

PR #56573 完整报告

vllm-project/vllm

[Pooling] Report actual input token usage for scoring APIs

合并时间: 2026-09-13 21:20

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/56573>

执行摘要

- 一句话: `/score` 与 `/rerank` 改用引擎实际输入统计 token usage
- 推荐动作: 值得一读, 但属于小而精的修复而非架构级变更: 重点看 `_build_response` 如何把「计量口径」从后处理产物切换到引擎真实输入, 以及为什么 `cross-encoder` 期望值不动、`bi-encoder/late-interaction` 期望值整体减 1。审阅时建议额外关注两点: 一是 `assert ctx.engine_inputs is not None` 是否应替换为显式错误处理, 以覆盖 `flash` `late-interaction` 与多模态路径; 二是是否需要为 `/score`、`/rerank` 的 usage 增加独立单测 (对齐 PR body 中 tokenizer 对照脚本), 避免未来再次回归到后处理累加。

功能与动机

PR body 指出: `/score` 与 `/rerank` 原先基于后处理产物

`PoolingRequestOutput.prompt_token_ids` 计算 usage。对 `bi-encoder` 和 `late-interaction` 模型, 后处理会为每个 document 复制同一份共享 query, 并插入一个合成分隔符 token, 于是 usage 被实现细节放大——「Those implementation details inflated reported token usage for 1:N requests even though the engine encoded the query only once」。复现数据给出确定性偏差公式 $(3 - 1) * 5 \text{ query tokens} + 3 \text{ synthetic separator tokens}$, 即 tokenizer 口径真实总量 21, 而 main 上两个端点都返回 34。这种偏差会让下游基于 usage 的计费、配额、可观测性指标与实际编码量脱节, 因此需要把口径绑定到「引擎真正提交了什么」。

实现拆解

1. 统计入口统一上移到 `_build_response` (`vllm/entrypoints/pooling/scoring/serving.py`): 旧实现在 `_request_output_to_score_response` 与 `_request_output_to_rerank_response` 内各自 `num_prompt_tokens = 0` 并在循环中累加; 新实现改为在 `_build_response` 内先 `assert ctx.engine_inputs is not None`, 再对 `ctx.engine_inputs` 逐项调用 `self._extract_prompt_len(engine_input["prompts"])` 求和, 得到一次请求级别的真实输入用量。
2. 以显式参数注入两个响应构造函数: 两个函数签名都新增 `num_prompt_tokens: int`, 并删除循环内的 `prompt_token_ids = final_res.prompt_token_ids` 与 `num_prompt_tokens += len(prompt_token_ids)`, 改为把入参直接填入 `UsageInfo` / `RerankUsage`。这样统计口径只依赖引擎提交的渲染输入, 不再受后处理是否复制 query、是否插入分隔符影响; `_extract_prompt_len` 为既有辅助方法, 本 PR 未改动其实现。

3. 业务路径保持不变：ScoreResponseData、RerankResult 的构造、按 relevance 排序与 top_n 截断都未触碰，因此只有 usage 数值变化，分数与输出顺序不变；cross-encoder 的 engine_inputs 本来就是 query、document 拼接后的成对输入，累加结果与旧口径一致。
4. 测试配套 (tests/models/language/pooling/test_max_tokens_per_doc.py) :
RERANK_CONFIGS 中 bi-encoder (intfloat/multilingual-e5-small) 与 late-interaction (answerdotai/answerai-colbert-small-v1) 的四个期望值整体下调 1，例如 e5-small 的 without_truncated_prompt_tokens 从 (285, 286) 变为 (284, 285)，ColBERT 从 285/155/155/25 变为 284/154/154/24；cross-encoder (Qwen3 reranker) 与 Jina reranker 期望值不变，反证口径修正只作用于共享 query 的场景。assert_prompt_tokens 的单值 / 区间容忍逻辑未改。

关键文件：

- vllm/entrypoints/pooling/scoring/serving.py (模块 评分服务；类别 source；类型 core-logic；符号 _build_response, _request_output_to_score_response, _request_output_to_rerank_response)：唯一的源码改动文件，承载 usage 口径从后处理产物切换到引擎真实输入的全部逻辑，包括 _build_response 的求和与两个响应构造函数的签名变更。
- tests/models/language/pooling/test_max_tokens_per_doc.py (模块 截断测试；类别 test；类型 test-coverage；符号 RERANK_CONFIGS, TestConfig, assert_prompt_tokens)：唯一测试配套文件，通过下调 bi-encoder 与 late-interaction 的期望 prompt token 数，锁定新的 usage 口径；cross-encoder 与 Jina reranker 期望值不变，构成对照。

关键符号：_build_response, _request_output_to_score_response,
_request_output_to_rerank_response

关键源码片段

vllm/entrypoints/pooling/scoring/serving.py

唯一的源码改动文件，承载 usage 口径从后处理产物切换到引擎真实输入的全部逻辑，包括 _build_response 的求和与两个响应构造函数的签名变更。

```
def _build_response(
    self,
    ctx: ScoringServeContext,
) -> JSONResponse:
    final_res_batch = ctx.final_res_batch
    request_id = ctx.request_id
    created_time = ctx.created_time
    model_name = ctx.model_name

    # 本次修复的核心：usage 不再从后处理后的 final_res_batch 反推，
    # 而是直接统计真正提交给引擎的渲染输入 engine_inputs。
    # 对 bi-encoder / late-interaction，query 只作为独立输入出现一次，
    # 不会被 1:N 的文档循环重复计数，也不含后处理插入的合成分隔符。
    assert ctx.engine_inputs is not None
    num_prompt_tokens = sum(
```

```

        self._extract_prompt_len(engine_input["prompts"])
        for engine_input in ctx.engine_inputs
    )

    if isinstance(ctx.request, ScoreRequest):
        return self._request_output_to_score_response(
            final_res_batch,
            request_id,
            created_time,
            model_name,
            num_prompt_tokens, # 真实用量改为显式入参，取代函数内的循环累加
        )
    elif isinstance(ctx.request, RerankRequest):
        return self._request_output_to_rerank_response(
            final_res_batch,
            request_id,
            model_name,
            ctx.request.documents,
            ctx.request.top_n if ctx.request.top_n > 0 else len(final_res_batch),
            num_prompt_tokens,
        )
    else:
        raise ValueError(f"Invalid {self.request_id_prefix} request type")

def _request_output_to_rerank_response(
    self,
    final_res_batch: list[PoolingRequestOutput],
    request_id: str,
    model_name: str,
    documents: ScoreInput | list[ScoreInput],
    top_n: int,
    num_prompt_tokens: int, # 新增：由 _build_response 基于 engine_inputs 计算
) -> JSONResponse:
    if not isinstance(documents, list):
        documents = [documents]

    results: list[RerankResult] = []
    for idx, final_res in enumerate(final_res_batch):
        classify_res = ScoringRequestOutput.from_base(final_res)

        document = documents[idx]
        if isinstance(document, str):
            rerank_document = RerankDocument(text=document)
        else:
            rerank_document = RerankDocument(
                multi_modal=document.get("content", [])
            )

        results.append(

```

```

        RerankResult(
            index=idx,
            document=rerank_document,
            relevance_score=classify_res.outputs.score,
        )
    )
    # 旧实现会在这里执行 num_prompt_tokens += len(final_res.prompt_token_ids);
    # 该字段来自后处理结果，N 个文档会重复计入同一份 query 以及合成分隔符。

# 排序与 top_n 截断逻辑保持原样，本次只改 usage 口径，不改打分为
results.sort(key=lambda x: x.relevance_score, reverse=True)
if top_n < len(documents):
    results = results[:top_n]

response = RerankResponse(
    id=request_id,
    model=model_name,
    results=results,
    usage=RerankUsage(
        total_tokens=num_prompt_tokens,
        prompt_tokens=num_prompt_tokens,
    ),
)
return JSONResponse(content=response.model_dump())

```

tests/models/language/pooling/test_max_tokens_per_doc.py

唯一测试配套文件，通过下调 bi-encoder 与 late-interaction 的期望 prompt token 数，锁定新的 usage 口径；cross-encoder 与 Jina reranker 期望值不变，构成对照。

3. bi-encoder: usage 口径切换为「引擎真实输入」后，期望总量整体下移 1，
括号区间仍用于容忍 tokenizer 在 CI / 本地缓存下的边界差异。

```

TestConfig(
    model="intfloat/multilingual-e5-small",
    args=[
        "--enforce-eager",
        "--max-model-len",
        "512",
        "--trust-remote-code",
    ],
    without_truncated_prompt_tokens=(284, 285),
    with_max_tokens_per_query_prompt_tokens=(154, 155),
    with_max_tokens_per_doc_prompt_tokens=154,
    with_max_tokens_per_query_and_doc_prompt_tokens=24,
),

```

4. late-interaction: 同样属于共享 query 的模型，四个期望值同步减 1

```

TestConfig(
    model="answerdotai/answerai-colbert-small-v1",
    args=[
        "--enforce-eager",
    ],
)

```

```

"--max-model-len",
"512",
"--trust-remote-code",
],
without_truncated_prompt_tokens=284,
with_max_tokens_per_query_prompt_tokens=154,
with_max_tokens_per_doc_prompt_tokens=154,
with_max_tokens_per_query_and_doc_prompt_tokens=24,
),
# 5. jinaai/jina-reranker-v3 与 Qwen3 cross-encoder 的期望值未变,
# 说明口径修正只影响后处理会复制 query 的模型族

```

评论区精华

该 PR 的评审讨论非常轻量，没有逐行技术交锋：

- claude[bot] 明确表示自动化 review 被跳过：「This pull request is from a fork — automated review is disabled.」——fork PR 需要维护者手动 @claude review 才会触发一次性评审，本 PR 未触发。
- 维护者 yewentao256 给出 APPROVED 并评论「LGTM, thanks for the work!」，随后完成合并。
- Issue 评论仅包含 CI 触发流程（作者 /ci run, github-actions[bot] 返回 Buildkite CI #88545 链接），没有设计层面的质疑或返工要求。

也就是说，评审结论主要建立在 PR body 自带的复现脚本、tokenizer 对照数据（34 → 21）以及作者在 H100 上的端到端手测（e5-small、ColBERT flash/vanilla 路径、TinyBERT cross-encoder）之上，而非 code review 讨论。

- fork PR 自动 review 被禁用，由维护者单点批准 (other): 变更以作者自测数据（tokenizer 对照 34 → 21、三个模型端到端验证）为依据被接受合并，没有产生设计层面的争议或返工。

风险与影响

- 风险：
 1. 断言做输入校验的健壮性：assert ctx.engine_inputs is not None 依赖 -O 下不被剥离的假设；若某些入口（例如 flash late-interaction 分支或未来新增的 serving 子类）未填充 engine_inputs，会直接抛 AssertionError 导致请求 500，而在 python -O 下更会退化为对 None 迭代的 TypeError。更稳妥的做法是显式判空并返回结构化错误。
 2. 对外可见的计量语义变更：/score 的 prompt_tokens / total_tokens 与 /rerank 的 RerankUsage 在 1:N 场景下会显著变小（示例 34 → 21）。依赖该字段做计费、配额或日志对账的下游必须同步口径，否则会出现「同一个请求用量突然下降」的对账差异。
 3. 对 engine_input["prompts"] 结构与 _extract_prompt_len 实现的隐式依赖：多模态 document（RerankDocument(multi_modal=document.get("content", []))）的渲染输入结构与纯文本不同，本 PR 未新增多模态 scoring 的 usage 用例，若 _extract_prompt_len 对 dict 形态 prompt 处理不周，可能出现少算而非报错，属静默偏差。

4. 回归保护偏弱：没有新增专门的 usage 单元测试（如 tests/entrypoints/pooling/scoring/test_io_processor_unit.py），只通过 test_max_tokens_per_doc.py 的期望值间接覆盖；若日后 usage 又被改回从后处理结果累加，只有这两个模型会暴露问题，cross-encoder 用例无法兜底。
5. 性能与并发：改动把 $O(N)$ 的累加从响应构造阶段提前到 `_build_response`，量级相同，无性能回归风险。- 影响：用户侧：/score 与 /rerank 返回的 usage 更贴近真实编码量，1:N rerank 场景下数值明显下降；分数、排序、top_n 与响应结构不变，因此对只看 score 的调用方零感知，对按 usage 计费或监控的调用方是一次需要同步的口径切换。系统侧：改动局限在 API server 进程的响应组装层，不涉及调度器、KV 缓存或 worker 执行路径，无吞吐 / 显存影响；cross-encoder 行为完全保持。团队侧：scoring 系列接口的 usage 口径统一为「引擎实际输入」，为后续其它 pooling 端点（embed、pooling）对齐同一计量口径提供了模板与先例。- 风险标记：对外计量语义变更，assert 做输入校验，缺少专用单测，多模态路径未覆盖

关联脉络

- PR #56639 [Pooling] Support prompt embeddings in MRV2 decoder pooling: 同属 MRV2 pooling 请求处理链路的修复，说明 pooling serving 路径近期在持续收敛输入端与后处理之间的语义错位，本 PR 是同一脉络下 usage 口径的补正。
- PR #55508 [Bugfix][Benchmark] Make streaming TTFT/E2E latency accounting consistent across endpoints: 同为「计量口径不一致」类修复：前者统一跨端点计时口径，本 PR 统一 scoring API 的 token 计量口径，设计思路一致（指标必须绑定真实执行事实而非派生数据）。
- PR #56432 [Bugfix][EPD] Preserve explicit multimodal UUIDs with caches disabled: 同样涉及前端渲染 / 输入处理层与后处理语义的对齐问题，且同样以「补单元测试」方式固化行为，可作为同类修复模式的参考。