PR #28926 完整报告

sgl-project/sglang

[diffusion]: enable RL rollout path for LTX-2.3 post-training

合并时间：2026-07-09 10:24

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/28926

执行摘要

- 一句话：为 LTX-2.3 添加 RL rollout 路径
- 推荐动作：PR 设计简洁、门控清晰，适合快速合并。建议关注点： 1) 确认 SGLang 自有 scheduler 的数值行为与 diffusers 版本一致； 2) 考虑为 rollout 路径添加单元测试，以防止未来代码重构时破坏； 3) _scheduler_step_kwargs 中的 batch.rollout 等属性访问可考虑使用 getattr 以增加健壮性（尽管当前 Req 数据类已有默认值）。

功能与动机

LTX-2/2.3 的联合音视频生成使用自定义 stage-1 guider 和 diffusers 的 FlowMatchEulerDiscreteScheduler，这些与现有 rollout 管线（/rollout/generate）不兼容。PR body 明确列出 rollout 需要的四项条件：标准 CFG（guidance_scale=1）、SDE 感知的 scheduler.step、rollout_trajectory_data 存活、以及 numpy 数组的正确序列化。

实现拆解

1. 配置层 (configs/sample/ltx_2.py)：在 LTX23SamplingParams.build_request_extra 中，当 self.rollout 为 True 时跳过注入 ltx2_stage1_guider_params，使 RL 使用标准 CFG 路径。
2. 调度器替换 (pipelines/ltx_2_pipeline.py)：将 diffusers 的 FlowMatchEulerDiscreteScheduler 导入替换为 SGLang 自有的 FlowMatchEulerDiscreteScheduler（位于 sglang.multimodal_gen.runtime.models.schedulers），该调度器支持 batch 参数以实现 SDE rollout 动态。
3. Denoising 阶段 (denoising.py)：新增 _scheduler_step_kwargs 方法，将 batch.generator、batch.eta、batch 等参数打包传递给 scheduler.step；在 denoising 循环中，当 batch.rollout 为 True 时，先手动设置 scheduler._step_index，然后使用 SDE 版本调用 scheduler.step。
4. 解码阶段 (decoding_av.py)：在 OutputBatch 构造时传入 rollout_trajectory_data=batch.rollout_trajectory_data，确保轨迹数据传递到 rollout API。
5. Rollout API 响应构建 (rollout_api.py)：在 _build_response 中，对 result.output[sample_idx] 先检查是否为 torch.Tensor，再调用 .contiguous()，避免在输出为 numpy 数组时出错。

6. 序列化工具 (utils.py)：在 _maybe_serialize 中添加对 numpy.ndarray 的处理，将其转换为 torch.Tensor 后复用已有的序列化逻辑。

关键文件：

- python/sglang/multimodal_gen/runtime/pipelines_core/stages/model_specific_stages/ltx_2/denoising.py（模块 扩散调度；类别 source；类型 core-logic；符号 _scheduler_step_kwargs）：核心变更：新增 _scheduler_step_kwargs 方法，并在 denoising 循环中根据 batch.rollout 切换 SDE 调度路径。
- python/sglang/multimodal_gen/runtime/entrypoints/post_training/rollout_api.py（模块 Rollout API；类别 source；类型 entrypoint）：修复 rollout 响应构建中对非 Tensor 输出的处理，避免 numpy 数组调用 .contiguous()。
- python/sglang/multimodal_gen/runtime/entrypoints/post_training/utils.py（模块 序列化工具；类别 source；类型 dependency-wiring）：扩展序列化函数以支持 numpy.ndarray，使解码后的视频帧能正确序列化。

关键符号：_scheduler_step_kwargs

关键源码片段

python/sglang/multimodal_gen/runtime/entrypoints/post_training/rollout_api.py

修复 rollout 响应构建中对非 Tensor 输出的处理，避免 numpy 数组调用 .contiguous()。

```
# 之前：out_i = result.output[sample_idx].contiguous()
# 现在：增加类型检查，兼容 numpy 输出
out_i = result.output[sample_idx]
if isinstance(out_i, torch.Tensor):
    out_i = out_i.contiguous()
serialized_generated_output = _maybe_serialize(out_i)
```

python/sglang/multimodal_gen/runtime/entrypoints/post_training/utils.py

扩展序列化函数以支持 numpy.ndarray，使解码后的视频帧能正确序列化。

```
def _maybe_serialize(obj: Any) -> Any:
    if isinstance(obj, torch.Tensor):
        # 原有 Tensor 序列化逻辑
        return {
            "__tensor__": True,
            "data": tensor_to_base64(obj),
            "shape": list(obj.shape),
            "dtype": str(obj.dtype),
        }
    # 新增 numpy 支持：先转为 Tensor 再序列化
    if isinstance(obj, np.ndarray):
        return _maybe_serialize(torch.from_numpy(obj))
    # 递归处理容器类型
    if isinstance(obj, dict):
        return {k: _maybe_serialize(v) for k, v in obj.items()}
```

```
    if isinstance(obj, (list, tuple)):
        return [_maybe_serialize(v) for v in obj]
    return obj
```

评论区精华

Gemini Code Assist 自动审查提出三点安全性建议：1）在 denoising.py 中直接访问 batch.rollout 可能在标准推理时引发 AttributeError，建议改用 getattr(batch, "rollout", False)；2）在 decoding_av.py 中对 batch.rollout_trajectory_data 同样建议使用 getattr；3）在 _scheduler_step_kwargs 中对 batch.generator 和 batch.eta 建议使用 getattr 默认值。作者 niehen6174 回应指出 generator 和 eta 是 Req 数据类声明的字段，带有默认值（None 和 0.0），因此直接访问是安全的。审核人 mickqian 最终批准了该 PR。

- 直接访问 batch.rollout 是否安全 (correctness): 作者回复确认 rollout 是 Req 数据类声明的字段，有默认值，直接访问安全。PR 保持原样。
- 直接访问 batch.generator 和 batch.eta (correctness): 作者回应指出这两个字段在 Req 数据类中有默认值（None 和 0.0），直接访问安全。
- batch.rollout_trajectory_data 安全访问 (correctness): 未收到作者回复，但该属性仅在 rollout 模式下设置，且访问点已受 batch.rollout 标志保护，风险较低。PR 合并时未修改。

风险与影响

- 风险：回归风险：低。所有 rollout 特定逻辑均通过 batch.rollout 标志门控，标准推理路径不受影响。兼容性风险：将 diffusers 的 scheduler 替换为 SGLang 自有的 scheduler，需确认该 scheduler 完全兼容原有接口，且未引入数值差异。数据契约风险：rollout_trajectory_data 在非 rollout 场景下可能未定义，虽然当前访问已通过标志保护，但若未来有其他路径调用解码阶段而未设置该属性，可能引发错误。类型安全风险：rollout_api.py 中新增的类型检查（isinstance(out_i, torch.Tensor)）消除了 numpy 数组调用 .contiguous() 的隐患。
- 影响：用户影响：对需要进行 RL 后训练的 LTX-2.3 用户提供直接支持，标准推理用户无感知。系统影响：仅修改 6 个文件，新增 32 行，变更量极小，不影响核心调度或推理性能。团队影响：为后续其他模型支持 rollout 路径提供了模式参考。
- 风险标记：缺少对 rollout_trajectory_data 的 getattr 保护

关联脉络

- PR #28527 [Diffusion][CPU] Adding AMX optimizations for CPU platform: 同为 diffusion 模型相关，涉及 ltx_2 管线的修改，但该 PR 聚焦 CPU AMX 优化，本 PR 聚焦 RL rollout 功能，不冲突。