PR #34988 完整报告

sgl-project/sglang

[Diffusion] Reuse SRT SigLIP vision model

合并时间：2026-08-17 09:16

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/34988

执行摘要

- 一句话：diffusion 复用 SRT SigLIP 视觉模型，删约 240 行重复代码
- 推荐动作：值得精读。这个 PR 展示了如何在不破坏 bit-exact 数值的前提下，跨运行时复用模型实现并同步并行组状态，尤其适合关注 diffusion/SRT 架构演进、视觉模型统一维护的工程师。建议重点阅读 gemma_3.py 的权重映射与 _vision_parallel_context，以及 parallel_state.py 的 patch/ 恢复对称逻辑——这是本 PR 最容易出问题的两个设计点。

功能与动机

PR body 明确说明要“replace the duplicated Gemma3 SigLIP implementation with sglang.srt.models.siglip.SiglipVisionModel”，目的是消除两套独立维护的视觉编码器实现，并“allow SigLIP callers to select the SRT vision attention backend and use SDPA for this fixed-length workload”。同时附上了 H100 基准（TP=1 -11.8%、TP=2 -17.3%），说明复用 SRT 实现不仅能减少重复代码，还能带来实际性能收益。

实现拆解

实现按以下 4 步完成：

1. 复用 SRT SigLIP 视觉模型（核心）：在 python/sglang/multimodal_gen/runtime/models/encoders/gemma_3.py 中删除 QuickGELU、SiglipVisionEmbeddings、SiglipMLP、SiglipAttention、SiglipEncoderLayer、SiglipEncoder、SiglipVisionTransformer 等约 270 行本地实现，改为从 sglang.srt.models.siglip 导入 SiglipVisionModel 并在 Gemma3ForConditionalGeneration.__init__ 中构造 vision_tower。由于 SRT 模型保留 vision_model 中间包裹层而 HuggingFace transformers 5.6.0 移除了该层，PR 保留并更新了 param_names_mapping 和 reverse_param_names_mapping，把 vision_tower.encoder... 映射到 vision_tower.vision_model.encoder...，并单独把 self_attn.out_proj 映射到 self_attn.proj。
2. SRT SigLIP 支持显式注意力后端：在 python/sglang/srt/models/siglip.py 中为 SiglipVisionModel、SiglipVisionTransformer、SiglipEncoder、SiglipEncoderLayer 透传新增的 qkv_backend 参数，最终传给 VisionAttention。gemma_3.py 调用时固定传 qkv_backend="sdpa"，针对固定长度视觉编码 workload 选择稳定后端。配套修改 python/sglang/srt/layers/attention/vision.py：_determine_attention_backend 捕获 get_mm() 抛出的 ValueError，当调用方显式传入 backend 且 mm 配置命名空间未发布时降级为使用显式 backend；新增 python/sglang/srt/runtime_context.py 的

is_config_namespace_published 用于判断命名空间是否已发布。

3. 对齐 diffusion TP 与 SRT attention-TP 状态：python/sglang/multimodal_gen/runtime/distributed/parallel_state.py 中 _sync_srt_tp_group/_clear_srt_tp_group 在原有 _TP 同步基础上新增 _ATTN_TP 同步；patch_tensor_parallel_group 在 patch diffusion TP 组时，按需同步 patch SRT 的 _TP 和 _ATTN_TP，并在退出时恢复，且只在 SRT 侧仍指向旧组时 patch，避免覆盖 SRT 自己管理、已指向其他组的场景。
4. 配套测试与加载工具调整：新增 python/sglang/multimodal_gen/test/unit/test_srt_siglip_reuse.py，覆盖 backend 传播、Gemma3 使用 SRT SigLIP、视觉 TP 组恢复、HF 名称映射；test_component_accuracy_parallel_runtime.py 新增 4 个测试验证 SRT TP 组跟随 diffusion 组、SRT 自有组不被覆盖、folding 上下文传播；test_vision_backend_selection.py 新增显式 backend 在 mm 命名空间未发布 / 已发布下的行为测试；python/sglang/multimodal_gen/test/single_test_file/component_accuracy/engine.py 的 AccuracyEngine.transfer_weights 在目标张量 shape 与源张量不一致时改用 load_param_with_weight_loader，以正确处理隐式 SRT shard 的权重加载。

关键文件：

- python/sglang/multimodal_gen/runtime/models/encoders/gemma_3.py（模块 视觉编码；类别 source；类型 data-contract；符号 QuickGELU, forward, SiglipVisionEmbeddings, init）：核心变更：删除约 270 行重复 SigLIP 实现，改为复用 SRT SiglipVisionModel，并保留 / 更新 HF 权重名映射与视觉 TP 组恢复逻辑
- python/sglang/multimodal_gen/runtime/distributed/parallel_state.py（模块 并行状态；类别 source；类型 dependency-wiring）：关键依赖接线：让 diffusion TP 组与 SRT 的 _TP/_ATTN_TP 组联动，避免 patch 覆盖 SRT 自有组
- python/sglang/srt/models/siglip.py（模块 视觉模型；类别 source；类型 data-contract）：SRT SigLIP 全链路新增 qkv_backend 参数，使 diffusion 调用方可以为视觉塔指定 SDPA 后端
- python/sglang/srt/layers/attention/vision.py（模块 后端选择；类别 source；类型 dependency-wiring）：让 VisionAttention 在 mm 配置命名空间不存在时仍支持显式 backend，是 diffusion 运行时能直接调用 SRT 视觉注意力的前提
- python/sglang/srt/runtime_context.py（模块 上下文；类别 source；类型 core-logic；符号 is_config_namespace_published）：新增 is_config_namespace_published API，供 vision.py 判断 mm 命名空间是否可用
- python/sglang/multimodal_gen/test/unit/test_srt_siglip_reuse.py（模块 单测；类别 test；类型 test-coverage；符号 _vision_config, test_siglip_encoder_propagates_attention_backend, test_gemma3_uses_srt_siglip_with_stable_backend, test_gemma3_restores_vision_tensor_parallel_group）：新增单元测试，覆盖 backend 传播、Gemma3 使用 SRT SigLIP、visual TP 组恢复、HF 权重名映射
- python/sglang/multimodal_gen/test/unit/test_component_accuracy_parallel_runtime.py（模块 并行测试；类别 test；类型 test-coverage；符号 test_srt_attention_tp_group_tracks_diffusion_tp_group, test_srt_owned_groups_are_not_overwritten_or_cleared, test_srt_tp_groups_follow_encoder_folding_context,

test_weight_transfer_uses_loader_for_implicit_srt_shard）：验证 SRT TP/ATTN TP 组与 diffusion 组同步、patch 上下文传播、以及隐式 SRT shard 的权重加载路径

- test/registered/unit/layers/attention/test_vision_backend_selection.py（模块 后端测试；类别 test；类型 test-coverage；符号 test_explicit_backend_without_published_mm_context, test_explicit_backend_keeps_published_context_errors）：覆盖 VisionAttention._determine_attention_backend 在 mm 命名空间缺失 / 已发布两种场景下的行为
- python/sglang/multimodal_gen/test/single_test_file/component_accuracy/engine.py（模块 测试引擎；类别 test；类型 test-coverage）：修正 transfer_weights 对 shape 不一致参数的加载策略，确保隐式 SRT shard 走 weight_loader

关键符号：Gemma3ForConditionalGeneration.init, Gemma3ForConditionalGeneration._vision_parallel_context, SiglipVisionModel.init, VisionAttention._determine_attention_backend, is_config_namespace_published, _sync_srt_tp_group, _clear_srt_tp_group, patch_tensor_parallel_group, AccuracyEngine.transfer_weights

关键源码片段

python/sglang/multimodal_gen/runtime/models/encoders/gemma_3.py

核心变更：删除约 270 行重复 SigLIP 实现，改为复用 SRT SiglipVisionModel，并保留 / 更新 HF 权重名映射与视觉 TP 组恢复逻辑

```
class Gemma3ForConditionalGeneration(nn.Module, LayerwiseOffloadableModuleMixin):
    # 适配 HF transformers 5.6.0：官方实现将 SiglipVisionModel 中间的
    # `vision_model` 包裹层去掉，而 SRT 的 SiglipVisionModel 保留了它，
    # 因此载入权重时需要把 HF 键重新映射回嵌套命名空间。
    layerwise_offload_dit_group_enabled = False
    layer_names = ["language_model.layers"]

    param_names_mapping = {
        r"^(vision_tower\.)(embeddings|encoder|post_layernorm|head)\.": r"\1vision_model.\2.",
        r"^(vision_tower\.vision_model\.encoder\.layers\.\d+\.self_attn\.)out_proj\.": r"\1proj.",
    }
    reverse_param_names_mapping = {
        r"^(vision_tower\.)vision_model\.(embeddings|encoder|post_layernorm|head)\.": r"\1\2.",
        r"^(vision_tower\.vision_model\.encoder\.layers\.\d+\.self_attn\.)proj\.": r"\1out_proj.",
    }

    def __init__(self, config, quant_config=None, prefix=""):
        super().__init__()
        self._vision_tensor_parallel_group = get_tp_group()
        # 直接复用 SRT 的 SiglipVisionModel，省去约 240 行重复实现；
        # qkv_backend 固定为 "sdpa"，这是固定长度视觉编码任务的稳定选择。
        self.vision_tower = SiglipVisionModel(
            config=config.vision_config,
            qkv_backend="sdpa",
```

```
            quant_config=quant_config,
            prefix=add_prefix("vision_tower", prefix),
        )
        self.multi_modal_projector = Gemma3MultiModalProjector(config)
        self.language_model = Gemma3TextModel(config)

    def _vision_parallel_context(self):
        # 仅在当前 diffusion TP 组与构造时记录的视觉组不同时才 patch，
        # 避免在常见路径上引入上下文切换开销。
        if get_tp_group() is self._vision_tensor_parallel_group:
            return nullcontext()
        return patch_tensor_parallel_group(self._vision_tensor_parallel_group)
```

python/sglang/multimodal_gen/runtime/distributed/parallel_state.py

关键依赖接线：让 diffusion TP 组与 SRT 的 _TP/_ATTN_TP 组联动，避免 patch 覆盖 SRT 自有组

```
def _sync_srt_tp_group() -> None:
    import sglang.srt.distributed.parallel_state as srt_parallel_state

    # diffusion 并行组初始化后，把 diffusion 的 _TP 同步到 SRT 的 _TP 与
    # _ATTN_TP；SRT 侧已存在的组不覆盖。
    if srt_parallel_state._TP is None:
        srt_parallel_state._TP = _TP
    if srt_parallel_state._ATTN_TP is None:
        srt_parallel_state._ATTN_TP = _TP


def _clear_srt_tp_group() -> None:
    import sglang.srt.distributed.parallel_state as srt_parallel_state

    if srt_parallel_state._ATTN_TP is _TP:
        srt_parallel_state._ATTN_TP = None
    if srt_parallel_state._TP is _TP:
        srt_parallel_state._TP = None


@contextmanager
def patch_tensor_parallel_group(tp_group: GroupCoordinator):
    global _TP_STATE_PATCHED
    assert not _TP_STATE_PATCHED, "Should not call when it's already patched"

    _TP_STATE_PATCHED = True
    old_tp_group = get_tp_group()
    import sglang.srt.distributed.parallel_state as srt_parallel_state

    # 记录 SRT 侧当前是否仍指向旧组，避免 patch 时覆盖 SRT 自己管理的组，
    # 退出时也只恢复那些由本次 patch 改动的组。
    patch_srt_tp = srt_parallel_state._TP is old_tp_group
```

```
    patch_srt_attention_tp = srt_parallel_state._ATTN_TP is old_tp_group
    global _TP
    _TP = tp_group
    if patch_srt_tp:
        srt_parallel_state._TP = tp_group
    if patch_srt_attention_tp:
        srt_parallel_state._ATTN_TP = tp_group
    try:
        yield
    finally:
        _TP_STATE_PATCHED = False
        _TP = old_tp_group
        if patch_srt_tp and srt_parallel_state._TP is tp_group:
            srt_parallel_state._TP = old_tp_group
        if patch_srt_attention_tp and srt_parallel_state._ATTN_TP is tp_group:
            srt_parallel_state._ATTN_TP = old_tp_group
```

评论区精华

该 PR 没有 Review 评论（review_comments_count 为 0），核心取舍只能从 commit 历史与测试设计推断：

- 第 3 个 commit “fix: allow explicit vision backend without SRT config” 暴露了关键设计点：diffusion 运行时没有 SRT 的 mm 配置命名空间，直接调用 VisionAttention 会因 get_mm() 抛 ValueError 而失败，因此需要在显式 backend 时优雅降级。
- 第 4 个 commit “fix: preserve folded SRT SigLIP semantics” 说明作者专门处理了 encoder-folding 场景下 SRT TP 组与 diffusion TP 组的同步问题，最终以“只恢复本次 patch 改动的组”的对称逻辑收口。
- 权重名映射方面，测试 `test_gemma3_maps_hf_siglip_projection_name` 明确锁定了 `out_proj → proj` 的映射关系，说明这是 HF transformers 5.6.0 扁平化后最容易出的兼容性坑。
- 暂无高价值评论线程

风险与影响

- 风险：
  - 并行组状态同步风险：patch_tensor_parallel_group 直接改写 SRT 的 _TP 与 _ATTN_TP，即使有 is old_tp_group 保护，若未来 SRT 侧在 patch 期间重新初始化并行组，仍可能出现指针悬挂或恢复失败。相关逻辑集中在 python/sglang/multimodal_gen/runtime/distributed/parallel_state.py。
  - 权重名映射回归：Gemma3ForConditionalGeneration.param_names_mapping 依赖 HF 与 SRT 命名差异的硬编码正则，若 SRT SigLIP 内部结构调整（如再去除 vision_model 层），映射会静默失效，导致权重错载。
  - 注意力后端语义变化：vision.py 现在会在 mm 上下文缺失时吞掉 ValueError 并降级到显式 backend，若未来有其他调用方误传显式 backend，可能掩盖配置错误；

test_vision_backend_selection.py 已覆盖已发布上下文仍抛错的场景。

  - 行为差异与性能：新实现改用 SRT 的 VisionAttention，在不同平台（如 NPU、AMD）上会走不同的默认 backend，可能与旧的 LocalAttention 行为有细微数值差异，H100 基准只覆盖了 CUDA 平台。
- 影响：影响范围集中在 multimodal_gen（diffusion 运行时）与 SRT 视觉层之间：
- 用户 / 功能影响：Gemma3 视觉编码结果保持 bit-exact（PR body 标注 max_abs=0），但视觉塔延迟在 H100 上下降约 11.8%～17.3%，且未来 SigLIP 类视觉模型可统一走 SRT 实现，减少维护成本。
- 系统影响：diffusion 的 TP 组与 SRT _TP/_ATTN_TP 形成联动，任何并行初始化顺序变化都要同步考虑，团队需要将这一约束纳入并行状态管理的通用约定。
- 代码结构影响：删除约 240 行重复模型代码，新增 4 个测试文件 / 用例，巩固了跨运行时复用视觉编码器的模式。

影响程度中等偏上：虽然不改用户 API，但触及两个子系统的并行状态契约，回归风险主要靠新增单测兜底。

- 风险标记：跨运行时 TP 组同步，权重名映射回归风险，注意力后端选择语义变化，回归由新增单测覆盖

关联脉络

- PR #34991 vlm: streamline vision sdpa reshapes: 与本次修改同一文件 python/sglang/srt/layers/attention/vision.py，同样聚焦视觉注意力热路径与 backend 行为
- PR #34929 [diffusion] Enable breakable CUDA graphs for LTX-2.3: 同属 diffusion 性能 / 架构演进方向，说明 multimodal_gen 运行时正在系统性优化 encoder 与 denoise 执行路径