PR #30040 完整报告

sgl-project/sglang

[diffusion] feat: add LingBot realtime prompt, KV window, and lazy VAE controls

合并时间：2026-07-05 11:11

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/30040

执行摘要

- 一句话：LingBot 实时新增复合输入、KV 窗口和延迟 VAE
- 推荐动作：值得精读。交互式 KV 窗口的设计模式（重置默认值 + 按需调整）和复合输入原子性校验可复用。关注 causal_attention_cache 的 recent_window_tokens 参数，它是通用层的变化，可能被其他模型使用。

功能与动机

来自 PR body："Add LingBot realtime support for composite prompt and camera-action inputs in a single event." 此外，为了在实时视频生成中减少 prompt 更新导致的 cross-attention cache 全量清除开销、改善 camera-motion 下的运动连续性、以及降低 VAE 编码计算负担。

实现拆解

1. 复合输入事件解析与原子性更新（lingbot_world_realtime_adapter.py）：新增 _ingest_composite_input 方法，先验证所有 input_types 再依次调用 _ingest_camera_actions 和 _ingest_prompt，确保部分失败时不改变状态。
2. 交互式 KV 窗口控制（lingbot_world_causal_denoising.py）：新增 _apply_causal_cache_overrides 重写父类方法，先重置 cache 配置为模型默认值，再调用父类 override，最后通过 _sync_interactive_kv_cache_window 根据 camera-motion 动态调整 sliding_window_num_frames 和 sink_size。_chunk_has_camera_motion 和 _uses_interactive_kv_window 判断是否需要启用。
3. 动态 recent_window_tokens 支持（causal_attention_cache.py）：update_and_get_attention_kv 新增 recent_window_tokens 参数，提取 _visible_attention_kv 方法返回 sink tokens + 动态窗口的 KV 切片，为交互式窗口提供底层实现。
4. Prompt 更新标记与 cross-attention 重置：adapter 在 ingest prompt 时设置 LINGBOT_PROMPT_UPDATED_CONDITION 条件，denoising stage 据此决定是否重置 cross-attention cache。
5. 延迟 VAE 编码（lingbot_world.py）：preprocess_vae_encode 根据 SGLANG_LINGBOT_LAZY_VAE_ENCODE_BLACK_FRAMES 环境变量或配置，只编码首帧和指定数量的填充帧，后续用最后一帧的 latent 重复填充。

6. 环境变量与配置项：新增 interactive_kv_window_enable、interactive_kv_still_window、interactive_kv_moving_window、lazy_vae_encode_black_frames 等字段，默认关闭 / 零。

关键文件：

- python/sglang/multimodal_gen/runtime/pipelines_core/stages/model_specific_stages/lingbot_world/lingbot_world_causal_denoising.py（模块 去噪阶段；类别 source；类型 data-contract；符号 _chunk_has_camera_motion, _uses_interactive_kv_window, _interactive_kv_window_enabled, _apply_causal_cache_overrides）：核心业务逻辑：交互式 KV 窗口控制、prompt 更新检测、cache override 重置和同步
- python/sglang/multimodal_gen/runtime/entrypoints/openai/realtime/adapters/lingbot_world_realtime_adapter.py（模块 实时适配器；类别 source；类型 dependency-wiring；符号 receive_camera_control_event_payload, parse_camera_control_event_payload, receive_parsed_camera_control_event_payload, _ingest_camera_actions）：复合输入事件解析、原子性校验和分发，prompt 更新标记传递
- python/sglang/multimodal_gen/runtime/layers/kvcache/causal_attention_cache.py（模块 KV 缓存；类别 source；类型 core-logic；符号 _visible_attention_kv）：通用因果 KV 缓存新增 recent_window_tokens 动态窗口选择，影响所有使用该缓存的模型
- python/sglang/multimodal_gen/configs/pipeline_configs/lingbot_world.py（模块 配置；类别 source；类型 core-logic；符号 preprocess_vae_encode）：配置类新增 interactive_kv_window 和 lazy_vae_encode 字段，并实现延迟 VAE 编码逻辑
- 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_realtime_attention_cache_samples_sink_and_recent_window, test_lingbot_interactive_kv_window_config_default_disabled, test_lingbot_lazy_vae_encode_black_frames_env, test_lingbot_interactive_kv_window_samples_base_moving_and_still）：单元测试覆盖交互式 KV 窗口、动态窗口采样和延迟 VAE 环境变量
- python/sglang/multimodal_gen/test/unit/realtime/test_realtime_runtime.py（模块 测试；类别 test；类型 test-coverage；符号 test_lingbot_realtime_adapter_ingests_composite_input_event, test_lingbot_realtime_adapter_rejects_composite_input_atomically, test_lingbot_realtime_prompt_event_marks_crossattn_reset）：集成测试验证复合输入正常流程、原子性拒绝和 prompt 更新标记

关键符号：_chunk_has_camera_motion, _uses_interactive_kv_window, _interactive_kv_window_enabled, _apply_causal_cache_overrides, _reset_causal_cache_config_defaults, _sync_interactive_kv_cache_window, _effective_interactive_kv_cache_num_frames, _build_realtime_causal_cache_policy, receive_camera_control_event_payload, parse_camera_control_event_payload, receive_parsed_camera_control_event_payload, _ingest_camera_actions, _ingest_prompt, _validate_prompt_payload, _ingest_composite_input, _parse_composite_input_item, _visible_attention_kv, preprocess_vae_encode

关键源码片段

python/sglang/multimodal_gen/runtime/layers/kvcache/causal_attention_cache.py

通用因果 KV 缓存新增 recent_window_tokens 动态窗口选择，影响所有使用该缓存的模型

```
  def _visible_attention_kv(
      self,
      *,
      local_start_index: int,
      updated_local_end: int,
      attn_start_index: int,
      recent_window_tokens: int | None,
      cache_head_slice: slice | None = None,
  ) -> tuple[torch.Tensor, torch.Tensor]:
      """返回当前 attention 调用可见的 KV 切片。

      * 当 ``recent_window_tokens`` 为 ``None`` 时，返回标准滑动窗口
        ``[attn_start_index, updated_local_end)``；
      * 当启用 recent-window 选择时，返回 sink tokens 加上当前 chunk
        以及最多 recent_window_tokens 个历史帧的 KV：
        ``[0, sink_end) + [recent_start, updated_local_end)``；
        其中 ``recent_start = max(sink_end, local_start_index - recent_window_tokens)``。
      * 传入 ``0`` 则只保留 sink tokens 加当前 chunk。
      * 负值视为 ``None``（全窗口）。
      """
      if recent_window_tokens is None:
          # 标准滑动窗口行为：无动态截断
          visible_k = self.k[:, attn_start_index:updated_local_end]
          visible_v = self.v[:, attn_start_index:updated_local_end]
      else:
          # 动态窗口：保持 sink tokens 和最近的 recent_window_tokens
          sink_end = min(self.sink_tokens, updated_local_end)
          # 计算 recent 部分的起始位置，不能早于 sink_end
          recent_start = max(sink_end, local_start_index - recent_window_tokens)
          # 拼接 sink 部分和 recent 部分
          visible_k = torch.cat(
              [self.k[:, :sink_end], self.k[:, recent_start:updated_local_end]], dim=1
          )
          visible_v = torch.cat(
              [self.v[:, :sink_end], self.v[:, recent_start:updated_local_end]], dim=1
          )
      if cache_head_slice is not None:
          visible_k = visible_k[:, :, cache_head_slice, :]
          visible_v = visible_v[:, :, cache_head_slice, :]
      return visible_k, visible_v
```

评论区精华

1. 状态污染风险：gemini-code-assist[bot] 指出共享 stage 实例的 self.sink_size 和 self.sliding_window_num_frames 在请求间需重置，否则状态污染。作者添加 _reset_causal_cache_config_defaults() 在 _apply_causal_cache_overrides 开头恢复默认值。
2. 复合输入原子性：gemini-code-assist[bot] 建议先验证所有输入类型再更新状态，避免部分失败留下脏数据。作者重写 _ingest_composite_input 为两阶段：先解析验证所有 input_types，再统一更新状态。
3. 常量文件位置：mickqian 建议将 LingBot 常量从通用 realtime 目录移到模型专属目录。作者将 constants.py 移至 model_specific_stages/lingbot_world/ 下。
4. 文档完善：mickqian 两次要求为 _visible_attention_kv 方法和 update_and_get_attention_kv 的 recent_window_tokens 参数添加文档。作者均已完成。

- 状态污染风险：共享 stage 属性需重置 (correctness): 作者添加 _reset_causal_cache_config_defaults() 在 _apply_causal_cache_overrides 开头恢复默认值，确保请求隔离。
- 复合输入原子性：部分失败导致脏状态 (correctness): 作者重写 _ingest_composite_input 先解析验证所有 input_types，再统一更新状态，保证原子性。
- 常量文件位置：迁移到模型专属目录 (design): 作者将 constants.py 移至 model_specific_stages/lingbot_world/constants.py。
- 文档完善：_visible_attention_kv 和 recent_window_tokens 参数 (documentation): 作者添加了完整的 docstring 和注释，解释 None/0/ 正值的含义及返回的 token 范围。

风险与影响

- 风险：状态重置有效性：虽然已增加 _reset_causal_cache_config_defaults()，但若模型 arch_config 中缺少相关属性，重置可能失败，导致配置残留。并发安全：real time 路径假设请求串行处理，但若引入并行 pipeline，共享 stage 属性仍可能有竞态。性能分支：recent_window_tokens 为 None 时走原路径，开销为零；非 None 时增加一次 token 拼接和判断，影响较小。测试覆盖：新增单元测试覆盖主要逻辑，但缺少集成端到端测试。
- 影响：用户影响：使用 LingBot-World 实时服务的用户可以传入 composite_input 事件并在一次请求中同时更新 prompt 和 camera-actions；可开启交互式 KV 窗口获得更流畅的运动变化；延迟 VAE 编码减少初始化延迟。系统影响：新增环境变量（SGLANG_LINGBOT_ENABLE_INTERACTIVE_KV_WINDOW、SGLANG_LINGBOT_LAZY_VAE_ENCODE_BLACK_FRAMES），默认关闭，不影响现有行为。团队影响：需维护新的配置项和条件分支，但代码有测试保护。
- 风险标记：核心缓存层变更，状态管理已修复，配置项默认关闭

关联脉络

- 暂无明显关联 PR