# PR #44255 完整报告

vllm-project/vllm

[ROCm][CI] Specifying time outs for the lm eval models

合并时间：2026-06-04 22:35

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/44255

## 1. 执行摘要

本 PR 为 vLLM 的 ROCm CI 上的 GSM8K 正确性测试添加了可配置的 aiohttp 客户端超时时间，并修复了因超时导致准确率下降的问题。通过新增 request_timeout_seconds 参数和平台感知的配置机制（rocm_request_timeout_seconds），仅对 ROCm 上两个慢速模型应用更长的 1800s 超时，同时改进错误日志以帮助诊断。

## 2. 功能与动机

PR 描述指出，在 ROCm CI 上，DeepSeek-V2-Lite-Instruct-FP8 和 Qwen1.5-MoE-W4A16-CT 两个模型的 GSM8K 测试因 aiohttp 客户端默认 600s 超时而失败。超时的请求返回空字符串，导致无效回答率升高，准确率下降。该问题在 CUDA/H200 和本地 ROCm MI355 上无法复现，因此需要为 ROCm 平台增加可配置的超时时间。

## 3. 实现拆解

1. 核心测试脚本 gsm8k_eval.py 修改：
   - 在 evaluate_gsm8k 函数签名中新增参数 request_timeout_seconds: float = 600，替代硬编码的 600s 超时。
   - 将 aiohttp.ClientSession 的超时从 ClientTimeout(total=600) 改为 ClientTimeout(total=request_timeout_seconds)。
   - 在 call_vllm_api 的异常处理中，将日志从 f"Error calling vLLM API: {e}" 改为 f"Error calling vLLM API ({type(e).__name__}): {e}"，以便区分超时（TimeoutError）和其他错误。
2. 测试入口 test_gsm8k_correctness.py 修改：
   - 在 run_gsm8k_eval 和 test_gsm8k_correctness 中，从 YAML 配置读取 request_timeout_seconds（默认 600）。
   - 如果当前平台是 ROCm（current_platform.is_rocm()），则优先使用配置中的 rocm_request_timeout_seconds。
   - 将解析后的超时传递至 evaluate_gsm8k，并在日志中打印超时值以便调试。
3. YAML 配置修改：
   - 为两个 ROCm 敏感模型 DeepSeek-V2-Lite-Instruct-FP8 和 Qwen1.5-MoE-W4A16-CT 分别添加 rocm_request_timeout_seconds: 1800，将超时从 600s 扩展至 1800s。

### gsm8k_eval.py - 新增参数和超时配置

```
def evaluate_gsm8k(
    num_questions: int = 1319,
    num_shots: int = 5,
    max_tokens: int = 256,
    host: str = "http://127.0.0.1",
    port: int = 8000,
    temperature: float = 0.0,
    seed: int | None = 42,
    request_timeout_seconds: float = 600, # 新增参数，默认 600s
) -> dict[str, float | int]:
    ...
    async def run_async_evaluation():
        ...
        timeout = aiohttp.ClientTimeout(total=request_timeout_seconds)
        async with aiohttp.ClientSession(timeout=timeout) as session:
            ...
```

gsm8k_eval.py - 改进异常日志 `except Exception as e:  print(f"Error calling vLLM API ({type(e).__name__}):{e}") # 输出如 TimeoutError  return "", 0`

test_gsm8k_correctness.py - 平台感知的超时读取 `defrun_gsm8k_eval(eval_config: dict, server_url: str) -> dict:  ...  request_timeout_seconds = eval_config.get("request_timeout_seconds", 600)  if current_platform.is_rocm(): request_timeout_seconds = eval_config.get(  "rocm_request_timeout_seconds", request_timeout_seconds  )  results = evaluate_gsm8k(..., request_timeout_seconds=request_timeout_seconds)`

## 5. 评论区精华

审核者 tjtanaa 审批通过，评语为“LGTM”，无其他讨论。

## 6. 风险与影响

- 风险：低风险。仅修改测试代码和配置，不影响核心推理逻辑。但超时时间从 600s 提升至 1800s 可能掩盖性能退化，不过若模型推理异常，准确率阈值仍会触发断言失败。
- 影响：仅影响 ROCm CI 上的 GSM8K 测试。稳定性提升将减少假阳性失败，提高 CI 可靠性。未来若其他模型在 ROCm 上遇到类似超时问题，只需在 YAML 配置中添加 rocm_request_timeout_seconds 即可复用此模式。

## 7. 关联脉络

本 PR 是独立的 CI 稳定性修复，与近期其他 CI 相关 PR（如 #44497 回退 gitignore 变更）无直接关联，但共同体现了 ROCm CI 的持续改进。