PR #37144 完整报告

sgl-project/sglang

[Diffusion] Fuse Qwen-Image final adaptive LayerNorm

合并时间：2026-08-31 18:22

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/37144

执行摘要

- 一句话：融合 Qwen-Image 最终 adaLN，bit-exact 自校验回退
- 推荐动作：值得精读。虽然端到端收益为 parity，但 _qwen_norm_out 展示了如何在生产推理引擎中安全地启用 bit-exact kernel 融合：以签名驱动验证、异常和未验证签名自动回退、compile 与 CUDA graph 场景隔离。对于想在 sglang 中做类似融合的开发者，这是很好的参考模板。建议关注后续是否将同类 gate 应用到更多 diT 模型的 final norm。

功能与动机

PR body 明确引用了 [Agentic Kernels in Production] 中描述的 Qwen-Image norm_out 优化：将 norm(hidden_states) * (1 + scale)[:, None, :] + shift[:, None, :] 的多个 eager 操作替换为一个融合 kernel，目标是降低每 denoise step 的 GPU kernel launch 次数，同时以 bit-exact 保证不改变输出。

实现拆解

1. 在 python/sglang/multimodal_gen/runtime/models/dits/qwen_image.py 中从 sglang.kernels.ops.diffusion 导入 BitExactFusionGate、can_use_fused_layernorm_modulate、fused_layernorm_modulate_raw、is_plain_layer_norm；模块级定义 _QWEN_NORM_OUT gate（per_signature=True）并新增 _qwen_norm_out 函数。
2. _qwen_norm_out 的处理顺序：torch.compiler.is_compiling() 时保留 diffusers 原始表达式；否则先计算调制向量并拆出 scale、shift；当 gate 关闭、不是 plain LayerNorm、或平台 / 形状不支持时回退 eager；随后构造签名并检查 verified 集合，CUDA graph 捕获期如果签名未验证也直接回退 eager；launch 内核并捕获异常；对已验证签名直接返回 fused 结果，未验证签名则用 torch.equal 与 eager 参考比对，通过后写入 verified 集合。
3. 在 QwenImageTransformer2DModel.forward 中将 hidden_states = self.norm_out(hidden_states, temb_txt) 改为 hidden_states = _qwen_norm_out(self.norm_out, hidden_states, temb_txt)，这是唯一的调用点替换。
4. 测试配套：test/registered/kernels/ops/diffusion/test_model_fast_paths.py 新增三个测试，分别验证融合结果与 AdaLayerNormContinuous 参考逐位一致、compile 路径不触发 kernel、CUDA graph 捕获期不验证新签名。

5. 基准配套：test/registered/kernels/benchmark/diffusion/bench_qwen_image_modulation.py 新增 bench_norm_out，覆盖 128/512/2048/4096/4608 token 的 norm_out 场景，并限制在非 ROCm 环境运行（PTX 内核不支持 AMD）。

关键文件：

- python/sglang/multimodal_gen/runtime/models/dits/qwen_image.py（模块 模型实现；类别 source；类型 core-logic；符号 _qwen_norm_out）：核心实现文件，新增 _qwen_norm_out 融合函数并将 forward 中的 self.norm_out 调用替换为融合入口，是整个 PR 的源码载体。
- test/registered/kernels/ops/diffusion/test_model_fast_paths.py（模块 快速路径；类别 test；类型 test-coverage；符号 test_qwen_norm_out_matches_adaln_reference, test_qwen_norm_out_preserves_compile_path, test_qwen_norm_out_does_not_verify_during_graph_capture）：新增三个核心单测，覆盖 bit-exact 参考一致性、compile 路径隔离和 CUDA graph 捕获期行为，是保障融合安全性的关键测试。
- test/registered/kernels/benchmark/diffusion/bench_qwen_image_modulation.py（模块 性能基准；类别 test；类型 test-coverage；符号 bench_norm_out）：新增 bench_norm_out 基准，提供 128-4608 token 的 norm_out 融合前后性能数据，并处理 ROCm 跳过逻辑。

关键符号：_qwen_norm_out, test_qwen_norm_out_matches_adaln_reference, test_qwen_norm_out_preserves_compile_path, test_qwen_norm_out_does_not_verify_during_graph_capture, bench_norm_out

关键源码片段

python/sglang/multimodal_gen/runtime/models/dits/qwen_image.py

核心实现文件，新增 _qwen_norm_out 融合函数并将 forward 中的 self.norm_out 调用替换为融合入口，是整个 PR 的源码载体。

```
def _qwen_norm_out(
    norm_out: AdaLayerNormContinuous,
    hidden_states: torch.Tensor,
    conditioning_embedding: torch.Tensor,
) -> torch.Tensor:
    # torch.compile 场景保留 diffusers 原始表达式，避免 trace 时进入 kernel 分支
    if torch.compiler.is_compiling():
        return norm_out(hidden_states, conditioning_embedding)

    # 先算调制向量并拆成 scale、shift；这里直接用子模块，绕开自定义算子包装的调度开销
    emb = norm_out.linear(norm_out.silu(conditioning_embedding).to(hidden_states.dtype))
    scale, shift = torch.chunk(emb, 2, dim=1)

    # 任一前置条件不满足（gate 关闭、非 plain LayerNorm、平台 / 形状不支持）都回退 eager
    if (
        _QWEN_NORM_OUT.disabled
```

```
    or not is_plain_layer_norm(norm_out.norm, hidden_states.shape[-1])
    or not can_use_fused_layernorm_modulate(hidden_states, scale, shift)
):
    return (
        norm_out.norm(hidden_states) * (1 + scale)[:, None, :] + shift[:, None, :]
    )

# 用 dtype/device/stride/eps 组成签名，匹配已 verified 的集合
sig = (
    hidden_states.dtype,
    hidden_states.device,
    hidden_states.shape[0],
    hidden_states.shape[-1],
    hidden_states.stride(-1),
    scale.stride(0) if scale.shape[0] > 1 else hidden_states.shape[-1],
    shift.stride(0) if shift.shape[0] > 1 else hidden_states.shape[-1],
    norm_out.norm.eps,
)
verified = sig in _QWEN_NORM_OUT_SIGS

# CUDA graph 捕获期禁止做首次签名验证，避免在 capture 中触发同步或编译
if not verified and torch.cuda.is_current_stream_capturing():
    return (
        norm_out.norm(hidden_states) * (1 + scale)[:, None, :] + shift[:, None, :]
    )

try:
    fused = fused_layernorm_modulate_raw(
        hidden_states, scale, shift, norm_out.norm.eps
    )
except Exception as exc:
    # 内核 launch 异常时关闭 gate 并回退 eager，防止平台问题导致推理失败
    _QWEN_NORM_OUT.on_exception(exc, logger=logger)
    return (
        norm_out.norm(hidden_states) * (1 + scale)[:, None, :] + shift[:, None, :]
    )

if verified:
    return fused

# 新签名：与 eager 参考逐位比对，一致则写入 verified 集合，否则回退
reference = (
    norm_out.norm(hidden_states) * (1 + scale)[:, None, :] + shift[:, None, :]
)
return _QWEN_NORM_OUT.accept_or_fallback(
    fused,
    reference,
    sig=sig,
    logger=logger,
```

```
        mismatch_msg=(
            'Qwen-Image fused norm_out is not bit-exact on this platform; '
            'falling back to eager'
        ),
    )
```

test/registered/kernels/ops/diffusion/test_model_fast_paths.py

新增三个核心单测，覆盖 bit-exact 参考一致性、compile 路径隔离和 CUDA graph 捕获期行为，是保障融合安全性的关键测试。

```
def test_qwen_norm_out_does_not_verify_during_graph_capture(monkeypatch):
    # 重置 gate 状态，确保捕获期间「未验证」时不会误用融合 kernel
    qwen_image._QWEN_NORM_OUT.disabled = False
    qwen_image._QWEN_NORM_OUT.verified = False
    qwen_image._QWEN_NORM_OUT_SIGS.clear()
    norm_out = (
        qwen_image.AdaLayerNormContinuous(3072, 3072, elementwise_affine=False, eps=1e-6)
        .cuda()
        .bfloat16()
    )
    hidden_states = torch.randn(1, 17, 3072, device='cuda', dtype=torch.bfloat16)
    conditioning = torch.randn(1, 3072, device='cuda', dtype=torch.bfloat16)
    expected = norm_out(hidden_states, conditioning)

    # 模拟 CUDA graph 捕获：任何内核 dispatch 都会让测试失败
    monkeypatch.setattr(torch.cuda, 'is_current_stream_capturing', lambda: True)
    monkeypatch.setattr(
        qwen_image,
        'fused_layernorm_modulate_raw',
        lambda *args, **kwargs: pytest.fail('capture must not verify a new layout'),
    )

    assert torch.equal(_qwen_norm_out(norm_out, hidden_states, conditioning), expected)
    # 捕获期间不允许新增任何签名
    assert not qwen_image._QWEN_NORM_OUT_SIGS
```

评论区精华

该 PR 没有任何 review 评论与讨论线程（author: BBuf 自行合入）。从实现本身可以提炼几个设计权衡：

- compile 路径保留原表达式：避免 torch.compile 追踪时进入 kernel 分支，测试 test_qwen_norm_out_preserves_compile_path 强制断言不 dispatch kernel。
- CUDA graph 捕获期禁止首次验证：避免在 capture 中引入同步或触发编译，未验证的签名直接走 eager（test_qwen_norm_out_does_not_verify_during_graph_capture）。
- 严格 torch.equal 而非容差：测试文件 docstring 明确说明，默认开启的 bit-exact 路径若用容差，真实回归会被 gate 静默吞掉。
- 暂无高价值评论线程

风险与影响

- 风险：
  1. 平台兼容性：fused_layernorm_modulate_raw 依赖 NVIDIA inline PTX，ROCm 上无法使用；测试里 requires_inline_ptx 跳过 AMD，bench 也跳过 norm_out 场景，其余 AMD 场景仅验证 *eager* 回退。
  2. CUDA graph 稳定性：捕获期间未验证的签名回退 *eager*，理论上不会破坏捕获，但若未来新增 shape/stride 组合未在捕获前验证，融合将长期不生效，属于隐式性能退化而非功能错误。
  3. 端到端收益不明确：PR body 实测 *end-to-end* 为 -0.392% 的微小回归（在 4.7-5.7 ms/step 标准差内），说明该优化对整体吞吐提升有限，若其他配置（非 B300、非 3072 hidden）收益更小甚至为负。
  4. 异常兜底：launch 异常时 on_exception 会关闭 *gate* 并回退，但异常路径本身可能掩盖 PTX 变更导致的兼容性问题，需要持续关注 kernel 报错日志。 - 影响：影响范围集中在 Qwen-Image 推理的最后一个 norm 环节，仅改动 qwen_image.py 中的一处调用点。对用户而言，生成图像像素与原先完全一致（PR 中 10/10 输出 SHA256 相同），性能上 kernel 级 latency 降低但端到端基本持平，主要收益是减少每步 GPU kernel launch 次数（每 *denoise step* 减 6 个 launch）。对团队而言，该 PR 提供了一个 BitExactFusionGate 应用的完整范例（签名验证 + 自动回退 + 捕获保护），可复用于后续其他模型的 norm/modulate 融合。 - 风险标记：依赖 NVIDIA PTX, CUDA graph 捕获期限制，端到端收益为 parity, ROCm 自动回退，测试仅覆盖 CUDA

关联脉络

- PR #37116 [diffusion] perf: absorb Qwen-Image output projection biases: 同为 Qwen-Image 模型文件 qwen_image.py 的性能优化，属于同一模型的 kernel 融合系列。
- PR #37141 [Diffusion] Fuse FLUX.2 token concatenation and NVFP4 quantization: 同为 diffusion 模型 kernel 融合工作，采用相似的 bit-exact 门控与自验证模式，可对照演进。