

# 云原生技术优化模型推理

主讲人：王璠

# Content

## 目录

1. 模型推理概述
2. 模型推理挑战
3. 云原生技术概述
4. 云原生技术优化模型推理

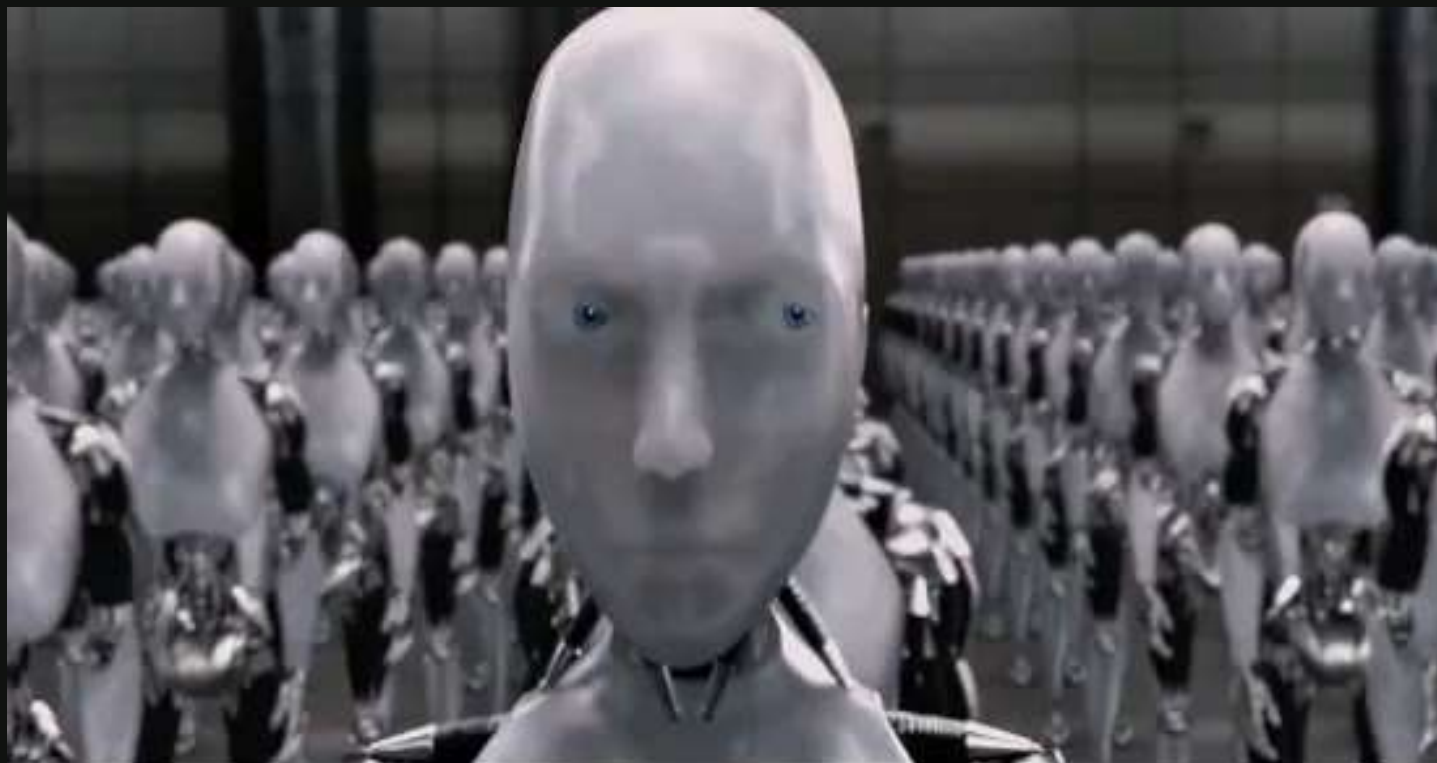
# Part 01

## 模型推理概述

# ■ 什么是人工智能

**人工智能 (AI)：** 模拟人类智能的计算机系统，通过学习和适应来执行任务。

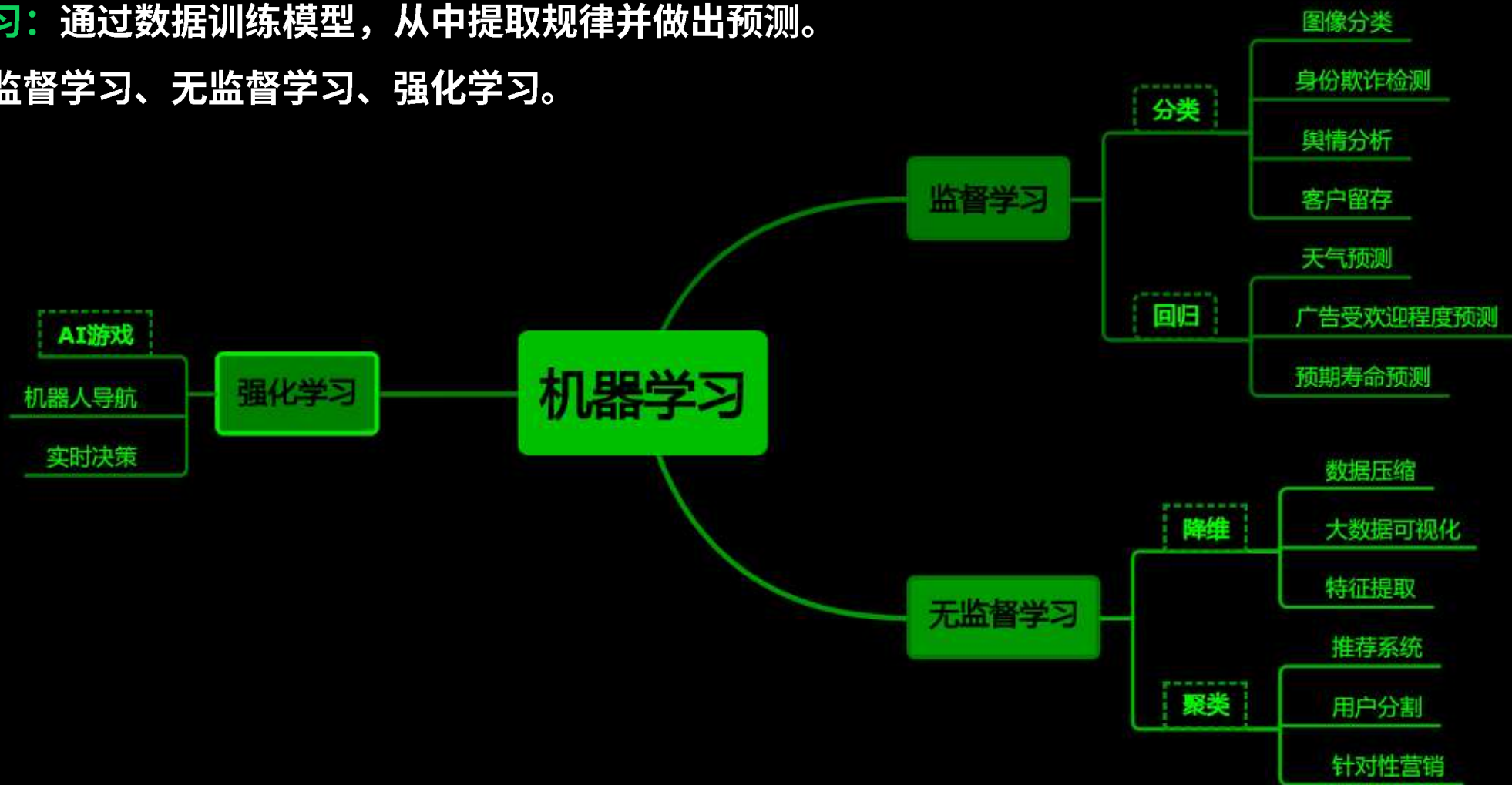
**主要分支：** 机器学习 (ML) 和深度学习 (DL)。



# ■ 什么是机器学习

**机器学习：**通过数据训练模型，从中提取规律并做出预测。

**分类：**监督学习、无监督学习、强化学习。



# ■ 什么是机器学习

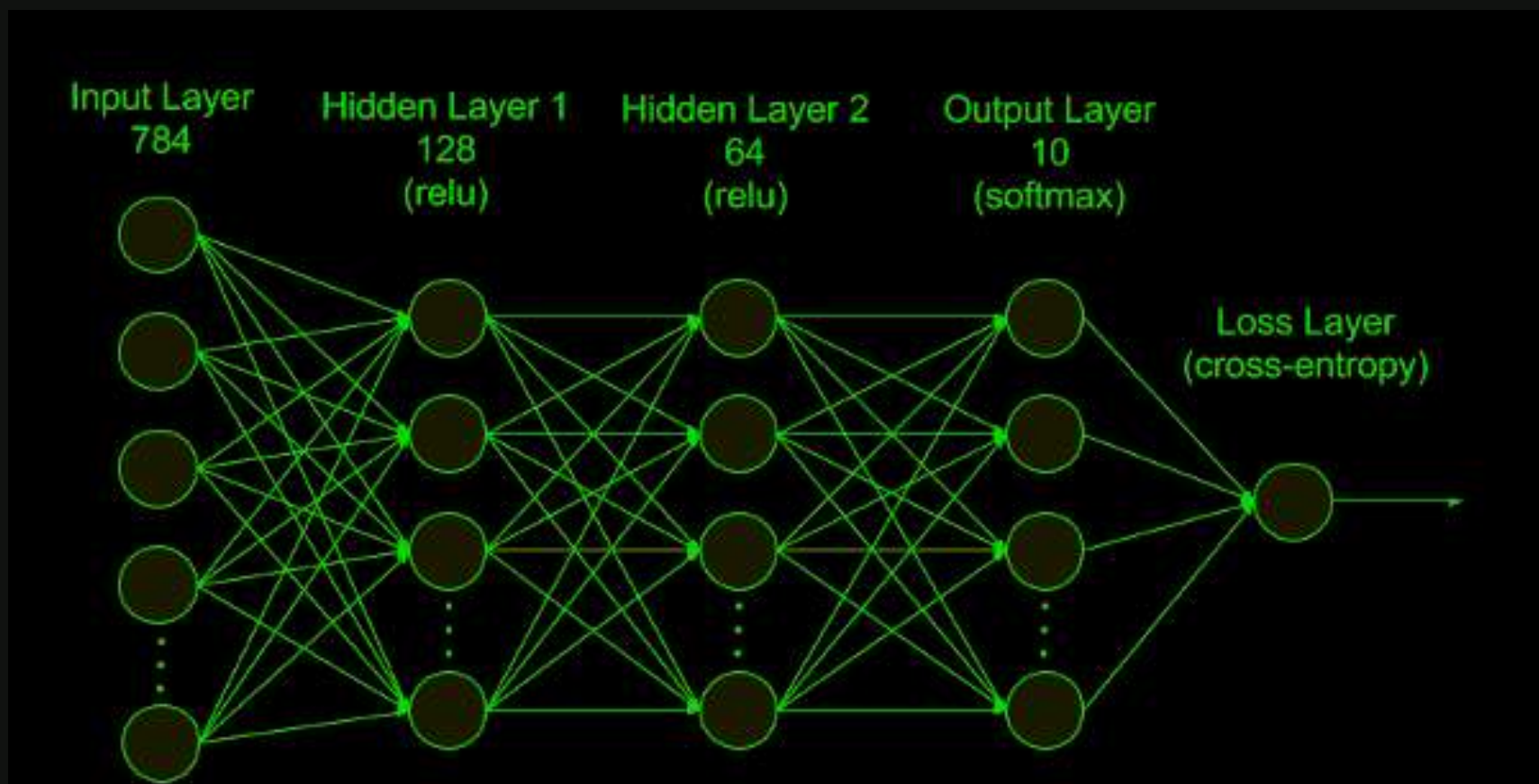
引用资料来源: <https://www.bilibili.com/video/BV1ot411P77s?t=126.5>

KnowingAI 知智 Bilibili

# ■ 什么是深度学习

**深度学习：**基于人工神经网络的机器学习方法，适用于处理复杂数据。

**典型应用：**图像识别、语音识别、自然语言处理。

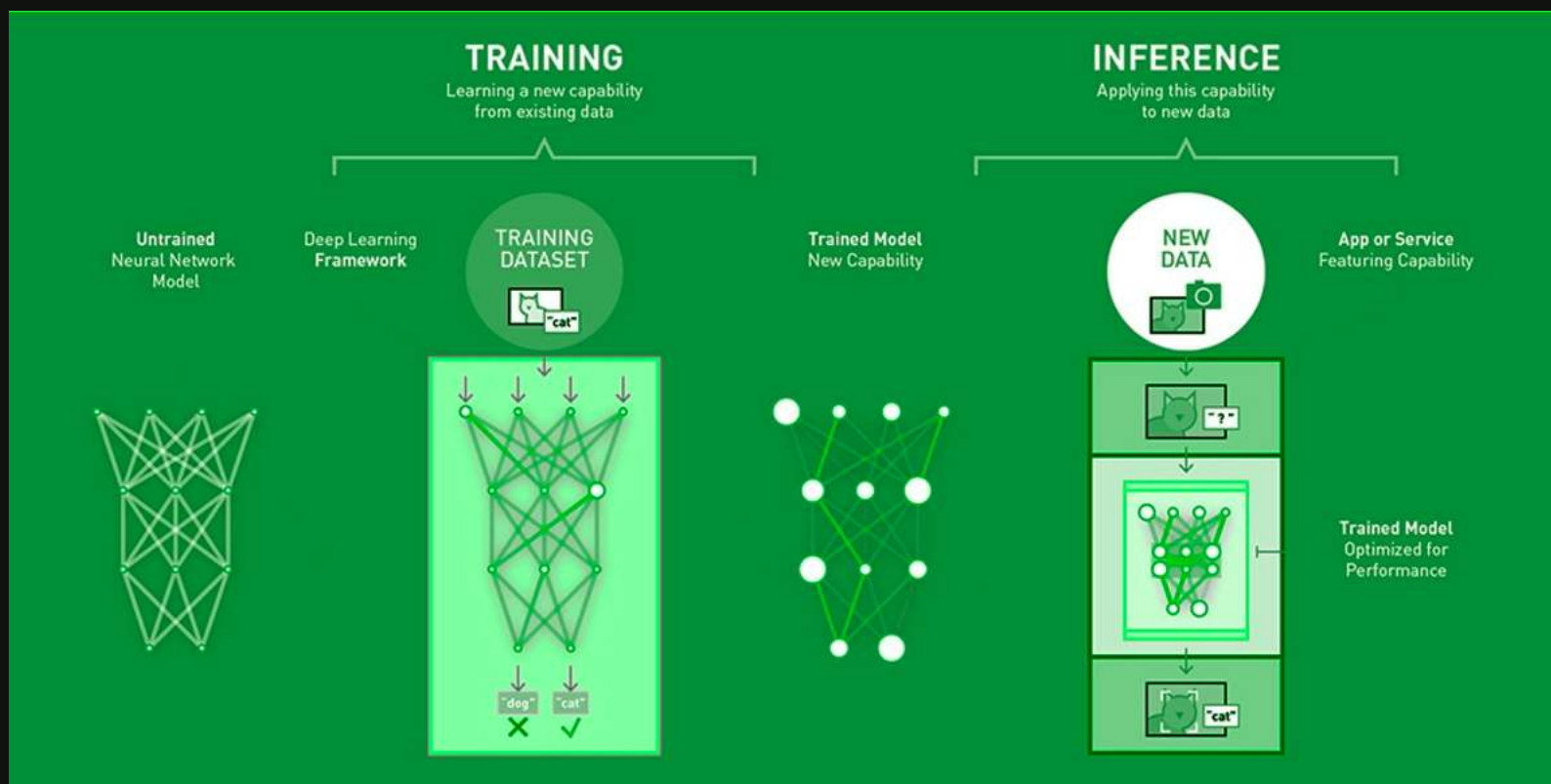




## ■ 从训练到推理

**训练：**使用大量数据训练机器学习模型，调整模型参数以提高准确性。

**推理：**在训练完成后，使用新数据进行预测或分类。



引用的部分论文资料：<https://arxiv.org/abs/2401.02038>



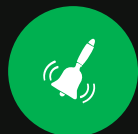
# Part 02

## 模型推理挑战

## ■ 挑战



资源利用率低



延迟敏感



跨平台兼容性



模型部署困难



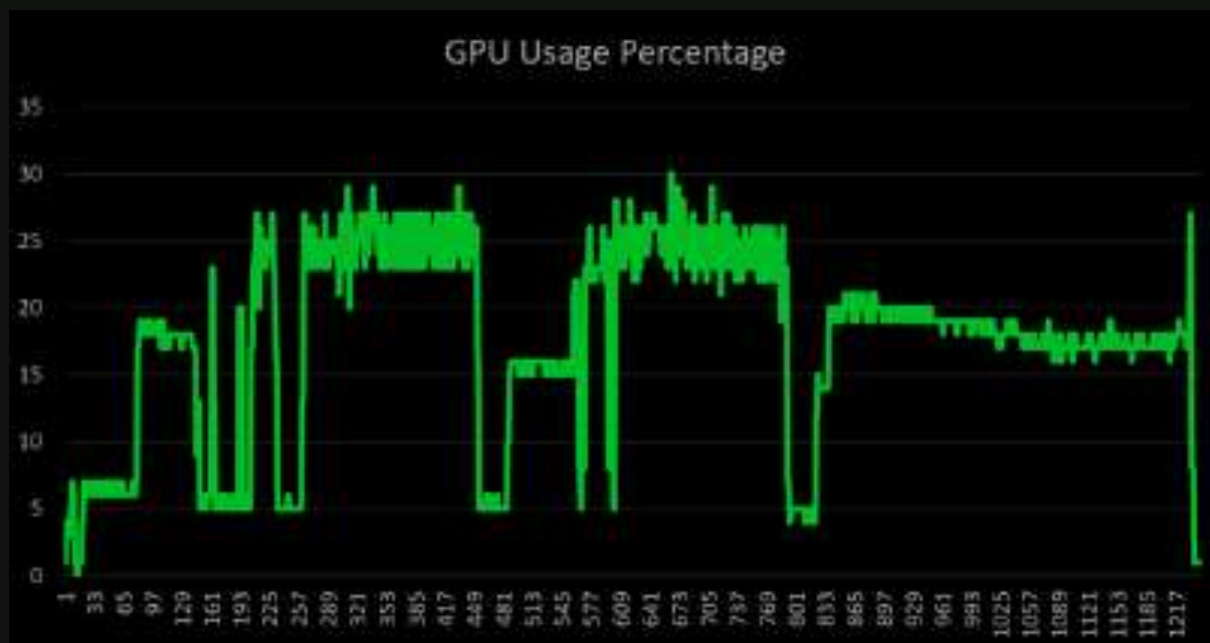
## ■ 资源利用率低

**场景：**一家提供图片识别服务的公司。

高峰期间资源利用率高，非高峰期资源闲置，增加运营成本。

**背景：**在非高峰时段，模型推理请求量较少，导致服务器资源闲置。

高峰时段请求量激增，但平时资源闲置，整体资源利用率低。



## ■ 延迟敏感

**业务需求：**某些业务需求必须在毫秒级响应，否则会造成巨大损失。

---

**用户体验：**高延迟会严重影响用户体验，降低用户满意度。

---

**实时性要求：**许多应用场景如自动驾驶、实时推荐系统要求低延迟。



## ■ 跨平台兼容性

### 1. 硬件兼容性

不同平台上的硬件架构（如 CPU、GPU、TPU）可能会导致模型推理的不一致性。

### 2. 软件依赖性

模型推理依赖特定的软件库和框架，不同平台上的库版本和实现可能会有所不同。

### 3. 平台特定优化

不同平台上的硬件可能对模型推理有不同的优化措施，例如 CPU 和 GPU 的不同优化策略。

## ■ 模型部署困难

### 集成

模型集成包括构建运行模型的基础设施和以可消费和支持的形式实现模型，代码复用和避免工程中的反模式（如抽象边界侵蚀、校正级联、胶水代码、管道丛林和配置债务等）是需要考虑的问题，同时让研究人员参与整个开发过程有助于提高产品交付的速度和质量。

### 监测

监测对于维护机器学习系统至关重要，但确定关键指标和触发系统警报的阈值是一个挑战，反馈循环可能会影响模型行为，异常检测和使用专门的工具（如早期干预系统）进行监测是重要的，同时需要注意监测指标与验证指标的重叠。

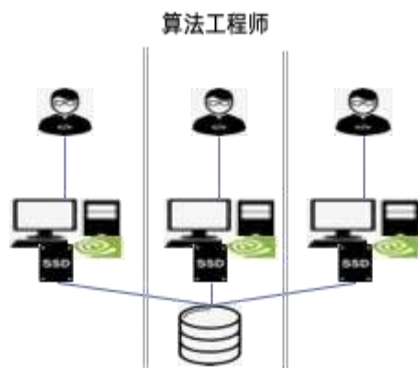
### 更新

模型更新需要考虑概念漂移（数据集变化）的问题，检测概念漂移对于维护模型在生产中的性能很重要，此外，还需要解决如何将模型更新部署到生产环境的问题，同时要注意模型更新可能会对用户或下游系统造成损害，因此需要兼容和可靠的更新方法。

# ■ 摆脱 AI 生产“小作坊”

## 市场现状

- 根据CNCF 2021年度报告，96%的企业正在使用或评估 Kubernetes。
- Gartner预测到 2026 年超过80%的企业将使用生成式AI的API或模型，或在生产环境中部署支持生成式AI的应用，而在2023年初这一比例不到5%。



AI小作坊



## 用户需求

- 用户希望在Kubernetes容器集群中管理GPU资源，运行深度学习和大数据任务，弹性管理AI服务。




云原生AI平台



# Part 03

## 云原生技术概述

## ■ 容器改变世界

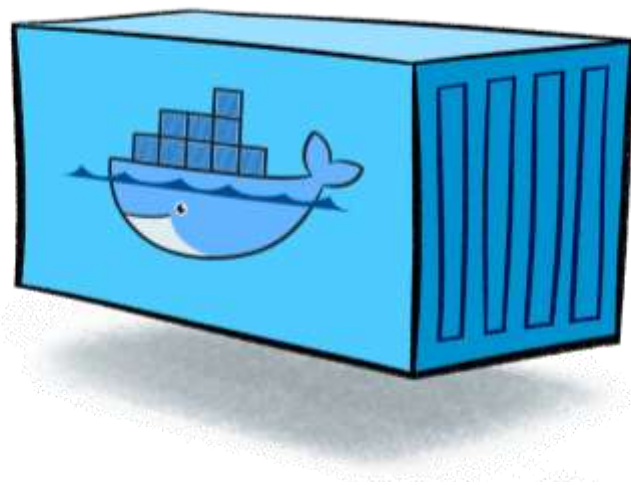
 静态网站

 用户数据库

 站点前端或 API

 队列

 分析数据库



容器类似集装箱的作用

使得持续交付体系变得标准化

实现了应用交付的标准化

降低了应用交付成本



开发者电脑



私有云



公有云

## ■ 什么是 Docker

- Docker 是一种 Linux 容器管理引擎
- Docker 是目前最火热的开源项目之一
  - Docker 公司主导开发
  - 遵循 Apache License 2.0 许可证协议
  - 托管于 GitHub
  - Go 语言编写
  - 适用于 Linux、Windows、macOS
- Docker 是一种实现打包、输送、运行任意应用的容器解决方案



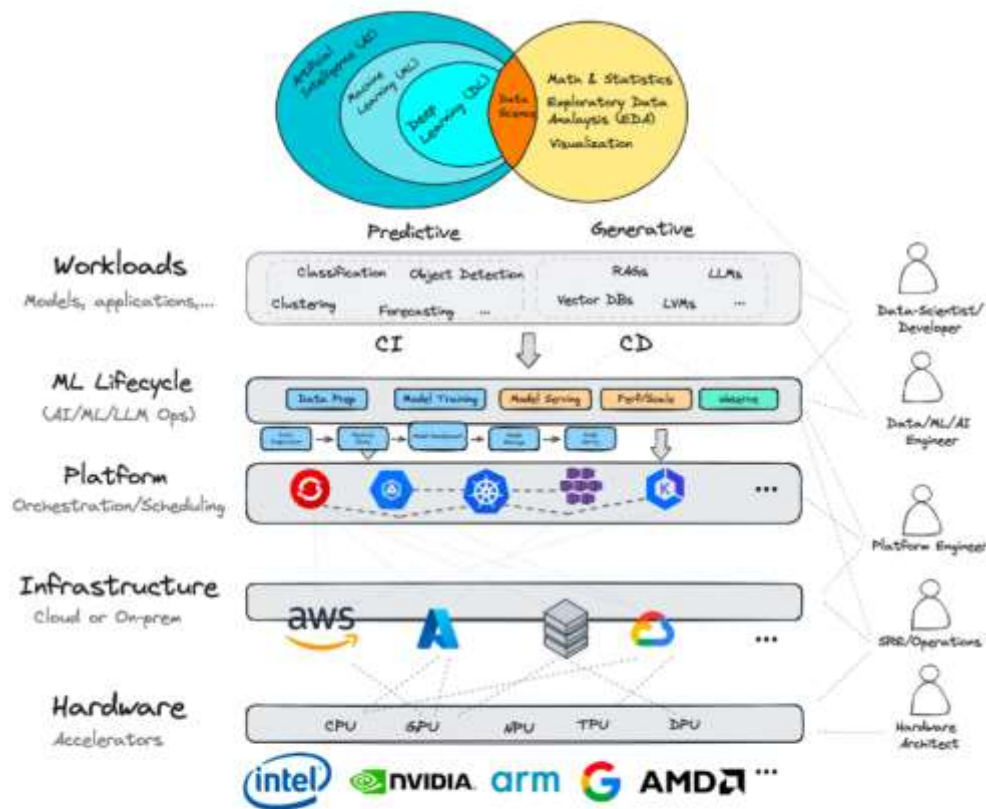
# ■ 什么是 Kubernetes

- Kubernetes 是 Google 开源的容器集群管理系统，由 Google 多年大规模容器管理技术 Borg 演化而来并赠给云原生计算基金会（CNCF），其主要功能包括
  - 基于容器的应用部署、维护和滚动升级
  - 负载均衡和服务发现
  - 跨机器和跨地区的集群调度
  - 自动伸缩
  - 无状态服务和有状态服务
  - 广泛的存储支持
  - 插件机制保证扩展性
- Kubernetes 发展非常迅速，已经成为容器编排领域的领导者



# ■ Cloud Native AI

挑战/需求	生成式人工智能	预测式人工智能
计算能力	极高。需要专用硬件。	中到高。通用硬件足够。
数据量和多样性	用于训练的大量、多样化的数据集。	特定的历史数据用于预测。
模型训练和微调	使用专门的计算方法进行复杂的迭代训练。	适度的训练
可扩展性和弹性	高度可扩展和弹性的基础设施（可变和密集的计算需求）	可扩展性是必要的，但弹性要求较低。批处理或事件驱动的任务。
存储与吞吐量	高性能的存储和出色的吞吐量。不同的数据类型需要高吞吐量和低延迟的访问数据。	需要高吞吐量和低延迟访问数据。
网络	高带宽和低延迟的数据传输和模型同步（例如，在分布式训练期间）。	数据访问的一致性和可靠性。



Disclaimer: The above is not an exhaustive list, the components of the figure are used to illustrate the layers of the stack and are not by any means a complete list of vendors/providers

# Part 04

## 云原生技术优化模型推理

## ■ 云原生技术优化模型推理

- 容器化与编排
- 分布式缓存
- 无服务器架构
- 基于 KubeEdge 的边缘 AI 实现
- MLOps





## ■ 容器化和编排



### 容器化

- 使用容器技术（如 Docker）将应用及其依赖打包成独立的单元
- 确保一致的运行环境，避免环境配置问题



### 重要性

- 提高应用部署和管理的效率
- 便于在不同环境中运行和迁移应用

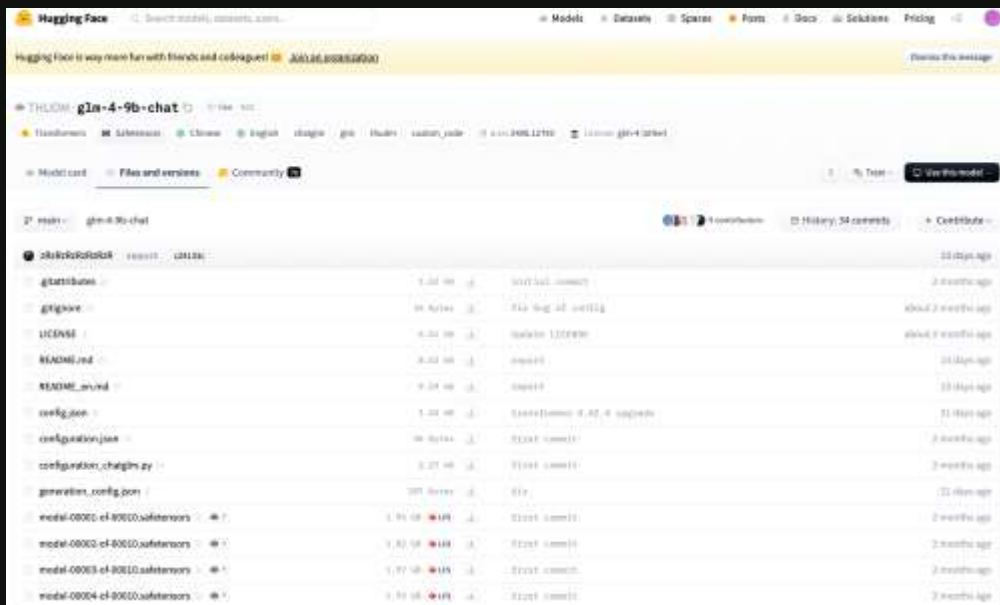


### 编排

- 使用编排工具（如 Kubernetes）管理容器的部署、扩展和运维
- 确保高可用性、弹性伸缩和自动化管理

# ■ 案例 1 - 构建模型镜像

## 1. 拉取模型文件



## 3. 构建 Docker 镜像

```
docker build -t vllm-openai-tiktoken-glm4-9b-server:v2.0.1 -f Dockerfile /data/llms/glm4-9b-chat
```

## 2. 创建以下 Dockerfile

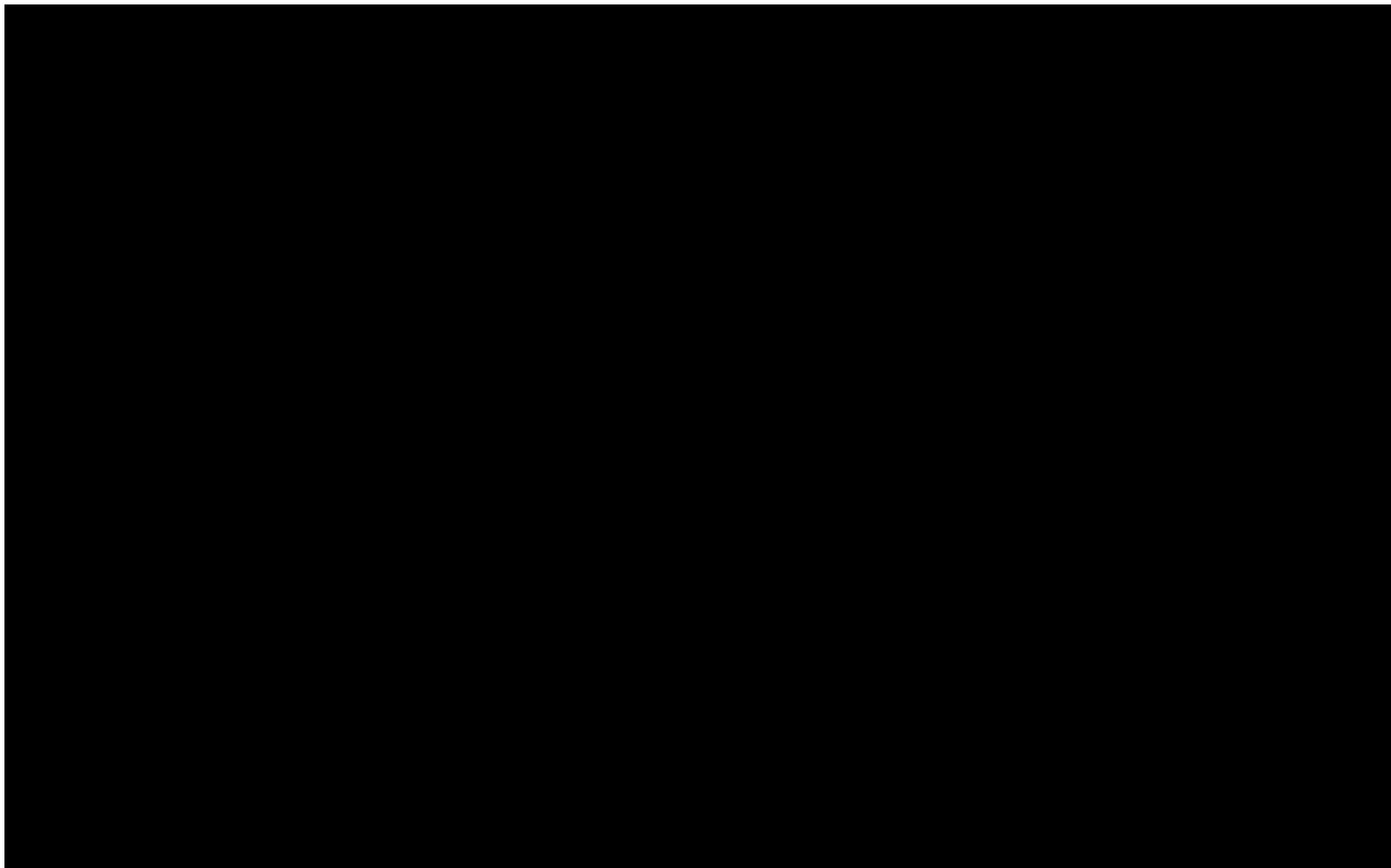
```
# 基于 vllm/vllm-openai:v0.4.1 构建  
FROM vllm/vllm-openai:v0.4.1
```

```
# 安装 Python 包  
RUN pip install tiktoken
```

```
# 复制模型到容器  
COPY . /data
```

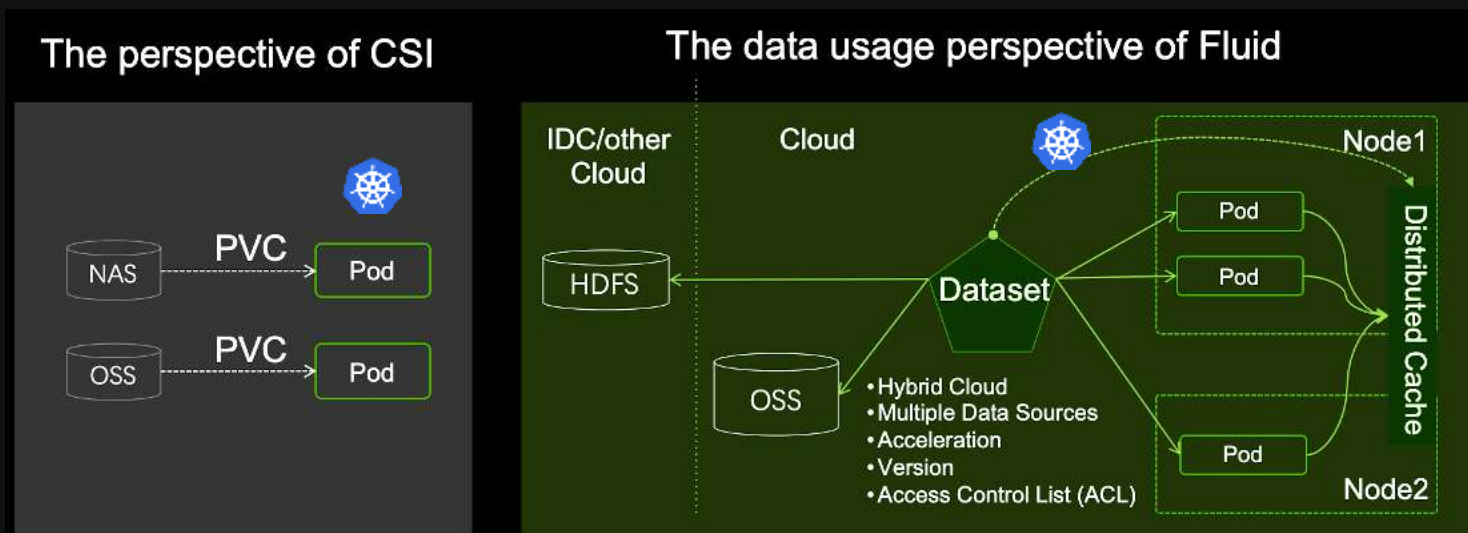
```
# 使用 shell 形式的 CMD  
ENTRYPOINT ["python3", "-m",  
"vllm.entrypoints.openai.api_server", "--model",  
"/data", "--trust-remote-code"]
```

## ■ 案例 1-一分钟部署大模型

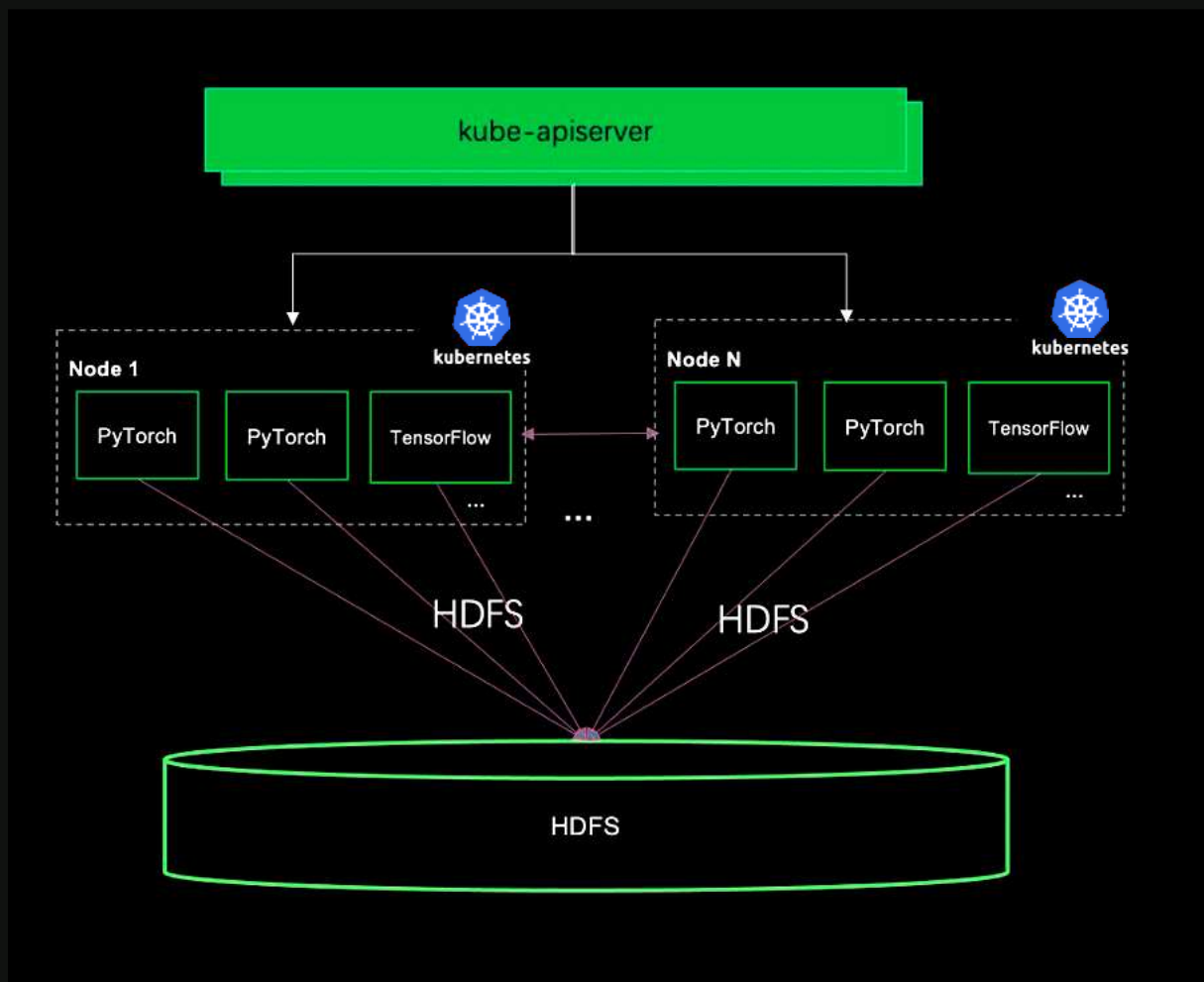


## ■ 分布式缓存

- **数据集抽象化:** Fluid 将数据集抽象为一等公民的资源,使得应用程序可以像使用其他 Kubernetes 资源一样使用数据集。
- **分布式缓存:** Fluid 提供了一个分布式缓存层,可以将数据集缓存到 Kubernetes 集群中的节点上,从而提高数据访问性能。
- **数据集生命周期管理:** Fluid 可以管理数据集的生命周期,包括数据集的创建、更新、删除等操作。
- **异构数据支持:** Fluid 支持多种类型的数据集,包括 HDFS、Alluxio、OSS 等。
- **数据亲和性调度:** Fluid 可以根据数据集的位置信息,将应用程序调度到靠近数据的节点上,从而减少数据传输时间。



## ■ 案例 2 - 微博海量深度学习模型训练效率跃升

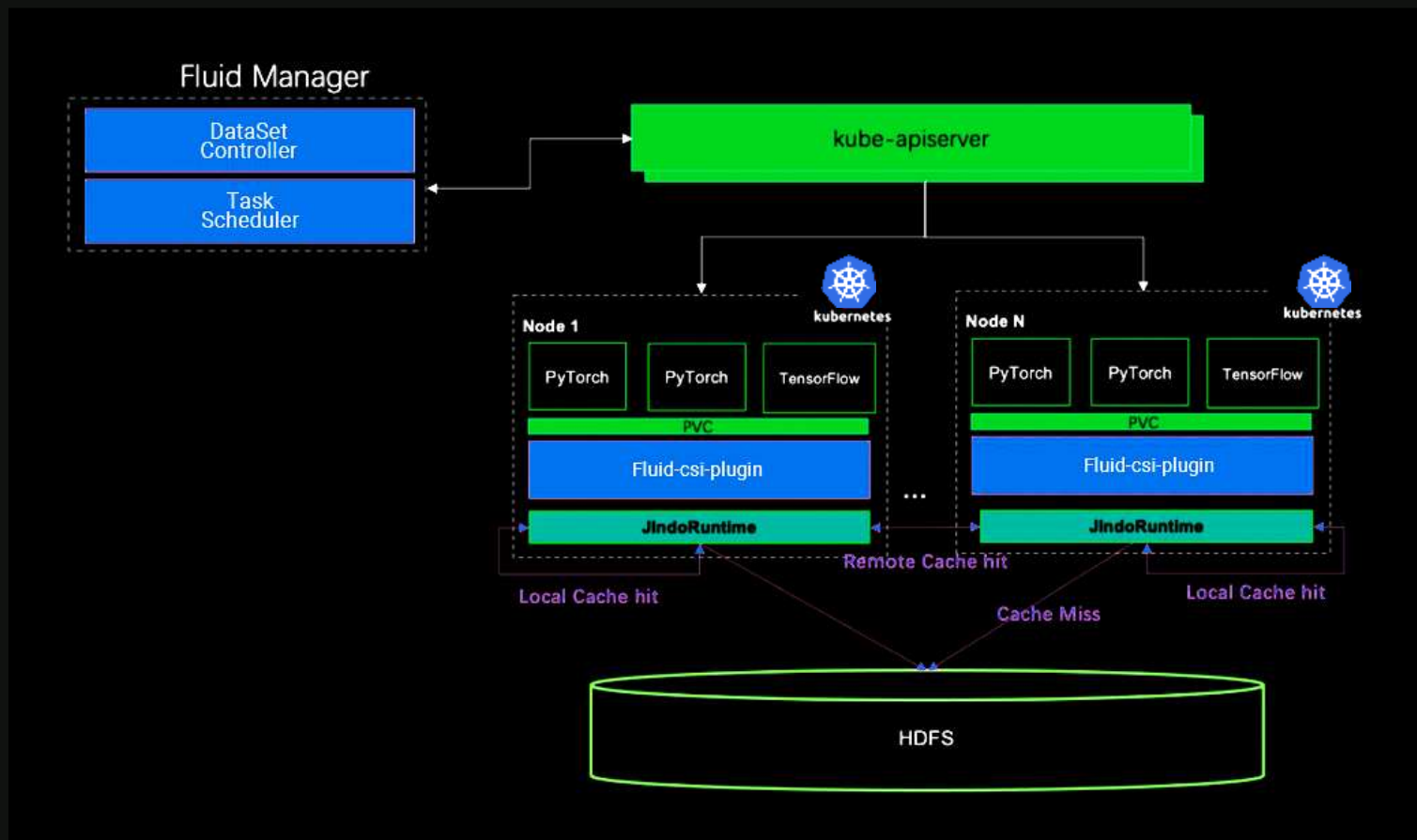


### 传统机器学习平台遇到的挑战

- **架构问题**：存储与计算分离导致数据访问高延时。
- **Kubernetes 调度问题**：无法感知数据缓存，导致性能未提升。
- **深度学习框架限制**：多数框架不支持 HDFS 接口，需要额外开发。
- **HDFS 瓶颈**：并发访问成为性能瓶颈，影响稳定性。

引用资料来源: <https://mp.weixin.qq.com/s/-gsSY71oWxF0pTErICNK1w>

## ■ 案例 2 - 微博海量深度学习模型训练效率跃升



**Fluid:** 解决数据访问延时高、多数据源分析难等问题。

**JindoRuntime:** 提供远端文件访问和缓存，支持多种存储产品。

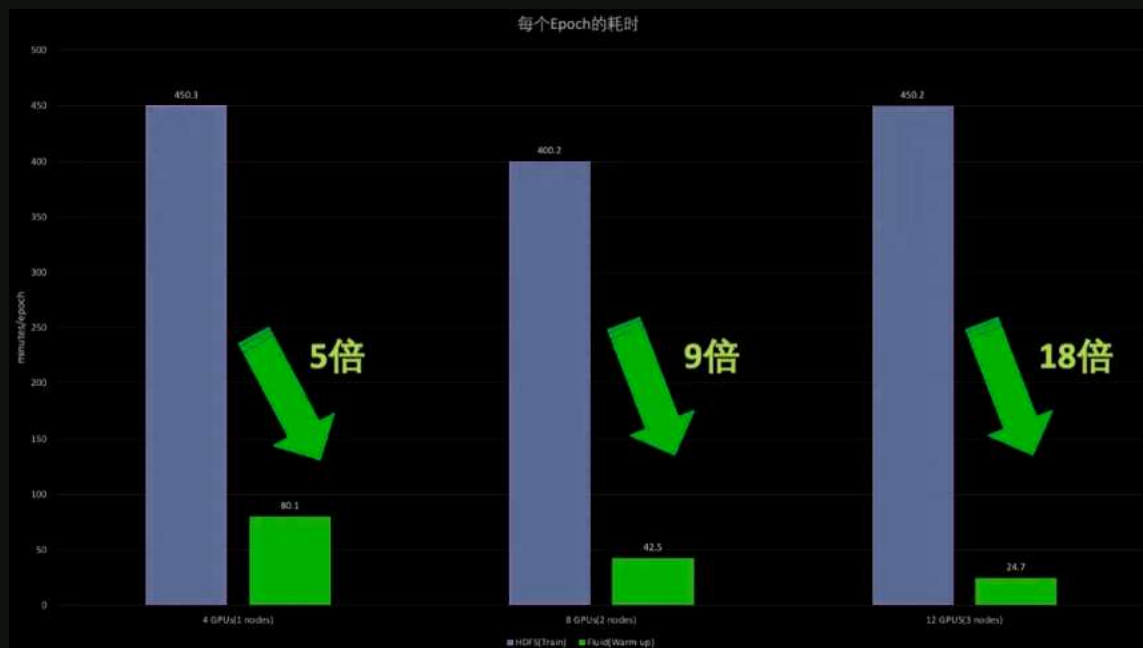
基于 Fluid 的新架构方案

引用资料来源: <https://mp.weixin.qq.com/s/-gsSY71oWxF0pTErICNK1w>

## ■ 案例 2 - 微博海量深度学习模型训练效率跃升

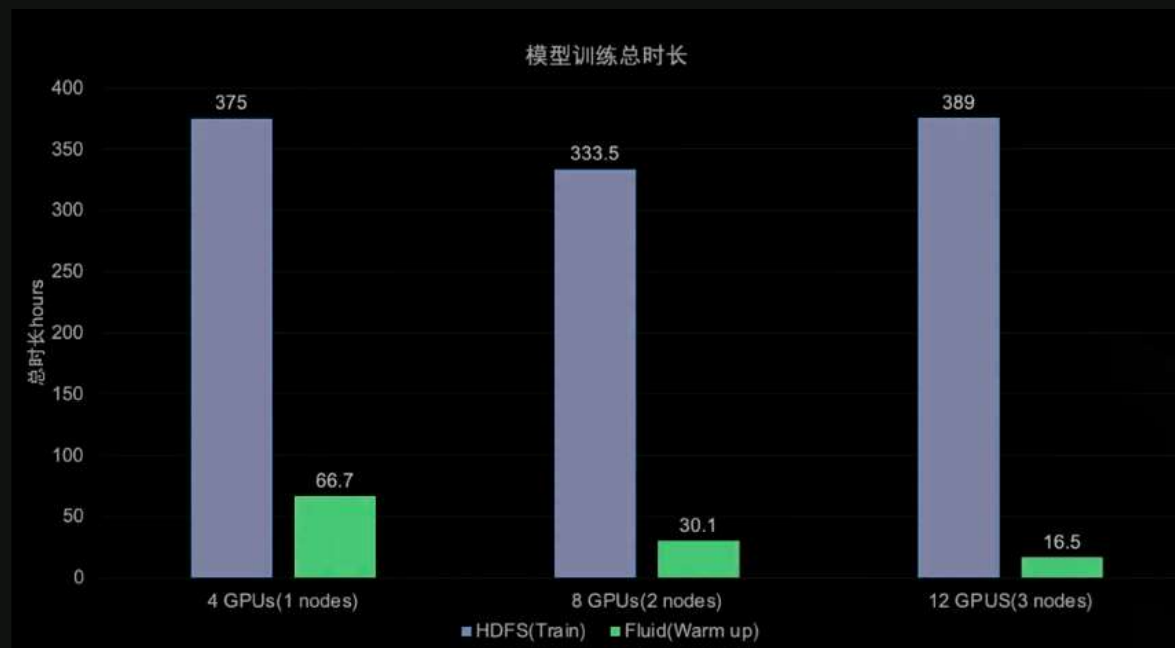
### 加速效果:

1 机 4 卡 5 倍加速, 2 机 8 卡 9 倍加速, 3 机 12 卡 18 倍加速。



### 训练速度:

训练总时长由原来的 389 小时 (16 天) 缩短到了 16 小时。



引用资料来源: <https://mp.weixin.qq.com/s/-gsSY71oWxF0pTErICNK1w>



# ■ 无服务器架构

01

**成本效益:** 无服务计算只需要为实际使用的资源付费, 无需支付闲置时的维护费用, 相比传统虚拟机更加经济实惠。

02

**适合物联网应用:** 无服务计算可以提供细粒度的弹性伸缩和资源配置, 非常适合物联网应用的尖峰负载。

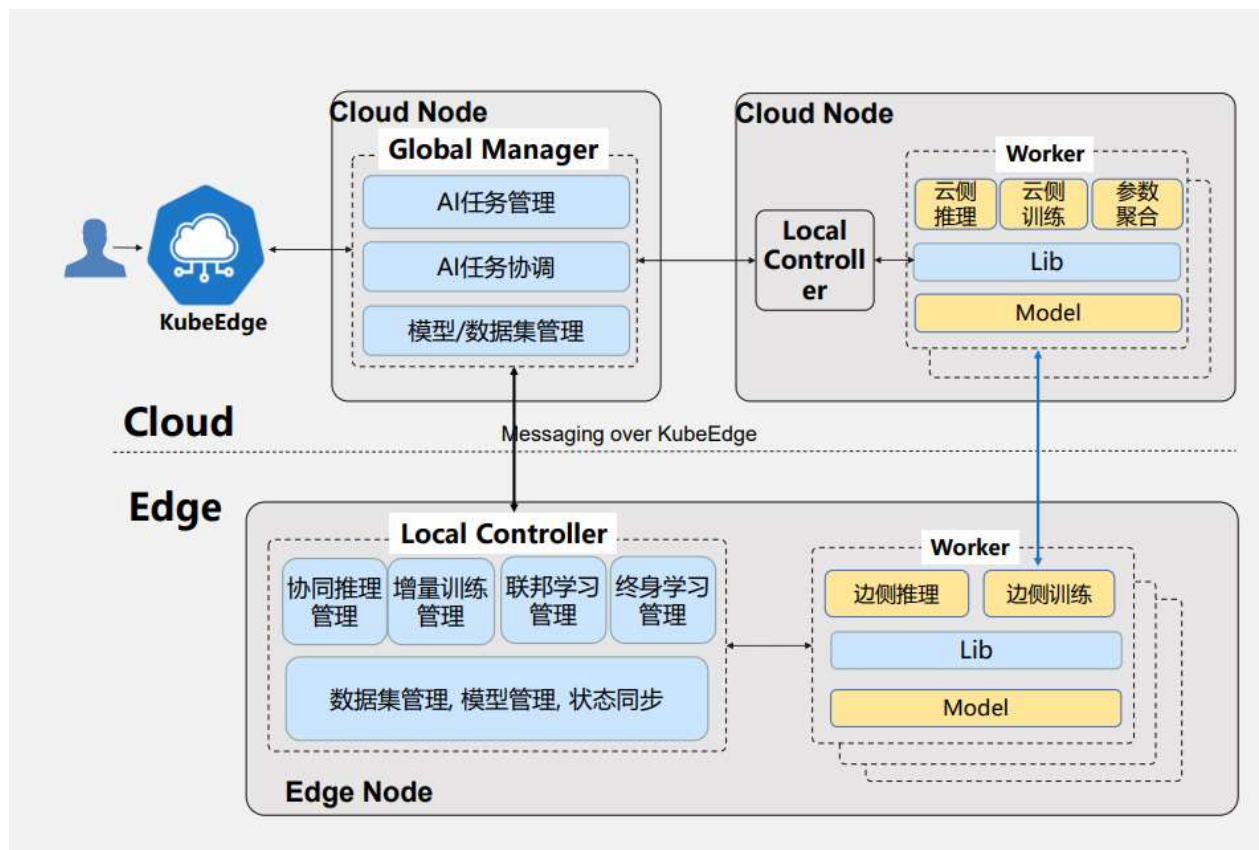
03

**支持并行执行:** 无服务计算可以支持应用程序并行执行计算任务, 例如国防物联网应用中需要及时响应并不影响其他关键任务。

04

**与云边缘计算集成灵活:** 虽然无服务计算最初是为云端设计的, 但现在也在向边缘计算延伸。

# ■ 基于 KubeEdge 的边缘 AI 实现 —— Sedna



## 1. 提供AI边云协同框架。

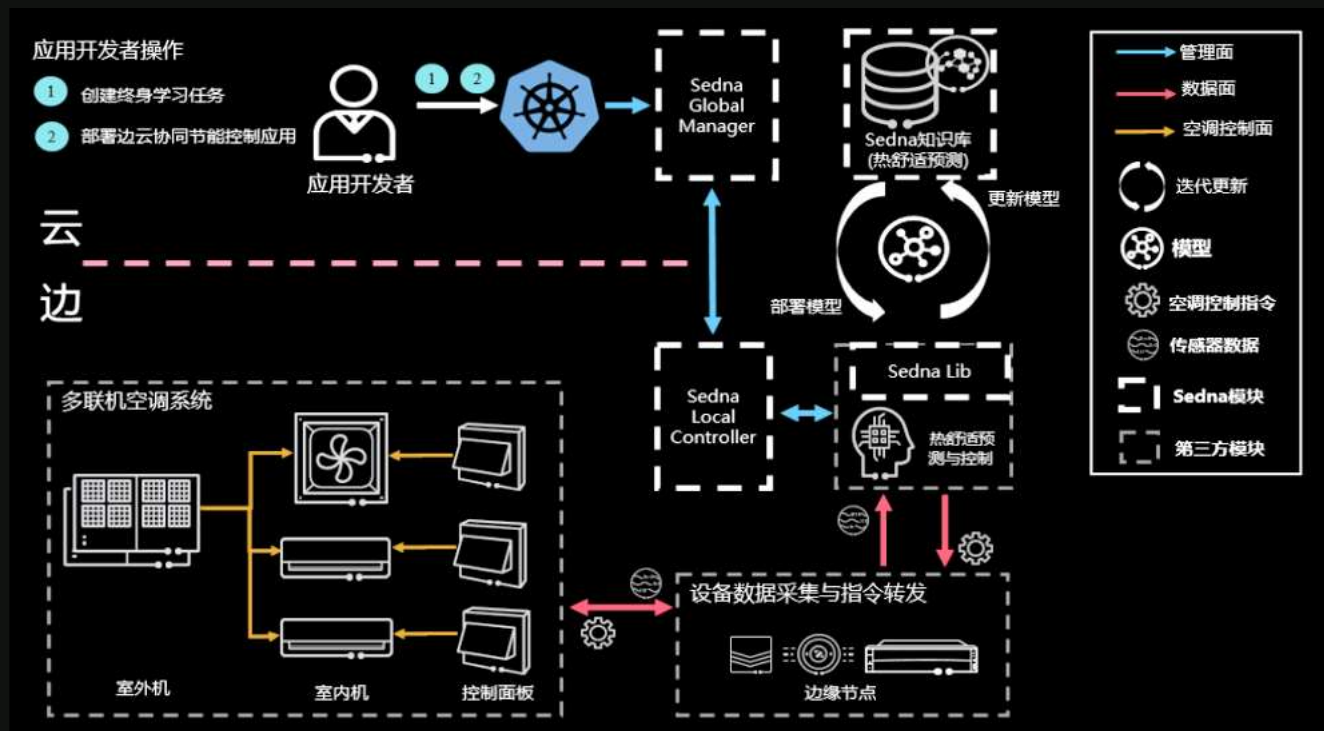
- 提供了边云协同数据集管理、模型管理

## 2. 支持多种边云协同训练和推理模式。包括联合推理，增量训练，联邦训练，终身学习

## 3. 具有开放生态。

- 兼容主流 AI 框架 TensorFlow、Pytorch、PaddlePaddle、MindSpore 等
- 提供可扩展接口，方便第三方算法快速集成

## ■ 案例 3 - 基于 KubeEdge/Sedna 的楼宇热舒适度预测控制



### 背景:

1. 智能楼宇：智慧城市的重要组成部分。
2. 热舒适度：对环境冷热的满意程度，连接物理参数和主观评估。
3. 挑战：数据异构与小样本问题。

### 基于Sedna的热舒适预测控制:

1. 云边协同：云侧Sedna知识库利用历史数据集初始化，边侧应用提供推理更新接口。
2. 终身学习：已知任务直接推理，未知任务联合知识库推理并学习更新知识库。

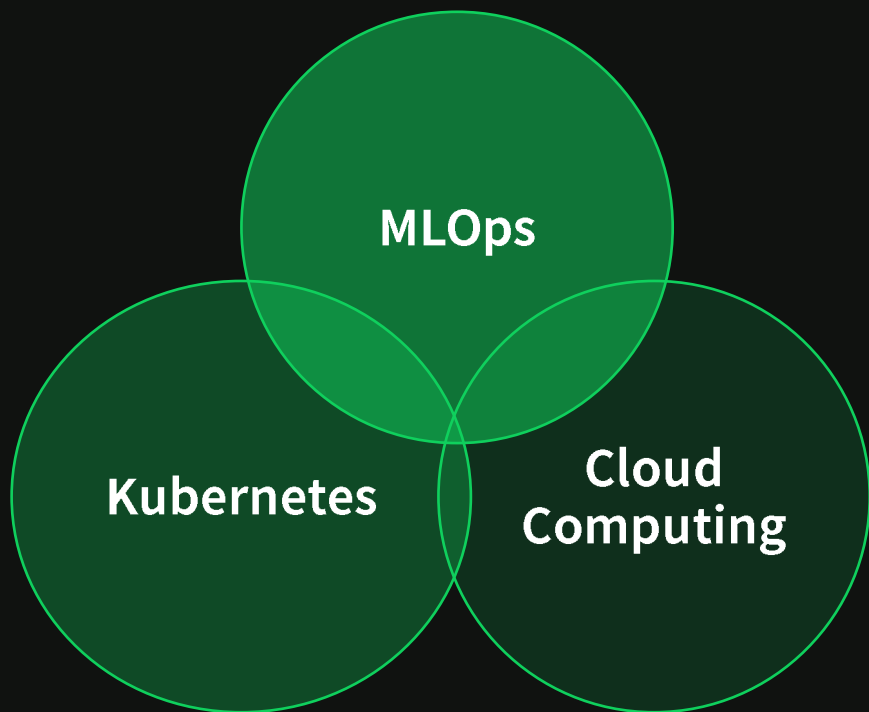
### 实验结果:

1. 在KotaKinabalu数据集上，热舒适度预测率提升24.04%。
2. 为楼宇温度调整策略提供依据

引用资料来源: <https://bbs.huaweicloud.com/blogs/432089>

# ■ MLOps

- **DevOps**: 开发和运行大规模软件系统的做法，包括持续集成(CI)和持续交付(CD)。
- **MLOps**: 将 DevOps 原则应用于机器学习系统，实现自动化和监控。

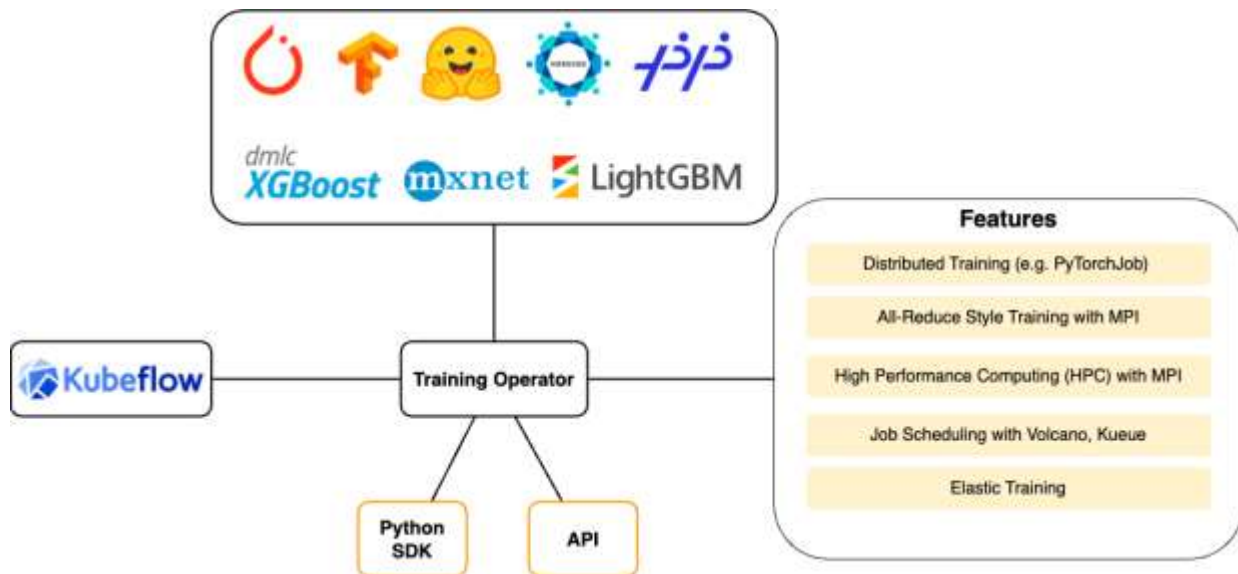


MLOps 级别 0: 手动过程，无自动化，脚本驱动，发布迭代不频繁。

MLOps 级别 1: 流水线自动化，持续训练模型，自动化数据和模型验证。

MLOps 级别 2: CI/CD流水线自动化，快速、可靠地更新流水线。

## ■ MLOps - 工具



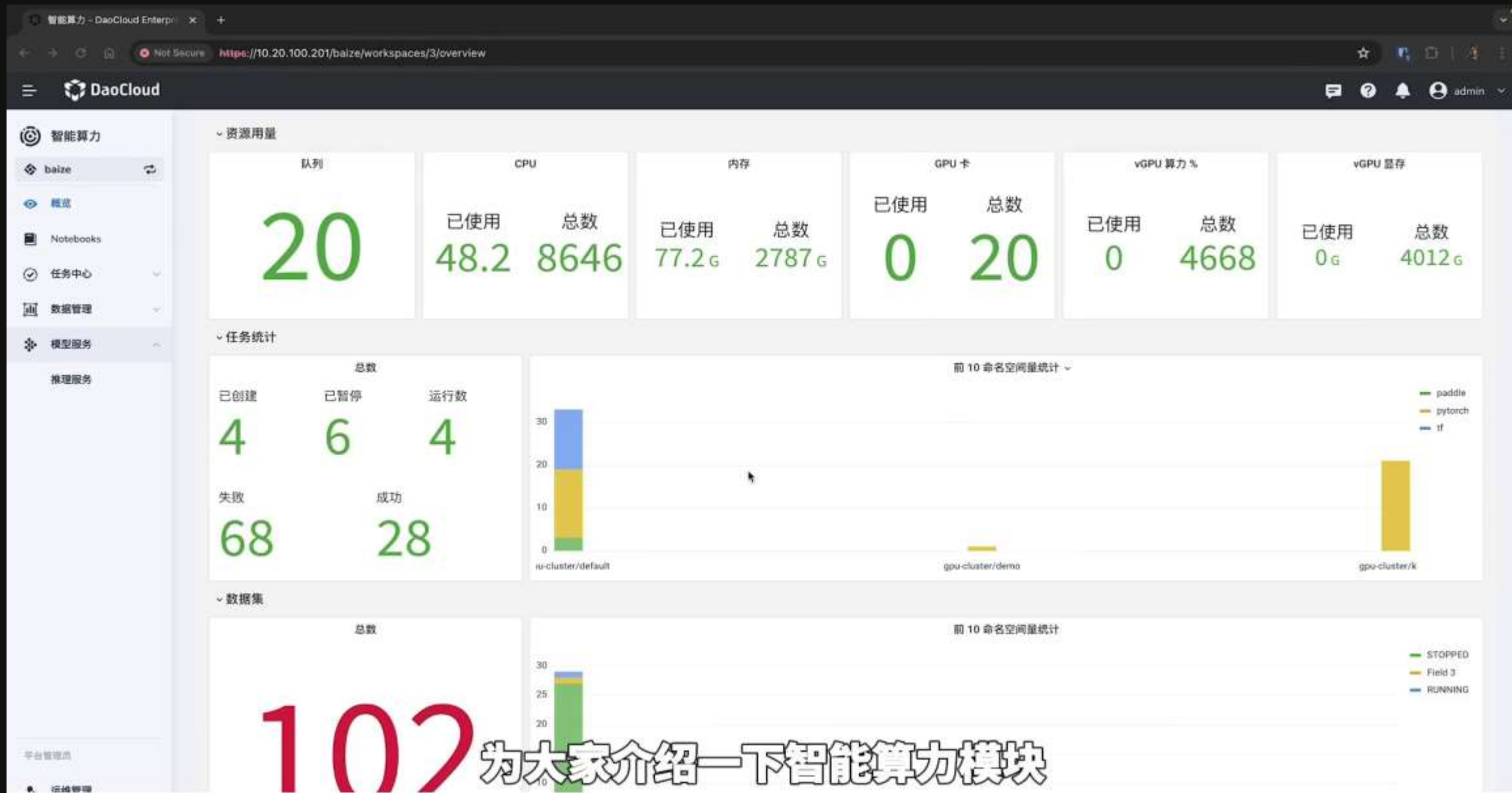
### 什么是Kubeflow?

- 一个开源平台，用于在Kubernetes上运行机器学习（ML）工作负载。
- 简化了在Kubernetes上部署、管理和运行机器学习 workflow 的过程，具有可扩展性和便携性。

### 核心组件：

- Central Dashboard: 提供访问 Kubeflow 及其生态系统组件的 Web 界面。
- Kubeflow Notebooks: 提供交互式环境进行数据探索和模型开发。
- Kubeflow Pipelines: 定义和执行 ML 工作流的 DAG 形式。
- Training Operator: 支持大规模分布式训练。
- Serving: 部署训练模型为生产就绪型服务。
- Metadata: 跟踪和管理 ML 实验、运行和工件的元数据。

## ■ 案例 4 - 模型训练



The background of the slide is a dark, swirling pattern of green and black. The green is a vibrant, slightly neon shade, and the black is a deep, dark charcoal. The pattern consists of fluid, organic shapes that swirl and flow, creating a sense of movement and depth. The overall effect is modern and tech-oriented.

# Thanks.