PR #24436 完整报告

sgl-project/sglang

[Gemma 4] Adding MTP support

合并时间：2026-05-08 05:08

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/24436

执行摘要

- 一句话：为 Gemma4 添加 FROZEN_KV_MTP 投机解码算法
- 推荐动作：此 PR 对于 Gemma 4 用户至关重要，值得精读。设计上选择冻结 KV 方案而非传统 EAGLE 是合理的。关注点是 TP>1 支持尚未完全验证，数值掩码稳定性有待改进。建议后续跟进 TP 测试和掩码修复。

功能与动机

Each Gemma 4 target ships with a small 'assistant' checkpoint trained for MTP. This PR introduces a new speculative algorithm — FROZEN_KV_MTP — that runs the assistant against the target's KV cache (the assistant has no KV of its own and a recurrent hidden state across draft steps, so it does not fit cleanly under EAGLE/EAGLE3 or NEXTN).

实现拆解

步骤 1：助理模型定义 在 python/sglang/srt/models/gemma4_mtp.py 中新增 Gemma4AssistantForCausalLM，继承自 Gemma4ForCausalLM。模型通过目标嵌入（target embed）和预投影 / 后投影实现循环隐藏状态，拥有自己的 lm_head。bind_frozen_kv_context 方法将助理逻辑层映射到目标物理层。

步骤 2：投机算法与数据模型 在 spec_info.py 中新增 FROZEN_KV_MTP 枚举值和 is_frozen_kv_mtp 判别方法。frozen_kv_mtp_info.py 定义了 FrozenKVMTPContext（存储目标 KV 池和层映射）、FrozenKVMTPDraftInput 和 FrozenKVMTPVerifyInput，后两者复用 EAGLE 的调度合约。

步骤 3：工作器实现 frozen_kv_mtp_worker.py 中的 FrozenKVMTPWorker 继承 TpModelWorker，负责草稿循环。它借用目标的 req_to_token_pool 和 KV 分配器（只读），强制禁用 CUDA 图嵌入和覆盖调度。_resolve_draft_backend_type 方法选择注意力后端。FrozenKVMTPWorkerV2 暂未实现，使用时必须 --disable-overlap-schedule。

步骤 4：CUDA 图支持 frozen_kv_mtp_cuda_graph_runner.py 中的 FrozenKVMTPCudaGraphRunner 支持 topk=1 的简单模式和 topk>1 的树验证模式。_capture_graph 方法捕捉整个草稿步骤，_replay 实现低延迟回放。同时定义 FrozenKVMTPInputBuffers 数据类传递输入张量。

步骤 5：服务器配置与自动提升 server_args.py 新增 _resolve_speculative_algorithm_alias，当草稿模型架构为 Gemma4AssistantForCausalLM 时自动将 NEXTN/EAGLE 提升为 FROZEN_KV_MTP，并拒绝 EAGLE3。当算法激活时，强制禁用 overlap_scheduler 和 mixed_chunked_prefill，默认 max-running-requests 为 48。

步骤 6：现有模型适配 gemma4_causal.py 和 gemma4_mm.py 暴露 get_embed_and_head 方法，供助理在加载时重新绑定到目标输入嵌入。hf_transformers/config.py 识别 gemma4_assistant 类型。

关键文件：

- python/sglang/srt/models/gemma4_mtp.py（模块 模型层；类别 source；类型 data-contract；符号 Gemma4AssistantForCausalLM, init, bind_frozen_kv_context, build_frozen_kv_mtp_context）：核心助理模型定义，包括 Gemma4AssistantForCausalLM 及其冻结 KV 绑定逻辑
- python/sglang/srt/speculative/frozen_kv_mtp_worker.py（模块 投机解码；类别 source；类型 dependency-wiring；符号 FrozenKVMTPWorker, init, draft_model_runner, get_attn_backend）：FrozenKVMTPWorker 实现草稿生成循环，协调目标 KV 访问和树验证
- python/sglang/srt/speculative/frozen_kv_mtp_cuda_graph_runner.py（模块 投机解码；类别 source；类型 dependency-wiring；符号 FrozenKVMTPInputBuffers, FrozenKVMTPCudaGraphRunner, init, can_run）：CUDA 图运行器实现低延迟草稿步骤捕获与回放
- python/sglang/srt/speculative/frozen_kv_mtp_utils.py（模块 投机解码；类别 source；类型 dependency-wiring；符号 frozen_kv_target_view, target_kv_pool_view, set_frozen_kv_positions, expand_for_topk_draft）：提供上下文管理器（冻结 KV 视图、目标 KV 池视图）和批处理扩展工具
- python/sglang/srt/speculative/frozen_kv_mtp_info.py（模块 投机解码；类别 source；类型 dependency-wiring；符号 FrozenKVMTPContext, get_physical_layer_id, FrozenKVMTPDraftInput, post_init）：定义 FrozenKVMTPContext、FrozenKVMTPDraftInput、FrozenKVMTPVerifyInput 等数据结构
- python/sglang/srt/speculative/frozen_kv_mtp_worker_v2.py（模块 投机解码；类别 source；类型 dependency-wiring；符号 FrozenKVMTPWorkerV2, init）：占位符 V2 工作器，当前抛出 NotImplementedError
- python/sglang/srt/server_args.py（模块 配置；类别 source；类型 dependency-wiring；符号 _resolve_speculative_algorithm_alias）：修改服务器参数解析，添加算法别名解析和强制配置
- python/sglang/srt/speculative/spec_info.py（模块 投机解码；类别 source；类型 dependency-wiring；符号 is_frozen_kv_mtp）：新增 FROZEN_KV_MTP 算法枚举和判别方法
- python/sglang/srt/models/gemma4_mm.py（模块 模型层；类别 source；类型 data-contract；符号 get_embed_and_head）：暴露 get_embed_and_head 方法供助理模型绑定

- python/sglang/srt/models/gemma4_causal.py（模块 模型层；类别 source；类型 data-contract；符号 get_embed_and_head）：暴露 get_embed_and_head 方法供助理模型绑定
- python/sglang/srt/utils/hf_transformers/config.py（模块 配置；类别 source；类型 core-logic）：识别 gemma4_assistant model_type

关键符号：Gemma4AssistantForCausalLM.init, Gemma4AssistantForCausalLM.get_embed_and_head, Gemma4AssistantForCausalLM.bind_frozen_kv_context, FrozenKVMTPWorker.init, FrozenKVMTPWorker.draft_model_runner, FrozenKVMTPCudaGraphRunner.init, FrozenKVMTPCudaGraphRunner._capture_graph, FrozenKVMTPCudaGraphRunner._replay, _resolve_speculative_algorithm_alias, is_frozen_kv_mtp, frozen_kv_target_view, expand_for_topk_draft, set_frozen_kv_positions

关键源码片段

python/sglang/srt/models/gemma4_mtp.py

核心助理模型定义，包括 Gemma4AssistantForCausalLM 及其冻结 KV 绑定逻辑

```
class Gemma4AssistantForCausalLM(Gemma4ForCausalLM):
    """Gemma 4 MTP 助理模型：使用目标嵌入 + 循环隐藏状态，拥有自己的 lm_head。"""

    base_model_prefix = "model"

    def __init__(self, config: PretrainedConfig, quant_config: Optional[QuantizationConfig] =
    None, prefix: str = "") -> None:
        # 深拷贝文本配置并禁用 KV 共享（助理不管理 KV）
        text_config = copy.deepcopy(_get_text_config(config))
        text_config.num_kv_shared_layers = 0
        PreTrainedModel.__init__(self, config=text_config)
        self.assistant_config = config
        self.config = text_config
        self.quant_config = quant_config

        self.vocab_size = text_config.vocab_size
        self.hidden_size = text_config.hidden_size
        # backbone_hidden_size 来自助理配置，是目标嵌入的维度
        self.backbone_hidden_size = config.backbone_hidden_size
        # 目标嵌入缩放因子，用于将目标嵌入投影到助理隐藏空间
        self.target_embed_scale = self.backbone_hidden_size ** 0.5
        self.use_ordered_embeddings = getattr(config, "use_ordered_embeddings", False)
        self.centroid_intermediate_top_k = int(getattr(config, "centroid_intermediate_top_k", 32))

        # 目标嵌入权重将在加载时由 bind_frozen_kv_context 绑定
        self.target_embed_weight = None

        # 预投影：拼接目标嵌入和循环隐藏状态
        self.pre_projection = ReplicatedLinear(2 * self.backbone_hidden_size, self.hidden_size,
```

```
        bias=False, quant_config=None)
        # 助理骨干网络，复用 Gemma4TextModel
        self.model = Gemma4TextModel(config=text_config, quant_config=quant_config, prefix=
        add_prefix("model", prefix))
        # 后投影：将骨干输出映射回 backbone_hidden_size
        self.post_projection = ReplicatedLinear(self.hidden_size, self.backbone_hidden_size, bias=
        False, quant_config=None)

        # 语言模型头：如果词汇嵌入共享则绑定到 embed_tokens
        if text_config.tie_word_embeddings:
            self.lm_head = self.model.embed_tokens
        else:
            self.lm_head = nn.Linear(self.hidden_size, self.vocab_size, bias=False)
        self.logits_processor = LogitsProcessor(text_config, skip_all_gather=True)

        # 如果启用有序嵌入（centroid），则配置 centroid 词汇头 ...（后续代码省略）
```

python/sglang/srt/speculative/frozen_kv_mtp_worker.py

FrozenKVMTPWorker 实现草稿生成循环，协调目标 KV 访问和树验证

```
class FrozenKVMTPWorker(TpModelWorker):
    """Frozen-KV MTP 工作器。助理只读目标 KV，重复使用 EAGLE 的验证合约。"""

    def __init__(self, server_args, gpu_id, tp_rank, dp_rank, moe_ep_rank, attn_cp_rank, moe_dp_
    rank, nccl_port, target_worker):
        self.server_args = server_args
        self.topk = server_args.speculative_eagle_topk
        self.speculative_num_steps = server_args.speculative_num_steps
        self.target_worker = target_worker

        # 确保算法类型正确
        assert self.speculative_algorithm.is_frozen_kv_mtp()

        # 助理上下文长度必须与目标一致（副作用警告）
        server_args.context_length = target_worker.model_runner.model_config.context_len

        # 禁用 CUDA 图（我们自己管理）
        backup_disable_cuda_graph = server_args.disable_cuda_graph
        server_args.disable_cuda_graph = True

        # 复用目标的内存池（只读）
        self.req_to_token_pool, self.token_to_kv_pool_allocator = target_worker.get_memory_pool()

        # 配置草稿注意力后端 ...（后续代码省略）
```

python/sglang/srt/speculative/frozen_kv_mtp_cuda_graph_runner.py

CUDA 图运行器实现低延迟草稿步骤捕获与回放

```
@dataclass
```

```
class FrozenKVMTPInputBuffers(ForwardInputBuffers):
    req_pool_indices: torch.Tensor
    positions: torch.Tensor
    seq_lens: torch.Tensor
    hidden_states: torch.Tensor # 循环隐藏状态
    topk_p: torch.Tensor # 树验证概率
    topk_index: torch.Tensor # 树验证索引


class FrozenKVMTPCudaGraphRunner:
    """CUDA 图运行器，用于 Frozen-KV MTP 的循环草稿步骤。"""

    def __init__(self, frozen_kv_mtp_worker):
        self.model_runner = frozen_kv_mtp_worker.draft_model_runner
        self.speculative_num_steps = self.model_runner.server_args.speculative_num_steps
        self.topk = self.model_runner.server_args.speculative_eagle_topk
        self.num_tokens_per_bs = self.topk
        # 获取待捕获的 batch sizes
        self.capture_bs, self.compile_bs = get_batch_sizes_to_capture(model_runner, self.num_
        tokens_per_bs)
        self.max_bs = max(self.capture_bs)
        self.max_num_token = self.max_bs * self.num_tokens_per_bs
        # 初始化输入缓冲区并分配 CUDA 张量 ...（后续代码省略）
```

评论区精华

1. TP>1 支持：gemini-code-assist[bot] 指出助理模型的嵌入查找和 centroid 掩码 gather 在 TP>1 时会崩溃。作者 kpham-sgl 回应 self.target_embed_weight 不是 TP 分片的，所以问题不存在，但未完全解决其他 gather 问题。
2. 数值掩码稳定性：review 指出 selected_logits.min() - 1.0 作为掩码值可能因 -inf 产生 nan，建议使用 -1e10。作者未回复，该问题可能未被采纳。
3. 服务器参数副作用：review 指出 server_args.context_length 原地突变是坏实践。作者回应 "Same pattern in Eagle"，沿用现有模式。
4. 信任远程代码：review 指出 trust_remote_code=True 被硬编码，应尊重用户 --trust-remote-code 标志。未看到修改。
5. 测试注册：Qiaolin-Yu 询问是否将手动测试注册到 CI。作者回复将写一个较小的测试，后续在提交中添加了 CI 测试。

- TP>1 时嵌入查找错误 (correctness): 作者回复 target_embed_weight 不是 TP 分片的，忽略该问题。但 centroid 掩码 gather 和词汇重排序在 TP>1 时可能仍会崩溃。
- centroid 掩码 gather 在 TP>1 时崩溃 (correctness): 作者回复 'ditto'，认为是相同原因忽略。
- 词汇重排序在 TP>1 时索引越界 (correctness): 作者回复 'ditto'。
- trust_remote_code 硬编码 (security): 未看到修改，可能维持硬编码。
- 掩码值数值不稳定 (correctness): 作者未回复，可能未采纳。

- server_args.context_length 原地突变 (design): 作者回复 'Same pattern in Eagle'，沿用现有模式。
- CUDA 图运行器使用通用 Exception (style): 未看到修改。
- 测试注册到 CI (testing): 已添加 CI 测试但后续因依赖版本暂移除，最终状态待定。

风险与影响

- 风险：
  1. TP>1 兼容性：尽管作者声称某些张量不是 TP 分片的，但 centroid 掩码 gather 和词汇重排序逻辑在 TP>1 时仍可能崩溃。需要针对 TP>1 进行专门测试。
  2. 数值稳定性：掩码值可能因 -inf 导致 nan，影响采样质量。
  3. 服务器参数副作用：server_args.context_length 被突变，可能引起日志混乱和调试困难。
  4. 通用异常：CUDA 图运行器中抛出的泛型 Exception 可能导致错误信息模糊。
  5. 覆盖调度器强制禁用：FROZEN_KV_MTP 强制禁用覆盖调度器，这可能影响与其他工作负载的兼容性。
- 影响：
  1. 用户影响：Gemma 4 用户可通过简单参数启用 MTP 加速，获得接近无损的准确度。但需要手动升级 Transformers 和设置正确参数。
  2. 系统影响：新投机算法增加了调度器分支，强制禁用某些优化（overlap schedule、mixed chunked prefill），可能影响整体性能。
  3. 团队影响：需要维护新的算法代码路径，特别是 TP 扩展和后续性能优化。 - 风险标记：TP>1 兼容性未验证，数值掩码稳定性风险，服务器参数副作用，通用异常抛出，覆盖调度器强制禁用

关联脉络

- PR #26335 [Spec] Async-assert probes across EAGLE/MTP; zero tgt_cache_loc: 同样属于 speculative decoding 基础设施，增强了 EAGLE/MTP 的异步断言和 NaN 检测，与本 PR 的 MTP 实现有间接关联。
- PR #26397 Reland "[perf][spec decoding] Skip full-vocab softmax in EAGLE draft when topk == 1 (#26235)": EAGLE 草稿性能优化，与本 PR 的草稿生成逻辑可能共享部分工具函数。