

## TensorRT-LLM保姆级教程（一）-快速入门



吃果冻不吐果冻皮

关注他

赞同 54



分享

54 人赞同了该文章

随着大模型的爆火，投入到生产环境的模型参数量规模也变得越来越来大（从数十亿参数到千亿参数规模），从而导致大模型的推理成本急剧增加。因此，市面上也出现了很多的推理框架，用于降低模型推理延迟以及提升模型吞吐量。

本系列将针对TensorRT-LLM推理进行讲解。本文为该系列第一篇，将简要概述TensorRT-LLM的基本特性。

另外，我撰写的大模型相关的博客及配套代码均整理放置在Github: [llm-action](#)，有需要的朋友自取。

### TensorRT-LLM 诞生的背景

第一、**大模型参数量大，推理成本高**。以10B参数规模的大模型为例，使用FP16数据类型进行部署至少需要20GB以上（模型权重+KV缓存\*等）。

第二、**纯TensorRT使用较复杂，ONNX存在内存限制**。深度学习模型通常使用各种框架（如PyTorch、TensorFlow、Keras等）进行训练和部署，而每个框架都有自己的模型表示和存储格式。因此，开发者通常使用 ONNX 解决深度学习模型在不同框架之间的互操作性问题。比如：TensorRT 就需要先将 PyTorch 模型转成 ONNX，然后再将 ONNX 转成 TensorRT。除此之外，一般还需要做[数据对齐\\*](#)，因此需要编写 [plugin\\*](#)，通过修改 ONNX 来适配 TensorRT plugin。另外，ONNX 使用Protobuf作为其模型文件的序列化格式。Protobuf是一种轻量级的、高效的数据交换格式，但它在序列化和反序列化大型数据时有一个默认的大小限制。在Protobuf中，默认的大小限制是2GB。这意味着单个序列化的消息不能超过2GB的大小。当你尝试加载或修改超过2GB的ONNX模型时，就会收到相关的限制提示。

第三、**纯FastTransformer使用门槛高**。FastTransformer 是用 C++ 实现的；同时，它的接口和文档相对较少，用户可能需要更深入地了解其底层实现和使用方式，这对于初学者来说可能会增加学习和使用的难度。并且 FastTransformer 的生态较小，可用的资源和支持较少，这也会增加使用者在理解和应用 FastTransformer 上的困难。因此，与 Python 应用程序的部署和集成相比，它可能涉及到更多的技术细节和挑战。这可能需要用户具备更多的[系统级编程\\*](#)知识和经验，以便将 FastTransformer 与其他系统或应用程序进行[无缝集成\\*](#)。

综上所述，TensorRT-LLM 诞生了。

### TensorRT-LLM 简介

TensorRT-LLM 为用户提供了易于使用的 Python API 来定义[大语言模型\\*](#) (LLM) 并构建 TensorRT 引擎，以便在 NVIDIA GPU 上高效地执行推理。TensorRT-LLM 还包含用于创建执行这些 TensorRT 引擎的 Python 和 C++ 运行时组件。此外，它还包括一个用于与 NVIDIA Triton 推理服务集成的后端；

同时，使用 TensorRT-LLM 构建的模型可以使用使用[张量\\*](#)并行和流水线并行在单 GPU 或者多机多 GPU 上执行。

TensorRT-LLM 的 Python API 的架构看起来与 PyTorch API 类似。它为用户提供了包含 einsum、softmax、matmul 或 view 等函数的 [functional](#) 模块。[layers](#) 模块捆绑了有用的构建块来组装 LLM；比如：Attention 块、MLP 或整个 Transformer 层。特定于模型的组件，例如：GPTAttention 或 BertAttention，可以在 [models](#) 模块中找到。

为了最大限度地提高性能并减少内存占用，TensorRT-LLM 允许使用不同的量化模式执行模型。TensorRT-LLM 支持 INT4 或 INT8 权重量化（也称为仅 INT4/INT8 权重量化）以及

收起

IT-LLM 诞生的背景

IT-LLM 简介

设备

性能

模型

速度

IT-LLM 的性能

### 支持的设备

TensorRT-LLM 在以下 GPU 上经过严格测试：

- H100
- L40S
- A100/A30
- V100 (试验阶段)

注意：如果是上面未列出 GPU，TensorRT-LLM 预计可在基于 Volta、Turing、Ampere、Hopper 和 Ada Lovelace 架构的 GPU 上工作。但是，可能存在某些限制。

### 关键特性

- 支持多头注意力(Multi-head Attention, MHA)
- 支持多查询注意力 (Multi-query Attention, MQA)
- 支持分组查询注意力(Group-query Attention, GQA)
- 支持飞行批处理 (In-flight Batching)
- Paged KV Cache for the Attention
- 支持 张量并行
- 支持 流水线并行
- 支持仅 INT4/INT8 权重量化 (W4A16 & W8A16)
- 支持 SmoothQuant 量化
- 支持 GPTQ 量化
- 支持 AWQ 量化
- 支持 FP8
- 支持贪心搜索 (Greedy-search)
- 支持波束搜索 (Beam-search)
- 支持[旋转位置编码](#)<sup>+</sup> (RoPE)

### 支持的模型

- Baichuan
- Bert
- Blip2
- BLOOM
- ChatGLM-6B
- ChatGLM2-6B
- Falcon
- GPT
- GPT-J
- GPT-Nemo
- GPT-NeoX
- LLaMA
- LLaMA-v2
- MPT
- OPT
- SantaCoder
- StarCoder

### 支持的精度

TensorRT-LLM 支持各种数值精度。但对其中一些数字精度的支持需要特定的GPU架构。

	FP32	FP16	BF16	FP8	INT8	INT4
--	------	------	------	-----	------	------

Turing (SM75)	Y	Y	N	N	Y	Y
Ampere (SM80, SM86)	Y	Y	Y	N	Y	Y
Ada-Lovelace (SM89)	Y	Y	Y	Y	Y	Y
Hopper (SM90)	Y	Y	Y	Y	Y	Y

对于目前发布的v0.5.0，并非所有模型都实现了对 FP8 和量化数据类型（INT8 或 INT4）的支持，具体如下所示。

Model	FP32	FP16	BF16	FP8	W8A8 SQ	W8A1 6	W4A1 6	W4A1 6 AWQ	W4A1 6 GPTQ
Baichuan	Y	Y	Y	.	.	Y	Y	.	.
BERT	Y	Y	Y	.	.	.	.	.	.
BLOOM	Y	Y	Y	.	Y	Y	Y	.	.
ChatGLM	Y	Y	Y	.	.	.	.	.	.
ChatGLM-v2	Y	Y	Y	.	.	.	.	.	.
Falcon	Y	Y	Y	.	.	.	.	.	.
GPT	Y	Y	Y	Y	Y	Y	Y	.	.
GPT-J	Y	Y	Y	Y	Y	Y	Y	Y	.
GPT-NeoX	Y	Y	Y	.	.	.	.	.	Y
LLaMA	Y	Y	Y	.	Y	Y	Y	Y	Y
LLaMA-v2	Y	Y	Y	Y	Y	Y	Y	Y	Y
OPT	Y	Y	Y	.	.	.	.	.	.
SantaCoder	Y	Y	Y	.	.	.	.	.	.
StarCoder	Y	Y	Y	.	.	.	.	.	.

TensorRT-LLM 的性能

注意：

下表中的数据作为参考进行提供，以帮助用户验证观察到的性能。这不是 TensorRT-LLM 提供的峰值性能。

不同模型基于 FP16 在 A100 GPU 上的吞吐量：

GPT-J 6B	64	1	128	128	3,679
GPT-J 6B	32	1	128	2048	1,558
GPT-J 6B	32	1	2048	128	526
GPT-J 6B	16	1	2048	2048	650
LLaMA 7B	64	1	128	128	3,486
LLaMA 7B	32	1	128	2048	1,459
LLaMA 7B	32	1	2048	128	529
LLaMA 7B	16	1	2048	2048	592
LLaMA 70B	64	4	128	128	1,237
LLaMA 70B	64	4	128	2048	1,181
LLaMA 70B	64	4	2048	128	272
LLaMA 70B	64	4	2048	2048	738
Falcon 180B	64	8	128	128	929
Falcon 180B	64	8	128	2048	923
Falcon 180B	64	8	2048	128	202

不同模型基于 FP16 在 A100 GPUs 上的首Token延迟:

针对批量大小为 1 时，第一个Token延迟的数据，代表终端用户感知在线流任务的延迟。

Model	Batch Size	TP (1)	Input Length	1st Token Latency (ms)
GPT-J 6B	1	1	128	12
GPT-J 6B	1	1	2048	129
LLaMA 7B	1	1	128	16
LLaMA 7B	1	1	2048	133
LLaMA 70B	1	4	128	47
LLaMA 70B	1	4	2048	377
Falcon 180B	1	8	128	61
Falcon 180B	1	8	2048	509

结语

本文简要概述了TensorRT-LLM诞生的原因以及基本特征。码字不易，如果觉得有帮助，欢迎点赞收藏加关注。


参考文档:

- [github.com/NVIDIA/Tenso...](https://github.com/NVIDIA/TensorRT-LLM)
- [github.com/NVIDIA/Tenso...](https://github.com/NVIDIA/TensorRT-LLM)
- [github.com/NVIDIA/Tenso...](https://github.com/NVIDIA/TensorRT-LLM)

发布于 2023-11-14 20:45 · IP 属地四川

2 条评论


默认 最新

 **五边形战士**

一点都不保姆🤔

2023-11-22 · 上海

回复 5

 **认输你就真了**

把readme都翻译成中文了还不保姆🤔

03-20 · 浙江

回复 1

文章被以下专栏收录

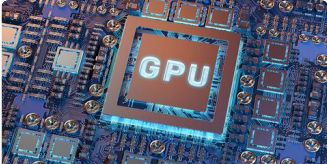
 动手学大模型

推荐阅读

[TensorRT-LLM]部署流程

TRT-LLM部署流程1. 编译trt-cpp文件  
cd TensorRT-LLM/cpp/build  
export  
TRT\_LIB\_DIR=/usr/local/tensorrt  
export  
TRT\_INCLUDE\_DIR=/usr/local/t...

平平无奇小熊猫



TensorRT-LLM 概念指南: Overview

李稀敏 发表于模型推理优...

长文详解--LLM高效预训练(一)

【推荐文章】MoE: MoE模型的前世今生 DeepSeek-V2和MLA 昆仑万维-SkyworkMoE 成本10w刀的JetMoE MoE的top-p routing 对MoE模型的一些观察 从dense到MoE -- sparse upcycling MoE...

Linsi... 发表于NLP-L...

关于LLM RAG的思考

RAG和Agent是两个被寄予厚望LLM落地路径。相比Agent，阶段RAG更具可行性，并已经通用的实现方案。目标/动机希望RAG能够带来如下几个方面升：解决幻觉、私域知识、失

sure