PR #26106 完整报告

sgl-project/sglang

Support Command A plus

合并时间：2026-06-03 11:23

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/26106

执行摘要

- 一句话：Cohere Command A Plus 模型推理支持
- 推荐动作：该 PR 值得精读，特别是自定义 Centered LayerNorm、sigmoid topk 路由、混合 MoE 后端分派策略以及推理 / 工具调用解析器的状态机设计。对于想扩展新模型支持的开发者，这是很好的参考模式。建议后续增加测试覆盖。

功能与动机

集成 CohereLabs 最新发布的 Command A Plus 系列模型（BF16/FP8/NVFP4），使用户能够在 SGLang 上运行其多模态推理和工具调用场景。

实现拆解

1. 模型配置注册(configs/cohere2_moe.py)：定义 Cohere2MoeConfig，继承 PreTrainedConfig，设置模型参数并自动生成 layer_types 列表（滑动窗口与全注意力交替），注册到 CONFIG_MAPPING。
2. 文本核心模型(models/cohere2_moe.py)：实现 Cohere2MoeForCausalLM，包含自定义 Centered LayerNorm（去均值归一化，@torch.compile 优化）、Cohere2MoeAttention（可选 RoPE，仅滑动窗层）、Cohere2MoeSparseMoeBlock（共享专家 + sigmoid topk 路由专家，量化感知后端子选择）。
3. 多模态模型(models/cohere2_vision.py)：实现 Cohere2VisionForConditionalGeneration，使用 SigLIP 视觉编码器 + Cohere2VisionMultiModalProjector（pixel shuffle 下采样 + SwiGLU MLP），包含量化配置名重映射函数 _remap_quant_config_for_sglang。
4. 多模态处理器(multimodal/processors/cohere2_vision.py)：实现 Cohere2VisionSGLangImageProcessor，处理图像 token 展开和异步加载。
5. 推理解析器(parser/reasoning_parser.py)：新增 CohereCommand4Detector 类，使用状态机处理 <|START_THINKING|>/<|END_THINKING|>/<|START_TEXT|>/<|END_TEXT|> 标记，支持 streaming 模式下分离思考和最终答案。
6. 工具调用解析器(function_call/cohere_command4_detector.py)：新增 CohereCommand4Detector 类，解析 <|START_ACTION|>[...]<|END_ACTION|> 格式的 JSON 工具调用，支持流式缓冲和容错解析。
7. 入口连接：修改 function_call_parser.py、configs/init.py、configs/model_config.py 注册新解析器和配置，确保模型可被加载。

关键文件：

- python/sglang/srt/models/cohere2_moe.py（模块 模型层；类别 source；类型 core-logic；符号 Cohere2MoeForCausalLM, Cohere2MoeLayerNorm, cohere2_sigmoid_topk, Cohere2MoeSparseMoeBlock）：核心文本模型实现，包含自定义 LayerNorm、MoE、Attention，是模型推理的主入口。
- python/sglang/srt/models/cohere2_vision.py（模块 模型层；类别 source；类型 core-logic；符号 Cohere2VisionForConditionalGeneration, Cohere2VisionMultiModalProjector, pixel_shuffle, _remap_quant_config_for_sglang）：多模态模型实现，包含 pixel shuffle 下采样投影器，将 SigLIP 视觉特征映射到文本空间。
- python/sglang/srt/function_call/cohere_command4_detector.py（模块 解析器；类别 source；类型 dependency-wiring；符号 CohereCommand4Detector, _normalize_calls, detect_and_parse, parse_streaming_increment）：工具调用解析器，处理 <|START_ACTION|>[...]<|END_ACTION|> 格式的 JSON 工具调用。
- python/sglang/srt/parser/reasoning_parser.py（模块 解析器；类别 source；类型 core-logic；符号 CohereCommand4Detector, _strip_text_markers, detect_and_parse, parse_streaming_increment）：推理解析器新增 CohereCommand4Detector 类，处理 <|START_THINKING|> 等标记，支持 reasoning 模式。
- python/sglang/srt/configs/cohere2_moe.py（模块 配置层；类别 source；类型 dependency-wiring；符号 Cohere2MoeConfig, post_init）：模型配置定义，继承 PreTrainedConfig，自动生成 layer_types，注册到 CONFIG_MAPPING。
- python/sglang/srt/multimodal/processors/cohere2_vision.py（模块 多模态处理；类别 source；类型 core-logic；符号 Cohere2VisionSGLangImageProcessor, process_mm_data_async）：多模态图像处理器，处理图像 token 展开和异步加载。
- python/sglang/srt/function_call/function_call_parser.py（模块 函数调用；类别 source；类型 dependency-wiring）：注册 CohereCommand4Detector 到函数调用检测注册表。
- python/sglang/srt/configs/model_config.py（模块 配置层；类别 source；类型 data-contract）：注册 cohere2_moe 模型类型映射。
- python/sglang/srt/configs/__init__.py（模块 配置层；类别 source；类型 dependency-wiring）：导出 Cohere2MoeConfig。

关键符号：_cohere_layer_norm, cohere2_sigmoid_topk, Cohere2MoeLayerNorm.forward, Cohere2MoeAttention.forward, Cohere2MoeSparseMoeBlock.forward, Cohere2MoeForCausalLM.forward, Cohere2VisionMultiModalProjector.forward, Cohere2VisionForConditionalGeneration.forward, CohereCommand4Detector.detect_and_parse, CohereCommand4Detector.parse_streaming_increment, Cohere2VisionSGLangImageProcessor.process_mm_data_async

关键源码片段

python/sglang/srt/models/cohere2_moe.py

核心文本模型实现，包含自定义 LayerNorm、MoE、Attention，是模型推理的主入口。

```
# SPDX-License-Identifier: Apache-2.0
# Copyright 2026 SGLang Team
# Adapted from vLLM v0.21.0
"""Cohere2Moe 模型：自定义 centered layer norm 与 sigmoid-topk 路由。"""

@torch.compile(backend=get_compiler_backend())
def _cohere_layer_norm(hidden_states, weight, variance_epsilon):
    input_dtype = hidden_states.dtype
    hidden_states = hidden_states.to(torch.float32)
    mean = hidden_states.mean(-1, keepdim=True)
    variance = (hidden_states - mean).pow(2).mean(-1, keepdim=True)
    hidden_states = (hidden_states - mean) * torch.rsqrt(variance + variance_epsilon)
    hidden_states = weight.to(torch.float32) * hidden_states
    return hidden_states.to(input_dtype)

class Cohere2MoeLayerNorm(nn.Module):
    def __init__(self, hidden_size, eps=1e-5):
        super().__init__()
        self.weight = nn.Parameter(torch.ones(hidden_size))
        self.variance_epsilon = eps
    def forward(self, hidden_states):
        return _cohere_layer_norm(hidden_states, self.weight, self.variance_epsilon)

def cohere2_sigmoid_topk(hidden_states, gating_output, topk, renormalize):
    """sigmoid 激活后进行 topk 选择，可选重归一化。
    常见 MoE 使用 softmax + topk，Cohere Command A Plus 使用 sigmoid + topk。
    """
    scores = gating_output.float().sigmoid()
    topk_weights, topk_ids = torch.topk(scores, k=topk, dim=-1, sorted=False)
    if renormalize:
        topk_weights = topk_weights / topk_weights.sum(dim=-1, keepdim=True)
    return topk_weights.to(torch.float32), topk_ids.to(torch.int32)
```

python/sglang/srt/models/cohere2_vision.py

多模态模型实现，包含 pixel shuffle 下采样投影器，将 SigLIP 视觉特征映射到文本空间。

```
class Cohere2VisionMultiModalProjector(nn.Module):
    """Pixel-shuffle 下采样 + SwiGLU MLP，将视觉特征映射到文本 hidden_size。"""
    def __init__(self, config: PretrainedConfig):
        super().__init__()
        self.downsample_factor = config.downsample_factor
        input_dim = config.vision_config.hidden_size * (config.downsample_factor ** 2)
        self.intermediate_size = config.alignment_intermediate_size // 2
        # HF 的 linear_1 是 SwiGLU 门控 + 值融合的线性层，SGLang 用
        MergedColumnParallelLinear 表示两个 shard
        self.linear_1 = MergedColumnParallelLinear(
            input_dim, [self.intermediate_size] * 2, bias=True
        )
        self.linear_2 = RowParallelLinear(
```

```
            self.intermediate_size, config.text_config.hidden_size, bias=True
        )

    def pixel_shuffle(self, image_features):
        """将 (B, S, D) 按空间分辨率 H=W=sqrt(S) reshape，然后按 downsample_factor
        降低空间分辨率、增加通道。"""
        B, S, D = image_features.shape
        H = W = int(math.isqrt(S))
        image_features = image_features.reshape(B, H, W, D)
        # 具体 reshape/permute 实现细节略
        return image_features # shape: (B, H//down, W//down, D*down^2)

    def forward(self, image_features):
        image_features = self.pixel_shuffle(image_features)
        B, H, W, D = image_features.shape
        image_features = image_features.reshape(B, H * W, D)
        gate_up, _ = self.linear_1(image_features)
        x, gate = gate_up.chunk(2, dim=-1)
        hidden_states = x * F.silu(gate)
        hidden_states, _ = self.linear_2(hidden_states)
        return hidden_states
```

评论区精华

审查中没有出现讨论线程，PR 直接被两位 reviewer (AgainstEntropy, mickqian) 批准。社区成员 AgainstEntropy 在 issue 中提醒 BF16 检查点需使用正确的 preprocessor_config.json。

- No review discussions (other): 无需修改直接合并。

风险与影响

- 风险：
  1. 缺少测试覆盖：新增模型和解析器没有对应的单元测试，回归风险较高。
  2. 量化路径特化：FP8 和 NVFP4 检查点需要特定的后端支持和权重映射，Blackwell 上曾因 Triton 不支持量化输入而出现 bug（commit 469ee3e3）。
  3. MoE 路由差异：sigmoid+topk 路由与常见的 softmax+topk 不同，可能影响负载均衡和模型精度，需验证。
  4. 解析器干扰：新增 CohereCommand4Detector 与已有解析器可能共存但无冲突检测，若同时使用多个格式检测可能有优先级问题。 - 影响：对用户：Cohere Command A Plus 用户可在 SGLang 上使用文本推理、多模态推理和工具调用功能，支持 BF16/FP8/NVFP4 多种量化格式。对系统：新增约 1.4k 行代码，集中于新模型模块，不影响已有模型性能或调度逻辑。对团队：需维护新增的配置映射和量化适配，未来模型更新时需同步。 - 风险标记：缺少测试覆盖，量化路径特化，MoE 路由差异，解析器干扰

关联脉络

- 暂无明显关联 PR