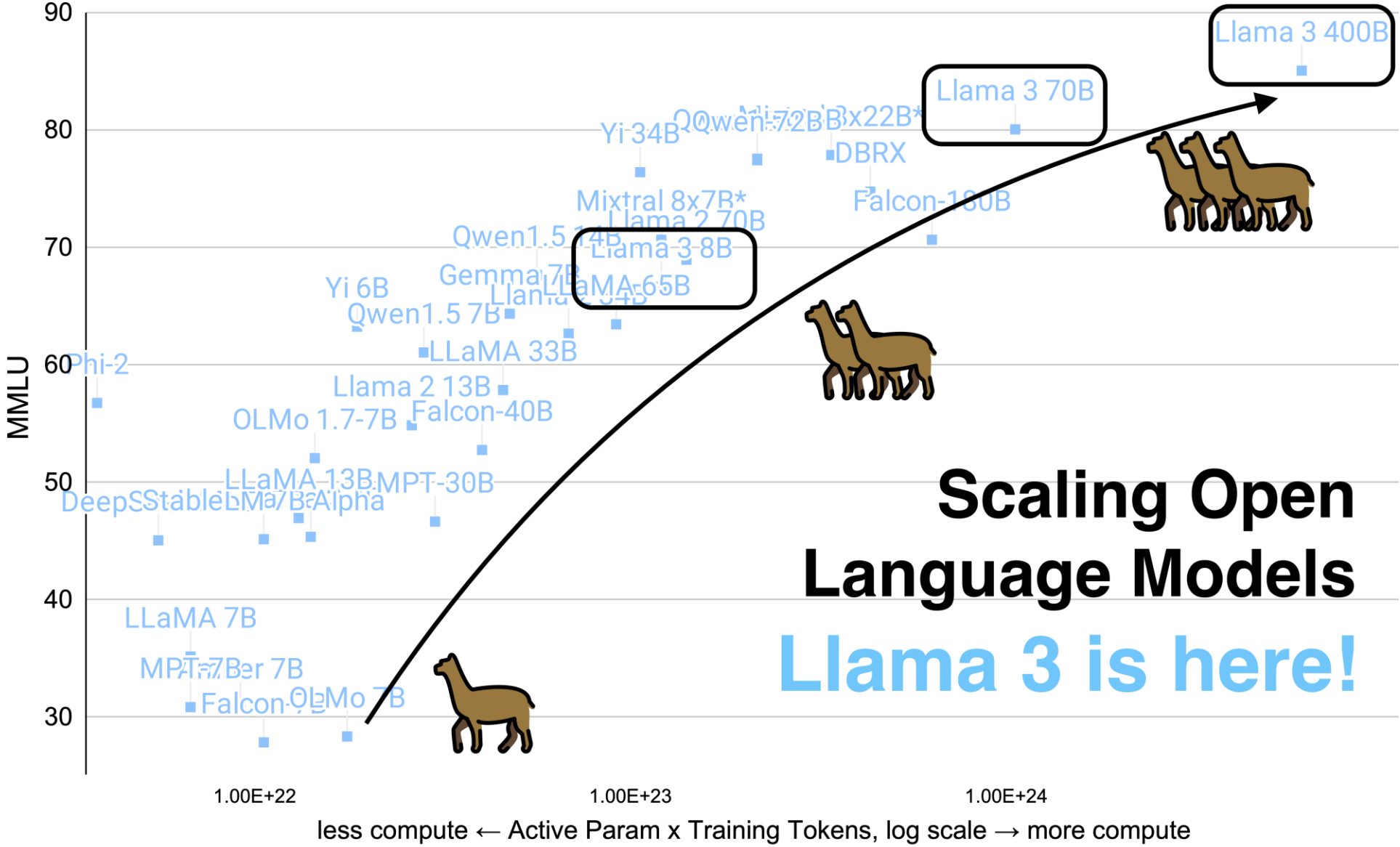
Llama 3.1 技术分析

ZOMI

关于本内容

1. Llama3.1 炸场开源：参数量最大/性能最好的开源大模型
2. Llama3.1 技术分析：从数据、模型结构、预训练和后训练进行分析
3. Llama3.1 对业界影响猜测：百模厂商冲击 && 产业思考


90
80
70
60
50
40
30
MMLU
Llama 3 400B
Llama 3 70B
Yi 34B
DBRX
Mixtral 8x7B*
Falcon-180B
Llama 2 70B
Llama 3 8B
Yi 6B
Gemma 7B
LLaMA-65B
Qwen1.5 7B
LLaMA 33B
Phi-2
Llama 2 13B
Falcon-40B
OLMo 1.7-7B
LLaMA 13B
MPT-30B
LLaMA 7B
MPT-7B
Falcon-7B
OLMo 7B
Scaling Open
Language Models
Llama 3 is here!
1.00E+22
1.00E+23
1.00E+24
less compute ← Active Param x Training Tokens, log scale → more compute

主要链接

- 模型使用 API：https://llama.meta.com/docs/model-cards-and-prompt-formats/llama3_1
- LLAMA3.1 文章：https://ai.meta.com/research/publications/the-llama-3-herd-of-models/
- LLAMA3.1 官网：https://llama.meta.com/

- 过渡到录屏

01. 模型性能

Llama 3.1 要点总结

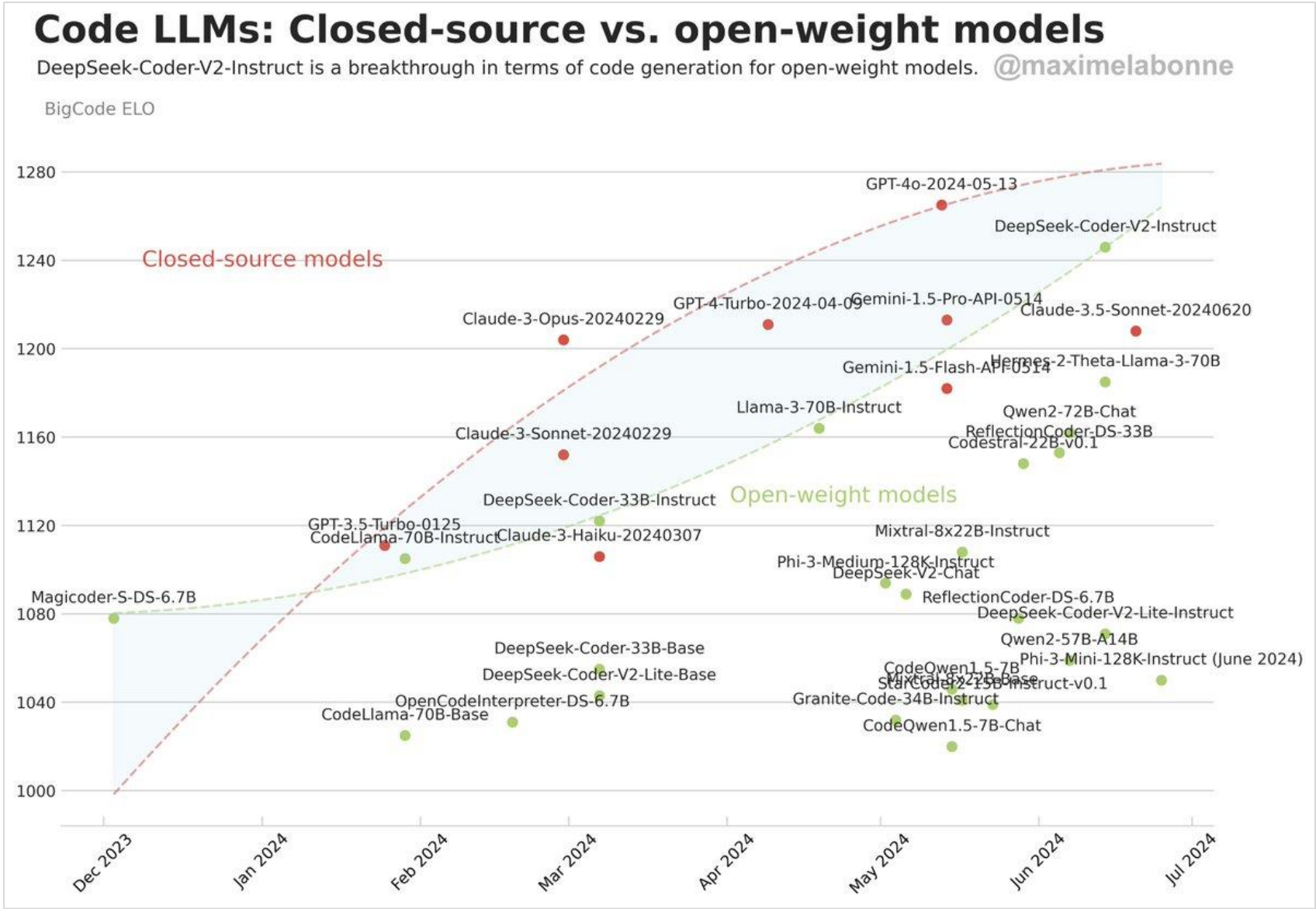
1. 8B、70B、405B 三版本，405B（~820GB） 是目前最大开源模型；
2. 405B 参数版本，性能（精度）部分评测超越 GPT4 大模型；
3. 微调后的版本使用 SFT 和 RLHF 来对齐可用性与安全偏好；
4. 引入长上下文窗口（~128K Tokens），能够处理更复杂任务和对话；
5. 支持多语言输入和输出，增强了模型通用性和适用范围；
6. 提高推理能力，特别是在解决复杂数学问题和即时生成内容方面表现突出。

模型性能分析

- Llama3.1 405B 模型性能与 GPT-4o 十分接近。
- Llama 3.1 各版本与 Open AI GPT-4o、Llama 3 8B/70B 的比较结果。即使 70B 也在多项基准超 GPT-4o。
- 首次开源模型超越 GPT4o 和 Claude Sonnet 3.5 等闭源模型，多个 benchmark 上达到 SOTA。

Category	Benchmark	Llama 3 8B	Gemma 2 9B	Mistral 7B	Llama 3 70B	Mixtral 8x22B	GPT 3.5 Turbo	Llama 3 405B	Nemotron 4 340B	GPT-4 (0125)	GPT-4o	Claude 3.5 Sonnet
General	MMLU (5-shot)	69.4	**72.3**	61.1	**83.6**	76.9	70.7	87.3	82.6	85.1	89.1	**89.9**
	MMLU (0-shot, CoT)	**73.0**	72.3△	60.5	**86.0**	79.9	69.8	88.6	78.7◁	85.4	**88.7**	88.3
	MMLU-Pro (5-shot, CoT)	**48.3**	–	36.9	**66.4**	56.3	49.2	73.3	62.7	64.8	74.0	**77.0**
	IFEval	**80.4**	73.6	57.6	**87.5**	72.7	69.9	**88.6**	85.1	84.3	85.6	88.0
Code	HumanEval (0-shot)	**72.6**	54.3	40.2	**80.5**	75.6	68.0	89.0	73.2	86.6	90.2	**92.0**
	MBPP EvalPlus (0-shot)	**72.8**	71.7	49.5	**86.0**	78.6	82.0	88.6	72.8	83.6	87.8	**90.5**
Math	GSM8K (8-shot, CoT)	**84.5**	76.7	53.2	**95.1**	88.2	81.6	**96.8**	92.3◇	94.2	96.1	96.4◇
	MATH (0-shot, CoT)	**51.9**	44.3	13.0	**68.0**	54.1	43.1	73.8	41.1	64.5	**76.6**	71.1
Reasoning	ARC Challenge (0-shot)	83.4	**87.6**	74.2	**94.8**	88.7	83.7	**96.9**	94.6	96.4	96.7	96.7
	GPQA (0-shot, CoT)	32.8	–	28.8	**46.7**	33.3	30.8	51.1	–	41.4	53.6	**59.4**
Tool use	BFCL	**76.1**	–	60.4	84.8	–	**85.9**	88.5	86.5	88.3	80.5	**90.2**
	Nexus	**38.5**	30.0	24.7	**56.7**	48.5	37.2	**58.7**	–	50.3	56.1	45.7
Long context	ZeroSCROLLS/QuALITY	81.0	–	–	90.5	–	–	**95.2**	–	**95.2**	90.5	90.5
	InfiniteBench/En.MC	65.1	–	–	78.2	–	–	**83.4**	–	72.1	82.5	–
	NIH/Multi-needle	98.8	–	–	97.5	–	–	98.1	–	**100.0**	**100.0**	90.8
Multilingual	MGSM (0-shot, CoT)	**68.9**	53.2	29.9	**86.9**	71.1	51.4	**91.6**	–	85.9	90.5	**91.6**

Table 2 Performance of finetuned Llama 3 models on key benchmark evaluations. The table compares the performance of the 8B, 70B, and 405B versions of Llama 3 with that of competing models. We **boldface** the best-performing model in each of three model-size equivalence classes. △Results obtained using 5-shot prompting (no CoT). ◁Results obtained without CoT. ◇Results obtained using zero-shot prompting.



Maxime Labonne @ ICML
@maximelabonne

New chart with Code LLMs

Based on the BigCodeBench Leaderboard, it shows the impressive performance of DeepSeek Coder V2 (236B, 21B active), which is on par with the best closed-source models.

Personally, I've switched to Sonnet and I'm very happy with its code performance.

Sunder Ali Khowaja , Ph.D., Senior Member IEEE • 2 度 关注
Researcher (Deep Learning, Responsible AI, Privacy Preservation M...
1 个月前 • 已编辑 •

Closed–source vs. Open–weight LLMs

The gap between closed–source and open–weight models is closing in terms of MMLU. LLMs are plateauing and the gap between closed vs. open is almost closed!

v/ Maxime Labonne https://lnkd.in/exq9Hghh

We now face a 6 to 10–month lag, rather than years as was the case when GPT–4 was released.

The release of GPT–4o is not a significant breakthrough with a score of 87.2% on the 5–shot MMLU (source: https://lnkd.in/etUtvuHD).

On the other side, Meta reported that Llama 3 405B's checkpoint already achieved 86.1% on the same benchmark last month.

This opens the question of having more discriminative benchmarks:

– MMLU–Pro, introduced a week ago (https://lnkd.in/e255Tn5a) is a good idea but seems to have issues with duplicates and missing information, as reported by Dorrin_Verrakai on Reddit (https://lnkd.in/eSY3f5N5).
–
@lmsysorg introduced a "hard prompts" category in the arena to evaluate models on more challenging tasks (https://lnkd.in/ej7_DeMb), but this is limited to models on the Chatbot Arena.

02. 模型技术分析

关于本内容

1. Llama3.1 炸场开源：参数量最大/性能最好的开源大模型
2. Llama3.1 技术分析：从数据、模型结构、预训练和后训练进行分析
3. Llama3.1 对业界影响猜测：百模厂商冲击 && 产业思考

看训练数据

数据准备

预训练

1. **总数据：**>15 万亿 token 数据预训练，数据日期截止到 2023 年 12 月。
2. **数据混合微调：**50% 的常识知识、25% 的数学和推理、17% 的代码数据和任务、8% 的多语言数据；
3. **多语言支持：**支持：英语、德语、法语、意大利语、葡萄牙语、印地语、西班牙语和泰语。

微调

1. **微调组成：**微调数据包括公开可用的指令数据集，以及超过 2500 万个综合生成的示例。
2. **合成数据进行微调：**使用合成数据生成 SFT 示例来训练模型。

数据处理

预训练

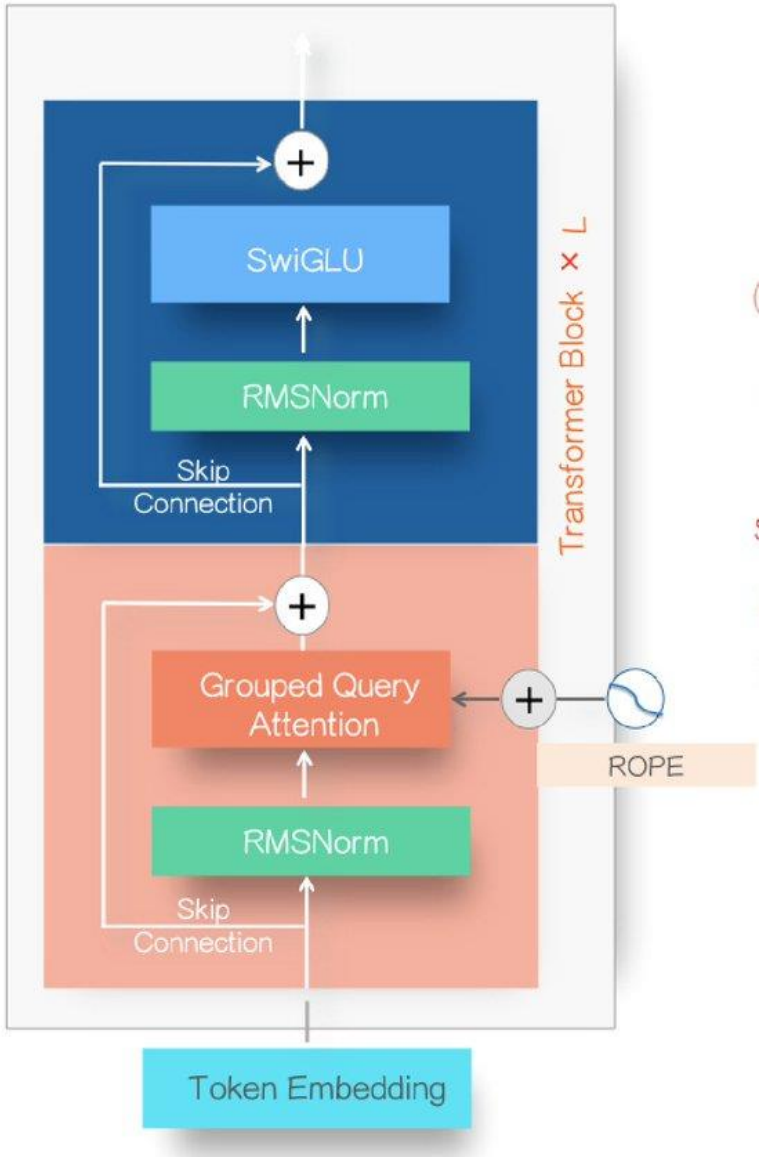
1. **预处理步骤：**使用 Roberta、DistilRoberta 和 fasttext 等 Bert 模型过滤出高质量数据。
2. **作用：**对大量数据去重和使用启发式方法，去除不良数据，保证数据安全性。

微调

1. **先大后小：**使用 405B 模型作为 70B / 8B 模型“教师模型”，从而从大型模型中提炼出适合各行各业需求的小型定制模型。

看模型架构

模型架构



- Norm：RMSNorm（Pre-Norm）
- Self Attention:GQA
- Embeding：ROPE
- FFN：SwiGLU

模型架构

- **注意力掩码 Attention Mask：**防止同一序列不同文档间出现自注意力，在标准预训练中效果有限，对长序列预训练时非常重要；
- **最大化达到 Scaling Law 算力数据比：**采用标准稠密 Transformer 架构，引入 GQA，而非 MOE，最大限度地提高训练稳定性。

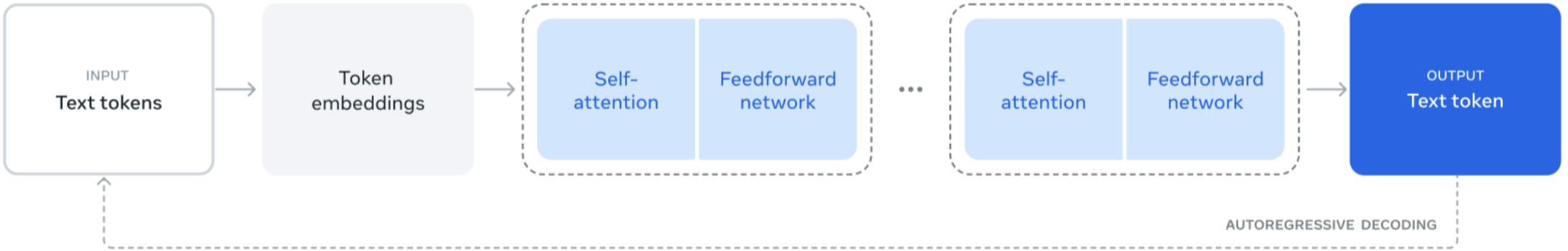


Figure 1 Illustration of the overall architecture and training of Llama 3. Llama 3 is a Transformer language model trained to predict the next token of a textual sequence. See text for details.

模型架构

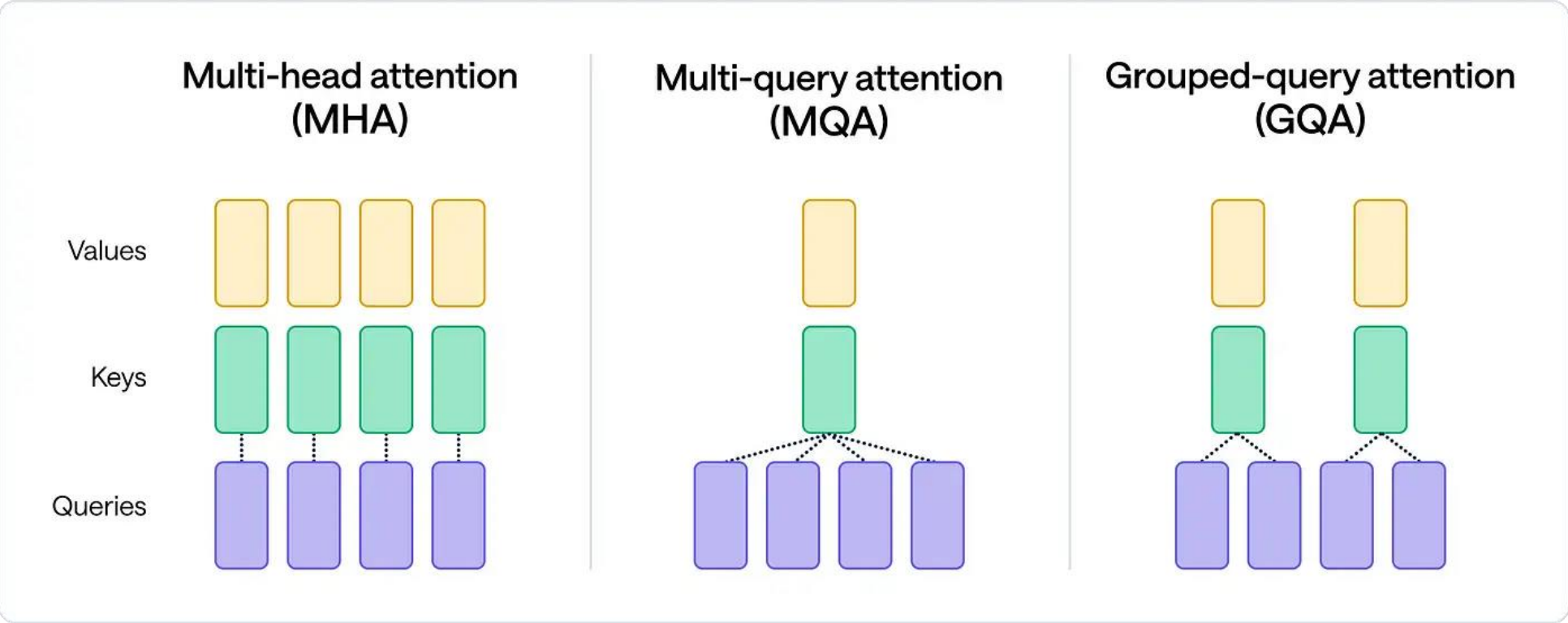
1. **分组查询注意力 GQA：**带有8个Key-Value，提升推理速度并减少解码时 KV Cache；
2. **128K Vocabulary Size：**想比 Llama2，同时提高英语/非英语压缩比率，更好支持三方语种；
3. **位置旋转编码 RoPE：**超参 θ 设为 500,000，更好支持长上下文；

	8B	70B	405B
Layers	32	80	126
Model Dimension	4,096	8192	16,384
FFN Dimension	6,144	12,288	20,480
Attention Heads	32	64	128
Key/Value Heads	8	8	8
Peak Learning Rate	3×10^{-4}	1.5×10^{-4}	8×10^{-5}
Activation Function	SwiGLU		
Vocabulary Size	128,000		
Positional Embeddings	RoPE ($\theta = 500,000$)		

- 基于数据量和训练算力，模型大小达到 Scaling Law 的算力最优比。

模型架构：GQA

- GQA 即把 MHA 分成多组，每组共用一个Key-Value。
- 从 Llama2-70b/Llama3全系列，GQA在性能与KV Cache 显存占用上获得了很好平衡。



看预训练过程

Pre-training

训练过程

1. **初始预训练（Initial pre-training）**
   - 常规预训练阶段
2. **长上下文预训练（longer contexts）**
   - 预训练后期，采用长文本数据对长序列进行训练，从 8K tokens 到 128K tokens 6 阶段逐步扩展，支持最多128K token 上下文窗口
3. **退火（Annealing）**
   - 预训练最后 4000 万个 Token，线性地将学习率 lr Annealing 至 0，同时保持上下文长度为 128K。调整了数据混合配比，增强高质量数据（数学、代码、逻辑内容）
4. **获取模型（Checkpoint）**
   - 将若干退火期间得到的模型权重 Checkpoint 求平均值，作为最终输出的预训练模型。

并行训练

- 1.6 万张 GPU H100 训练 405B 大模型，需要重点考虑并行策略和故障处理。
- Llama 3.1 训练采用 4D 并行（张量TP + 流水线PP + 上下文CP + 数据DP）；
- BF16 混精下 GPU 算力利用率（MFU）约为 38%～41%。

GPUs	TP	CP	PP	DP	Seq. Len.	Batch size/DP	Tokens/Batch	TFLOPs/GPU	BF16 MFU
8,192	8	1	16	64	8,192	32	16M	430	43%
16,384	8	1	16	128	8,192	16	16M	400	41%
16,384	8	16	16	4	131,072	16	16M	380	38%

Table 4 Scaling configurations and MFU for each stage of Llama 3 405B pre-training. See text and Figure 5 for descriptions of each type of parallelism.

并行训练

- **RoPE 扩展**：缩放 inv_freq 向量，可以一次性完成计算，不需要动态计算。从 8K tokens 到 128 K tokens 6 阶段逐步扩展，总共使用 800B tokens。

GPUs	TP	CP	PP	DP	Seq. Len.	Batch size/DP	Tokens/Batch	TFLOPs/GPU	BF16 MFU
8,192	8	1	16	64	8,192	32	16M	430	43%
16,384	8	1	16	128	8,192	16	16M	400	41%
16,384	8	16	16	4	131,072	16	16M	380	38%

Table 4 Scaling configurations and MFU for each stage of Llama 3 405B pre-training. See text and Figure 5 for descriptions of each type of parallelism.

并行训练：4D 并行


GPU 2
TP0 CP1 PP0 DP0
GPU 3
TP1 CP1 PP0 DP0
GPU 10
TP0 CP1 PP0 DP1
GPU 11
TP1 CP1 PP0 DP1
GPU 0
TP0 CP0 PP0 DP0
GPU 1
TP1 CP0 PP0 DP0
GPU 8
TP0 CP0 PP0 DP1
GPU 9
TP1 CP0 PP0 DP1
PP
GPU 6
TP0 CP1 PP1 DP0
GPU 7
TP1 CP1 PP1 DP0
GPU 14
TP0 CP1 PP1 DP1
GPU 15
TP1 CP1 PP1 DP1
CP
GPU 4
TP0 CP0 PP1 DP0
GPU 5
TP1 CP0 PP1 DP0
GPU 12
TP0 CP0 PP1 DP1
GPU 13
TP1 CP0 PP1 DP1
TP
DP

并行训练：PP 1F1B

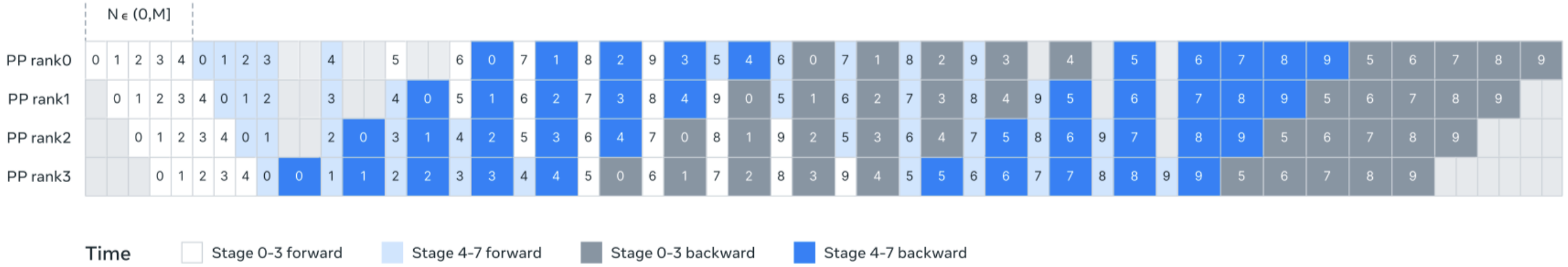


Figure 6 Illustration of pipeline parallelism in Llama 3. Pipeline parallelism partitions eight pipeline stages (0 to 7) across four pipeline ranks (PP ranks 0 to 3), where the GPUs with rank 0 run stages 0 and 4, the GPUs with P rank 1 run stages 1 and 5, *etc.* The colored blocks (0 to 9) represent a sequence of micro-batches, where M is the total number of micro-batches and N is the number of continuous micro-batches for the same stage's forward or backward. Our key insight is to make N tunable.

训练故障

Component	Category	Interruption Count	% of Interruptions
Faulty GPU	GPU	148	30.1%
GPU HBM3 Memory	GPU	72	17.2%
Software Bug	Dependency	54	12.9%
Network Switch/Cable	Network	35	8.4%
Host Maintenance	Unplanned Maintenance	32	7.6%
GPU SRAM Memory	GPU	19	4.5%
GPU System Processor	GPU	17	4.1%
NIC	Host	7	1.7%
NCCL Watchdog Timeouts	Unknown	7	1.7%
Silent Data Corruption	GPU	6	1.4%
GPU Thermal Interface + Sensor	GPU	6	1.4%
SSD	Host	3	0.7%
Power Supply	Host	3	0.7%
Server Chassis	Host	2	0.5%
IO Expansion Board	Host	2	0.5%
Dependency	Dependency	2	0.5%
CPU	Host	2	0.5%
System Memory	Host	2	0.5%

Table 5 Root-cause categorization of unexpected interruptions during a 54-day period of Llama 3 405B pre-training. About 78% of unexpected interruptions were attributed to confirmed or suspected hardware issues.

- Llama 3.1 训练集群故障处理十分优秀，超过 90% 有效训练时间；
- 依旧意味着，共 54 天（39 30 万 GPU 小时） 16K H10 0 集群预训练过程中，每天都至少有一次中断。

训练故障

Component	Category	Interruption Count	% of Interruptions
Faulty GPU	GPU	148	30.1%
GPU HBM3 Memory	GPU	72	17.2%
Software Bug	Dependency	54	12.9%
Network Switch/Cable	Network	35	8.4%
Host Maintenance	Unplanned Maintenance	32	7.6%
GPU SRAM Memory	GPU	19	4.5%
GPU System Processor	GPU	17	4.1%
NIC	Host	7	1.7%
NCCL Watchdog Timeouts	Unknown	7	1.7%
Silent Data Corruption	GPU	6	1.4%
GPU Thermal Interface + Sensor	GPU	6	1.4%
SSD	Host	3	0.7%
Power Supply	Host	3	0.7%
Server Chassis	Host	2	0.5%
IO Expansion Board	Host	2	0.5%
Dependency	Dependency	2	0.5%
CPU	Host	2	0.5%
System Memory	Host	2	0.5%

Table 5 Root-cause categorization of unexpected interruptions during a 54-day period of Llama 3 405B pre-training. About 78% of unexpected interruptions were attributed to confirmed or suspected hardware issues.

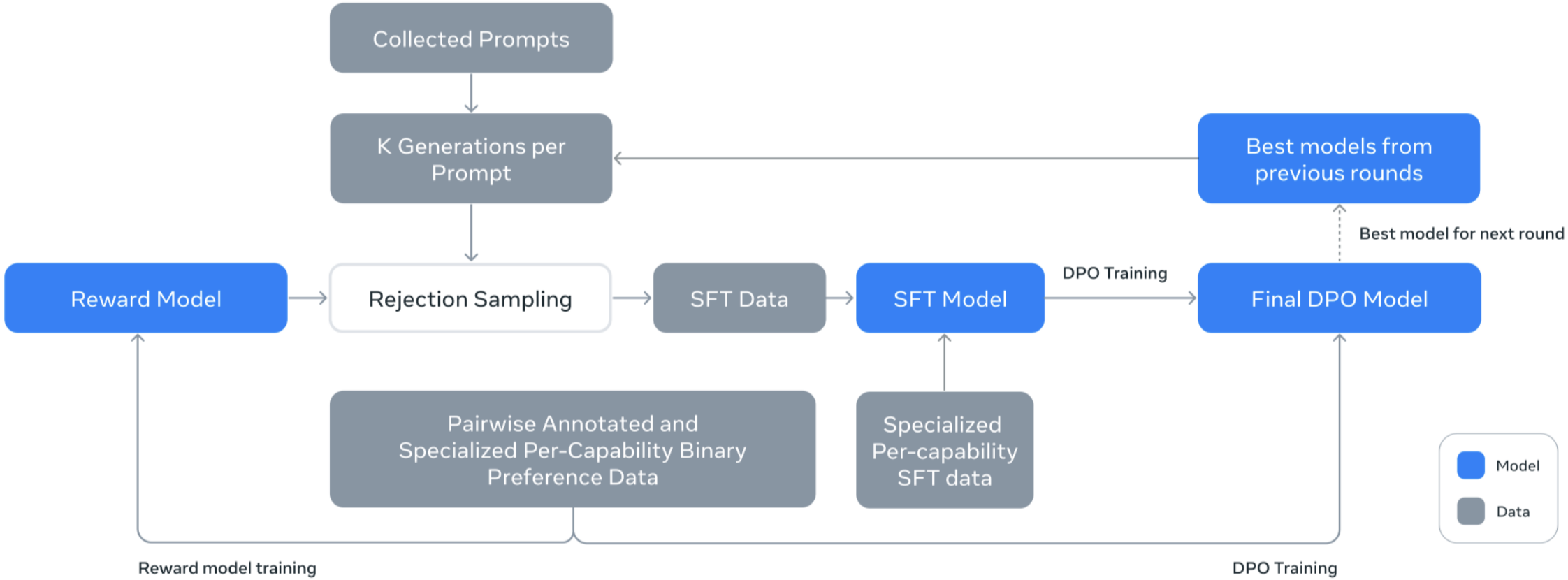
- 训练过程一共 419 次意外中断故障。
- 其中确认/怀疑与硬件相关问题占比达到了78%。
- 由于集群自动化运维比较完善，尽管故障次数多，但大部分都可以被自动处理。整个过程中，只有3次故障需要手动干预。

看后训练

Post-Training

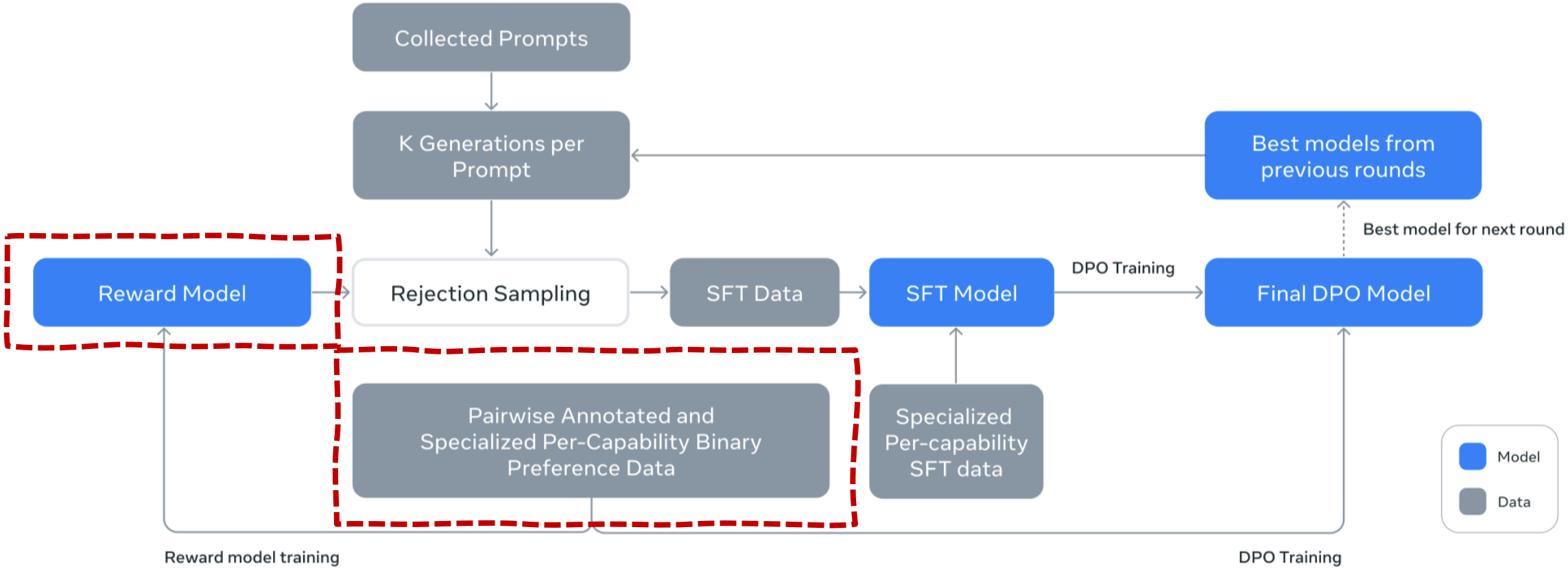
后训练过程

- 通过多轮对齐来完善 Chat 模型，基于监督微调（SFT）、拒绝采样（RS）和通过 DOP 直接优化偏好。



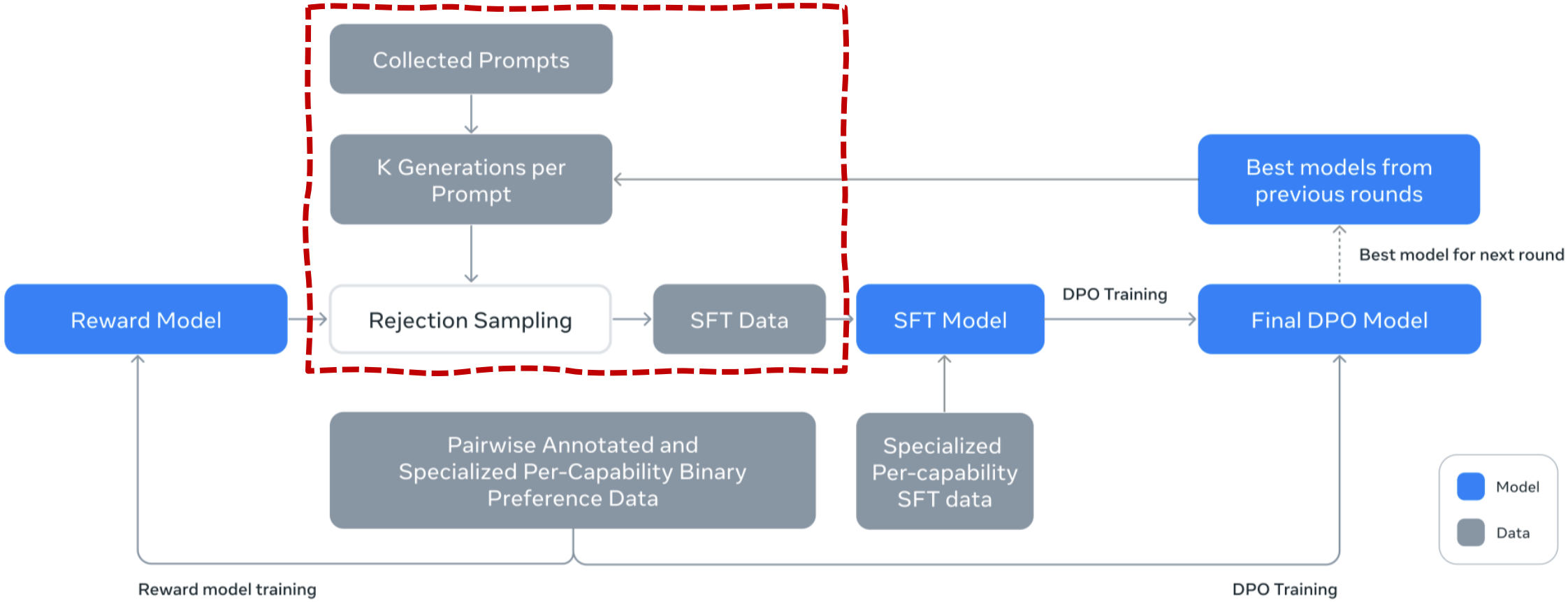
后训练过程

1. **训练 RM 模型：**人工标注数据训练 RM 模型，用来评价 <prompt, answer> 数据对质量；



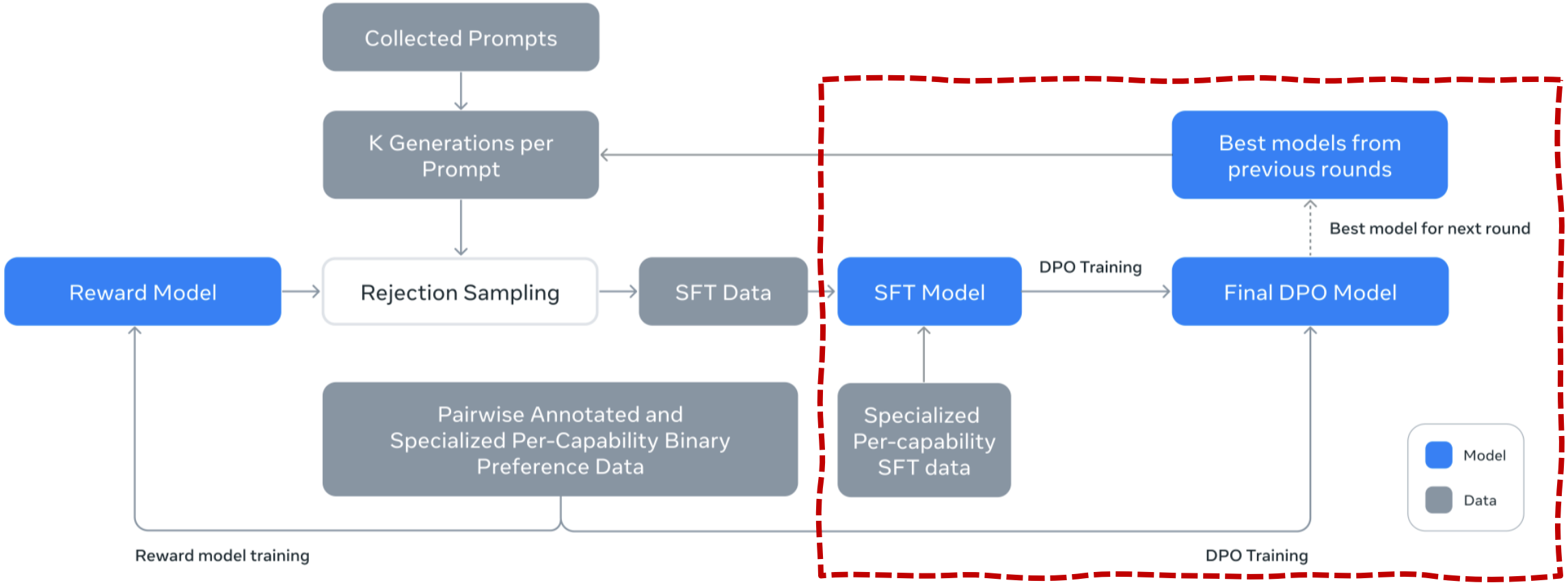
后训练过程

2. **拒绝采样（Rejection Sampling）**：对输入 Prompt，模型生成若干个回答，RM 给予质量打分，选择得分最高保留作为 SFT 数据，其它抛掉；



后训练过程

3. **DPO 训练：**人工标注数据给 DPO 模型调整 LLM 参数。DPO本质为二分类，从人工标注 <Prompt, Good Answer, Bad Answer> 三元数据里学习，调整模型参数并鼓励输出 Good Answer。



后训练过程

- 上述过程会反复迭代，流程相同，区别 Rejection Sampling 阶段用来对给定 Prompt 产生回答 LLM 模型，从上一轮流程最后产生若干不同 DPO 模型，选择最好的那个在下一轮拒绝采样阶段给Prompt生成答案。随迭代增加 DPO 模型越来越好，拒绝采样能选出的答案质量越高，SFT 模型就越好，形成正反馈循环。

- **RM 在后训练作用区别：**RLHF 是把 RM 打分用在 PPO 强化学习阶段；Llama3.1 用 RM 筛选高质量 SFT 数据。
- **SFT 数据合成：**因为拒绝采样过程的回答由 LLM 产生，因此采用合成数据来训练 SFT 模型。

提高特定下游任务性能

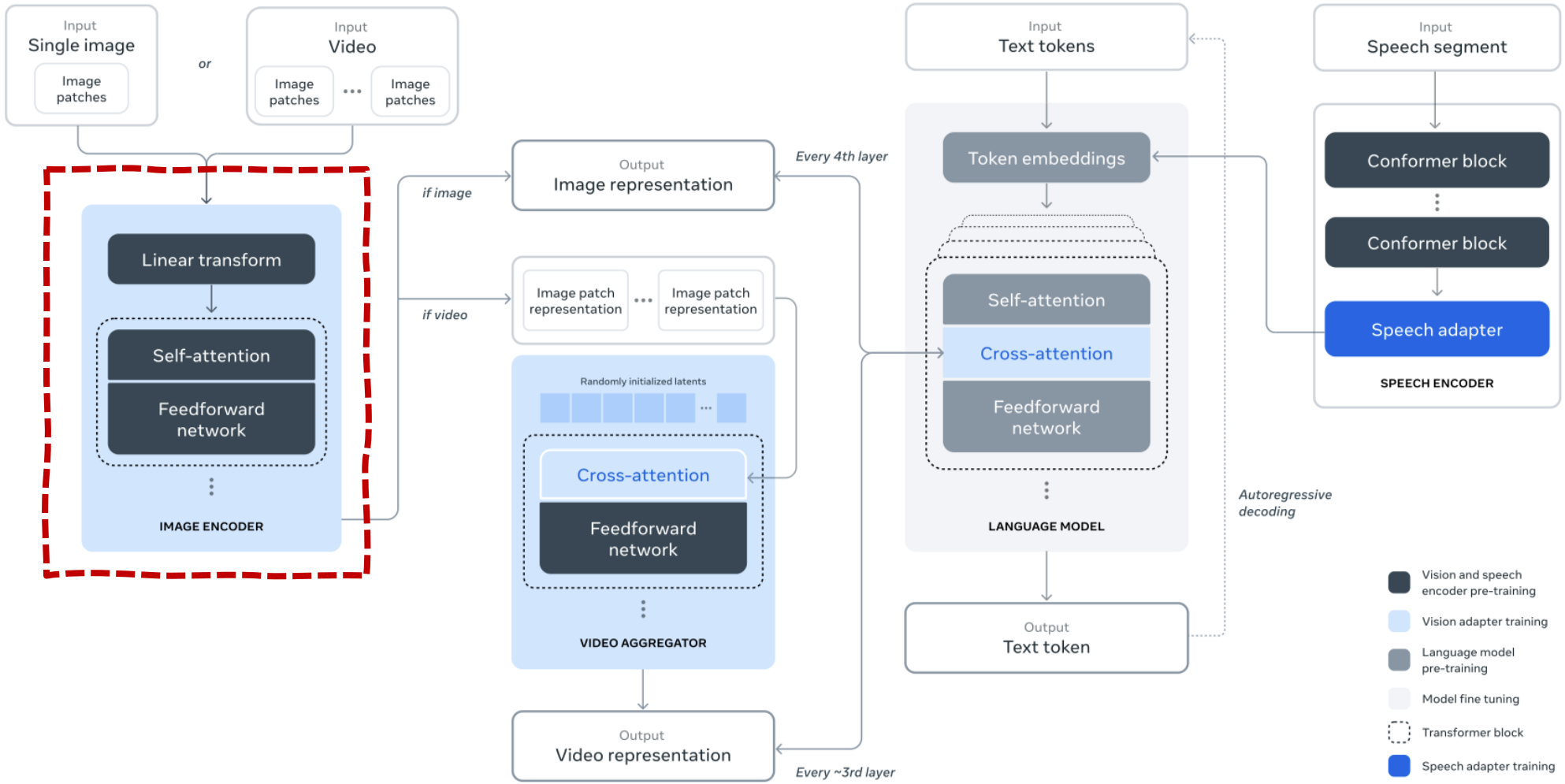
1. **DPO 原因：**采用 DPO 的 RLHF 而非更复杂 RL 强化学习算法， 因为后者稳定性不确定且制约AI 集群规模扩展 Scaling Law；
2. **提高模型编码能力：**采用训练代码专家、生成 SFT 合成数据、通过系统提示引导改进格式，以及创建质量过滤器（从训练数据中删除不良样本）等方法。
3. **Float8 量化推理：**将权重/输入量化为 fp8，然后乘以缩放因子 scale，fp8 x fp8 输出 bf16。使得推理速度更快，显存占用更少（推理在 H100 云资源集群）。

多模态加持

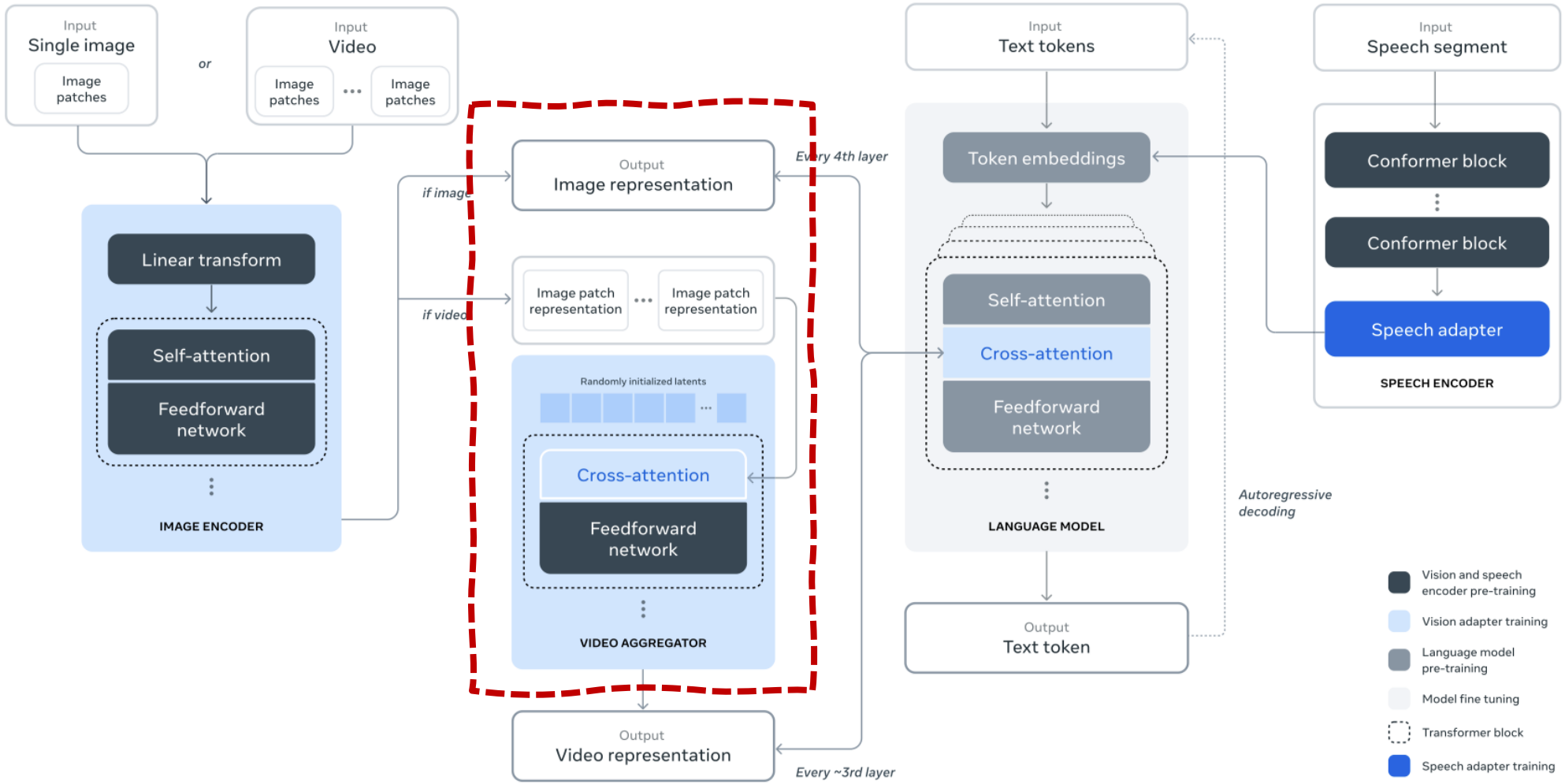
多模态适配

- 使用 3 个额外阶段为 Llama 3.1 添加图像、视频、语音多模态能力：
  1. **多模态 Encoder 预训练**：图像和语音 Encoder 分开训练，CV 预训练数据是<图像-文本>对，Audio 采用自监督，通过离散化 token 重建语音 Mask 部分；
  2. **视觉 Adapter**：由一系列 Cross Attention 组成，将 CV Encoder 后的特征加入到 LLM 中；
  3. **语音 Adapter**：通过 Conformer 处理语音数据后，与输入 Token Concat 到一起作为 LLM 输入；

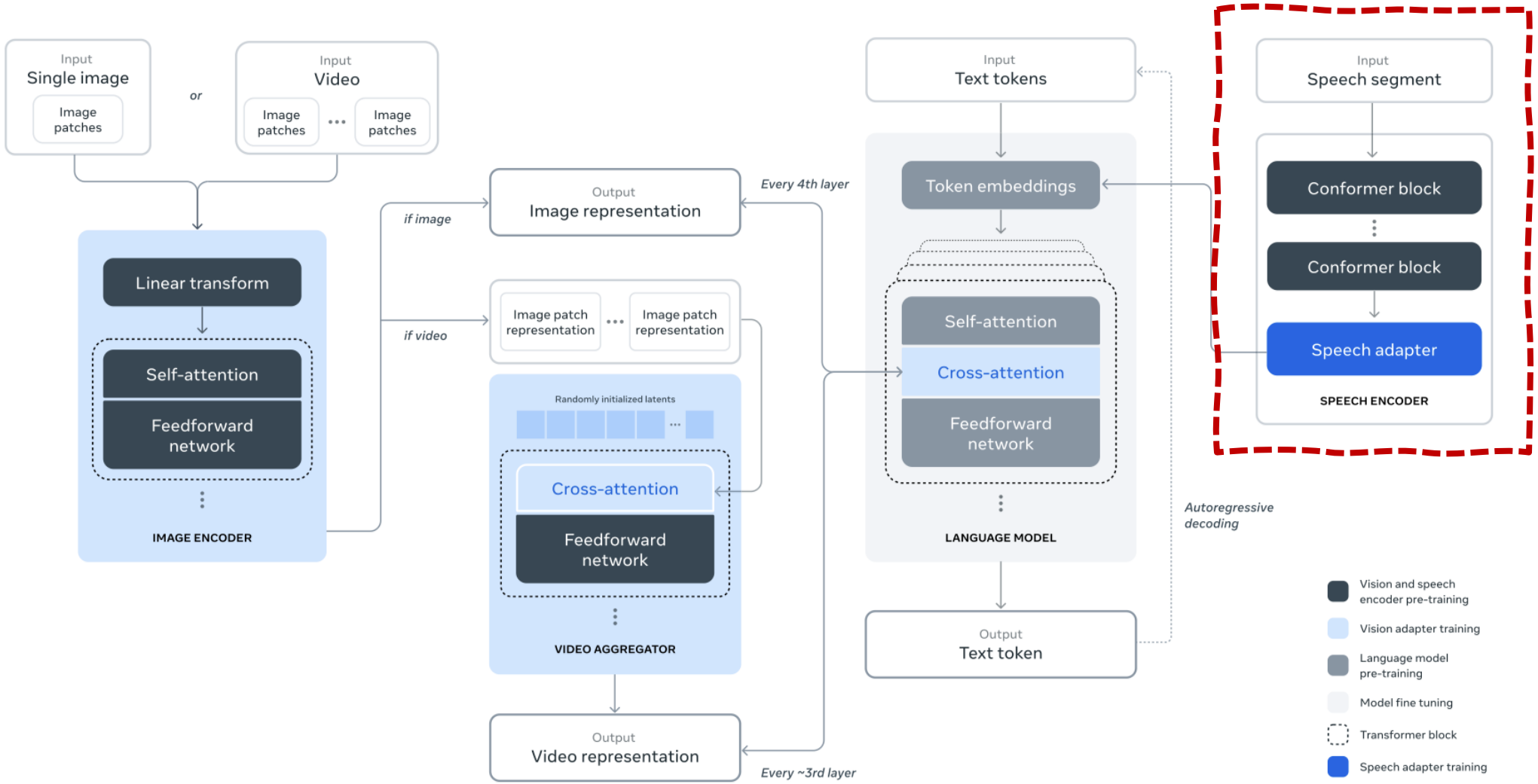
多模态适配：视觉


Input
Single image
Image patches
or
Input
Video
Image patches
Image patches
Linear transform
Self-attention
Feedforward network
IMAGE ENCODER
if image
if video
Output
Image representation
Image patch representation
Image patch representation
Randomly initialized latents
Cross-attention
Feedforward network
VIDEO AGGREGATOR
Output
Video representation
Every 4th layer
Every ~3rd layer
Input
Text tokens
Token embeddings
Self-attention
Cross-attention
Feedforward network
LANGUAGE MODEL
Output
Text token
Autoregressive decoding
Input
Speech segment
Conformer block
Conformer block
Speech adapter
SPEECH ENCODER
Vision and speech encoder pre-training
Vision adapter training
Language model pre-training
Model fine tuning
Transformer block
Speech adapter training

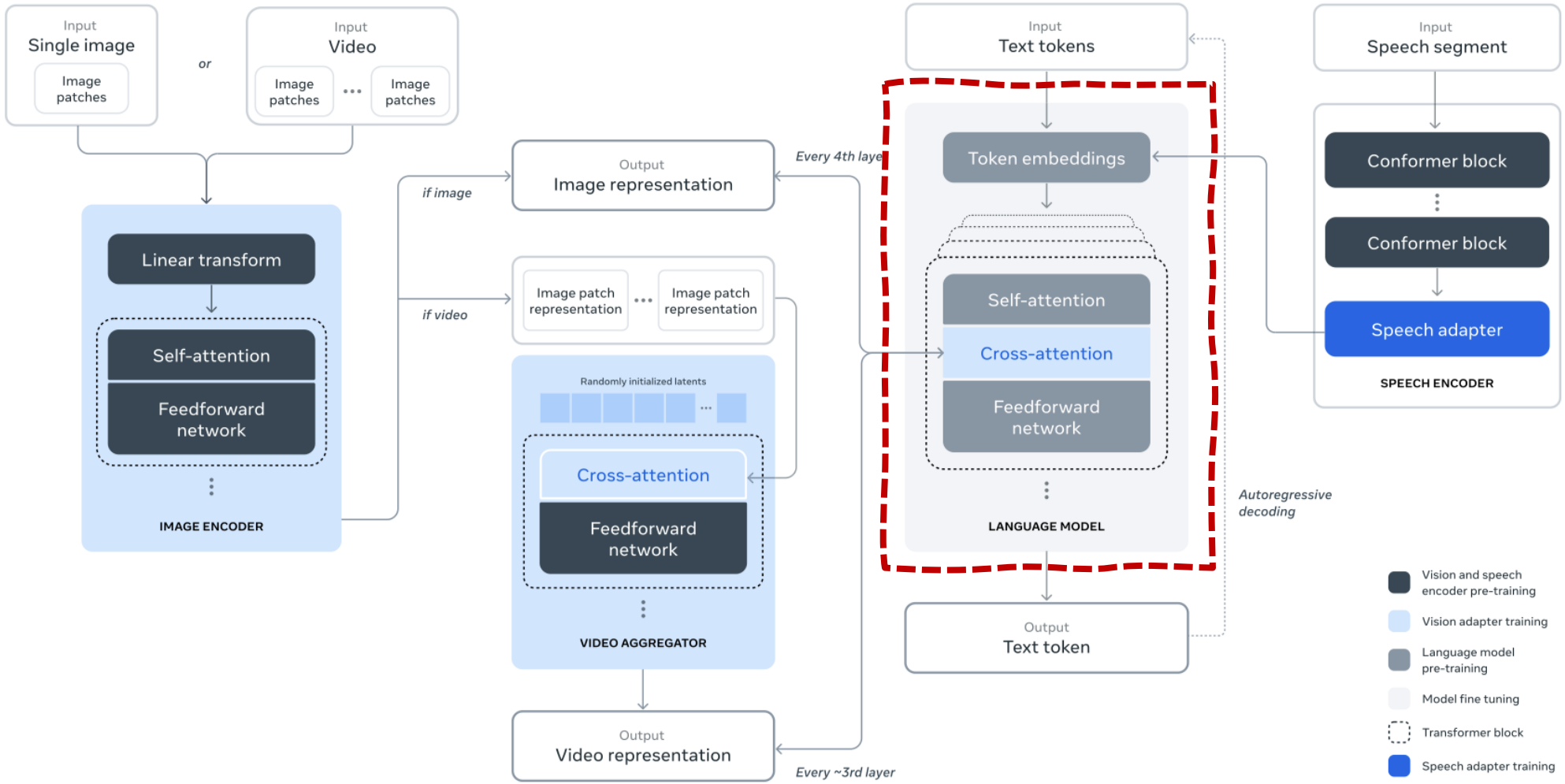
多模态适配：视觉


Input
Single image
Image patches
or
Input
Video
Image patches
Image patches
Linear transform
Self-attention
Feedforward network
IMAGE ENCODER
if image
if video
Output
Image representation
Image patch representation
Image patch representation
Randomly initialized latents
Cross-attention
Feedforward network
VIDEO AGGREGATOR
Output
Video representation
Every 4th layer
Every ~3rd layer
Input
Text tokens
Token embeddings
Self-attention
Cross-attention
Feedforward network
LANGUAGE MODEL
Output
Text token
Autoregressive decoding
Input
Speech segment
Conformer block
Conformer block
Speech adapter
SPEECH ENCODER
Vision and speech encoder pre-training
Vision adapter training
Language model pre-training
Model fine tuning
Transformer block
Speech adapter training

多模态适配：语音


Input
Single image
Image patches
or
Input
Video
Image patches
Image patches
Linear transform
Self-attention
Feedforward network
IMAGE ENCODER
if image
if video
Output
Image representation
Image patch representation
Image patch representation
Randomly initialized latents
Cross-attention
Feedforward network
VIDEO AGGREGATOR
Output
Video representation
Every 4th layer
Every ~3rd layer
Input
Text tokens
Token embeddings
Self-attention
Cross-attention
Feedforward network
LANGUAGE MODEL
Output
Text token
Autoregressive decoding
Input
Speech segment
Conformer block
Conformer block
Speech adapter
SPEECH ENCODER
Vision and speech encoder pre-training
Vision adapter training
Language model pre-training
Model fine tuning
Transformer block
Speech adapter training

多模态适配：语音


Input
Single image
Image patches
or
Input
Video
Image patches
Image patches
Linear transform
Self-attention
Feedforward network
IMAGE ENCODER
if image
if video
Output
Image representation
Image patch representation
Image patch representation
Randomly initialized latents
Cross-attention
Feedforward network
VIDEO AGGREGATOR
Output
Video representation
Every 4th layer
Every ~3rd layer
Input
Text tokens
Token embeddings
Self-attention
Cross-attention
Feedforward network
LANGUAGE MODEL
Output
Text token
Autoregressive decoding
Input
Speech segment
Conformer block
Conformer block
Speech adapter
SPEECH ENCODER
Vision and speech encoder pre-training
Vision adapter training
Language model pre-training
Model fine tuning
Transformer block
Speech adapter training

03. 对业界思考

大模型结构演进

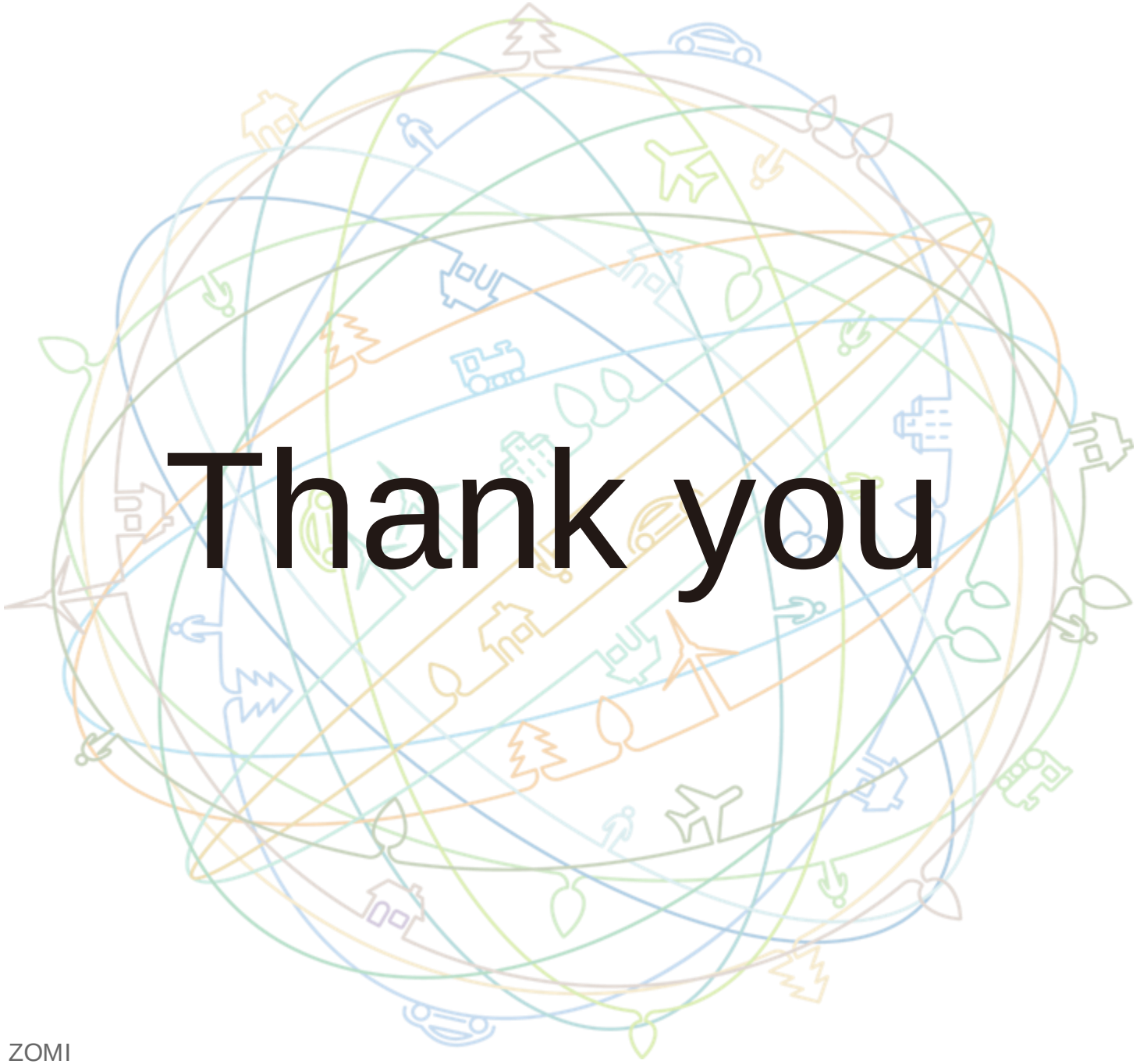
- **大模型选 MOE 稀疏还是 Transform 稠密?**
  - **MOE 优势：**减少训练和推理成本；**MOE 代价：**训练不够稳定、推理需要大内存存储参数量。
  - 用户量大请求数多，推理成本占比高，使用 MOE 推理更友好，主要出于成本、性能而非效果角度考虑。

数据枯竭问题

- **合成数据在 NVIDIA Nemotron 提供 DEMO后，真正真能用?**
  - 合成数据已经进入实用化阶段，Post-Training 阶段（特别 SFT）已经产品化，也成为主导方向。
  - LLama 3.1 和 Gemma2 在 SFT 阶段数据很大比例由大模型合成，证明合成数据质量不比人工标注差。

大模型的能力上线

- **Llama3 成为开源最强大模型，大模型的能力上线在哪里？还能够继续提升吗？**
  - 继续扩大模型和数据规模（Scaling Law）；
  - 强调数据质量作用，增加数学、逻辑、代码这种能够提升大模型能力的数据配比比例；



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

Course chenzomi12.github.io

GitHub github.com/chenzomi12/DeepLearningSystem