
大模型-AI集群 (存)
存储&大模型中挑战
ZOMI

大模型业务全流程

数据 & 模型算法

2. 数据处理

- 开源数据
- 数据预处理
- 向量数据库

3. 模型算法

- LLM模型架构
- 多模态一切皆Tokens

模型训练 & 微调

4. 模型训练

- 混合精度
- 梯度检查
- 梯度累积
- …

5. 分布式并行

训练集群稳定性

6. 模型微调

- 全参微调
- 低参微调
- 指令微调

模型验证 & 推理部署

7. 模型验证

- 下游任务
- 测评标准
- bench mark

8. 推理与智能体

- 量化压缩
- 推理加速
- 9. Agent 智能体

集群算力准备

1. AI 集群建设

- 计算、存储、网络
- AI 集群机房建设
- AI 集群上线与运维

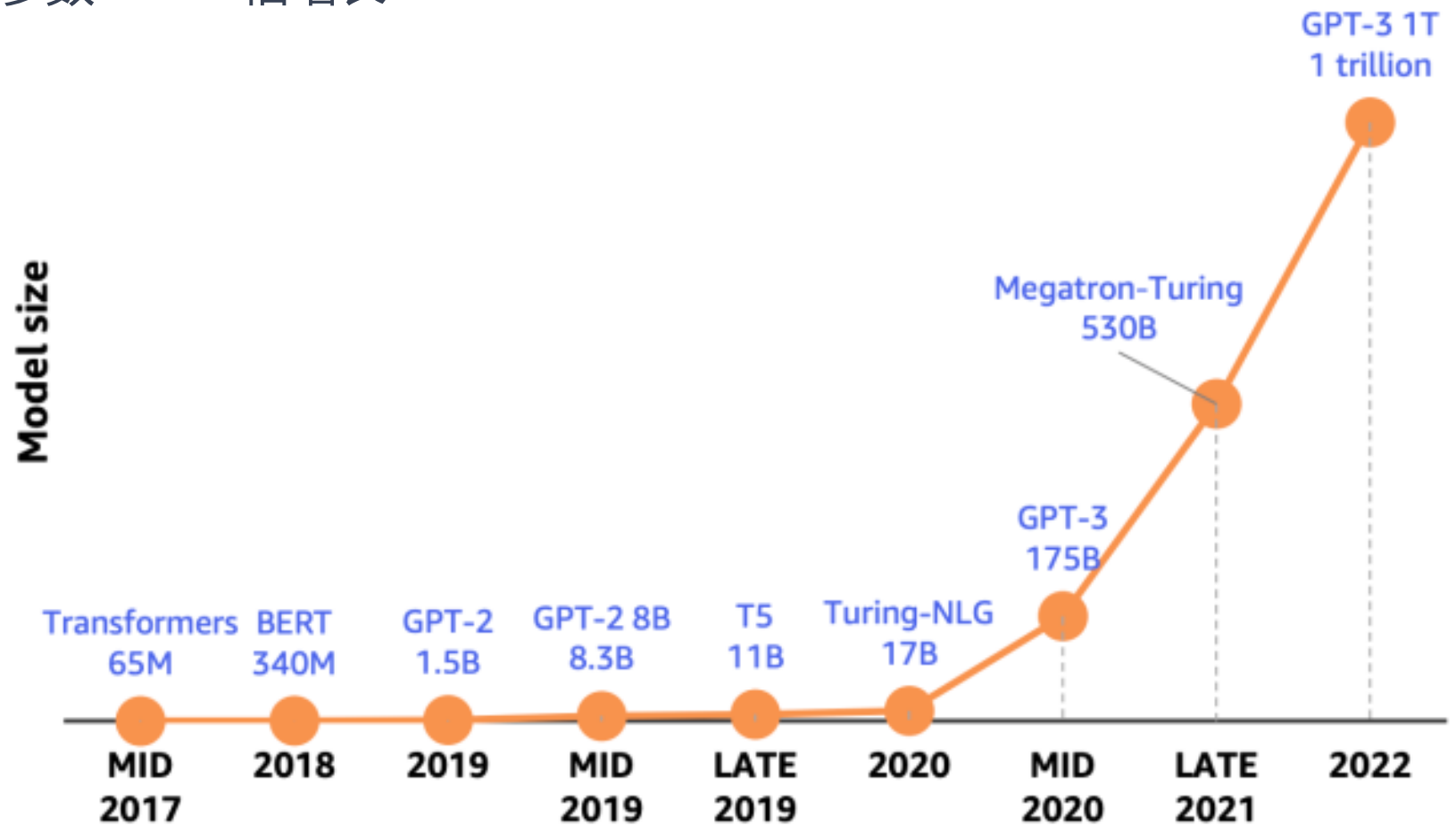
关于本内容

- **内容背景**
  - AI 集群 + 大模型
- **具体内容**
  - **数据存储现状和场景：**存储软件类型、存储硬件类型的发展
  - **大模型对存储的挑战：**存储性能指标、存储遇到大模型挑战与新机会点
  - **大模型训练CKPT优化：**大模型训练过程、CKPT过程分解、CKPT优化
  - **大模型时代对存储的思考：**什么样的存储架构才是AI大模型时代的选择？

1. 大模型の基本背景

大模型参数规模发展变化

- 近 3 年模型参数 1000~ 倍增长



模型参数与数据集规模增长

- 千亿规模参数模型，数据集在TB量级；万亿规模参数模型，数据集在10TB量级。

模型	发布时间	参数		预训练数据集	模型类型
GPT-1	2018.06	1.17 亿	0.1 B	5GB	NLP
GPT-2	2019.02	15 亿	1.5 B	40GB	NLP
GPT-3	2020.05	1750 亿	175 B	3TB	NLP
GPT-4	2022.05	/	/	45TB	NLP
Switch Transformer	2021.03	1.6 万亿	1.6 T	750GB	CV
Megatron	2021.10	5300 亿	530 B	/	NLP
LLAMA	2022.10	650 亿	65 B	2TB	NLP
LLAMA2	2023.07	700 亿	70 B	4.5TB	NLP

Date	Model	Model size	Dataset size (Tokens)	HumanEval (Pass@1)	MBPP (Pass@1)
2021 Jul	Codex-300M [CTJ+21]	300M	100B	13.2%	-
2021 Jul	Codex-12B [CTJ+21]	12B	100B	28.8%	-
2022 Mar	CodeGen-Mono-350M [NPH+23]	350M	577B	12.8%	-
2022 Mar	CodeGen-Mono-16.1B [NPH+23]	16.1B	577B	29.3%	35.3%
2022 Apr	PaLM-Coder [CND+22]	540B	780B	35.9%	47.0%
2022 Sep	CodeGeeX [ZXZ+23]	13B	850B	22.9%	24.4%
2022 Nov	GPT-3.5 [Ope23]	175B	N.A.	47%	-
2022 Dec	SantaCoder [ALK+23]	1.1B	236B	14.0%	35.0%
2023 Mar	GPT-4 [Ope23]	N.A.	N.A.	67%	-
2023 Apr	Replit [Rep23]	2.7B	525B	21.9%	-
2023 Apr	Replit-Finetuned [Rep23]	2.7B	525B	30.5%	-
2023 May	CodeGen2-1B [NHX+23]	1B	N.A.	10.3%	-
2023 May	CodeGen2-7B [NHX+23]	7B	N.A.	19.1%	-
2023 May	StarCoder [LAZ+23]	15.5B	1T	33.6%	52.7%
2023 May	StarCoder-Prompted [LAZ+23]	15.5B	1T	40.8%	49.5%
2023 May	PaLM 2-S [ADF+23]	N.A.	N.A.	37.6%	50.0%
2023 May	CodeT5+ [WLG+23]	2B	52B	24.2%	-
2023 May	CodeT5+ [WLG+23]	16B	52B	30.9%	-
2023 May	InstructCodeT5+ [WLG+23]	16B	52B	35.0%	-
2023 Jun	WizardCoder [LXZ+23]	16B	1T	57.3%	51.8%
2023 Jun	**phi-1**	1.3B	7B	50.6%	55.5%

Table 1: We use self-reported scores whenever available. Despite being trained at vastly smaller scale, **phi-1** outperforms competing models on HumanEval and MBPP, except for GPT-4 (also WizardCoder obtains better HumanEval but worse MBPP).

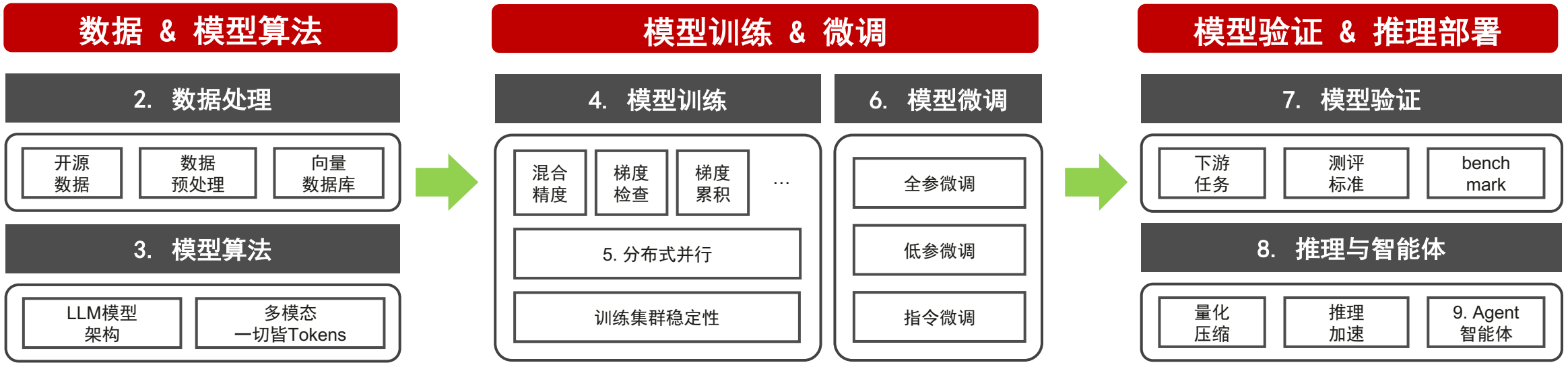
大模型对存储的改变

1. 模型参数量增大，训练时候新需要的内存 and/or 显存增大
2. 随着模型参数量增大，需要配套的训练数据增大，对存储的需求增加

2. 大模型全流程

遇到存储

大模型业务全流程


数据 & 模型算法
2. 数据处理
开源数据
数据预处理
向量数据库
3. 模型算法
LLM模型架构
多模态一切皆Tokens
模型训练 & 微调
4. 模型训练
混合精度
梯度检查
梯度累积
…
5. 分布式并行
训练集群稳定性
6. 模型微调
全参微调
低参微调
指令微调
模型验证 & 推理部署
7. 模型验证
下游任务
测评标准
benchmark
8. 推理与智能体
量化压缩
推理加速
9. Agent智能体

大模型业务全流程

- 与存储有强关联的流程主要分为3部分：
  1. 模型开发和数据准备
  2. 模型训练和微调
  3. 推理部署与智能体

Step1：模型开发和数据准备

- **特点**：对海量数据的存储和处理，包括对数据采集导入、清洗、转换、标注、共享和长期归档

- **对存储要求**：跟大数据相类似，需要多协议支持、多级存储空间、高吞吐、大容量。

Step2：模型训练和微调

- **特点**：模型训练和微调包括开发过程和实际的训练
  - 开发过程讲究效率，包括 MLOPS 的实验管理、大模型代码开发、实验效果评估等。
  - 训练过程中一是对训练数据的读取，二是为了容错对 checkpoint 的保存和加载。

- **对存储要求**：
  - 非结构化的数据集尽量读取更快，减少计算对 I/O 的等待
  - Checkpoint 要求高吞吐、减少训练中断的时间。

Step3：推理部署与智能体

- **特点**：需要把训练完的模型 checkpoint 进行聚合，不断执行和部署在具体设备或者系统上

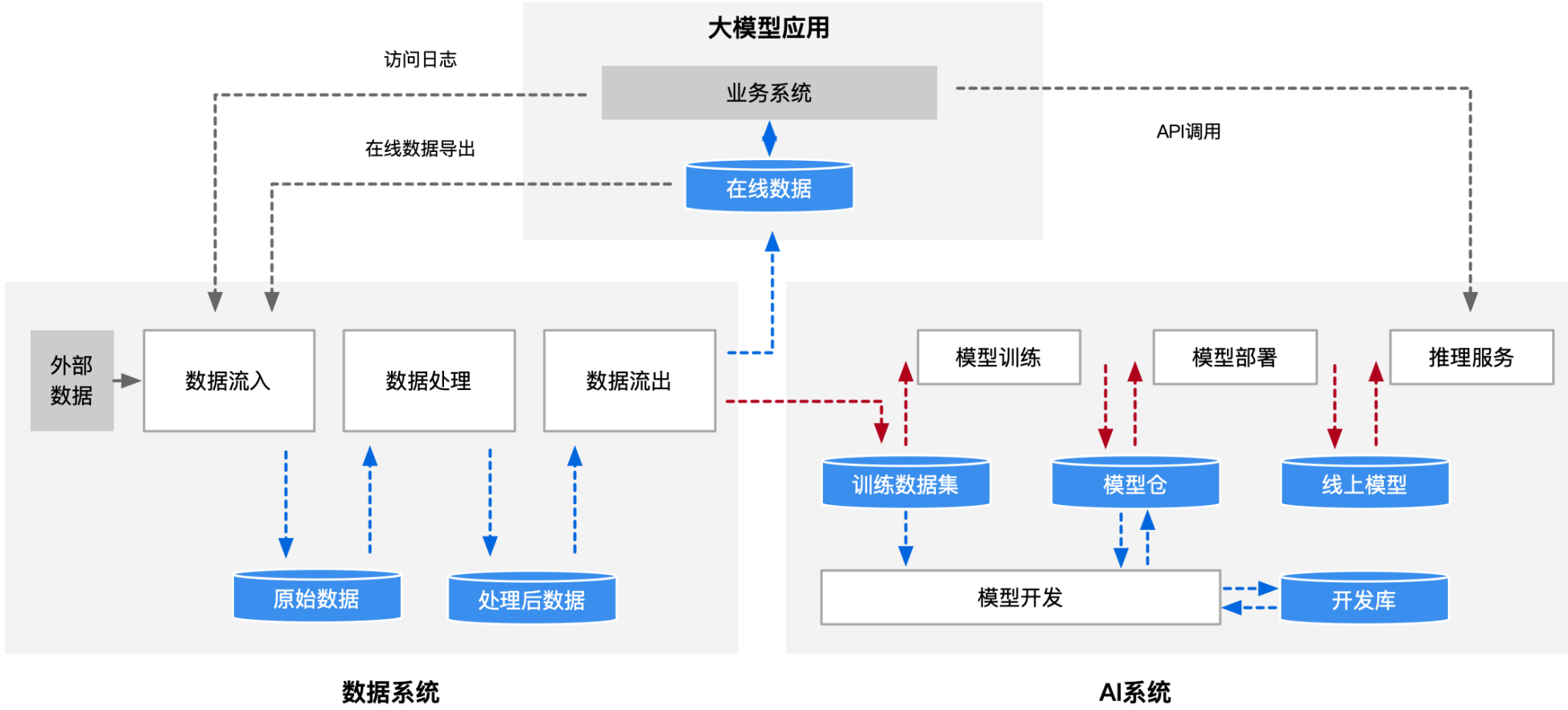
- **对存储要求**：高频、反复执行推理，既要求高并发、高吞吐，又要求整个流程尽量简单高效。

大模型业务全流程对存储的需求

关键词	**海量数据、持续更新**	**快速开发、断点保存、多级聚合**	**高频执行、快速部署**
业务场景	**数据存储和处理**	**模型开发的训练和微调**	**模型推理和智能体**
	数据采集和导入	MLOPS 实验管理、交互式开发	Checkpoint 模型部署
	数据清洗、转换、标注	效果评估、数据集读取和加载	实际数据调度
	训练数据共享和导出	Checkpoint 保存和加载	
	训练数据归档和存储	Checkpoint 聚合与分发	
存储需求	跨数据协议、对象支持 高吞吐、大容量	训练数据集高读写性能 训练数据集预读取、客户端缓存 Checkpoint稳定高效写、加载耗时少	高并发、高吞吐、高效率

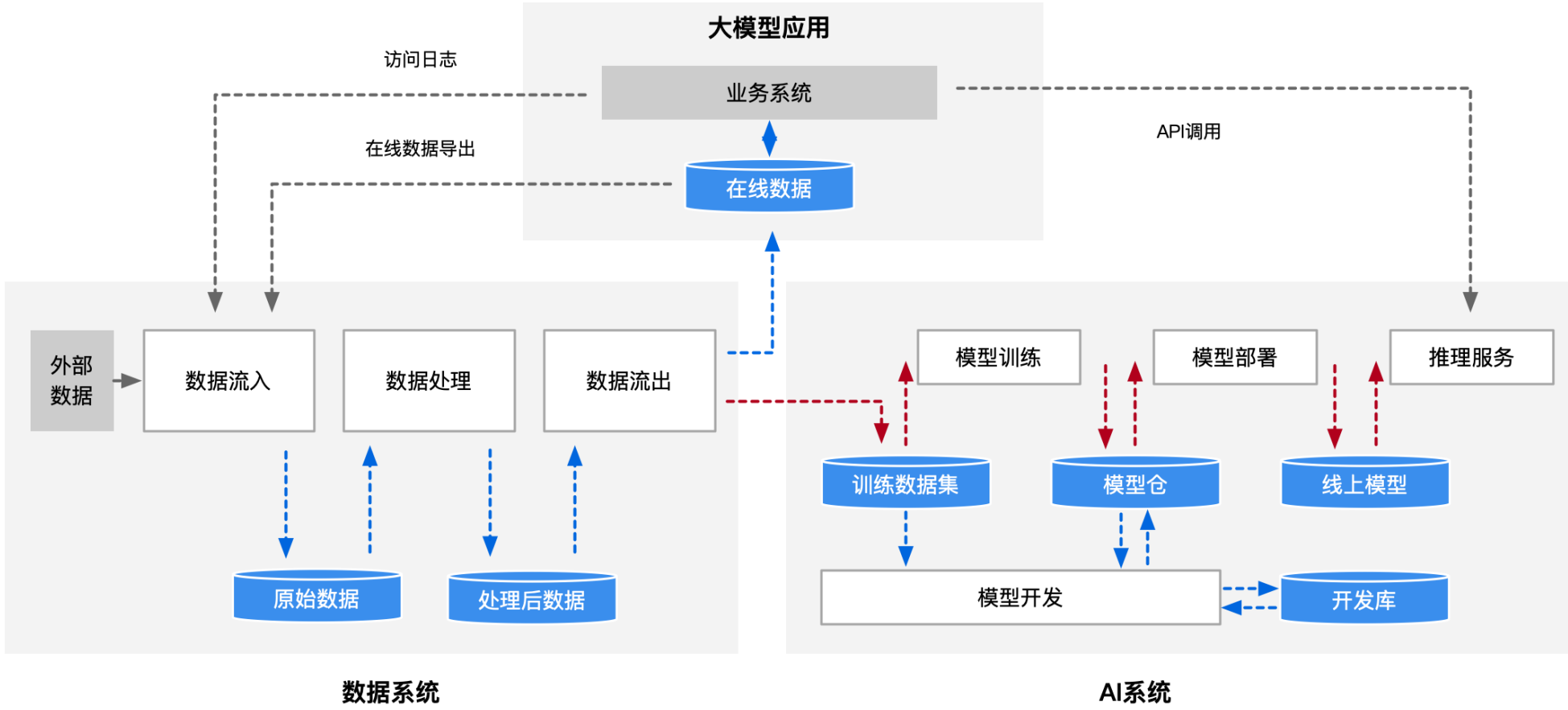
大模型训练在 AI 系统中划分

- 将AI集群划分为：大模型应用、数据系统、AI 系统三部分。



大模型系统流程对存储的需求

- 各环节间的衔接，本质上是对数据在不同子系统中广泛、高效流动的需求。



3. 大模型时代的挑战

挑战 1：海量小文件的高并发、低延迟读写

- **特点**：海量多模态、异构小文件的高并发、低延迟读写
  - eg.，LLMs 以文本语料为主，同时存在pdf、doc、excel 等各种格式，甚至存在医疗、气象等专用格式。
  - 大部分为半结构化的数据，其大小从 10KB 到 100 KB 不等小文本文件，存在数亿级的文件数量。
  - 大模型训练过程中，频繁从数据集取 Token，每 Token 4 Byte，实时高并发小IO性能需要极低延迟。
  - 存储性能瓶颈在于对小文件 OPS，而非带宽

挑战 II：异构多模态数据，需分布式并行读写

- **特点**：多模态数据快速训练加载，高度依赖分布式并行能力
  - 异构多模态文件数据间存在关联、嵌套关系，如图-文对应、文-视频对应等
  - 异构数据需被高速加载到 LLMs 进行分布式训练，高并发、低延迟成为巨大挑战

- 对于 LLM 大语言模型来说，暂时异构并不是主要矛盾

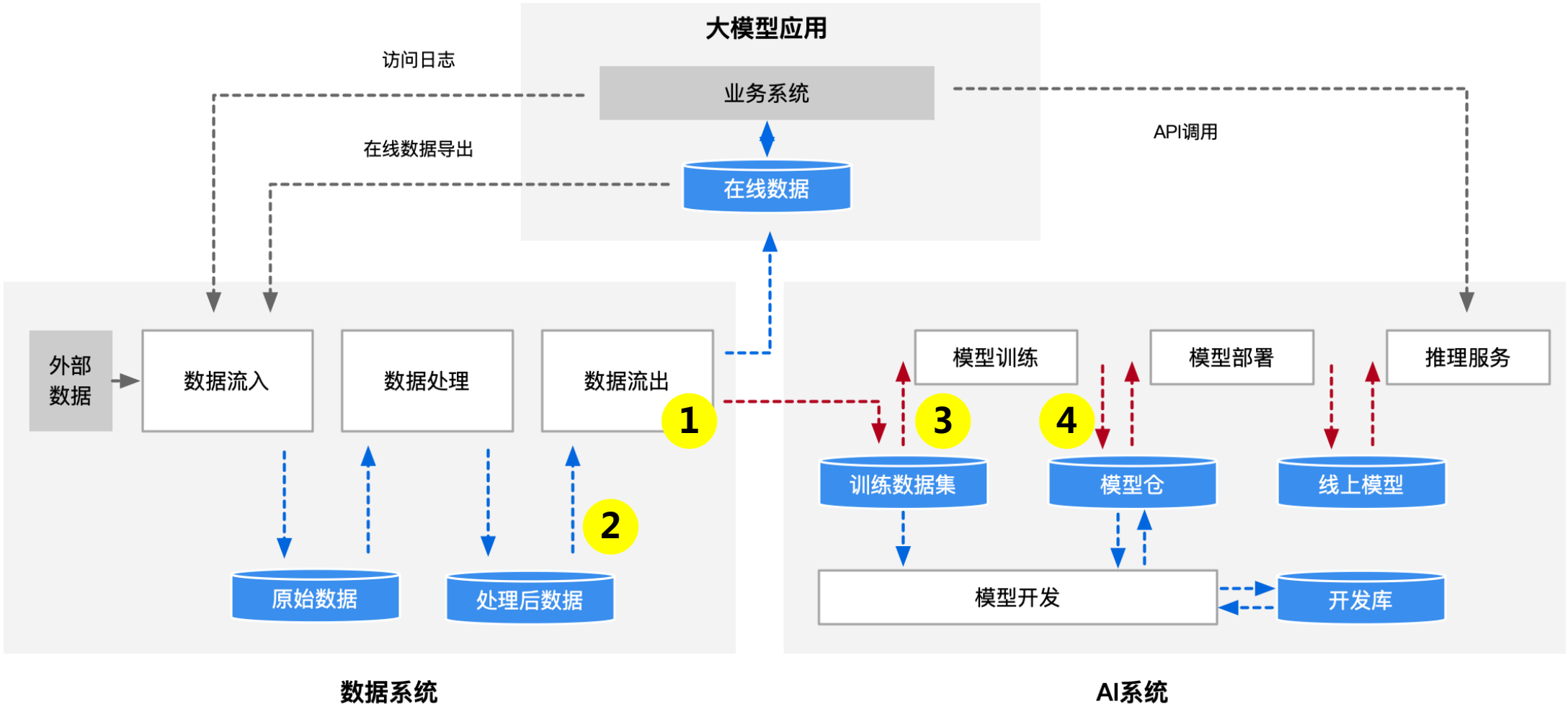
挑战 III：超大规模非结构化数据，需海量存储空间

- **特点**：AI 大模型需要高质量、大规模、多样性的数据集
  - 随着数据和模型规模的增长，数据量会呈现指数级增长，独立存储无法满足应用需求
  - 海量存储空间和可以横向扩展的存储系统尤为重要，分布式存储解决方案势在必行

挑战 IIII：模型训练存储稳定性

- **特点**：模型训练过程中，需要提供稳定的训练断点保存和回复的存储能力
  - 为了减少 TTA，高吞吐和低延时为 NPU 计算提供数据支撑，减少 NPU 计算等待
  - 模型训练 Checkpoint 必不可少，优化 Checkpoint 并缩短其耗时，减少训练中断时间
  - 存储模型 Checkpoint 时，为 Checkpoint 数据可快速写入，需要高带宽
  - 提升存储稳定性，避免每次故障元数据同步，需要停止服务重新拉起

总体存储的挑战


大模型应用
访问日志
业务系统
在线数据导出
在线数据
API调用
外部数据
数据流入
数据处理
数据流出
1
2
原始数据
处理后数据
数据系统
模型训练
模型部署
推理服务
3
4
训练数据集
模型仓
线上模型
模型开发
开发库
AI系统

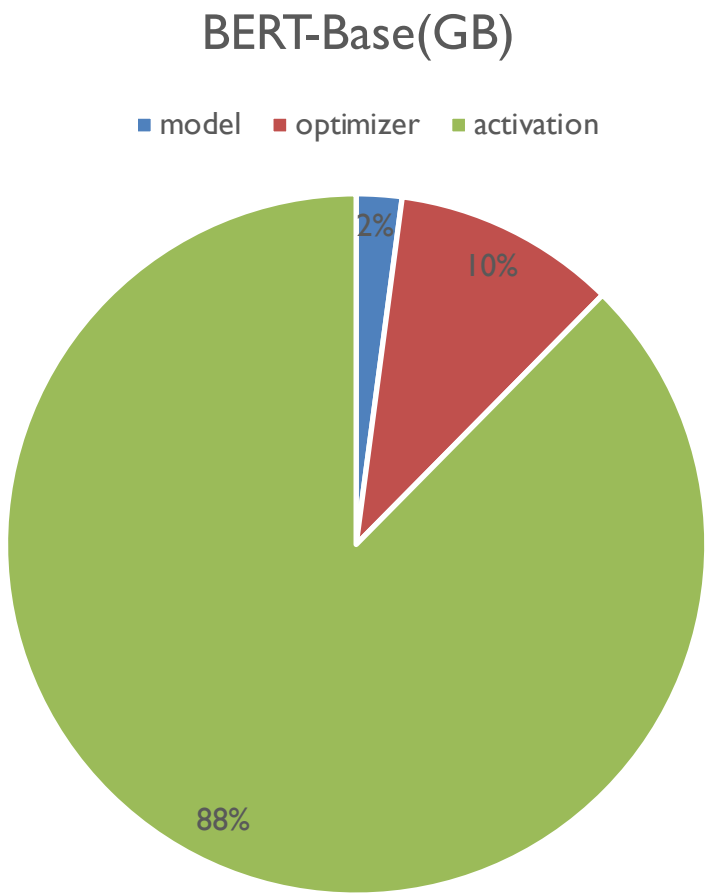
4. 存储的性能指标

大模型训练内存占用

- 内存占用跟1）模型、2）优化器、3）激活值有强关联
- 模型 Model
  - Parameters 权重参数 (half) 2 bytes，
  - Gradient 梯度参数 (half) 2 bytes，
- 优化器状态 Optimizers status
  - Master Weight (FP32) 4 bytes
  - Adam m (FP32) 4 bytes
  - Adam v (FP32) 4 bytes
- 总的内存开销，Total amount of memory needed
  - 公式

大模型训练内存占用

- 内存占用跟1）模型、2）优化器、3）激活值有强关联
- **模型 Model**
  - Parameters 权重参数 (half) 2 bytes，
  - Gradient 梯度参数 (half) 2 bytes，
- **优化器状态 Optimizers status**
  - Master Weight (FP32) 4 bytes
  - Adam m (FP32) 4 bytes
  - Adam v (FP32) 4 bytes
- **激活函数：forward中保存，用于反向传播**



大模型训练内存占用

- b 代表 batch size， s 代表 sequence length，L 代表层数，h 代表隐变量大小，a 代表 attention head 头数。

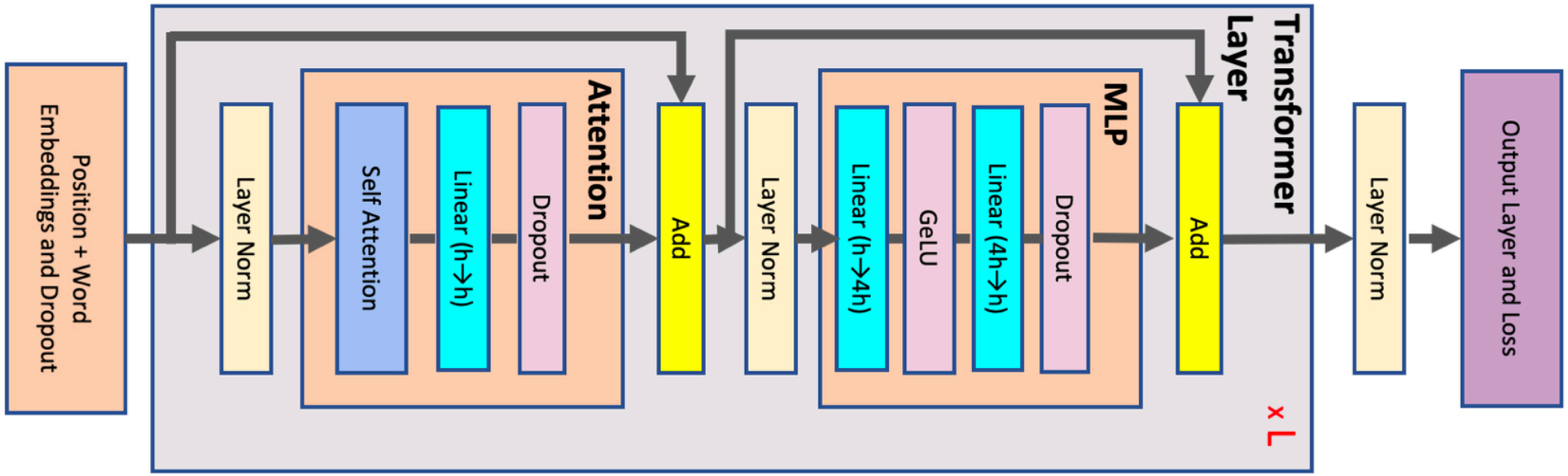


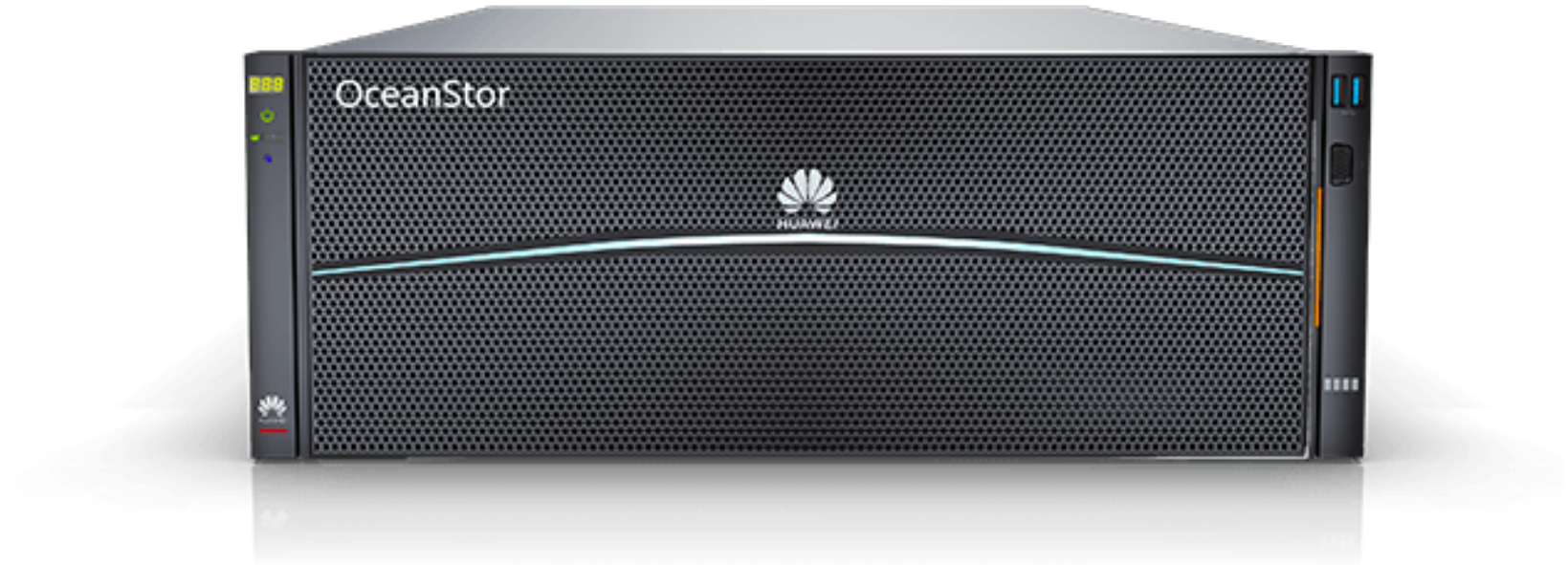
Figure 2: Transformer Architecture. Each gray block represents a single transformer layer that is replicated L times.

激活函数：forward中保存，用于反向传播

Configuration	Activations Memory Pre Transformer Layers
no Parallelism	$sbh(32 + 5\,as/h)$
TP (Baseline)	$sbh(10 + 24/t + 5as/ht)$
TP + SP	$sbh(34/t + 5as/ht)$
TP + SAR	$sbh(10 + 24/t)$
TP + SP +SAR	$sbh(34/t)$
FAR (full activation recompute)	$sbh(2)$

性能指标

1. 衡量块存储产品的性能指标主要包括 IOPS、吞吐量和访问时延。
2. 对于分布式存储系统的性能评判指标主要包括：延时、 iops、 带宽等。



性能指标 1：IOPS

- OPS（Input / Output Operations per Second），每秒钟可以处理 I/O 数， 表示块存储处理读写（输出/输入）的能力，单位为次，衡量存储系统 I/O 处理能力。

指标	描述	数据访问方式
总IOPS	每秒执行的I/O操作总次数	对硬盘存储位置的不连续访问和连续访问
随机读IOPS	每秒执行的随机读I/O操作的平均次数	对硬盘存储位置的不连续访问
随机写IOPS	每秒执行的随机写I/O操作的平均次数	
顺序读IOPS	每秒执行的顺序读I/O操作的平均次数	对硬盘存储位置的连续访问
顺序写IOPS	每秒执行的顺序写I/O操作的平均次数	

性能指标 II：吞吐量 Throughput

- 吞吐量 = IOPS * I/O大小，单位时间内可以成功传输的数据数量，单位为MB/s，存储介质的 I/O 越大，IOPS 越高，那么每秒 I/O 吞吐量就越高。
- 当应用 I/O 大小较大，例如离线分析、数据仓库等应用，可以选择吞吐量更大的存储产品；如果部署大量顺序读写的应用，需要关注吞吐量。当应用 I/O 对时延较为敏感，比较随机且 I/O 大小相对较小，例如OLTP事务型数据库、企业级等应用，可以选择 IOPS 更高的ESSD云盘、SSD云盘。

- 对于 LLM 大语言模型来说，可以根据模型规模在 AI 集群中选择集中式存储吞吐量较高的产品。

性能指标 III：访问时延 Latency

- 时延指块存储处理一个 I/O 所需时间，即发起 I/O 请求到 I/O 处理完成的时间间隔，单位为 s、ms 或者 μs。过高的时延会导致应用性能下降或报错。
- 完成一个 IO 所花费时间（lat_io），对分布式存储系统来说，延时通常和如下几个因素有关:
  - lat_send：发送请求的延时
  - lat_recv：接收回复的延时
  - lat_srv_process：服务器处理请求的延时
  - lat_client_process: 客户端处理请求的延时
  - n: 一次 IO 需要请求的数量
  - $lat\_io = (lat\_send + lat\_recv + lat\_srv\_process + lat\_client\_process) * n$

性能指标 III：访问时延 Latency

- 此外，延迟还和如下因素有关：
  - 缓存及命中率：若 IO 请求缓存命中可以大幅减少 IO 延时。
  - IO size：IO 越大时延越大，但也不是绝对的，还和块对齐以及具体实现相关。
  - IO wait：IO 等待时间，无论是网络 IO 还是磁盘 IO 或是存储实现，多采用队列保存请求，当队列长度比较长时，就会有一部分时间是在队列中等待，进而影响 IO 延时。

性能指标 IIII：带宽 bandwidth

- 以字节为单位衡量每秒钟的IO速率（bandwidth），即每秒钟可以处理的数据量，常以 MB/s 或 GB/s 为单位，用于衡量存储系统的吞吐量。其通常与如下因素有关：
  - IO Size：每次IO的字节数量（io_size）
  - 并发数量：同时有多少并发请求（con_num）
  - IOPS：每秒钟完成的IO数量。iops = con_num * lat_io
  - $bandwidth = io\_size * iops = io\_size * con\_num * lat\_io$
- io_size 越大，带宽越大，但实践中通常4k、8k、16k当IO较少时满足这种正比关系，当io_size达到1M、4M、8M时系统吞吐量并不会继续增加。因为太大 IO实现也会分解为若干小的IO执行，此外io_size增加可能导致lat_io增加，可能会互相抵消。

小结&思考

小结

1. 从大模型的发展看大模型业务全流程跟存储的关系。
2. 大模型时代对存储的挑战主要在于高并发、高吞吐、小文件高 IOPS、稳定性。
3. 了解大模型训练过程中对存储的要求，存储的一般性能指标。

Reference

1. https://arxiv.org/pdf/1910.02054.pdf
2. https://arxiv.org/abs/2205.05198
3. https://zhuanlan.zhihu.com/p/643950399
4. https://www.marktechpost.com/2023/06/27/microsoft-research-introduces-phi-1-a-new-large-language-model-specialized-in-python-coding-with-significant-smaller-size-than-competing-models/
5. https://zhuanlan.zhihu.com/p/665172400
6. https://www.elunicornio.co/?q=beginner-s-guide-to-build-large-language-models-from-ee-Db78K7ux
7. https://help.aliyun.com/zh/ecs/user-guide/block-storage-performance
8. https://developer.aliyun.com/article/774807
9. https://zhuanlan.zhihu.com/p/510124232
10. https://support.huawei.com/enterprise/zh/doc/EDOC1000181438/b44e5c36

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