PR #24434 完整报告

sgl-project/sglang

[NemotronH] Fix expert scale weight loading

合并时间：2026-05-09 03:37

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/24434

执行摘要

- 一句话：修复 Nemotron-H NVFP4 专家权重加载 KeyError
- 推荐动作：该 PR 修复了一个明确的启动崩溃 bug，变更小且包含配套测试，建议正常合入。值得关注的是，它展示了 SGLang 模型中 weight loading 不同分支的防御性编程模式，可作为类似修复的参考。

功能与动机

关联 Issue #24137 描述了加载 Nemotron-3-Nano-Omni-30B-A3B-Reasoning-NVFP4 时因 down_proj.input_scale 等 expert scale 权重在 params_dict 中不存在而触发 KeyError，导致服务器启动崩溃。PR body 指出 expert 分支是唯一缺少存在性检查的分支，其他分支（stacked params 和默认路径）均有相应的跳过或警告逻辑。

实现拆解

1. 修改 python/sglang/srt/models/nemotron_h.py 第 975-976 行：在 expert 权重映射分支中，在通过 name_mapped 访问 params_dict 前，增加 if name_mapped not in params_dict: continue 检查；若映射后的参数名不在 params_dict 中，则跳过该权重，防止 KeyError。
2. 新增 test/registered/unit/models/test_nemotron_h_weight_loading.py 测试文件：
   - 创建 _FakePPGroup 和 _FakeParam 模拟对象，隔离模型依赖。
   - TestNemotronHWeightLoading._make_minimal_model 构造最小 NemotronHForCausalLM 实例（仅设置必要的属性，如 config.n_routed_experts=2）。
   - test_expert_input_scale_without_target_parameter_is_skipped：传入不存在的 scale 权重 model.layers.1.mixer.experts.0.down_proj.input_scale，验证 load_weights 不抛出异常。
   - test_expert_weight_with_target_parameter_is_loaded：传入存在的 expert 权重，验证 weight_loader 正确调用并传递 shard_id 与 expert_id。
3. CI 注册：使用 register_cpu_ci(est_time=4, suite="stage-a-test-cpu") 将测试注册为 CPU 测试，无需 GPU。

关键文件：

- python/sglang/srt/models/nemotron_h.py（模块 模型加载；类别 source；类型 data-contract）：核心修复文件，在 load_weights 的 expert 分支新增前置检查，防止 KeyError。
- test/registered/unit/models/test_nemotron_h_weight_loading.py（模块 测试用例；类别 test；类型 test-coverage；符号 _FakePPGroup, _FakeParam, init, weight_loader）：新增的单元测试文件，覆盖了跳过失权重和正常加载两种场景，确保修复有效且不破坏已有功能。

关键符号：NemotronHForCausalLM.load_weights

关键源码片段

python/sglang/srt/models/nemotron_h.py

核心修复文件，在 load_weights 的 expert 分支新增前置检查，防止 KeyError。

```
# nemotron_h.py load_weights 方法中 expert 权重分支（关键修复位置）
# ... 循环遍历 expert_params_mapping ...
name_mapped = name.replace(weight_name, param_name)
# 修复：检查映射后的参数是否存在，不存在则跳过，防止 KeyError
if name_mapped not in params_dict:
    continue
param = params_dict[name_mapped]
param.weight_loader(
    param,
    loaded_weight,
    name_mapped,
    shard_id=shard_id,
    expert_id=expert_id,
)
name = name_mapped
break
```

test/registered/unit/models/test_nemotron_h_weight_loading.py

新增的单元测试文件，覆盖了跳过失权重和正常加载两种场景，确保修复有效且不破坏已有功能。

```
"""
Unit tests for NemotronHForCausalLM.load_weights.

Regression test for Nemotron-H expert scale checkpoint tensors that map to
parameters absent from the current runtime model.
"""

from sglang.test.ci.ci_register import register_cpu_ci
register_cpu_ci(est_time=4, suite="stage-a-test-cpu")

import unittest
from types import SimpleNamespace
```

```
import torch
from sglang.srt.models.nemotron_h import NemotronHForCausalLM

class _FakeParam:
    def __init__(self):
        self.loaded = None
    def weight_loader(self, param, loaded_weight, name, *, shard_id=None, expert_id=None):
        self.loaded = (param, loaded_weight, name, shard_id, expert_id)

class TestNemotronHWeightLoading(unittest.TestCase):
    def _make_minimal_model(self, named_parameters=()):
        model = object.__new__(NemotronHForCausalLM)
        model.config = SimpleNamespace(n_routed_experts=2)
        model.model = SimpleNamespace()
        model.pp_group = SimpleNamespace(is_first_rank=True, is_last_rank=True)
        model.remap_prefix = {}
        model.remap_substr = {}
        model.stacked_params_mapping = []
        model.named_parameters = lambda: iter(named_parameters)
        return model

    def test_expert_input_scale_without_target_parameter_is_skipped(self):
        """Expert scale weights absent from params_dict should not raise KeyError."""
        model = self._make_minimal_model()
        weights = [
            ("model.layers.1.mixer.experts.0.down_proj.input_scale", torch.ones(1))
        ]
        model.load_weights(weights) # 应该不会抛出异常

    def test_expert_weight_with_target_parameter_is_loaded(self):
        param = _FakeParam()
        model = self._make_minimal_model(
            [("model.layers.1.mixer.experts.w2_weight", param)]
        )
        loaded_weight = torch.ones(1)
        weights = [
            ("model.layers.1.mixer.experts.0.down_proj.weight", loaded_weight)
        ]
        model.load_weights(weights)
        # 验证 weight_loader 被正确调用
        self.assertEqual(
            param.loaded,
            (param, loaded_weight, "model.layers.1.mixer.experts.w2_weight", "w2", 0),
        )
```

评论区精华

PR 获得了 gemini-code-assist[bot] 的自动 review，表示无反馈。另外 chfeng-cs 请求 Ying1123 和 hnyls2002 审查，Kangyan-Zhou 触发了 CI。无其他人工 review 讨论。

- 暂无高价值评论线程

风险与影响

- 风险：风险极低。修改仅引入了一个前置 if 检查，当参数不存在时 continue 跳过，不会改变任何已存在的加载逻辑或模型前向计算。新增的测试直接覆盖了这个跳过分支，并验证了 happy path 仍正常工作。
- 影响：直接影响：修复了 Nemotron-H 模型在加载 NVFP4 checkpoint 时的启动崩溃，使得此类 checkpoint 可以被成功加载。 间接影响：无，因为不相关分支不会执行此检查。 影响范围：仅限于 NemotronHForCausalLM.load_weights 方法在 expert_params_mapping 分支中的执行路径。
- 风险标记：暂无

关联脉络

- 暂无明显关联 PR