PR #42944 完整报告

vllm-project/vllm

fix: glm5.1 pp model loading

合并时间：2026-06-01 15:14

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/42944

执行摘要

- 一句话：修复 GLM5.1 的 FP8 模型在 PP 模式下的加载失败
- 推荐动作：建议精读 _try_load_fp8_indexer_wk 的改动以理解 PP 下权重加载的最佳实践。该 PR 改动简洁、目标明确，是学习如何为已有加载逻辑添加 PP 支持的好例子。

功能与动机

当使用 vllm serve zai-org/GLM-5.1-FP8 --pipeline-parallel-size 8 启动服务时，加载过程抛出 KeyError: 'layers.0.self_attn.indexer.wk_weights_proj.weight'。原因是 FP8 索引器 weight 加载函数 _try_load_fp8_indexer_wk 未考虑 PP 下部分层不存在的场景，导致尝试访问不存在的参数。

实现拆解

1. 在 deepseek_v2.py 中新增 get_pp_missing_layer_names 导入，并修改 _try_load_fp8_indexer_wk 函数签名，增加 pp_missing_layer_names 参数。
2. 在 _try_load_fp8_indexer_wk 函数体开头，计算 fused_name 后，检查当前权重名称是否以任意 pp_missing_layer_names 开头；若是，则直接 return True（跳过该权重的处理），避免后续尝试访问不存在的参数。
3. 在 deepseek_v2.py 的 load_weights 方法中，调用 get_pp_missing_layer_names(self) 获取缺失层名列表，并将该列表传递给 _try_load_fp8_indexer_wk。
4. 在 deepseek_mtp.py 中同步修改：新增 get_pp_missing_layer_names 的导入，在 load_weights 中计算缺失层名并传递给 _try_load_fp8_indexer_wk。

关键文件：

- vllm/model_executor/models/deepseek_v2.py（模块 模型加载；类别 source；类型 core-logic；符号 _try_load_fp8_indexer_wk, get_pp_missing_layer_names）：核心修复文件：修改 _try_load_fp8_indexer_wk 函数签名与行为，新增跳过缺失层的逻辑；同时在 load_weights 中集成 get_pp_missing_layer_names。
- vllm/model_executor/models/deepseek_mtp.py（模块 推测解码；类别 source；类型 data-contract）：同步修改 MTP 模型的权重加载，使其在 PP 场景下也能正确跳过缺失层的 FP8 WK 权重。

关键符号：_try_load_fp8_indexer_wk

关键源码片段

vllm/model_executor/models/deepseek_v2.py

核心修复文件：修改 _try_load_fp8_indexer_wk 函数签名与行为，新增跳过缺失层的逻辑；同时在 load_weights 中集成 get_pp_missing_layer_names。

```
def _try_load_fp8_indexer_wk(
    name, tensor, buf, params_dict, loaded_params, pp_missing_layer_names
):
    """
    We fuse the WK and weights_proj projections, but in some checkpoints WK is stored
    in FP8 with a separate weight_scale_inv, while weights_proj is stored in BF16.
    Upcasting to BF16 during loading enables the fusion. This function loads the FP8 WK
    weights and scale, and when both are available, dequantizes to BF16 and stores into
    the fused wk_weights_proj.weight parameter.
    """
    if "indexer.wk." not in name or "wk_weights" in name:
        return False # Weight is not an isolated WK weight for the indexer, ignore.
    is_weight = name.endswith(".weight") and tensor.dtype == torch.float8_e4m3fn
    is_scale = "weight_scale_inv" in name
    if not is_weight and not is_scale:
        return False # WK is not in FP8 format, ignore.
    # Buffer this tensor (weight or scale) until both have arrived.
    layer_prefix = name.rsplit(".wk.", 1)[0] # e.g. "model.layers.0.self_attn.indexer"
    fused_name = f"{layer_prefix}.wk_weights_proj.weight"
    # If this layer is missing in the current pipeline stage, skip it entirely.
    if any(
        name.startswith(missing_layer_name)
        for missing_layer_name in pp_missing_layer_names
    ):
        return True
    entry = buf.setdefault(layer_prefix, {})
    entry["weight" if is_weight else "scale"] = tensor
    if "weight" not in entry or "scale" not in entry:
        return True # still waiting for the other param
    # We have both weight and scale: dequantize FP8 to BF16.
    weight_fp8, scale_inv = entry["weight"], entry["scale"]
    del buf[layer_prefix]
    block_size = weight_fp8.shape[1] // scale_inv.shape[1]
    weight_bf16 = scaled_dequantize(
        weight_fp8,
        scale_inv,
        group_shape=GroupShape(block_size, block_size),
        out_dtype=torch.bfloat16,
    )
    # Load the dequantized weight into shard 0 of the fused buffer.
    param = params_dict[fused_name]
    param.weight_loader(param, weight_bf16, 0)
    loaded_params.add(fused_name)
```

```
    return True
```

评论区精华

Reviewer cjackal 指出 deepseek_mtp 也需要同步修改以支持 MTP（多 token 预测），因为 _try_load_fp8_indexer_wk 的函数签名已变更。作者 UranusSeven 确认了该问题，并在后续提交中同步更新了 deepseek_mtp.py。

- deepseek_mtp 也需要同步修改 (correctness): 作者 UranusSeven 确认并跟进修改，最终在 deepseek_mtp.py 中增加了 get_pp_missing_layer_names 的导入和传递。

风险与影响

- 风险：风险较低。变更集中仅在权重加载路径中添加跳过逻辑，不影响推理性能。主要风险在于：若 get_pp_missing_layer_names 返回的列表格式或含义发生变化，可能导致加载行为异常；但该函数已经在 PP 场景中广泛使用，稳定性有保障。
- 影响：直接修复了 GLM-5.1-FP8 在 PP=8 下的加载崩溃，使得该模型可正常进行 pipeline parallel 推理。对不使用 PP 或非该模型的场景无影响。
- 风险标记：核心路径变更，缺少测试覆盖

关联脉络

- 暂无明显关联 PR