PR #32279 完整报告

sgl-project/sglang

[Fix] Fail fast when a safetensors index references missing shard files

合并时间：2026-07-24 17:18

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/32279

执行摘要

- 一句话：检查 Tensor index 中所有 shard 文件必须存在
- 推荐动作：建议合并。该 PR 修复了一个典型的“静默腐蚀” bug，代码简洁，测试充分，边界情况覆盖完整，值得精读其防御式编程思路。

功能与动机

修复一个隐蔽的数据损坏 bug：当 sharded checkpoint 下载不完整时，缺失的 shard 文件被 filter_duplicate_safetensors_files 静默丢弃，模型以未初始化的权重加载，CUDA graph 正常捕获但推理结果错误。PR body 明确描述了该问题的表现和危害。

实现拆解

1. 在 filter_duplicate_safetensors_files 中增加缺失 shard 检查：在解析 weight_map 构建 weight_files_in_index 集合后，遍历该集合，找出所有不在磁盘上的文件，若存在则直接 raise RuntimeError，错误信息包含缺失文件列表和“incomplete download?”提示。
2. 保持早期返回逻辑不变：对于单文件模型（无 index）、dummy load 或对象存储路径，不会进入该检查，避免引入回归。
3. 新增单元测试文件 test/registered/unit/model_loader/test_weight_utils.py：覆盖三个场景：
   - test_missing_shard_raises：index 引用两个 shard，磁盘上只有第一个，验证抛出 RuntimeError 且消息包含缺失 shard 名。
   - test_complete_checkpoint_filters_non_indexed：所有 shard 齐全，额外存在一个不在 index 中的文件，验证被过滤掉，返回正确列表。
   - test_single_file_model_no_index_returns_unchanged：无 index 文件，验证原样返回。

关键文件：

- python/sglang/srt/model_loader/weight_utils.py（模块 权重加载；类别 source；类型 data-contract）：核心修改文件，在 filter_duplicate_safetensors_files 函数中增加了 index-to-disk 一致性验证，确保 index 引用的所有 shard 文件都存在，否则立即抛异常。
- test/registered/unit/model_loader/test_weight_utils.py（模块 权重加载；类别 test；类型 test-coverage；符号 _write_index, _touch, TestFilterDuplicateSafetensorsFiles, setUp）：新增的单元测试文件，完整覆盖了三种场景：缺失 shard 抛出异常、完整 checkpoint 正

常过滤、无 index 文件原样返回。测试结构清晰，使用了临时目录隔离，无外部依赖。

关键符号：filter_duplicate_safetensors_files

关键源码片段

python/sglang/srt/model_loader/weight_utils.py

核心修改文件，在 filter_duplicate_safetensors_files 函数中增加了 index-to-disk 一致性验证，确保 index 引用的所有 shard 文件都存在，否则立即抛异常。

```
# python/sglang/srt/model_loader/weight_utils.py
# 在 filter_duplicate_safetensors_files 函数中，解析 weight_map 后增加缺失 shard 检查

    # Iterate through the weight_map (weight_name: safetensors files)
    # to identify weights that we should use.
    with open(index_file_name) as f:
        weight_map = json.load(f)["weight_map"]
    weight_files_in_index = set()
    for weight_name in weight_map:
        weight_files_in_index.add(os.path.join(hf_folder, weight_map[weight_name]))

    # Fail fast if the index references shard files that are not on disk (e.g. an
    # incomplete or interrupted download). Otherwise those shards are silently
    # dropped and the model loads with uninitialized weights.
    missing_files = sorted(f for f in weight_files_in_index if not os.path.isfile(f))
    if missing_files:
        raise RuntimeError(
            f"{index_file} references {len(missing_files)} shard file(s) missing "
            f"from {hf_folder} (incomplete download?): "
            f"{[os.path.basename(f) for f in missing_files]}"
        )

    # Filter out any fields that are not found in the index file.
    hf_weights_files = [f for f in hf_weights_files if f in weight_files_in_index]
    return hf_weights_files
```

test/registered/unit/model_loader/test_weight_utils.py

新增的单元测试文件，完整覆盖了三种场景：缺失 shard 抛出异常、完整 checkpoint 正常过滤、无 index 文件原样返回。测试结构清晰，使用了临时目录隔离，无外部依赖。

```
# test/registered/unit/model_loader/test_weight_utils.py
# 测试 filter_duplicate_safetensors_files 的缺失 shard 检查

import json
import os
import tempfile
import unittest

from sglang.srt.model_loader.weight_utils import filter_duplicate_safetensors_files
```

```
from sglang.test.ci.ci_register import register_cpu_ci
from sglang.test.test_utils import CustomTestCase

register_cpu_ci(est_time=5, suite="base-a-test-cpu")

INDEX_NAME = "model.safetensors.index.json"

def _write_index(folder, weight_map):
    """Helper: 在 folder 下写一个 index JSON 文件。"""
    with open(os.path.join(folder, INDEX_NAME), "w") as f:
        json.dump({"weight_map": weight_map}, f)

def _touch(folder, name):
    """Helper: 在 folder 下创建一个空文件。"""
    path = os.path.join(folder, name)
    open(path, "w").close()
    return path

class TestFilterDuplicateSafetensorsFiles(CustomTestCase):
    def setUp(self):
        self._tmp = tempfile.TemporaryDirectory()
        self.folder = self._tmp.name

    def tearDown(self):
        self._tmp.cleanup()

    def test_missing_shard_raises(self):
        # 场景：index 引用两个 shard，但磁盘上只有第一个（模拟下载中断）
        _write_index(self.folder, {
            "w1": "model-00001-of-00002.safetensors",
            "w2": "model-00002-of-00002.safetensors",
        })
        present = _touch(self.folder, "model-00001-of-00002.safetensors")

        with self.assertRaises(RuntimeError) as cm:
            filter_duplicate_safetensors_files(
                hf_weights_files=[present],
                hf_folder=self.folder,
                index_file=INDEX_NAME,
            )
        # 断言异常消息中包含缺失的 shard 文件名
        self.assertIn("model-00002-of-00002.safetensors", str(cm.exception))

    def test_complete_checkpoint_filters_non_indexed(self):
        # 场景：所有 shard 齐全，额外存在一个不在 index 中的文件，验证被过滤掉
        _write_index(self.folder, {
            "w1": "model-00001-of-00002.safetensors",
            "w2": "model-00002-of-00002.safetensors",
        })
```

```
        shard1 = _touch(self.folder, "model-00001-of-00002.safetensors")
        shard2 = _touch(self.folder, "model-00002-of-00002.safetensors")
        extra = _touch(self.folder, "consolidated.safetensors")

        result = filter_duplicate_safetensors_files(
            hf_weights_files=[shard1, shard2, extra],
            hf_folder=self.folder,
            index_file=INDEX_NAME,
        )
        self.assertEqual(sorted(result), sorted([shard1, shard2]))

    def test_single_file_model_no_index_returns_unchanged(self):
        # 场景：无 index 文件（单文件模型 / dummy load / 对象存储），验证原样返回
        single = _touch(self.folder, "model.safetensors")
        result = filter_duplicate_safetensors_files(
            hf_weights_files=[single],
            hf_folder=self.folder,
            index_file=INDEX_NAME,
        )
        self.assertEqual(result, [single])

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

评论区精华

无 review 讨论。

- 暂无高价值评论线程

风险与影响

- 风险：变更范围极小（+10 行源码，+88 行测试），且仅在 index 文件存在时才执行新增检查，不影响单文件模型、dummy load 或对象存储路径。风险低，主要风险在于若某些异常流程（如 index 已损坏但可读取）可能提前暴露，但这是期望行为。
- 影响：用户：之前因不完整下载导致的静默推理错误现在会提前以清晰错误提示失败，便于用户快速诊断。系统：新增的 RuntimeError 会阻止服务器启动，避免了后续非确定性行为。团队：减少了难复现的推理错误 bug 排查成本。影响程度中等 — 修复了重要但特定场景下的 bug。
- 风险标记：低风险，测试覆盖完整

关联脉络

- 暂无明显关联 PR