PR #35228 完整报告

sgl-project/sglang

[Quant] Load compressed-tensors quantized lm_head instead of value-casting it

合并时间：2026-08-20 06:37

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/35228

执行摘要

- 一句话：修复 compressed-tensors 量化 lm_head 加载乱码问题
- 推荐动作：建议精读。该 PR 展示了一个典型的"配置语义与加载器约束冲突"问题：量化方案的 target 匹配规则由上游 llm-compressor 决定，而权重加载器对分片维度有硬性限制。get_lm_head_scheme 中对"按层名匹配 vs 模块类型匹配"的区分、以及 matched_target 透传的设计，在后续支持更多量化格式时值得复用。

功能与动机

量化生态中存在大量把输出头也量化的 checkpoint（典型 target 写法为 re:.*lm_head 的 FP8 per-channel 权重），但 SGLang 的 CompressedTensorsConfig.get_quant_method 对 ParallelLMHead 直接返回 None，导致 lm_head 回退到 UnquantizedEmbeddingMethod：打包的 FP8 权重被 value-cast 成 bf16 参数，lm_head.weight_scale 被静默丢弃（加载时出现 Parameter lm_head.weight_scale not found in params_dict 警告），模型生成完全退化。PR body 用 A/B 测试证实：main 分支 GSM8K 为 0.000、85% 输出无效，而本 PR 修复后 GSM8K 达到 0.955。

实现拆解

1. 扩展 get_quant_method 的 ParallelLMHead 分支：在 python/sglang/srt/layers/quantization/compressed_tensors/compressed_tensors.py 中，get_quant_method 原本只处理 LinearBase 与 FusedMoE，对 ParallelLMHead 一律返回 None。本次在 LinearBase 分支后新增 ParallelLMHead 分支：调用新方法 get_lm_head_scheme 解析方案，若返回 None 则维持原有未量化 embedding 行为，否则为 head 挂载 scheme 并返回 CompressedTensorsLinearMethod。这一步直接决定了权重加载走 CompressedTensorsW8A8Fp8 等线性量化路径，而不是被 value-cast 成 bf16。
2. 新增 get_lm_head_scheme 解析方法：该方法只接受按层名命中的 target（精确名、re: 正则、点号后缀），模块类型 target（如 Linear）一律不参与，从而保证遵循"head 未量化且未在 config 提及"惯例的 checkpoint 不受影响；ignore 列表优先生效；当命中 block-FP8 策略时直接抛错，因为 vocab-parallel 权重加载器无法对 vocab/block_n 权重尺度做分片。
3. 改造 get_linear_scheme 透传匹配结果：get_linear_scheme 新增可选参数 matched_target 并把它传给 get_scheme_dict。原因是后缀匹配（如 target lm_head 对应 language_model.lm_head）能通过 check_equal_or_regex_match，但无法被下游更严

格的 find_matched_target 重新推导，透传可避免加载期 ValueError。

4. 配套单元测试：新增 test/registered/unit/layers/quantization/test_compressed_tensors_lm_head.py（CPU，注册到 base-a-test-cpu 套件），覆盖正则 / 精确 / 后缀匹配、ignore、模块类型 target、无层名、block-FP8 拒绝等场景。无需其他配套改动：现有 vocab-parallel 权重加载器已支持 output_dim=0 参数分片，模型加载器已执行 process_weights_after_loading，logits 路径已通过 should_apply_lm_head_quant_method 路由。

关键文件：

- python/sglang/srt/layers/quantization/compressed_tensors/compressed_tensors.py（模块 量化层；类别 source；类型 core-logic；符号 get_quant_method, get_lm_head_scheme, get_linear_scheme）：核心修复文件：新增 ParallelLMHead 量化分支与 get_lm_head_scheme 解析方法，并改造 get_linear_scheme 支持透传 matched_target，是决定量化 lm_head 能否正确加载的关键。
- test/registered/unit/layers/quantization/test_compressed_tensors_lm_head.py（模块 量化测试；类别 test；类型 test-coverage；符号 TestGetLmHeadScheme, test_regex_target_resolves, test_exact_target_resolves, test_ignored_head_is_none）：新增 CPU 单元测试，覆盖 scheme 解析的全部关键分支与回归场景，是验证 target 匹配语义和 block 拒绝行为的主要依据。

关键符号：get_lm_head_scheme, get_quant_method, get_linear_scheme

关键源码片段

python/sglang/srt/layers/quantization/compressed_tensors/compressed_tensors.py

核心修复文件：新增 ParallelLMHead 量化分支与 get_lm_head_scheme 解析方法，并改造 get_linear_scheme 支持透传 matched_target，是决定量化 lm_head 能否正确加载的关键。

```
# CompressedTensorsConfig.get_quant_method 中新增的 ParallelLMHead 分支
# 入口：量化配置解析时，SGLang 为每个层调用本方法获取量化方法。
# 这里让量化了 lm_head 的 checkpoint 走压缩张量线性量化路径，
# 而不是回退到 UnquantizedEmbeddingMethod（此前会把 FP8 权重 value-cast
# 成 bf16 造成输出乱码）。

from sglang.srt.layers.vocab_parallel_embedding import ParallelLMHead

if isinstance(layer, ParallelLMHead):
    # 解析 head 的量化方案；若 checkpoint 未量化 head，
    # get_lm_head_scheme 返回 None，维持原先的未量化 embedding 行为。
    scheme = self.get_lm_head_scheme(layer=layer, layer_name=prefix)
    if scheme is None:
        return None
    layer.scheme = scheme
    return CompressedTensorsLinearMethod(self)
```

test/registered/unit/layers/quantization/test_compressed_tensors_lm_head.py

新增 CPU 单元测试，覆盖 scheme 解析的全部关键分支与回归场景，是验证 target 匹配语义和 block 拒绝行为的主要依据。

```
# 单元测试：验证 get_lm_head_scheme 的 target 匹配规则
# CPU-only，注册到 base-a-test-cpu 套件。

class TestGetLmHeadScheme(CustomTestCase):
    # head 仅在 config target 按层名命中时才解析为量化 scheme；
    # 模块类型 target 与 ignore 的 head 保持未量化。

    def test_regex_target_resolves(self):
        # 正则 target 会命中（如 re:.*lm_head），并把匹配到的 target
        # 透传给 get_linear_scheme，避免下游 find_matched_target
        # 因后缀匹配无法二次推导而抛错。
        config = _config(["re:.*lm_head", "re:.*mlp\\.down_proj$"])
        head = _Head()
        with patch(_GET_LINEAR_SCHEME, return_value="scheme") as mock_resolve:
            scheme = config.get_lm_head_scheme(head, "lm_head")
        self.assertEqual(scheme, "scheme")
        mock_resolve.assert_called_once_with(
            layer=head, layer_name="lm_head", matched_target="re:.*lm_head"
        )

    def test_ignored_head_is_none(self):
        # ignore 列表中的 head 应保持未量化，且不调用 get_linear_scheme。
        config = _config(["re:.*lm_head"], ignore=["lm_head"])
        with patch(_GET_LINEAR_SCHEME) as mock_resolve:
            self.assertIsNone(config.get_lm_head_scheme(_Head(), "lm_head"))
        mock_resolve.assert_not_called()

    def test_module_type_target_is_none(self):
        # llm-compressor 会为 decoder 线性层输出 "Linear" 这类模块类型 target；
        # 未被提及的 head 必须留在未量化路径，而不是触发
        # find_matched_target 的未匹配层错误。
        config = _config(["Linear"])
        with patch(_GET_LINEAR_SCHEME) as mock_resolve:
            self.assertIsNone(config.get_lm_head_scheme(_Head(), "lm_head"))
        mock_resolve.assert_not_called()
```

评论区精华

审查评论界面没有留下可引用的文字记录（review comments 为空），但从提交历史可还原两轮 review 反馈：

- 第一轮实现未将 head 的 target 匹配结果传递到下游，导致 language_model.lm_head 这类带前缀的 head 在 find_matched_target 处抛 ValueError。提交 4c238895（"Address

review: share the head's target match downstream, reject block scales"）通过给 get_linear_scheme 增加 matched_target 参数解决了该问题，并补充 test_prefixed_head_with_plain_target_resolves 回归测试。

- 同一提交还回应了 block-FP8 头无法被 vocab-parallel 加载器分片的问题，改为显式报错。
- 提交 8a7fa13 补充了多 config group 命中 head 时"第一个 target 生效"的 first-match 规则说明。

最终 BBuf 批准："LGTM."。

- 后缀匹配 target 在下游会二次推导失败 (correctness): get_linear_scheme 新增 matched_target 参数并透传给 get_scheme_dict；新增 test_prefixed_head_with_plain_target_resolves 回归测试验证 CompressedTensorsW8A8Fp8 路径。
- block-FP8 头无法被 vocab-parallel 加载器分片 (correctness): get_lm_head_scheme 在检测到 block 策略时抛出带明确信息的错误；新增 test_block_quantized_head_is_rejected 用例。

风险与影响

- 风险：
  - 回归风险：未量化 head 的 checkpoint 不受影响，因为 get_lm_head_scheme 只会在 target 按层名命中时才返回方案，且 get_quant_method 的 ParallelLMHead 分支在 scheme 为 None 时原样返回 None。
  - 兼容性风险：block-FP8 量化的 head 会从"静默乱码"变为"加载时报错"，行为变化符合预期，但依赖这种乱码运行的用户会感知到中断。
  - 覆盖缺口：update_weights_from_disk 与量化 head 的组合未测试；无 prefix 构造的 ParallelLMHead 永远匹配不到 target（空层名不参与匹配）；speculative-decoding 的 draft 路径借用主模型 head 的场景被明确列为后续工作。
  - 性能风险：方案解析只在初始化执行一次，对未量化 head 无影响；量化 head 将 bf16 matmul 换成标准 FP8 linear 路径，理论上有助于降低延迟。
- 影响：
  - 用户影响：所有使用 compressed-tensors 格式且按层名量化了 lm_head 的模型从"完全不可用"变为"正确推理"，典型场景如 Qwen3.8-27B 混合精度 checkpoint；PR body 的 A/B 测试显示 GSM8K 从 0.000 提升到 0.955，无效输出占比从 85% 降到 0。
  - 系统影响：改动集中在量化配置解析路径，仅涉及 compressed_tensors.py 一个源码文件和一个测试文件，对调度、KV Cache、注意力等运行时路径无影响。
  - 团队影响：修复了被标记为 high priority 的 bug，明确了 compressed-tensors 下 lm_head 量化的支持边界（支持 per-channel/per-token，拒绝 block 量化），为后续支持其他量化格式提供了参考。
  - 风险标记：量化加载路径变更，block 量化兼容性变更，update_weights_from_disk 未覆盖

关联脉络

- 暂无明显关联 PR