大模型（LLMs）加速篇

来自： AiGC面试宝典

宁静致远

2023年09月29日 12:44

1. 当前优化模型最主要技术手段有哪些?

- 算法层面：蒸馏、量化
- 软件层面：计算图优化、模型编译
- 硬件层面：FP8（NVIDIA H系列GPU开始支持FP8，兼有fp16的稳定性和int8的速度）

2. 推理加速框架有哪一些? 都有什么特点?

- FasterTransformer：英伟达推出的FasterTransformer不修改模型架构而是在计算加速层面优化 Transformer 的 encoder 和 decoder 模块。具体包括如下：
    - 尽可能多地融合除了 GEMM 以外的操作
    - 支持 FP16、INT8、FP8
    - 移除 encoder 输入中无用的 padding 来减少计算开销
- TurboTransformers：腾讯推出的 TurboTransformers 由 computation runtime 及 serving framework 组成。加速推理框架适用于 CPU 和 GPU，最重要的是，它可以无需预处理便可处理变长的输入序列。具体包括如下：
    - 与 FasterTransformer 类似，它融合了除 GEMM 之外的操作以减少计算量
    - smart batching，对于一个 batch 内不同长度的序列，它也最小化了 zero-padding 开销
    - 对 LayerNorm 和 Softmax 进行批处理，使它们更适合并行计算
    - 引入了模型感知分配器，以确保在可变长度请求服务期间内存占用较小

3 vLLM 篇

3.1 vLLM 的 功能有哪些?

- Continuous batching：有iteration-level的调度机制，每次迭代batch大小都有所变化，因此vLLM在大量查询下仍可以很好的工作；
- PagedAttention：受操作系统中虚拟内存和分页的经典思想启发的注意力算法，这就是模型加速的秘诀

3.2 vLLM 的 优点有哪些?

1. **文本生成的速度**：实验多次，发现vLLM的推理速度是最快的；
2. **高吞吐量服务**：支持各种解码算法，比如parallel sampling, beam search等；
3. **与OpenAI API兼容**：如果使用OpenAI API，只需要替换端点的URL即可；

3.3 vLLM 的 缺点有哪些?

1. **添加自定义模型**：虽然可以合并自己的模型，但如果模型没有使用与vLLM中现有模型类似的架构，则过程会变得更加复杂。例如，增加Falcon的支持，这似乎很有挑战性；
2. **缺乏对适配器（LoRA、QLoRA等）的支持**：当针对特定任务进行微调时，开源LLM具有重要价值。然而，在当前的实现中，没有单独使用模型和适配器权重的选项，这限制了有效利用此类模型的灵活性。
3. **缺少权重量化**：有时，LLM可能不需要使用GPU内存，这对于减少GPU内存消耗至关重要。

3.4 vLLM 离线批量推理?

```
# pip install vllm
from vllm import LLM, SamplingParams



prompts = [
```

```
    "Funniest joke ever:",
    "The capital of France is",
    "The future of AI is",
]
sampling_params = SamplingParams(temperature=0.95, top_p=0.95, max_tokens=200)
llm = LLM(model="huggyllama/llama-13b")
outputs = llm.generate(prompts, sampling_params)

for output in outputs:
    prompt = output.prompt
    generated_text = output.outputs[0].text
    print(f"Prompt: {prompt!r}, Generated text: {generated_text!r}")
```

3.5 vLLM API Server?

```
# Start the server:
python -m vllm.entrypoints.api_server --env MODEL_NAME=huggyllama/llama-13b

# Query the model in shell:
curl http://localhost:8000/generate \
    -d '{
        "prompt": "Funniest joke ever:",
        "n": 1,
        "temperature": 0.95,
        "max_tokens": 200
    }'
```

4 Text generation inference 篇

4.1 介绍一下 Text generation inference?

Text generation inference是用于文本生成推断的Rust、Python和gRPC服务器，在HuggingFace中已有LLM 推理API使用。

4.2 Text generation inference 的 功能有哪些?

- 内置服务评估：可以监控服务器负载并深入了解其性能；
- 使用flash attention（和v2）和Paged attention优化transformer推理代码：并非所有模型都内置了对这些优化的支持，该技术可以对未使用该技术的模型可以进行优化；

4.3 Text generation inference 的 优点有哪些?

- 所有的依赖项都安装在Docker中：会得到一个现成的环境；
- 支持HuggingFace模型：轻松运行自己的模型或使用任何HuggingFace模型中心；
- 对模型推理的控制：该框架提供了一系列管理模型推理的选项，包括精度调整、量化、张量并行性、重复惩罚等；

4.4 Text generation inference 的 缺点有哪些?

- 缺乏对适配器的支持：需要注意的是，尽管可以使用适配器部署LLM（可以参考 https://www.youtube.com/watch?v=HI3cYN0c9ZU），但目前还没有官方支持或文档；
- 从源代码（Rust+CUDA内核）编译：对于不熟悉Rust的人，将客户化代码纳入库中变得很有挑战性；
- 文档不完整：所有信息都可以在项目的自述文件中找到。尽管它涵盖了基础知识，但必须在问题或源代码中搜索更多细节；

4.5 Text generation inference 的 使用docker运行web server?

```
mkdir data
```

```
docker run --gpus all --shm-size 1g -p 8080:80 \
-v data:/data ghcr.io/huggingface/text-generation-inference:0.9 \
  --model-id huggyllama/llama-13b \
  --num-shard 1
```