PR #46775 完整报告

vllm-project/vllm

[CLI] Add flag to print TTFT and TPS in `vllm chat`

合并时间：2026-06-27 13:15

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/46775

执行摘要

- 一句话：vllm chat/complete 添加 --stats 打印 TTFT 和 TPS
- 推荐动作：该 PR 值得精读，展示了如何在 CLI 中优雅地添加性能统计，代码简洁、风险可控。对于需要快速验证性能的用户非常有用。

功能与动机

PR 正文指出这是一个 opt-in 功能，用于在 demo 和快速验证中方便地检查性能是否正确、以及推测解码等场景下的加速效果。作者提到 'it's super handy for demonstrations and quick sanity checks that correctness is maintained and ballpark perf is good'。

实现拆解

1. 新增 --stats 参数：在 _add_query_options 中为 chat 和 complete 子命令添加 --stats 标志，类型为 store_true。
2. 流式函数增加计时逻辑：在 _print_chat_stream 和 _print_completion_stream 中添加 stats: bool = False 参数。当 stats=True 时，使用 time.perf_counter() 记录开始时间，从 chunk.usage.completion_tokens 获取 token 数，在第一个有内容的块到来时记录 TTFT，流结束后调用 _print_metrics 打印。
3. 新增 _print_metrics 函数：计算并格式化输出 TTFT（毫秒）和 TPS（tokens/s，含总耗时与 token 数）。
4. 修改调用处：在 ChatCommand.cmd 中根据 args.stats 设置 create_kwargs，包含 stream_options={'include_usage': True} 以获取 usage 信息，并将 stats 参数传递给流式打印函数。
5. 文档更新：在 docs/cli/README.md 的 chat 和 complete 示例后各添加一行注释和命令示例。

关键文件：

- vllm/entrypoints/cli/openai.py（模块 CLI 入口；类别 source；类型 core-logic；符号 _print_chat_stream, _print_completion_stream, _print_metrics）：核心实现文件，新增了 --stats 参数处理、流式计时逻辑和 _print_metrics 函数
- docs/cli/README.md（模块 文档；类别 docs；类型 documentation）：文档文件，添加了 --stats 的使用示例

关键符号：_print_chat_stream, _print_completion_stream, _print_metrics

关键源码片段

vllm/entrypoints/cli/openai.py

核心实现文件，新增了 --stats 参数处理、流式计时逻辑和 _print_metrics 函数

```
# vllm/entrypoints/cli/openai.py
import time

def _print_chat_stream(stream, stats: bool = False) -> str:
    output = ""
    start = time.perf_counter()
    ttft: float | None = None
    completion_tokens = 0
    for chunk in stream:
        # 尝试从 chunk 获取 usage 信息
        if chunk.usage is not None:
            completion_tokens = chunk.usage.completion_tokens
        if not chunk.choices:
            continue
        delta = chunk.choices[0].delta
        if delta.content:
            # 记录 TTFT（首次 token 时间）
            if ttft is None:
                ttft = time.perf_counter() - start
            output += delta.content
            print(delta.content, end="", flush=True)
    print()
    if stats:
        _print_metrics(start, ttft, completion_tokens)
    return output


def _print_metrics(start: float, ttft: float | None, completion_tokens: int) -> None:
    total_time = time.perf_counter() - start
    if ttft is None or total_time <= 0:
        return
    # 格式化输出 TTFT（单位 ms）和 TPS（单位 tokens/s）
    print(f"{'TTFT:':<5} {ttft * 1000:.2f} ms")
    print(
        f"{'TPS:':<5} {completion_tokens / total_time:.2f} tokens/s "
        f"({completion_tokens} tokens in {total_time:.2f}s)"
    )
```

评论区精华

评审仅由 bot 和一名维护者完成：DarkLight1337 直接批准，无讨论。bot 的自动审查因 fork 被禁用。没有其他评论。

- 暂无高价值评论线程

风险与影响

- 风险：风险极低：
  - 功能默认关闭，不影响现有用户使用。
  - 计时使用 time.perf_counter()，性能开销可以忽略。
  - chunk.usage 可能为 None（当未启用 include_usage 或旧版 API），代码已做 is not None 检查，避免崩溃。
  - _print_completion_stream 中条件从 if text is not None 改为 if text，可能改变空字符串的行为，但原逻辑已隐含。
  - 影响：影响范围有限：仅影响 CLI 用户。新增 --stats 标志，对现有 pipeline 无破坏性。文档同步更新。
  - 风险标记：低风险

关联脉络

- 暂无明显关联 PR