PR #35946 完整报告

sgl-project/sglang

[diffusion] feature: use a directory for the vae mapping gate

合并时间：2026-08-22 19:12

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/35946

执行摘要

- 一句话：修复 VAE 映射门禁误用 repo id 导致匿名内存复制
- 推荐动作：值得精读，但重点不在代码量而在缺陷模式：这是一个典型的“函数入参语义与调用方假设不匹配”导致的静默失效——checkpoint_bytes 对不存在的路径返回 0 而非报错，使门禁逻辑静默失效且无任何告警。值得关注的设计决策：(1) 用“组件路径的父目录”作为部署体量的代理指标，避开 repo id 非本地路径问题；(2) 测试用同一个用例同时钉住两个半边（父目录求和、repo id 归零），把缺陷的根因固化为契约。后续类似“gate/ 策略基于路径或字节数判断”的代码都值得借鉴这种双断言测试。

功能与动机

PR body 明确指出这是 #35862 的 follow-up defect：门禁用 checkpoint_bytes(server_args.model_path) 称重部署，但 model_path 在非本地启动时是 hub repo id（MiniMaxAI/MiniMax-H3），glob 不到任何文件返回 0，host_copies_would_not_fit(0) 永远为 False，门禁从未触发。实际后果是 MiniMax-H3 的视频 VAE 在恰好 32 GB 的主机上被复制进 9.7 GB 匿名主机内存（负载日志从 host pageable: 9.70 GB 变为修复后的 host mmap: 9.70 GB），而该映射机制正是为这类主机设计的。

实现拆解

实现拆解如下：

1. 定位根因：vae_loader.py 的 load_customized 在判断 keep_mapping 时，调用 checkpoint_bytes(server_args.model_path) 称重整个部署。model_path 在非本地启动时是 hub 仓库 ID，不是本地目录，checkpoint_bytes 内部 glob 不到任何 .safetensors 文件，返回 0，导致 keep_checkpoint_mapped 永远返回 False，VAE 权重走匿名内存复制路径。
2. 更换称重对象：将 weight_bytes 的来源从 server_args.model_path 改为 os.path.dirname(str(component_model_path))。理由：交给 loader 的 component_model_path 始终是本地目录，而其父目录（如 .../FL2VA/）持有该变体部署的其余部分（video_vae、transformer 等兄弟组件），因此父目录大小能真实反映整个部署的体量，门禁可以正确触发。改动仅 7 行，位于 keep_mapping 计算分支内。

3. 测试配套：在 test_vae_loader.py 新增 TestDeploymentBytesRoot 测试类，方法 test_the_component_parent_carries_the_variant_weight 在一个临时目录中构造 FL2VA/video_vae 与 FL2VA/transformer 两个组件并写入大小不同的假权重文件，断言 checkpoint_bytes 对父目录返回两者之和（640 字节），对 repo id MiniMaxAI/MiniMax-H3 返回 0。测试同时钉住两个半边：父目录能正确称重、repo id 必须返回 0（从而证明门禁绝不能接收 repo id）。测试文件同步从 runtime.loader.utils 导入 checkpoint_bytes。
4. 验证：PR 在真实机器上验证，加载日志显示 host pageable: 9.70 GB 变为 host mmap: 9.70 GB，证明 VAE 权重改为文件映射而非匿名内存复制。

关键文件：

- python/sglang/multimodal_gen/runtime/loader/component_loaders/vae_loader.py（模块 VAE 加载；类别 source；类型 core-logic；符号 load_customized）：核心修复文件：load_customized 中 keep_mapping 门禁的称重对象从 server_args.model_path（可能是 hub repo id）改为组件路径的父目录，是本次缺陷修复的唯一源码改动。
- python/sglang/multimodal_gen/test/unit/test_vae_loader.py（模块 VAE 加载；类别 test；类型 test-coverage；符号 TestDeploymentBytesRoot, test_the_component_parent_carries_the_variant_weight）：新增 TestDeploymentBytesRoot 测试类，用同一个用例钉住两个半边：父目录累加兄弟分片、repo id 归零，防止 gate 再次被喂 repo id 的回归。

关键符号：load_customized, checkpoint_bytes, keep_checkpoint_mapped, test_the_component_parent_carries_the_variant_weight

关键源码片段

python/sglang/multimodal_gen/runtime/loader/component_loaders/vae_loader.py

核心修复文件：load_customized 中 keep_mapping 门禁的称重对象从 server_args.model_path（可能是 hub repo id）改为组件路径的父目录，是本次缺陷修复的唯一源码改动。

```
# vae_loader.py 中 load_customized 的核心内存策略分支（head 版本关键片段）

loaded = {}
for sf_path in safetensors_list:
    loaded.update(safetensors_load_file(sf_path))
_backfill_ltx2_audio_vae_latent_stats(loaded, component_name)
strict_load = native_only

# `loaded` 持有 safetensors 映射的视图。当组件在 CPU 上启动且主机无法
# 承受整个部署的副本时，用 assign 方式加载可以让权重保持文件映射，
# 而不是复制进匿名主机内存：页面缓存可以丢弃并重新取回文件字节，
# 而匿名内存的每个字节都会挤占 stepped 组件的 pin 预算——
# MiniMax-H3 的 video VAE 高达 9.70 GiB，而预算只有 32 GiB。
# 主机内存充裕时仍默认走复制，因为副本的页面常驻，首次访问无需缺页。
```

```
# MPS 总是 assign，内存是统一的。dtype 与参数不匹配的张量会被转换，
# 恰好只复制那些无法保持映射的张量。
keep_mapping = component_starts_on_cpu and (
    current_platform.is_mps()
    or keep_checkpoint_mapped(
        # server_args.model_path 可能是 hub 仓库 ID，在任何地方都不是目录，
        # glob 不到文件会返回 0 字节，导致门禁永不触发。组件路径则始终是
        # 本地目录，且其父目录持有该变体部署的其余部分（如 FL2VA/ 下的
        # video_vae 与 transformer），因此称重父目录才能反映真实部署体量。
        weight_bytes=checkpoint_bytes(
            os.path.dirname(str(component_model_path))
        ),
        component=f"{component_name or 'vae'} (VAE)",
    )
)
if keep_mapping:
    _match_checkpoint_dtypes(loaded, vae.state_dict())
vae.load_state_dict(
    loaded,
    strict=strict_load,
    assign=keep_mapping,
)
```

python/sglang/multimodal_gen/test/unit/test_vae_loader.py

新增 TestDeploymentBytesRoot 测试类，用同一个用例钉住两个半边：父目录累加兄弟分片、repo id 归零，防止 gate 再次被喂 repo id 的回归。

```
# test_vae_loader.py 中新增的 TestDeploymentBytesRoot（head 版本）

class TestDeploymentBytesRoot(unittest.TestCase):
    """A hub repo id is not a directory; the component path always is."""
    # 一个 hub 仓库 ID 不是任何目录；组件路径则始终是本地目录。

    def test_the_component_parent_carries_the_variant_weight(self):
        with TemporaryDirectory() as root:
            # 模拟 MiniMax-H3 的部署布局：父目录 FL2VA 下既有 video_vae
            # 也有 transformer，两者都是该变体的一部分。
            variant = pathlib.Path(root) / "FL2VA"
            (variant / "video_vae").mkdir(parents=True)
            (variant / "transformer").mkdir()
            (variant / "video_vae" / "w.safetensors").write_bytes(b"x" * 128)
            (variant / "transformer" / "w.safetensors").write_bytes(b"x" * 512)
            # 父目录应累加所有兄弟分片：128 + 512 = 640。
            self.assertEqual(
                checkpoint_bytes(str(variant)),
                640,
                "the parent of a component dir sums every sibling's shards",
            )
            # repo id 不是目录，glob 不到任何文件，必须返回 0——
```

```
        # 这正是门禁绝不能接收 repo id 的原因，钉住此行为防止回归。
        self.assertEqual(
            checkpoint_bytes("MiniMaxAI/MiniMax-H3"),
            0,
            "a repo id globs nothing -- which is why the gate must never "
            "be fed one",
        )
```

评论区精华

该 PR 无 review 评论和讨论线程，提交说明由板主自行合并。值得注意的设计表达体现在 PR body 和测试注释中：

> "The component path handed to the loader is always a local directory, and its parent holds the rest of the variant being deployed (.../FL2VA/), so the gate now weighs that."
>
> "a repo id globs nothing -- which is why the gate must never be fed one."

测试断言消息本身承担了文档职责，把“为什么 gate 不能接收 repo id”固化为可执行的契约。

- 暂无高价值评论线程

风险与影响

- 风险：技术风险评估：

1. 依赖目录结构约定：修复假设组件路径的父目录恰好包含该变体部署的全部组件分片。若未来组件打包方式变化（例如组件直接位于部署根目录、或父目录混入无关大文件），可能导致称重偏高或偏低，门禁误判。当前 FL2VA/ 结构在 MiniMax-H3 部署中是固定的，风险可控。
2. 边界情况：os.path.dirname 对相对路径或根路径可能返回空字符串或意外值；但 loader 接收的 component_model_path 在现有代码路径中始终是本地绝对目录，此风险低。
3. 回归风险：此前的行为（喂 repo id）在本地启动时恰好可用（本地 model_path 是目录），更换称重对象后本地与远端行为统一，反而消除了分支差异；但任何后续改动若重新引入基于 server_args.model_path 的称重，会再次触发同类缺陷，测试 TestDeploymentBytesRoot 对 repo id 返回 0 的断言提供了回归保护。
4. 影响面：局限在 VAE 组件 CPU 启动 + 主机内存受限的路径，不涉及 GPU 路径、MPS（始终 assign）和现有映射逻辑本身。 - 影响：影响评估：

- 用户 / 部署影响：修复 MiniMax-H3 在 32 GB 主机上的匿名内存占用问题，单个 VAE 即可省下 9.7 GB 匿名内存，这些字节原本会挤占 stepped 组件的 pin budget。日志从 host pageable 变为 host mmap，权重改为文件映射后页面缓存可丢弃和重取。
- 系统影响：涉及 load_customized 内存策略分支，但逻辑变化极小（仅称重对象换源），不改变 keep_checkpoint_mapped 决策函数本身。
- 团队影响：改动规模小（2 文件，+34/-2），单 commit 合入；测试覆盖了关键契约，后续维护者修改该路径时有明确回归保护。
- 风险标记：依赖目录结构约定，repo id 误用回归风险，内存策略核心路径，单点测试覆盖

关联脉络

- PR #35862 VAE mapping gate 的原始引入 PR（PR body 明确指出的 follow-up 对象）：本 PR 修复 #35862 引入的缺陷：gate 用 server_args.model_path 称重，而该字段在非本地启动时是 hub repo id，导致 gate 永不触发。
- PR #36085 [Diffusion] Support VAE weight-file overrides: 同样修改 vae_loader.py，演进 VAE 组件的权重加载能力，与本 PR 的组件路径处理逻辑同属 VAE 加载链路。
- PR #36078 [Diffusion] Add composable component weight path CLI: 为组件引入可组合的权重路径 CLI，与本 PR 中 component_model_path 的本地目录语义相关，两者共同完善组件级权重部署能力。