLM 框架的软硬件协同设计 .....	10
3.1.1 软件层面的并行优化 .....	11
3.1.2 针对软件并行策略的硬件优化 .....	12
3.1.3 针对硬件架构的软件优化 .....	13
3.2 FP8 训练框架体系 .....	13
3.2.1 低比特训练框架的构建 .....	13
3.2.2 对英伟达 GPU 市场有巨大影响? .....	15
3.3 DualPipe 优化 .....	16
3.4 跨节点 All-to-All 通信与显存优化 .....	17
3.4.1 对于 SM 与 NVLink 的优化 .....	17
3.4.2 显存节省技术 .....	18
3.4.3 打破了 CUDA 生态壁垒? .....	18


3.4.4 挖了 NVLink 的墙角？ .....	19
4 V3 的训练流程 .....	20
4.1 V3 的基础预训练 .....	20
4.2 V3 长文扩展训练 .....	21
4.3 V3 的后训练/精调 .....	22
4.3.1 V3 的有监督精调（SFT） .....	22
4.3.2 V3 的强化学习 .....	22
5 R1 的训练流程 .....	23
5.1 无 SFT 的 R1-Zero 训练 .....	23
5.2 DeepSeek-R1 的训练流程 .....	25
5.2.1 冷启动（Cold Start）：CoT SFT .....	25
5.2.2 面向推理的强化学习 .....	25
5.2.3 拒绝采样与 SFT .....	26
5.2.4 面向全场景的强化学习与对齐 .....	27
5.3 从 MoE 回归 Dense（蒸馏 SFT） .....	27
5.4 更大显存容量显得尤为重要？ .....	28
6 结语 .....	30
6.1 DeepSeek 的关键贡献 .....	30
6.2 R1 的出现是国运级的贡献吗？ .....	31
6.3 对于国产 AI 芯片的启示 .....	32

DeepSeek 的最新模型 DeepSeek-V3 和 DeepSeek-R1 都属于 MoE（混合专家）架构，并在开源社区产生了较大的影响力。特别是 2025 年 1 月开源的 DeepSeek-R1，模型性能可挑战 OpenAI 闭源的 o1 模型。

随着热度的提升，DeepSeek 也被大模型行业之外的各路媒体不断提起，“打破 CUDA 垄断”，“挖了 NVLink 的墙角”，“引发英伟达市值大跌”，“证明大模型算力建设浪费”，“算力霸权转移”，“国运级的创新”，似乎有用皮衣卡住老黄脖子的架势。

那么，从技术和架构的角度深入最新的 V3 和 R1 模型，是否真的有“国运级的创新”，又有哪些误传？

下面我们从 V3 与 R1 的架构分析开始，分层解读 DeepSeek 的创新。



陈巍  
MemoryCompute

主编作者

AI芯片+大模型，高级职称，中国计算机学会（CCF）专委委员，国际计算机学会（ACM）会员。

曾担任领域**华X系AI企业（自然语言处理）首席科学家、国际存储大厂3D NAND芯片团队/架构负责人、中科院副主任（SoC/IP核）**，毕业于清华大学，个人**中国发明专利、美国发明专利与软件著作权70+项**，著有 Sora视频大模型与GPT-4相关著作。

**曾带队完成：**

- 国内首个医疗领域专用AI处理器
- 首个RISC-V/x86/ARM平台兼容的AI加速编译器
- 国内首个3D 存储器芯片架构与设计团队建立（对标三星，已成为国家级存储企业的前置工作）
- 国内首个嵌入式闪存平台与编译器（对标台积电/SST，该平台流片量数十亿颗）

3D Chiplet处理器

大模型架构

稀疏量化压缩与部署加速

存算一体

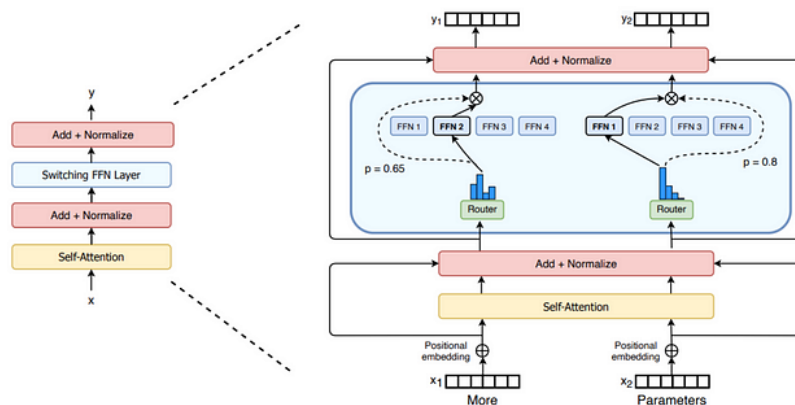
# 1 V3 与 R1 的主要特征

DeepSeek-R1 的模型架构来自于 V3，甚至可以说 R1 是具有推理（Reasoning）能力的 V3。下面先分别分析 V3 和 R1 的架构特征。

## 1.1 V3/R1 架构特征

DeepSeek-V3 是一个混合专家（MoE）语言模型，具有 6710 亿(671B)参数，其中每个 Token（词元）的计算约激活 370 亿（37B）参数。这个模型参数量与 GPT-4 大致在同一数量级。

MoE（Mixture of Experts）是组合多个专家模型提升深度学习模型性能和效率的架构。其核心思想是通过引入多个专家模型（Experts），每个输入数据只选择和激活其中的一部分专家模型进行处理，从而减少计算量，提高训练和推理速度。MoE 的概念在 1991 年就已提出，训练不容易收敛是 MoE 在大模型领域应用的主要障碍。



MoE 模型基本结构示意图（来源：互联网）

DeepSeek-V3 采用了多头潜注意力（MLA，对传统多头注意力机制的改进）和 DeepSeekMoE 架构（对传统 MoE 架构的改进），无辅助损失的负载平衡策略等创新技术，基于 14.8 万亿 Token 的数据进行训练，在代码生成、逻辑分析等任务中表现出色。

其中多头潜注意力（MLA）机制和 DeepSeekMoE 是 V3 和 R1 模型提高计算效率，减少算力浪费的关键。其中 MLA 大概贡献了 2-4 倍的计算效率提升，MoE 大概贡献了 4 倍以上的计算效率提升。

## 1) MLA（Multi-Head Latent Attention）

在“All you need is attention”的背景下，传统的多头注意力（MHA，Multi-Head Attention）的键值（KV）缓存机制事实上对计算效率形成了较大阻碍。缩小 KV 缓存（KV Cache）大小，并提高性能，在之前的模型架构中并未很好的解决。DeepSeek 引入了 MLA，一种通过低秩键值联合压缩的注意力机制，在显著减小 KV 缓存的同时提高计算效率。低秩近似是快速矩阵计算的常用方法，在 MLA 之前很少用于大模型计算。在这里我们可以看到 DeepSeek 团队的量化金融基因在发挥关键作用。当然实现潜空间表征不止低秩近似一条路，预计后面会有更精准高效的方法。

从大模型架构的演进情况来看，Prefill 和 KV Cache 容量瓶颈的问题正一步步被新的模型架构攻克，巨大的 KV Cache 正逐渐成为历史。（事实上在 2024 年 6 月发布 DeepSeek-V2 的时候就已经很好的降低了 KV Cache 的大小）

## 2) DeepSeekMoE

为了让 1991 年就提出的 MoE 架构更好的融入大模型体系，克服传统 MoE 模型的训练难题。DeepSeek 采用了细粒度专家+共享/通用专家的思路，不再使用少数大专家的结构，而是使用共享专家+大量极小的专家结构。这个思路的本质在于将知识空间进行离散细化，以更好的逼近连续的多维知识空间，是一个非常好的方法。

无辅助损失的负载平衡策略可在不依赖辅助损失函数的情况下平衡分配计算/训练负载，更好的提高训练稳定性。

基于以上关键的改进，V3 实现了更高的训练效率，比性能类似的 Llama 3.1 405B 少了大约 10 倍的训练计算量。

## 1.2 R1 在 CoT 的进化

广义上的 DeepSeek-R1 不是一个单一的模型，还包括了 R1 的初始阶段模型 DeepSeek-R1-Zero，以及几个基于 R1 蒸馏的较小的大模型。在这里我们主要讨论 R1-Zero 和 R1。

### 1.2.1 DeepSeek-R1-Zero

DeepSeek-R1-Zero 最大的特点在于，该模型仅使用强化学习进行的训练，通过各种思维链（CoT，Chain of Thought）数据特别是 Long CoT 数据来激活模型的推理能力。

DeepSeek-R1-Zero 是一个独特的通过大规模强化学习（RL，Reinforcement Learning）训练的模型，无需有监督精调（SFT，Supervised Fine-Tuning），具备较强的推理（Reasoning）能力。

首先要区分两个容易混淆的概念：

Reasoning（推理）：通过对事实的考虑和分析来得出结论的过程。推理强调的是思考和决策的过程，比“推断”具有更多的逻辑和分析过程。

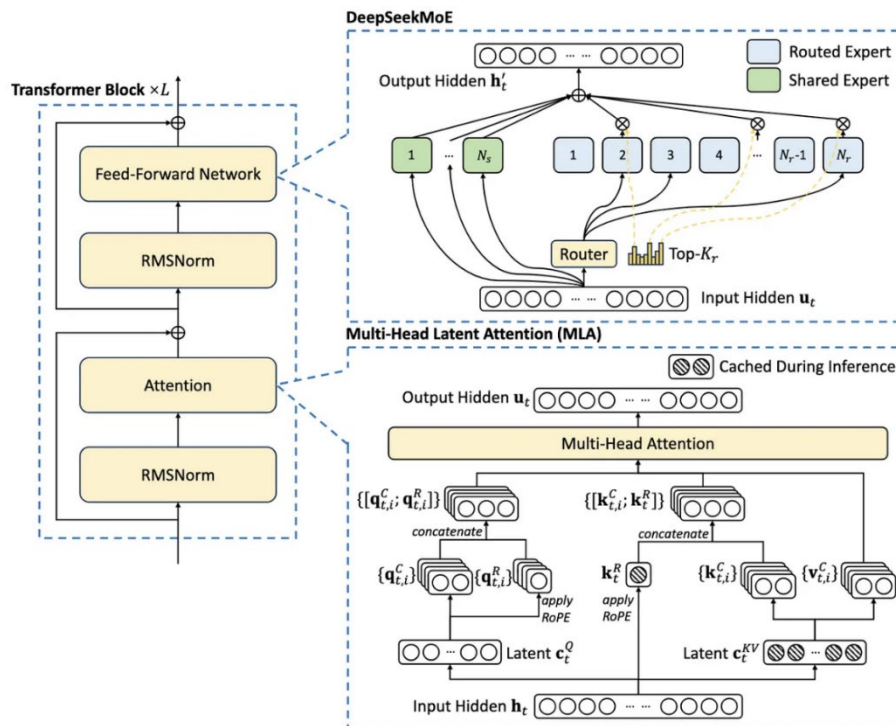
Inference（推断）：基于已有信息或数据推导出新的信息或结论的过程。推断侧重于通过既定的算法或模型来实现，与“推理”相比，更强调形式化和计算过程。

R1-Zero 展示出自我验证、反思和长链思维推理能力，甚至在推理方面得分略微超过 R1。虽然 R1-Zero 有一些明显的局限性，特别是在输出可读性和语言一致性方面，仍需要解决可读性差和语言混合等问题。

R1-Zero 大概是第一个公开验证大模型的推理（Reasoning）能力可以仅通过强化学习来完成训练的范例。在我们看来，R1-Zero 的价值远超 R1。按照 NLP 领域对语言的理解，人类的自然语言并不是最完美的推理语言。在 R1-Zero 的进一步进化过程中，或许可以构建出更适合推理的混合语言 IR，建立更高效的推演体系。未来的机器人或许会拥有更高效的逻辑自然语言。

### 1.2.2 DeepSeek-R1

相比之下，DeepSeek-R1 采用了多阶段训练方法，加入了 SFT，而不是采用纯粹的强化学习，R1 从一小组精心挑选的示例数据（称为“冷启动数据”）进行有监督精调（SFT），再进入强化学习。这种方法改善了 DeepSeek-R1-Zero 的语言可读性和连贯性，同时在推理之外的测试中实现了更好的性能。



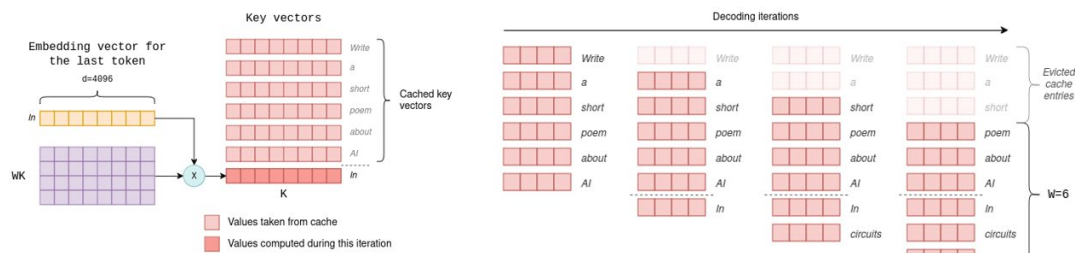
DeepSeek-V3 整体架构（来源：DeepSeek）

## 2 V3/R1 的架构提升

### 2.1 多头潜注意力 (MLA)

#### 2.1.1 从 KV Cache (KV 缓存) 说起

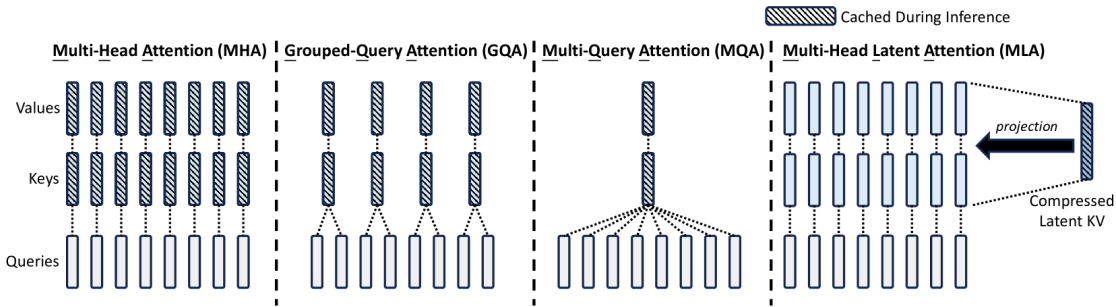
当使用传统 Transformer 在推断（Inference）过程中计算并生成 Token（词元）时，模型需要读入所有过去 Token 的上下文，以决定接下来输出什么 Token。最直观的方法就是简单的再次进行一次包括所有过去 Token 的前向传播（计算）。



KV Cache（来源：互联网）

传统的基于 Transformer 的模型在推理过程中会计算每个键值(KV)对，但事实上这种方法效率并不高，因为大部分过去的 Token 已经在上一次计算中处理过了，重复计算会产生大量的浪费。

目前常用的方法是缓存所有过去 Token 的相关内部状态，主要是注意力机制中的键（Key）和值（Value）向量。这也是键值缓存（简称 KV 缓存，也就是常说的 KV Cache）名称的由来。



不同注意力机制的对比（来源：DeepSeek V2）

目前开源大模型中的主流方法是分组查询注意力（Grouped-Query Attention）机制。在这种机制中，为每对键和值头分配多个查询头，将查询头有效的分组在一起。在 Llama 3.3 70B 和 Mistral Large 2 等模型中，仅分组查询注意力机制就将 KV 缓存大小减少了大约一个数量级。

### 2.1.2 MLA 的原理与优势

DeepSeek 使用的 Multi-Head Latent Attention 技术可大大节省 KV 缓存，从而显著降低了计算成本。

MLA 的本质是对 KV 的有损压缩，提高存储信息密度的同时尽可能保留关键细节。该技术首次在 DeepSeek-V2 中引入，与分组查询和多查询注意力等方法相比，MLA 是目前开源模型里显著减小 KV 缓存大小的最佳方法。

MLA 的方法是将 KV 矩阵转换为低秩形式：将原矩阵表示为两个较小矩阵（相当于潜向量）的乘积，在推断过程中，仅缓存潜向量，而不缓存完整的键 KV。这规避了分组查询注意力和多查询注意力的查询的信息损失，从而在降低 KV 缓存的前提下获得更好的性能。

• If

$$V = WH$$

• then

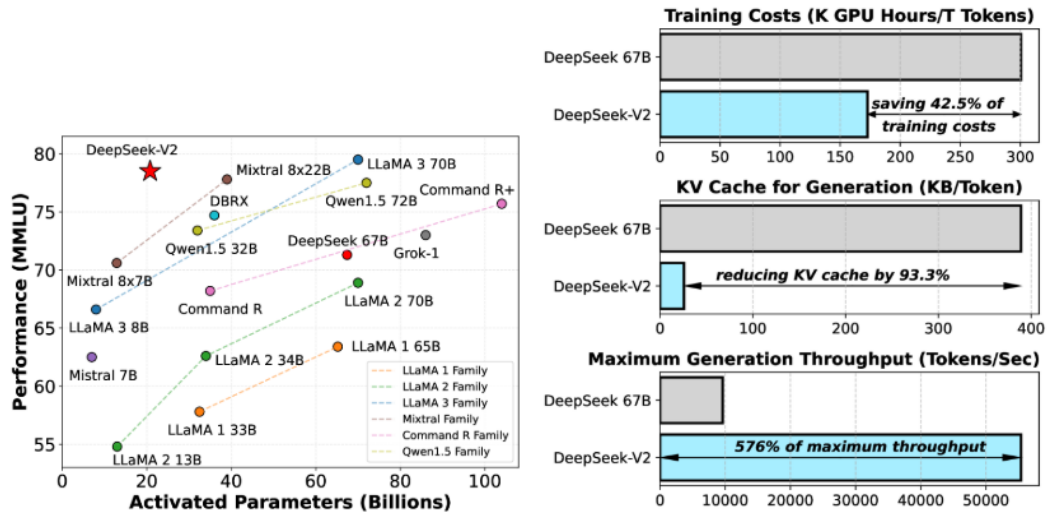
$$v_{i,j} = w_{i,*} h_{*,j}$$
$$= \sum_{x=1}^k w_{i,x} h_{x,j}$$

The diagram illustrates the low-rank approximation  $V = WH$ . Matrix  $V$  is a large grid representing the original key-value matrix, with one row highlighted in blue. Matrix  $W$  is a smaller grid representing the weight matrix, with one row highlighted in orange. Matrix  $H$  is a grid representing the hidden state matrix, with one column highlighted in yellow. The equation  $V = WH$  is shown between the matrices.

矩阵的低秩近似（来源：互联网）



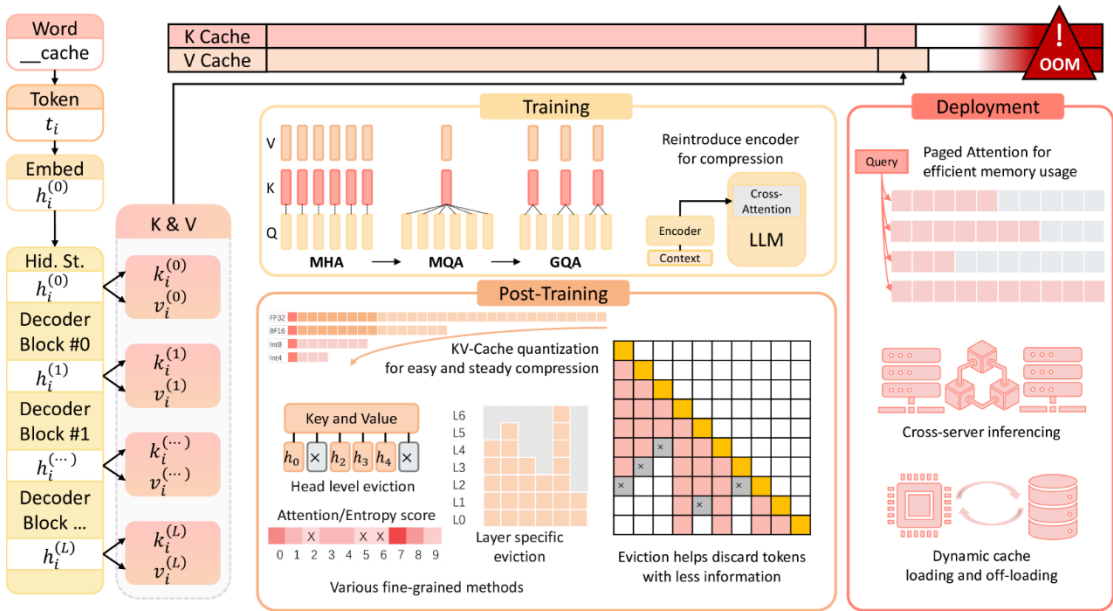
另外为了与 MLA 技术的低秩 KV 压缩兼容，DeepSeek 也将编码方式 RoPE 进行了改进，使 V2/V3/R1 获得了长上下文的外延能力。



MLA 方法有效降低 KV Cache 和训练成本（来源：DeepSeek）

### 2.1.4 MLA 是颠覆性创新吗？

我们认为 MLA 是个非常有趣且先进的创新，这一工作建立在对注意力机制深度理解的之上，并且需要进行大胆谨慎的验证。限于算力条件和个人 ROI，能够独立完成这一创新的团队并不多。能做出 MLA 这样的工作，确实是达到国际一线架构水平了。换一个角度看，MLA 也是建立在 DeepSeek 团队的量化金融基因之上，不禁让我们联想到优秀的量化码农对每个矩阵计算的 FPGA 底层优化。



MLA 之外的 KV Cache 优化方法（来源：武汉大学）

我们认为，MLA 之后，应该还会有 QMLA（量化 MLA）或者 CMLA（压缩 MLA），甚至是超越现有 Attention 模式的技术出现，而用了很多年的 Transformer 也将经历大的变革。真正的颠覆创新可能，正摆在 DeepSeek 和国内其他大模型团队的面前。

## 2.2 MoE 架构与辅助无损负载均衡

### 2.2.1 MoE 与 Dense 模型的混战

目前的主流的大模型架构可以分为 Dense（稠密）架构和 MoE 架构。

Dense 模型在深度学习中通常指的是一种全部神经元都参与计算的网路结构。这种结构使得模型能够充分利用数据特征，并且训练过程中参数共享，减少了计算量和过拟合的风险。

一般来说，Dense 模型可以视为仅有一个专家的 MoE 模型。在大模型领域，Dense 模型和 MoE 各有各的应用场景和优势，MoE 还无法代替 Dense 模型的行业应用。

	Dense 模型	MoE 模型
优势	在专业领域计算参数量更少，更节省计算资源	在通用计算领域激活的参数少，更节省计算资源
劣势	在通用领域需要激活更多的参数，计算资源消耗大	在专业领域无需多位专家，容易产生大量参数冗余，浪费资源

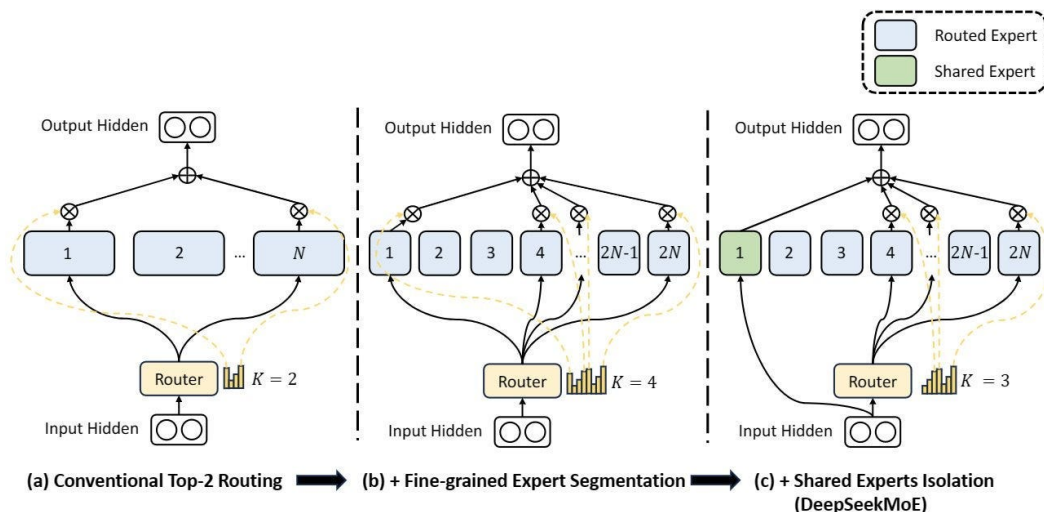
相比 Dense 模型，MoE 主要增加了专家路由，通过门控（仅有开或关）的方法，负责将数据流引向不同的专家模块。专家路由事实上引入了一个不连续的门控函数（对梯度计算不友好），这导致梯度下降优化方法在 MoE 训练中表现不佳，甚至出现“路由崩溃”，即模型容易陷入始终为每个 Token 激活相同的少数专家的窘境，而不是将计算合理的传播到所有的可用专家。这也是 MoE 模型训练的难点。

### 2.2.2 无辅助损耗负载均衡

传统的规避路由崩溃的方法是强制“平衡路由”，即通过训练策略让每个专家在足够大的训练批次中被激活的次数大致相等。这一策略也就是“辅助损失”。但这种强制性的辅助损失会由于训练数据的结构不均衡特征，导致同领域的专家能力分散到不同的专家模块之中，极度损害 MoE 模型的性能。理想的 MoE 应该有一些经常访问高频通用信息，并具备其他访问较少的专业领域专家。如果强制平衡路由，将失去实现此类路由设置的能力，并且必须在不同的专家之间冗余的复制信息。

DeepSeek 采用了“增加共享专家+无辅助损耗负载均衡”的方法解决这一问题。

DeepSeek 将专家分为两类：共享专家和路由专家。共享专家始终会被路由，在训练中重点确保路由专家的路由均衡。



DeepSeekMoE 与传统的多路由和细粒度专家 MoE 对比（来源：DeepSeek）

无辅助损耗负载均衡（Auxiliary-Loss-Free Load Balancing）方法是将特定于专家的偏差项添加到路由机制和专家亲和力中。偏差项不会通过梯度下降进行更新，而是在整个训练过程中持续监控并进行调整以确保负载均衡。如果训练中某个专家没有获得合理的命中次数，可以在每个梯度步骤中精调偏差项增加命中概率。

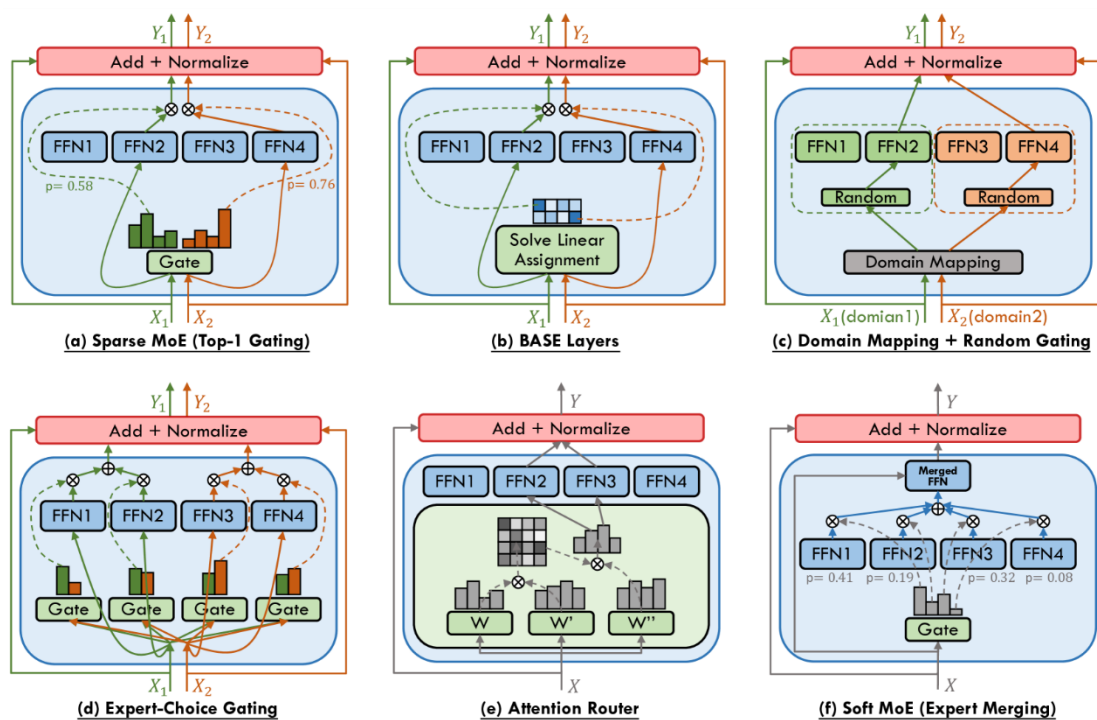
通过动态调整，DeepSeek-V3 在训练过程中获得了比有辅助损失均衡模型更好的性能。

从模型架构分析看，我们认为这种分配策略可能还不是理论最优的，但该方法已经比强制辅助损失有了显著的改进。

## 2.2.3 MoE 会是大模型的终局吗？

MoE 架构自 1991 年提出后，就一直在与 Dense 架构共生发展。

DeepSeek R1 的推出确实大大推动了开源 MoE 大模型的发展，并为 MoE 的落地应用提供了更多可能。但是我们也应看到，MoE 模型与应用领域高度和 TOC（Total Owning Cost，总拥有成本）密切相关，很多场景 MoE 未必比 Dense 模型好。



不同的 MoE 架构（来源：香港科技大学）

另外，MoE 模型也有不同的细分架构类型。不是所有的 MoE 的参数量都远大于计算带宽要求。

MoE 架构的本质是模型参数分布式存储，MoE 减少计算量的代价可能是不同专家模型的参数重复和总参数量增加，这往往也意味着更大更贵的 HBM 成本。外界传言的 MoE 模型可以更小，其实是指的 MoE 模型蒸馏的 Dense 模型可以兼顾参数量和推理（Reasoning）性能。

不同应用场景对 Dense 和 MoE 模型的需求

	To B 计算场景	To C 云计算场景	To C 边缘/端侧计算场景
特点	专业领域应用多，对 RAG 高度依赖，不需要多专家	通用领域多，对检索和训练数据更新时间敏感，需要多专家	通用领域多，可不需要高性能/精度回答，不需要多专家
主力架构	行业大模型，主要是 Dense 架构	通用基础模型，主要是 MoE 或 MoA 架构	限于成本，主要是 Dense 架构
占有率	较高	较低	目前较低

按照上表的分析，基于高速存储的成本考虑，目前只有 To C 云计算场景（类似 OpenAI 的网页版服务）才会真正用上 MoE 这种多专家的模型架构。

### 3 V3/R1 训练架构的独特优势

DeepSeek 的优势不仅仅来自于其模型架构。从低比特 FP8 训练到 All-to-All 通信优化，其专用训练框架旨在尽可能提高训练的速度，以最高效率在解空间中找到较优的 MoE 参数集。

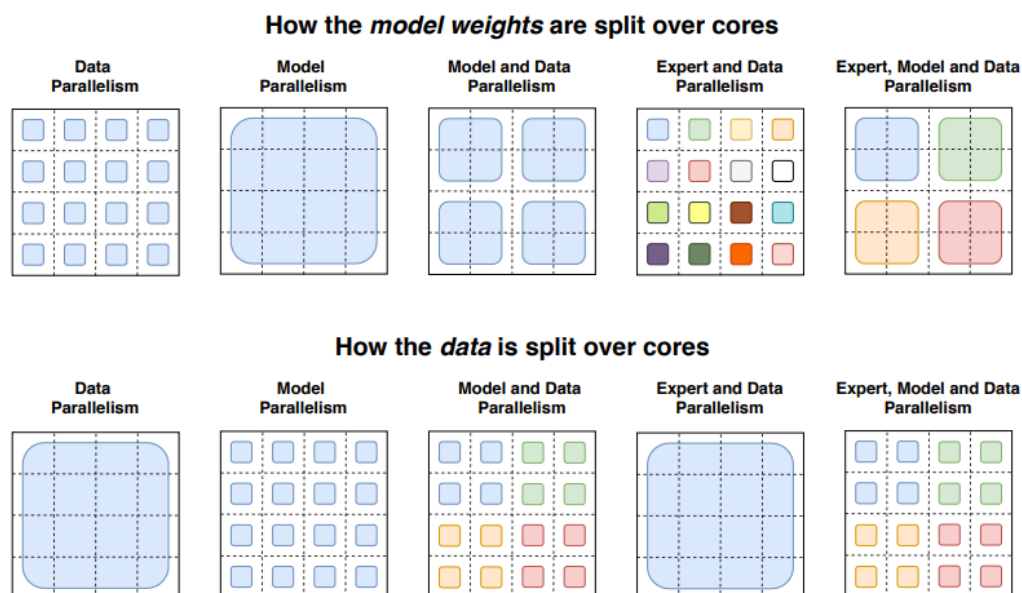
国内很多人在看 DeepSeek 团队时，更多关注了他们做 Training 的能力，但实际上 DeepSeek 的 AI Infra 能力，特别是软硬件协同优化能力，才是他们超越国内其他大模型团队的关键。

这一架构的核心优势包括：

- 1) 引入了 FP8 混合精度训练框架，并首次在超大规模大模型训练上验证了这一框架的有效性。通过对 FP8 低比特计算和存储的支持，实现了训练的加速和 GPU 内存使用的减少。
- 2) 设计了 DualPipe 算法来实现更高效的流水线并行，并通过计算-通信重叠隐藏了大模型训练过程中的大部分通信开销。
- 3) 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；对显存使用进行了优化，无需使用昂贵的张量并行即可训练 DeepSeek-V3。

#### 3.1 HAI-LLM 框架的软硬件协同设计

V3 的训练基于 DeepSeek 自研的 HAI-LLM 框架。HAI-LLM 是一个高效、轻量级的训练框架，其设计充分考虑了多种并行策略，包括 DP、PP、TP、EP 和 FSDP 的并行模式。



并行模式对比（来源：互联网）

### 3.1.1 软件层面的并行优化

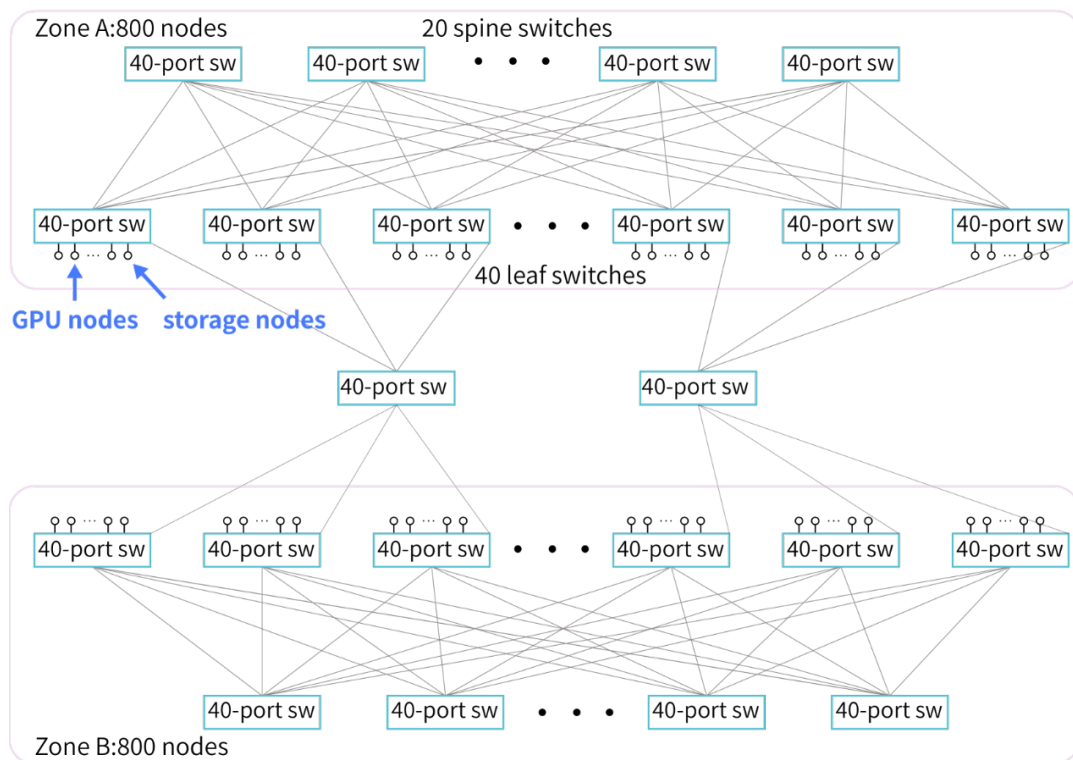
HAI-LLM 框架在软件层面所作的并行改进和效率提升如下表所示：

HAI-LLM 框架所作的并行改进（来源：中存算）

缩写	简介	DeepSeek 所做的工作或优化
TP	张量并行（Tensor Parallelism）：将模型层放置在并行执行计算的多个设备（计算芯片）上，包括逐行和逐列并行，	将 NVLink Bridge 集成到系统中，在每对 GPU 之间建立了 600GB/s 的带宽，增加 TP 效率
PP	流水线并行（Pipeline Parallelism）：每个设备（计算芯片）都包含一部分模型层，每个训练批次分为串行的小批次以进行流水线执行	通过配置数据并行排队，规避单个节点（服务器）8 个 GPU 共享一个 IB NIC 流水线并行（PP）期间出现的网络带宽竞争，实现 GPU 的交替通信和 91% 的并行效率
FSDP	全共享数据并行（Fully Sharded Data Parallel）基于 ZeRO Stage 3 算法，对模型的参数、优化器状态和梯度分布到不同的设备（计算芯片）上。在正向传播期间，FSDP 执行 allgather 作来组装完整的参数，并正向传播完成后释放；反向传播期间，FSDP 执行 allgather 获取完整参数，并进行反向梯度计算，然后执行 reduce-scatter 以同步所有设备之间的梯度，每个设备只保留部分梯度、参数和优化器更新	基于 ZeRO Stage-3 算法实现 FSDP。将 allgather 和 reduce-scatter 通信与前向和反向传播计算重叠，拆分反向传播步骤以增强重叠，减少通信量。与 PyTorch 的 FSDP 相比，HAI-LLM 的 FSDP 将训练时间缩短了近一半
DP	数据并行（Data Parallelism）：模型和优化器的状态在多个设备（计算芯片）之间复制，数据均匀分布给所有设备进行并行计算	对 PCIe 进行工程优化，提升 DP
EP	专家并行（Expert Parallelism）：在 MoE 训练期间，MoE 模型的不同专家分布在不同的设备（计算芯片）上，由门控单元将输入的 Token 分配给不同的专家	对 PCIe 进行工程优化，提升 EP

根据 DeepSeek 的论文，V3 应用了 16 路流水线并行（PP）、跨越 8 个（服务器）节点的 64 路专家并行（EP）和 ZeRO-1 数据并行（DP）。

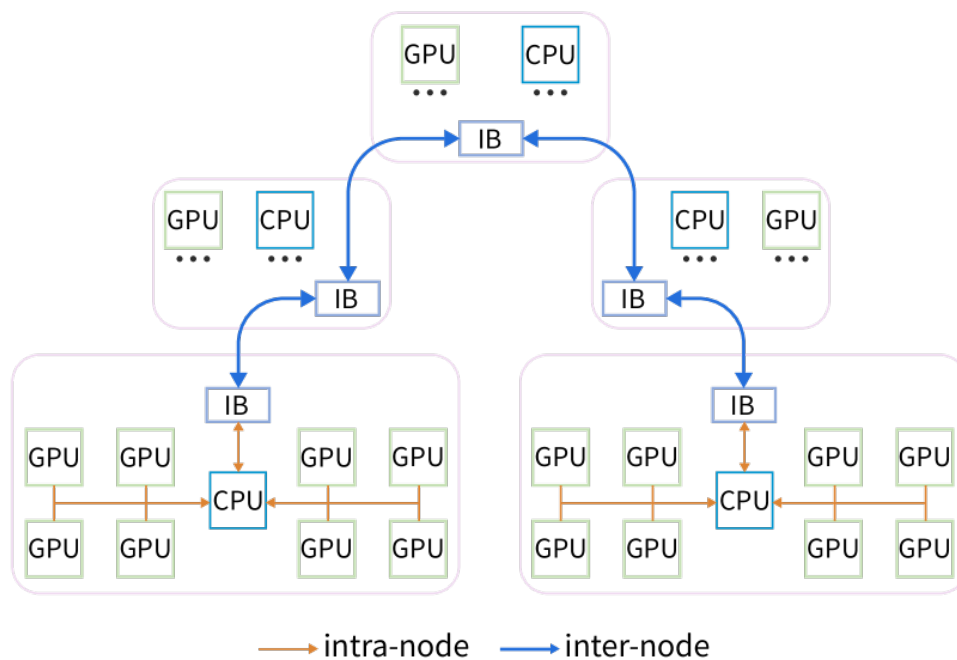
### 3.1.2 针对软件并行策略的硬件优化



低成本的万卡集群架构（来源：DeepSeek）

为了配合 HAI-LLM 训练框架（软件），DeepSeek（曾经）采用两层 Fat-Tree 拓扑+ InfiniBand（IB）作为集群架构（硬件）。（作者注：DeepSeek 目前在用的集群架构应该比这个更先进）这一集群架构的核心思路是减少互连层次，降低训练的综合成本。相对 DGX-A100 的标准万卡集群三层 Fat-Tree 的 1320 个交换机，DeepSeek 的同规模集群仅仅需要 122 台交换机，至少节省了 40% 的互连成本。

### 3.1.3 针对硬件架构的软件优化



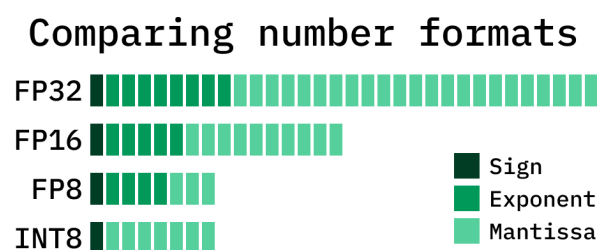
针对硬件架构优化的 HFReduce（来源：DeepSeek）

针对所采用的硬件架构特点，DeepSeek 开发了 HFReduce（针对不使用 NVLink 的方案），以执行高效的 allreduce 操作。HFReduce 会首先执行节点内 reduce，然后通过 CPU 执行节点间 allreduce，最后将 reduced 数据传输到 GPU。这样的优化需要 DeepSeek 团队对硬件互连有非常深刻的理解。

当然 DeepSeek 团队也开发了基于 NVLink 的 HFReduce with NVLink，在将梯度传递给 CPU 之前，先在 NVLink 互连的 GPU 之间执行 reduce 减作；当 CPU 返回结果时，会将数据切分并分别返回给 NVLink 连接的配对 GPU，再通过 NVLink 执行 allgather。

## 3.2 FP8 训练框架体系

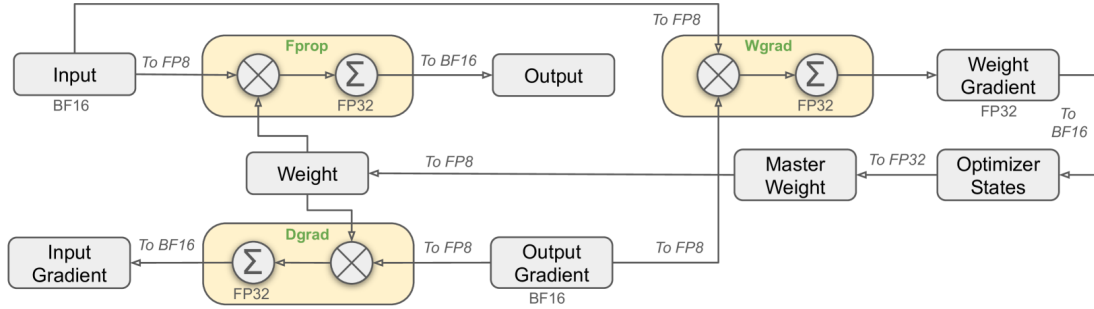
### 3.2.1 低比特训练框架的构建



FP8 与其他数据格式占据空间对比（来源：互联网）



通常的大模型训练会采用 BF16（16 位浮点）或 FP32/TF32（32 位浮点）精度作为数据计算和存储的格式，来确保较高的训练精度。相比之下，FP8 占用的数据位宽仅为 FP32 的 1/4，可以有力的提升计算速度，降低对存储的消耗。当然，FP8 也存在精度不高，容易导致训练失败的潜在问题。



FP8 训练框架局部方案（来源：DeepSeek）

DeepSeek-V3 主要使用 FP8（8 位浮点数）来提高计算速度并减少训练期间的显存使用量。为了让 FP8 更好的完成训练，DeepSeek 专门设计了针对 FP8 的训练框架体系。当然，就在撰写本文的时候，微软已经跑通了 FP4（4 位浮点数）的完整模型训练。

使用 FP8 框架进行训练的主要挑战在于精度与误差的处理。DeepSeek 为其 FP8 低比特训练框架做了以下优化：

### 1) 细粒度量化

将数据分解成更小的组，每个组都使用特定乘数进行调整以保持高精度。这一方法类似于 Tile-Wise 或 Block-Wise。对于激活，在 1x128 大小的基础上对计算数据进行分组和缩放；对于权重，以 128x128 大小对计算数据进行分组和缩放。该方法可以根据最大或最小数据调整缩放系数，来更好的适应计算中的异常值。

### 2) 在线量化

为了提高精度并简化框架，该框架在线计算每个 1x128 激活块或 128x128 权重块的最大绝对值，在线推算缩放因子，然后将激活或权重在线转化为 FP8 格式，而不是采用静态的历史数据。相对静态的量化方法，该方法可以获得更高的转换精度，减小误差的累积。

### 3) 提高累加精度

FP8 在大量累加时会累积出现随机误差。例如 FP8 GEMM 在英伟达 H800 GPU 上的累加精度保留 14 位左右，明显低于 FP32 累加精度。以 K= 4096 的两个随机矩阵的 GEMM 运算为例，Tensor Core 中的有限累加精度可导致最大相对误差接近 2%。

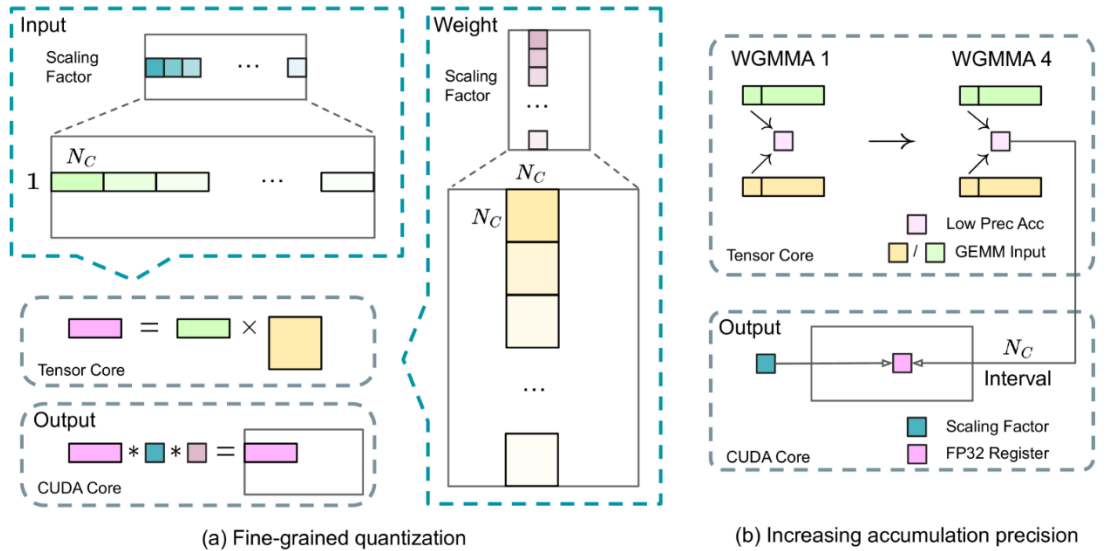
DeepSeek 将中间结果储存计算升级为 FP32（32 位浮点），实行高精度累加，然后再转换回 FP8，以降低大量微小误差累加带来的训练偏差。

### 4) 低精度/混合精度存储与通信

为了进一步减少 MoE 训练中的显存和通信开销，该框架基于 FP8 进行数据/参数缓存和处

理激活，以节省显存与缓存空间并提升性能，并在 BF16（16 位浮点数）中存储低精度优化器状态。

该框架中以下组件保持原始精度（例如 BF16 或 FP32）：嵌入模块、MoE 门控模块、归一化算子和注意力算子，以确保模型的动态稳定训练。为保证数值稳定性，以高精度存储主要权重、权重梯度和优化器状态。



细粒度量化与提高累加精度（来源：DeepSeek）

以上这些针对 FP8 训练的优化设计，都是精雕细作的工作，需要框架设计人员对 GPU 硬件架构和训练误差具有很强的整合分析能力。从 DeepSeek 的 FP8 训练框架来看，这个团队具有很强的技术和工程整合能力，已经不是单纯的大模型算法或 AI Infra 团队。

### 3.2.2 对英伟达 GPU 市场有巨大影响？

2025 年 1 月 27 日，英伟达股价暴跌近 17%，市值蒸发近 6000 亿美元，创下美国历史上单日最大市值跌幅纪录。AI 领域的明星公司普遍遭受重创：博通（Broadcom）下跌 17.4%，AMD 下跌 6.4%。微软下跌 2.1%。此外，AI 产业链的衍生板块也未能幸免，电力供应商 Constellation Energy 下跌近 21%，Vistra 下跌 28%。国内很多媒体认为这是 DeepSeek 的崛起，引起投资者对于英伟达等半导体企业估值过高的担忧。

英伟达估值是否过高不好说，毕竟 MoE 架构的发展已经展现出“存力重要性优于算力+对存储带宽瓶颈依赖下降”的倾向。但从技术角度看，DeepSeek 的大模型目前依然存在对英伟达 GPU 的路径依赖。

1) 目前英伟达仍在低比特计算方面领先。包括 DeepSeek 使用的 FP8 和微软使用的 FP4，都是由英伟达率先产品化并推向市场的。FP8 训练最早也是在英伟达内部开始验证的。英伟达之外，暂时还没有企业有这样的生态推动力和落实能力。

2) MoE 模型仍属于大模型演进的常规路径，并不会因为 MoE 模型的路径切换导致 GPU 应用不及预期。目前主要的 MoE 模型依然是基于英伟达生态构建的，在算力单价昂贵、模型性能仍需提升的现在，MoE 的应用事实上是基于有限的算力成本，进一步提升通用大模型

（以 to C 为主）性能的有效路径。这个路线早已有之，不管 DeepSeek 的影响力是否扩大，目前通用大模型都在朝着这个方向发展。过于夸大 DeepSeek 对 AI 产业的影响，只会加速美国商务部对 DeepSeek 的封禁速度，对 DeepSeek 自身反而不利。

3) DeepSeek 使用的一些训练成本优化技术属于定制化技术，其他竞品企业未必有类似的定制能力。例如前面提到的混合精度存储/计算，与模型本身的特征高度绑定，迁移起来并不简单，属于 DeepSeek 内部的定制化技术，与量化交易中的 FPGA 优化有原理类似之处。这类定制化技术一般难以简单的复制，其他企业短期内难以复盘，进行规模化成本降低的概率不高。有这个 AI Infra 能力的早就已经做了，没有这个能力也不会冒着成本不可控的风险贸然进入。

我们认为 DeepSeek 的 V3/R1 模型事实上为英伟达 GPU 开拓了除 Llama 开源系列 Dense 模型之外的 MoE 开源模型新大陆，等同于为苹果的 IOS 市场增加了新的免费 Killer App。

DeepSeek 本身对英伟达的股价影响，看起来更像是骆驼背上的最后一根稻草，大概不会超过以下几个因素：

- 1) 美国贸易关税风险。
- 2) B200/5090 不达市场预期的风险。
- 3) 大陆高成本 GPU（主要是 H100）算力过剩的风险。
- 4) 对大陆禁运加强的风险。

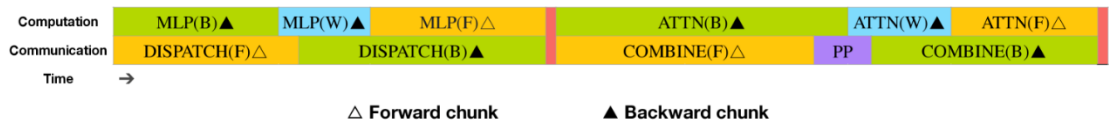
### 3.3 DualPipe 优化

V3/R1 的训练框架中引入 DualPipe 算法以实现高效的流水线并行性。

与现有的流水线并行（PP）方法相比，DualPipe 具备以下优势：

- 1) DualPipe 的流水线气泡更少，信道使用效率更高。
- 2) DualPipe 将前向和后向传播中的计算和通信重叠，解决了跨节点专家并行（EP）带来的繁重通信开销问题。
- 3) 在确保计算与通信比例恒定的情况下，具有很好的 Scale-out 能力。

DualPipe 算法将每个数据块分为四个部分：attention（图中 ATTN）、all-to-all dispatch（图中 DISPATCH）、MLP 和 all-to-all combine（图中 COMBINE）。对于后向块，attention 和 MLP 都进一步分为后向输入、后向权重。对于一对前向和后向块，针对通信和计算的过程和瓶颈进行优化。DualPipe 采用双向流水线调度，同时从流水线发送前向和后向数据，尽可能提高使用率。



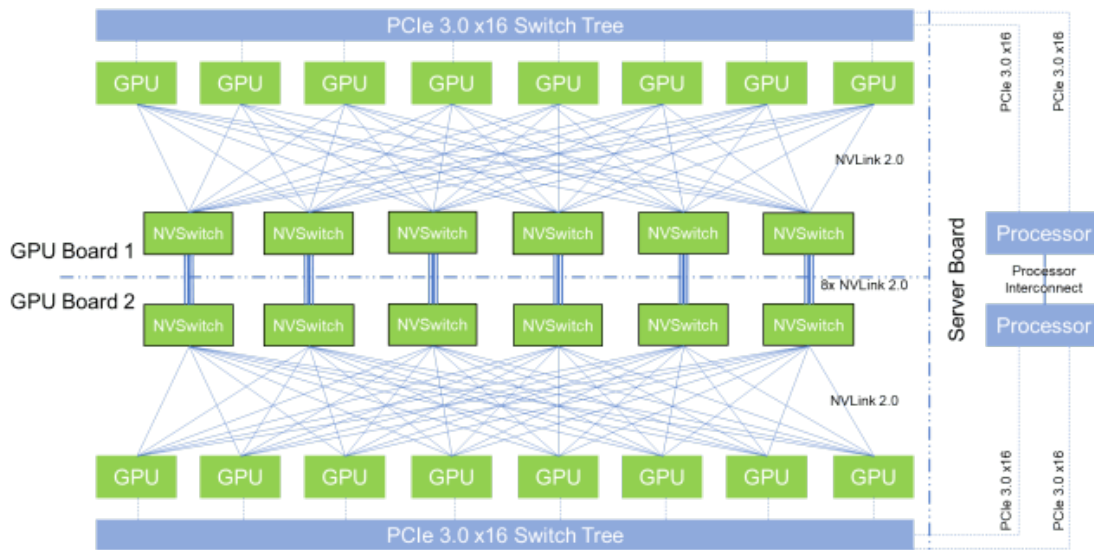
DualPipe 的流水线（来源：DeepSeek）

### 3.4 跨节点 All-to-All 通信与显存优化

V3/R1 的训练框架还定制了高效的跨节点 All-to-All 通信内核，以充分利用 IB 和 NVLink 带宽，并节约流式多处理器（SM，（Stream Multiprocessor））。DeepSeek 还优化了显存分配，以在不使用或少使用张量并行（TP）的情况下训练 V3/R1。

#### 3.4.1 对于 SM 与 NVLink 的优化

为了保证 DualPipe 的计算性能不被通信影响，DeepSeek 定制了高效的跨节点 All-to-All 通信内核（包括 dispatching 和 combining），以节省专用于通信的 SM 数量。



传统的基于 NVSwitch 的 All-to-All 通信结构（来源：互联网）

通信内核（通信 SM 控制代码）的实现与 MoE 门控算法和集群网络拓扑是按照软硬件协同的思路来进行设计的。具体来说，在集群中，跨节点 GPU 与 IB 完全互连，节点内（单台服务器内）通信通过 NVLink 完成。NVLink 提供 160 GB/s 的带宽，约是 IB 的 3.2 倍（50 GB/s）。

为了有效利用 IB 和 NVLink 的不同带宽，DeepSeek 将每个 Token（词元）的分发限制为最多 4 个节点，从而减少 IB 流量限制的影响。（本质的通点还是节点间带宽不足）对于每个 Token，在做节点间路由决策时，先通过 IB 传输到目标节点上具有相同节点内索引的 GPU；到达目标节点后，再通过 NVLink 转发到托管目标专家的特定 GPU。通过这种方式，通过 IB 和 NVLink 的通信重叠，平均每个 Token 可以在每个节点选择 3.2 名专家，而不会产生额外的 NVLink 开销。

实际算法中，V3/R1 只通过路由选择了 8 个专家，但在保持相同通信成本的情况下，该架构可以扩展到最多 13 个专家（4 个节点 x 3.2 个专家/节点）。

DeepSeek 还采用了 warp（线程束）专用化技术，将 20 个 SM 划分为 10 个通信信道。

1) 在调度过程中，(a) IB 发送、(b) IB 到 NVLink 转发、(c) 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。

2) 在合并过程中，(1) NVLink 发送、(2) NVLink 到 IB 的转发和累积、(3) IB 接收和累积也由动态调整的 warp 处理。

3) dispatching 和 combining kernel 都与计算流重叠，采用定制的 PTX (Parallel Thread Execution) 指令以自动调整通信块大小，减少了对 L2 缓存的使用和对其他 SM 的干扰。

### 3.4.2 显存节省技术

为了减少训练期间的内存占用，V3/R1 还采用了以下技术节省显存：

DeepSeek 采用的显存节省技术（来源：中存算）

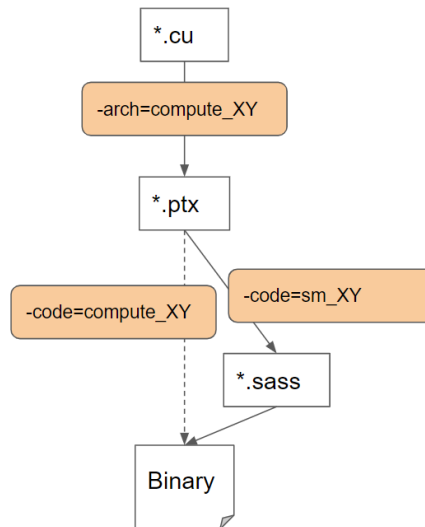
技术	方法说明	优势
RMSNorm 和 MLA Up-Projection 的重新计算	在反向传播期间重新计算所有 MSNorm 操作和 MLA Up-Projection，无需持久存储其输出激活	以算代存，充分利用 GPU 内算力充沛但缓存不足的特点
在 CPU 内存中保存指数平均数指标（EMA）	在 CPU 内存中保存 EMA，并在每个训练步骤后异步更新	把 EMA 从 GPU 显存占用改为 CPU 内存占用，释放动态存储空间
在多标记预测（MTP）中共享嵌入和输出头	使用 DualPipe 策略，将模型最浅的层（包括嵌入层）和最深的层（包括输出头）部署在相同的 PP 等级上	允许 MTP 模块和主模型之间物理共享参数、梯度、嵌入和输出头，提升显存效率

### 3.4.3 打破了 CUDA 生态壁垒？

网上很多人，看到 DeepSeek 使用了 PTX 指令，而没有直接使用 CUDA 进行 SM 编程，就认为 DeepSeek 打破了 CUDA 生态的垄断。

但实际上，

- 1) PTX 指令集也是 CUDA 完整生态的一环，是 CUDA 生态的基础。
- 2) PTX 指令比 CUDA 更底层，与英伟达的绑定比 CUDA 更深。
- 3) CUDA 是以 PTX 指令集为基础构建的，是 PTX 的外壳和泛化。
- 4) PTX 的移植比 CUDA 移植挑战更大，难以在国产 GPU 上直接移植。



CUDA 与 PTX、SASS 的层次关系（来源：互联网）

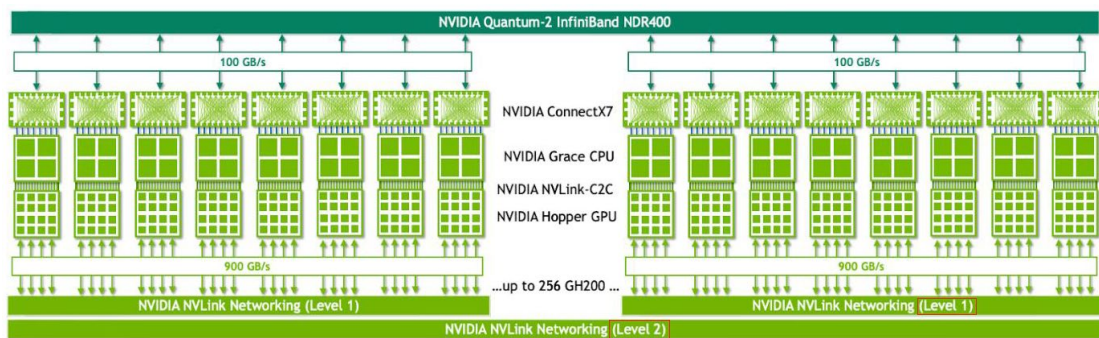
如果把 CUDA 理解为 C 语言的话，PTX 相当于 CUDA 的高级汇编语言，可以在不同的 GPU 上运行。另一种更加底层的指令集是 Streaming Assembly（SASS），与 GPU 的型号深度帮绑定。无论 PTX 还是 SASS 都是英伟达的根基，其他 GPU 厂家很难插手。

DeepSeek 在训练过程中使用 PTX，感觉就是量化码农用 C 语言写交易代码，发现优化效率不够，那么就尝试在 C 语言中嵌入汇编语言来提高硬件调度效率。难道这就等于打破了 C 语言的江湖地位？

### 3.4.4 挖了 NVLink 的墙角？

有传言说 DeepSeek 主要使用 Infiniband，以 EP（专家并行）代替 TP（张量并行），挖了 NVLink 的墙角，从而坚定的认为以 PCIe（节点内互连）+IB（节点间互连）就足以进行大模型的训练。

在这里面，NVLink 主要负责芯片间（C2C）的通信，而 Infiniband 负责节点间（服务器间）通信。如果使用 PCIe 进行 C2C 通信，带宽远不如 NVLink。



NVLink+Infiniband 互连（来源：英伟达）



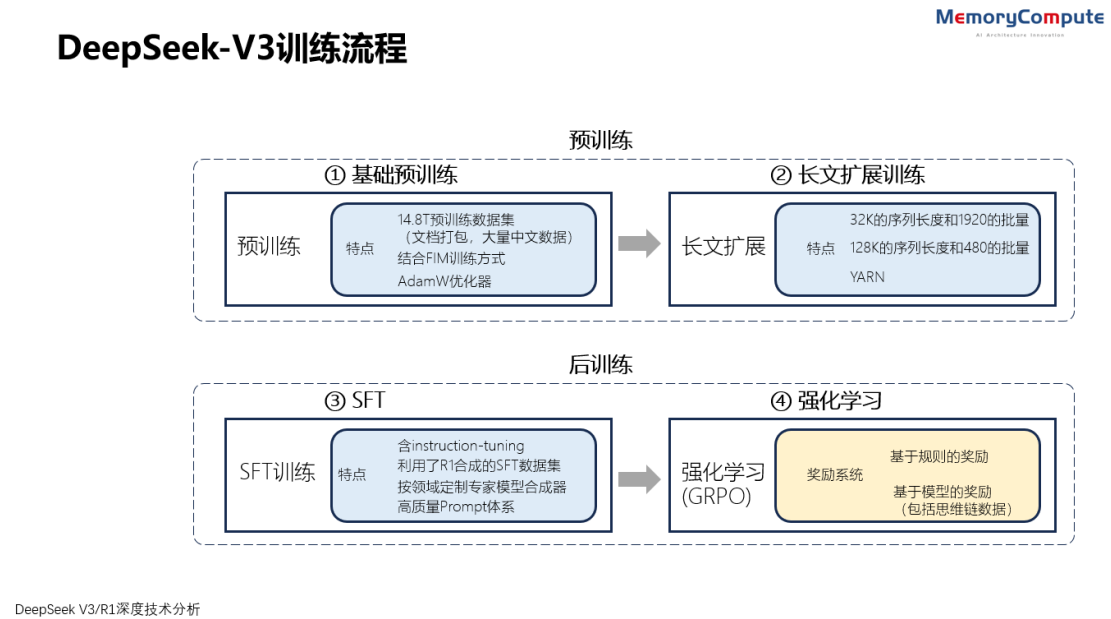
规避 NVLink 的想法很美好，但现实很骨感。按照 DeepSeek 发表的论文描述，只是在早期没有加入 NVSwitch 的时候用过 PCIe+InfiniBand 实现 HFReduc，当具备 NVSwitch 后就立刻增加了 HFReduc with NVLink。特别是在 V3 的论文中，明确写了针对 NVLink 信道加入了定制 PTX 优化，好让更多的有效 SM 参与计算。

这就好比学校的教学高楼里没有大电梯，怕楼梯上孩子太多出危险，就先用“算法+楼梯”代替，拿到“算法许可”的小孩才能到不同楼层去报道。但不能说这样就挖了“电梯”的墙角，卡住了“电梯”的脖子。一个高效的训练系统，依然需要大量的 C2C 或 D2D 互连实现更优的拓扑结构。咱不能因为玄奘法师能克服艰难险阻走到古印度取到真经，就认为需要反思火车飞机的重要性。

## 4 V3 的训练流程

DeepSeek 的 R1 是以 V3 为基础构建的（冷启动）。如果想深入理解 R1 的训练，就要先看 V3 的训练流程。V3 的训练包括预训练（含基础预训练和上下文长度扩展）、后训练三个阶段。

在预训练阶段后，对 DeepSeek-V3 进行了两次上下文长度扩展，第一阶段将最大上下文长度扩展到 32K，第二阶段进一步扩展到 128K。然后在 DeepSeek-V3 的基础模型上进行包括有监督精调（SFT）和强化学习(RL)在内的后训练，使其更贴近人类的偏好。



DeepSeek-V3 训练流程（来源：中存算）

### 4.1 V3 的基础预训练

DeepSeek-V3 总共包含 671B 参数，其中每个 Token 激活了 37B。在路由专家中，每个 Token 激活 8 个专家，并确保每个 Token 最多发送到 4 个节点，以减小通信资源的浪费。多 Token 预测（MTP）深度设置为 1，即除了下一个 Token 之外，每个 Token 还将预测一个额外的

Token。

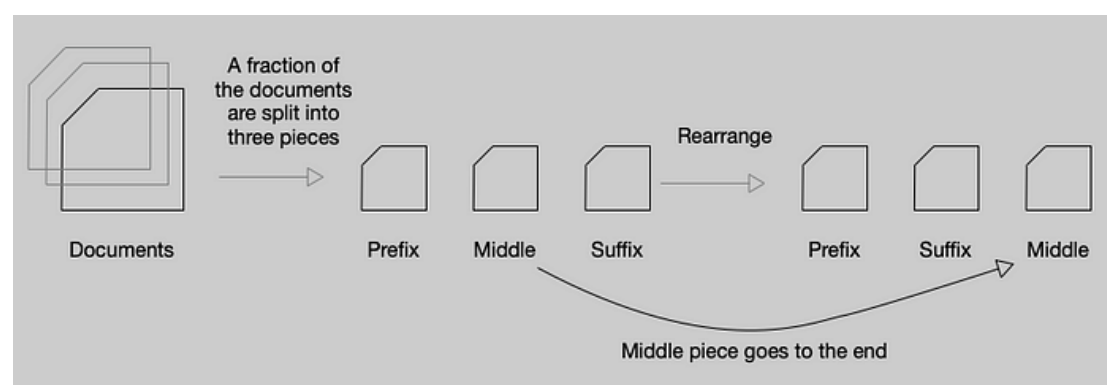
在 14.8T 预训练数据集结构上，V3 采用了以下策略：

- 1) 提高数学和编程样本的比例来优化预训练语料库，以提升推理能力。
- 2) 基于中国互联网可用的语料库整合了更多的中文数据。（这也是 V3 能玩中文梗的关键）
- 3) 将多语言覆盖范围扩展到英文和中文之外。
- 4) 优化数据处理/过滤算法，在保持语料库多样性的同时最大限度减少信息冗余。

过滤掉有争议的内容，减少特定区域文化引入的数据偏差

- 5) 通过文档打包，减少在短文本块的训练浪费，同时在训练过程中没有使用交叉样本注意力屏蔽。

高质量的数据结构与数据投喂顺序，其实是大模型性能提升的关键。可惜 DeepSeek 并没有给出预训练数据更具体的构建方法。



Fill-in-Middle 方法（来源：互联网）

V3 的训练采用前缀-后缀-中间（PSM）框架来构建 FIM 训练数据。Fill-in-Middle（FIM，中间补全）是一种针对代码补全能力的预训练方式，模型在训练过程中学习使用上下文的语境来预测文段中间缺失的部分。FIM 策略不会损害下一 Token 预测（NTP）能力，同时可使模型能够根据上下文线索准确预测中间文本。

V3 使用 AdamW 优化器来预训练，同时避免过度拟合。

## 4.2 V3 长文扩展训练

在基础预训练后，V3 使用 YARN 技术将上下文长度，按照两阶段训练扩展到 128K，每个阶段包括 1000 步。在第一阶段，使用 32K 的序列长度和 1920 的批量来执行 1000 步训练。在第二阶段，采用 128K 的序列长度和 480 个序列的批量大小执行 1000 步训练。



## 4.3 V3 的后训练/精调

### 4.3.1 V3 的有监督精调 (SFT)

V3 的有监督精调做了以下这些事：

- 1) 梳理指令精调 (instruction-tuning) 数据集。该数据集包括 1.5M 个实例，跨多个域，每个域都量身定制的不同的数据合成方法。
- 2) 利用 DeepSeek-R1 模型合成与推理 (Reasoning) 相关的 SFT 数据集。这里很有意思，基于 R1 来 SFT V3，再基于 V3 冷启动 R1。感觉上这里有关键的训练信息没有透露，DeepSeek 应该还是留了一手。
- 3) 为特定领域（例如代码、数学或一般推理）构建量身定制的专家模型数据合成器。使用复合有监督精调和强化学习训练该专家模型。训练过程中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耦合，格式为<problem, original response>，而第二种将系统提示与问题和 R1 响应合并，格式为<system prompt, problem, R1 response>。
- 4) 建立高质量提示 (Prompt) 体系，引导模型形成自身的反馈与验证机制。同时整合了来自 R1 合成的数据，通过强化学习加强这一能力。
- 5) 对于非推理数据（例如创意写作、角色扮演和简单的问答），利用 DeepSeek-V2.5 生成数据，并通过人工注释验证数据的准确性。

### 4.3.2 V3 的强化学习

V3 的强化学习包括奖励模型与组相对策略优化 (GRPO)。

与 GPT-4 类似，V3 中奖励模型包括：

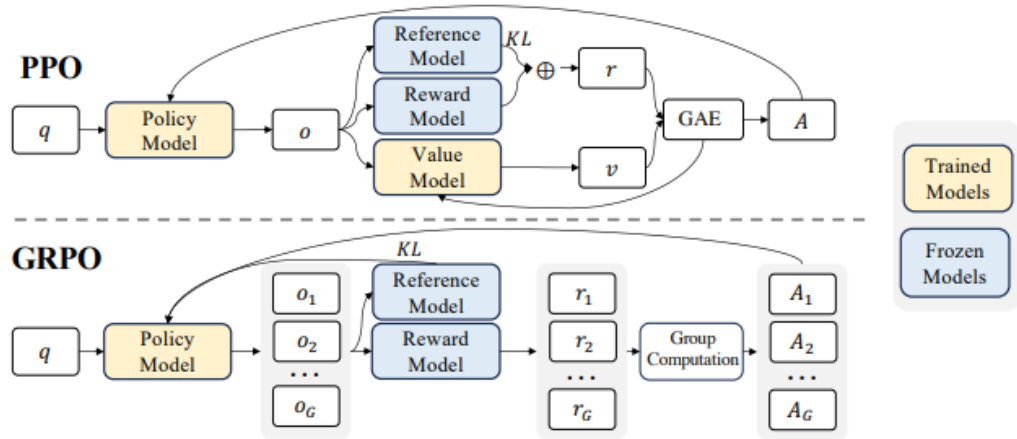
#### 1) 基于规则的奖励模型

对于可以使用特定规则验证的问题，采用基于规则的奖励模型来提供训练反馈。例如，对于 LeetCode 问题，可以利用编译器根据测试用例生成反馈。基于规则的模型验证，可以确保更高的生成可靠性。

#### 2) 基于模型的奖励模型

对于不具有收敛性的问题，依靠模型判断来进行强化学习。V3 训练中构建了特殊的偏好数据，该数据同时包括提供最终奖励结果和形成奖励的思维链，以降低特定任务中的奖励破解攻击风险。

大模型的训练通常用无监督进行预训练，然后通过有监督精调 (SFT) 进一步学习。然而 SFT 有时难以将人类的偏好显式地整合进去，这时就需要强化学习来进行精调。在以往的大模型训练中一般使用 PPO (Proximal Policy Optimization) 来形成梯度策略。PPO 的代价在于需要维护较大的价值网络（也是一个神经网络），需要占用较大的显存与计算资源。



GRPO 与 PPO 对比（来源：DeepSeek）

V3 中则采用了 DeepSeek 提出的 GRPO（Group Relative Policy Optimization）策略，只需要在一个分组内进行多个采样输出的比较，再根据比较结果选择较优的策略。GRPO 中不再需要一个显式的价值网络，从而降低了显存占用并提高了训练速度。

GRPO 的计算流程包括：

- 1) 采样一组输出并计算每个输出的奖励。
- 2) 对组内奖励进行归一化处理。
- 3) 使用归一化后的奖励计算优势函数。
- 4) 通过最大化目标函数更新策略模型。
- 5) 迭代训练，逐步优化策略模型。

## 5 R1 的训练流程

### 5.1 无 SFT 的 R1-Zero 训练

DeepSeek-R1 建立在其基础模型 DeepSeek-V3 的混合专家（MoE）架构之上，采用专家并行方式，对于任意输入，只有部分参数处于活跃状态。

作为 R1 的无 SFT 版本，R1-Zero 使用 DeepSeek-V3-Base 作为基础模型，直接使用 GRPO 进行强化学习来提升模型的推理（Reasoning）性能，根据准确度和格式进行训练奖励。

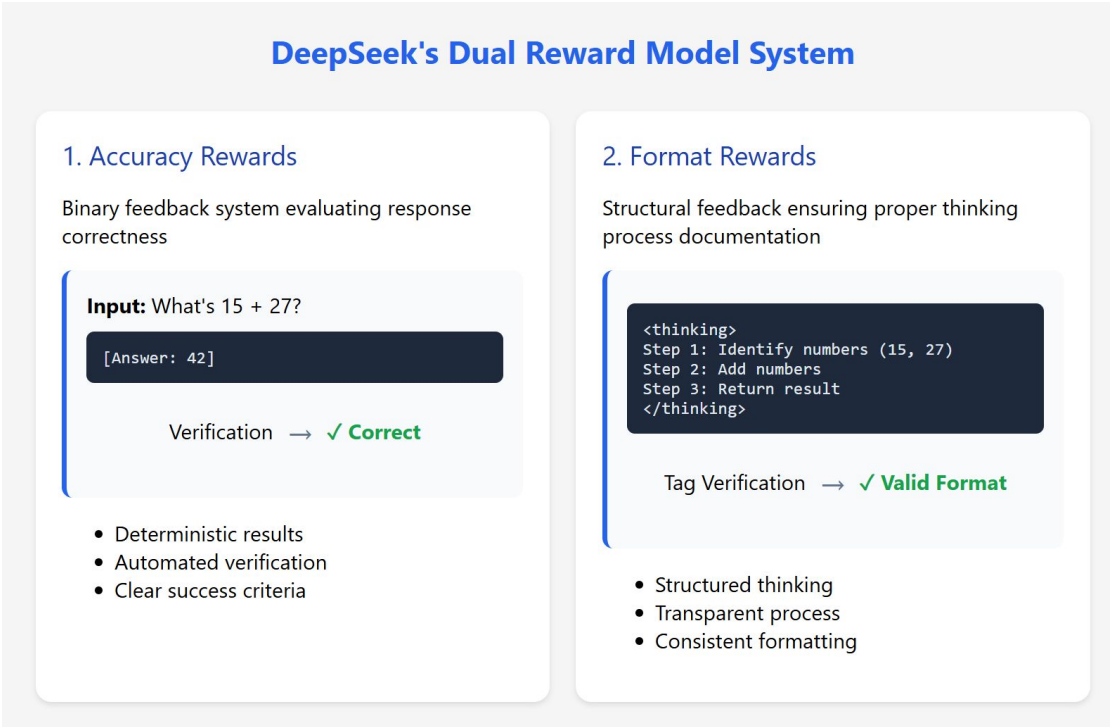
R1-Zero 的训练过程具有重要意义：

- 1) 在大模型训练领域，SFT 需要高质量的人工标注数据（标注过程一般需要很长周期、成本高，且可能因标记者的偏好而引入潜在偏差）。
- 2) 复杂的推理任务可能超出了普通人类的能力。无 SFT 的纯强化学习方法也许可以使模型能够涌现出超越传统人类思维上限的超级推理能力。

3) 无 SFT 的纯强化学习不依赖于显式标注, 允许模型使用非自然语言表征方法进行“思考”, 从而具有超越自然语言进行逻辑推理的潜力。

奖励的计算方式在很大程度上决定了强化学习训练的效果。DeepSeek-R1-Zero 的基于规则的奖励系统包括:

- 1) 准确度奖励 (Accuracy rewards)。评估响应是否正确。
- 2) 格式奖励 (Format rewards)。奖励模型将其思考过程置于 “<think>” 和 “</think>” 标签之间。



DeepSeek 的准确度奖励与格式奖励 (来源: 互联网)

通过强化学习训练, R1-Zero 形成了复杂的推理能力, 包括反思 (模型重新审视并重新评估其先前的回答) 以及探索解决问题的替代方法。这些能力并非通过常规编程或提示工程实现的, 而是大模型在强化学习环境中自发产生的能力。

根据 R1 的论文, 强化学习训练中 R1-Zero 形成的另一个关键特征是顿悟时刻 (Aha Moment)。R1-Zero 通过重新评估其初始方法学会为问题分配更多的思考时间 (更长的推理)。无需明确的教模型如何解决问题, 只需为其提供适当的激励, 模型就会自主形成解决问题的策略。这也说明强化学习有潜力解锁新的智能水平, 为未来更自主、更具适应性的模型铺平道路, 提供了形成超级智能的可能路线。

与 OpenAI 的 GPT-4 相比, DeepSeek-R1-Zero 在推理任务上表现出了显著的改进。例如, 在 AIME 2024 基准 (推理能力测试) 上, DeepSeek-R1-Zero 的性能从 15.6% 跃升至 71.0%, 这表明 R1-Zero 的无 SFT 推理训练方法是有效的。

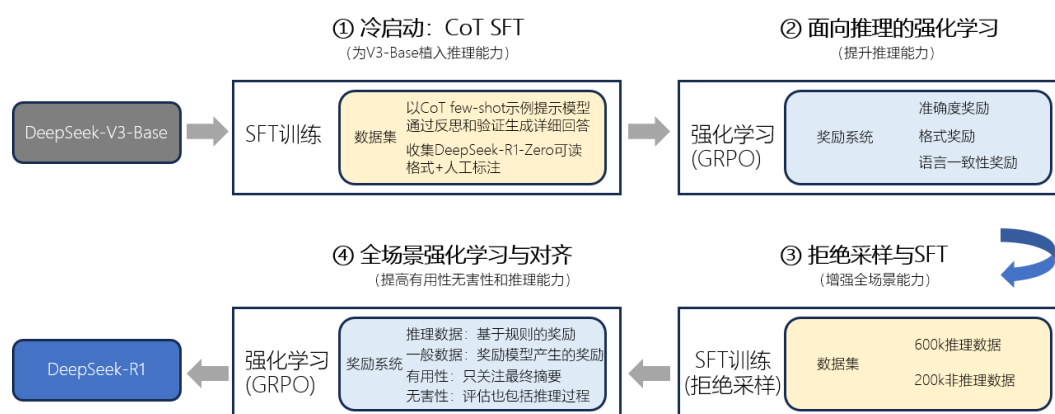
## 5.2 DeepSeek-R1 的训练流程

DeepSeek-R1 的训练过程分为 4 个阶段，包括使用数千高质量 CoT 示例进行 SFT 的冷启动，面向推理的强化学习，通过拒绝抽样的 SFT，面向全场景任务的强化学习与对齐。

两个 SFT 阶段进行推理和非推理能力的能力植入，两个强化学习阶段旨在泛化学习推理模式并与人类偏好保持一致。

### DeepSeek-R1训练流程

MemoryCompute  
AI Architecture Innovation



DeepSeek V3/R1 深度技术分析

DeepSeek-R1 训练流程（来源：中存算）

### 5.2.1 冷启动 (Cold Start): CoT SFT

与 R1-Zero 不同，R1 首先基于 DeepSeek-V3-Base 进行有监督精调（SFT），以克服强化学习的早期不稳定。DeepSeek 认为这种基于人类先验知识冷启动并进行迭代训练的方式更适合推理模型。

由于这一训练阶段主要采用 CoT 数据，我们更喜欢将其称为 CoT SFT。

为构建少量的长 CoT 数据，DeepSeek 探索了几种合成方法：使用长 CoT 的 few-shot 提示作为示例，直接提示模型通过反思和验证生成详细回答，以可读格式收集 DeepSeek-R1-Zero 输出，并通过人工标注员的后处理来完善结果。在此步骤中收集了数千个冷启动样本以进行精调。

其中可读模式指为每个回答在末尾包含一个摘要，并过滤掉不易阅读的部分。其输出格式为 `|special_token|<reasoning_process>|special_token|<summary>`。

### 5.2.2 面向推理的强化学习

在基于冷启动数据对 V3-Base 精调后，采用与 R1-Zero 相当的强化学习训练流程，基于 GRPO 进行强化学习，根据准确度和格式进行训练奖励。为了解决语言混杂问题，还在强化

学习训练中引入了语言一致性奖励，该奖励以 CoT 中目标语言单词的比例计算。

此阶段主要提升模型的推理（Reasoning）性能，特别是在编码、数学、科学和逻辑推理等推理密集型任务，这些任务涉及定义明确且解决方案明确的问题。

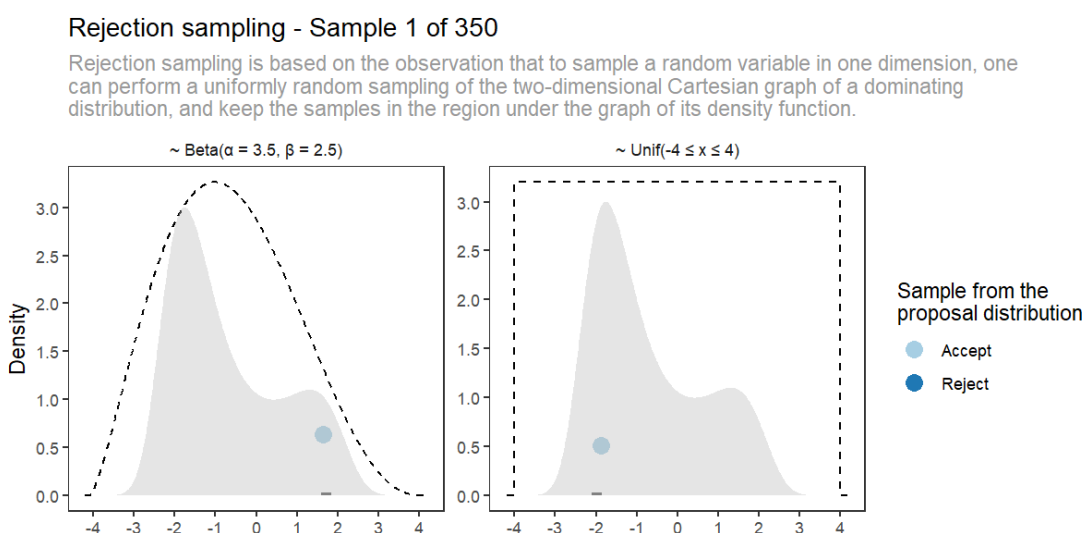
### 5.2.3 拒绝采样与 SFT

这是另一个使用标记数据的有监督精调 (SFT) 训练阶段，分批进行了两个 epoch 的精调，样本量为 800k。800k 中包括 600k 推理数据和 200k 非推理数据。

与主要侧重于推理的冷启动数据不同，此阶段结合了来自其他领域的数据，以增强模型在写作、角色扮演和其他通用任务中的能力。

拒绝采样（Rejection Sampling）提供了一种桥梁，使用易于采样的分布来近似训练真正感兴趣的复杂分布。目标响应（ground-truth）从一组生成的回答经过拒绝采样生成，其分数由奖励系统确定。

拒绝采样（Rejection Sampling）是一种蒙特卡洛方法，和重要性采样一样，都是在原始分布难以采样时，用一个易于采样的建议分布进行采样，通过拒绝原始分布之外的采样数据来获得采样结果。拒绝采样只是为了解决目标分布采样困难问题，该方法需要原始分布是已知的。



拒绝采样示意（来源：互联网）

#### 600k 推理数据的生成：

- 1) 通过从上一轮强化学习训练的检查点进行拒绝抽样，整理推理提示并生成推理轨迹（Reasoning Trajectories）。
- 2) 除基于规则奖励进行评估的数据外，还包括了基于奖励模型的 V3 判断生成数据。

- 3) 过滤掉了混合语言、长段落和代码块的思路链数据。
- 4) 对于每个提示 (Prompt)，会生成多个回答，然后并仅保留正确的响应。

#### 200k 非推理数据的生成（如写作、事实问答、自我认知和翻译等）：

- 1) 采用 DeepSeek-V3 流程并复用 V3 的部分 SFT 数据集。
- 2) 可调用 V3 生成潜在的思路链，再通过提示回答。
- 3) 对于更简单的查询（例如“你好”），不提供 CoT 回答。

### 5.2.4 面向全场景的强化学习与对齐

最后，再次进行面向全场景的强化学习和人类偏好对齐，以提高模型的有用性和无害性，并完善推理能力。此阶段还整合了来自不同管道的数据，将奖励信号与不同的提示分布相结合。

- 1) 使用奖励信号和多种提示分布 (Diverse Prompt Distributions) 的组合来训练模型。
- 2) 对于推理数据，利用基于规则的奖励来指导数学、代码和逻辑推理领域的训练过程。
- 3) 对于一般数据，采用奖励模型来捕捉复杂微妙场景中的人类偏好。即参考 DeepSeek-V3 训练流程，采用类似的偏好对和训练提示分布。
- 4) 对于有用性，只关注最终摘要，以确保重点响应对用户的实用性和相关性，最大限度减少对底层推理过程的干扰。
- 5) 对于无害性，评估模型的整个响应，包括推理过程和摘要，以识别和减轻生成过程中可能出现的潜在风险、偏见或有害内容。

至此已完成 R1 的完整训练过程，获得了具备全场景推理能力的通用 MoE 模型，上下文长度均为 128K。

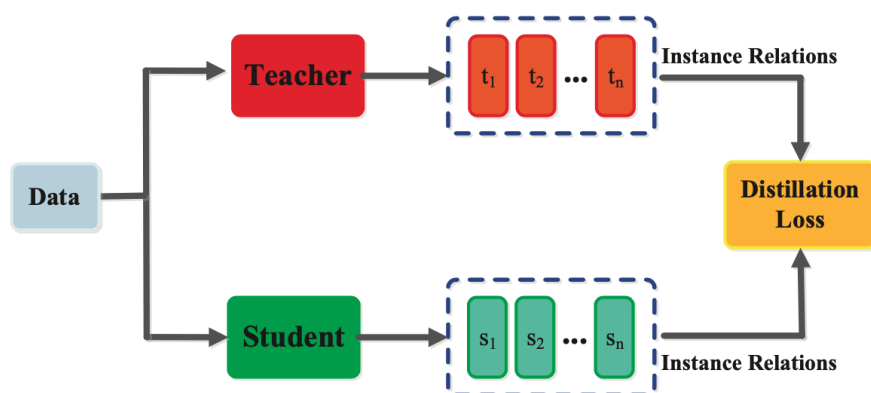
DeepSeek R1 系列模型

Model	#Total Params	#Activated Params	Context Length
DeepSeek-R1-Zero	671B	37B	128K
DeepSeek-R1	671B	37B	128K

## 5.3 从 MoE 回归 Dense（蒸馏 SFT）

尽管 MoE 架构有各种优点，特别是在通用的 to C 领域具备低成本的优势。但是 MoE 的架构特点使得其可能不太适用于专业应用场景（例如单一专家场景）和资源限制场景（例如端侧推理）。

蒸馏是将复杂的大型神经网络压缩为更小、更简单的神经网络，同时尽可能多的保留结果模型的性能的过程。此过程涉及训练较小的“学生”神经网络，通过其预测或内部表示的精调来学习模拟更大、更复杂的“教师”网络的行为。



模型蒸馏方法（来源：互联网）

为了能够将推理能力迁移到 MoE 架构不适合的场景，DeepSeek 选择 Llama 和 Qwen 系列开源大模型进行蒸馏，使相应的 Dense 模型也能获得推理能力。与使用强化学习相比，直接 SFT 更适合较小的大模型，蒸馏完成的 Dense 模型推理能力明显好于原开源模型。

DeepSeek-R1-Distill 模型（来源：DeepSeek）

Model	Base Model
DeepSeek-R1-Distill-Qwen-1.5B	<a href="#">Qwen2.5-Math-1.5B</a>
DeepSeek-R1-Distill-Qwen-7B	<a href="#">Qwen2.5-Math-7B</a>
DeepSeek-R1-Distill-Llama-8B	<a href="#">Llama-3.1-8B</a>
DeepSeek-R1-Distill-Qwen-14B	<a href="#">Qwen2.5-14B</a>
DeepSeek-R1-Distill-Qwen-32B	<a href="#">Qwen2.5-32B</a>
DeepSeek-R1-Distill-Llama-70B	<a href="#">Llama-3.3-70B-Instruct</a>

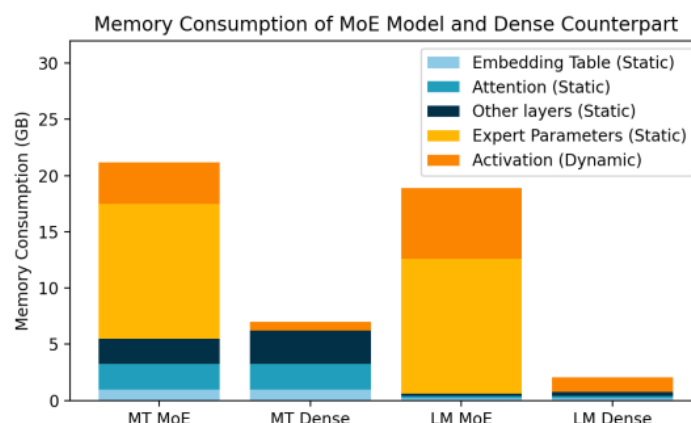
## 5.4 更大显存容量显得尤为重要？

随着 MoE 架构大模型的快速推广，产业界也有看法认为在单块 GPU 上集成更大的超过对等算力的显存或扩展存储显得尤为重要。

我们对此持不同看法，首先要看产品应用场景占有率，其次要看实际的部署方案，最后要看成本比较：

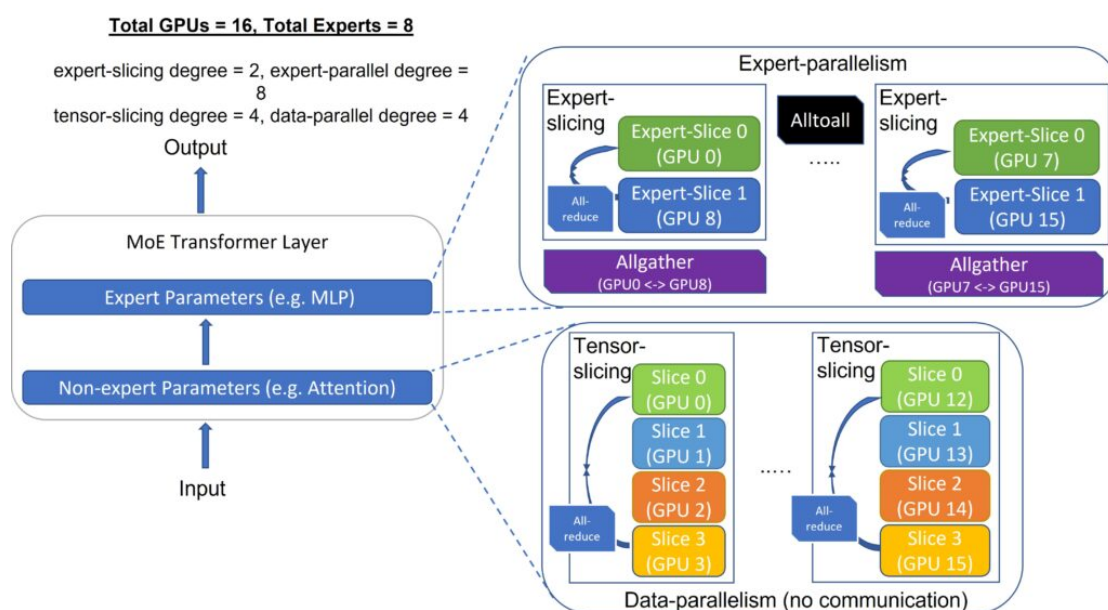
1) 根据前面分析，目前主力的专业行业应用仍是使用 Dense 模型，能部署 MoE 模型的通用 AI 巨头早已下场完成部署，从目前的应用比例来看，使用 Dense 模型的依然占据应用主体。对于 Dense 模型（实际上是单专家的特例），超过对等算力的单卡大显存或扩展存储容易形成浪费。





同样模型性能下 MoE 模型需要更大的显存（来源：Meta）

2) 根据从厂商和 V3 论文获得的实际部署方案, 为保证 MoE 部分不同专家之间的负载均衡, 会将共享专家和高负载的细粒度专家在集群的不同 GPU 做多个复制, 让 GPU 把更多的热数据（发给共享专家的）跑起来, V3 部署中每个 GPU 大概托管 9 个专家（其中一个冗余专家）。如果考虑这 9 个专家中有一个是参数最多的共享专家（否则容易出现 GPU 闲置），那么事实上每块 GPU 上的空闲细粒度专家占据的参数总和可能不超过单块 GPU 上总参数量的 1/3。

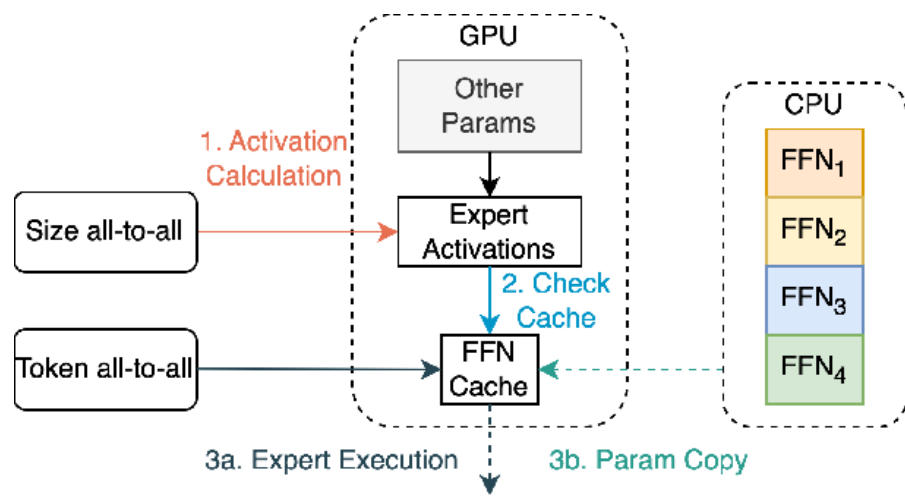


MoE 的跨 GPU 部署模式（来源：微软）

3) 从成本上看, 可能把一部分专家放到 CPU 上更划算。我们认为 MoE 上的专家可以分为高频专家、中频专家, 低频专家。高频专家（包括共享专家）和中频专家搭配部署在 GPU 上比较合适, 低频专家调度很少, 更适合放在服务器上已有的闲置 CPU 的内存上（CPU 上的标准内存比 GPU 的 HBM 便宜很多, 扩展性更好），以充分利用全服务器的算力降低综合成本。



另外，R1 自己都做 Dense 模型的蒸馏版本了，我们为何还要纠结于高于对等算力带宽的超大显存？



MoE 模型的 GPU+CPU 混合部署（来源：Meta）

## 6 结语

### 6.1 DeepSeek 的关键贡献

DeepSeek 由梁文锋于 2023 年 5 月创立，公司独立运营，并由幻方支持研发资金。这种独特的模式使 DeepSeek 能够在没有外部投资者压力的情况下开展跨越式的人工智能研发，优先考虑长期研发和价值。

成立 6 个月后，DeepSeek 于 2023 年 11 月发布的 DeepSeek Coder，随后是 DeepSeek LLM 67B，DeepSeek 逐渐踏上了基础大模型的竞争赛道。2024 年 5 月推出的 DeepSeek-V2 就呈现出 MLA 和 AI Infra 技术的明显领先优势，极具性价比，引发了中国大模型市场的价格跳水和血拼。

在 V2 发布后，我们就已经关注到 DeepSeek 在算法-硬件协同优化方面的出色表现。主编作者在 2024 年 6 月，就已为来咨询的 AI 投资圈朋友推荐 DeepSeek 的高性价比路线，盛赞 MLA 架构和 FP8 框架。

V3 和 R1 发布并开源后，DeepSeek 进一步巩固了其在 MoE 性价比和 AI Infra 的地位，并以开源模型中几乎最出色的推理性能，赢得社会的广泛关注。

对于 R1/V3 的主要创新，可以参考下表：

DeepSeek-R1/V3 的主要创新（来源：中存算）

R1/V3 的优化	R1/V3 的创新与价值	其他开源大模型
	（实现了与 OpenAI-o1-1217 相当的性能）	（相当于 Llama3.3 的性能）

软硬件结合，提高模型计算效率降低成本	提出 MLA，通过将键值 (KV) 缓存显著压缩为潜在向量来保证高效推理	采用 GQA 或 MHA，占用 KV 缓存比 MLA 大一个数量级
减少冗余，提高模型计算效率降低成本	提出 DeepSeekMoE，采用细粒度专家分割和共享专家隔离，减少冗余的专家参数	采用粗粒度专家，模型参数冗余大
改进算法，提高训练效率	提出无辅助损失策略，改善 MoE 模型训练	采用常规辅助损失策略，容易训练失败
简化算法，提高训练效率	采用 GRPO 进一步使模型与人类偏好对齐，提高训练效率	采用 PPO，导致训练效率不高
软硬件结合，提高训练效率	基于开源代码开发自有的 FP8 混合精度训练框架，提升训练效率	传统开源训练框架，以 BF16 或 FP32 为主，显存占用超过 FP8，训练速度较慢
软硬件结合，提高训练效率	DualPipe 算法来实现高效的流水线并行	默认流水线并行算法，气泡较多
软硬件结合，提高训练效率	跨节点 All-to-All 通信内核，使用 PTX 编程以充分利用 InfiniBand (IB) 和 NVLink 带宽	默认通信内核，
改进数据，提高模型性能	使用长思维链(CoT)数据进行模型训练，提升模型能力	几乎无长思维链训练

我们经过分析，认为 DeepSeek 的算法架构能力已经达到国际一线水平（例如 MLA 和 GRPO 算法），而其 AI Infra 团队的软硬件协同设计水平（例如 FP8 训练框架和基于 PTX 进行 All-to-All 通信内核优化）和自由探索，可能已暂时超越大部分国际大模型企业。基本上 DeepSeek 团队对 GPU 的性能使用率已接近技术上限，实现了在现有 GPU 体系内的软件 Sacle-up。

如果有这样超越其他大厂一个数量级的训练效率提升，估计很多大模型炼丹师梦里都要乐开花了。

## 6.2 R1 的出现是国运级的贡献吗？

有人提出 DeepSeek 所作的工作可能是一种国运级别的科技成果。

作为本文的主编作者，我个人只在小时候看过一点点梅花易数的介绍，对于推算国运的太乙神数一直没有机会了解和获得传承，不好随意讲这是不是国运级的成果。

我对最近 DeekSeek 的影响力传播看法如下：

1) 对 DeepSeek 的成果，特别是 V3/R1 开源，应有产业的高度肯定。但过度褒扬对 DeekSeek 大概率是不利的甚至是极为有害的，会导致专心做事的人要疲于应付各种俗务，也大概率导

致 DeepSeek 招致美国商务部的打压。

2) 梁文锋本人一贯低调，在创立幻方后那么多年，几乎没有出来宣传或炒作。媒体和社会如能参考 DeepSeek 创始人的个人行事风格，给更多类似 DeepSeek 的本土人才和企业予以支持，会更有利于国运。

3) 脱离实际技术分析的对 DeepSeek 的评判多数是雾里看花，少一些人与亦云的评判，多一些实干，对所有心怀理想的人都是好事。

4) 会有比 R1 更颠覆性的先进大模型出现。如同前几年的热炒 OpenAI 和 Kimi，技术总会不断进步，对于 DeepSeek 来说，还有更加重要的目标和星辰大海，R1 仅仅是海边新发现的璀璨贝壳。

5) 是否是对 OpenAI 模型的蒸馏根本不重要。学习和参考是人类社会进步的阶梯，开源更是对大模型技术进步的头号贡献。一两家闭源巨头大概率没有足够的资源储备来推动人类 AGI 的颠覆式发展，只有更多的 DeepSeek 这样的力量贡献到开源社区，才能形成合力实现超级人工智能。

### 6.3 对于国产 AI 芯片的启示

DeepSeek 的进步和成果，也给国产 AI 芯片的发展提供了一些启示。

一方面，一级市场需要升级投资逻辑，不用再崇洋媚外。事实证明纯本土的研发团队，甚至是纯本土新人团队，完全由能力做出有国际影响力的成果和产品。国内算法不再死跟着老美屁股后面，国内的 AI 芯片也大可不死跟着英伟达做传统 GPU。新的架构 AI 芯片，新的 GPU 架构，跨领域的技术融合，正形成新的产业窗口。

另一方面，DeepSeek 的技术成果，事实也凸显出算力对模型进步的重要性。DeepSeek 的算法进步速度之快，与其算力使用效率比其他团队高约一个数量级有非常密切的关系。（当然 DeepSeek 可使用的算力总量也不低）在这样的一个算法大发展契机，尽快发展新架构 AI 芯片，发展 3D 封装集成，发展片间高速互连，发展开源编译生态，抢占 MoE 模型发展期的技术红利，对中国的芯片产业就显得尤为重要。

陈巍博士组织的讨论群组：

大模型算法技术	大模型投资创业	DeepSeek 与 MoE 技术	大模型算力供需
AI 大模型 • GPT-4 算法技术讨论群-2	AI 大模型 • GPT-4 创业投资讨论群-2	DeepSeek 与 MoE 技术讨论群	AI 大模型 • GPT-4 算力供需交流群
GPGPU (GPGPU 大群组已超过 1000 人)	存算一体/存储器	DSA	EDA 大模型

GPGPU 与 先进 GPU 设计讨论群-3	存算一体与存储器 技术讨论群-2	AI 芯片与 DSA 设计 讨论群	开源 EDA 与 EDA 大模型讨论群
---------------------------	---------------------	----------------------	------------------------



请添加群助理，加入相应讨论群。添加时请注明：称呼-所在单位-要加入的群

DeepSeek 资源汇总：

<https://github.com/chenweiphd/DeepSeek-MoE-ResourceMap>