PR #41050 完整报告

vllm-project/vllm

[Kernel][MoE] Support GELU on TRT-LLM NvFP4 fused MoE for Gemma4

合并时间：2026-05-01 11:37

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/41050

执行摘要

- 一句话：为 TRT-LLM NvFP4 MoE 启用 GELU 激活，支持 Gemma4
- 推荐动作：值得精读，尤其是 _supports_activation 集中管理激活列表的设计模式，以及 kernel 测试中如何校准 NvFP4 精度误差。同时展示了通过 Python 层启用 GPU 内核功能的低风险思路。

功能与动机

Gemma4 MoE blocks use gated GELU (GeGLU). Without this change, vLLM falls through to Marlin kernel when running quantized Gemma4 on Blackwell. The GPU kernel and the dispatcher already support GeGLU end-to-end — only the Python allow-list and assertions were gating it out. (from PR body)

实现拆解

1. 修改激活允许列表与断言：在 vllm/model_executor/layers/fused_moe/experts/trtllm_nvfp4_moe.py 中，将 _supports_activation 的返回列表增加 MoEActivation.GELU，并将两个 apply 方法（Modular 和 Monolithic）中的硬编码断言 assert activation in [...] 改为 assert self._supports_activation(activation)，确保新增激活自动生效。
2. 修复测试工具函数：在 tests/kernels/moe/utils.py 的 make_test_quant_config 函数中添加布尔参数 is_nvfp4_scale_swizzled（默认 True 向后兼容），并将其传递给 FusedMoEQuantConfig.make。FlashInfer 内核要求非 swizzled 的 scale 布局，生产代码已在 oracle/nvfp4.py 中正确设置，但测试 helper 默认硬编码了 swizzled 布局，导致新增测试失败。
3. 新增 kernel 测试文件：创建 tests/kernels/moe/test_trtllm_nvfp4_moe.py，通过 TrtLlmNvFp4ExpertsModular 后端的 Modular 路径进行端到端验证。测试参数化三种激活（SiLU、RELU²、GELU）和三种形状（包括 Gemma4 的 (64, 704, 4096) 非 256 对齐场景），使用 make_test_quant_config 生成 NvFP4 量化权重与激活 scale。对 RELU² 在 Gemma4 形状下单跳过错（FlashInfer 缺少 tile 配置，不影响 Gemma4 的 GELU 支持）。
4. 配套计划：测试文件尚未纳入 CI 配置（.buildkite/test_areas/kernels.yaml），review 中已确认由作者在后续 PR 中单独添加。

关键文件：

- tests/kernels/moe/test_trtllm_nvfp4_moe.py（模块 MoE 测试；类别 test；类型 test-coverage；符号 test_trtllm_fp4_moe_no_graph）：新增首个针对 TRT-LLM × NvFP4 后端的 kernel 级测试，覆盖 SiLU、RELU²、GELU，包括 Gemma4 形状。
- vllm/model_executor/layers/fused_moe/experts/trtllm_nvfp4_moe.py（模块 MoE 后端；类别 source；类型 data-contract）：核心变更：将 GELU 加入支持列表，并统一 apply 断言。
- tests/kernels/moe/utils.py（模块 测试工具；类别 test；类型 test-infra）：添加 is_nvfp4_scale_swizzled 参数以修复测试 helper 的默认行为，使 NvFP4 测试可通过。

关键符号：_supports_activation, apply (TrtLlmNvFp4ExpertsModular), apply (TrtLlmNvFp4ExpertsMonolithic), make_test_quant_config, test_trtllm_fp4_moe_no_graph

关键源码片段

tests/kernels/moe/test_trtllm_nvfp4_moe.py

新增首个针对 TRT-LLM × NvFP4 后端的 kernel 级测试，覆盖 SiLU、RELU²、GELU，包括 Gemma4 形状。

```
# tests/kernels/moe/test_trtllm_nvfp4_moe.py ( 新增 )
# 测试 shape 包括 Gemma4 的 (64, 704, 4096)，用于验证非 256 对齐中间尺寸的 padding 路径
MNK_FACTORS = [
    (2, 1024, 1024),
    (64, 2048, 1536),
    (64, 704, 4096),
]

@pytest.mark.parametrize("m,n,k", MNK_FACTORS)
@pytest.mark.parametrize("activation", [
    MoEActivation.SILU,
    MoEActivation.RELU2_NO_MUL,
    MoEActivation.GELU, # 本 PR 新增支持
])
@torch.inference_mode()
def test_trtllm_fp4_moe_no_graph(m, n, k, e, topk, dtype, activation, workspace_init):
    # FlashInfer 的 trtllm_batched_gemm_runner 在 non-gated RELU^2 且非 256 对齐时不提供
    tile 配置
    # Gemma4 使用 gated GELU，不受此限制影响
    if activation == MoEActivation.RELU2_NO_MUL and (m, n, k) == (64, 704, 4096):
        pytest.skip("FlashInfer trtllm_batched_gemm_runner: no tile config for non-gated RELU^2
        at int_size=704")

    set_random_seed(7)
    a = torch.randn((m, k), device="cuda", dtype=dtype) / 10

    # 生成 NvFP4 量化权重和激活 scale，关键：is_nvfp4_scale_swizzled=False 以匹配内核要求
    w1_q, w2_q, quant_config = make_test_quant_config(
        e, n, k, in_dtype=dtype, quant_dtype="nvfp4",
```

```
        make_gate=activation.is_gated,
        is_nvfp4_scale_swizzled=False,
    )

    score = torch.randn((m, e), device="cuda", dtype=dtype)
    topk_weights, topk_ids, _ = fused_topk(a, score, topk, renormalize=False)

    # 构造 Modular 专家模块并执行前向
    trtllm_experts = TrtLlmNvFp4ExpertsModular(...) # 配置省略
    output = trtllm_experts.apply(...)

    # 与 PyTorch 基线对比，使用校准后的 tolerance（约 0.22 最大差异）
    ref = torch_moe(a, w1, w2, score, topk, activation=activation)
    torch.testing.assert_close(output, ref, rtol=1e-1, atol=1e-1)
```

vllm/model_executor/layers/fused_moe/experts/trtllm_nvfp4_moe.py

核心变更：将 GELU 加入支持列表，并统一 apply 断言。

```
# vllm/model_executor/layers/fused_moe/experts/trtllm_nvfp4_moe.py ( 部分修改 )
class TrtLlmNvFp4ExpertsBase:
    @staticmethod
    def _supports_activation(activation: MoEActivation) -> bool:
        """返回此后端支持的激活类型，现包含 GELU。"""
        return activation in [
            MoEActivation.SILU,
            MoEActivation.RELU2_NO_MUL,
            MoEActivation.GELU, # 新增：gated GELU (GeGLU)
        ]

class TrtLlmNvFp4ExpertsModular(TrtLlmNvFp4ExpertsBase, mk.FusedMoEExpertsModular):
    def apply(self, output, hidden_states, w1, w2, topk_weights, topk_ids,
              activation, global_num_experts, expert_map, a1q_scale, a2_scale,
              workspace13, workspace2, expert_tokens_meta, apply_router_weight_on_input):
        # 使用新增的 _supports_activation 统一校验，避免硬编码重复
        assert self._supports_activation(activation), f"Unsupported activation {activation}"
        assert a1q_scale is not None
        assert self.quant_config.w1_scale is not None
        assert self.quant_config.w2_scale is not None
        # ... 后续调用 flashinfer kernel

class TrtLlmNvFp4ExpertsMonolithic(...):
    def apply(self, ...):
        # 同样的修改：使用 _supports_activation 断言代替硬编码列表
        assert self._supports_activation(activation)
        # ...
```

tests/kernels/moe/utils.py

添加 is_nvfp4_scale_swizzled 参数以修复测试 helper 的默认行为，使 NvFP4 测试可通过。

```
# tests/kernels/moe/utils.py ( 部分修改 )
def make_test_quant_config(
    e: int,
    n: int,
    k: int,
    in_dtype: torch.dtype,
    quant_dtype: torch.dtype | str | None = None,
    per_act_token_quant: bool = False,
    block_shape: list[int] | None = None,
    make_gate: bool = True,
    is_nvfp4_scale_swizzled: bool = True, # 新增参数，默认 True 保持向后兼容
) -> tuple[torch.Tensor, torch.Tensor, FusedMoEQuantConfig]:
    (_, w1, w1_s, w1_gs), (_, w2, w2_s, w2_gs) = make_test_weights(...)
    # ...
    return (
        w1,
        w2,
        FusedMoEQuantConfig.make(
            quant_dtype,
            w1_scale=w1_s,
            w2_scale=w2_s,
            is_nvfp4_scale_swizzled=is_nvfp4_scale_swizzled, # 传递参数
            # ...
        ),
    )
```

评论区精华

- gemini-code-assist[bot] 提出测试中量化 scale 应使用 a.abs().max() 而非 a.amax()，以避免负值 clipping。该评论未导致代码修改，PR 合并时认为现有 tolerance 已足够。
- gemini-code-assist[bot] 建议将 apply 中的硬编码断言改为调用 _supports_activation 以避免维护不一致。该建议已被采纳，最终代码使用了 self._supports_activation(activation)。
- LopezCastroRoberto 询问 tolerance 是否高于其他类似测试。作者解释该测试是首个 kernel 级测试，tolerance 基于数据校准且漂移与激活无关。
- ZJY0516 要求将新测试文件添加到 CI 配置，作者同意在后续 PR 中处理。
- 断言硬编码风险 (design): 已采纳并实现，最终代码使用 self._supports_activation(activation)。
- 测试 scale 计算正确性 (correctness): 未修改但合并，可能认为当前 tolerance 已足够。
- Tolerance 校准合理性 (testing): 作者解释合理，未要求修改。
- CI 集成缺失 (testing): 计划后续添加。

风险与影响

- 风险：低风险。核心变更仅为添加一个激活类型到允许列表，不影响现有 SiLU/RELU² 路径。潜在风险包括：测试尚未加入 CI（可能遗漏回归）；FLashInfer 上游 tile 配置缺失导致 RELU² 在特定形状下跳过（已知限制）；第一个 kernel 测试的 tolerance 可能过松，但作

者已给出校准依据。测试 helper 的 is_nvfp4_scale_swizzled 参数默认 True，不破坏现有调用。

- 影响：对用户：Gemma4 用户在 Blackwell GPU 上使用 NvFP4 量化时可自动启用 GELU，推理性能显著提升。对系统：无新增依赖、无运行时开销。对团队：统一了激活允许列表的管理模式，降低后续添加新激活的维护成本。
- 风险标记：测试未加入 CI（计划跟进），RELU² 在特定形状下跳过（已知上游限制），首个 kernel 测试 tolerance 需观察

关联脉络

- PR #39510 [Kernel][MoE] Non-aligned intermediate_size weight padding for TRT-LLM NvFP4 MoE: 该 PR 依赖 #39510 中的非对齐 intermediate_size padding 支持，新测试使用了该路径。
- PR #40563 [WIP][Kernel][MoE] GELU support for TRT-LLM NvFP4 MoE: 此 PR 完成后将关闭 #40563，是同一功能的早期实现。