

PR #56309 完整报告

vllm-project/vllm

[watermarking hardening] Add e2e test and basic GSM8K quality tests

合并时间: 2026-09-14 23:29

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/56309>

执行摘要

- 一句话: 为水印实现补齐 E2E 检测用例与 GSM8K 质量回归 CI
- 推荐动作: 值得精读: 如果你正在为某个新特性设计回归护栏或为 Buildkite 配置定向触发的评测步骤, 本 PR 是简洁的样板, 尤其是 `source_file_dependencies` 的收敛思路、`autorun_on_main` 的取舍, 以及 GSM8K 阈值的 3 sigma 校准方式。重点看 `tests/watermarking/test_watermarking_e2e.py` 中「开 / 关对照 + golden p-value + 布尔断言」三层结构, 以及 TQCB 关于确定性的论证。可以略读: 只关心水印算法本身的读者, 本 PR 没有任何算法或内核改动, 算法细节应回到 #54053 与 RFC #53916。后续建议: (a) 把 `recorded_p_value` 的失败信息增强为打印实际 p-value 与 token 序列长度, 降低定位成本; (b) 在 tracking issue #56105 上补一条明确的后续计划, 逐步补上 SD/TP2 组合与速度回归; (c) 观察几次 main 构建, 确认 GSM8K 阈值不抖动后再考虑收紧容差。

功能与动机

PR body 明确说明动机: 这是为 issue #56105 服务的后续工作, 为 #54053 合入的水印实现 (设计来自 RFC #53916) 补充粗粒度质量评估, 并且强调「extends that implementation with tests rather than duplicating it」。issue #56105 列出两条核心诉求: 其一是 Feature compatibility, 希望「从客户端视角」验证水印与非水印生成是否被正确识别, 并尽快暴露特性组合失败; 其二是 Quality Regressions, 担心水印的理论分布保持性在实践中退化, 建议用 GSM8K 之类数据集加容差阈值做基线对比。issue 中还有两条关键约束被本 PR 遵守: temperature 不能为 0 (否则水印失效), 以及需要一个可复用的评测入口。

实现拆解

1. E2E 检测用例的入口与对照设计: 新增 `tests/watermarking/test_watermarking_e2e.py`, 唯一用例为 `test_llm_generated_sequence_is_watermarked`。它通过 `vllm_runner` 以 `facebook/opt-125m` 起一个最小引擎 (`enforce_eager=True`、`max_model_len=512`、`gpu_memory_utilization=0.2`、`seed=0`), 把同一份 `watermark_config = {"key": 42, "context_width": 4}` 同时交给引擎与 `GumbelWatermarkDetector`, 确保采样端的 Gumbel 扰动与检测端的统计检验使用同一密钥, 否则 p-value 断言没有意义。用例用同一 `prompt`、`temperature=1.0`、`max_tokens=256` 跑两次采样, 分别带 `SamplingParams(watermarking=True)` 与 `watermarking=False`, 再对两条 `token_ids` 调用 `detector.detect()`。这种「开 / 关对照」写法直接对应 issue #56105 的第一条诉求 (客户端视角判断是否正确识别), 并避免只断言正例导致检测器恒返回 `True` 也能通过。整条

用例用 `@pytest.mark.skipif(not current_platform.is_cuda_alike(), reason="requires aCUDA-like accelerator")` 守卫，与仓库其它依赖加速器的测试保持一致的平台策略，非 CUDA-like 平台安全跳过而非失败。

2. 断言从布尔升级为 golden p-value: 用例先用 `assert result_use_wm.is_watermarked / assert not result_no_wm.is_watermarked` 做方向性断言，再通过 `recorded_p_value = 9.83e-20` 与 `assert result_use_wm.p_value < recorded_p_value * 10` 做数值断言；后者来自 review 讨论 (commit 9605104 「Add p-value check」)。这样水印器或检测器的细微行为漂移（即便仍能判定为「含水印」）也会被数值断言捕获，而不只是「能检出 / 不能检出」。
3. 质量回归配置与模型清单: 新增 `tests/evals/gsm8k/configs/Qwen3.5-4B-watermark.yaml`，用全量 1319 题、`num_fewshot: 5`、`max_concurrency: 128`、`max_tokens: 1024`、`temperature: 0.7`、`seed: 42` 运行，并把 `--watermark-config '{"key": 42, "context_width": 4}'` 放进 `server_args` 由服务端启用。`accuracy_threshold: 0.885` 与 `tolerance: 0.015` 由 13 次无水印运行的均值与 3 sigma 校准（配置内注释说明）。`temperature: 0.7` 是对 issue #56105 中「temperature should be not 0.0, since this deactivates watermarking」约束的落实。同时新增单行清单 `tests/evals/gsm8k/configs/models-watermark.txt`（内容为 `Qwen3.5-4B-watermark.yaml`），使该配置能通过既有的 `--config-list-file` 机制被批量驱动。
4. CI 编排与触发收敛: `.buildkite/test_areas/lm_eval.yaml` 新增 `lm-eval-watermarking` 步骤，标签为 `:nvidia: (H200 MIG 35GB) LM Eval Watermarking`，`timeout_in_minutes: 10`（注释记录实测单次 80–140 秒），`autorun_on_main: true` 保证 main 分支持续回归。`source_file_dependencies` 覆盖 `.buildkite/test_areas/lm_eval.yaml`、`tests/evals/gsm8k/vllm/config/watermarking.py`、`vllm/sampling_params.py`、`vllm/v1/watermarking/`、`vllm/v1/worker/gpu/`，把这条较重的评测限制在水印相关路径变动时才触发，同时把 GPU 采样 worker 也纳入依赖，避免采样侧改动绕过评测。命令为 `pytest -s -v evals/gsm8k/test_gsm8k_correctness.py --config-list-file=configs/models-watermark.txt`，与相邻的 `H200 small-models` 步骤写法一致。
5. 无生产代码与 schema 改动: 本 PR 不修改 `vllm/v1/watermarking/` 等实现代码，也不改配置 schema；`documentation` 标签主要来自 `models-watermark.txt` 这类清单文件。commit 历史显示作者为跟进 main 做了 4 次 merge（含 `mergify[bot]` 的合并），且 PR body 声明分支重建后「Runtime tests and model evaluations were not rerun」，验证主要依赖合并前后的 Buildkite 运行。

关键文件:

- `tests/watermarking/test_watermarking_e2e.py`（模块 水印测试；类别 test；类型 test-coverage；符号 `test_llm_generated_sequence_is_watermarked`）：本 PR 的核心新增文件：引擎级水印 E2E 用例，用同一密钥驱动采样与检测，并以布尔 + golden p-value 双重断言覆盖「开 = 可检出、关 = 不可检出」，是 review 讨论 (TQCB 关于 golden value) 的唯一落点。
- `tests/evals/gsm8k/configs/Qwen3.5-4B-watermark.yaml`（模块 质量评测；类别 test；类型 test-coverage）：质量回归评测的配置入口：定义全量 GSM8K、采样参数、3 sigma 容差阈值，并通过 `server_args` 里的 `--watermark-config` 在服务端启用同款水印密钥，是

issue #56105 「Quality Regressions」诉求的直接实现。

- `.buildkite/test_areas/lm_eval.yaml` (模块 持续集成; 类别 `config`; 类型 `configuration`) : CI 侧的编排改动: 新增 `lm-eval-watermarking` 步骤, 用水印相关的 `source_file_dependencies` 与 `autorun_on_main` 把这条较重的评测限制在必要触发条件下, 是 issue #56105 「放 CI / nightly 里持续回归」的落地方式。
- `tests/evals/gsm8k/configs/models-watermark.txt` (模块 评测清单; 类别 `docs`; 类型 `documentation`) : 单行配置清单 (`Qwen3.5-4B-watermark.yaml`), 把新评测接入既有 `--config-list-file` 驱动方式; 内容极简但决定了 CI 步骤实际跑哪些配置。

关键符号: `test_llm_generated_sequence_is_watermarked`

关键源码片段

`tests/watermarking/test_watermarking_e2e.py`

本 PR 的核心新增文件: 引擎级水印 E2E 用例, 用同一密钥驱动采样与检测, 并以布尔 + golden p-value 双重断言覆盖「开 = 可检出、关 = 不可检出」, 是 review 讨论 (TQCB 关于 golden value) 的唯一落点。

```
# SPDX-License-Identifier: Apache-2.0
```

```
# SPDX-FileCopyrightText: Copyright contributors to the vLLM project
```

```
import pytest
```

```
from vllm import SamplingParams
```

```
from vllm.platforms import current_platform
```

```
from vllm.v1.watermarking import GumbelWatermarkDetector
```

```
# 水印仅在 CUDA-like 加速器上实现, 非该类平台直接跳过而不是判失败
```

```
@pytest.mark.skipif(
```

```
    not current_platform.is_cuda_alike(), reason="requires a CUDA-like accelerator"
```

```
)
```

```
def test_llm_generated_sequence_is_watermarked(vllm_runner):
```

```
    """Engine-level watermarking must produce tokens the detector recognizes."""
```

```
    # 同一份 config 同时喂给采样端 (runner) 与检测端 (detector),
```

```
    # 两侧 key/context_width 必须完全一致, 否则统计检验失去意义
```

```
    watermark_config = {"key": 42, "context_width": 4}
```

```
    detector = GumbelWatermarkDetector(**watermark_config)
```

```
# opt-125m + enforce_eager + 0.2 显存占用, 让这条 E2E 用例在 CI 小卡上快速跑完
```

```
runner = vllm_runner(
```

```
    "facebook/opt-125m",
```

```
    watermark_config=watermark_config,
```

```
    seed=0,
```

```
    enforce_eager=True,
```

```
    max_model_len=512,
```

```
    gpu_memory_utilization=0.2,
```

```
)
```

```

request = "Tell me a story about an explorer who discovers a mysterious island."
# temperature 必须 > 0: temperature=0 退化为贪心采样会直接关闭水印
params_use_wm = SamplingParams(temperature=1.0, max_tokens=256, watermarking=True)
params_no_wm = SamplingParams(temperature=1.0, max_tokens=256, watermarking=False)

with runner:
    # 同一 prompt 下跑两次, 构造「开 / 关」对照, 避免检测器恒真也能通过
    output_use_wm = runner.llm.generate(request, params_use_wm)
    output_no_wm = runner.llm.generate(request, params_no_wm)

# detect 接收 token id 序列而非文本, 绕开 detokenize 引入的不确定性
result_use_wm = detector.detect(list(output_use_wm[0].outputs[0].token_ids))
result_no_wm = detector.detect(list(output_no_wm[0].outputs[0].token_ids))

# 方向性断言: 开启水印的序列必须被判为含水印
assert result_use_wm.is_watermarked, result_use_wm
# review 中 TQCB 要求记录 golden p-value: 单 key Gumbel 下检测应跨运行完全一致,
# 这里留 10 倍容差, 任何数量级漂移都说明 watermark/detector 有细微行为变化
recorded_p_value = 9.83e-20
assert result_use_wm.p_value < recorded_p_value * 10 # tolerance

# 反向断言: 显式关闭水印的序列不应被判为含水印
assert not result_no_wm.is_watermarked, result_no_wm

```

tests/evals/gsm8k/configs/Qwen3.5-4B-watermark.yaml

质量回归评测的配置入口: 定义全量 GSM8K、采样参数、3 sigma 容差阈值, 并通过 server_args 里的 --watermark-config 在服务端启用同款水印密钥, 是 issue #56105 「Quality Regressions」诉求的直接实现。

```

model_name: "Qwen/Qwen3.5-4B"
# GSM8K 全集 1319 题, 跑全量以获得统计上有效的精度结论
num_questions: 1319
num_fewshot: 5
max_concurrency: 128
max_tokens: 1024
use_chat_completions: true
# temperature 不能设为 0, 否则 Gumbel 水印会被关闭 (issue #56105 明确约束)
temperature: 0.7
seed: 42

# 阈值由 13 次无水印运行校准: threshold 取平均精度, tolerance 取 3 sigma
accuracy_threshold: 0.885
tolerance: 0.015

server_args: >-
  --language-model-only
  --max-model-len 8192
  --max-num-seqs 128
  --gpu-memory-utilization 0.4

```

```
# 关闭 thinking, 避免长思维链把精度评测变成长度评测
--default-chat-template-kwarg 'enable_thinking': false'
# 与 E2E 用例保持同一密钥, 便于两侧结果互相印证
--watermark-config '{"key": 42, "context_width": 4}'
```

评论区精华

讨论集中在一条测试断言的设计上, 是典型的「弱断言 vs golden 值」权衡:

- TQCB 提议记录 golden score / p-value: 在 tests/watermarking/test_watermarking_e2e.py:38 提出「Should we get golden values for score/p-value instead? It should be quite stable for the watermarked response (should be a deterministic response?), we could detect a subtle change in the watermarker or detector」, 目标是让细微实现漂移可被检出。
- tomasruizt 澄清意图范围: 「You mean that we hardcode the current p-value with a tolerance threshold? The concrete value would depend on key, model, prompt, etc. right? Or you mean something like assert p_value < 0.01」, 担心硬编码值受 key/ 模型 /prompt 影响而脆弱。
- TQCB 明确选择: 「P-value + threshold yes, in this test with single-key gumbel we should expect the watermarking and detection to be identical across runs. The p-value changing would indicate something subtly wrong somewhere」。结论被采纳, 新增 recorded_p_value = 9.83e-20 与 10 倍容差断言 (对应 commit 「Add p-value check」)。
- 覆盖范围未展开讨论: issue #56105 列出的 speculative decoding、tool calling、structured output、TP2、多 key 鲁棒性与速度回归等条目在本 PR 中未实现, 也没有 review 明确确认后续由谁承接, 属于留给 tracking issue 的开放项。
- 审批意见简洁: TQCB 给出「lgTM」, LucasWilkinson 以 **APPROVED** + 「LGTM thanks for doing this!」合并; claude[bot] 因 PR 来自 fork 未执行自动审查, 实际把关依赖人工与 CI。
 - 是否用 golden p-value 替代弱阈值断言 (testing): 采纳数值断言方案: 保留 is_watermarked 布尔断言, 同时加入 recorded_p_value = 9.83e-20 与 10 倍容差 (对应 commit 9605104 「Add p-value check」)。
 - 评测覆盖范围与后续接手项 (testing): 作为 hardening 第一步合入, 其余条目仍留在 tracking issue #56105 中, 尚未有明确的后续 PR 或负责人。
 - 分支重建后是否重跑运行时验证 (other): 审查者未要求重跑: TQCB 给出「lgTM」, LucasWilkinson 以 **APPROVED** + 「LGTM thanks for doing this!」合并, 实际验证由 Buildkite #88335/#88349/#88792/#88838 承担。

风险与影响

- 风险:
 1. 硬编码 p-value 对采样 RNG 变动敏感: recorded_p_value = 9.83e-20 与 10 倍容差依赖「单 key Gumbel 下生成与检测跨运行完全一致」这一前提 (TQCB 在 review 中明确以此为理由)。一旦采样 RNG 路径调整 (issue #56105 关联的 #54143 「

Backend-stable stateless RNG for Gumbel sampling」、#52461「Async scheduling and stateful logits processors」正是相关议题），或检测器统计量定义变化，该断言会失败，且失败信息只暴露一个浮点比较，定位成本较高。

2. GSM8K 阈值可能随环境漂移：accuracy_threshold: 0.885 与 tolerance: 0.015 由 13 次无水印运行校准，评测本身使用 temperature: 0.7 采样，若 chat template 变更、Qwen3.5-4B 权重版本更新、或评测框架的评分 / 抽取逻辑调整，3 sigma 容差可能不足，产生 flaky CI；同时 PR body 声明重建分支后未重跑模型评测，阈值是否仍成立未被重新验证。
3. CI 成本与资源占用：全量 1319 题、max_tokens: 1024 的评测被 autorun_on_main: true 变为每次 main 构建都执行，占用 H200 MIG 35GB 设备；虽然 timeout_in_minutes: 10 相对实测 80-140 秒留有余量，但主分支繁忙时排队与设备争用会间接拖慢其它步骤。
4. 覆盖缺口：本轮只覆盖单 key 基础路径与单卡最小模型，issue #56105 中的特性组合（SD、tool calling、structured output、TP2）、速度回归（目标 1% 开销内）与多 key 鲁棒性统计检验均未落地，水印在这些场景下仍无自动护栏。
5. 验证方式偏弱：分支经历了 4 次 main 合流与一次 mergify 合并，作者明确「Runtime tests and model evaluations were not rerun during this branch cleanup」，最终有效性由合并前后的 CI（Buildkite #88335/#88349/#88792/#88838）兜底。
6. 无生产代码与安全面变化：不涉及线上推理路径，测试中使用的 key: 42、context_width: 4 属于公开测试参数，不引入密钥泄露风险。- 影响：对线上用户无直接影响，本 PR 不触碰推理路径、API 契约或配置 schema。对开发者与维护者影响明显：水印相关目录（vllm/v1/watermarking/、vllm/config/watermarking.py、vllm/sampling_params.py、vllm/v1/worker/gpu/）一旦改动，CI 会自动跑 E2E 检测用例与 GSM8K 精度评测，把水印质量退化的发现时机从「用户反馈」提前到「合并前 / main 构建」。对 CI 体系而言新增一条 GPU 评测步骤，属于成本可接受但会长期占用 H200 MIG 资源的常驻项。对团队而言，它把 watermarking 从「实验性特性」推进为「带回归护栏的特性」，并为 issue #56105 其余条目（SD/TP 组合、速度 bench、多 key 统计检验）提供了可复制的模板：新增评测配置 + 单行清单 + 基于 source_file_dependencies 的定向 CI 步骤。- 风险标记：硬编码 p-value 依赖确定性 RNG，分支重建后未重跑模型评测，GSM8K 阈值可能随环境漂移，特性组合覆盖缺口，main 构建新增 GPU 评测开销

关联脉络

- PR #54053 watermarking 实现 PR（标题未在上下文中给出，PR body 明确引用为被加固对象）：本 PR 是为其实补测试：body 写明「Adds coarse quality evaluation for the watermarking implementation merged in #54053 ... This extends that implementation with tests rather than duplicating it」，E2E 用例依赖其导出的 vllm.v1.watermarking.GumbelWatermarkDetector 与 SamplingParams(watermarking=...) 接口。
- PR #54934 [CI][CPU] Add speculative-decoding coverage to CPU CI: 同类 CI 回归护栏改动：同样新增 E2E 用例 + Buildkite 步骤 + 文档，且都用 source_file_dependencies 与

main 分支自动运行控制触发范围，可对照两者在超时设置与依赖列表上的取舍。