PR #25001 完整报告

sgl-project/sglang

[LoRA] MLA attention LoRA: q_b_proj / kv_b_proj support

合并时间：2026-05-14 06:15

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/25001

执行摘要

- 一句话：支持 MLA 注意力 q_b_proj 和 kv_b_proj LoRA 适配器
- 推荐动作：值得精读。特别是 SGMM Triton 内核的设计——将 B@A 分解为两步，避免物化大矩阵，同时兼容两种 LoRA 后端（Triton/csgmv）的 segment-routing 方案。此外，对 fused_qkv_a_proj_with_mqa 快速路径的 LoRA 保护也是一个典型模式。建议未来若添加测试覆盖率，应优先覆盖混合秩、零 slot、和 csgmv 后端场景。

功能与动机

DeepSeek-styleMLA注意力有四个投影矩阵——q_a_proj、kv_a_proj_with_mqa、q_b_proj、kv_b_proj——但 main 分支只支持前两个（通过 fused_qkv_a_proj_with_mqa）。使用 Kimi-K2.5 等包含 q_b_proj/kv_b_proj 的 LoRA 适配器时，要么在加载时验证失败，要么被静默丢弃。kv_b_proj 尤为困难，因为在 absorbed-MLA 路径中，运行时从不调用 kv_b_proj.forward()，K/V 贡献被折叠进 w_kc/w_vc 的 BMM 中，标准 LoRA 包装器无效。naive 的逐 slot 物化 B@A 方法会引入约每层每 slot 268M FMAs 的巨大开销。PR 通过 SGMM 分解解决了此问题。

实现拆解

1. 目标模块注册：在 SUPPORTED_LORA_TARGET_MODULES 中添加 q_a_proj、kv_a_proj_with_mqa、q_b_proj、kv_b_proj，并在 get_hidden_dim 中定义它们的输入 / 输出维度。get_normalized_target_modules 中前两个折叠到 fused_qkv_a_proj_with_mqa，后两个保持不变。
2. fused_qkv_a_proj_with_mqa 的 LoRA 保护：在 prepare_qkv_latent 中检查 fused 模块是否设置了 LoRA（set_lora 属性），若已激活则跳过 dsv3_fused_a_gemm 快速路径，改用标准 forward 路径使 LoRA 生效。
3. 核心修正逻辑：新增 lora/deepseek_mla_correction.py，包含 is_kv_b_lora_active（快速非 LoRA 门控）和 apply_q_correction/apply_v_correction。它们通过调用 SGMM 内核在预吸收 BMM 结果上叠加 LoRA delta，全程无 Python 循环。
4. SGMM Triton 内核：新增 lora/triton_ops/kv_b_lora_absorbed.py（约 850 行），实现了四个内核：step_a_q_fwd、step_b_q_fwd、step_a_v_fwd、step_b_v_fwd。它们沿 LoRA-A/B 边界分解数学，每个内核使用三维网格（输出 tile、head_id、segment_id），支持 segment-indptr 路由和混合秩。

5. 后端兼容：修正逻辑自动适配 Triton 后端（单 segment 每请求）和 csgmv 后端（分组 permutation 路由）。内核读取 batch_info.permutation 进行行路由。
6. 调用点注入：在 forward_mla.py 的 forward_absorb_prepare 和 forward_absorb_core 中，通过 is_kv_b_lora_active 门控，在 BMM 后调用对应的修正函数。非 LoRA 路径仅增加一次 getattr 开销。
7. 配套修改：更新 common.py 的 CLI 参数列表、lora/utils.py 的已知模块集合、__init__.py 导出新内核。注意：本次变更未包含测试文件。

关键文件：

- python/sglang/srt/lora/deepseek_mla_correction.py（模块 LoRA 修正；类别 source；类型 core-logic；符号 is_kv_b_lora_active, _get_state, apply_q_correction, apply_v_correction）：核心修正模块：提供 is_kv_b_lora_active 门控和 apply_q_correction/apply_v_correction 函数，封装 SGMM 内核调用入口，是设计精髓所在。
- python/sglang/srt/lora/triton_ops/kv_b_lora_absorbed.py（模块 Triton 内核；类别 infra；类型 infrastructure；符号 _num_segments, _max_segment_len, _segment_grid_size, _step_a_q_kernel）：SGMM Triton 内核实现，约 850 行，性能关键。定义了 step_a_q_fwd、step_b_q_fwd、step_a_v_fwd、step_b_v_fwd 四个内核，处理混合秩和多后端路由。
- python/sglang/srt/models/deepseek_common/attention_forward_methods/forward_mla.py（模块 前向注入；类别 source；类型 data-contract）：注入 LoRA 修正调用点，确保 BMM 后立即调用修正函数，并添加 is_kv_b_lora_active 门控避免非 LoRA 开销。
- python/sglang/srt/models/deepseek_v2.py（模块 Attention 模型；类别 source；类型 data-contract；符号 prepare_qkv_latent）：对 fused_qkv_a_proj_with_mqa 快速路径添加 LoRA 活跃度检查，阻止快速路径绕过 LoRA。
- python/sglang/srt/lora/utils.py（模块 LoRA 工具；类别 source；类型 core-logic）：扩展 get_hidden_dim 以支持 q_b_proj/kv_b_proj 的维度，并将它们添加到 _KNOWN_LORA_TARGET_MODULES。
- python/sglang/srt/utils/common.py（模块 CLI 参数；类别 source；类型 core-logic）：在 SUPPORTED_LORA_TARGET_MODULES 中添加四个新模块名，使 CLI 参数可用。
- python/sglang/srt/lora/triton_ops/__init__.py（模块 导出入口；类别 infra；类型 infrastructure）：导出新内核以便上层导入。

关键符号：is_kv_b_lora_active, _get_state, apply_q_correction, apply_v_correction, step_a_q_fwd, step_b_q_fwd, step_a_v_fwd, step_b_v_fwd, get_hidden_dim, prepare_qkv_latent, forward_absorb_prepare, forward_absorb_core

关键源码片段

python/sglang/srt/lora/triton_ops/kv_b_lora_absorbed.py

SGMM Triton 内核实现，约 850 行，性能关键。定义了 step_a_q_fwd、step_b_q_fwd、step_a_v_fwd、step_b_v_fwd 四个内核，处理混合秩和多后端路由。

```
"""Triton kernels for absorbed-MLA kv_b_proj LoRA correction.
```

```
沿 LoRA-A/B 边界分解数学，避免物化 B@A 矩阵。
每个内核使用三维网格 (output_tile, head_id, segment_id)，
透过 seg_indptr 和 weight_indices 实现 segment 路由。
同时支持 Triton 后端（连续 segment）和 csgmv 后端（permutation 路由）。
"""

from __future__ import annotations
import torch
import triton
import triton.language as tl

# 四内核的 block 大小针对自然形状选取
# step_a_q: 收缩 qk_nope (~128) -> rank (~16-32)
# step_b_q: 收缩 rank (~16-32) -> kv_lora_rank (~512)
# ...

@triton.jit(do_not_specialize=["num_segments"])
def _step_a_q_kernel(
    # 参数省略，展示核心 grid 和 routing
    ...
):
    """SGMM: (S,H,qk_nope) @ B_kc (qk_nope, rank) -> (S,H,rank)"""
    pid_s = tl.program_id(0) // num_pid_n
    pid_n = tl.program_id(0) % num_pid_n
    head_id = tl.program_id(1)
    segment_id = tl.program_id(2)

    # 透过 seg_indptr 获取 segment 的 token 范围
    seg_start = tl.load(seg_indptr + segment_id)
    seg_end = tl.load(seg_indptr + segment_id + 1)
    token_count = seg_end - seg_start

    # 加载该 segment 对应的 LoRA 权重索引和缩放
    w_index = tl.load(weight_indices + segment_id)
    scale = tl.load(scalings + w_index)
    cur_rank = tl.load(lora_ranks + w_index)
    K_eff = tl.minimum(K, cur_rank) # 混合秩：有效 K 取实际秩
    ...

    # 核心 tile 循环，使用 tl.dot 计算
    ...

def step_a_q_fwd(inp, weight, batch_info, full_K_per_head):
    """包装函数：计算 grid size、调用 _step_a_q_kernel。"""
    ...
```

评论区精华

Review 中 Fridge003 提出了几点重要异议：

- 文件位置争议（设计）：_get_kv_b_lora_state 和 _apply_kv_b_lora_q_correction 最初放在 deepseek_v2.py，Fridge003 指出应放 lora 文件夹。最终作者创建了独立的 deepseek_mla_correction.py 模块。
- AMD ROCm 文件保护：作者最初修改了 forward_mla_fused_rope_rocm.py，Fridge003 警告不要改动此文件，它是专为 AMD 设备的。作者立即回退。
- 性能开销关注（性能）：Fridge003 要求将新增代码用 is_kv_b_lora_active 保护，避免非 LoRA 场景下引入任何额外 GPU 操作。作者照做。
- 修正逻辑重复问题（正确性）：Fridge003 询问为何在 quant 路径中再次调用修正，作者解释是尝试对齐 AMD 的 quant 路径，后续已移除多余的第二次调用。

最终经过迭代，Fridge003 审批通过。

- 修正代码应放在 lora 文件夹而非 deepseek_v2.py (design): 作者创建独立文件 deepseek_mla_correction.py 进行封装。
- AMD ROCm 文件保护 (other): 作者回退了对该文件的修改。
- 非 LoRA 场景性能开销 (performance): 作者添加 is_kv_b_lora_active 条件保护，仅在 LoRA 活跃时执行修正。
- quant 路径中重复修正 (correctness): 作者解释是试图对齐 AMD quant 路径，后移除第二次调用。

风险与影响

- 风险：
  - 新 Triton 内核风险：四个 SGMM 内核全新增，未在广泛硬件上验证（主要针对 CUDA），可能存在数值稳定性或性能退化，尤其在空 / 边缘段或零秩时。kernel 中使用了 do_not_specialize=["num_segments"]，但其余参数可被 Triton 重 JIT。
  - 非 LoRA 路径回归：forward_mla.py 中的张量操作被移动（如 transpose/flatten 提前），依赖于上游结果正确性。已对 quant/ 非 quant 路径做了条件分支，但可能欠缺某些硬件组合（AMD、Intel、NPU）的测试。
  - CUDA Graph 兼容性：commit 消息提到“Stabilize kv_b LoRA CUDA graph grid”，说明早期版本存在 CUDA graph 恢复问题。当前已修复，但 graph 与动态 segment 数量交互仍可能异常。
  - 缺少测试覆盖：无新增测试文件。虽然 PR body 提到现有 CI 应该覆盖，但新增内核缺乏单元测试，可能漏掉边界条件（如 rank=0、混合秩、全 adapter 组等）。
- 影响：
  - 用户影响：对使用 DeepSeek-style MLA 模型的 LoRA 微调用户是直接利好，特别是 Kimi-K2.5 等包含 q_b_proj/kv_b_proj 的适配器。对不使用 LoRA 的用户，仅在 attention 前向多一个 getattr 检查（is_kv_b_lora_active），影响可忽略。
  - 系统影响：增加了约 1.8k 行新代码（含 Triton 内核和修正模块）。Triton 内核编译可能略微增加首次启动时间。但运行时仅在 LoRA 适配器活跃时才调用内核，不影响典型未使用 LoRA 的部署。

- 团队影响：模块架构清晰，修正逻辑从 attention 类中解耦到 lora 文件夹，便于后续扩展其他投影的 LoRA 支持。
- 风险标记：新 Triton 内核，缺少测试覆盖，CUDA Graph 兼容性，影响深层网络所有路径

关联脉络

- 暂无明显关联 PR