academicGPT 模型部署

一、 vLLM 模型部署

https://github.com/vllm-project/vllm/pull/289

https://github.com/vllm-project/vllm/pull/1804

https://github.com/Yard1/vllm/blob/multi_lora/examples/multilora_inference.py

https://github.com/vllm-project/vllm/pull/555

https://github.com/vllm-project/vllm/issues/2915

https://github.com/Yard1/vllm/tree/main/docs/source/serving

https://github.com/Yard1/vllm/blob/main/docs/source/serving/deploying_with_docker.rst

https://github.com/triton-inference-server/tutorials/blob/main/Quick_Deploy/vLLM/README.md#deploying-a-vllm-model-in-triton

需要安装 vllm，但是安装以后，July 那个 longqlora 的环境会被污染，用不了了。所以另外建一个 longqlora_vllm 专门用于部署，除了 july 那些安装环境的步骤，再额外装 vllm 就行。

二、问题和解决

（1）怎么让 vllm 支持把 llama-2 的 context size 拓展到 12288 呢

如果不用 vllm，那就是通过加载预训练模型（不是 adapter）的 congfig.json，然后增加 rope_scaling 这个参数：12888 // 4096。

以下代码出自：inference_chat_sft_example.py

```
# Set RoPE scaling factor
config = AutoConfig.from_pretrained(model_name_or_path)
orig_ctx_len = getattr(config, "max_position_embeddings", None)  # this value
if orig_ctx_len and context_size > orig_ctx_len:
    scaling_factor = float(math.ceil(context_size / orig_ctx_len))
    config.rope_scaling = {"type": "linear", "factor": scaling_factor}
```

然后在加载模型的时候传入这个参数。

```python
    # 加载base model
    model = AutoModelForCausalLM.from_pretrained(
        model_name_or_path,
        config=config,
        load_in_4bit=load_in_4bit,
        trust_remote_code=True,
        low_cpu_mem_usage=True,
        torch_dtype=torch.float16,
        device_map='auto',
        quantization_config=quantization_config
    )
```

所以其实还有一种方法就是直接修改 config.json，加上这个参数（看最后一个）：

```json
{
  "_name_or_path": "m42-health/med42-70b",
  "architectures": [
    "LlamaForCausalLM"
  ],
  "bos_token_id": 1,
  "eos_token_id": 2,
  "hidden_act": "silu",
  "hidden_size": 8192,
  "initializer_range": 0.02,
  "intermediate_size": 28672,
  "max_position_embeddings": 2048,
  "model_type": "llama",
  "num_attention_heads": 64,
  "num_hidden_layers": 80,
  "num_key_value_heads": 8,
  "pad_token_id": 0,
  "rms_norm_eps": 1e-05,
  "tie_word_embeddings": false,
  "torch_dtype": "float32",
  "transformers_version": "4.28.1",
  "use_cache": true,
  "vocab_size": 32000,
  "rope_scaling": {
    "factor": 2.0,
    "type": "dynamic"
  }
}
```

由于 vllm 封装比较好，所以我们选择直接修改 config.json 的方式。vllm 加载模型时，有个参数是 max_model_len，这个意思是模型设定的最大处理长度已经确定的情况下，根据自己应用场景，为了减少显存占用来设定的合理的最大长度，比如这次是 112288 是模型设定，那么如果 max_model_len 比 12288 大，就不行。或者设置 max_num_seqs = 12288 也一样的。

这次在 config.json 里面加了：{"rope_scaling": "factor":3.0, "type": "linear"}，因为 12288 / 4096 = 3.0。

（2）怎么离线预测

这里涉及的问题就是怎么把 基座模型和 LoRA 参数分别加载，而不是需要把这俩 merge 以后导出新模型。

这个可以通过 vllm 提供的 engine 来。

涉及的代码：

inference_chat_sft_vllm.py

component/multilora_inference.py

跑 inference_chat_sft_vllm.py 进行测试

性能测试

vllm 本身有个 metrics.py 会计算性能指标。

跑 10 条数据，单卡：

```
INFO 03-04 22:51:50 metrics.py:213] Avg prompt throughput: 1445.5 tokens/s, Avg generation throughput: 9.7 tokens/s, Running: 1 reqs, Swapped: 0 reqs, Pend
ing: 0 reqs, GPU KV cache usage: 27.9%, CPU KV cache usage: 0.0%
INFO 03-04 22:51:55 metrics.py:213] Avg prompt throughput: 0.0 tokens/s, Avg generation throughput: 27.7 tokens/s, Running: 1 reqs, Swapped: 0 reqs, Pendin
g: 0 reqs, GPU KV cache usage: 28.3%, CPU KV cache usage: 0.0%
INFO 03-04 22:52:00 metrics.py:213] Avg prompt throughput: 0.0 tokens/s, Avg generation throughput: 27.6 tokens/s, Running: 1 reqs, Swapped: 0 reqs, Pendin
g: 0 reqs, GPU KV cache usage: 28.8%, CPU KV cache usage: 0.0%
INFO 03-04 22:52:05 metrics.py:213] Avg prompt throughput: 1491.9 tokens/s, Avg generation throughput: 9.6 tokens/s, Running: 1 reqs, Swapped: 0 reqs, Pend
ing: 0 reqs, GPU KV cache usage: 25.2%, CPU KV cache usage: 0.0%
INFO 03-04 22:52:10 metrics.py:213] Avg prompt throughput: 0.0 tokens/s, Avg generation throughput: 29.1 tokens/s, Running: 1 reqs, Swapped: 0 reqs, Pendin
g: 0 reqs, GPU KV cache usage: 25.6%, CPU KV cache usage: 0.0%
INFO 03-04 22:52:19 metrics.py:213] Avg prompt throughput: 1035.0 tokens/s, Avg generation throughput: 15.8 tokens/s, Running: 1 reqs, Swapped: 0 reqs, Pen
ding: 0 reqs, GPU KV cache usage: 30.6%, CPU KV cache usage: 0.0%
INFO 03-04 22:52:24 metrics.py:213] Avg prompt throughput: 0.0 tokens/s, Avg generation throughput: 27.2 tokens/s, Running: 1 reqs, Swapped: 0 reqs, Pendin
g: 0 reqs, GPU KV cache usage: 31.0%, CPU KV cache usage: 0.0%
INFO 03-04 22:52:30 metrics.py:213] Avg prompt throughput: 1424.7 tokens/s, Avg generation throughput: 10.0 tokens/s, Running: 1 reqs, Swapped: 0 reqs, Pen
ding: 0 reqs, GPU KV cache usage: 29.0%, CPU KV cache usage: 0.0%
INFO 03-04 22:52:35 metrics.py:213] Avg prompt throughput: 0.0 tokens/s, Avg generation throughput: 27.4 tokens/s, Running: 1 reqs, Swapped: 0 reqs, Pendin
g: 0 reqs, GPU KV cache usage: 29.5%, CPU KV cache usage: 0.0%
INFO 03-04 22:52:40 metrics.py:213] Avg prompt throughput: 0.0 tokens/s, Avg generation throughput: 27.4 tokens/s, Running: 1 reqs, Swapped: 0 reqs, Pendin
g: 0 reqs, GPU KV cache usage: 29.9%, CPU KV cache usage: 0.0%
```

双卡下

```
INFO 03-04 22:56:24 metrics.py:213] Avg prompt throughput: 1492.7 tokens/s, Avg generation throughput: 20.8 tokens/s, Running: 1 reqs, Swapped: 0 reqs, Pending: 0 reqs, GPU KV cache usage: 9.2%, CPU KV cache usage: 0.0%
INFO 03-04 22:56:29 metrics.py:213] Avg prompt throughput: 0.0 tokens/s, Avg generation throughput: 43.1 tokens/s, Running: 1 reqs, Swapped: 0 reqs, Pending: 0 reqs, GPU KV cache usage: 9.4%, CPU KV cache usage: 0.0%
INFO 03-04 22:56:35 metrics.py:213] Avg prompt throughput: 1531.0 tokens/s, Avg generation throughput: 22.4 tokens/s, Running: 1 reqs, Swapped: 0 reqs, Pending: 0 reqs, GPU KV cache usage: 11.1%, CPU KV cache usage: 0.0%
INFO 03-04 22:56:40 metrics.py:213] Avg prompt throughput: 0.0 tokens/s, Avg generation throughput: 31.9 tokens/s, Running: 1 reqs, Swapped: 0 reqs, Pending: 0 reqs, GPU KV cache usage: 11.3%, CPU KV cache usage: 0.0%
INFO 03-04 22:56:45 metrics.py:213] Avg prompt throughput: 1718.7 tokens/s, Avg generation throughput: 16.5 tokens/s, Running: 1 reqs, Swapped: 0 reqs, Pending: 0 reqs, GPU KV cache usage: 10.6%, CPU KV cache usage: 0.0%
INFO 03-04 22:56:50 metrics.py:213] Avg prompt throughput: 0.0 tokens/s, Avg generation throughput: 38.1 tokens/s, Running: 1 reqs, Swapped: 0 reqs, Pending: 0 reqs, GPU KV cache usage: 10.8%, CPU KV cache usage: 0.0%
```

（3）怎么做出 servering 服务

通过调用 vllm 中的 openai.api_server 来做 servering

通过这个脚步来启动服务：[run_inference_sft_vllm.sh]

```
python -u -m vllm.entrypoints.openai.api_server \
              --host 0.0.0.0 \
              --served-model-name AcademicGPT \
              --model "/root/AcademicGPTLongQLoRA/model_download/Llama-2-7b-chat-hf" \
              --enable-lora \
              --max-lora-rank=64 \
              --max-num-seqs 12288 \
              --lora-modules academic="./output/LongQLoRA-Llama2-7b-8k/checkpoint-144"
'
```

服务接口

https://blog.csdn.net/spicy_chicken123/article/details/135813924

```
curl http://localhost:8000/v1/completions \
    -H "Content-Type: application/json" \
    -d '{
        "model": "AcademicGPT",
        "prompt": "",
        "max_tokens": 900,
        "temperature": 0.35
    }'
```

把包含 instruction 和 input 的 prompt ，填到上面的 json 里面，然后发送到 servring 去请求。

VLLM 服务接口输出的结果有点奇怪，暂时没找到原因。

VLLM

[Significance and novelty]

<Interpolation of RoPE embeddings> The paper proposes a novel method for interpolating RoPE embeddings to extend the context window of transformer-based language models, which is a significant contribution to the field of natural language processing.

<Comparison with existing methods> The paper provides a comprehensive comparison of the proposed method with existing methods, demonstrating its effectiveness and superiority over existing approaches.

[Potential reasons for acceptance]

<Technical soundness> The paper is technically sound and well-written, with thorough experiments and analysis.

<Clear presentation> The paper is well-organized and clearly presented, making it easy for readers to follow and understand the proposed method.

[Potential reasons for rejection]

<Lack of clarity in method description> There are concerns about the clarity of the method description, particularly in section 3.3, which may require further explanation and clarification.

<Limited experimental evaluation> The experimental evaluation is limited to a single dataset and a small number of models, which may not fully demonstrate the effectiveness of the proposed method.

[Suggestions for improvement]

<Clarify method description> The authors should consider providing clearer explanations and definitions for the proposed method, particularly in section 3.3, to ensure better understanding by readers.

<Expand experimental evaluation> The paper could benefit from a more comprehensive experimental evaluation, including a wider range of datasets and models, to demonstrate the effectiveness of the

proposed method.

HF

[Significance and novelty]

<Extension of RoPE interpolation methods> The paper proposes a novel method to extend the context window of transformer-based language models using RoPE interpolation, which is a significant and somewhat new contribution to the field.

[Potential reasons for acceptance]

<Clear and well-written paper> The paper is well-written and easy to follow, which is a potential reason for acceptance.

<Good experimental results> The experimental results show that the proposed method outperforms existing methods, which could be a potential reason for acceptance.

[Potential reasons for rejection]

<Lack of clarity in the methodology> The methodology section is not well-organized, and the authors need to provide more details on the interpolation method to ensure clarity and reproducibility.

<Insufficient experimental results> The experimental results are not comprehensive, and the paper lacks a comparison with other methods, which could be a potential reason for rejection.

[Suggestions for improvement]

<Clarify methodology> The authors should provide more details on the interpolation method, including the specific implementation and the reasoning behind the choices made, to ensure clarity and reproducibility.

<Comprehensive experimental results> The paper should include a comprehensive comparison with other methods, including the original PI and "NTK-aware" interpolation, to demonstrate the effectiveness of the proposed method.