PR #28105 完整报告

sgl-project/sglang

[Spec] Move `prepare_for_draft` to `EagleDraftWorkerBase`

合并时间：2026-06-13 08:42

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/28105

执行摘要

- 一句话：将 prepare_for_draft 移到 EagleDraftWorkerBase
- 推荐动作：此 PR 值得所有关注 speculative decoding 开发的工程师阅读。它清晰展示了如何通过移动方法实现职责分离这一常见重构模式。重点关注 base_spec_worker.py 中新增的 prepare_for_draft 方法和 duplicate_prefix_tail_to_draft_branches 函数。建议后续合并 gemini-code-assist 提议的类型注解改进，以进一步提升代码质量。

功能与动机

根据 PR 描述，构建 draft forward batch 是 worker 的行为责任，不应由 spec_info 数据类承载。将 prepare_for_draft 移到 worker 基类能使职责划分更清晰，有利于后续的维护和扩展。

实现拆解

1. 在 base_spec_worker.py 中新增 duplicate_prefix_tail_to_draft_branches 函数和 EagleDraftWorkerBase.prepare_for_draft 方法，并引入必要的类型导入（如 ReqToTokenPool, ModelRunner, EAGLEDraftCudaGraphRunner, EagleDraftInput）。
2. 从 eagle_info_v2.py 中删除上述函数及对应的导入语句（如 ModelRunner, assign_draft_cache_locs_contiguous, EAGLEDraftCudaGraphRunner 等），消除重复定义，并将 EagleDraftInputV2Mixin 简化。
3. 更新 eagle_worker_v2.py 和 multi_layer_eagle_worker_v2.py 中 draft 方法的调用，将 draft_input.prepare_for_draft(...) 替换为 self.prepare_for_draft(draft_input, ...)，并将 draft_input 作为第一个参数传递。
4. 所有逻辑维持原样，仅改变调用者和方法所处位置，测试套件已通过确认无回归。

关键文件：

- python/sglang/srt/speculative/base_spec_worker.py（模块 规约解码；类别 source；类型 core-logic；符号 duplicate_prefix_tail_to_draft_branches, prepare_for_draft）：核心变更文件，新增了 duplicate_prefix_tail_to_draft_branches 函数和 prepare_for_draft 方法，并更新了导入关系，是重构的目的地。
- python/sglang/srt/speculative/eagle_info_v2.py（模块 规约解码；类别 source；类型 core-logic；符号 duplicate_prefix_tail_to_draft_branches, prepare_for_draft）：原定义所在文件，被删除 prepare_for_draft 和 duplicate_prefix_tail_to_draft_branches，减少

了模块耦合。

- python/sglang/srt/speculative/eagle_worker_v2.py（模块 规约解码；类别 source；类型 core-logic）：调用者文件，改为调用 self.prepare_for_draft，反映了新职责归属。
- python/sglang/srt/speculative/multi_layer_eagle_worker_v2.py（模块 规约解码；类别 source；类型 core-logic）：另一个调用者文件，同样改为调用 self.prepare_for_draft，保持一致性。

关键符号：duplicate_prefix_tail_to_draft_branches, prepare_for_draft, prepare_for_draft_extend

评论区精华

gemini-code-assist 机器人提出了三点类型安全建议：

- 为 token_to_kv_pool 添加 KVCache 类型注解（base_spec_worker.py 第 11 行）
- 将 cuda_graph_runner 参数类型标记为 None 可选（base_spec_worker.py 第 174 行）
- 对can_cuda_graph使用bool()包装以确保严格布尔值（base_spec_worker.py第261行）

这些建议均未在本次 PR 中采纳，但指出了可进一步提升代码健壮性的方向。

- 添加 KVCache 类型注解 (style): 未采纳，当前未使用类型注解。
- 为 token_to_kv_pool 添加 KVCache 类型注解 (style): 未采纳。
- cuda_graph_runner 类型标记为可选 (style): 未采纳。
- 对 can_cuda_graph 使用 bool() 包装 (style): 未采纳。

风险与影响

- 风险：本次重构为纯代码移动，未涉及任何逻辑更改，且所有已有测试（包括 spec 解码相关测试）均已通过，回归风险极低。但需注意 prepare_for_draft 所在的基类 EagleDraftWorkerBase 现在同时承担 prepare_for_draft 和 prepare_for_draft_extend 两个核心方法，将来子类若覆盖其中之一可能引入不一致。此外，gemini-code-assist 建议的类型安全问题虽未解决，但实际运行时不受影响。
- 影响：本次变更为纯重构，对用户和系统行为无任何可见影响。团队内部可受益于更清晰的职责划分：worker 类完全掌控 draft 前向批处理的构建流程，spec_info 仅作为纯数据容器。后续若需扩展新类型的 draft worker，只需继承基类并重写相关方法即可。与 multi_layer_eagle_worker_v2.py 等子类的兼容性已通过测试验证。
- 风险标记：无行为变更，纯代码移动，测试已通过

关联脉络

- PR #28093 [Spec] Move draft-extend prep to EagleDraftWorkerBase; unify prepare_for_* names: 同一系列重构，将另一个核心方法 prepare_for_draft_extend 也移到了基类并统一命名，本 PR 是其延续。
- PR #28081 [refactor] Fold FrozenKVMTPCudaGraphRunner onto the shared DecodeCudaGraphRunner base: 同为 speculative decoding 子系统的重构，改进基类共享性，与本 PR 目标一致。