PR #50981 完整报告

vllm-project/vllm

[MISC][Bench] refactor throughput and reuse serve's get samples

合并时间：2026-08-06 18:31

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/50981

执行摘要

- 一句话：throughput 复用 serve 数据集分发，删除 215 行 if-else
- 推荐动作：对 benchmark 工具维护者值得精读：_to_serve_args 适配器加共享分发模式可作为 CLI 复用范例，而 review 中未解决的 ASR、prefix-repetition、custom-audio 兼容性缺口建议在后续 PR 补齐（优先 prefix_repetition 的 None 默认值与 ASR 白名单）。普通用户无需关注。

功能与动机

PR body 明确指出：vllm bench serve 的数据集是 vllm bench throughput 的超集，让 throughput 复用 serve 的 get_samples 后两者共享同一数据集，无需手工把每个数据集分别加入 throughput 或 serve，并可消除 throughput 中大量 if else 的维护负担。关联 issue #50838（MMVU 无法通过 throughput 使用）正是重复分发清单遗漏新数据集的实例，#50849 为该重构的跟踪 issue。

实现拆解

- get_samples 新增关键字参数 multimodal_backends，默认 ("openai-chat", "openai-audio") 保持 serve 现状；原硬编码的后端判断改为参数化判断，错误消息改为动态列出允许后端。
- BenchmarkDataset.get_lora_request、get_random_lora_request、get_round_robin_lora_request 改为 staticmethod，便于 throughput 在共享分发之外按请求索引直接调用。
- 新增 _to_serve_args：复制原始 namespace，对 random_input_len、random_output_len、random_prefix_len 做新旧参数回退，把 --output-len 映射到 hf_output_len、sharegpt_output_len 及 sonnet 系列字段，为 vllm-chat 后端自动开启 enable_multimodal_chat，并用 setdefault 补齐 disable_shuffle 等 serve-only 属性。
- get_requests 由约 180 行逐数据集分支收缩为三步：_to_serve_args → get_samples → assign_loras → filter_requests_for_dp，同时删除 14 个数据集类的直接 import。
- LoRA 信息不在共享分发路径中，assign_loras 在 get_samples 之后按 --lora-assignment（random / round-robin）为每个 SampleRequest 统一附加 LoRARequest；未传 --lora-path 时原样返回，兼容旧行为。
- 新增 session 级 hf_tokenizer fixture（gpt2 tokenizer）。

- assign_loras 三个单测：无 lora_path 时 no-op、round-robin 产生确定性 ID [1, 2, 3, 1, 2, 3]、random 落在 [1, max_loras] 且覆盖全部 ID。
- test_get_requests_resolves_mmvu：用 monkeypatch 将 MMVUDataset 替换为 stub，在不联网条件下验证 yale-nlp/MMVU 经共享分发路径解析成功，锁定 #50838 回归。
- 期间曾新建 tests/benchmarks/test_throughput_dataset_reuse.py，经 Isotr0py 指出与既有 CLI 测试冗余后合并入 test_throughput_cli.py。

关键文件：

- vllm/benchmarks/throughput.py（模块 基准工具；类别 source；类型 core-logic；符号 _to_serve_args, assign_loras, get_requests）：重构主战场：删除约 215 行逐数据集 if-else 分发，新增 _to_serve_args 适配器与 assign_loras LoRA 后处理，是共享分发方案落地的关键。
- vllm/benchmarks/datasets/datasets.py（模块 数据集分发；类别 source；类型 core-logic；符号 get_samples, get_lora_request, get_random_lora_request, get_round_robin_lora_request）：共享分发入口被参数化：get_samples 新增 multimodal_backends 门控并改用动态错误消息，LoRA 辅助方法改为 staticmethod，serve 与 throughput 共用同一分发逻辑。
- tests/benchmarks/test_throughput_cli.py（模块 测试；类别 test；类型 test-coverage；符号 hf_tokenizer, _sr, _lora_args, test_assign_loras_noop_without_lora_path）：测试配套：新增 assign_loras 三个单测与 MMVU 解析回归测试（stub HF 下载），并合并了作者最初单独创建的重叠测试文件。

关键符号：_to_serve_args, assign_loras, get_requests, get_samples, get_lora_request, get_random_lora_request, get_round_robin_lora_request

关键源码片段

vllm/benchmarks/throughput.py

重构主战场：删除约 215 行逐数据集 if-else 分发，新增 _to_serve_args 适配器与 assign_loras LoRA 后处理，是共享分发方案落地的关键。

```
def _to_serve_args(args) -> argparse.Namespace:
    """把 throughput 的 CLI 参数翻译成 get_samples 可读的 namespace。

    bench serve 的 get_samples 需要约 45 个属性，throughput 的 CLI 多数同名，
    这里补齐剩余字段并保留 throughput 原有 flag 名，避免破坏既有脚本。
    """
    d = vars(args).copy()
    # random_* 优先于旧的 --input/--output/--prefix-len 写法
    d["random_input_len"] = getattr(args, "random_input_len", None) or args.input_len
    d["random_output_len"] = getattr(args, "random_output_len", None) or args.output_len
    d["random_prefix_len"] = getattr(args, "random_prefix_len", None) or args.prefix_len
    # 共享分发按数据集读取各自的 output_len 入口，统一映射自 --output-len
    d["hf_output_len"] = args.output_len
    d["sharegpt_output_len"] = args.output_len
    # sonnet 读取独立字段；未设置时回落到 SonnetDataset 自身默认值
```

```
    d["sonnet_input_len"] = args.input_len if args.input_len is not None else 550
    d["sonnet_output_len"] = args.output_len if args.output_len is not None else 150
    d["sonnet_prefix_len"] = args.prefix_len
    # 显式 --enable-multimodal-chat 优先，否则 vllm-chat 后端自动开启
    d["enable_multimodal_chat"] = bool(
        getattr(args, "enable_multimodal_chat", False) or args.backend == "vllm-chat"
    )
    # serve 独有、throughput 未暴露的属性，保持 serve 默认值
    d.setdefault("disable_shuffle", False)
    d.setdefault("skip_chat_template", False)
    d.setdefault("no_stream", False)
    d.setdefault("request_id_prefix", "")
    d.setdefault("chat_template_kwargs", None)
    return argparse.Namespace(**d)


def get_requests(args, tokenizer):
    serve_args = _to_serve_args(args)
    # throughput 复用 serve 的数据集分发；vllm-chat 是唯一可跑多模态数据的
    # 后端，其他后端传空元组（review 指出此时错误消息会变成 use one of []）
    mm_backends = ("vllm-chat",) if args.backend == "vllm-chat" else ()
    requests = get_samples(serve_args, tokenizer, multimodal_backends=mm_backends)
    # LoRA 信息不在共享分发路径中，作为 throughput 特有后处理统一附加
    requests = assign_loras(requests, args)
    # 多节点 DP 场景按 rank 切分请求，保持重构前行为
    requests = filter_requests_for_dp(requests, args.data_parallel_size)
    return requests


def assign_loras(requests, args):
    """按 --lora-path 为每个样本附加 LoRARequest，未配置时原样返回。"""
    lora_path = getattr(args, "lora_path", None)
    if not lora_path:
        return requests
    max_loras = args.max_loras
    lora_assignment = getattr(args, "lora_assignment", "random")
    for i, req in enumerate(requests):
        req.lora_request = BenchmarkDataset.get_lora_request(
            index=i,
            max_loras=max_loras,
            lora_path=lora_path,
            lora_assignment=lora_assignment,
        )
    return requests
```

vllm/benchmarks/datasets/datasets.py

共享分发入口被参数化：get_samples 新增 multimodal_backends 门控并改用动态错误消息，LoRA 辅助方法改为 staticmethod，serve 与 throughput 共用同一分发逻辑。

```
def get_samples(
    args,
    tokenizer: TokenizerLike | None,
    *,
    multimodal_backends: tuple[str, ...] = ("openai-chat", "openai-audio"),
) -> list[SampleRequest]:
    """serve 与 throughput 共享的数据集分发入口。

    multimodal_backends 由调用方控制：serve 使用 OpenAI 端点，
    throughput 使用 vllm-chat。默认值保持 serve 既有语义不变。
    """
    if not hasattr(args, "request_id_prefix"):
        args.request_id_prefix = ""
    if hasattr(args, "random_range_ratio") and isinstance(args.random_range_ratio, str):
        args.random_range_ratio = _parse_range_ratio(args.random_range_ratio)

    # ... 其余按 dataset_name 分发的分支（custom / hf / random-mm 等） ...

    # 多模态数据集只允许指定的聊天类后端进入采样；
    # 注意 throughput 的 ASR 数据集也命中此门控（见风险分析）
    if (
        dataset_class.IS_MULTIMODAL
        and args.backend not in multimodal_backends
        and "embeddings-" not in args.backend
    ):
        raise ValueError(
            f"Multi-modal content is not supported on backend "
            f"{args.backend!r}; use one of {sorted(multimodal_backends)}."
        )
```

评论区精华

核心交锋集中在自动化 review 对行为兼容性的质疑与维护者对重构方向的认可：

- Copilot（vllm/benchmarks/throughput.py:574）：非 vllm-chat 后端时 multimodal_backends 传空元组，多模态数据集报错变成 use one of []，用户无法得知应改用 vllm-chat。
- Copilot（throughput.py:549）：custom_audio 此前在未指定 --custom-output-len 时会回退到 --output-len，重构后该回退丢失。
- Codex 标 P1：ASRDataset（如 openslr/librispeech_asr）IS_MULTIMODAL=True，旧实现对 vllm 后端显式放行（采样参数含 asr_min_audio_len_sec 等），空 allowlist 使所有离线 ASR benchmark 不可用。
- Codex 标 P1：prefix_repetition 的 --prefix-repetition-* 参数缺省为 None 时会被原样传给数据集 sample，覆盖其默认值，num_requests // num_prefixes 直接失败，适配器应像 sonnet 一样补默认值。
- Codex 标 P2：BlazeditDataset（vdaita/edit_5k_char）无条件读取 blazedit_min_distance / blazedit_max_distance，throughput 未定义这两个 CLI 参数，

会 AttributeError。

- Isotr0py：新测试文件 test_throughput_dataset_reuse.py 与现有 test_throughput_cli.py 冗余，建议只补 mm 边界用例；作者回复 great insight! 和 done! 并完成合并。
- Isotr0py 最终 APPROVED：Nice clean up!
- multimodal_backends 空元组导致错误消息 use one of [] (design): 合入版本中 get_requests 仍传空元组，错误消息问题未修复。
- custom_audio 丢失 --output-len 回退 (correctness): head 版本 _to_serve_args 未补该映射，行为差异仍在。
- 离线 ASR 数据集在 vllm 后端被多模态门控拒绝 (correctness): head 版本门控逻辑未区分 ASR 数据集与 chat 风格多模态数据集，问题存留。
- prefix-repetition 参数 None 覆盖数据集默认值 (correctness): head 版本适配器未补默认值，与 sonnet 的处理不对称。
- Blazedit 数据集缺 blazedit_min/max_distance 参数 (correctness): head 版本未见补充 CLI 参数或适配器默认值。
- 新测试文件与既有 CLI 测试冗余 (testing): 作者已采纳，最终测试并入 test_throughput_cli.py。

风险与影响

- 风险：
  1. 离线 ASR 基准回归（高危）：非 vllm-chat 后端时 get_requests 传入空 multimodal_backends，所有 IS_MULTIMODAL 的 HF 数据集（含 ASR）在 vllm 后端下被拒绝，而旧实现明确支持；Codex 的 P1 在合入版本中未见修复。
  2. prefix_repetition 崩溃（高危）：--prefix-repetition-num-prefixes 未显式给出时 namespace 中为 None，get_samples 会把它传给 sample（num_requests // num_prefixes），与重构前数据集默认值行为不同。
  3. custom_audio 参数静默失效（中危）：--output-len 在未给 --custom-output-len 时不再生效，请求退回数据集默认输出长度。
  4. 适配层脆弱性：get_samples 读取约 45 个属性，serve 后续新增数据集属性而 _to_serve_args 未同步时，throughput 会以 AttributeError 暴露，需把适配层纳入版本演进检查。
  5. Blazedit 数据集（vdaita/edit_5k_char）在 throughput 下 AttributeError。
  6. serve 侧错误消息文本变化（低风险）：多模态后端门控参数化后，拒绝消息由固定文案改为动态后端列表，依赖文案的集成测试可能受影响。
- 影响：影响范围集中在 benchmarks 工具链，不触及推理核心路径：
  - benchmark 用户：现有 CLI flag 全部保留，常规 random / sharegpt / sonnet / hf 数据集路径行为不变；多模态、ASR、prefix-repetition、custom-audio 等边缘组合存在上述行为差异。

- 维护者：新增数据集只需在 datasets.py 注册一次，serve 与 throughput 自动同时获得支持，消除双份维护；throughput.py 删除 215 行、净减 143 行。
- 测试：新增 4 个单元测试均挂 benchmark mark，由 CI benchmark 任务执行。
- 风险标记：ASR 离线基准被多模态门控拒绝，prefix_repetition 参数 None 覆盖默认值，custom_audio 的 --output-len 回退丢失，适配层属性缺省风险，review 指出的 P1 合入时未修复

关联脉络

- 暂无明显关联 PR