PR #37912 完整报告

vllm-project/vllm

[Bugfix] Fuse Qwen3.5 in_qkvz_proj forwarding with LoRA enabled

合并时间：2026-05-10 18:59

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/37912

执行摘要

- 一句话：统一 Qwen3.5 LoRA 前向路径
- 推荐动作：建议重点关注 LoRA 场景下的回归测试，特别是 TP>1 的配置。设计决策 ' 将复杂性转移到 LoRA 层 ' 值得在其他类似模型中复用。

功能与动机

PR body: 'There're 2 forwarding code path for Qwen3.5 after #36976. This PR unifies them by adapting the LoRA layer implementation.' 目的是消除因 LoRA 启用与否导致的分叉逻辑，降低维护复杂度。

实现拆解

1. 移除模型中的 LoRA 条件分支：在 Qwen3_5DecoderLayer 和 GatedDeltaNetAttention 中去掉 create_in_proj_qkvz 参数，始终创建融合的 in_proj_qkvz 投影。Qwen3_5Model 移除 enable_lora 标志。同时修改 load_weights，统一从 in_proj_qkvz 张量中加载并映射到 in_proj_qkv（Q、K、V）和 in_proj_z。
2. 引入 expand_packed_lora 方法：在 MergedColumnParallelLinearWithLoRA 类中新增该方法，当 set_lora 接收的 lora_b 数量与 n_slices 不匹配时（例如将 in_proj_qkv（覆盖 3 个 slice）和 in_proj_z（覆盖 1 个 slice）作为 2 个适配器传递给 4-slice 层），自动根据 output_sizes 拆分并复制 lora_a，使每 slice 都有对应的适配器。
3. 更新 create_dummy_lora 的 packed_modules_mapping 处理：在 ModelManager.create_dummy_lora 中，通过 getattr(module, 'n_slices', ...) 获取真实的 slice 数，当与 packed_modules_mapping 中的替代名数量不一致时，使用 slice_i 作为命名，确保 dummy LoRA 权重创建正确。
4. 移除 Qwen3_5ForCausalLMBase 和 Qwen3_5ForConditionalGeneration 中的 packed_modules_mapping 运行时修改：不再根据 LoRA 启用与否动态调整映射，保持映射一致。

关键文件：

- vllm/model_executor/models/qwen3_5.py（模块 模型定义；类别 source；类型 data-contract；符号 update_packed_mapping）：核心模型文件，移除 LoRA 条件分支，统一 load_weights 映射路径。

- vllm/lora/layers/column_parallel_linear.py（模块 LoRA 层；类别 source；类型 core-logic；符号 expand_packed_lora）：新增 expand_packed_lora 方法，实现打包适配器展开的核心逻辑。
- vllm/model_executor/layers/mamba/gdn_linear_attn.py（模块 注意力层；类别 source；类型 data-contract）：移除 create_in_proj_qkvz 参数，始终使用融合投影，简化前向路径。
- vllm/lora/model_manager.py（模块 LoRA 管理；类别 source；类型 data-contract）：适配新架构，通过 n_slices 属性修正 create_dummy_lora 的映射逻辑。

关键符号：expand_packed_lora

关键源码片段

vllm/model_executor/models/qwen3_5.py

核心模型文件，移除 LoRA 条件分支，统一 load_weights 映射路径。

```
def load_weights(self, weights: Iterable[tuple[str, torch.Tensor]]) -> set[str]:
    # Unified stacked params mapping - no dynamic branching based on LoRA
    stacked_params_mapping = [
        # GDN: in_proj_qkvz -> in_proj_qkv (Q,K,V) and in_proj_z
        ('in_proj_qkvz', 'in_proj_qkv', (0, 1, 2)),
        ('in_proj_qkvz', 'in_proj_z', 3),
        # self attention
        ('qkv_proj', 'q_proj', 'q'),
        ('qkv_proj', 'k_proj', 'k'),
        ('qkv_proj', 'v_proj', 'v'),
        ('in_proj_ba', 'in_proj_b', 0),
        ('in_proj_ba', 'in_proj_a', 1),
    ]
    # Previously, when self.enable_lora was True, extra mappings for
    # in_proj_qkv and in_proj_z were added; now they are fixed.
    params_dict = dict(self.named_parameters())
    loaded_params: set[str] = set()
    for name, loaded_weight in weights:
        # ... expert weights processing omitted for brevity ...
        param = params_dict[name]
        weight_loader = param.weight_loader
        # Always pass shard_id (previously omitted for in_proj_z in LoRA case)
        weight_loader(param, loaded_weight, shard_id)
        loaded_params.add(name)
    return loaded_params
```

vllm/lora/layers/column_parallel_linear.py

新增 expand_packed_lora 方法，实现打包适配器展开的核心逻辑。

```
def expand_packed_lora(
    self,
    lora_a: list[torch.Tensor],
    lora_b: list[torch.Tensor],
```

```
) -> tuple[list[torch.Tensor], list[torch.Tensor]]:
    '''
    Expand packed adapter groups when they don't match n_slices.
    E.g. in_proj_qkv (covers Q+K+V) + in_proj_z
    '''
    expanded_a: list[torch.Tensor] = []
    expanded_b: list[torch.Tensor] = []
    start_idx = 0
    for a_i, b_i in zip(lora_a, lora_b):
        # Determine which output slices this b_i covers.
        b_rows, cu_rows, covered = b_i.shape[0], 0, 0
        for i in range(start_idx, self.n_slices):
            cu_rows += self.output_sizes[i]
            if cu_rows == b_rows:
                covered = i - start_idx + 1
                break
        else:
            raise ValueError(
                f'Cannot determine how to split lora_b with {b_rows} rows '
                f'into {self.n_slices} slices with output sizes '
                f'{self.output_sizes} starting from index {start_idx}.'
            )
        # Split b_i into per-slice tensors and replicate a_i for each.
        start = 0
        for j in range(covered):
            size = self.output_sizes[start_idx + j]
            expanded_b.append(b_i[start : start + size, :])
            expanded_a.append(a_i)
            start += size
        start_idx += covered
    return expanded_a, expanded_b
```

评论区精华

Reviewer gemini-code-assist[bot] 指出 model_manager.py 中的 'HACK' 注释值得担忧，建议替换为详细解释或更健壮的方案。作者未在评论中回应，但该 PR 已被批准合并。未解决的疑虑是：该 HACK 是否会在未来维护中引入问题。

- model_manager.py 中的 HACK 注释 (design): 作者未回应，PR 仍被合并。HACK 未解决。

风险与影响

- 风险：主要风险是回归：统一路径后，未提供测试文件变更（仅描述了测试计划），可能遗漏 LoRA 启用场景下的行为差异。特别是 expand_packed_lora 对 output_sizes 的依赖需要确保对齐。另外，model_manager.py 中的 HACK 代码可能在其他打包模块上表现异常。
- 影响：影响 Qwen3.5 和 Qwen3.5-MoE 模型用户，尤其是使用 LoRA 微调的部署。变更后模型代码路径单一，逻辑更简洁，但需要验证 LoRA 功能正确。系统其他部分不受影响。
- 风险标记：核心路径变更，缺少测试覆盖，HACK 代码，依赖 output_sizes 对齐

关联脉络

- PR #36976 [Feature] Qwen3.5 LoRA support (introduces two forwarding paths): 该 PR 引入了因 LoRA 分裂的两条前向路径，本 PR 在其基础上进行统一。