PR #25845 完整报告

sgl-project/sglang

Revert "[codex] Update Wan2.2 ModelOpt CI checkpoints"

合并时间：2026-05-20 12:45

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/25845

执行摘要

- 一句话：回滚 #25483，恢复 Wan2.2 ModelOpt 检查点为旧版 lmsys 路径
- 推荐动作：该 revert 保证主线稳定性，应被接受。建议后续：
  - 分析原 PR 的 CI 失败原因（可能是环境变量或权重版本不匹配）。
  - 若需要重新引入，应先在分支上充分验证。
  - 文档和示例应尽快更新，避免用户使用已回滚的路径。

功能与动机

原 PR #25483 将 Wan2.2 ModelOpt FP8/NVFP4 检查点切换到 NVIDIA 官方 Diffusers 仓库，并调整了量化配置默认值和加载逻辑。然而合并后 CI 测试（如 PR body 所示）始终失败，表明该路径存在环境依赖或配置兼容性问题。为保持主分支稳定，作者选择回滚，待问题定位后再重新提交。

实现拆解

该回滚通过一个 commit 反向应用了 #25483 的变更，主要涉及 7 个方面：

1. 量化配置恢复：将 swap_weight_nibbles 默认值从 True 改回 False，并恢复 from_config 中对 checkpoint_uses_packed_qkv 的 fallback 逻辑。
2. 构建工具恢复：build_modelopt_nvfp4_transformer.py 中 --swap-weight-nibbles 默认值恢复为统一 False，移除基于 pattern_preset 的特殊处理。
3. 加载器合并策略恢复：transformer_load_utils.py 中 _merge_modelopt_fp4_configs 的 swap_weight_nibbles 默认值从 True 改回 False，并调整优先级解除对 checkpoint_uses_packed_qkv 的连带。
4. 模型注册恢复：registry.py 中将 Wan2.2 T2V A14B 的 hf_model_paths 恢复为仅包含 Wan-AI/Wan2.2-T2V-A14B-Diffusers，移除 nvidia/Wan2.2-T2V-A14B-Diffusers-NVFP4。
5. 测试用例恢复：gpu_cases.py 中的 Wan2.2 测试从 --transformer-path 切换回直接 model_path，并恢复 run_consistency_check=True；testcase_configs.py 更新常量定义，移除新增常量，恢复旧环境变量。
6. 一致性阈值和工具函数恢复：移除新增的 NVFP4 一致性阈值，恢复 test_utils.py 中的超时值。

7. 文档恢复：quantization.mdx 中 Wan2.2 的加载方式从 --model-path 改回 --transformer-path，并调整检查点来源描述。

关键文件：

- python/sglang/multimodal_gen/runtime/layers/quantization/modelopt_quant.py（模块 量化配置；类别 source；类型 core-logic；符号 ModelOptFp4Config, ModelOptFp4Config.from_config, ModelOptFp4LinearMethod.process_weights_after_loading）：核心量化配置文件，回滚了 swap_weight_nibbles 默认值及加载逻辑，影响 NVFP4 权重处理。
- python/sglang/multimodal_gen/tools/build_modelopt_nvfp4_transformer.py（模块 构建工具；类别 source；类型 core-logic；符号 build_modelopt_nvfp4_transformer, _parse_args）：构建 NVFP4 transformer 的工具脚本，回滚了 swap_weight_nibbles 默认值和 pattern_preset 的相关逻辑。
- python/sglang/multimodal_gen/runtime/loader/transformer_load_utils.py（模块 加载器；类别 source；类型 core-logic；符号 _merge_modelopt_fp4_configs）：加载器配置合并逻辑，回滚了 swap_weight_nibbles 默认值，影响运行时 NVFP4 权重加载。
- python/sglang/multimodal_gen/registry.py（模块 模型注册；类别 source；类型 core-logic；符号 _register_configs）：模型注册入口，移除了 nvidia 官方 Wan2.2 NVFP4 路径，直接影响模型加载行为。
- python/sglang/multimodal_gen/test/server/gpu_cases.py（模块 测试用例；类别 test；类型 test-coverage）：GPU 测试用例，回滚了 Wan2.2 测试的模型加载方式和环境变量，影响 CI 覆盖。
- docs_new/docs/sglang-diffusion/quantization.mdx（模块 量化文档；类别 other；类型 core-logic）：扩散量化文档，回滚了 Wan2.2 检查点来源和加载方式描述。
- python/sglang/multimodal_gen/test/server/testcase_configs.py（模块 测试配置；类别 test；类型 test-coverage）：测试配置常量，回滚了 Wan2.2 相关的模型和环境变量常量定义。
- python/sglang/multimodal_gen/test/server/consistency_threshold.json（模块 一致性阈值；类别 test；类型 test-coverage）：一致性阈值配置文件，移除了新增的 NVFP4 阈值条目。
- python/sglang/multimodal_gen/test/test_utils.py（模块 测试工具；类别 test；类型 test-coverage）：测试工具函数，回滚了超时等相关默认值。

关键符号：ModelOptFp4Config.init, ModelOptFp4Config.from_config, ModelOptFp4LinearMethod.process_weights_after_loading, build_modelopt_nvfp4_transformer, _merge_modelopt_fp4_configs, _register_configs, _make_modelopt_ci_case

关键源码片段

python/sglang/multimodal_gen/runtime/layers/quantization/modelopt_quant.py

核心量化配置文件，回滚了 swap_weight_nibbles 默认值及加载逻辑，影响 NVFP4 权重处理。

```
# modelopt_quant.py ( 回滚后 )

class ModelOptFp4Config(ModelOptQuantConfig):
    """Config class for NVFP4."""

    def __init__(
        self,
        is_checkpoint_nvfp4_serialized: bool = False,
        group_size: int = None,
        exclude_modules: List[str] = None,
        packed_modules_mapping: Optional[Dict[str, List[str]]] = None,
        checkpoint_uses_packed_qkv: bool = False,
        swap_weight_nibbles: bool = False, # 回滚后的默认值
    ) -> None:
        ...

    @classmethod
    def from_config(cls, config: Dict[str, Any]) -> ModelOptFp4Config:
        group_size = None
        exclude_modules = []
        swap_weight_nibbles = False # 回滚后的初始化

        quant_method = config.get("quant_algo")
        if quant_method is not None:
            ...
            swap_weight_nibbles = config.get(
                "swap_weight_nibbles",
                config.get("checkpoint_uses_packed_qkv", False),
            )
        else:
            ...
            swap_weight_nibbles = quant_config.get(
                "swap_weight_nibbles",
                config.get(
                    "swap_weight_nibbles",
                    config.get("checkpoint_uses_packed_qkv", False),
                ),
            )
        ...
```

python/sglang/multimodal_gen/registry.py

模型注册入口，移除了 nvidia 官方 Wan2.2 NVFP4 路径，直接影响模型加载行为。

```
# registry.py ( 回滚后 )

def _register_configs():
    ...
    register_configs(
        sampling_param_cls=Wan2_2_T2V_A14B_SamplingParam,
```

```
        pipeline_config_cls=Wan2_2_T2V_A14B_Config,
        hf_model_paths=["Wan-AI/Wan2.2-T2V-A14B-Diffusers"], # 回滚后仅保留原始路径
    )
    ...
```

python/sglang/multimodal_gen/test/server/gpu_cases.py

GPU 测试用例，回滚了 Wan2.2 测试的模型加载方式和环境变量，影响 CI 覆盖。

```
# gpu_cases.py ( 回滚后 )

_make_modelopt_ci_case(
    "wan22_modelopt_fp8_t2v",
    model_path=DEFAULT_WAN_2_2_T2V_A14B_MODEL_NAME_FOR_TEST, # 直接使用模型名
    modality="video",
    sampling_params=MODELOPT_T2V_CI_sampling_params,
    extras=[], # 不再需要 --transformer-path
    run_consistency_check=True, # 恢复一致性检查
)
```

评论区精华

无主动 review 评论。PR body 仅包含 CI 状态和 bot 部署预览。作者 ch-wan 触发 /tag-and-rerun-ci 命令重新运行 CI，表明该 revert 为快速修复动作，未经过深入讨论。

- 暂无高价值评论线程

风险与影响

- 风险：回滚操作本身技术风险低，但存在以下潜在风险：
  - 原 PR 引入的依赖更新（如 transformers 库）可能未被完全回滚，若其他部分依赖新特性则可能破坏功能。
  - 恢复旧版量化默认值 swap_weight_nibbles=False 可能导致 NVFP4 权重加载错误（但之前已验证通过）。
  - 若原 PR 已被下游分支或文档引用，回滚可能导致同步问题。
  - 模型注册移除 nvidia 路径后，直接使用 NVIDIA 官方仓库的用户将无法加载，需手动指定 lmsys 仓库。
  - 影响：对用户而言，Wan2.2 ModelOpt 推理必须使用 lmsys 的 transformer 仓库，而非直接使用 NVIDIA 官方发布的 Diffusers 格式。对 CI 而言，B200 扩散测试用例减少 3 个，总检查点覆盖从 9 个降回 6 个，测试运行时间缩短。对团队开发而言，恢复了稳定的基线，后续可以基于健康的 CI 状态重新评估原 PR 的改进方案。
  - 风险标记：回滚可能导致功能缺失，未解决 Wan2.2 ModelOpt 加载兼容性问题，依赖第三方库版本可能已更新

关联脉络

- PR #25483 [codex] Update Wan2.2 ModelOpt CI checkpoints: 本 PR 回滚了 #25483 的变更，原 PR 切换了 Wan2.2 ModelOpt 检查点为 NVIDIA 官方仓库并可配置 swap_weight_nibbles 默认值。