# PR #35222 完整报告

sgl-project/sglang

[CPU] Enable ERNIE models on CPU

合并时间：2026-08-27 10:13

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/35222

## 执行摘要

- 一句话：CPU 后端启用 ERNIE 模型推理
- 推荐动作：建议精读。这个 PR 以极小 diff 解决“内核契约与权重格式不一致”的典型问题，并通过 review 展示了原地 vs 非原地张量操作的坑；值得关注的设计点是：把设备差异尽量收敛在 Python 层、让内核契约保持简单。合并前建议补一条针对 BF16 + 2D correction_bias 的单元测试。

## 功能与动机

PR 标题即动机：[CPU] Enable ERNIE models on CPU。作者在 body 中说明，topk_softmax_cpu 已在 #31956 实现，是 ERNIE 唯一缺失的内核；剩下的适配点有两个：correction_bias 参数是 BF16 而非 FP32，需要在 Python 接口显式转换 dtype；权重是 2D 张量 [1, num_experts]，而内核实现期望 1D，需要先压缩。

## 实现拆解

### 实施步骤

1. 模型侧适配（python/sglang/srt/models/ernie4.py）：引入 is_cpu() 探测并缓存模块级 _is_cpu，将 Ernie4Moe.__init__ 中 correction_bias.squeeze(0) 的设备条件从 _is_npu 扩展为 _is_npu or _is_cpu，并使用非原地 squeeze(0) 生成新张量，避免修改 nn.Parameter 的原始视图。
2. 内核调用侧适配（python/sglang/srt/layers/moe/topk.py）：在 fused_topk_cpu 中，进入 topk_softmax_cpu / topk_sigmoid_cpu 之前，将非 FP32 的 correction_bias 显式转换为 FP32，统一 CPU 内核输入契约。
3. 验证与配套：PR 没有保留自动化测试（曾提交 2D correction_bias UT，后因 cpp 层改动回退而一并撤回）；作者用 sglang serve --model-path baidu/ERNIE-4.5-21B-A3B-PT --device cpu --tp 6 --disable-overlap-schedule 及 300B 模型在本地完成端到端验证，并给出了 21B 模型输出样例。

关键文件：

- python/sglang/srt/models/ernie4.py（模块 模型适配；类别 source；类型 data-contract；符号 Ernie4Moe.init）：模型入口：负责在 CPU/NPU 上把 2D 的 correction_bias 压缩为 1D，并缓存设备判断，是数据契约适配的核心。

- python/sglang/srt/layers/moe/topk.py（模块 专家路由；类别 source；类型 core-logic；符号 fused_topk_cpu）：公共路由入口：在 fused_topk_cpu 中统一处理 correction_bias 的 dtype 归一化，保证 CPU 内核输入契约。

关键符号：Ernie4Moe.init, fused_topk_cpu

## 关键源码片段

### python/sglang/srt/models/ernie4.py

模型入口：负责在 CPU/NPU 上把 2D 的 correction_bias 压缩为 1D，并缓存设备判断，是数据契约适配的核心。

```
# python/sglang/srt/models/ernie4.py
# 模块级缓存设备判断，避免每层 MoE 构建时重复探测设备类型
_is_cpu = is_cpu()
_is_npu = is_npu()

class Ernie4Moe(nn.Module):
    def __init__(self, config, layer_id, quant_config=None, prefix=""):
        super().__init__()
        self.layer_id = layer_id
        self.tp_size = get_parallel().tp_size
        self.moe_num_shared_experts = getattr(config, "moe_num_shared_experts", 0)

        self.gate = MoEGate(config=config, prefix=add_prefix("gate", prefix))

        correction_bias = self.gate.e_score_correction_bias
        # CPU/NPU 的 topk 内核只接受 1D 张量，而 ERNIE 权重是 2D [1, num_experts]；
        # 使用非原地 squeeze 生成新视图，避免永久改写 nn.Parameter 的原始形状
        if _is_npu or _is_cpu:
            correction_bias = correction_bias.squeeze(0)

        self.topk = TopK(
            top_k=config.moe_k,
            layer_id=layer_id,
            renormalize=True,
            use_grouped_topk=False,
            correction_bias=correction_bias,
        )
```

### python/sglang/srt/layers/moe/topk.py

公共路由入口：在 fused_topk_cpu 中统一处理 correction_bias 的 dtype 归一化，保证 CPU 内核输入契约。

```
# python/sglang/srt/layers/moe/topk.py
def fused_topk_cpu(hidden_states, gating_output, topk, renormalize,
                   correction_bias=None, scoring_func="softmax", ...):
    # CPU 内核契约：correction_bias 必须是 FP32 的 1D 张量；
    # ERNIE 权重自带 BF16 的 [1, num_experts] 参数，先做 dtype 归一化再下发
```

```
    if correction_bias is not None and correction_bias.dtype != torch.float32:
        correction_bias = correction_bias.to(torch.float32)

    if scoring_func == "softmax":
        topk_weights, topk_ids = torch.ops.sgl_kernel.topk_softmax_cpu(
            hidden_states=hidden_states,
            gating_output=gating_output,
            topk=topk,
            renormalize=renormalize,
            correction_bias=correction_bias,
        )
    elif scoring_func == "sigmoid":
        topk_weights, topk_ids = torch.ops.sgl_kernel.topk_sigmoid_cpu(
            hidden_states=hidden_states,
            gating_output=gating_output,
            topk=topk,
            renormalize=renormalize,
            correction_bias=correction_bias,
        )
    else:
        # 其余打分函数回退到 torch 原生实现
        return fused_topk_torch_native(
            hidden_states, gating_output, topk, renormalize,
            correction_bias=correction_bias, scoring_func=scoring_func,
        )

    return topk_weights, topk_ids
```

## 评论区精华

审查中有一场有价值的交锋和一条未闭合的担忧：

- 原地修改风险：mingfeima 指出 squeeze_(0) 会原地改写调用者的张量，[1, E] 的 nn.Parameter 会在第一次 forward 后被永久重塑，建议用非原地 squeeze(0) 或在 Python 层处理；作者采纳并回应："Updated. Squeezing is executed at Python level, like NPU. Test case update also reverted since no change at cpp level now."
- 测试覆盖要求：mingfeima 的 CHANGES_REQUESTED 指出现有 test_topk.py 只覆盖 1D float32 correction_bias，而本 PR 核心点是 BF16 + [1, num_experts]；作者曾补充 2D UT，最终因 cpp 层无改动而回退，该担忧在合并时未彻底解除。
  - squeeze 原地修改 nn.Parameter 的风险 (correctness): 作者接受建议，将 squeeze 移动到 Python 层（ernie4.py），并随 cpp 改动回退一并撤回了对应测试用例更新。
  - 缺少 test_topk.py 的 2D BF16 用例 (testing): 最终未保留自动化测试覆盖，该担忧在合并时未彻底解除；作者认为 python 层类型转换不需要内核级 UT。

## 风险与影响

- 风险：变更虽小，但有三个值得注意的风险：

- 测试缺口：BF16 + 2D correction_bias 的核心场景没有任何单元测试，未来的 topk 重构可能无声回归。
- 公共路径影响：fused_topk_cpu 是所有 CPU MoE 路由的公共入口，新增 dtype 归一化对全部 CPU 模型生效；BF16 → FP32 转换通常可忽略，但对位级敏感的路由权重可能有微小数值差异。
- 保护范围局限：squeeze(0) 仅在 dim == 2 && size(0) == 1 时生效，若未来模型 correction_bias 形状变为 (B, E)，该安全网不会触发，会直接暴露内核的形状断言；同时这是模型层改动，对 CUDA/NPU 主路径无影响。
- 影响：
  - 用户侧：CPU 用户可直接部署 baidu/ERNIE-4.5-21B-A3B-PT 与 baidu/ERNIE-4.5-300B-A47B-PT，服务命令需要 --device cpu --disable-overlap-schedule。
  - 系统侧：补齐了 ERNIE 模型在 CPU 上的 MoE topk 链路，主路径改动集中在 fused_topk_cpu 数据类型契约和 Ernie4Moe 的 correction_bias 形状处理。
  - 团队侧：为其他依赖非标准 correction_bias 权重格式的模型提供了 Python 层适配样板，但也暴露了 topk 路径测试覆盖薄弱的问题。
  - 风险标记：缺少测试覆盖，核心路径变更（topk 库）

## 关联脉络

- 暂无明显关联 PR