PR #39141 完整报告

vllm-project/vllm

[Perf] Update TRTLLM supported MoE routing methods

合并时间：2026-04-28 02:16

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/39141

执行摘要

- 一句话：更新 TRTLLM MoE 路由枚举，新增 SigmoidRenorm 和 MiniMax2
- 推荐动作：值得精读，特别是在枚举分类和路由方法检测逻辑上的设计决策，以及如何系统性地更新所有专家内核的支持列表。对 DeepSeek、MiniMax 模型部署和维护有兴趣的工程师应重点关注。

功能与动机

FlashInfer v0.6.8 recently adds more support for different MoE routing methods (e.g., minimax m2). This PR updates the routing methods enums corresponding to flashinfer, as well as the supported routing methods in Trtllm MoE fp8 and nvfp4. In addition, the requirement for fp32 router logits for deepseek MoE is now removed.

实现拆解

1. 更新 RoutingMethodType 枚举：在 vllm/model_executor/layers/fused_moe/config.py 中新增 SigmoidRenorm（sigmoid → TopK → renormalize）和 MiniMax2（sigmoid + bias → TopK → scaled sum normalize），并将 Custom、Simulated、DeepseekV4 重新分类为内部类型（ID ≥ 100），确保这些值不会直接传入 FlashInfer 内核。
2. 扩展路由推导逻辑：在 get_routing_method_type 函数中增加 has_e_score_bias + sigmoid → MiniMax2 的分支，以及 sigmoid + renormalize → SigmoidRenorm 的分支，确保新路由方法能被正确识别。
3. 同步专家内核支持列表：在 vllm/model_executor/layers/fused_moe/experts/trtllm_fp8_moe.py 和 trtllm_nvfp4_moe.py 的 _supports_routing_method 静态方法中添加 SigmoidRenorm 和 MiniMax2；同时将 _supports_router_logits_dtype 简化为始终返回 True，不再对 fp32 router logits 进行特殊限制。
4. 清理 apply 方法：移除 FP8 和 NVFP4 内核中针对 autotuning 的跳过逻辑（trtllm 内核现已支持 autotuning），移除 FP8 内核中因 FlashInfer bug 而需要的 output.copy_(result) workaround（FlashInfer 已修复原地写入）。在 NVFP4 内核中，移除对 router_logits 的显式 fp32 转换，因为内核已支持 bfloat16。
5. 简化 DeepSeek 模型适配：在 vllm/model_executor/models/deepseek_v2.py 中移除对 RoutingMethodType 的导入和强制 gate 输出为 fp32 的逻辑，因为 monolithic 内核不再需要特殊处理。

关键文件：

- vllm/model_executor/layers/fused_moe/config.py（模块 路由配置；类别 source；类型 data-contract；符号 RoutingMethodType, get_routing_method_type）：核心定义：新增 SigmoidRenorm 和 MiniMax2 路由枚举，重新分类内部类型，更新 get_routing_method_type 映射逻辑。
- vllm/model_executor/layers/fused_moe/experts/trtllm_fp8_moe.py（模块 FP8 专家；类别 source；类型 data-contract；符号 TrtLlmFp8ExpertsBase.apply, TrtLlmFp8ExpertsMonolithic.supports_routing_method, TrtLlmFp8ExpertsMonolithic._supports_router_logits_dtype）：FP8 专家内核：更新 _supports_routing_method 列表，简化 _supports_router_logits_dtype，移除 autotuning 跳过和 output.copy workaround。
- vllm/model_executor/layers/fused_moe/experts/trtllm_nvfp4_moe.py（模块 NVFP4 专家；类别 source；类型 data-contract；符号 TrtLlmNvFp4ExpertsBase.apply, TrtLlmNvFp4ExpertsMonolithic._supports_routing_method, TrtLlmNvFp4ExpertsMonolithic._supports_router_logits_dtype, TrtLlmNvFp4ExpertsMonolithic.apply）：NVFP4 专家内核：类似 FP8，更新路由支持列表，简化 dtype 检查，移除 router_logits 的 fp32 转换。
- vllm/model_executor/models/deepseek_v2.py（模块 DeepSeek 模型；类别 source；类型 data-contract；符号 DeepseekV2ForCausalLM.init）：DeepSeek 模型：移除对 RoutingMethodType 的导入和 gate.set_out_dtype(fp32) 强制逻辑，因为这些已由内核统一处理。
- vllm/model_executor/layers/fused_moe/oracle/nvfp4.py（模块 回溯选择；类别 source；类型 data-contract）：回溯选择器：移除一行多余的 logger.info_once 调用，属于清理。

关键符号：TrtLlmFp8ExpertsBase.apply, TrtLlmFp8ExpertsMonolithic._supports_routing_method, TrtLlmFp8ExpertsMonolithic._supports_router_logits_dtype, TrtLlmNvFp4ExpertsBase.apply, TrtLlmNvFp4ExpertsMonolithic._supports_routing_method, TrtLlmNvFp4ExpertsMonolithic._supports_router_logits_dtype, TrtLlmNvFp4ExpertsMonolithic.apply, get_routing_method_type, DeepseekV2ForCausalLM.init

关键源码片段

vllm/model_executor/layers/fused_moe/config.py

核心定义：新增 SigmoidRenorm 和 MiniMax2 路由枚举，重新分类内部类型，更新 get_routing_method_type 映射逻辑。

```
# 文件：vllm/model_executor/layers/fused_moe/config.py
# 关键变更：新增路由枚举并重新分类

class RoutingMethodType(IntEnum):
    # 以下是传给 FlashInfer 内核的标准类型（ID 0-8）
    Default = (0,)
    Renormalize = (1,)
```

```
    DeepSeekV3 = (2,)
    Llama4 = (3,)
    RenormalizeNaive = (4,)
    TopK = (5,)
    # 新增：Sigmoid -> TopK -> Renormalize（除以 Top-K 之和）
    SigmoidRenorm = (6,)
    # 新增：Sigmoid + Bias -> TopK -> ScaledSumNormalize（用于 MiniMax M2 等）
    MiniMax2 = (7,)
    Unspecified = (8,)

    # 以下为 vLLM 内部路由类型，不直接传入 FlashInfer 内核
    DeepseekV4 = (100,) # sqrtsoftplus + Bias + Normalize
    Custom = (101,)
    Simulated = (102,)


def get_routing_method_type(
    scoring_func: str,
    top_k: int,
    renormalize: bool,
    num_expert_group: int | None,
    has_e_score_bias: bool,
) -> RoutingMethodType:
    # DeepSeek V4 专用
    if scoring_func == "sqrtsoftplus":
        return RoutingMethodType.DeepseekV4 if renormalize else RoutingMethodType.
        Unspecified

    # 有 e_score_correction_bias 时
    if has_e_score_bias:
        if (num_expert_group or 0) > 0 and scoring_func == "sigmoid":
            return RoutingMethodType.DeepSeekV3 # 分组 TopK
        elif scoring_func == "sigmoid":
            return RoutingMethodType.MiniMax2 # 新增：MiniMax2 路由
        else:
            return RoutingMethodType.Unspecified

    # 无 bias 的 sigmoid
    if scoring_func == "sigmoid":
        if top_k == 1:
            return RoutingMethodType.Llama4
        elif renormalize:
            return RoutingMethodType.SigmoidRenorm # 新增：SigmoidRenorm
        else:
            return RoutingMethodType.Unspecified

    # 标准 softmax 路由
    if scoring_func == "softmax":
        return RoutingMethodType.Renormalize if renormalize else RoutingMethodType.Default
```

```
        return RoutingMethodType.Unspecified
```

vllm/model_executor/layers/fused_moe/experts/trtllm_fp8_moe.py

FP8 专家内核：更新 supports_routing_method 列表，简化 _supports_router_logits_dtype，移除 autotuning 跳过和 output.copy workaround。

```
# 文件：vllm/model_executor/layers/fused_moe/experts/trtllm_fp8_moe.py
# 关键变更：TrtLlmFp8ExpertsBase.apply 方法简化

class TrtLlmFp8ExpertsBase(...):
    def apply(
        self,
        output: torch.Tensor,
        hidden_states: torch.Tensor,
        w1: torch.Tensor,
        w2: torch.Tensor,
        topk_weights: torch.Tensor,
        topk_ids: torch.Tensor,
        activation: MoEActivation,
        global_num_experts: int,
        expert_map: torch.Tensor | None,
        a1q_scale: torch.Tensor | None,
        a2_scale: torch.Tensor | None,
        workspace13: torch.Tensor,
        workspace2: torch.Tensor,
        expert_tokens_meta: mk.ExpertTokensMetadata | None,
        apply_router_weight_on_input: bool,
    ):
        import flashinfer
        from flashinfer.fused_moe import Fp8QuantizationType, WeightLayout

        packed_topk_ids = trtllm_moe_pack_topk_ids_weights(topk_ids, topk_weights)

        # 移除了 autotuning 跳过逻辑：trtllm 内核已支持 autotuning
        # 移除了 docstring 中过时的注释

        assert a1q_scale is not None

        is_mxfp8 = self.quant_config.block_shape == [1, 32]
        if is_mxfp8:
            fp8_quant_type = Fp8QuantizationType.MxFp8
            use_shuffled_weight = True
            weight_layout = WeightLayout.MajorK
            hidden_states_scale = a1q_scale
        else:
            fp8_quant_type = Fp8QuantizationType.DeepSeekFp8
            use_shuffled_weight = True
            weight_layout = WeightLayout.BlockMajorK
            hidden_states_scale = a1q_scale.t().contiguous()
```

```
    # 直接传入 output 参数，FlashInfer 已修复原地写入 bug
    flashinfer.fused_moe.trtllm_fp8_block_scale_routed_moe(
        topk_ids=packed_topk_ids,
        routing_bias=None,
        hidden_states=hidden_states,
        hidden_states_scale=hidden_states_scale,
        gemm1_weights=w1,
        gemm1_weights_scale=self.quant_config.w1_scale,
        gemm2_weights=w2,
        gemm2_weights_scale=self.quant_config.w2_scale,
        num_experts=global_num_experts,
        top_k=self.topk,
        n_group=None,
        topk_group=None,
        intermediate_size=self.intermediate_size_per_partition,
        local_expert_offset=self.ep_rank * self.local_num_experts,
        local_num_experts=self.local_num_experts,
        routed_scaling_factor=None,
        routing_method_type=1, # 不活跃（保留参数）
        use_shuffled_weight=use_shuffled_weight,
        weight_layout=weight_layout,
        fp8_quantization_type=fp8_quant_type,
        output=output, # 直接传入，无需后续 copy
    )
```

评论区精华

- Simulated 路由是否应包含在 monolithic 内核支持列表中：*gemini-code-assist[bot]* 指出 RoutingMethodType.Simulated 是内部类型（ID=102），不应出现在直接传给 FlashInfer 的 monolithic 内核列表中，否则可能传递无效值。最终代码在 FP8 block-scale 分支中保留了 Simulated，per-tensor 分支中已移除，NVFP4 中同样保留。该问题未完全解决，存在潜在风险。
- IntEnum 成员使用元组值：*gemini-code-assist[bot]* 提醒新增的 SigmoidRenorm = (6,) 等应使用纯整数而非元组，以避免类型错误。但 PR 未修改（遗留代码也使用元组），属于风格问题，未修正。
- Docstring 与代码不一致：*gemini-code-assist[bot]* 发现 `_supports_router_logits_dtype` 的 docstring 仍提及已移除的 `Simulated` 支持。作者在最终版本中直接移除了 docstring（函数体简化为 `return True`），该问题已解决。
  - Simulated 路由在 monolithic 内核中的支持问题 (correctness): 最终代码中，FP8 per-tensor 分支已移除 Simulated，但 FP8 block-scale 和 NVFP4 分支仍保留。作者未明确回复，PR 仍被合并。存在将内部 ID 传入 FlashInfer 的风险。
  - RoutingMethodType IntEnum 成员使用元组值 (style): PR 未修改此问题（遗留代码也使用元组）。虽不直接影响正确性，但不符合 IntEnum 惯例。
  - _supports_router_logits_dtype docstring 不一致 (documentation): 最终代码直接删除了 docstring（函数体简化为 return True），问题已解决。

风险与影响

- 风险：
  1. Simulated 路由可能传入 FlashInfer：在 trtllm_fp8_moe.py 的 block-scale 分支和 trtllm_nvfp4_moe.py 中，RoutingMethodType.Simulated 仍被列入支持列表。由于 Simulated 的枚举值为 102（内部类型），FlashInfer 内核可能无法处理此值，导致未定义行为或崩溃。当前未验证是否有模型实际使用此路由方法。
  2. autotuning 跳过逻辑移除：之前因为 trtllm 内核不支持 autotuning 而跳过 dummy run，移除后若用户使用的 FlashInfer 版本低于 v0.6.8，autotuning 可能失败。但 PR 假设用户已升级。
  3. router logits dtype 变更：移除 fp32 强制转换后，monolithic 内核现在使用 bfloat16 router logits（默认 gate 输出 dtype）。虽然作者通过 e2e 精度测试验证了 DeepSeek R1/V4 等模型，但数值敏感性高的场景可能会有微小差异。
- 影响：
  - 用户影响：使用 DeepSeek V3/V4、MiniMax M2.5 等模型的用户，只需正常升级即可获得更多路由方法支持和性能提升（减少数据类型转换、消除冗余拷贝）。不再需要显式设置 gate 输出为 fp32。
  - 系统影响：MoE 内核选择路径会根据新的路由方法类型确定，确保使用最匹配的内核。FP8/NVFP4 的 apply 流程更简洁，去除了 autotuning 跳转和 workaround 逻辑。
  - 团队影响：路由枚举与 FlashInfer 的版本对齐，未来新增路由方法时有清晰的模式可循；内部类型与内核类型的分离降低了误传风险。
  - 风险标记：Simulated 路由可能传入 FlashInfer，移除 autotuning skip 潜在兼容风险，router logits dtype 变更数值敏感

关联脉络

- PR #38191 [Bugfix] Fix k_norm weight sharding in MiniMaxM2Attention when total_num_kv_heads < tp_size: MiniMax M2 模型相关，本 PR 新增的 MiniMax2 路由方法直接服务于该类模型。
- PR #40860 [Feat] DeepSeek V4 Rebased: DeepSeek V4 使用本 PR 涉及的 TRTLLM MoE 内核和路由方法枚举。