PR #24027 完整报告

sgl-project/sglang

Bugfix

合并时间：2026-04-29 21:13

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/24027

执行摘要

- 一句话：修复 Mistral GQA 及 transformers v5 权重加载兼容性
- 推荐动作：建议核心开发者仔细审查权重重映射的正则字典，确保覆盖所有已知 Mistral 原生权重名。同时应增加相关单元的测试，当前仅依赖 pre-commit 和手动的 cookbook 验证。设计上选择基于 model_type 动态选择骨架，思路清晰，值得借鉴。

功能与动机

Pixtral / Mistral3 目前不支持使用普通 GQA 注意力的文本配置——语言模型分支硬编码为 MLA 路径，且 Mistral3ForConditionalGeneration 无法加载 transformers v5 权重布局的检查点。

实现拆解

1. 新增 Mistral GQA 权重重映射类：在 models/mistral.py 中添加 MistralForCausalLMMistralFormat，继承自 MistralForCausalLM（实际为 LlamaForCausalLM），通过正则字典将 Mistral 原生权重名映射为 HF/Llama 格式，并在 load_weights 中调用 _remap_mistral_to_llama。
2. Pixtral 语言模型动态选择：在 models/pixtral.py 的 __init__ 中，根据 text_config.model_type 是否为 "deepseek_v3" 决定使用 MistralLarge3ForCausalLM（MLA）或 MistralForCausalLMMistralFormat（GQA）。
3. Mistral3 多模态权重规范化：修改 Mistral3ForConditionalGeneration.load_weights，处理 transformers v5 的 model.language_model.X → language_model.model.X 等重映射，并委托给 LlavaForConditionalGeneration.load_weights。
4. 配置适配：
   - configs/model_config.py：通过 kv_lora_rank is not None 判断是否走 MLA 路径，不再对所有 PixtralForConditionalGeneration 硬编码。
   - utils/hf_transformers/mistral_utils.py：为非 MoE Mistral 设置 model_type="mistral" 和 rope_is_neox_style=False，移除空值的 MLA 字段。
   - utils/hf_transformers/common.py：在子配置中传播父配置的 dtype，避免 transformers v5 下 dtype 为 None 导致 fp16 下溢。
5. 更新文档：新增 Mistral Medium 3.5 cookbook 和部署交互组件。

关键文件：

- python/sglang/srt/models/mistral.py（模块 模型定义；类别 source；类型 core-logic；符号 MistralForCausalLMMistralFormat, load_weights, _remap_mistral_to_llama, normalize）：核心变更文件，新增 MistralForCausalLMMistralFormat 实现 Mistral 原生权重到 HF 格式的重映射，并修改 Mistral3ForConditionalGeneration.load_weights 以支持 transformers v5 布局。
- python/sglang/srt/models/pixtral.py（模块 模型定义；类别 source；类型 data-contract）：修改语言模型选择逻辑，根据 text_config 的 model_type 动态选用 MLA 或 GQA 骨架，实现 Pixtral 对两种注意力架构的支持。
- python/sglang/srt/utils/hf_transformers/mistral_utils.py（模块 配置适配；类别 source；类型 core-logic）：为非 MoE Mistral 配置设置 model_type 和 rope_is_neox_style，并移除空值 MLA 字段，避免影响形状推导。
- python/sglang/srt/utils/hf_transformers/common.py（模块 配置适配；类别 source；类型 core-logic）：修复 transformers v5 下子配置 dtype 为 None 的问题，避免视觉塔 fp16 溢出。
- python/sglang/srt/configs/model_config.py（模块 配置定义；类别 source；类型 data-contract）：停止无条件将 PixtralForConditionalGeneration 视为 MLA，改为通过 kv_lora_rank 判断。

关键符号：MistralForCausalLMMistralFormat.load_weights, MistralForCausalLMMistralFormat._remap_mistral_to_llama, PixtralForConditionalGeneration.init, Mistral3ForConditionalGeneration.load_weights, adapt_config_dict

关键源码片段

python/sglang/srt/models/mistral.py

核心变更文件，新增 MistralForCausalLMMistralFormat 实现 Mistral 原生权重到 HF 格式的重映射，并修改 Mistral3ForConditionalGeneration.load_weights 以支持 transformers v5 布局。

```
class MistralForCausalLMMistralFormat(MistralForCausalLM):
    """Mistral GQA model loaded from mistral native format (params.json).

    Handles weight name remapping from mistral native format to HF/Llama
    format. This is the GQA counterpart to MistralLarge3ForCausalLM which
    handles MLA models in mistral native format.
    """

    # fmt: off
    # 将 Mistral 原生参数名映射为 HF/Llama 格式的正则表达式字典
    remapping = {
        r"layers\.(\d+)\.attention_norm\.weight": r"model.layers.\1.input_layernorm.weight",
        r"layers\.(\d+)\.attention\.wq\.(\w+)": r"model.layers.\1.self_attn.q_proj.\2",
        r"layers\.(\d+)\.attention\.wk\.(\w+)": r"model.layers.\1.self_attn.k_proj.\2",
        r"layers\.(\d+)\.attention\.wv\.(\w+)": r"model.layers.\1.self_attn.v_proj.\2",
        r"layers\.(\d+)\.attention\.wo\.(\w+)": r"model.layers.\1.self_attn.o_proj.\2",
```

```
        r"layers\.(\d+)\.ffn_norm\.weight": r"model.layers.\1.post_attention_layernorm.weight",
        r"layers\.(\d+)\.feed_forward\.w1\.(\w+)": r"model.layers.\1.mlp.gate_proj.\2",
        r"layers\.(\d+)\.feed_forward\.w2\.(\w+)": r"model.layers.\1.mlp.down_proj.\2",
        r"layers\.(\d+)\.feed_forward\.w3\.(\w+)": r"model.layers.\1.mlp.up_proj.\2",
        r"norm\.weight": "model.norm.weight",
        r"tok_embeddings\.weight": "model.embed_tokens.weight",
        r"output\.weight": "lm_head.weight",
    }
    # fmt: on

    def load_weights(self, weights: Iterable[tuple[str, torch.Tensor]]):
        # 先通过 _remap_mistral_to_llama 转换权重名，再让父类加载
        return super().load_weights(self._remap_mistral_to_llama(weights))

    def _remap_mistral_to_llama(
        self, weights: Iterable[tuple[str, torch.Tensor]]
    ) -> Iterable[tuple[str, torch.Tensor]]:
        """Remap Mistral native format weight names to HF/Llama format."""
        for name, loaded_weight in weights:
            # 如果权重名已经符合 HF 格式（以 model. 或 lm_head. 开头），
            # 则直接透传，支持混合格式检查点（如原生权重 +HF 投影器）
            if name.startswith("model.") or name.startswith("lm_head."):
                yield name, loaded_weight
                continue

            # 逐一尝试正则匹配，找到则替换为 HF 格式
            for k, v in self.remapping.items():
                match = re.fullmatch(k, name)
                if match:
                    name = match.expand(v)
                    break
            else:
                # 未匹配到的权重名给出警告并跳过
                logger.warning(f"Unrecognized weight: {name}. Skipping.")
                continue

            # 处理量化缩放参数的命名差异
            if name.endswith(".qscale_act"):
                name = re.sub(r"\.qscale_act$", ".input_scale", name)
            elif name.endswith(".qscale_weight"):
                name = re.sub(r"\.qscale_weight$", ".weight_scale", name)

            yield name, loaded_weight
```

python/sglang/srt/models/pixtral.py

修改语言模型选择逻辑，根据 text_config 的 model_type 动态选用 MLA 或 GQA 骨架，实现 Pixtral 对两种注意力架构的支持。

```
    # 在 PixtralForConditionalGeneration.__init__ 中，
```

```
# 根据文本配置的注意力类型选择语言模型骨架
text_config = self.config.text_config
# 通过 model_type 判断是否 MLA：deepseek_v3 表示 MLA，否则为普通 GQA
is_mla = getattr(text_config, "model_type", "") == "deepseek_v3"
if is_mla:
    # MLA 骨架：使用 DeepSeek V3 风格的 MistralLarge3ForCausalLM
    self.language_model = MistralLarge3ForCausalLM(
        config=text_config,
        quant_config=kwargs.get("quant_config"),
    )
else:
    # GQA 骨架：使用标准 Llama 风格的 MistralForCausalLMMistralFormat
    self.language_model = MistralForCausalLMMistralFormat(
        config=text_config,
        quant_config=kwargs.get("quant_config"),
    )
```

python/sglang/srt/utils/hf_transformers/mistral_utils.py

为非 MoE Mistral 配置设置 model_type 和 rope_is_neox_style，并移除空值 MLA 字段，避免影响形状推导。

```
# 在 adapt_config_dict 中，非 MoE Mistral 分支下增加：
else:
    config_dict["architectures"] = ["MistralForCausalLM"]
    # 设置模型类型为 mistral，使得后续能区分 GQA 与 MLA
    config_dict["model_type"] = "mistral"
    # Mistral 使用非交错 RoPE（is_neox_style=False），不同于 Llama 默认的 True
    config_dict["rope_is_neox_style"] = False
    # 移除值为 None 的 MLA 字段，避免 getattr 返回 None 而非默认值
    for mla_key in (
        "q_lora_rank",
        "qk_rope_head_dim",
        "qk_nope_head_dim",
        "kv_lora_rank",
        "v_head_dim",
    ):
        if config_dict.get(mla_key) is None:
            config_dict.pop(mla_key, None)
```

评论区精华

在 review 中，gemini-code-assist[bot] 指出初始的 _remap_mistral_to_llama 会跳过已符合 HF/Llama 格式的权重，导致 transformers v5 检查点加载失败。作者随后在提交 d6b6fee 中添加了 name.startswith("model.") or name.startswith("lm_head.") 的直通路径，使该方法能够容忍混合格式的检查点。

- 权重重映射兼容 HF 格式 (correctness): 作者添加了 name.startswith("model.") 和 name.startswith("lm_head.") 的透传逻辑，使得混合格式权重能被正确加载。

风险与影响

- 风险：风险点包括：1）正则重映射覆盖有限，可能遗漏某些不规则权重名，但已通过直通 HF 格式和警告日志缓解；2）Pixtral 语言模型分支的改变影响所有 Pixtral 推理路径，缺乏测试覆盖增加了回归风险；3）*dtype* 传播修改可能影响其他多模态模型的精度。
- 影响：影响范围：主要影响 Mistral 系列和 Pixtral 系列模型的加载。用户现在可以部署 Mistral Medium 3.5 等 GQA 模型，以及 transformers v5 布局的检查点。对现有 MLA 路径无影响。团队需关注新增的 mistral.py 逻辑和 pixtral.py 分支。
- 风险标记：核心路径变更，缺少测试覆盖，配置依赖隐性

关联脉络

- 暂无明显关联 PR