4.trt_llm

- Optimizing Inference on Large Language Models with NVIDIA TensorRT-LLM, Now Publicly Available | NVIDIA Technical Blog

参考资料：

- Welcome to TensorRT-LLM's documentation!

1.定义

`trt-llm`（全称 **TensorRT-LLM**）是由 **NVIDIA** 开发的一个**高性能、开源的大语言模型（LLM）推理框架**，专为在 NVIDIA GPU 上实现**极致推理速度与低延迟**而设计。它构建在 **TensorRT**（NVIDIA 的高性能推理 SDK）之上，并针对 LLM 的结构（如 Transformer）做了深度优化。

- **定位**：专用于 **大语言模型（如 LLaMA、Falcon、ChatGLM、Baichuan、Qwen 等）的推理加速框架**
- **开源**：GitHub 开源（Apache 2.0 许可），由 NVIDIA 维护
- **底层依赖**：
  - **TensorRT**：用于图优化、kernel 自动调优、量化等
  - **CUDA / cuBLAS / cuDNN / NCCL**：底层计算与通信
  - **自定义 CUDA kernels**：针对 attention、FFN、beam search 等模块高度优化

> 简单说：**trt-llm = TensorRT + LLM-specific optimizations + 多 GPU 支持 + 量化 + 高效解码**

2.核心特性与优化技术

1. **极致性能优化**

- **Kernel 融合（Kernel Fusion）**:将多个操作（如 MatMul + Bias + Activation）融合为单个 CUDA kernel，减少内存读写和 kernel launch 开销。
- **PagedAttention（受 vLLM 启发）**:支持高效 KV Cache 管理，减少内存碎片，提升显存利用率（尤其在动态 batch 场景）。
- **Continuous Batching（迭代式批处理）**:动态合并不同长度的请求，避免短请求等待长请求完成，提升吞吐。

2. **多 GPU 与并行支持**

- 支持 **Tensor Parallelism（TP）** 和 **Pipeline Parallelism（PP）**
- 自动处理跨 GPU 的 **All-Reduce / Send-Recv** 通信
- 与**Megatron 张量并行思想兼容**，但用 TensorRT 实现更高效

> **Note**
>
> **点对点（point-to-point）的数据传输** 就通过 `Send` 和 `Recv` 操作实现：

操作	作用
`Send`	当前 GPU 将张量发送给另一个 GPU
`Recv`	当前 GPU 从另一个 GPU 接收张量

3. **低精度与量化**

- 原生支持 **FP16、BF16**
- 支持 **INT8 / INT4 量化**（使用 **SmoothQuant** 或 **AWQ** 等方法）
- 量化模型可直接加载，自动调用 Tensor Core 加速

4. **高效解码策略**

- 内置 **C++ 实现的 Beam Search / Sampling / Top-p / Top-k**
- 支持 **并行解码（speculative decoding）**（需配合 draft model）
- 解码逻辑完全在 GPU 上运行，**避免 CPU-GPU 来回切换**（类似 FasterTransformer）

5. **易用性与生态集成**

- 提供 **Python API**（用于构建 engine）和 **C++ runtime**（用于部署）
- 支持从 Hugging Face 模型一键转换（通过 `trtllm` 命令行工具）
- 可与 **Triton Inference Server** 集成，用于生产级服务部署

3.典型使用流程

```
# 1. 安装 trt-llm（需 NVIDIA GPU + CUDA + TensorRT）
git clone https://github.com/NVIDIA/TensorRT-LLM
cd TensorRT-LLM
pip install -e .

# 2. 将 Hugging Face 模型转换为 TRT-LLM engine
python examples/llama/convert_checkpoint.py --model_dir ./hf_llama --output_dir
./trt_llm_llama

# 3. 构建优化后的推理 engine
trtllm-build --checkpoint_dir ./trt_llm_llama --output_dir ./engine --
max_batch_size 8 --max_input_len 512

# 4. 运行推理
python examples/run.py --engine_dir ./engine --input_text "What is AI?"
```

4.与 FasterTransformer（FT）的关系

特性	**FasterTransformer (FT)**	**TensorRT-LLM (trt-llm)**
开发方	NVIDIA（早期）	NVIDIA（当前主力）
状态	**已归档（archived）**，不再更新	**活跃开发中**，NVIDIA 官方推荐
技术栈	自研 C++/CUDA kernels	基于 **TensorRT** + 自定义插件
易用性	需手动编译、配置复杂	提供高层 API，支持 HF 模型一键转换

特性	FasterTransformer (FT)	TensorRT-LLM (trt-llm)
功能	支持 TP/PP、beam search	更丰富：量化、PagedAttention、continuous batching 等
未来	被 trt-llm 取代	**NVIDIA LLM 推理的官方标准方案**

> **结论**：如果你现在要部署 LLM 推理，**应优先选择 TensorRT-LLM**，而非 FasterTransformer。