PR #35349 完整报告

sgl-project/sglang

[VLM] Size the multimodal preprocessing pool by where preprocessing runs

合并时间：2026-08-27 16:32

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/35349

执行摘要

- 一句话：VLM 预处理池按路径动态调 worker 数
- 推荐动作：建议精读。该 PR 是“用真实测量驱动默认值决策”的典范：同一份代码在 H200 与 GB300 上得出相反的并发结论，最终以路径区分取代无条件默认。同时暴露并修复了 deepcopy 时机、闭包方法绑定、AST 审计等并发细节陷阱，对维护多模态推理栈的工程师有很高的借鉴价值。值得重点关注 base_processor.py 的路径决策函数与 executor.py 的懒克隆设计。

功能与动机

Preprocessing an image costs real CPU time on the tokenizer's event loop, so with a single worker a model's request throughput is capped at 1 / preprocess_time no matter how much GPU is idle. That cap is why PaddleOCR-VL saturated at 6.7 req/s on an H200 with GPU headroom to spare — and the failure is silent. The first version defaulted every processor to 2 workers, but a Blackwell check with heavy preprocessing overturned that default; the default is now path-aware instead of unconditional.

实现拆解

实现分五步推进：

1. 路径感知的 worker 数解析：在 python/sglang/srt/multimodal/processors/base_processor.py 中新增 _preprocessing_competes_with_the_scheduler() 判断预处理是否在调度器服务设备（GPU fast image processor）上运行，新增 _resolve_auto_mm_processor_worker_num() 根据路径返回 2（CPU）或 1（GPU）；auto_mm_processor_worker_num 类属性默认改为 None（跟随路径），supports_mm_processor_concurrency 默认改为 True（默认开启并发）。用户显式传入 --mm-processor-worker-num 时优先于路径决策。此改动影响所有多模态模型：Qwen-VL、PaddleOCR-VL 等声明 auto_mm_processor_worker_num = 2 的 GPU 路径模型被上限至 1 个 worker，CPU 路径仍尊重声明值。
2. 修复 worker 池 clone 时机与解析方式：python/sglang/srt/multimodal/processors/executor.py 中 MultimodalProcessorExecutor.__init__ 改为接收 resolve_processor 可调用对象，不再在 super().__init__() 内部提前 deepcopy；每次 worker 首次使用时才复制 self._resolve_processor()，保证克隆的是子类 __init__ 定制完成后的 processor。同时

base_processor.py 新增 _tokenizer_of() 统一构造与 clone 路径的 tokenizer 解析，避免 InternVL 这类以裸 tokenizer 作为 _processor 的处理器在 clone 时解析不一致。

3. 修复 Sarashina2Vision 的 kwarg 过滤器绑定：python/sglang/srt/multimodal/processors/sarashina2_vision.py 将原先闭合原始 image_processor 的 patched_preprocess 改为 _install_preprocess_kwarg_filter()，用 types.MethodType 绑定到实例，使 copy.deepcopy 重新绑定 __self__ 到每个 worker 自己的 clone，避免 worker 线程调用未打补丁的 _preprocess 抛 TypeError: unexpected keyword argument 'do_pad'。
4. 修复绕过注入 clone 的处理器并声明不参与池的处理器：ernie45_vl.py 和 midashenglm.py 的 process_mm_data override 改为接收 processor=None 并通过 _resolve_processor(processor) 解析注入的 clone；test_processor_async_call_sites.py 用 AST 审计所有 process_mm_data override，并显式列出 11 个不经过 process_and_combine_mm_data 的处理器（inkling、llava、whisper 等），强制新增处理器必须决定是否参与池。
5. 测试配套：test_mm_process_config.py 新增 CPU 路径 2 worker、GPU 路径 1 worker、显式参数优先、GPU 上限覆盖模型声明等测试，并 pin 住 server_args 的关键属性避免 MagicMock 全真值误入 CPU 分支；新增 test_processor_clone_isolation.py 验证 kwarg filter 在 deepcopy 后仍绑定到自身 clone；更新 test_processor_async_call_sites.py 和 test_paddleocr_vl_serving_defaults.py 适配新默认。整体共 62 个单元测试通过。

关键文件：

- python/sglang/srt/multimodal/processors/base_processor.py（模块 处理器基类；类别 source；类型 core-logic；符号 _tokenizer_of, _preprocessing_competes_with_the_scheduler, _resolve_auto_mm_processor_worker_num）：核心变更文件：新增 _tokenizer_of、_preprocessing_competes_with_the_scheduler、_resolve_auto_mm_processor_worker_num，将默认 worker 数改为路径感知，默认开启并发支持，并统一 tokenizer 解析。
- python/sglang/srt/multimodal/processors/sarashina2_vision.py（模块 处理器适配；类别 source；类型 core-logic；符号 _install_preprocess_kwarg_filter, _preprocess, patched_preprocess）：修复 Sarashina2Vision 的 kwarg 过滤器绑定：从闭包原始对象改为 types.MethodType 绑定实例，使 copy.deepcopy 能重新绑定到 worker 自己的 clone，避免并发下 TypeError。
- python/sglang/srt/multimodal/processors/executor.py（模块 预处理池；类别 source；类型 core-logic；符号 init）：worker 池 clone 时机修复：改为通过 callable 在首次使用时 deepcopy，避免克隆到 super().__init__() 期间的半成品 processor，并保证克隆成本只发生在 worker 首次使用时。
- test/registered/unit/managers/test_mm_process_config.py（模块 配置测试；类别 test；类型 test-coverage；符号 test_cpu_preprocessing_path_gets_two_workers, test_gpu_preprocessing_path_stays_at_one_worker, test_explicit_request_overrides_the_path_decision, test_gpu_path_caps_a_count_the_model_declared）：新增路径决策相关测试：CPU 路径 2 worker、GPU 路径 1 worker、显式参数优先、GPU 上限覆盖模型声明、clone tokenizer 解析一致等；并修复 MagicMock 全真值导致的路由误判。

- test/registered/unit/multimodal/test_processor_clone_isolation.py（模块 克隆隔离；类别 test；类型 test-coverage；符号 _NarrowImageProcessor, init, _preprocess, TestSarashina2PreprocessFilterSurvivesCloning）：新增测试文件，用 _NarrowImageProcessor 验证 Sarashina2Vision 的 kwarg filter 在 deepcopy 后仍绑定到自己的 clone，防止回归并发下的 TypeError。
- test/registered/unit/multimodal/test_processor_async_call_sites.py（模块 调用点测试；类别 test；类型 test-coverage；符号 test_default_worker_count_follows_the_preprocessing_path, _process_mm_data_overrides, test_overrides_take_the_worker_pools_processor_clone, test_processors_outside_the_worker_pool_are_declared）：新增 AST 审计测试 test_overrides_take_the_worker_pools_processor_clone，阻止 process_mm_data override 绕过注入 clone；并显式声明 11 个不参与 pool 的处理器。
- python/sglang/srt/multimodal/processors/ernie45_vl.py（模块 VLM 处理器；类别 source；类型 core-logic）：修复 process_mm_data override 硬编码 self._processor 导致 worker 线程共享同一 HF processor 的问题，改为解析注入的 clone。
- python/sglang/srt/multimodal/processors/midashenglm.py（模块 VLM 处理器；类别 source；类型 core-logic）：与 Ernie4.5-VL 相同的 clone 绕过修复，避免多 worker 下共享单例 HF processor。
- test/registered/unit/models/test_paddleocr_vl_serving_defaults.py（模块 服务默认值；类别 test；类型 test-coverage；符号 test_processor_opts_into_concurrency, test_processor_preprocesses_pages_concurrently, test_concurrency_opt_in_is_not_inherited_by_accident, test_io_worker_count_is_this_model_own）：适配 PaddleOCR-VL 并发默认行为变化，验证并发参与、页面并发预处理、IO worker 计数等模型专属默认值。

关键符号：_tokenizer_of, _preprocessing_competes_with_the_scheduler, _resolve_auto_mm_processor_worker_num, _install_preprocess_kwarg_filter, MultimodalProcessorExecutor.init, MultimodalProcessorExecutor._run, _process_mm_data_overrides

关键源码片段

python/sglang/srt/multimodal/processors/base_processor.py

核心变更文件：新增 _tokenizer_of、_preprocessing_competes_with_the_scheduler、_resolve_auto_mm_processor_worker_num，将默认 worker 数改为路径感知，默认开启并发支持，并统一 tokenizer 解析。

```
def _tokenizer_of(processor):
    """返回 HF processor 背后真正的 tokenizer。

    部分处理器（如 InternVL）直接以 tokenizer 作为 `_processor`，
    而不是一个包装了 tokenizer 的 processor。所有解析 tokenizer 的路径
    （构造、worker clone）都必须经过这里，保证 clone 与原始对象解析一致。
    """
    return processor.tokenizer if hasattr(processor, "tokenizer") else processor
```

```
class BaseMultimodalProcessor(ABC):
    # None 让 worker 数跟随预处理实际运行位置；
    # 模型若自己测量过最优值，可在此赋值一个数字。
    auto_mm_processor_worker_num = None
    auto_mm_io_worker_num = 4

    # 处理器只有在预处理非线程安全时才显式退出并发。
    # worker 池给每个线程单独 deepcopy 一份 HF processor 并注入，
    # 且唯一运行入口 `process_and_combine_mm_data` 解析注入的 clone，
    # 因此隔离性不依赖子类实现。
    supports_mm_processor_concurrency = True

    def __init__(self, hf_config, server_args, _processor, transport_mode, *args, **kwargs):
        ...
        # 统一走 _tokenizer_of，避免 clone 与 __init__ 解析结果分叉
        self._tokenizer = _tokenizer_of(self._processor)
        ...
        # 路径感知 worker 数：CPU 预处理 2 个，GPU 预处理 1 个；
        # 用户显式参数优先于路径决策。
        self.mm_processor_worker_num = (
            1 if skip_mm_pool
            else requested_mm_processor_worker_num
            or self._resolve_auto_mm_processor_worker_num()
        )
        ...
```

python/sglang/srt/multimodal/processors/sarashina2_vision.py

修复 Sarashina2Vision 的 kwarg 过滤器绑定：从闭包原始对象改为 types.MethodType 绑定实例，使 copy.deepcopy 能重新绑定到 worker 自己的 clone，避免并发下 TypeError。

```
import types

# Sarashina2Vision 的 remote-code `_preprocess` 只接受窄 kwarg 集合，
# 而 transformers 的 `preprocess` 会转发完整集合，多余的参数必须被丢弃。
_PREPROCESS_PARAMS = frozenset({
    "do_resize",
    "resample",
    "do_rescale",
    "rescale_factor",
    "do_normalize",
    "image_mean",
    "image_std",
    "do_convert_rgb",
    "data_format",
    "input_data_format",
})


def _install_preprocess_kwarg_filter(image_processor) -> None:
```

```
    """丢弃 Sarashina2Vision 的 `_preprocess` 无法接受的 kwargs。

    使用 `types.MethodType` 而不是闭包 `image_processor`，这样预处理 worker 的
    `copy.deepcopy` 会把 `__self__` 重新绑定到它自己的 clone，
    避免所有线程都路由回同一个共享对象。
    """
    unfiltered_preprocess = type(image_processor)._preprocess

    def _preprocess(self, *args, **kwargs):
        return unfiltered_preprocess(
            self,
            *args,
            **{k: v for k, v in kwargs.items() if k in _PREPROCESS_PARAMS},
        )

    image_processor._preprocess = types.MethodType(_preprocess, image_processor)
```

python/sglang/srt/multimodal/processors/executor.py

worker 池 clone 时机修复：改为通过 callable 在首次使用时 deepcopy，避免克隆到 super().__init__() 期间的半成品 processor，并保证克隆成本只发生在 worker 首次使用时。

```
class MultimodalProcessorExecutor:
    """在隔离的、线程本地的 processor clone 上运行处理器调用。"""

    def __init__(self, resolve_processor: Callable[[], Any], max_workers: int):
        # 按需解析而不是在构造时捕获：子类在 super().__init__() 返回后
        # 还会继续定制 `_processor`，此刻克隆会漏掉那些修改。
        self._resolve_processor = resolve_processor
        # 探针一次：若处理器不可克隆，启动时就回退到同步处理，
        # 而不是在 worker 线程内才失败。
        copy.deepcopy(resolve_processor())
        self._executor = concurrent.futures.ThreadPoolExecutor(
            max_workers=max_workers,
            thread_name_prefix="sglang-mm-processor",
        )
        ...

    def _run(self, function, *args, **kwargs) -> T:
        processor = self._worker_state.processor
        if processor is None:
            # 每次只克隆一个：克隆会读取共享 processor，
            # worker 路径也会读取 token 计数辅助函数，因此需要加锁。
            with self._clone_lock:
                processor = copy.deepcopy(self._resolve_processor())
            self._worker_state.processor = processor
        return function(*args, processor=processor, **kwargs)
```

评论区精华

PR 没有外部 review 评论，作者在关联 Issue 的评论中记录了三个关键发现与修复：

- “Two follow-ups pushed after review of what 'every processor benefits' actually requires. … Two — Ernie4.5-VL and MiDashengLM — hardcoded processor = self._processor … Both now resolve the injected clone.” 即并发默认开启后暴露 process_mm_data override 绕过 clone 的隐患，已修复。
- “CI caught a real one, now fixed in 721adcd. test_mm_process_config.py::test_parallel_workers_require_processor_support asked for two workers and asserted it got one — which only held because the base class did not support concurrency.” 测试改为显式 patch 掉 supports_mm_processor_concurrency。
- “The CPU suite found something better than a stale assertion — a real latent bug this PR would have activated. … _resolve_processor — which every pooled process_mm_data call goes through — did not: its clone branch reached straight for processor.tokenizer.” 现在通过 _tokenizer_of 统一。
- process_mm_data 覆写绕过注入 clone (correctness): 两个处理器均改为接受 processor=None 参数并通过 self._resolve_processor(processor) 解析注入的 clone；新增 AST 审计测试防止未来回归。
- 默认翻转导致旧测试断言失效 (testing): 测试改为显式 patch.object(BaseMultimodalProcessor, "supports_mm_processor_concurrency", False) 以保持原守卫语义，并新增 test_default_is_concurrent 验证新默认。
- clone 的 tokenizer 解析与 __init__不一致 (correctness): 新增 _tokenizer_of(processor) 统一构造与 clone 路径的 tokenizer 解析，并补充 test_clone_resolves_tokenizer_like_init 测试。

风险与影响

- 风险：
  1. 默认行为变更：supports_mm_processor_concurrency 从 False 翻转为 True，所有多模态处理器默认进入并发池，若某个处理器存在未覆盖的共享状态或非线程安全代码，可能在 worker 线程中触发竞态或崩溃。deepcopy 隔离降低了风险，但不为零。
  2. GPU 路径性能回退：GB300 上全页图像场景，第二个 worker 使吞吐下降 56%（9.30 → 4.02 req/s），TTFT 从 590 ms 升至 5755 ms；虽然默认已按路径收敛到 1 worker，但用户若显式设置 --mm-processor-worker-num=2 且未意识到路径差异，仍可能复现回退。
  3. clone 成本：MultimodalProcessorExecutor._run 每次新 worker 首次使用时 deepcopy 整个 HF processor（可能包含大权重），虽只发生一次，但探针 deepcopy 在 __init__ 阶段同步执行，可能增加启动时间。
  4. 测试脆弱性：test_mm_process_config.py 的 _make_processor 依赖 MagicMock 并需手动 pin server_args.disable_fast_image_processor 等属性，若未来新增读取的属性未 pin，测试可能静默走错分支。

5. 未声明处理器：dots_note_omni.py 等 11 个处理器显式声明不参与池，但如果后续维护者新增处理器时未更新 _NO_WORKER_POOL_ROUTE 集合，AST 审计测试会失败，这是有意的强制措施，但也可能造成 CI 噪声。 - 影响：影响范围覆盖所有使用多模态预处理的服务端进程。对 CPU 预处理路径（pil 后端、无 fast image processor 的模型、音频模型）吞吐提升显著（H200 +36%，GB300 +24%）；对 GPU 快速图像处理器路径，默认保持 1 worker，避免 Blackwell 上全页文档类重预处理的吞吐坍缩。行为变更明确：Qwen-VL 与 PaddleOCR-VL 在默认后端下从声明 2 worker 降为 1 worker，但实测数据表明这是正确取舍。同时修复了 Sarashina2Vision 在多 worker 下必然崩溃的隐藏 bug，以及 Ernie4.5-VL、MiDashengLM 的 clone 绕过问题。团队需在发布说明中强调默认并发开启，并建议重预处理负载在 CPU 路径下显式验证。 - 风险标记：默认行为变更，GPU 路径回退风险，并发线程安全依赖，测试用 MagicMock 脆弱性

关联脉络

- PR #35342 [VLM] Route every multimodal processor through the worker pool's call site: 本 PR 显式 Stacked on #35342，后者将 33 个同步 call site 统一改走 worker 池调用点，本 PR 才真正开启池并赋予路径感知的 worker 数。