

PR #55370 完整报告

vllm-project/vllm

[Bugfix] Make `mm\_device\_do\_normalize` encoder-cudagraph safe

合并时间: 2026-09-10 01:43

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/55370>

执行摘要

- 一句话: 归一化下沉视觉塔, 修复编码器 CUDA Graph 下图像嵌入错乱
- 推荐动作: 值得精读。关键看点有三个: 一是「归一化所有权从外层下沉到视觉塔」这一结构性决策, 以及它如何一次性覆盖 eager / encoder cudagraph / DP 三条路径; 二是 coderabbitai 与 cjackal 关于 dtype 预 cast 是否损失精度的争论, 说明 FusedInputNorm 的 uint8 输入契约是这套设计的前提; 三是评论区揭示的未解性能代价 (归一化被捕获进 CG、padding 行被重复计算、每 replay 一次 D2D cast), 阅读时应把它当作后续 #56711 / #56798 的背景。建议顺手检查其他启用了 mm_device_do_normalize 的 VL 模型是否也把 self.input_norm 留在了外层, 并考虑补一个带 cudagraph_mm_encoder 的回归用例。

功能与动机

PR body 明确说明: `mm_device_do_normalize` (由 #50411 引入) 只在 `_process_*_input` 中应用, 而 encoder cudagraph 路径 (`cg_manager.execute` → `encoder_eager_forward()` / `encoder_cudagraph_forward()`) 直接调用 `self.visual(...)`, 于是「unscaled uint8 pixels are passed to the vision tower and image embedding is silently wrong - resulting in complete gibberish on vision questions (mostly whitespaces in case of Qwen2.5)」。作者给出的 before/after 对照显示 main 上模型只输出空白字符, 本 PR 后能正确描述图片。动机本质是消除一条「不崩溃但静默错误」的视觉输入路径。

实现拆解

1. 变更入口: 把归一化模块变成视觉塔的注入依赖。vllm/model_executor/models/qwen2_5_vl.py 的 Qwen2_5_VisionTransformer.__init__ 与 vllm/model_executor/models/qwen2_vl.py 的 Qwen2VisionTransformer.__init__ 新增形参 input_norm: nn.Module | None = None (插在 prefix 之前), 并在 patch_embed 构造之后保存 self.input_norm = input_norm if input_norm is not None else FusedInputNorm.identity()。原因是 encoder cudagraph 路径只经过 self.visual(), 归一化只有放在视觉塔内部才能覆盖全部调用链。
2. 视觉塔 forward: 归一化成为第一道工序。两个 VisionTransformer.forward 的首行由 x.to(device=self.device, dtype=self.dtype) 改为 self.input_norm(x.to(device=self.device), self.dtype); 先搬 device、保持整数像素, 再交给 FusedInputNorm 做 rescale + normalize 并输出 self.dtype。归一化因此被 CUDA

Graph 捕获，与 eager 路径的调用链完全一致。

3. 外层模型改造：删除自有 `self.input_norm`，并清理两个消费点。

Qwen2_5_VLForConditionalGeneration / Qwen2VLForConditionalGeneration 的 `__init__` 改为 `input_norm=FusedInputNorm.from_model_config(self.model_config)` 传入视觉塔；`_process_image_input`、`_process_video_input` 删除显式的 `self.input_norm(...)` 调用，否则视觉塔会二次归一化。

4. DP 分支的 dtype 契约保留。`mm_encoder_tp_mode == "data"` 分支改为传

`pixel_values.type(self.visual.dtype) / pixel_values_videos.type(self.visual.dtype)` 给 `run_dp_sharded_mrope_vision_model`。原先它收到的是「已归一化且已 cast」的浮点张量，现在归一化推迟到视觉塔内，必须显式对齐 dtype，避免 helper 拿到 uint8（其分片与 all-gather 输出构造依赖 dtype）。

5. 测试配套：让测试预处理与 E2E 推理对齐。`tests/models/multimodal/generation/test_qwen2_vl.py` 的 `batch_make_image_embeddings` 与 `batch_make_video_embeddings` 在调用 `image_processor.preprocess / video_processor.preprocess` 前，先取 `llm.get_llm().model_config.multimodal_config.merge_mm_processor_kwargs({})` 并透传 `**mm_kwargs`。原因是测试此前依赖 HF processor 的归一化，而视觉塔现在也会归一化，形成 double normalization，直接导致 mi250 CI 数值比对失败（作者在测试文件评论中确认）。该改动无新增的 cudagraph_mm_encoder 自动化断言，修复验证依赖 PR body 里的手动 e2e 脚本。

关键文件：

- `vllm/model_executor/models/qwen2_5_vl.py`（模块 视觉编码器；类别 source；类型 core-logic；符号 `Qwen2_5_VisionTransformer.init`, `Qwen2_5_VisionTransformer.forward`, `Qwen2_5_VLForConditionalGeneration.init`, `Qwen2_5_VLForConditionalGeneration._process_image_input`）：主修复文件：`Qwen2_5_VisionTransformer.__init__` 新增 `input_norm` 注入参数、`forward` 首行改为 `self.input_norm(...)`，外层 `Qwen2_5_VLForConditionalGeneration` 删除 `self.input_norm` 并在 `_process_image_input / _process_video_input` 中改为 DP 分支显式 dtype cast。
- `vllm/model_executor/models/qwen2_vl.py`（模块 视觉编码器；类别 source；类型 core-logic；符号 `Qwen2VisionTransformer.init`, `Qwen2VisionTransformer.forward`, `Qwen2VLForConditionalGeneration.init`, `Qwen2VLForConditionalGeneration._process_image_input`）：同构修复的第二个模型文件：`Qwen2VisionTransformer.__init__` 增加 `input_norm` 注入、`forward` 首行改为 `self.input_norm(...)`，`Qwen2VLForConditionalGeneration` 同步删除 `self.input_norm` 并在 `_process_image_input / _process_video_input` 的 DP 分支补 `.type(self.visual.dtype)`。
- `tests/models/multimodal/generation/test_qwen2_vl.py`（模块 多模态测试；类别 test；类型 test-coverage；符号 `batch_make_image_embeddings`, `batch_make_video_embeddings`）：测试配套：`batch_make_image_embeddings` 与 `batch_make_video_embeddings` 引入 `merge_mm_processor_kwargs({})`，让测试预处理与推理侧一致，修复因视觉塔内部归一化导致的 double normalization（mi250 CI 失败根因）。

关键符号: Qwen2_5_VisionTransformer.init, Qwen2_5_VisionTransformer.forward,
Qwen2_5_VLForConditionalGeneration._process_image_input,
Qwen2_5_VLForConditionalGeneration._process_video_input,
Qwen2VisionTransformer.init, Qwen2VisionTransformer.forward,
Qwen2VLForConditionalGeneration._process_image_input,
Qwen2VLForConditionalGeneration._process_video_input,
batch_make_image_embeddings, batch_make_video_embeddings

关键源码片段

vllm/model_executor/models/qwen2_5_vl.py

主修复文件: Qwen2_5_VisionTransformer.__init__ 新增 input_norm 注入参数、forward 首行改为 self.input_norm(...), 外层 Qwen2_5_VLForConditionalGeneration 删除 self.input_norm 并在 _process_image_input / _process_video_input 中改为 DP 分支显式 dtype cast。

```
class Qwen2_5_VisionTransformer(nn.Module):
    def __init__(
        self,
        vision_config: Qwen2_5_VLVisionConfig,
        norm_eps: float = 1e-6,
        quant_config: QuantizationConfig | None = None,
        # 新增: 归一化模块改由外层 conditional model 注入,
        # 这样视觉塔自己就对「一定被归一化」负责,
        # 而不再依赖调用方是否走 _process_*_input
        input_norm: nn.Module | None = None,
        prefix: str = "",
    ) -> None:
        super().__init__()
        ...
        self.patch_embed = Qwen2_5_VisionPatchEmbed(
            patch_size=patch_size,
            temporal_patch_size=temporal_patch_size,
            in_channels=in_channels,
            hidden_size=self.hidden_size,
        )
        # 默认 identity: 旧调用方不会因缺参报错, 但也不会归一化,
        # 属于需要上游保证输入已归一化的历史行为
        self.input_norm = (
            input_norm if input_norm is not None else FusedInputNorm.identity()
        )
        ...

    def forward(
        self,
        x: torch.Tensor,
        grid_thw: list[list[int]] | None,
        *,
```

```

encoder_metadata: dict[str, torch.Tensor] | None = None,
) -> torch.Tensor:
    # 核心修复：归一化被提到 forward 的第一行。
    # encoder cudagraph 路径 (EncoderRunner -> encoder_cudagraph_forward)
    # 只调用 self.visual(), 此前会跳过归一化，把未缩放的 uint8 像素直接送进
    # patch_embed, 图像 embedding 静默错误 (Qwen2.5 上表现为大量空白字符)。
    # 这里先搬到 device (仍是整数像素)，再由 FusedInputNorm 完成
    # rescale + normalize 并按 self.dtype 输出。
    hidden_states = self.input_norm(x.to(device=self.device), self.dtype)
    hidden_states = self.patch_embed(hidden_states)
    ...

```

```

class Qwen2_5_VLForConditionalGeneration(nn.Module, SupportsMultiModal, SupportsPP):

```

```

    def __init__(self, *, vllm_config: VllmConfig, prefix: str = ""):
        ...
        with self._mark_tower_model(vllm_config, {"image", "video"}):
            self.visual = Qwen2_5_VisionTransformer(
                vision_config=config.vision_config,
                norm_eps=getattr(config, "rms_norm_eps", 1e-6),
                quant_config=self.quant_config,
                # 所有权下沉：外层不再持有 self.input_norm,
                # 而是把配置好的归一化模块作为依赖注入视觉塔,
                # 顺带让它能被 encoder CUDA Graph 捕获
                input_norm=FusedInputNorm.from_model_config(self.model_config),
                prefix=maybe_prefix(prefix, "visual"),
            )

```

```

    def _process_image_input(self, image_input):
        ...
        if image_input["type"] == "image_embeds":
            image_embeds = image_input["image_embeds"].type(self.visual.dtype)
        else:
            pixel_values = image_input["pixel_values"]
            # 删除了原先的 self.input_norm(pixel_values, self.visual.dtype):
            # 归一化已在 self.visual.forward 内完成，保留会造成 double normalize
            if self.use_data_parallel:
                # DP 分支绕过 visual.forward 的外部归一化，必须自己保持输入契约：
                # 之前的输入是已归一化 + 已 cast 的浮点张量,
                # run_dp_sharded_mrope_vision_model 依赖该 dtype，因此显式 cast
                return run_dp_sharded_mrope_vision_model(
                    self.visual,
                    pixel_values.type(self.visual.dtype),
                    grid_thw_list,
                    rope_type="rope_3d",
                )
            else:
                image_embeds = self.visual(pixel_values, grid_thw=grid_thw_list)

```

vllm/model_executor/models/qwen2_vl.py

同构修复的第二个模型文件：Qwen2VisionTransformer.__init__ 增加 input_norm 注入、forward 首行改为 self.input_norm(...), Qwen2VLForConditionalGeneration 同步删除 self.input_norm 并在 _process_image_input / _process_video_input 的 DP 分支补 . type(self.visual.dtype)。

```
class Qwen2VisionTransformer(nn.Module):
    # Vision checkpoints store qkv pre-fused; merge separate q/k/v into qkv.
    hf_to_vllm_mapper = WeightsMapper(
        orig_to_new_stacked={
            ".q_proj": (".qkv", "q"),
            ".k_proj": (".qkv", "k"),
            ".v_proj": (".qkv", "v"),
        }
    )

    def __init__(
        self,
        vision_config: Qwen2VLVisionConfig,
        norm_eps: float = 1e-6,
        quant_config: QuantizationConfig | None = None,
        # 与 Qwen2.5-VL 保持同一契约：归一化由外部注入
        input_norm: nn.Module | None = None,
        prefix: str = "",
    ) -> None:
        super().__init__()
        ...
        self.patch_embed = Qwen2VisionPatchEmbed(
            patch_size=patch_size,
            temporal_patch_size=temporal_patch_size,
            in_channels=in_channels,
            embed_dim=embed_dim,
        )
        # identity 兜底，避免直接构造视觉塔时报错
        self.input_norm = (
            input_norm if input_norm is not None else FusedInputNorm.identity()
        )
        ...

    def forward(
        self,
        x: torch.Tensor,
        grid_thw: torch.Tensor | list[list[int]] | None,
        *,
        encoder_metadata: dict[str, torch.Tensor] | None = None,
    ) -> torch.Tensor:
        # patchify
        # 归一化前移：encoder cudagraph 直接调用视觉塔时也能正确缩放像素
```

```

x = self.input_norm(x.to(device=self.device), self.dtype)
x = self.patch_embed(x)
...

```

tests/models/multimodal/generation/test_qwen2_vl.py

测试配套: `batch_make_image_embeddings` 与 `batch_make_video_embeddings` 引入 `merge_mm_processor_kwargs({})`, 让测试预处理与推理侧一致, 修复因视觉塔内部归一化导致的 double normalization (mi250 CI 失败根因)。

```

# tests/models/multimodal/generation/test_qwen2_vl.py
def batch_make_image_embeddings(
    image_batches: list[Union[Image.Image, list[Image.Image]]],
    processor: AutoProcessor,
    llm: VllmRunner,
) -> list[Qwen2VLPromptImageEmbeddingInput]:
    ...
    # image to pixel values
    image_processor = processor.image_processor

    # 关键: 让测试预处理与 E2E 推理使用同一套 mm processor kwargs。
    # 当启用 on-device normalization (mm_device_do_normalize) 时, 推理侧会让 processor
    # 跳过 CPU 侧归一化; 测试若仍按 processor 默认参数处理, 像素会先在 CPU 上被归一化一次,
    # 而视觉塔 forward 现在也会归一化一次, 形成 double normalization,
    # 这正是 mi250 CI 数值比对失败的根因。
    mm_config = llm.get_llm().model_config.multimodal_config
    mm_kwargs = mm_config.merge_mm_processor_kwargs({})
    preprocess_result = image_processor.preprocess(
        images=images, return_tensors="pt", **mm_kwargs
    ).data
    pixel_values = preprocess_result["pixel_values"]
    image_grid_thw = preprocess_result["image_grid_thw"]

    # pixel values to embeddings & grid_thws
    def get_image_embeds(model):
        with torch.no_grad():
            visual = model.visual
            pixel_values_on_device = pixel_values.to(visual.device, dtype=visual.dtype)
            # 直接调用视觉塔: 归一化现在发生在它内部,
            # 与推理 (含 encoder cudagraph) 的调用链保持一致
            return visual(pixel_values_on_device, grid_thw=image_grid_thw).cpu()

```

评论区精华

1) coderabbitai 在 `qwen2_5_vl.py:1447` 提出「移除 vision tower 调用前的预 cast」, cjackal 以 `FusedInputNorm` 只接受 uint8、与 cast 可交换 (bit-identical) 为由反驳, bot 随后撤回并承认 uint8 在 bf16 中可精确表示。2) cjackal 在测试文件说明 mi250 CI 失败源于 double normalization, 并把 mm_kwargs 引入测试以对齐推理行为。3) nooooop 在 issue 评论中质疑是否应把 input_norm 放到 `EncoderRunner`, Isotr0py 认为那样很难融进

encoder CUDA Graph; cjackal 进一步承认 CG buffer dtype 固定为 bf16 导致每次 replay 有 D2D cast、且归一化对 CG padding 行也执行，是隐藏算力浪费。Isotr0py 建议启用 fused input norm 时把 buffer dtype 改成 uint8 (#56711)，cjackal 认为 #56798 解决 fp32 立即数内存膨胀更关键，两者都需要。该性能议题在本 PR 中未落地。

- vision tower 调用前的 dtype 预 cast 是否损失精度 (correctness): bot 撤回该结论并确认 uint8 值在 bf16 中可精确表示，预 cast 与 FusedInputNorm 内部 cast 结果一致，finding 不适用于当前输入契约。
- input_norm 是否应该移到 EncoderRunner (design): 维持把归一化放在视觉塔 forward 内的方案，理由是它与 eager 调用链一致且可被 encoder CUDA Graph 捕获；放回 EncoderRunner 会破坏融合路径。
- 编码器 CUDA Graph 下归一化的隐藏开销 (performance): 本 PR 未消解该开销，只作为已知代价接受，后续由 #56711 (uint8 CG buffer) 与 #56798 (fp32 立即数内存) 分头解决。
- mi250 CI 失败源于双重归一化 (testing): 在 batch_make_image_embeddings / batch_make_video_embeddings 中通过 merge_mm_processor_kwargs({}) 透传 mm processor kwargs，使测试预处理与 E2E 推理一致，CI 恢复。

风险与影响

- 风险：1) 数值回归面较宽：归一化位置变化影响 Qwen2-VL / Qwen2.5-VL 的所有视觉输入路径 (eager、encoder CUDA Graph、DP 的 run_dp_sharded_mrope_vision_model，以及测试中直接调用 visual 的路径)，任何未同步的调用方都会出现「归一化一次」或「归一化两次」的偏差。2) input_norm=None 时默认 FusedInputNorm.identity()，即忘记注入就静默不归一化——与本 PR 要修的失效模式同类 (silent wrong)，缺少断言兜底。3) 性能开销未消解：cjackal 确认归一化被捕获进 encoder CG 后，padding 出来的行也会跑归一化；CG buffer dtype 固定为 bf16 导致每次 replay 需要一次 uint8 → bf16 的 D2D cast。4) 内存风险：归一化产生的 fp32 立即数张量在 CG padding 放大下可能达到数百 MB，依赖 #56798。5) 测试覆盖不足：仓库没有针对 cudagraph_mm_encoder 的自动化用例，修复验证依赖手动 e2e；测试改动仅对齐 mm_kwargs 以修复既有 CI。6) 兼容性细节：新增 input_norm 形参插在 prefix 之前，若存在位置参数调用 Qwen2VisionTransformer / Qwen2_5_VisionTransformer 的第三方代码会错位（仓库内调用点均为关键字传参）。
- 影响：用户侧：开启 --compilation-config.cudagraph_mm_encoder True 的 Qwen2-VL / Qwen2.5-VL 用户从「视觉问答输出乱码但不报错」恢复为可用，eager 路径用户预期无感知变化。系统侧：改变了视觉塔的输入契约——现在接受未归一化的整数像素，所有多模态 encoder 入口（含 CUDA Graph 与数据并行）都必须经由视觉塔 forward 才会被归一化；DP 分支需要自行保证 dtype 对齐。工程侧：确立了「归一化归属于视觉塔 forward」的约定，为其他 VL 模型迁移 mm_device_do_normalize 提供范式；同时把 encoder CG buffer dtype (#56711) 与 fp32 立即数内存 (#56798) 两个后续议题显式暴露出来，形成清晰的演进路线。
- 风险标记：视觉输入契约变更，DP 分支 dtype 依赖，默认 identity 静默失败，编码器 CG 计算浪费，缺少 CUDA Graph 自动化回归

关联脉络

- PR #56729 [Security] Cap Qwen-VL video sampling knobs: 同属 Qwen-VL 多模态输入处理链路的近期改动（改 vllm/multimodal/video.py 的采样上限），与本 PR 一样在调整视觉输入进入模型前的契约与约束，可对照阅读。
- PR #56786 [Bugfix][EPD] Preserve media processing options in encoder requests: 关注编码器（encoder）请求中媒体处理选项的透传，与本 PR 中「encoder 路径必须与 eager 路径行为一致」的问题域重叠，也解释了测试里合并 mm processor kwargs 的必要性。