PR #31941 完整报告

sgl-project/sglang

[Benchmark] Remove obsolete auto-benchmark remnants

合并时间：2026-07-21 20:44

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/31941

执行摘要

- 一句话：移除废弃的 auto-benchmark 代码 3000+ 行
- 推荐动作：安全且及时的清零操作，建议快速合并。对关注基准测试的开发者，可以确认自己的脚本不依赖被删除的接口。

功能与动机

仓库内 llm-serving-auto-benchmark 技能已在 #29487 中移除，但其 Python 支持代码残留。全仓引用审计未发现外部调用者，保留这些未维护代码会造成混淆和增加维护负担。

实现拆解

实现分为五步：

1. 删除 python/sglang/auto_benchmark.py 入口文件：移除 main、build_parser、add_dataset_args 等函数，该文件是 CLI 的入口。
2. 删除 python/sglang/auto_benchmark_lib.py 库文件：移除所有工具函数（如 load_yaml、slugify、canonicalize_flags、flatten）和异常类 SearchDeadlineExceeded，以及核心编排逻辑。
3. 删除 python/sglang/benchmark/datasets/autobench.py 数据集适配器：移除 AutoBenchmarkDataset 类及相关的 prompt 标准化函数（_normalize_messages、_normalize_prompt 等）。
4. 删除 test/registered/unit/auto_benchmark/ 下所有测试文件：包括 __init__.py（测试基类）以及 test_search_tools.py、test_dataset_tools.py、test_run_candidate.py 中的测试用例。
5. 从 python/sglang/benchmark/datasets/__init__.py 中移除 autobench 导入（-2 行），从 python/sglang/benchmark/serving.py 中移除 --dataset-name autobench 选项（-1 行），避免引用缺失模块。

关键文件：

- python/sglang/auto_benchmark_lib.py（模块 基准测试；类别 source；类型 deletion；符号 SearchDeadlineExceeded, load_yaml, as_list, slugify）：核心库文件，删除 1985 行，包含所有 auto-benchmark 编排工具函数和搜索逻辑。

- python/sglang/benchmark/datasets/autobench.py（模块 数据集；类别 source；类型 deletion；符号 _load_json_if_needed, _normalize_messages, _normalize_legacy_system_content, _normalize_prompt）：数据集适配器，删除 299 行，包含 AutoBenchmarkDataset 类和 prompt 标准化逻辑。
- python/sglang/auto_benchmark.py（模块 命令行；类别 source；类型 deletion；符号 add_dataset_args, build_parser, main）：CLI 入口文件，删除 82 行，包含 main 函数和参数解析。
- test/registered/unit/auto_benchmark/test_search_tools.py（模块 测试；类别 test；类型 deletion；符号 TestAutoBenchmarkSearchTools, test_build_candidates_by_tier, test_parallel_search_derives_dp_size, test_build_server_candidates_filters_unsupported_fa3_on_sm100）：被删除的主要测试文件之一，包含搜索工具测试用例。
- test/registered/unit/auto_benchmark/__init__.py（模块 测试；类别 test；类型 deletion；符号 create_lightweight_tokenizer, AutoBenchmarkTestCase, setUp, tearDown）：测试基类，包含测试辅助函数。
- python/sglang/benchmark/datasets/__init__.py（模块 数据集；类别 source；类型 dependency-wiring）：从模块导出中移除 autobench 导入。
- python/sglang/benchmark/serving.py（模块 基准测试；类别 source；类型 core-logic）：移除 --dataset-name autobench 选项。

关键符号：SearchDeadlineExceeded, load_yaml, slugify, canonicalize_flags, flatten, AutoBenchmarkDataset, _normalize_prompt, main, add_dataset_args, build_parser

关键源码片段

python/sglang/auto_benchmark_lib.py

核心库文件，删除 1985 行，包含所有 auto-benchmark 编排工具函数和搜索逻辑。

```
# 此前位于 python/sglang/auto_benchmark_lib.py （已删除）
# 该文件包含 auto-benchmark 的核心编排逻辑和工具函数

class SearchDeadlineExceeded(RuntimeError):
    """当自动基准测试搜索超过全局预算时抛出。"""


def load_yaml(path: str) -> Dict[str, Any]:
    with open(path, "r", encoding="utf-8") as f:
        return yaml.safe_load(f)


def as_list(value: Any) -> List[Any]:
    return value if isinstance(value, list) else [value]


def slugify(text: str) -> str:
    return "".join(ch.lower() if ch.isalnum() else "-" for ch in text).strip("-")
```

```
def canonical_flag_name(name: str) -> str:
    return FLAG_ALIASES.get(name, name)


def canonicalize_flags(flags: Dict[str, Any]) -> Dict[str, Any]:
    return {canonical_flag_name(key): value for key, value in flags.items()}


def flatten(data: Dict[str, Any], prefix: str = "") -> Dict[str, Any]:
    flat: Dict[str, Any] = {}
    for key, value in data.items():
        name = f"{prefix}.{key}" if prefix else key
        if isinstance(value, dict):
            flat.update(flatten(value, name))
        else:
            flat[name] = value
    return flat


def log_line(message: str) -> None:
    tqdm.write(message)
```

python/sglang/benchmark/datasets/autobench.py

数据集适配器，删除 299 行，包含 AutoBenchmarkDataset 类和 prompt 标准化逻辑。

```
# 此前位于 python/sglang/benchmark/datasets/autobench.py （已删除）
# 该文件定义了 AutoBenchmarkDataset 类和专用工具函数

AUTOBENCH_RESERVED_FIELDS = {
    "prompt", "messages", "output_len", "max_tokens",
    "prompt_len", "image_data", "extra_request_body",
    # ... 其他字段
}

def _load_json_if_needed(value: Any) -> Any:
    if not isinstance(value, str):
        return value
    value = value.strip()
    if not value or value[0] not in "[{":
        return value
    try:
        return json.loads(value)
    except json.JSONDecodeError:
        return value

def _normalize_messages(messages: Any) -> Optional[List[Dict[str, Any]]]:
    messages = _load_json_if_needed(messages)
    if not isinstance(messages, list) or not messages:
        return None
```

```
    if not all(isinstance(message, dict) for message in messages):
        return None
    normalized = []
    for message in messages:
        if "role" not in message or message.get("content") is None:
            return None
        normalized.append({"role": message["role"], "content": message["content"]})
    return normalized

def _normalize_prompt(row: Dict[str, Any]) -> Tuple[Any, str]:
    """检查多种 prompt 字段（ prompt、messages、prompt_origin ）并按优先级返回归一化结果。"""
    prompt = row.get("prompt")
    messages = row.get("messages")
    prompt_origin = row.get("prompt_origin")
    if messages is not None:
        normalized = _normalize_messages(messages)
        if normalized is not None:
            return normalized, "messages"
    if prompt is not None:
        prompt = _load_json_if_needed(prompt)
        if isinstance(prompt, list) and prompt and isinstance(prompt[0], dict):
            normalized = _normalize_messages(prompt)
            if normalized is not None:
                return normalized, "messages"
        # 其他分支：多轮文本、token ID 等
        # ...
    if prompt_origin is not None:
        normalized = _normalize_messages(prompt_origin)
        if normalized is not None:
            return normalized, "messages"
    raise ValueError("Unsupported auto benchmark row: missing prompt/messages")
```

python/sglang/auto_benchmark.py

CLI 入口文件，删除 82 行，包含 main 函数和参数解析。

```
# 此前位于 python/sglang/auto_benchmark.py （已删除）
# 该文件提供了 sglang.auto_benchmark 命令行工具的入口

def add_dataset_args(parser: argparse.ArgumentParser) -> None:
    parser.add_argument("--kind", required=True, choices=sorted(SUPPORTED_DATASETS))
    parser.add_argument("--path", default="")
    parser.add_argument("--tokenizer", required=True)
    parser.add_argument("--num-prompts", type=int, default=1000)
    # ... 其他数据集参数

def build_parser() -> argparse.ArgumentParser:
    parser = argparse.ArgumentParser(description="SGLang auto benchmark utilities.")
    subparsers = parser.add_subparsers(dest="command", required=True)
    run_parser = subparsers.add_parser("run")
```

```
    run_parser.add_argument("--config", required=True)
    convert_parser = subparsers.add_parser("convert")
    add_dataset_args(convert_parser)
    convert_parser.add_argument("--output", required=True)
    validate_parser = subparsers.add_parser("validate")
    validate_parser.add_argument("--dataset-path", required=True)
    validate_parser.add_argument("--tokenizer", required=True)
    return parser

def main() -> None:
    args = build_parser().parse_args()
    if args.command == "run":
        run_auto_benchmark(args.config)
    elif args.command == "convert":
        convert_dataset(args)
    elif args.command == "validate":
        validate_dataset(args)

if __name__ == "__main__":
    main()
```

评论区精华

本 PR 没有 review 评论。作者 BBuf 在 body 中说明了充分的清理依据并自行验证后合并。

- 暂无高价值评论线程

风险与影响

- 风险：风险极低。全仓 grep 和 AST 检查确认无外部引用；关联技能已提前移除；原有 unittest 一并删除，不会留下孤立的测试。潜在风险：若存在文档或外部脚本引用了被删除的接口，但仓库本身的审计未发现，且文档未包含在删除范围中。Body 中没有提及文档更新，但原功能已废弃，影响可控。
- 影响：对用户：python -m sglang.auto_benchmark 和 bench_serving --dataset-name autobench 不再可用。其他 benchmark 功能和数据集不受影响。对系统：仓库减少约 3000+ 行代码，降低维护成本。对团队：未来不需要为废弃模块进行迁移或修复。
- 风险标记：已审计无外部引用

关联脉络

- PR #29487 [CI] Remove llm-serving-auto-benchmark skill: 本 PR 删除的是该技能残留的 Python 支持代码，两者构成完整的清理链条。