PR #35677 完整报告

sgl-project/sglang

fix(cpu): skip GPU JIT MoE top-k on CPU

合并时间：2026-08-28 17:31

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/35677

执行摘要

- 一句话：CPU 跳过 GPU JIT MoE top-k，修复 CPU MoE 推理崩溃
- 推荐动作：值得快速阅读：这是一个典型「后端误入 GPU-only 路径」的最小修复范本，改动仅 6 行，评审讨论体现了支持边界的取舍。建议 CPU 维护者关注 review 中关于非 AMX CPU 支持范围的决策，并后续补充自动化测试与非 AMX CPU 性能评估。

功能与动机

关联 Issue #35779 报告：MiniMax-M2 在 CPU 上启动成功，但处理推理请求时在 MoE 路由阶段抛错。PR body 指出根因是 PR #34926 删除了 use_jit_fused_gate 标志，强制 sigmoid MoE 路由进入 biased_topk_jit_kernel_impl，而 JIT 路由依赖 GPU-only 的 topk_sigmoid / topk_softmax 导入，CPU 设备在请求处理（而非启动）时失败。

实现拆解

1. 定位回归入口：问题集中在 python/sglang/srt/layers/moe/topk.py 的 select_experts 函数，custom_routing_function is None 分支内。PR #34926 移除 use_jit_fused_gate 后，scoring_func 为 sigmoid 或 sqrtsoftplus 的 MoE 模型会无条件进入 biased_topk_jit_kernel_impl，该实现依赖仅在 GPU 后端导入的 topk_sigmoid / topk_softmax 符号。
2. 加入设备守卫：在 sigmoid 判断前新增局部变量 _can_use_jit_kernel = not _is_cpu，并把原 if 条件改写为 _can_use_jit_kernel and (scoring_func == "sqrtsoftplus" or scoring_func == "sigmoid")，语义为非 CPU 设备才允许进入 GPU JIT top-k 路径。
3. CPU 回落路径：守卫令 CPU 跳过 _biased_topk 调用，继续评估后续分支；flashinfer_trtllm_routed 分支仅对 softmax 生效，因此 sigmoid 模型最终落入 else 分支的 torch 原生 fused topk 路径（_fused_topk_pack 关闭），不再触碰任何 GPU-only 符号。
4. 评审驱动的范围收敛：最初改动还包含 biased_grouped_topk_cpu 附近为非 AMX CPU 强制 fused_topk = fused_topk_torch_native 的额外分支；评审人 mingfeima 认为项目不承诺完整支持非 AMX CPU 且 if-else-if 不直观，作者随后回滚该部分（对应提交 revert），仅保留最小守卫。
5. 配套与验证：未新增单元测试（checklist 未勾选）；社区成员 ZailiWang 使用 --device cpu --tp 6 --disable-overlap-schedule --dtype bfloat16 --mem-fraction-static 0.85 启动

命令手工验证了 MiniMax-M2 与 MiniMax-M2.5 的补全请求正常。

关键文件：

- python/sglang/srt/layers/moe/topk.py（模块 MoE 路由；类别 source；类型 core-logic；符号 select_experts）：唯一变更文件，核心修复位于 select_experts 的 MoE 路由分支：新增 _can_use_jit_kernel = not _is_cpu 守卫，阻止 CPU 进入依赖 GPU-only 导入的 biased_topk_jit_kernel_impl，使 CPU 回落到 torch 原生 topk 路径。

关键符号：select_experts

关键源码片段

python/sglang/srt/layers/moe/topk.py

唯一变更文件，核心修复位于 select_experts 的 MoE 路由分支：新增 _can_use_jit_kernel = not _is_cpu 守卫，阻止 CPU 进入依赖 GPU-only 导入的 biased_topk_jit_kernel_impl，使 CPU 回落到 torch 原生 topk 路径。

```
# select_experts 中 custom_routing_function is None 的分支入口。
# 此前 PR #34926 移除了 use_jit_fused_gate 开关，导致 scoring_func 为
# sigmoid / sqrtsoftplus 时无条件进入 biased_topk_jit_kernel_impl，
# 而该 JIT 实现依赖仅在 GPU 后端导入的 topk_sigmoid / topk_softmax，
# CPU 上会在请求处理阶段直接崩溃。
elif custom_routing_function is None:
    if scoring_func not in ("sqrtsoftplus", "sigmoid"):
        assert not apply_routed_scaling_factor_on_output, "Not implemented"

    # JIT 路由依赖 GPU-only 的 topk_sigmoid / topk_softmax 导入，
    # 因此 CPU 上必须跳过该分支，落到下方 torch 原生路径。
    _can_use_jit_kernel = not _is_cpu

    if _can_use_jit_kernel and (
        scoring_func == "sqrtsoftplus" or scoring_func == "sigmoid"
    ):
        # XPU 走专用实现，其余设备走 JIT kernel 实现。
        _biased_topk = biased_topk_xpu if _is_xpu else biased_topk_jit_kernel_impl
        topk_weights, topk_ids = _biased_topk(
            hidden_states=hidden_states,
            gating_output=router_logits,
            correction_bias=correction_bias,
            topk=num_routed_topk if _use_aiter else top_k,
            renormalize=renormalize,
            scoring_func=scoring_func,
            num_fused_shared_experts=num_fused_shared_experts,
            routed_scaling_factor=routed_scaling_factor,
            num_token_non_padded=num_token_non_padded,
            expert_location_dispatch_info=expert_location_dispatch_info,
            apply_routed_scaling_factor_on_output=apply_routed_scaling_factor_on_output,
        )
    elif (
```

```
        get_moe_runner_backend().is_flashinfer_trtllm_routed()
        and scoring_func == "softmax"
        and correction_bias is None
    ):
        # flashinfer_trtllm_routed 使用原始 logits 做 topk。
        topk_weights, topk_ids = fused_topk_softmax_torch_raw_logits(
            hidden_states=hidden_states,
            gating_output=router_logits,
            topk=num_routed_topk if _use_aiter else top_k,
            renormalize=renormalize,
        )
    else:
        # CPU 上的 sigmoid / sqrtsoftplus 会走到这里，随后进入 torch
        # 原生 fused topk / 普通 topk 路径，避免引用任何 GPU-only 符号。
        _fused_topk_pack = False
```

评论区精华

核心讨论围绕「非 AMX CPU 支持边界」展开。评审人 mingfeima 对最初的 extra 分支提出：项目不声明完整支持非 AMX CPU，且嵌套 if-else-if 结构不直观，建议改为 if _is_cpu: if amx: xxx else yyy else zzz 形式。作者 xinguozhu-2026 认同并选择直接回滚该额外分支，仅保留 _can_use_jit_kernel = not _is_cpu 的最小守卫，最终 mingfeima 转为 APPROVED。

- 非 AMX CPU 支持边界与 if-else 结构 (design): 作者 xinguozhu-2026 认同并回滚额外分支，仅保留 _can_use_jit_kernel = not _is_cpu 的最小守卫，避免扩大非 AMX CPU 支持面；mingfeima 最终 APPROVED。

风险与影响

- 风险：
  1. 精度风险：CPU 上 sigmoid 语义从 GPU JIT kernel 切换为 torch 原生 fused topk，renormalize 等数值行为未做对比测试，存在轻微差异可能。
  2. 性能风险：非 AMX CPU 只能走 torch 原生路径，性能未优化，且项目明确不承诺完整支持非 AMX CPU。
  3. 测试缺口：无自动化测试覆盖，仅靠社区手工验证 M2 / M2.5 两个模型，其他 sigmoid 路由 MoE 模型（如 M2.7 相关）的回归风险未被自动化兜底。
  4. 范围控制：改动只影响 _is_cpu 为真的分支，GPU / XPU 路径完全不变，回归面很小。 - 影响：对用户：CPU 设备上 MiniMax-M2 / M2.5 等 sigmoid 路由 MoE 模型从「启动成功但请求必崩」变为可正常推理，是功能性修复。对系统：仅 CPU 后端行为变化，GPU / XPU / NPU 等路径零影响。对团队：明确了非 AMX CPU 的支持边界，为后续 CPU MoE 专项优化留下清晰注释与决策记录。 - 风险标记：CPU 路径行为变更，无自动化测试，非 AMX CPU 性能与精度未验证

关联脉络

- PR #34926 Remove use_jit_fused_gate flag: 本 PR body 明确指出该 PR 移除了 use_jit_fused_gate 开关并强制 sigmoid 路由走 GPU JIT 路径，是本 CPU 回归的根因。
- PR #36529 [Fix][XPU/ROCm/NPU] Defer sgl_kernel.quantization import in expert_pack: 同属「后端加载 GPU-only 符号导致崩溃」的跨后端修复，模式一致，体现了 SGLang 多后端下条件导入的常见回归问题。