# 模型推理服务化框架Triton保姆式教程（二）：架构解析

吃果冻不吐果冻皮









前面给大家分享了**模型推理服务化框架Triton保姆式教程（一）：快速入门**，对于一个模型推理服务框架来说，通常关注的指标有延时、吞吐、效率、灵活性和可扩展性⁺等。那么，针对每个点应该如何去解决，这是在进行推理框架⁺设计时需要去考虑的问题。因此，本文从更高的技术视野来看一个推理系统以及Triton的架构是如何进行设计的。

## 推理系统架构设计

对于一个推理系统来说，主要由两部分组成：客户端⁺和服务端。

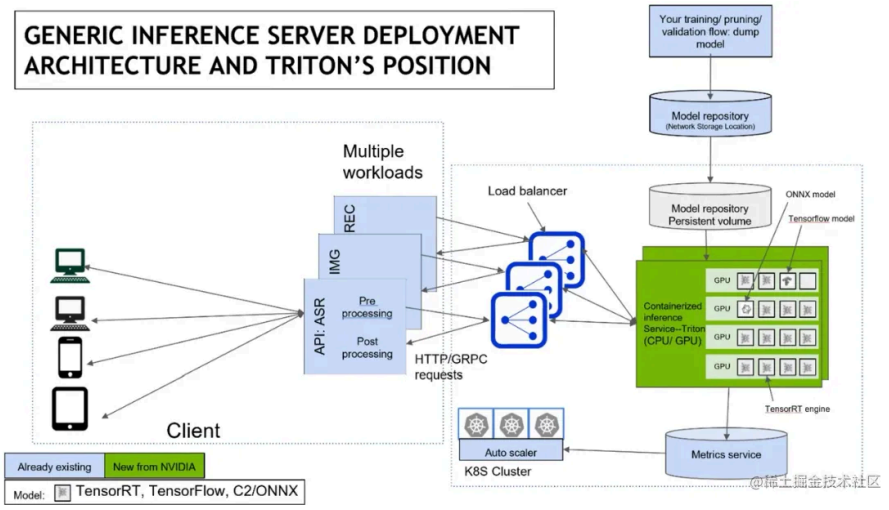



image.png

对于客户端来说，通常比较简单，如上图左半部分所示。它可以是手机、平板、电脑或者下游应用系统等。

而对于服务端，通常比较复杂，它也是整个推理系统的核心，如上图右半部分所示。下面，我们来看下服务端架构应该怎么设计？

首先，它得有一个**部署平台/集群**（如：Kubernetes）。用于对一个推理服务的生命周期的管理（模型的加载/卸载，模型服务实例的弹性扩缩容等）。

然后，它还应该有一个**负载均衡器**⁺（如：Ingress），解决模型推理服务实例负载均衡的问题。将客户端的请求，均衡的分配给推理服务实例。

其次，它还应该有一个**模型仓库**（如：本地文件系统），对模型权重⁺文件和模型输入输出配置进行管理。

之后，它还应该有一个**模型监控服务**，用于观察模型的宏观/微观的数据。

最后，也是最核心的，它还应该有一个**推理服务**，进行模型推理。比如，Triton就是一个推理服务化框架，这也是Triton的定位：

- 单节点推理服务。
- 支持异构，比如：CPU、GPU、TPU等。
- 支持多框架，比如：TensorRT、Pytorch、FasterTransformer等。

接下来，我们一起来看一下 Triton 的整体架构是如何设计的？

知乎 首发于 AI工程（MLOps）

下图展示了 Triton 推理服务的整体架构。

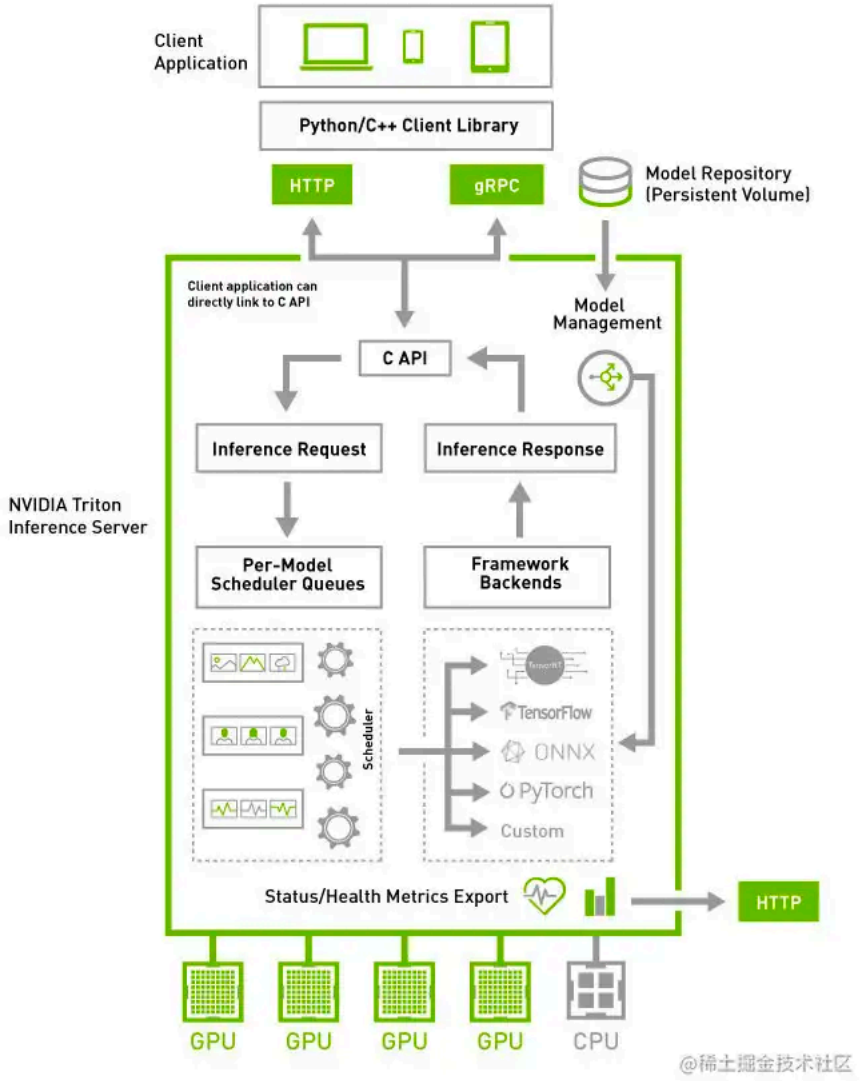



image.png

Triton推理服务的大体流程为推理请求通过 HTTP/REST、GRPC 或 C API 到达推理服务，然后路由到每个模型相关的调度队列中。 Triton 实现了多种调度和批处理算法，可以在每个模型上面进行配置。 每个模型的调度器可选择执行推理请求的批处理，然后将请求传递给与模型类型对应的后端。后端使用批处理请求中提供的输入执行推理以生成请求的输出。 然后返回输出。

下面是Triton生态的重要组成组件和重要特性：

第一，Triton需要一个**模型仓库**，一个基于文件系统的模型存储库，Triton 将使其可用于推理。

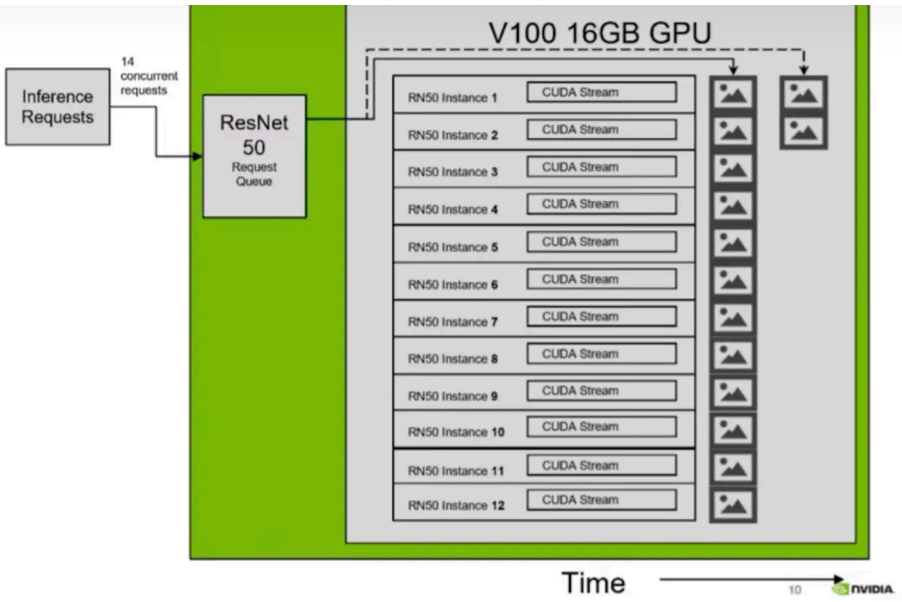
第二，Triton **支持 C API 后端**，允许 Triton 扩展新功能，例如：自定义预处理和后处理操作，甚至是新的深度学习框架。

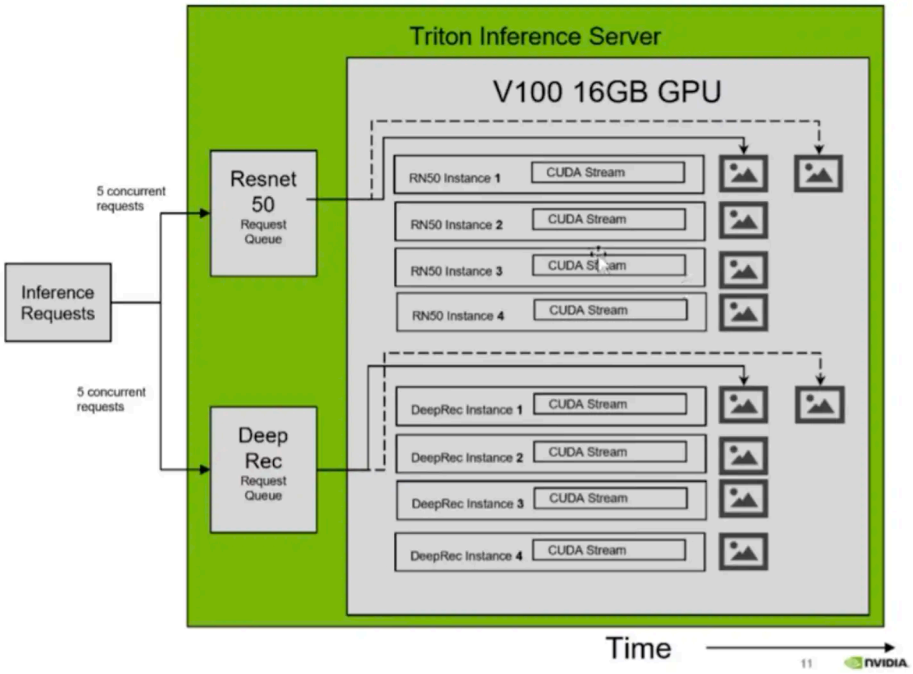
第三，Triton 提供的模型可以通过**专门的模型管理 API 进行查询和控制**，该 API 可通过 HTTP/REST 或 GRPC 协议或 C API 获得。

第四，Triton 服务中的**准备就绪（readiness）和存活（liveness）健康接口、利用率、吞吐量和延迟指标**简化了 Triton 与部署框架（如：Kubernetes）的集成。

另外，Triton 服务架构**允许多个模型或同一模型的多个实例在同一系统上并行执行**，具体如下图所示， 系统可能有零个、一个或多个 GPU 卡。

知乎 首发于 AI工程（MLOps）




**模型并行执行推理的逻辑**

下图显示了具有两个模型（模型 0 和模型 1）并行执行推理的流程。 假设 Triton 当前没有处理任何请求，当两个请求同时到达时Triton服务，每个模型一个请求，Triton 立即将它们都调度到 GPU 上，GPU 的硬件调度程序[+]开始并行处理这两个计算。

在系统的 CPU 上执行的模型由 Triton 进行类似地处理，除了 CPU 线程执行每个模型的调度由系统的操作系统处理之外。

知乎 首发于 AI工程（MLOps）

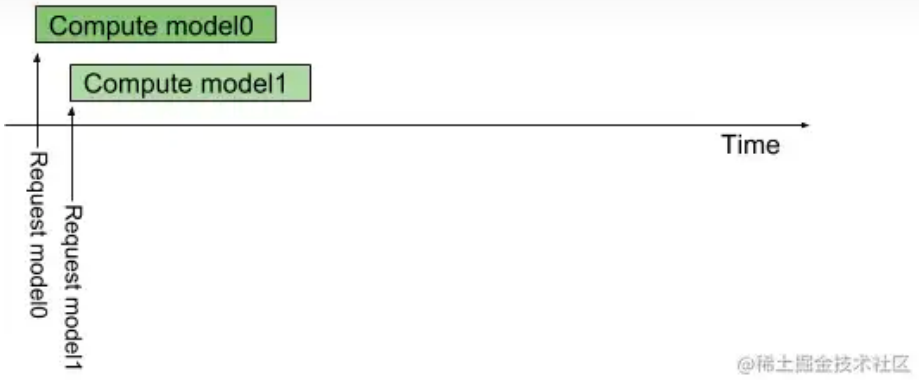



image.png

默认情况下，如果同一模型的多个请求同时到达，Triton 将通过在 GPU 上一次只调度一个来序列化它们的执行，如下图所示。

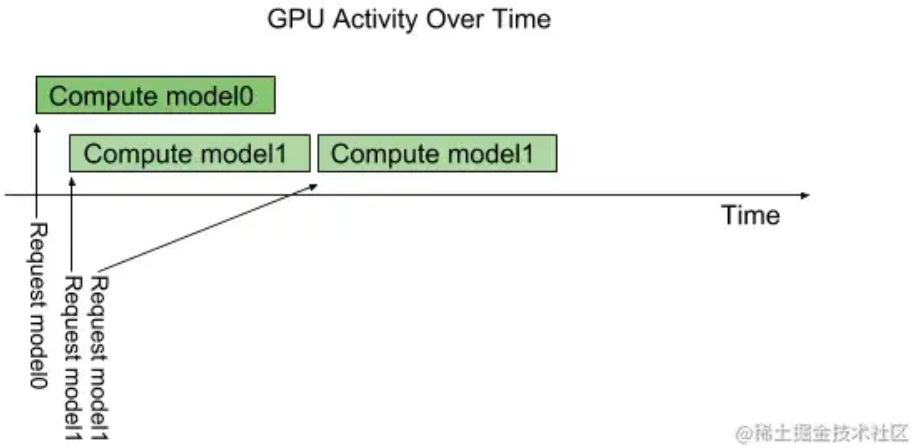



image.png

Triton 提供了一个名为 instance-group 的模型配置选项，它允许每个模型指定该模型的并行执行数。每个这样启用的并行执行*都称为一个实例。 默认情况下，Triton 会在系统中每个可用的 GPU 上为每个模型提供一个实例。 通过使用模型配置中的 instance_group 字段，可以更改模型的执行的实例数。 下图显示了当 model1 配置为允许三个实例的模型执行。 如下图所示，前三个 model1 服务的推理请求立即并行执行。 第四个 model1 服务的推理请求必须等到前三个执行中的一个执行完成才能开始。

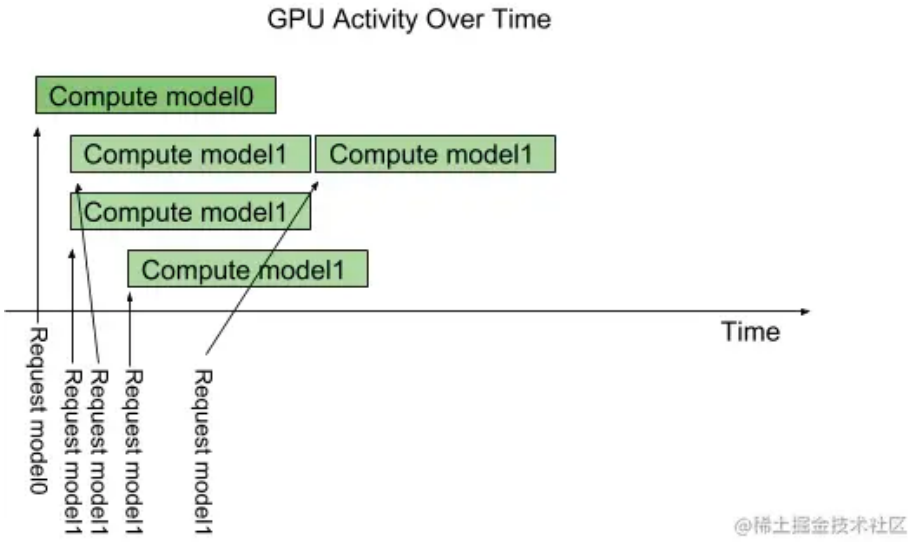



image.png

Triton 支持**多种调度和批处理（batching）算法**，可以为每个模型独立设置。

知乎 首发于 AI工程（MLOps）

Triton定义了三种模型类型：无状态（stateless）、有状态（stateful）和集成（ensemble）模型，同时，Triton 提供了调度器来支持这些模型类型。

### 模型类型

Triton中三种模型类型如下：

- 无状态模型（Stateless models）：简单来说就是应对不同推理请求没有相互依赖的情况。平常遇到的大部分模型都属于这一类模型，比如：文本分类、实体抽取、目标检测等。
- 有状态模型（Stateful Models）：当前的模型输出依赖上一刻的模型的状态（比如：中间状态或输出）。对于推理服务来说，就是不同推理请求之间有状态依赖，比如：顺序依赖。
- 集成模型（Ensemble model）：表示一个或多个模型组成的工作流(有向无环图)。最常见的使用场景就是：**数据预处理->模型推理->数据后处理**。通过集成模型可以避免传输中间tensor的开销，并且可以最小化请求次数。比如： `bert` 实现的文本分类任务，需要在前置处理中对输入文本做Tokenizer，Tokenizer输出结果作为模型属性输入。如下所示，preprocess表示前置处理模型；bert_onnx表示文本分类模型；ensemble_bert_classification表示集成模型。

```
├── bert_onnx
├── ensemble_bert_classification
└── preprocess
```

### 调度器

Triton提供了如下几种调度策略：

- Default Scheduler：默认调度策略，可以理解为推理框架内部的模型实例负载，将推理请求按比例分配到不同的模型实例执行推理。
- Dynamic Batcher：在应对高吞吐的场景，将一定时间内的请求聚合成一个批次（batch）。该调度器用于调度无状态的模型。
- Sequence Batcher：类似Dynamic batcher，Sequence batcher也是动态聚合推理请求。区别是Sequence batcher应用于有状态模型，并且一个序列的请求只能路由到同一个模型实例。
- Ensemble Scheduler：只能调度集成模型，不能用于其他类型的模型。

对于给定的模型，调度策略的选择和配置是通过模型的配置文件完成的，后续的文章会进行深入讲解。

## 总结

本文简要介绍了一个推理系统架构应该如何设计以及Triton的架构、Triton推理服务请求的执行流程、Triton中的重要组成组件，比如：

- Triton **支持 C API 后端**，允许 Triton 扩展新功能。
- Triton 提供的模型可以通过**专门的模型管理 API 进行查询和控制。**
- Triton 服务中的**准备就绪（readiness）和存活（liveness）健康端点、利用率、吞吐量和延迟指标**简化了 Triton 与部署框架（如：Kubernetes）的集成。
- Triton 服务架构**允许多个模型或同一模型的多个实例在同一系统上并行执行。** Triton 支持**多种调度和批处理（batching）算法**，可以为每个模型独立设置。

希望通过本文的讲解可以给你带来帮助。

**参考文档**:

- **Triton User Guide**
- **Triton 视频教程**

编辑于 2023-06-10 08:46 · IP 属地四川

大模型推理 大模型 人工智能

知乎 首发于 AI工程（MLOps）

理性发言，友善互动

发布

还没有评论，发表第一个评论吧

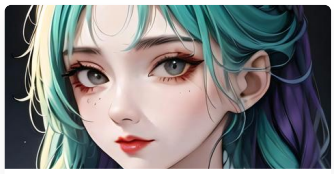
**文章被以下专栏收录**

AI工程（MLOps）
AI System/MLOps

**推荐阅读**

**15年软件架构师经验总结：在ML领域，初学者踩过的5个坑**

机器之心 发表于机器之心



**你知道的RAG框架，都在这了！**

大林 发表于吃透大模型...

**triton-inference-server的backend（一）——关于推...**

OLDPA... 发表于深度学习那...


**屠榜各大CV任务！微软提“统一模型”BEiT-3**

机器学习社... 发表于机器学