PR #23419 完整报告

sgl-project/sglang

[model_runner] Label forward steps in profile traces with mode and token counts

合并时间：2026-04-22 17:31

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/23419

执行摘要

- 一句话：为模型前向步骤添加性能追踪标签
- 推荐动作：值得合并，提升可观测性且零开销。

功能与动机

PR body 指出之前查看 Chrome 追踪时需要猜测步骤边界，现在通过标签让每一步的模式和 token 数量一目了然。

实现拆解

1. 新增导入：在 model_runner.py 顶部添加 import contextlib，用于未启用 profiler 时的空上下文。
2. 创建标签生成函数：新增 _build_step_span_name(forward_batch: ForwardBatch) -> str，根据 forward_batch.forward_mode 的 is_idle/is_decode/is_extend 判断模式，并提取 batch_size、extend_num_tokens、extend_seq_lens 等字段，生成格式如 step[decode bs=N]、step[prefill bs=N toks=T]、step[mixed bs=N ext=T dec=D]、step[idle] 的字符串。
3. 包装前向计算：在 forward() 方法中，在调用 _forward_raw 之前，根据 torch.autograd._profiler_enabled() 动态选择 record_function 或 nullcontext，并将该上下文与原有的 with_forward_pass 上下文合并为 with (step_span_ctx, ...): 复合上下文。
4. 零性能开销：当 profiler 未启用时，nullcontext 不产生额外开销。

关键文件：

- python/sglang/srt/model_executor/model_runner.py（模块 模型执行器；类别 source；类型 core-logic；符号 _build_step_span_name）：核心变更文件，新增标签生成函数并修改 forward 方法。

关键符号：_build_step_span_name, ModelRunner.forward

关键源码片段

python/sglang/srt/model_executor/model_runner.py

核心变更文件，新增标签生成函数并修改 forward 方法。

```
# 根据 forward_batch 构建步骤标签字符串
```

```
def _build_step_span_name(forward_batch: ForwardBatch) -> str:
    """
    根据前向批次的模式生成 Chrome Trace 标签，格式示例：
      step[decode bs=4]
      step[prefill bs=2 toks=512]
      step[mixed bs=8 ext=256 dec=4]
      step[idle]
    """
    mode = forward_batch.forward_mode
    bs = forward_batch.batch_size
    if mode.is_idle():
        return "step[idle]"
    if mode.is_decode():
        return f"step[decode bs={bs}]"
    if mode.is_extend():
        ext_toks = forward_batch.extend_num_tokens or 0
        ext_seqs = (
            forward_batch.extend_seq_lens.shape[0]
            if forward_batch.extend_seq_lens is not None
            else bs
        )
        dec_seqs = bs - ext_seqs
        # 若同时有 decode 请求，标记为混合步骤
        if dec_seqs > 0:
            return f"step[mixed bs={bs} ext={ext_toks} dec={dec_seqs}]"
        return f"step[prefill bs={bs} toks={ext_toks}]"
    return f"step[{mode.name} bs={bs}]"


# 在 forward 方法中使用
class ModelRunner:
    def forward(self, forward_batch, ...):
        self.forward_pass_id += 1
        # 仅在 profiler 启用时创建 record_function，否则使用空上下文
        step_span_ctx = (
            torch.profiler.record_function(_build_step_span_name(forward_batch))
            if torch.autograd._profiler_enabled()
            else contextlib.nullcontext()
        )
        with (
            step_span_ctx,
            get_global_expert_distribution_recorder().with_forward_pass(...),
        ):
            output = self._forward_raw(...)
            # ... 后续逻辑
```

评论区精华

PR 仅有一条审核者 merrymercy 的 APPROVED 评论，无其他讨论。

- 暂无高价值评论线程

风险与影响

- 风险：风险极低。record_function 仅在 profiler 启用时激活，否则使用 nullcontext 零开销。新增的 _build_step_span_name 函数逻辑简单，只读取 forward_batch 的字段，不修改状态。
- 影响：对用户无影响（仅改变 profiler 输出的标签）。对开发者和运维人员调试性能更友好，可快速识别追踪中每一步的模式和 token 数量。影响范围仅限于 model_runner.py 一个文件。
- 风险标记：暂无

关联脉络

- 暂无明显关联 PR