PR #27297 完整报告

sgl-project/sglang

[diffusion] Optimize LingBot realtime transport and camera conditioning

合并时间：2026-06-05 16:00

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/27297

执行摘要

- 一句话：优化 LingBot 实时传输和相机条件化，延迟降低 10%
- 推荐动作：值得精读，尤其是相机条件器缓存设计中基于 source tensor identity 的键构建和条件判断，以及传输层将 delta-gzip 降级为 raw bytes 的权衡决策。测试覆盖充分，可作为性能优化 PR 的典范。

功能与动机

降低 LingBot 实时推理延迟，特别是相机条件化计算的重复开销和原始帧传输的编码延迟。基准测试显示这两个部分有显著优化空间，且测试中 CI OOM 问题需要修复。

实现拆解

1. 相机条件器 scale/shift 缓存机制：在 LingBotWorldCamConditioner 中提取 compute_scale_shift 方法，使 forward 可接受预计算值；在 CausalLingBotWorldTransformerBlock 中添加 _cam_conditioner_scale_shift 方法，基于 forward_batch 缓存键（含 data_ptr、shape、stride、dtype、device 和 _version）存储计算结果，仅当 sequence_shard 启用且 timestep>=0 时缓存，避免每 timestep 重复计算。
2. 输出传输优化：在 RawRGBRealtimeOutputAdapter 中，默认使用 raw bytes 而非 delta-gzip 传输 lossless raw RGB；当 payload 超过 64KB 时，拆分为 msgpack header 和独立 raw bytes，减小延迟；移除 _last_raw_rgb_frame 和 _last_event_id 状态，简化 _build_transport_payload，删除 delta-gzip 分支。
3. CI OOM 修复：在 realtime_video_api.py 中添加 _wait_for_server_warmup，确保 server warmup 完成后再接受 websocket 请求；在 lingbot_world_causal_denoising.py 中更新缓存时清理 lingbot_cam_conditioner 条目。
4. 测试配套：修改 test_realtime_output_transport.py 验证默认 raw payload 和分片行为；修改 test_lingbot_causal_denoising.py 覆盖缓存匹配、重用和跳过条件。

关键文件：

- python/sglang/multimodal_gen/runtime/models/dits/lingbot_world.py（模块 模型层；类别 source；类型 data-contract；符号 compute_scale_shift, _cam_conditioner_scale_shift, _should_cache_cam_conditioner, _prepare_cam_conditioner_scale_shifts）：核心模型文件，实现了相机条件器的缓存机制

，包含 compute_scale_shift、_cam_conditioner_scale_shift 等新方法

- python/sglang/multimodal_gen/test/unit/realtime/test_lingbot_causal_denoising.py（模块 测试；类别 test；类型 test-coverage；符号 test_lingbot_cam_conditioner_scale_shift_matches_forward, test_lingbot_cam_conditioner_cache_reuses_source_tensor, test_lingbot_cam_conditioner_cache_skips_non_sequence_shard, test_lingbot_cam_conditioner_cache_skips_single_ulysses_world）：为相机条件器缓存机制提供测试覆盖，验证缓存匹配、重用、跳过等场景
- python/sglang/multimodal_gen/test/unit/realtime/test_realtime_output_transport.py（模块 测试；类别 test；类型 test-coverage；符号 test_raw_rgb_realtime_output_adapter_uses_lossless_raw_payload_by_default, test_raw_rgb_realtime_output_adapter_offloads_default_lossless_payload_build, test_raw_rgb_realtime_output_adapter_sends_large_payload_separately, _unpack_frame_batch_messages）：测试传输层改动：默认 raw payload、分片消息解析、大 payload 拆分
- python/sglang/multimodal_gen/runtime/entrypoints/openai/realtime/realtime_output_adapter.py（模块 传输层；类别 source；类型 core-logic；符号 _pack_frame_batch_header, _build_transport_payload, RawRGBRealtimeOutputAdapter）：传输层核心逻辑简化，移除 delta-gzip，新增 _pack_frame_batch_header，支持大 payload 分片
- python/sglang/multimodal_gen/runtime/entrypoints/openai/realtime/realtime_video_api.py（模块 API 入口；类别 source；类型 entrypoint；符号 _wait_for_server_warmup）：入口层增加 server warmup 等待，修复 CI OOM
- python/sglang/multimodal_gen/runtime/pipelines_core/stages/model_specific_stages/lingbot_world/lingbot_world_causal_denoising.py（模块 模型层；类别 source；类型 data-contract）：在 context cache 更新时清理相机条件缓存，保持一致性
- python/sglang/multimodal_gen/test/server/gpu_cases.py（模块 测试；类别 test；类型 test-coverage）：测试配置调整，可能涉及硬件条件

关键符号：compute_scale_shift, _cam_conditioner_scale_shift, _should_cache_cam_conditioner, _prepare_cam_conditioner_scale_shifts, _pack_frame_batch_header, _build_transport_payload, RawRGBRealtimeOutputAdapter.send, _wait_for_server_warmup

关键源码片段

python/sglang/multimodal_gen/runtime/entrypoints/openai/realtime/realtime_output_adapter.py

传输层核心逻辑简化，移除 delta-gzip，新增 _pack_frame_batch_header，支持大 payload 分片

```
def _pack_frame_batch_header(header: RealtimeFrameBatchHeader) -> bytes:
    # 新增函数：将 header 单独编码为 msgpack，用于大 payload 分片时的首个小消息
    return msgspec.msgpack.encode(header)


def _build_transport_payload(
```

```
    transport_frames: list[bytes],
    *,
    content_type: str,
    metadata: dict[str, int | str],
    output_format: str | None,
    transport_quality: int | None,
    preview_max_width: int | None,
    # 移除了 reference_frame 和 event_id 参数
) -> _TransportPayload:
    # 简化：对于 raw RGB，直接拼接 transport_frames 为 raw_payload
    # 不再使用 delta-gzip
    if content_type == RAW_RGB_CONTENT_TYPE and transport_frames:
        raw_payload = b"".join(transport_frames)
        payload_metadata = {
            "raw_size": len(raw_payload),
            "encoding": RAW_LOSSLESS_OUTPUT_FORMAT,
        }
        # ... 原有 delta-gzip 分支已删除
```

评论区精华

PR 无 review 讨论，作者通过几次 /tag-and-rerun-ci 命令触发 CI 重跑以通过测试。

- 暂无高价值评论线程

风险与影响

- 风险：
  1. 缓存键依赖 tensor_version 和指针，若 tensor 被 in-place 修改但版本未更新可能导致缓存不一致，但测试已覆盖。
  2. 默认传输从 delta-gzip 改为 raw，可能增加带宽消耗，但 payload 构建延迟从 ~20ms 降至 ~1ms，权衡合理。
  3. _wait_for_server_warmup 引入 websocket 接受前的阻塞，可能增加首次请求延迟，但避免 OOM 更关键。
  4. 清理 lingbot_cam_conditioner 缓存在 context cache update 时执行，逻辑正确但需异常处理。 - 影响：对用户：LingBot 实时视频推理延迟降低约 10%，体验明显提升。对系统：传输层简化，去除 delta-gzip 依赖，降低了维护成本；raw payload 构建开销极小，但带宽略有增加（对于网络瓶颈场景仍需关注）。对团队：提供了在特定场景下计算缓存和传输协议优化的参考模式。 - 风险标记：缓存依赖 tensor 版本号，带宽权衡，新增 websocket 延迟点

关联脉络

- 暂无明显关联 PR