DeepSeek V3
洞察与思考

ZOMI

鸣谢

- 洞察贡献来自于华为云的 @肖洋博士

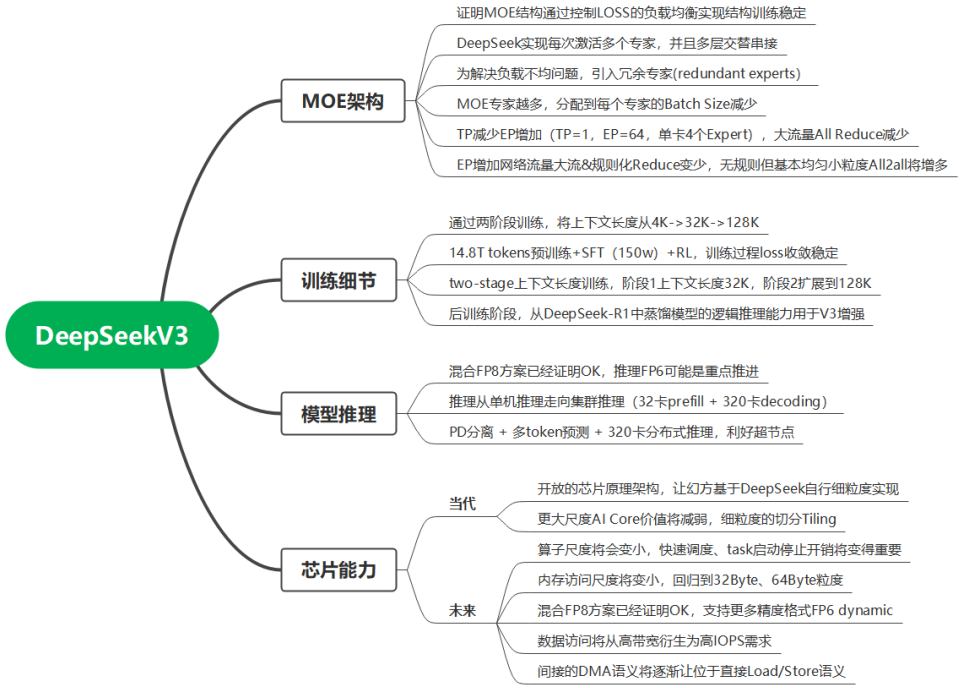
MFU 计算应用于腾讯的 @方佳瑞博士

Question 为什么要分析 DeepSeek?

1. DeepSeek V3 对大模型架构有深远影响
2. DeepSeek V3 对 AI Infra 层发展有很大启发
3. DeepSeek V3 对 NV 类的计算产业带来新的革命

Question 为什么要分析 DeepSeek?

1. MOE 架构对大模型的架构的冲击有多大?
2. 大模型训练的形式有哪些改变?
3. 推理和训练对算力需求的变化在哪里? aka 对计算领域的冲击
4. 未来AI 芯片可以有哪些改进点?



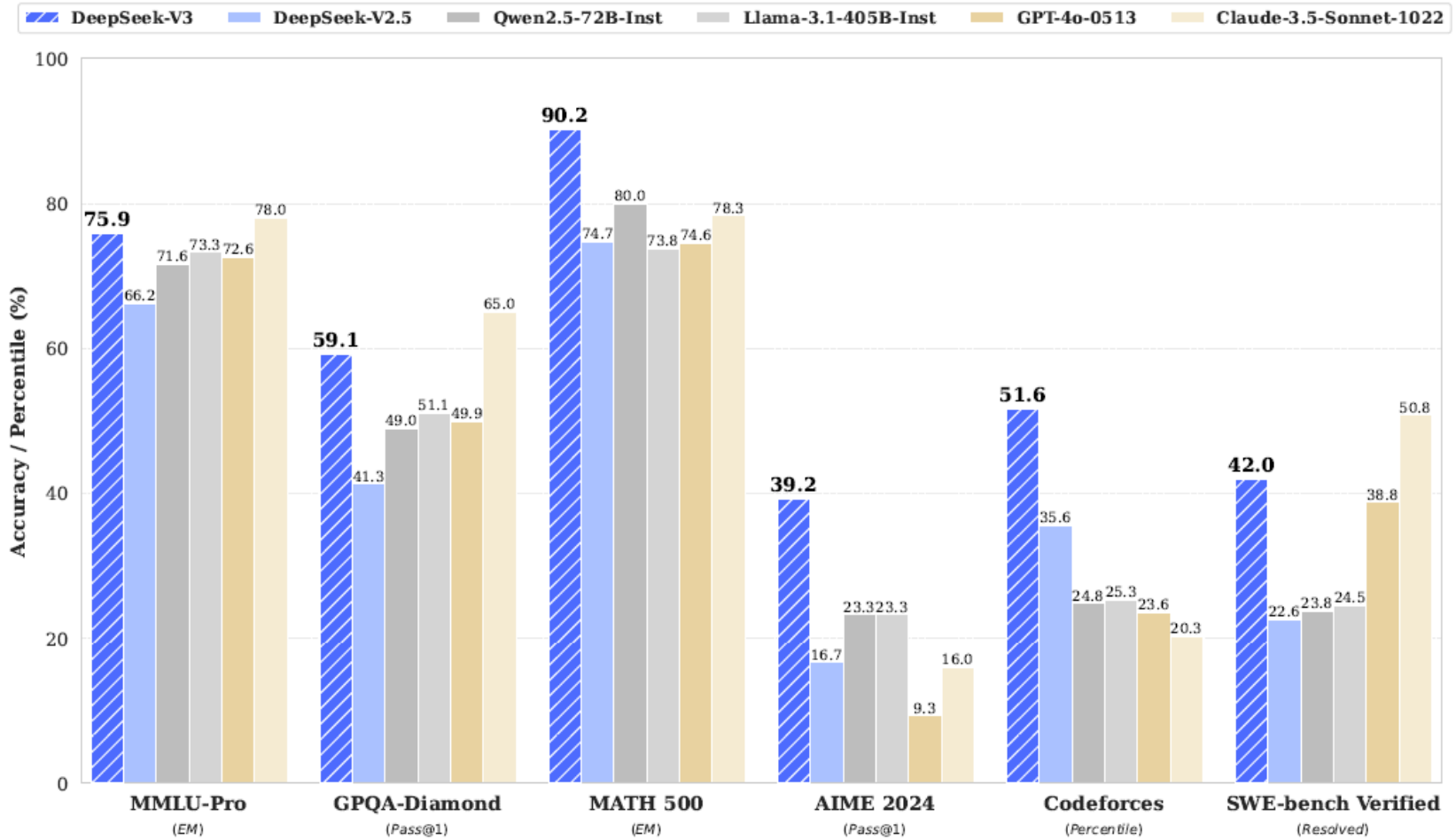
目录

01

DeepSeek V3 总览

DeepSeek V3 特点

- 模型效果好 训练过程快 推理成本低；相比同等性能开源模型训练成本成倍降低



DeepSeek V3 特点

1. **当前效果最好的开源模型**
   - 在多个benchmark上取得了所有开源模型中最好的效果
   - 尤其是在code和math方面，其效果甚至超越了闭源模型GPT-4o和Claude-3.5-Sonnet
2. **三阶段训练逐步提升模型能力**
   - 基于14.8T高质量token的pre-training
   - 逐步扩展序列长度（4K-32K-128K）
   - 包含SFT、RL、知识蒸馏的post-training

DeepSeek V3 特点

1. **引入两大训练算法创新点**
   - 提出无辅助损失的负载均衡策略和多目标预测机制
   - 训练过程极其稳定，没有出现任何loss抖动和训练回滚，同步提升最终模型效果
2. **多方位联合提升训练性能**
   - 自研大模型训练加速框架HAI-LLM，融合多项性能优化工程技巧
   - 在超大规模训练任务中首次使用FP8混合精度提升训练效率
3. **百卡部署保障推理性能**
   - 采用PD分离部署，32卡全量推理，320卡增量推理
   - TP/EP混合并行，冗余专家动态调整，保障SLO的同时追求极致吞吐

DeepSeek V3 与 Llama3.1 相比

- DeepSeekV3：2K H800 训练58天（预计）
- Llama3.1-405B：16K H100 训练54天
- 大量工程优化加速模型训练，训练成本是 Llama3.1-405B 的 1/10
- 大规模集群推理并行加速，整体吞吐相比 V2 提升 2 倍

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

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

DeepSeek V3 与其他模型相比

Feature	DeepSeek V3	GPT-4 (OpenAI)	Llama 3.1-405B (META)	Claude 3.5 Sonnet (Anthropic)
Parameters	671 billion	175 billion	405 billion	52 billion
Architecture	Mixture-of-Experts (MoE)	Transformer	Transformer	Transformer
Processing Speed	60 tokens per second	20 tokens per second	18 tokens per second	25 tokens per second
Training Dataset	14.8 trillion tokens	1 trillion tokens	1.2 trillion tokens	1 trillion tokens
Programming Ability	Excellent	Excellent	Good	Good
Open Source	Yes	No	Yes	No
Benchmark Performance	Outperforms Llama and Qwen	Best in NLP	Lower than DeepSeek V3	Lower than DeepSeek V3

02

DeepSeek V3模型与训练细节

模型结构

- **DeepSeek V3和V2模型结构大体一致**
  - 模型层数相差不大，增加模型宽度以及专家数量，调整token路由策略实现负载均衡
- **调整模型超参 引入FP8量化：**
  - 引入FP8量化减少显存需求，提升训练效率，降低训练成本
- **分阶段训练：**
  - 模型效果的提升主要依赖训练算法的升级（post-training，RL、knowledge distillation等）

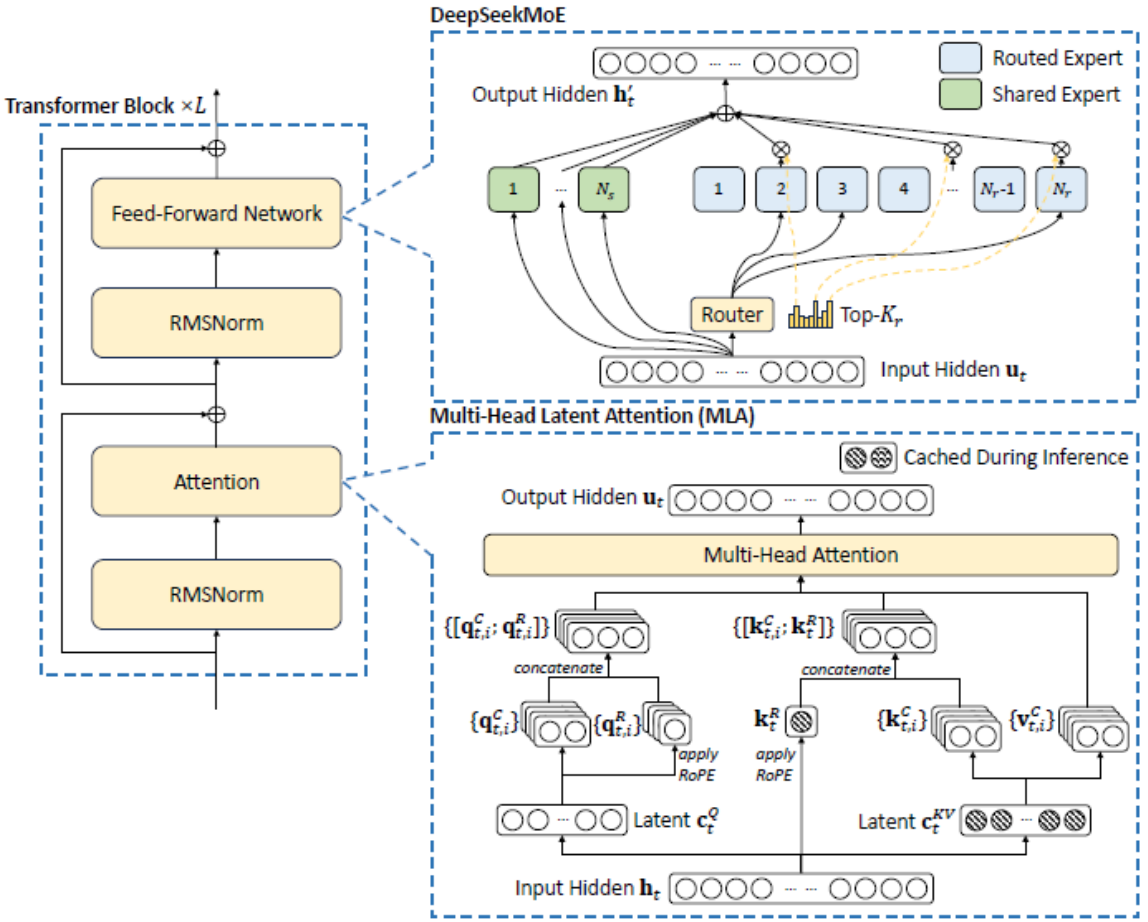


Figure 2 | Illustration of the basic architecture of DeepSeek-V3. Following DeepSeek-V2, we adopt MLA and DeepSeekMoE for efficient inference and economical training.

模型结构对比

配置项	DeepSeekV2	DeepSeekV3
总参数量	**236B**	**671B**
激活参数量	**21B**	**37B**
quantization	**None**	**FP8 (e4m3, dynamic)**
attention	MLA	MLA
hidden_size	5120	7168
num_layer	60	61
first_k_dense_replace	1	3
moe_layer_freq	1	1
num_routed_expert	160	256
num_shared_expert	2	1
num_experts_per_tok	6	8
moe_intermediate_size	1536	2048
norm_topk_prob	False	True
routed_scaling_factor	16.0	2.5
scoring_func	softmax	sigmoid
topk_group	3	4
topk_method	group_limited_greedy	noaux_tc
vocab_size	102400	129280

模型结构对比

超参数	DeepseekV2	DeepseekV3
dense层个数	1	3
moe层个数	59	58
moe层专家选择topk个数	6	8
moe层路由专家个数	160	256
moe层共享专家个数	2	1

模型细节	DeepseekV2	DeepseekV3
hidden_size	5120	7160
intermediate_size	12288	18432
moe层intermediate_size	1536	2048
moe层gate激活方式	softmax	sigmoid
moe层gate-topk模式	group_limited_greedy	noaux_tc
moe层gate-topk-group大小	3	4
vocab_size	102400	129280

MTP：Multi-Token Prediction

1. **算法来源：**
   - MTP：Multi-Token Prediction 多token预测提升模型效果
   - Mate：https://arxiv.org/pdf/2404.19737，Better & faster large language models via multi-token prediction
2. **关键信息：**
   - MTP模块仅在训练中使用，提升模型训练效果，推理阶段可以不使用MTP模块，基础模型能够独立完成正常推理
   - 参考投机采样，MTP模块也可以被重新配置用于speculative decoding，加速解码过程，降低整体时延

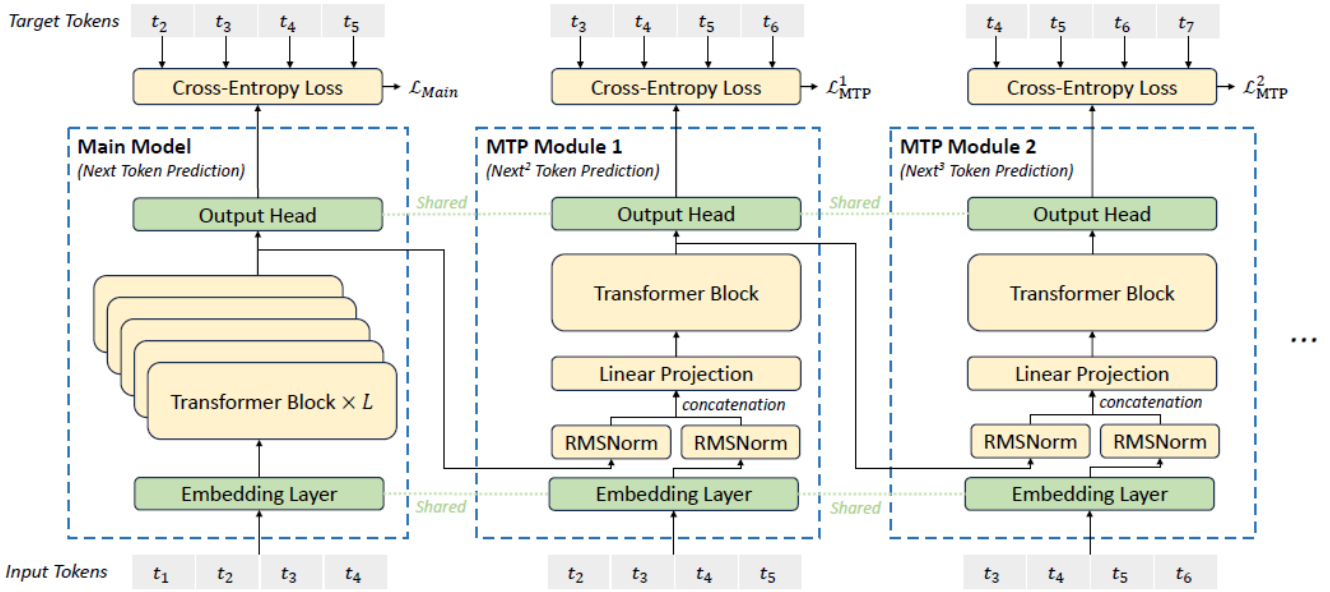


Figure 3 | Illustration of our Multi-Token Prediction (MTP) implementation. We keep the complete causal chain for the prediction of each token at each depth.

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

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

MTP：Multi-Token Prediction

① 模型结构

- 每个MTP模块共享嵌入层和输出头
- 每个MTP模块独占一个Transformer Block和MLP
- 多个MTP模块串联保持完整的因果关系链

② 训练策略

- 每个MTP模块输出预测token的概率分布
- 每个MTP模块计算对应的交叉熵损失函数
- 多个MTP模块的损失函数加权平均得到最终训练目标

③ 关键作用

- 提升每批训练数据的使用效率，强化训练信号
- 优化模型表达能力，提升next-token的预测效果
- 可参考投机采样改造MTP模块，加速推理效率

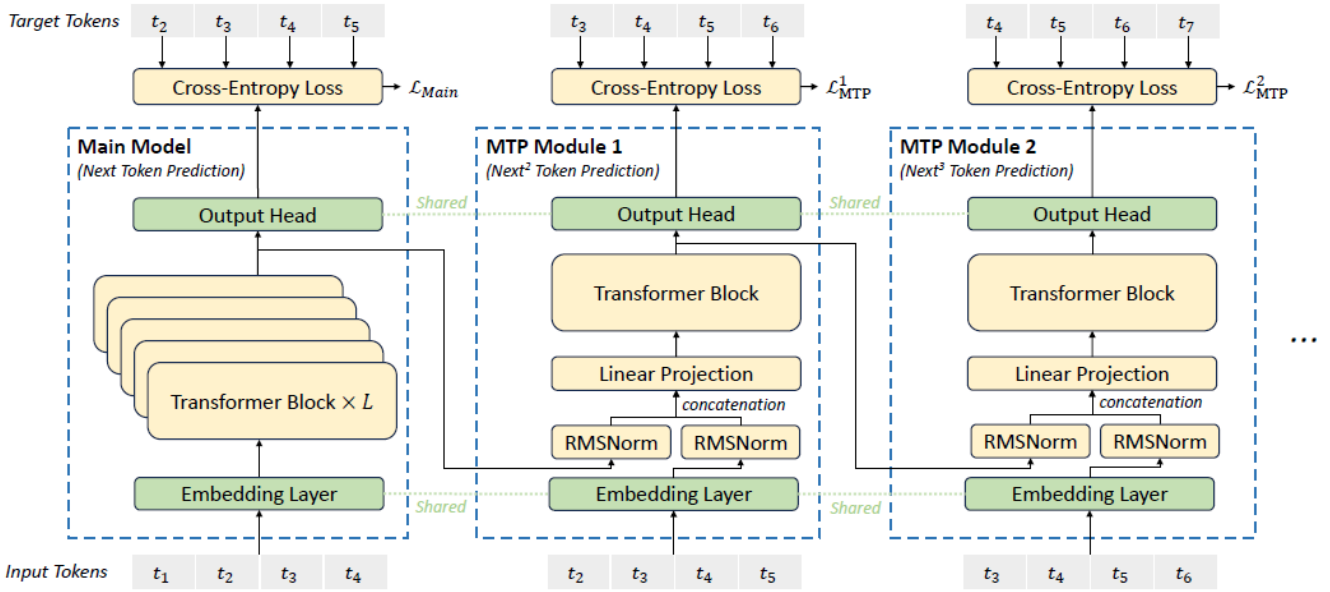


Figure 3 | Illustration of our Multi-Token Prediction (MTP) implementation. We keep the complete causal chain for the prediction of each token at each depth.

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

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

DeepSeek-V3模型训练全景，预训练占比超过90%，后训练依赖深度思考模型

预训练
(2664K GPU机时，占比95.5%)

预训练：DeepSeek-V3 使用了 14.8T tokens 训练，序列长度4K。训练过程稳定，无精度溢出带来的回滚。278.8万H800 GPU小时，训练成本约557万美元；

模型结构优化

- 更加Sparse的MOE，激活37B/671B，8+1/256专家
- MLA减少KV缓存需求（继承V2）
- 使用无辅助损失的负载均衡策略
- 使用多token预测训练目标（MTP）

计算优化

- FP8混合精度训练，提升30%吞吐
- 大量工程优化技巧保持精度和性能有效平衡，

通信优化

- DualPipe算法减少bubble
- All2ALL通信和计算掩盖

内存优化

- 重计算RMSNorm和MLA上采样，以算换存
- 将EMA权重存储在 CPU 内存，异步更新

注：其中模型结构优化、通信优化、内存优化昇腾可复用

序列长度扩展
(119K GPU机时，占比4.3%)

两个阶段上下文长度扩展：

- **第一阶段4K->32K**
- **第二阶段32K->128K**

长上下文扩展

- 通过两阶段训练，将上下文长度从4K->32K->128K

后训练
(5K GPU机时，占比0.2%)

充分利用深度思考模型DeepSeek-R1生成数据进行后训练，代码和数学能力提升明显

监督微调（SFT）

- 使用150万个指令微调实例
- 通过DeepSeek-R1模型生成推理数据

强化学习（RL）

- 使用基于规则和基于模型的奖励模型
- 采用分组相对策略优化GRPO，提升模型性能

知识蒸馏

- 从DeepSeek-R1系列模型中蒸馏推理能力，显著提升数学和编程任务表现。

03

DeepSeek V3系统架构与创新

DeepSeek V3和LLAMA对比，计算资源是后者1/10

计算集群

	DeepSeek-V3	Llama 3
GPU 类型	NVIDIA H800	NVIDIA H100
GPU 数量	2048	16000
GPU 显存		80GB HBM3
节点配置	每节点 8 个 GPU，使用 NVLink 和 NVSwitch 互连	每节点 8 个 GPU 和 2 个 CPU，GPU 使用 NVLink 互连
节点数量	256 个节点	每机架 2 节点，每 pod 192 机架，共 8 个 pod，覆盖 24000 GPU
机架规模		每机架 16 GPU；机架通过 Minipack2 交换机连接；pod 通过集群交换机连接；8 个 pod 构成 24,000 GPU 集群

网络架构

	DeepSeek-V3	Llama 3
节点内连接	• NVLink，160 GB/s 带宽	• NVLink
节点间连接	• InfiniBan，50 GB/s 的带宽	• RoCE ；NVIDIA Quantum2 InfiniBand
网络拓扑	•	• 三层 Clos 网络，覆盖 24000 个 GPU
网络细节	•	• 每机架 16 GPU，机架顶 (ToR) 使用 Minipack2 交换机 • 中间层 192 个机架通过集群交换机连接，形成 3072 GPU 的 pod • 顶层 8 个 pod 通过聚合交换机连接，形成 24,000 GPU 的集群，过订阅率为 1:7

2048 卡 H800 训练

并行策略 TP1，PP16，EP64，DP128，zero-1

2048 卡 H800 训练

① 高效率并行策略减少集群通信耗时

- 计算通信流重排，降低未掩盖通信比例
- PP组双向管道调度，减少流水线并行bubble
- 增大EP数量减少单卡权重配比，峰值显存消耗几乎不变

② MoE路由All-to-All优化设计

- 限制每个token的路由节点不超过4个
- 基于IB和NVLink的带宽差异设计多级路由流程
- 细粒度划分SM计算资源及L2缓存

③ 全栈AI能力端到端自研

- 自研大模型训练加速框架 HAI-LLM
- 自研AI训练平台 HAI-platform
- 自研软硬件协同高性能架构 Fire-Flyer AI-HPC

并行策略 TP1，PP16，EP64，DP128，zero-1

2048 卡 H800 训练

- 集群通信的实现占用GPU中的SMs资源，导致计算可用的SMs数量下降，影响整体训练性能
- 未来硬件设计中，应考虑把通信任务从计算核心SM中分离出来，利用其他协同处理器完成
- 为降低开发难度，硬件能统一IB和NVLink的网络接口，让各类网络通信操作实现更加便捷

DualPipe 实现双流并行使计算和通信几乎完全掩盖

① 细粒度的计算通信并行

- 将PP stage拆分为更细的模块，提升模块交替编排的灵活度
- 参考ZeroBubble，反向传递中权重更新和梯度传递独立操作
- 经细粒度的拆分和编排后，计算流和通信流barrier刚好可重叠

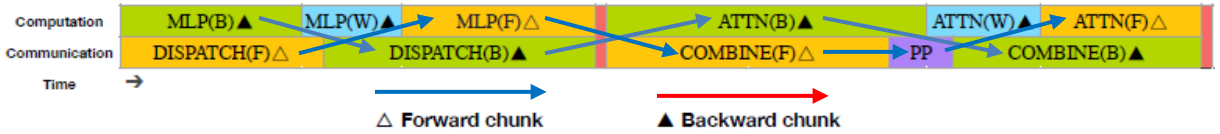


Figure 4 | Overlapping strategy for a pair of individual forward and backward chunks (the boundaries of the transformer blocks are not aligned). Orange denotes forward, green denotes "backward for input", blue denotes "backward for weights", purple denotes PP communication, and red denotes barriers.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

② 双向管道调度减少PP中的气泡

- 1F1B中每个batch拆分为1个forward和1个backward
- ZeroBubble中把backward拆分为input和weight两个部分
- DualPipe中使用对称处理，不同batch从不同device上流水

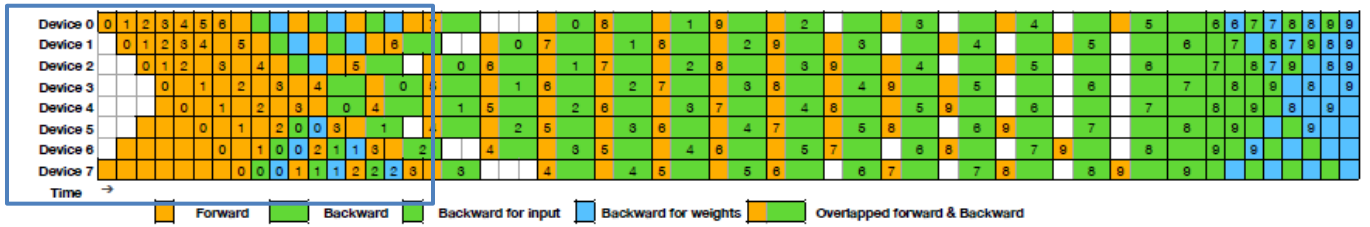


Figure 5 | Example DualPipe scheduling for 8 PP ranks and 20 micro-batches in two directions. The micro-batches in the reverse direction are symmetric to those in the forward direction, so we omit their batch ID for illustration simplicity. Two cells enclosed by a shared black border have mutually overlapped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

③ 每卡显存占用略微增大

- 双向管道训练，需要存两份参数来进行训练(Parameter 2x)
- 模型总参数量671B，每个卡上4个routed expert对应26.8B，同时考虑到PP-16和FP8量化，每个卡上显存占用为1.675GB

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

FP8 大规模训练首次成功实践混合精度

① 降低显存的同时保障模型精度

- 大部分GEMM运算采用FP8提升整体训练速度（前向传播、激活反向、权重反向）
- 对精度敏感模块仍保留原有精度（Embedding、RMSNorm）
- 权重梯度和优化器状态均采用高精度存储

② 细粒度量化技术

- 激活值采用 1x128 tile basis进行分组和量化（每个token对应128个通道）
- 权重采用 128x128 block basis进行分组和量化（输入、输出均为128通道）
- 在GEMM操作内部进行量化操作，提高了量化过程对极端值的适应能力

③ 累计精度优化

- 在Tensor Cores中执行MMA，中间结果定期转移CUDA Cores FP32 RF中进行累加
- 结合细粒度量化因子，系统可以在CUDA Cores上高效完成反量化
- 所有张量计算中统一采用E4M3格式以及在线量化方式，保障整体模型训练精度

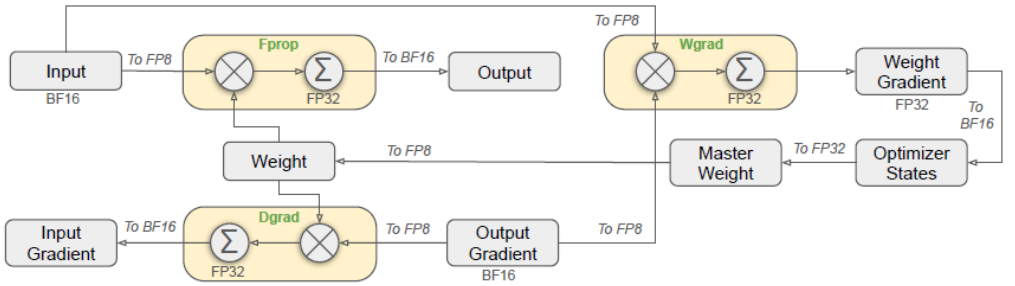
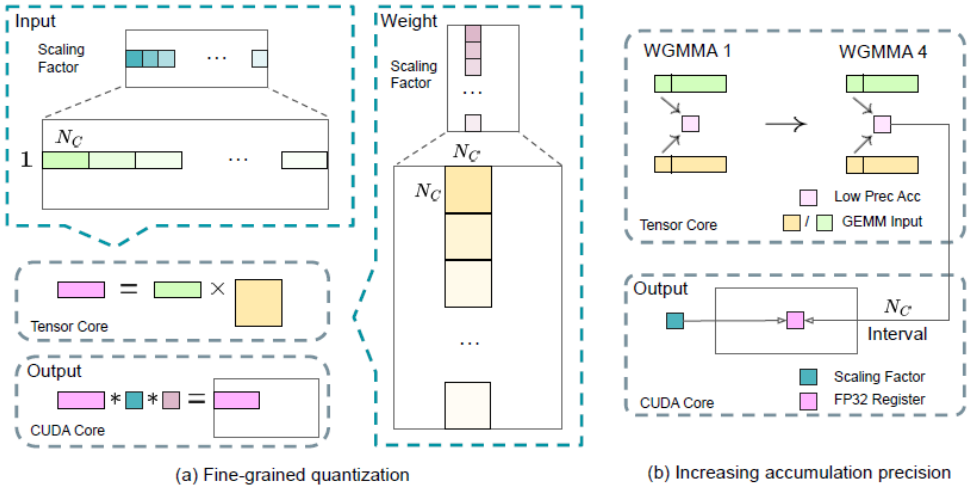


Figure 6 | The overall mixed precision framework with FP8 data format. For clarification, only the `Linear` operator is illustrated.



(a) Fine-grained quantization (b) Increasing accumulation precision

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

MFU 计算

- qk_head_dim = args.qk_nope_head_dim + args.qk_rope_head_dim

- **Q down+up pro：**
  - flops = 2 * *bs* * *seq_len* * args.dim * args.q_lora_rank
  - flops += 2 * *bs* * *seq_len* * args.q_lora_rank * args.n_heads * args.qk_head_dim
- **KV down proj：**
  - flops += 2 * *bs* * *seq_len* * args.dim * (args.kv_lora_rank + args.qk_rope_head_dim)
- **KV up proj：**
  - flops += 2 * *bs* * *seq_len* * args.kv_lora_rank * args.n_heads * (args.qk_nope_head_dim + args.v_head_dim)
- **score (Q x K^T)：**
  - flops += 2 * *bs* * *seq_len* * *seq_len* * args.n_heads * args.qk_head_dim / 2
- **score x V，由于是 causal 要除以 2：**
  - flops += 2 * *bs* * *seq_len* * *seq_len* * args.n_heads * args.v_head_dim / 2
- **Wo：**
  - flops += 2 * *bs* * *seq_len* * args.n_heads * args.v_head_dim * args.dim
- **MoE 的 forward FLOP：**
  - flops += 2 * *bs* * *seq_len* * args.dim * args.moe_inter_dim * 3
  - flops += *bs* * *seq_len* * args.moe_inter_dim #matmul
- **MLP 的 forward FLOP：**
  - flops = 2 * *bs* * *seq_len* * args.dim * args.inter_dim * 3
  - flops += *bs* * *seq_len* * args.inter_dim #matmul
- **embedding 的 forward FLOPS：**
  - flops = 2 * bs * seq_len * args.dim
- **lm head 的 forward FLOPS：**
  - flops = 2 * *bs* * *seq_len* * args.dim * args.vocab_size

MFU 计算

- qk_head_dim = args.qk_nope_head_dim + args.qk_rope_head_dim

- **Q down+up pro：**
  - flops = 2 * *bs* *
  - flops += 2 * *bs* * m
- **KV down pro**
  - flops += 2 * *bs* * dim)
- **KV up proj：**
  - flops += 2 * *bs* * *seq_len* * args.kv_lora_rank * args.n_heads * (args.qk_nope_head_dim + args.v_head_dim)
- **score (Q x K^T)：**
  - flops += 2 * *bs* * *seq_len* * *seq_len* * args.n_heads * args.qk_head_dim / 2
- **score x V，由于是 causal 要除以 2：**
  - flops += 2 * *bs* * *seq_len* * *seq_len* * args.n_heads * args.v_head_dim / 2
- **Wo：**
  - flops += 2 * *bs* * *seq_len* * args.n_heads * args.v_head_dim * args.dim
- **MoE 的 forward FLOP：**
  - n * 3
- **embedding 的 forward FLOPS：**
  - flops = 2 * bs * seq_len * args.dim
- **lm head 的 forward FLOPS：**
  - flops = 2 * *bs* * *seq_len* * args.dim * args.vocab_size

MFU 计算

- V3总共有 61 层，前 3 层用 MLP，后 58 层用MoE 来计算，另外每个 token 激活 9（1 个 share + 8 个 router）MoE Expert，context length按照4K估算，H100_peak_bf16_flops 按照 989.5 Tflops：

$$gpu\_hours = 2.664 \times 3600/1024$$

$$MFU = flops\_per\_1T\_tokens \times 14.8/(gpu\_hours \times H100\_peak\_bf16\_flops) = 37.2\%$$

MFU 计算

- 6ND+attn估算法，按照计算 attn_flosp：

$$MFU\_ref = (37 \times 6 + 3 \times attn\_flops \times 61) \times 14.8/(gpu\_hours \times H100\_peak\_bf16\_flops) = 34\%$$

- 8ND+attn估算法，按照计算 attn_flosp：

$$MFU\_ref = (37 \times 8 + 3 \times attn\_flops \times 61) \times 14.8/(gpu\_hours \times H100\_peak\_bf16\_flops) = 43\%$$

MFU 计算

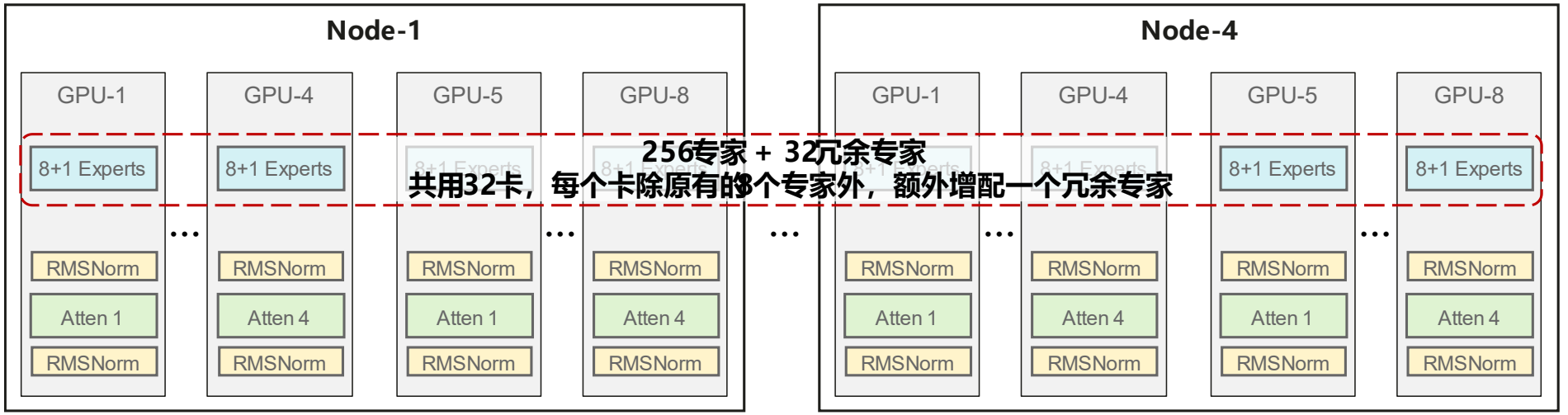
模型	Tflops	Hour	总算力	模型大小	激活量	数据量（T）	计算量（6ND）	算力利用率	MFU
deepseekV3	H800	989	2800000	9.96912E+12	671	37	14.8	3.2856E+12	32.96%
LLama3 405B	H100	989	30800000	1.0966E+14	405	405	15.6	3.7908E+13	34.57%

模型	Tflops	Hour	总算力	模型大小	激活量	数据量（T）	计算量（8ND）	算力利用率	MFU
deepseekV3	H800	989	2800000	9.96912E+12	671	37	14.8	4.3808E+12	43.94%
LLama3 405B	H100	989	30800000	1.0966E+14	405	405	15.6	5.0544E+13	46.09%

单机推理走向百卡集群推理

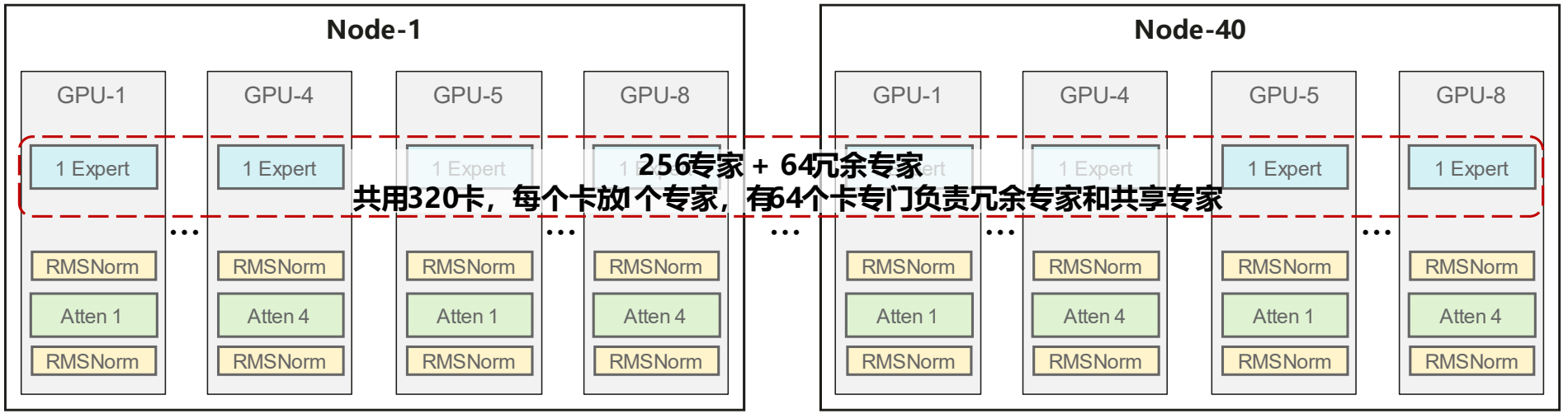
Prefill：

4机32卡



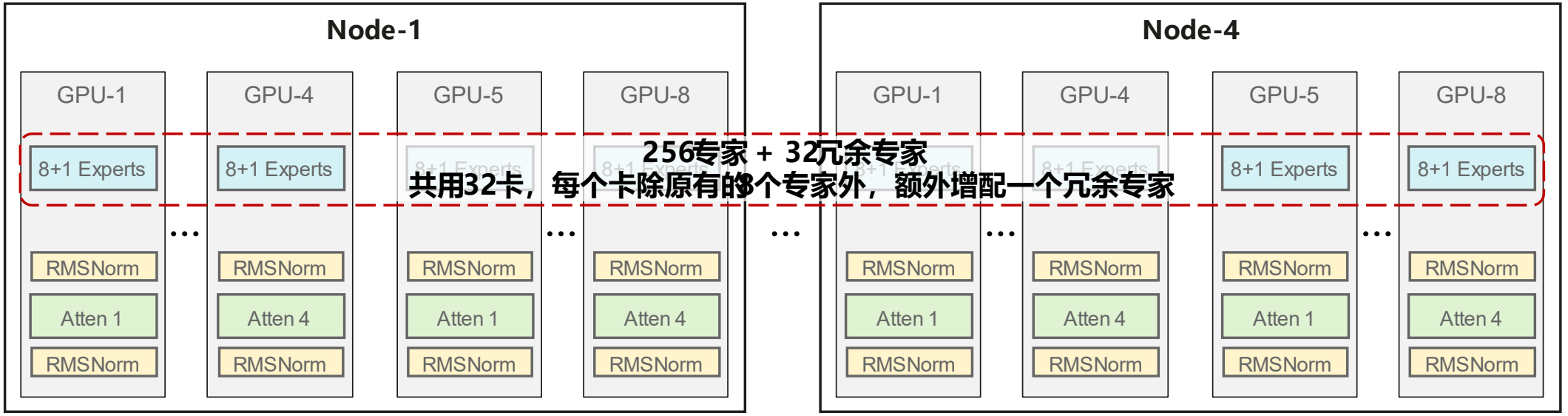
Decoding：

40机320卡



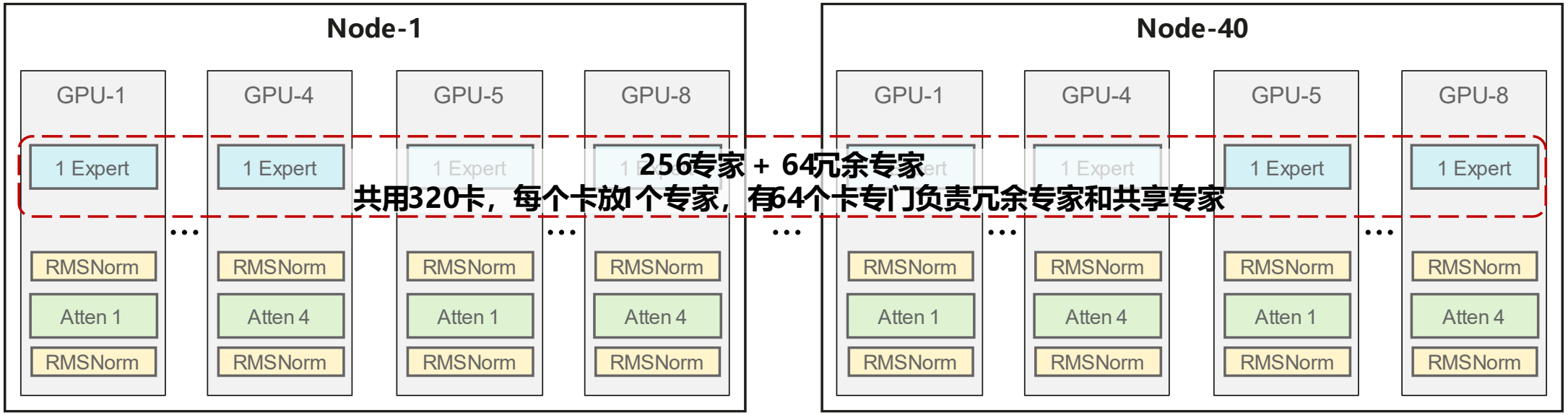
单机推理走向百卡集群推理 Prefill：4机32卡

- **Attention：**采用张量并行TP4配合序列并行SP，结合数据并行DP8
- **MoE：**采用专家并行EP32，确保每个专家能够处理足够规模的批量数据，提升计算效率
- **冗余专家：**识别高负载的专家进行复制和冗余部署，并定期调整，确保每个GPU处理token数相近
- **双批次并行：**将一个批次的attention、MoE计算和另一个批次的通信重叠，提升整体吞吐量



单机推理走向百卡集群推理 Decoding： 40机320卡

- **Attention：** 采用张量并行TP4配合序列并行SP，结合数据并行DP80
- **MoE:** 采用专家并行EP320，每个GPU仅分配一个专家，每个专家处理的token数量通常不超过256
- **冗余专家：** 与prefill类似，系统基于在线服务的专家负载统计数据，定期确定冗余专家
- **双批次并行：** 与prefill不同，decoding阶段attention耗时占比更大，将该操作与其他操作拆开并行

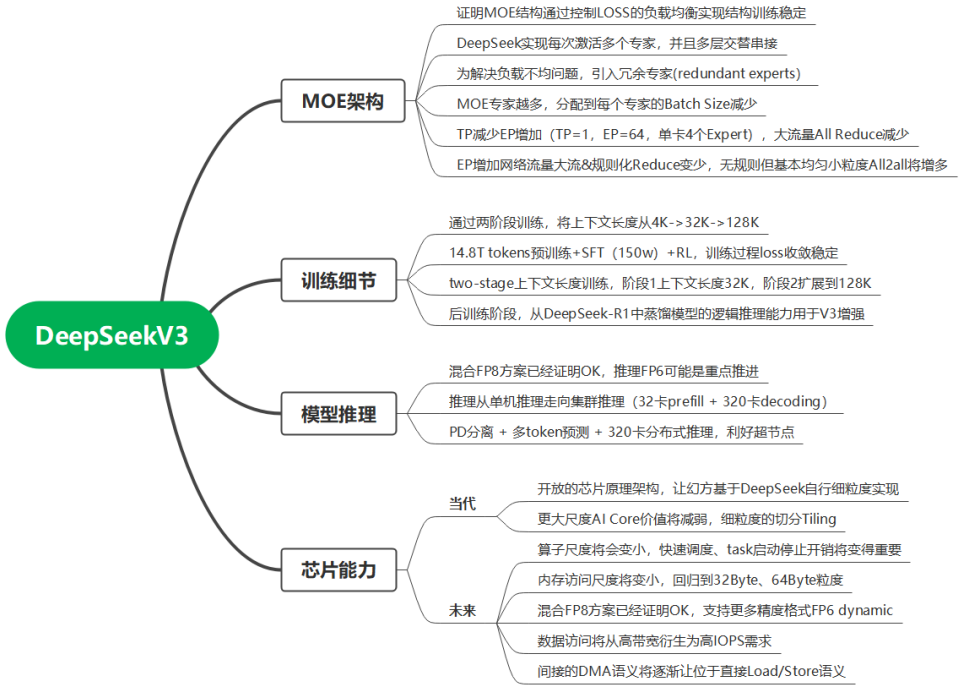


04

DeepSeek V3对业界的冲击

Question 为什么要分析 DeepSeek?

1. MOE 架构对大模型的架构的冲击有多大?
2. 大模型训练的形式有哪些改变?
3. 推理和训练对算力需求的变化在哪里? aka 对计算领域的冲击
4. 未来AI 芯片可以有哪些改进点?



MOE 架构会成为主流吗？

- DeepSeekV3
  - MOE架构
    - 证明MOE结构通过控制LOSS的负载均衡实现结构训练稳定
    - DeepSeek实现每次激活多个专家，并且多层交替串接
    - 为解决负载不均问题，引入冗余专家(redundant experts)
    - MOE专家越多，分配到每个专家的Batch Size减少
    - TP减少EP增加（TP=1，EP=64，单卡4个Expert），大流量All Reduce减少
    - EP增加网络流量大流&规则化Reduce变少，无规则但基本均匀小粒度All2all将增多
  - 训练细节
    - 通过两阶段训练，将上下文长度从4K->32K->128K
    - 14.8T tokens预训练+SFT（150w）+RL，训练过程loss收敛稳定
    - two-stage上下文长度训练，阶段1上下文长度32K，阶段2扩展到128K
    - 后训练阶段，从DeepSeek-R1中蒸馏模型的逻辑推理能力用于V3增强
  - 模型推理
    - 混合FP8方案已经证明OK，推理FP6可能是重点推进
    - 推理从单机推理走向集群推理（32卡prefill + 320卡decoding）
    - PD分离 + 多token预测 + 320卡分布式推理，利好超节点
  - 芯片能力
    - 当代
      - 开放的芯片原理架构，让幻方基于DeepSeek自行细粒度实现
      - 更大尺度AI Core价值将减弱，细粒度的切分Tiling
    - 未来
      - 算子尺度将会变小，快速调度、task启动停止开销将变得重要
      - 内存访问尺度将变小，回归到32Byte、64Byte粒度
      - 混合FP8方案已经证明OK，支持更多精度格式FP6 dynamic
      - 数据访问将从高带宽衍生为高IOPS需求
      - 间接的DMA语义将逐渐让位于直接Load/Store语义

预训练大模型跟 O1 类大模型之间的关系?

- DeepSeekV3
  - MOE架构
    - 证明MOE结构通过控制LOSS的负载均衡实现结构训练稳定
    - DeepSeek实现每次激活多个专家，并且多层交替串接
    - 为解决负载不均问题，引入冗余专家(redundant experts)
    - MOE专家越多，分配到每个专家的Batch Size减少
    - TP减少EP增加（TP=1，EP=64，单卡4个Expert），大流量All Reduce减少
    - EP增加网络流量大流&规则化Reduce变少，无规则但基本均匀小粒度All2all将增多
  - 训练细节
    - 通过两阶段训练，将上下文长度从4K->32K->128K
    - 14.8T tokens预训练+SFT（150w）+RL，训练过程loss收敛稳定
    - two-stage上下文长度训练，阶段1上下文长度32K，阶段2扩展到128K
    - 后训练阶段，从DeepSeek-R1中蒸馏模型的逻辑推理能力用于V3增强
  - 模型推理
    - 混合FP8方案已经证明OK，推理FP6可能是重点推进
    - 推理从单机推理走向集群推理（32卡prefill + 320卡decoding）
    - PD分离 + 多token预测 + 320卡分布式推理，利好超节点
  - 芯片能力
    - 当代
      - 开放的芯片原理架构，让幻方基于DeepSeek自行细粒度实现
      - 更大尺度AI Core价值将减弱，细粒度的切分Tiling
    - 未来
      - 算子尺度将会变小，快速调度、task启动停止开销将变得重要
      - 内存访问尺度将变小，回归到32Byte、64Byte粒度
      - 混合FP8方案已经证明OK，支持更多精度格式FP6 dynamic
      - 数据访问将从高带宽衍生为高IOPS需求
      - 间接的DMA语义将逐渐让位于直接Load/Store语义

大模型推理对推理的算力需求

未来芯片会往什么方向演进?

- DeepSeekV3
  - MOE架构
    - 证明MOE结构通过控制LOSS的负载均衡实现结构训练稳定
    - DeepSeek实现每次激活多个专家，并且多层交替串接
    - 为解决负载不均问题，引入冗余专家(redundant experts)
    - MOE专家越多，分配到每个专家的Batch Size减少
    - TP减少EP增加（TP=1，EP=64，单卡4个Expert），大流量All Reduce减少
    - EP增加网络流量大流&规则化Reduce变少，无规则但基本均匀小粒度All2all将增多
  - 训练细节
    - 通过两阶段训练，将上下文长度从4K->32K->128K
    - 14.8T tokens预训练+SFT（150w）+RL，训练过程loss收敛稳定
    - two-stage上下文长度训练，阶段1上下文长度32K，阶段2扩展到128K
    - 后训练阶段，从DeepSeek-R1中蒸馏模型的逻辑推理能力用于V3增强
  - 模型推理
    - 混合FP8方案已经证明OK，推理FP6可能是重点推进
    - 推理从单机推理走向集群推理（32卡prefill + 320卡decoding）
    - PD分离 + 多token预测 + 320卡分布式推理，利好超节点
  - 芯片能力
    - 当代
      - 开放的芯片原理架构，让幻方基于DeepSeek自行细粒度实现
      - 更大尺度AI Core价值将减弱，细粒度的切分Tiling
    - 未来
      - 算子尺度将会变小，快速调度、task启动停止开销将变得重要
      - 内存访问尺度将变小，回归到32Byte、64Byte粒度
      - 混合FP8方案已经证明OK，支持更多精度格式FP6 dynamic
      - 数据访问将从高带宽衍生为高IOPS需求
      - 间接的DMA语义将逐渐让位于直接Load/Store语义

Thank you

把AI系统带入每个开发者、每个家庭、每个组织，构建万物互联的智能世界

Bring AI System to every person, home and organization for a fully connected, intelligent world.

ZOMI

GitHub https://github.com/chenzomi12/AIFoundation