PR #34980 完整报告

sgl-project/sglang

[Diffusion] Native Hunyuan3D Paint and Delight models

合并时间：2026-08-16 10:03

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/34980

执行摘要

- 一句话：Hunyuan3D Paint/Delight 迁移为 SGLang 原生模型，提速约 4 倍
- 推荐动作：值得精读。重点看三处设计：一是 Hunyuan3DPaintTransformerBlock 的 reference 特征缓存与 mode 机制；二是 StableDiffusionUNetConfig.validate 的严格数据契约；三是 paint 阶段如何通过 use_declared_component 把模型加载让渡给组件管理器。若团队计划迁移其他 diffusion 模型，此 PR 是 diffusers 解除依赖的教科书式案例。

功能与动机

PR body 明确说明动机是“replace the Diffusers-owned Hunyuan3D Paint and Delight modules with native SGLang model ownership”，并强调“so residency, CPU offload, and layerwise offload are managed consistently”。旧实现中 Paint/Delight 组件在 stage 内自行通过 Diffusers pipeline 加载模型，无法纳入组件的统一内存与 offload 管理，且存在静默回退路径掩盖模型缺失问题。

实现拆解

1. 新增原生 SD 2.1 模型层：在 python/sglang/multimodal_gen/runtime/models/dits/stable_diffusion.py 新增 918 行，定义 StableDiffusionUNetConfig（frozen dataclass, from_dict 解析 checkpoint 配置并 validate 严格校验四层 SD2.1 布局）、StableDiffusionAttention（基于 F.scaled_dot_product_attention）、BasicTransformerBlock、Transformer2DModel、ResnetBlock2D 等；同时在 configs/models/vaes/stable_diffusion.py 新增 StableDiffusionVAEArchConfig 与 StableDiffusionVAEConfig，并将 runtime/models/vaes/autoencoder.py 的 AutoencoderKL 构造函数签名从 FluxVAEConfig 泛化为 VAEConfig。
2. 新增 Hunyuan3DPaintUNet：在 hunyuan3d_paint.py 中把 base UNet 深拷贝为 unet 与 unet_dual 两份，通过 _replace_transformer_blocks 将 transformer 块替换为 Hunyuan3DPaintTransformerBlock；unet 开启 Multiview Attention 与 Reference View Attention，unet_dual 保持标准 UNet。conv_in 改为 12 通道并新增 learned_text_clip_gen/learned_text_clip_ref 参数与 class embedding；forward 通过 cross_attention_kwargs 中的 mode（w/r）与 condition_embed_dict 跨 denoising step 缓存参考特征，避免重复计算。
3. 重写 paint 阶段：paint.py 净删 200+ 行，删除对 StableDiffusionInstructPix2PixPipeline 的 stage-local 加载与静默回退，改为构造函数注入已装配的原生组件（

transformer/vae/text encoder/tokenizer/scheduler），通过 use_declared_component 声明组件使用，component_uses 向组件管理器注册 prompt/encode/denoise/decode 各 phase。

4. 管线装配与清理：hunyuan3d_pipeline.py 将 _resolve_paint_dir 泛化为 _resolve_model_subfolder，新增 _load_texture_components、_load_stable_diffusion_unet、_load_stable_diffusion_vae、_load_component_weights 等加载助手，以 meta device + strict 权重加载；CLIP 文本编码器走 TextEncoderLoader。hunyuan3d.py 删除 841 行 Diffusers 胶水代码（SGLangAttentionWrapper、Basic2p5DTransformerBlock 等）；shape.py 增加 warmup 短路避免启动时导出文件。
5. 配套测试与语义修正：新增 test_hunyuan3d_native_texture_models.py（原生 UNet/VAE 与 Diffusers 逐层对齐、Paint reference 分支、warmup 输出、turbo schedule）；修正 Delight 采样语义以保持与旧路径 bit-exact；turbo 改走 canonical LCM schedule，去掉 redundant scheduler 告警。

关键文件：

- python/sglang/multimodal_gen/runtime/models/dits/stable_diffusion.py（模块 UNet 模型；类别 source；类型 core-logic；符号 StableDiffusionUNetConfig, from_dict, validate, StableDiffusionAttention）：新增 918 行原生 SD 2.1 UNet 实现，是整个迁移的模型地基，包含严格配置校验与原生注意力实现
- python/sglang/multimodal_gen/runtime/models/dits/hunyuan3d_paint.py（模块 Paint 模型；类别 source；类型 core-logic；符号 Hunyuan3DPaintTransformerBlock, Hunyuan3DPaintUNet, compute_voxel_grid_mask, compute_multi_resolution_mask）：新增 386 行 Hunyuan3D Paint 专用多视角 UNet，实现 reference/multiview 注意力与跨 step 特征缓存
- python/sglang/multimodal_gen/runtime/pipelines_core/stages/model_specific_stages/hunyuan3d/paint.py（模块 Paint 阶段；类别 source；类型 core-logic；符号 _run_delight, _encode_delight_prompts, component_uses, Hunyuan3DPaintPreprocessStage）：Paint/Delight 推理阶段整体重写，本 PR 最大的行为变更点（+625/-881），从 Diffusers pipeline 切换为原生组件组合
- python/sglang/multimodal_gen/runtime/pipelines/hunyuan3d_pipeline.py（模块 管线装配；类别 source；类型 core-logic；符号 _resolve_model_subfolder, _load_texture_components, _load_stable_diffusion_unet, _load_stable_diffusion_vae）：管线装配层增加原生组件加载逻辑（+233/-32），负责 Paint/Delight 的 transformer/VAE/CLIP 组件实例化与设备放置
- python/sglang/multimodal_gen/runtime/models/dits/hunyuan3d.py（模块 3D 模型；类别 source；类型 refactor；符号 SGLangAttentionWrapper, Basic2p5DTransformerBlock, _chunked_feed_forward）：删除 841 行 Diffusers 胶水代码（SGLangAttentionWrapper、Basic2p5DTransformerBlock 等），是本 PR 清理旧路径的关键动作
- python/sglang/multimodal_gen/test/unit/test_hunyuan3d_native_texture_models.py（模块 单元测试；类别 test；类型 test-coverage；符号 TestNativeStableDiffusionUNet, TestNativeStableDiffusionVAE, TestHunyuan3DWarmupOutput,

TestHunyuan3DPaintTurboSchedule）：新增测试覆盖原生 UNet/VAE 与 Diffusers 的逐层对齐、Paint reference 分支、warmup 输出与 turbo schedule，是数值 parity 的关键保障

- python/sglang/multimodal_gen/configs/models/vaes/stable_diffusion.py（模块 VAE 配置；类别 source；类型 data-contract；符号 StableDiffusionVAEArchConfig, StableDiffusionVAEConfig）：新增 Stable Diffusion VAE 配置类，支撑 Paint/Delight 的 VAE 原生加载
- python/sglang/multimodal_gen/runtime/models/vaes/autoencoder.py（模块 VAE 模型；类别 source；类型 refactor；符号 AutoencoderKL）：AutoencoderKL 构造函数从 FluxVAEConfig 泛化为 VAEConfig，使同一实现可承载 SD VAE
- python/sglang/multimodal_gen/configs/pipeline_configs/hunyuan3d.py（模块 管线配置；类别 source；类型 data-contract）：pipeline 配置新增 CLIP 文本编码器与 native_only_components 声明，支撑原生组件注册
- python/sglang/multimodal_gen/runtime/pipelines_core/stages/model_specific_stages/hunyuan3d/shape.py（模块 形状阶段；类别 source；类型 bugfix）：增加 warmup 短路，避免服务启动时导出文件，修复 Paint 场景下 warmup 失败问题
- python/sglang/multimodal_gen/runtime/managers/memory_managers/layerwise_offload_components.py（模块 卸载管理；类别 source；类型 core-logic）：补充 Paint/Delight 组件的 layerwise offload 分组注册，使新组件纳入统一 offload 管理

关键符号：Hunyuan3DPaintTransformerBlock.forward, Hunyuan3DPaintUNet.init, Hunyuan3DPaintUNet.forward, compute_voxel_grid_mask, compute_multi_resolution_mask, StableDiffusionUNetConfig.from_dict, StableDiffusionUNetConfig.validate, StableDiffusionAttention.forward, _load_texture_components, _load_stable_diffusion_unet, _load_stable_diffusion_vae, _run_delight, _encode_delight_prompts, component_uses

关键源码片段

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

新增 918 行原生 SD 2.1 UNet 实现，是整个迁移的模型地基，包含严格配置校验与原生注意力实现

```
# 原生 SD 2.1 UNet 配置：frozen dataclass，直接承载 Hunyuan3D checkpoint 的 config.json
@dataclass(frozen=True)
class StableDiffusionUNetConfig:
    sample_size: int
    in_channels: int
    out_channels: int
    down_block_types: tuple[str, ...]
    up_block_types: tuple[str, ...]
    block_out_channels: tuple[int, ...]
    layers_per_block: int
    cross_attention_dim: int
    attention_head_dim: tuple[int, ...]
```

```
    use_linear_projection: bool

    @classmethod
    def from_dict(cls, config: dict[str, Any]) -> StableDiffusionUNetConfig:
        block_out_channels = tuple(config['block_out_channels'])
        # attention_head_dim 可能是 int（表示所有 stage 相同），也可能是列表，
        # 这里统一展开成与 block_out_channels 等长的元组。
        attention_head_dim_value = config['attention_head_dim']
        attention_head_dim = (
            (attention_head_dim_value,) * len(block_out_channels)
            if isinstance(attention_head_dim_value, int)
            else tuple(attention_head_dim_value)
        )
        parsed = cls(
            sample_size=int(config['sample_size']),
            in_channels=int(config['in_channels']),
            out_channels=int(config['out_channels']),
            down_block_types=tuple(config['down_block_types']),
            up_block_types=tuple(config['up_block_types']),
            block_out_channels=block_out_channels,
            layers_per_block=int(config['layers_per_block']),
            cross_attention_dim=int(config['cross_attention_dim']),
            attention_head_dim=attention_head_dim,
            use_linear_projection=bool(config.get('use_linear_projection', False)),
        )
        parsed.validate()
        return parsed

    def validate(self) -> None:
        # 原生实现目前只支持 Hunyuan3D 使用的 SD2.1 四层布局：
        # 一旦 checkpoint 出现新的 block 类型组合，这里会立即抛错，
        # 避免在静默状态下产出错误结果。
        expected_down = (
            'CrossAttnDownBlock2D', 'CrossAttnDownBlock2D',
            'CrossAttnDownBlock2D', 'DownBlock2D',
        )
        expected_up = (
            'UpBlock2D', 'CrossAttnUpBlock2D',
            'CrossAttnUpBlock2D', 'CrossAttnUpBlock2D',
        )
        if self.down_block_types != expected_down or self.up_block_types != expected_up:
            raise ValueError(
                'The native SD2 UNet currently supports only the Hunyuan3D '
                'four-level SD2.1 block layout.'
            )
        if not self.use_linear_projection:
            raise ValueError('Hunyuan3D SD2.1 checkpoints require linear projection.')
```

python/sglang/multimodal_gen/runtime/models/dits/hunyuan3d_paint.py

新增 386 行 Hunyuan3D Paint 专用多视角 UNet，实现 reference/multiview 注意力与跨 step 特征缓存

```
# 核心多视角 UNet 的 transformer 块包装器：在标准 SD2 注意力之外，
# 追加 Multiview Attention（MVA）与 Reference View Attention（RVA）两条分支。
class Hunyuan3DPaintTransformerBlock(nn.Module):
    def __init__(self, transformer, layer_name, *, use_multiview_attention, use_reference_
    attention, is_turbo):
        super().__init__()
        self.transformer = transformer # 复用原生 SD2 的 BasicTransformerBlock
        self.layer_name = layer_name
        self.attn_multiview = StableDiffusionAttention(...) if use_multiview_attention else None
        self.attn_refview = StableDiffusionAttention(...) if use_reference_attention else None
        if is_turbo:
            self._initialize_added_attention()

    def _initialize_added_attention(self) -> None:
        # turbo 模式没有独立训练的 MVA/RVA 权重：从 attn1 复制权重并把输出投影清零，
        # 使新增分支初始化为零贡献，保证未激活时不改变输出。
        for attention in (self.attn_multiview, self.attn_refview):
            if attention is None:
                continue
            attention.load_state_dict(self.transformer.attn1.state_dict())
            with torch.no_grad():
                for parameter in attention.to_out[0].parameters():
                    parameter.zero_()

    def forward(self, hidden_states, encoder_hidden_states, attention_mask=None, cross_
    attention_kwargs=None):
        options = {} if cross_attention_kwargs is None else cross_attention_kwargs
        num_views = int(options.get('num_in_batch', 1))
        mode = options.get('mode')
        condition_embeddings = options.get('condition_embed_dict')
        if mode is not None and not isinstance(condition_embeddings, dict):
            raise ValueError('Hunyuan3D reference attention requires a shared cache.')

        normalized = self.transformer.norm1(hidden_states)
        # 先走标准自注意力（与普通 SD2 UNet 完全一致）
        hidden_states = hidden_states + self.transformer.attn1(normalized, attention_mask=
        attention_mask)

        # mode 含 'w'：把当前层归一化特征写入共享缓存，供参考视角复用
        if mode is not None and 'w' in mode:
            condition_embeddings[self.layer_name] = rearrange(
                normalized, '(b n) l c -> b (n l) c', n=num_views
            )

        # mode 含 'r'：从缓存取参考特征并按 batch 展开，做参考视角注意力
        if mode is not None and 'r' in mode and self.use_reference_attention:
```

```
            reference = condition_embeddings[self.layer_name]
            reference = reference.unsqueeze(1).repeat(1, num_views, 1, 1)
            reference = rearrange(reference, 'b n l c -> (b n) l c')
            reference_output = self.attn_refview(normalized, encoder_hidden_states=reference)
            reference_scale = self._broadcast_scale(
                1.0 if self.is_turbo else options.get('ref_scale', 1.0),
                reference_output, num_views,
            )
            hidden_states = hidden_states + reference_scale * reference_output

        if num_views > 1 and self.use_multiview_attention:
            # 把 (b n) 维度折叠成 batch，让 n 个视角互相做注意力
            multiview = rearrange(normalized, '(b n) l c -> b (n l) c', n=num_views)
            position_mask = None
            if isinstance(options.get('position_attn_mask'), dict):
                position_mask = options['position_attn_mask'].get(multiview.shape[1])
            multiview_output = self.attn_multiview(
                multiview, encoder_hidden_states=multiview, attention_mask=position_mask
            )
            multiview_output = rearrange(multiview_output, 'b (n l) c -> (b n) l c', n=num_views)
            hidden_states = hidden_states + multiview_output

        return hidden_states + self.transformer.ff(self.transformer.norm3(hidden_states))
```

评论区精华

该 PR 没有任何 review 评论，仅有一条作者自己的 /tag-and-rerun-ci extra 指令。核心验证结论全部来自 PR body：

> Paint UNet parity at B=2, N=6, H=W=64: max absolute error 0, cosine 1.0 Delight two-step image path at 512x512: pixel-exact with the previous Diffusers path Paint denoising step: previous 416.688 ms → native first step 98.038 ms → cached step 76.888 ms

值得注意的设计权衡（从实现推导）：reference attention 通过共享 condition_embed_dict 跨 step 缓存，以空间换时间；turbo 模式新增注意力分支从 attn1 复制权重并零化输出投影，使分支初始化为恒等映射。这些决策没有经过 review 交锋，建议后续维护者关注。

- 无 review 评论，验证数据集中在 PR body (testing): 无待解决的 review 意见；建议后续维护者独立复跑 parity 测试确认数据可复现。
- 移除静默回退路径的行为影响 (design): 设计上有意收紧；实际影响是模型目录不完整时启动更早失败，错误信息更明确。

风险与影响

- 风险：
  1. 大段删除的引用风险：hunyuan3d.py 删除 841 行，SGLangAttentionWrapper、Basic2p5DTransformerBlock、_chunked_feed_forward 等符号被移除，若有其他模块

（如自定义 checkpoint 或工具脚本）仍在引用会直接 ImportError。

2. 精度非完全一致：native CLIP 与 Transformers checkpoint 输出 cosine 为 0.9999982，并非 bit-exact，可能对依赖精确文本特征的场景产生微小差异。
3. 硬编码配置校验：StableDiffusionUNetConfig.validate() 硬编码四层 SD2.1 布局与 use_linear_projection=True，未来若出现新的 checkpoint 布局会直接拒绝加载。
4. 行为变更：移除静默回退后，模型缺失或下载失败会直接抛错，对依赖旧静默行为的用户是行为变化。
5. 测试覆盖范围：单元测试使用 32 通道小配置验证结构层面 parity，真实 checkpoint 的 E2E 验证主要依赖作者自测数据，缺少 CI 中可重复的数值回归基准。 - 影响：影响范围限定在 sglang.multimodal_gen 下的 Hunyuan3D2 pipeline：Paint/Delight 组件从 Diffusers 管控转为 SGLang 原生组件，CPU offload、layerwise offload 与 residency 控制从此统一；Paint 去噪单步提速约 4-5 倍，显著降低纹理生成延迟。对使用标准 --dit-cpu-offload、--dit-layerwise-offload、VAE offload 的现有用户，CLI 行为不变；pipeline config 新增 text_encoder_configs 与 native_only_components 字段，仅影响内部装配。对团队而言，此 PR 是 diffusion 原生集成契约（PR#34952）的首次完整落地，为后续其他模型去 Diffusers 化提供了可参照样板。 - 风险标记：核心路径重构，大段代码删除，精度非完全一致，移除静默回退，硬编码配置校验，缺少第三方 review

关联脉络

- PR #34952 docs: define native diffusion model integration contract: 该 PR 定义了 diffusion 原生集成的契约（TP/SP、VAE 并行与 offload 要求），本 PR 是第一个完整落地该契约的模型迁移。
- PR #34949 [Diffusion] Route MiniMax H3 VAE attention through native backends: 同一原生化的演进方向：把 VAE/ 注意力组件从 Diffusers 自有实现切到 SGLang 原生后端，与本 PR 的模型归属迁移目标一致。
- PR #34891 fix(diffusion): scope attention backend fallback: 修复 diffusion 注意力后端回退过严的问题，与本 PR 中移除静默回退、统一组件管理的行为变更同属 diffusion 路径可靠性治理。