PR #2531 完整报告

radixark/miles

fix(flops): stop assuming every HF config has intermediate_size

合并时间：2026-08-14 11:52

原文链接：http://prhub.com.cn/radixark/miles/pull/2531

执行摘要

- 一句话：修复全 MoE 配置缺 intermediate_size 导致 FSDP 训练崩溃
- 推荐动作：值得精读。重点看 flops_utils.py 中“按需校验 + 显式拒绝”和 actor.py 中“非关键指标失败即降级”的防御性设计；同时留意 except Exception 的粒度是否过宽，以及测试是否应该在 GPU E2E 上补一条回归验证。

功能与动机

PR body 明确指出两点：一是 flops_args_from_hf_config 被 FSDPTrainRayActor.init（miles/backends/fsdp_utils/actor.py:120）无条件调用，配置读不出不是“缺指标”而是“训练崩溃”；二是 Qwen3.5-35B-A3B 是 all-MoE 架构，intermediate_size 字段完全缺失，AttributeError 使 tests/e2e/fsdp/r3/test_qwen3_5_35b_a3b_r3.py 自 #2353 合并后一直是红的，并且因为没有 GPU 常驻跑的 PR 而被判定为无关 PR 的失败。

实现拆解

实现分三步演进（对应 3 个 commit），核心集中在 flops_utils.py 与 fsdp_utils/actor.py：

1. miles/utils/flops_utils.py：给缺失字段加保护并调整推导顺序。把 config.intermediate_size 的裸读改为 getattr(config, "intermediate_size", None)，得到 dense_ffn；moe_ffn 优先取 moe_intermediate_size，缺失时借用 dense_ffn（GPT-OSS 风格）；shared_ffn 如果未显式声明且存在 n_shared_experts，用 n_shared_experts * moe_ffn 计算——利用已推导的专家宽度避免第二次对缺失字段的裸读。
2. flops_utils.py：按层模式按需校验宽度并提前拒绝不可解析配置。通过 _moe_layer_pattern 算出 moe_layer_freq 后，判断是否确实需要 dense 宽度（存在 dense 层）与 MoE 宽度（存在 MoE 层），只有真正需要却拿不到宽度时才抛出带明确信息的 ValueError，避免把 None 传进 FLOPs 模型后在 calculate_fwd_flops 里以难以理解的异常炸掉。
3. miles/backends/fsdp_utils/actor.py：sizing 失败降级为不报告 MFU。init 中给 flops_args_from_hf_config 加 try/except，失败时 self._flops_args = None 并发出 logger.warning；train 中 compute_total_fwd_flops 在 _flops_args is None 时传 None，从而跳过 FLOPS 一栏，保证训练流程照常。
4. 测试配套：tests/fast/utils/test_flops_from_hf_config.py 新增 3 个用例——全 MoE 无 dense 宽度（test_all_moe_config_without_a_dense_ffn_width）、共享专家借用专家宽度

（test_shared_experts_sized_from_the_borrowed_expert_width）、隐藏 FFN 宽度的配置被提前拒绝（test_a_config_that_hides_its_ffn_width_is_rejected_up_front）。前两个在修复前会失败，是“承重”回归测试。

关键文件：

- miles/utils/flops_utils.py（模块 FLOPs 计算；类别 source；类型 core-logic；符号 flops_args_from_hf_config）：核心修复：为 intermediate_size 与 moe_intermediate_size 两个裸读加保护，调整共享专家宽度推导顺序，并新增按层模式校验与提前拒绝逻辑，是本次回归的根因修复点。
- miles/backends/fsdp_utils/actor.py（模块 FSDP 后端；类别 source；类型 core-logic；符号 FSDPTrainRayActor.init, FSDPTrainRayActor.train）：FSDP actor 初始化与训练指标入口：把 FLOPs sizing 失败从“崩溃”降级为“不报告 MFU”，保证训练不被可观测性辅助逻辑中断。
- tests/fast/utils/test_flops_from_hf_config.py（模块 FLOPs 单测；类别 test；类型 test-coverage；符号 test_all_moe_config_without_a_dense_ffn_width, test_shared_experts_sized_from_the_borrowed_expert_width, test_a_config_that_hides_its_ffn_width_is_rejected_up_front）：新增 3 个针对性回归测试，覆盖两个真实 bug（all-MoE 无 dense 宽度、共享专家借宽度）和一个防御性拒绝路径；测试在修复前均会失败，确保证明修复有效。

关键符号：flops_args_from_hf_config, FSDPTrainRayActor.init, FSDPTrainRayActor.train

关键源码片段

miles/utils/flops_utils.py

核心修复：为 intermediate_size 与 moe_intermediate_size 两个裸读加保护，调整共享专家宽度推导顺序，并新增按层模式校验与提前拒绝逻辑，是本次回归的根因修复点。

```
def flops_args_from_hf_config(config):
    # 取 text config：部分 HF 包装类把真实配置嵌套在 text_config 下
    getter = getattr(config, "get_text_config", None)
    config = (getter() if callable(getter) else getattr(config, "text_config", None)) or config

    num_attention_heads = config.num_attention_heads
    hidden_size = config.hidden_size
    num_layers = _first(config, "num_hidden_layers", "num_layers")
    assert num_layers is not None, f"no layer count on {type(config).__name__}; cannot size the FLOPs model"
    num_experts = _first(config, "n_routed_experts", "num_experts", "num_local_experts")

    # 1. dense FFN 宽度：all-MoE 配置（如 Qwen3.5-35B-A3B）根本没有
    # intermediate_size，缺失是合法状态，不应对字段本身报错
    dense_ffn = getattr(config, "intermediate_size", None)

    # 2. MoE 专家宽度：优先取 moe_intermediate_size；若只有 dense 宽度
    # （GPT-OSS 风格），则借用 dense_ffn
    moe_ffn = getattr(config, "moe_intermediate_size", None)
```

```
    if num_experts is not None and moe_ffn is None:
        moe_ffn = dense_ffn

    # 3. 共享专家宽度：显式字段优先；Inkling 风格用
    # n_shared_experts * 专家宽度，这里复用上一步推导出的
    # moe_ffn，避免第二次裸读缺失字段
    shared_ffn = getattr(config, "shared_expert_intermediate_size", None)
    if shared_ffn is None and getattr(config, "n_shared_experts", None):
        shared_ffn = config.n_shared_experts * moe_ffn

    # 4. 按层模式决定哪些宽度真正需要：全 MoE 时不需要 dense 宽度，
    # 全 dense 时不需要 MoE 宽度；只有真正需要却缺失时才报错
    moe_layer_freq = _moe_layer_pattern(config, num_layers, num_experts)
    needs_dense = moe_layer_freq is None or any(f == 0 for f in moe_layer_freq)
    needs_moe = moe_layer_freq is not None and any(f > 0 for f in moe_layer_freq)
    if (needs_dense and dense_ffn is None) or (needs_moe and moe_ffn is None):
        # DBRX 这类把宽度嵌套在 ffn_config 子配置里的架构，在这里提前显式拒绝，
        # 而不是把 None 传进 FLOPs 模型后炸出难懂的异常
        raise ValueError(
            f"{type(config).__name__} does not expose the FFN widths the FLOPs model needs "
            f"(dense={dense_ffn}, moe={moe_ffn}); it likely nests them under a sub-config"
        )

    return SimpleNamespace(
        hidden_size=hidden_size,
        num_attention_heads=num_attention_heads,
        num_query_groups=_first(config, "num_key_value_heads", default=num_attention_heads),
        vocab_size=config.vocab_size,
        num_layers=num_layers,
        ffn_hidden_size=dense_ffn,
        kv_channels=_first(config, "head_dim", default=hidden_size // num_attention_heads),
        num_experts=num_experts,
        moe_ffn_hidden_size=moe_ffn,
        moe_router_topk=_first(config, "num_experts_per_tok", "moe_topk", default=1),
        moe_shared_expert_intermediate_size=shared_ffn,
        moe_layer_freq=moe_layer_freq,
        q_lora_rank=getattr(config, "q_lora_rank", None),
        kv_lora_rank=getattr(config, "kv_lora_rank", None),
        qk_head_dim=getattr(config, "qk_nope_head_dim", None) or 0,
        qk_pos_emb_head_dim=getattr(config, "qk_rope_head_dim", None) or 0,
        v_head_dim=getattr(config, "v_head_dim", None) or 0,
    )
```

miles/backends/fsdp_utils/actor.py

FSDP actor 初始化与训练指标入口：把 FLOPs sizing 失败从“崩溃”降级为“不报告 MFU”，保证训练不被可观测性辅助逻辑中断。

```
# miles/backends/fsdp_utils/actor.py，FSDPTrainRayActor.init 内
self.precision_policy = resolve_precision_policy(self.hf_config, self.args)
```

```
try:
    self._flops_args = flops_args_from_hf_config(self.hf_config)
except Exception as e:
    # MFU 只是可观测性辅助指标，不能让 sizing 失败断送整个训练；
    # 记录告警后降级为不报告 MFU，保证训练照常启动
    self._flops_args = None
    logger.warning(f"MFU will not be reported, {type(self.hf_config).__name__} could not be sized:
    {e}")

# train 里消费端同样做空值保护：
compute_total_fwd_flops=(
    (lambda seq_lens: fwd_tflops_per_gpu(seq_lens, self._flops_args, dist.get_world_size()))
    if self._flops_args is not None
    else None
),
```

tests/fast/utils/test_flops_from_hf_config.py

新增 3 个针对性回归测试，覆盖两个真实 bug（all-MoE 无 dense 宽度、共享专家借宽度）和一个防御性拒绝路径；测试在修复前均会失败，确保证明修复有效。

```
def test_all_moe_config_without_a_dense_ffn_width():
    # Qwen3.5-35B-A3B 场景：全 MoE、无 intermediate_size，缺失是合法状态
    hf = hf_config(num_experts=256, moe_intermediate_size=512, num_experts_per_tok=8)
    del hf.intermediate_size
    assert_same(
        hf,
        megatron_args(
            ffn_hidden_size=None, # 全 MoE，不要求 dense 宽度
            num_experts=256,
            moe_ffn_hidden_size=512,
            moe_router_topk=8,
            moe_layer_freq=[1] * 8,
        ),
    )
```

评论区精华

该 PR 没有 review 评论，但 commit message 中记录了关键设计权衡：

> “Patching architectures one at a time cannot close this, because load_hf_config runs with trust_remote_code and half the models Miles supports ship their config class with the checkpoint. So make the unknown case safe rather than enumerating it.”

即：与其逐个架构打补丁，不如把“未知配置”变成安全的显式拒绝与降级路径。第二个 commit 也提到，修复两个裸读后 DBRX 会因 ffn_config 嵌套导致宽度全部为 None 而在 FLOPs 模型内爆炸，因此需要提前校验。最终第三个 commit 把任何 sizing 失败都降级为“不报告 MFU”，从根上避免此类问题再次断送训练。

- 暂无高价值评论线程

风险与影响

- 风险：主要风险集中在可观测性降级与异常吞噬：
  1. actor.py 的 except Exception 会吞掉所有 sizing 异常，包括未来可能出现的真实 bug，MFU 指标会静默缺失，dashboard 上无法直观发现问题；
  2. flops_utils.py 中 ffn_hidden_size=None 会进入 SimpleNamespace，虽然当前消费端 fwd_tflops_per_gpu 只会在有 _flops_args 时被调用，且全 MoE 路径不需要 dense 宽度，但其它潜在消费方需要确认能容忍 None；
  3. PR 验证只覆盖了 fast 测试与单机 H200 上的手动 sweep，tests/e2e/fsdp/r3/test_qwen3_5_35b_a3b_r3.py 的 GPU 回归是否恢复绿色未在 PR 中明确给出结果。 - 影响：影响面较大：所有 FSDP 后端的 all-MoE 架构训练（Qwen3.5-35B-A3B、Inkling-Small 等）之前会在 actor init 阶段崩溃，本 PR 使它们能正常启动；同时引入“MFU 报告缺省降级”机制，让任何未来无法解析的模型配置不会中断训练，但代价是 dashboard 上可能缺失 MFU 信息。对团队而言，这是 #2353 引入回归的修复，扫清了其它 PR 被无辜牵连的问题。 - 风险标记：核心路径防御变更，MFU 指标可能静默缺失，异常吞噬可能掩盖问题，GPU E2E 未明确验证

关联脉络

- PR #2353 dashboard: report model FLOPs utilization: 本 PR 是 #2353 引入的回归修复，PR body 明确说明 tests/e2e/fsdp/r3/test_qwen3_5_35b_a3b_r3.py 自 #2353 合并后一直是红的，且 #2353 新增的 flops_args_from_hf_config 是本 PR 修复的核心函数。