PR #46998 完整报告

vllm-project/vllm

[Perf] fuse more rmsnorm and all-reduce in qwen3.5

合并时间：2026-07-10 15:37

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/46998

执行摘要

- 一句话：移除预分配缓冲，启用融合优化，提升 Qwen3.5 性能
- 推荐动作：值得精读，展示了一个常见的性能优化模式：移除临时缓冲区以解锁编译器融合。对于关注 Qwen3.5 模型或 PyTorch 编译优化的工程师有参考价值。

功能与动机

PR body 指出：当前 Qwen3NextDecoderLayer 向注意力子模块传递预分配输出缓冲区 (output = torch.empty_like(hidden_states)) 并通过切片赋值写入结果，这使得 PyTorch 编译器的 all-reduce RMSNorm 融合优化无法生效。日志显示 main 分支上 Replaced 0 patterns，而本 PR 后替换为 2 patterns。Issue 中 vadiklyutiy 询问 E2E 改进，作者确认存在提升。

实现拆解

1. 移除输出参数：在 qwen_gdn_linear_attn.py 中，所有 forward 变体（forward, forward_hip, forward_cuda, forward_xpu, forward_cpu）去掉 output 参数，将最后的切片赋值 output[:num_tokens], _ = self.out_proj(...) 改为局部变量 out, _ = self.out_proj(...); return out。辅助方法 _output_projection 同理，移除 output 和 num_tokens，直接返回投影结果。
2. 调整调用方：在 qwen3_next.py 中，Qwen3NextAttention.forward 去掉 output 参数，直接返回 o_proj 结果；Qwen3NextDecoderLayer.forward 不再使用 torch.empty_like 预分配缓冲区，也无需传 output，直接接收子模块返回的张量并赋值给 hidden_states。
3. 一致性保证：同步修改了 HIP/CUDA/XPU/CPU 四个后端的同类函数，保持接口统一。
4. 测试与验证：无新增测试文件，但通过 lm_eval 在 gsm8k 上验证精度无损（PR body 列出了详细对比），且编译日志显示融合模式替换成功。

关键文件：

- vllm/model_executor/layers/mamba/gdn/qwen_gdn_linear_attn.py（模块 线性注意力层；类别 source；类型 core-logic；符号 forward, _output_projection, forward_hip, forward_cuda）：核心文件，修改了所有 forward 变体的签名，移除了 output 参数并改为返回张量，从而解除编译器融合阻塞。
- vllm/model_executor/models/qwen3_next.py（模块 Qwen3Next 模型层；类别 source；类型 core-logic；符号 Qwen3NextAttention.forward, Qwen3NextDecoderLayer.forward）：调用端适配，移除预分配缓冲区，直接使用子模块

返回值。

关键符号：QwenGatedDeltaNetAttention.forward，QwenGatedDeltaNetAttention._output_projection，QwenGatedDeltaNetAttention.forward_hip，QwenGatedDeltaNetAttention.forward_cuda，QwenGatedDeltaNetAttention.forward_xpu，QwenGatedDeltaNetAttention.forward_cpu，Qwen3NextAttention.forward，Qwen3NextDecoderLayer.forward

关键源码片段

vllm/model_executor/layers/mamba/gdn/qwen_gdn_linear_attn.py

核心文件，修改了所有 forward 变体的签名，移除了 output 参数并改为返回张量，从而解除编译器融合阻塞。

```
def _output_projection(
    self,
    core_attn_out: torch.Tensor,
    z: torch.Tensor,
) -> torch.Tensor:
    """Part 3: RMSNormGated + output linear projection.
    现在直接返回张量，使得编译器可以检测到后续 all-reduce 与 norm 的融合机会。
    """
    z_shape_og = z.shape
    core_attn_out = core_attn_out.reshape(-1, core_attn_out.shape[-1])
    z = z.reshape(-1, z.shape[-1])
    # norm 内部包含 RMSNorm 与门控，融合后可与 all-reduce 合并
    core_attn_out = self.norm(core_attn_out, z)
    core_attn_out = core_attn_out.reshape(z_shape_og)
    core_attn_out = core_attn_out.flatten(-2) # ... h d -> ... (h d)
    output, _ = self.out_proj(core_attn_out)
    return output # 返回张量，而非写入传入的预分配缓冲区
```

vllm/model_executor/models/qwen3_next.py

调用端适配，移除预分配缓冲区，直接使用子模块返回值。

```
def forward(
    self,
    hidden_states: torch.Tensor,
    residual: torch.Tensor | None,
    positions: torch.Tensor = None,
    **kwargs: object,
):
    if residual is None:
        residual = hidden_states
        hidden_states = self.input_layernorm(hidden_states)
    else:
        hidden_states, residual = self.input_layernorm(hidden_states, residual)

    # 直接使用子模块返回的张量，无需预分配缓冲区
```

```
if self.layer_type == "linear_attention":
    hidden_states = self.linear_attn(hidden_states=hidden_states)
elif self.layer_type == "full_attention":
    hidden_states = self.self_attn(
        hidden_states=hidden_states,
        positions=positions,
    )
else:
    raise ValueError("Invalid layer_type")

if self.layer_scale:
    if len(hidden_states.shape) == 2:
        hidden_states = hidden_states * (
            self.attn_layer_scale.to(hidden_states.dtype)[0] + 1
        )
    else:
        hidden_states = hidden_states * (
            self.attn_layer_scale.to(hidden_states.dtype) + 1
        )

# Fully Connected
hidden_states, residual = self.post_attention_layernorm(hidden_states, residual)
hidden_states = self.mlp(hidden_states)
# ... 后续不变 ...
```

评论区精华

- vadiklyutiy 要求补充描述，随后作者更新了 PR body。
- vadiklyutiy 询问是否有端到端性能提升，作者确认 "yes"。
- vadiklyutiy 最终批准该 PR，无其他争议。
- 确认 E2E 改进 (question): 确认有端到端提升。

风险与影响

- 风险：
  - 回归风险：修改了四个后端（CUDA/ROCm/XPU/CPU）的 forward 接口，若其他调用点（如外部插件或未覆盖的路径）仍依赖旧签名则可能报错。但仓库内所有调用均已同步修改。
  - 精度风险：输出计算路径完全一致（仅用局部变量代替切片赋值），数值等价，精度不变。
  - 性能不确定性：融合优化依赖 PyTorch 编译器能力，不同版本或运行环境表现可能不同。
  - 测试覆盖：未新增专向测试，依赖已有单测和 lm_eval，对于缺少 CI 的后端（如 XPU、CPU）可能未充分验证。
- 影响：
  - 用户：Qwen3.5 模型推理性能提升（作者确认 E2E 改进），功能无变化。
  - 系统：编译时融合 pattern 增加，可能略微缩短编译时间并减少显存拷贝。

- 团队：降低了后续进一步融合优化的障碍；代码结构更简洁（返回张量而非副作用）。
- 风险标记：缺少直接测试覆盖，多后端改动可能回归

关联脉络

- 暂无明显关联 PR