# PR #50574 完整报告

vllm-project/vllm

[Model Runner V2] Enable encoder token embedding

合并时间：2026-08-01 19:55

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/50574

## 执行摘要

- 一句话：MRV2 启用 token_embed 并接入 ColBERT 打分
- 推荐动作：值得精读。重点关注三点：一是「组合 LateInteractionRunner 而非把打分逻辑写进 PoolingRunner」的模块化决策，代码量虽小但扩展性好；二是生命周期钩子的放置顺序与缓存失效语义的精确定义——finish_requests 先于 preempted 合并、reset_encoder_cache 与 reset_mm_cache 的语义区分，都有测试锁定契约；三是 review 中「少改 MRV2 核心文件」的权衡过程，体现了核心模块演进时的改动面控制意识。

## 功能与动机

MRV2 是 vLLM 新一代模型执行器，pooling 支持按任务分步落地。PR body 明确说明：This PR enables the token_embed pooling task on Model Runner V2 for encoder-only models, and wires the existing LateInteractionRunner into the V2 pooling path so ColBERT-style scoring works under MRV2. 这是继 embed/classify（#48791）与 token_classify（#50293）之后的第三个任务，目的是补齐词级嵌入能力，并让检索型模型的 late-interaction 打分在 V2 下与 V1 行为一致。

## 实现拆解

实现分四步：

1. 任务白名单与筛选（vllm/v1/worker/gpu/pool/pooling_runner.py）：_SUPPORTED_TASKS 从 {embed, classify, token_classify} 扩展为 {embed, classify, token_embed, token_classify}，新增 _TOKEN_TASKS 常量统一标识 token 级任务；_get_enabled_tasks 对 decoder 模型统一剔除 _TOKEN_TASKS（原实现只剔除 token_classify）。原因：token 级 pooling 只在非分块的 encoder-only prefill 下才有语义。
2. LateInteractionRunner 组合集成（pooling_runner.py）：__init__ 创建 self.late_interaction_runner = LateInteractionRunner()；add_request 签名新增 req_id 参数，并在登记 pooling 状态时调用 register_request(req_id, pooling_params)，使 doc/query 关联按 req_id 记账；pool() 在 pooler 输出后追加 postprocess_pooler_output(...)，借助 finished_mask 决定 query 缓存回收时机。非 late-interaction 任务（embed/classify/token_classify）也会经过该后处理，但 LateInteractionRunner 对无关任务原样透传。

3. 生命周期挂钩（vllm/v1/worker/gpu/model_runner.py）：finish_requests 在 preempted_req_ids 合并之前调用 on_requests_finished(finished_req_ids)，注释明确被抢占的 doc 保留 query-use 预留直到重新调度；reset_encoder_cache 追加 pooling_runner.clear()，保证权重重载后失效的 query 嵌入缓存被清空；add_requests 给 pooling_runner.add_request 透传 req_id。review 后按 yewentao256 的建议删除了 reset_mm_cache 中的 clear()，因为多模态缓存重置与打分缓存无关。
4. 测试配套：tests/models/language/pooling/test_splade_sparse_pooler.py 新增 4 个用例，覆盖任务筛选（test_pooling_runner_supports_encoder_token_embedding、test_pooling_runner_filters_decoder_token_embedding）、aborted doc 释放（test_pooling_runner_releases_aborted_late_interaction_doc）与缓存重置语义锁定（test_encoder_cache_reset_clears_late_interaction_state）；tests/models/language/pooling/test_colbert.py 的 test_colbert_hf_comparison 参数化 (backend, use_v2)，bert/modernbert 走 V2、jina/lfm2 走 V1，通过 monkeypatch.setenv("VLLM_USE_V2_MODEL_RUNNER", ...) 强制路径并断言配置实际生效。

关键文件：

- vllm/v1/worker/gpu/pool/pooling_runner.py（模块 池化执行；类别 source；类型 core-logic；符号 PoolingRunner.init, PoolingRunner._get_enabled_tasks, PoolingRunner.add_request, PoolingRunner.pool）：核心源码：扩展 _SUPPORTED_TASKS 支持 token_embed，新增 _TOKEN_TASKS 常量，把 LateInteractionRunner 组合进 PoolingRunner 并在 pool() 中完成后处理接线，同时新增 on_requests_finished/clear 生命周期转发方法。
- vllm/v1/worker/gpu/model_runner.py（模块 模型执行；类别 source；类型 lifecycle-hook；符号 GPUModelRunner.finish_requests, GPUModelRunner.reset_encoder_cache, GPUModelRunner.add_requests）：生命周期钩子接入点：finish_requests 在 preempted 合并前通知 late-interaction 清理，reset_encoder_cache 触发 clear()，add_requests 透传 req_id；review 讨论集中在这些改动的必要性与放置位置。
- tests/models/language/pooling/test_splade_sparse_pooler.py（模块 池化测试；类别 test；类型 test-coverage；符号 test_pooling_runner_releases_aborted_late_interaction_doc, test_encoder_cache_reset_clears_late_interaction_state, test_pooling_runner_supports_encoder_token_embedding, test_pooling_runner_filters_decoder_token_embedding）：新增 4 个测试用例，覆盖任务筛选、aborted doc 释放与缓存重置语义，其中 test_encoder_cache_reset_clears_late_interaction_state 精确锁定了 review 讨论得出的 reset 语义。
- tests/models/language/pooling/test_colbert.py（模块 检索打分；类别 test；类型 test-coverage；符号 test_colbert_hf_comparison）：将 HF 对比测试参数化 (backend, use_v2)，bert/modernbert 在 V2 下验证 token 嵌入与 HF 参考一致，并通过环境变量断言确保 V2 路径真实生效。

关键符号：PoolingRunner.init, PoolingRunner._get_enabled_tasks, PoolingRunner.add_request, PoolingRunner.pool, PoolingRunner.on_requests_finished, PoolingRunner.clear, GPUModelRunner.finish_requests,

GPUModelRunner.reset_encoder_cache, GPUModelRunner.add_requests, LateInteractionRunner.postprocess_pooler_output, LateInteractionRunner.register_request

## 关键源码片段

### vllm/v1/worker/gpu/pool/pooling_runner.py

核心源码：扩展 _SUPPORTED_TASKS 支持 token_embed，新增 _TOKEN_TASKS 常量，把 LateInteractionRunner 组合进 PoolingRunner 并在 pool() 中完成后处理接线，同时新增 on_requests_finished/clear 生命周期转发方法。

```
# vllm/v1/worker/gpu/pool/pooling_runner.py（整理后实现片段）
# token 级任务统一收敛到 _TOKEN_TASKS，decoder 模型据此整体剔除
_TOKEN_TASKS: frozenset[PoolingTask] = frozenset({"token_embed", "token_classify"})


class PoolingRunner:
    def __init__(self, model: nn.Module, vllm_config: VllmConfig) -> None:
        ...
        self.pooling_params: dict[int, PoolingParams] = {}
        self.pooling_states: dict[int, PoolingStates] = {}
        self.prompt_token_ids: dict[int, torch.Tensor] = {}
        # 组合 LateInteractionRunner：让池化路径复用既有的 ColBERT query 缓存
        # 与 late-interaction 后处理，避免把打分逻辑硬编码进 PoolingRunner
        self.late_interaction_runner = LateInteractionRunner()

    @staticmethod
    def _get_enabled_tasks(model_config: ModelConfig) -> frozenset[PoolingTask]:
        # token 级 pooling 只允许在非分块的 encoder-only prefill 上使用；
        # decoder 模型（如生成式模型）一律剔除 token_embed / token_classify
        if model_config.attn_type == "encoder_only":
            return _SUPPORTED_TASKS
        return _SUPPORTED_TASKS - _TOKEN_TASKS

    def add_request(
        self,
        req_id: str, # 新增参数：late-interaction 状态按 req_id 记账
        req_index: int,
        pooling_params: PoolingParams,
        prompt_token_ids: list[int],
    ) -> None:
        ...
        self.model.pooler.get_pooling_updates(task).apply(pooling_params)
        # req_id 是 doc-query 关联与引用计数的键，必须在此注册
        self.late_interaction_runner.register_request(req_id, pooling_params)
        self.pooling_params[req_index] = pooling_params
        ...

    def pool(
```

```
        self,
        hidden_states: torch.Tensor,
        input_batch: InputBatch,
        req_states: RequestState,
    ) -> tuple[PoolerOutput, list[bool]]:
        ...
        pooler_output = self.model.pooler(hidden_states, pooling_metadata)
        # 从 pooling cursor 得到本步内各请求是否到达终点（finished_mask）
        finished_mask = pooling_metadata.get_pooling_cursor().is_finished().tolist()
        # late-interaction 后处理：只有注册过 query 缓存的请求会被改写输出，
        # 其余任务（embed/classify 等）原样透传；finished_mask 用于决定
        # query 缓存何时可以安全释放
        pooler_output = self.late_interaction_runner.postprocess_pooler_output(
            raw_pooler_output=pooler_output,
            pooling_params=pooling_metadata.pooling_params,
            req_ids=input_batch.req_ids,
            finished_mask=finished_mask,
        )
        return pooler_output, finished_mask
```

## vllm/v1/worker/gpu/model_runner.py

生命周期钩子接入点：finish_requests 在 preempted 合并前通知 late-interaction 清理，reset_encoder_cache 触发 clear()，add_requests 透传 req_id；review 讨论集中在这些改动的必要性与放置位置。

```
# vllm/v1/worker/gpu/model_runner.py（整理后实现片段）
def finish_requests(self, scheduler_output: SchedulerOutput) -> None:
    finished_req_ids = scheduler_output.finished_req_ids
    if self.pooling_runner is not None:
        # 必须在 preempted 合并之前通知：被抢占的文档请求仍然保留
        # 其 query-use 预留（引用计数），等重新调度时继续使用；
        # 过早清理会造成状态错乱，这是顺序敏感的生命周期契约
        self.pooling_runner.on_requests_finished(finished_req_ids)
    preempted_req_ids = scheduler_output.preempted_req_ids
    if preempted_req_ids:
        finished_req_ids = finished_req_ids.union(preempted_req_ids)
    for req_id in finished_req_ids:
        self._remove_request(req_id)


def reset_encoder_cache(self) -> None:
    # 权重重载会让缓存的 query 嵌入失效，必须同步清空 late-interaction
    # 状态，防止用旧权重算出的嵌入继续参与打分；多模态缓存重置
    #  （reset_mm_cache）与打分缓存无关，因此不清空（review 后收敛）
    if self.encoder_cache is not None:
        self.encoder_cache.reset_encoder_cache()
    if self.pooling_runner is not None:
        self.pooling_runner.clear()
```

### tests/models/language/pooling/test_splade_sparse_pooler.py

新增 4 个测试用例，覆盖任务筛选、aborted doc 释放与缓存重置语义，其中 test_encoder_cache_reset_clears_late_interaction_state 精确锁定了 review 讨论得出的 reset 语义。

```
# tests/models/language/pooling/test_splade_sparse_pooler.py（整理后片段）
def test_encoder_cache_reset_clears_late_interaction_state() -> None:
    # 缓存的 query 嵌入只被权重重载失效（经由 reset_encoder_cache 入口）；
    # 多模态缓存重置与之无关，必须保留这些状态——这个断言锁定了
    # review 讨论中得出的缓存失效语义
    runner = GPUModelRunner.__new__(GPUModelRunner)
    runner.encoder_cache = MagicMock()
    runner.pooling_runner = MagicMock()

    runner.reset_mm_cache()
    runner.encoder_cache.reset_mm_cache.assert_called_once_with()
    runner.pooling_runner.clear.assert_not_called()

    runner.reset_encoder_cache()
    runner.encoder_cache.reset_encoder_cache.assert_called_once_with()
    runner.pooling_runner.clear.assert_called_once_with()
```

## 评论区精华

核心讨论围绕「尽量少改 MRV2 核心文件」展开：

- yewentao256 在 model_runner.py 的 reset_mm_cache diff 上提出：we should try to edit MRv2 file less, is there any chance we can move to other files?
- taneem-ibrahim 回应：三处改动（add_requests、finish_requests、reset_*）都是只有 runner 才能观察到的生命周期事件；并主动提议删掉 reset_mm_cache 中的 clear()（-2 行），因为 reload_weights 会同时调用两个 reset，权重更新覆盖不变。
- yewentao256 认可：Sounds good! Thanks。

最终实现（commit 552d6bf）正是该讨论的落地：只保留 reset_encoder_cache 的清理，且新增测试同时断言两个方向（mm 缓存重置不清、encoder 缓存重置清）。该线程展示了 MRV2 演进过程中对核心文件改动面的敏感度权衡。

- MRV2 核心文件最小改动与生命周期钩子归属 (design): 接受折中方案：保留 model_runner.py 中的生命周期钩子，但只保留 reset_encoder_cache 的清理，移除 reset_mm_cache 中的 clear()；新增测试同时断言两个方向的语义。

## 风险与影响

- 风险：
  1. 生命周期时序敏感（model_runner.py::finish_requests）：on_requests_finished 必须在 preempted_req_ids 合并之前执行，否则被抢占 doc 的 query 引用计数会被提前释放，重调度后状态错乱；现有测试只覆盖 aborted 场景，未直接覆盖 preempted 场景。

1. 缓存失效依赖单一入口（model_runner.py::reset_encoder_cache）：query 嵌入缓存只在 reset_encoder_cache 时清空，依赖 reload_weights 同时调用两个 reset 的前提；未来若新增其它权重更新路径，需确保同样经过该入口，否则会用陈旧嵌入打分。测试 test_encoder_cache_reset_clears_late_interaction_state 已锁定该契约，但不会阻止新增入口。
2. add_request 签名变更（新增 req_id）：内部 API 变化，当前调用方仅 GPUModelRunner.add_requests，已同步；若外部测试或插件直接调用需适配。
3. V2 对比覆盖有限：HF 对比仅对 bert/modernbert 后端启用 V2，Jina/LFM2 仍走 V1；Jina/LFM2 在 V2 下的行为未直接验证，但强制 V2 的 14 个 online score/rerank 用例全部通过，缓解了该风险。
4. 本 PR 未包含文档改动，MRV2 支持任务矩阵尚无 token_embed 的说明，存在轻度可发现性风险。 - 影响：对用户：ColBERT 系列模型（如 answerdotai/answerai-colbert-small-v1、lightonai/GTE-ModernColBERT-v1）在 VLLM_USE_V2_MODEL_RUNNER=1 下可直接使用 token_embed、late-interaction score/rerank 接口，输出与 HuggingFace 参考在 1e-2 容差内一致；V1 路径完全不受影响。

对系统：PoolingRunner 成为 V2 pooling 后处理的统一入口，后续新增其它 late-interaction 类型任务可复用该「组合 LateInteractionRunner」的模式；finish_requests/reset_encoder_cache 的生命周期钩子成为 pooling 状态管理的固定接点。

对团队：MRV2pooling四任务已完成三个（embed/classify、token_classify、token_embed），与 #52425 等同期 PR 共同推动 MRV2 收敛 pooling 场景；测试中 VLLM_USE_V2_MODEL_RUNNER 参数化 + 断言生效的做法，成为 MRV2 回归测试的标准模式。

- 风险标记：生命周期钩子顺序敏感，缓存失效依赖 reset_encoder_cache，add_request 签名变更，V2 对比覆盖仅部分后端

## 关联脉络

- PR #48791 [Model Runner V2] Enable embed/classify pooling: PR body 列为 MRV2 pooling 启用的第一步（已合并）；本 PR 是同一功能线的第三步，共享 PoolingRunner 的任务筛选与启用逻辑。
- PR #50293 [Model Runner V2] Enable token_classify pooling: PR body 列为第二步（已合并）；token_classify 与 token_embed 同属 _TOKEN_TASKS，共同走 encoder-only 校验路径，本 PR 复用了其 _get_enabled_tasks 模式。
- PR #52425 [ModelRunner v2] Support Transformers pooling model: 同期 MRV2 pooling 相关 PR，扩展 MRV2 对池化模型的支持面，与本 PR 共同推进 MRV2 pooling 场景收敛。