PR #47677 完整报告

vllm-project/vllm

[XPU] Add DSpark speculative decoding support for DeepSeek-V4

合并时间：2026-07-16 08:59

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/47677

执行摘要

- 一句话：为 DeepSeek-V4 添加 XPU DSpark 投机解码支持
- 推荐动作：值得精读，特别是 XPU 平台如何复用已有 MHC 自定义算子而非移植 tilelang 的设计决策。注意缺少测试覆盖，建议合并后尽快补充功能测试和性能基准。

功能与动机

PR 旨在将 DSpark 投机解码扩展到 Intel XPU 平台，使 DeepSeek-V4 在 XPU 上也能利用半自回归投机解码加速推理。此前 DSpark 仅支持 NVIDIA 和 AMD GPU，XPU 用户无法使用。

实现拆解

1. 新增 XPU DSpark draft 模型（xpu/dspark.py）：创建 DSparkDeepseekV4Model 类，使用平台无关的 HCHeadOp/MHCPostOp 自定义算子（来自 vllm-xpu-kernels）替代 NVIDIA 的 tilelang 内核。该模型包含嵌入层、主投影、多个解码层、归一化与 HC 头、Markov 头等模块。
2. 扩展主模型接口（xpu/model.py）：使 DeepseekV4Model 继承 EagleModelMixin，使其能够收集辅助隐藏状态（aux_hidden_states）供 DSpark draft 模型输入；使 DeepseekV4ForCausalLM 继承 SupportsEagle3，标识其支持 Eagle3 投机解码协议。
3. 注册 XPU 平台 DSpark 类（__init__.py）：在 __init__.py 的平台分支中，为 is_xpu() 分支添加 DSparkDeepseekV4ForCausalLM 导入，移除原有的 None 占位符，使模型注册表能够正确加载 XPU 的 DSpark 实现。
4. 配套说明：本次变更未包含测试、配置或部署脚本的配套修改，需后续补充。

关键文件：

- vllm/models/deepseek_v4/xpu/dspark.py（模块 模型定义；类别 source；类型 core-logic；符号 DSparkDeepseekV4Model, init, embed_input_ids, combine_hidden_states）：新增的 XPU DSpark draft 模型核心文件，包含 DSparkDeepseekV4Model 和 DSparkDeepseekV4ForCausalLM 实现，占总变更近 97% 代码量。
- vllm/models/deepseek_v4/xpu/model.py（模块 模型定义；类别 source；类型 data-contract；符号 DeepseekV4Model, DeepseekV4ForCausalLM）：修改主模型以支持 DSpark：集成 EagleModelMixin 并收集 aux_hidden_states；集成 SupportsEagle3 到 ForCausalLM。

- vllm/models/deepseek_v4/__init__.py（模块 模型入口；类别 source；类型 configuration）：修改平台分支导入，为 XPU 注册 DSparkDeepseekV4ForCausalLM，替换原有 None 占位符。

关键符号：DSparkDeepseekV4Model.init, DSparkDeepseekV4Model.forward, DSparkDeepseekV4Model.embed_input_ids, DSparkDeepseekV4Model.combine_hidden_states, DSparkDeepseekV4Model.precompute_and_store_context_kv, DSparkDeepseekV4Model._insert_context_kv, DSparkDeepseekV4ForCausalLM, DeepseekV4Model.forward (aux_hidden_states collection), DeepseekV4ForCausalLM

关键源码片段

vllm/models/deepseek_v4/xpu/dspark.py

新增的 XPU DSpark draft 模型核心文件，包含 DSparkDeepseekV4Model 和 DSparkDeepseekV4ForCausalLM 实现，占总变更近 97% 代码量。

```
class DSparkDeepseekV4Model(nn.Module):
    # XPU DSpark draft 模型，使用 HCHeadOp 和 MHCPostOp 自定义算子
    #  （来自 vllm-xpu-kernels）替代 NVIDIA 的 tilelang 内核
    def __init__(self, *, vllm_config: VllmConfig, prefix: str = '') -> None:
        super().__init__()
        assert vllm_config.speculative_config is not None
        config = vllm_config.speculative_config.draft_model_config.hf_config
        self.config = config
        self.hidden_size = config.hidden_size
        self.hc_mult = config.hc_mult
        self.hc_eps = config.hc_eps
        self.rms_norm_eps = config.rms_norm_eps
        self.num_hidden_layers = config.num_hidden_layers
        self.target_layer_ids = tuple(config.dspark_target_layer_ids)
        self.num_dspark_layers = getattr(config, 'n_mtp_layers', None) or 3

        # 与目标模型共享的嵌入层（通过 speculator 加载工具别名）
        self.embed_tokens = VocabParallelEmbedding(
            config.vocab_size, config.hidden_size,
            prefix=maybe_prefix(prefix, 'embed_tokens'),
        )
        # 主投影：将多个 target layer 的 aux hidden concat 后映射回 hidden size
        self.main_proj = ReplicatedLinear(
            config.hidden_size * len(self.target_layer_ids),
            config.hidden_size, bias=False, return_bias=False,
            quant_config=vllm_config.quant_config,
            prefix=maybe_prefix(prefix, 'main_proj'),
        )
        self.main_norm = RMSNorm(config.hidden_size, eps=config.rms_norm_eps)

        current_vllm_config = get_current_vllm_config()
        self.layers = nn.ModuleList([
```

```
            DeepseekV4DecoderLayer(
                current_vllm_config,
                prefix=maybe_prefix(prefix, f'layers.{self.num_hidden_layers + i}'),
            )
            for i in range(self.num_dspark_layers)
        ])

        # 头：最终归一化 + HC 头，以及 Markov 头
        self.norm = RMSNorm(config.hidden_size, eps=config.rms_norm_eps)
        hc_dim = self.hc_mult * config.hidden_size
        self.hc_head_fn = nn.Parameter(torch.empty(self.hc_mult, hc_dim, dtype=torch.float32),
        requires_grad=False)
        self.hc_head_base = nn.Parameter(torch.empty(self.hc_mult, dtype=torch.float32),
        requires_grad=False)
        self.hc_head_scale = nn.Parameter(torch.empty(1, dtype=torch.float32), requires_grad=
        False)
        draft_vocab_size = getattr(config, 'draft_vocab_size', None) or config.vocab_size
        self.markov_head = DSparkMarkovHead(
            config.vocab_size, draft_vocab_size,
            config.dspark_markov_rank,
            prefix=maybe_prefix(prefix, 'markov_head'),
        )

        # XPU MHC 算子（替代 tilelang）
        self.mhc_post_op = MHCPostOp()
        self.hc_head_op = HCHeadOp()
```

vllm/models/deepseek_v4/xpu/model.py

修改主模型以支持 DSpark：集成 EagleModelMixin 并收集 aux_hidden_states；集成 SupportsEagle3 到 ForCausalLM。

```
# DeepseekV4Model 现在继承 nn.Module, EagleModelMixin
class DeepseekV4Model(nn.Module, EagleModelMixin):
    def forward(self, input_ids, positions, intermediate_tensors, inputs_embeds=None):
        # ... 前置代码 ...
        aux_hidden_states: list[torch.Tensor] = []
        for idx, layer in enumerate(
            islice(self.layers, self.start_layer, self.end_layer),
            start=self.start_layer,
        ):
            hidden_states, residual, post_mix, res_mix = layer(
                hidden_states, positions, input_ids,
                post_mix, res_mix, residual,
            )
            if idx + 1 in self.aux_hidden_state_layers:
                aux_recon = layer.hc_post(hidden_states, residual, post_mix, res_mix)
                aux_hidden_states.append(aux_recon.mean(dim=1))
        if len(aux_hidden_states) > 0:
            return hidden_states, aux_hidden_states
```

```
        return hidden_states

class DeepseekV4ForCausalLM(nn.Module, SupportsPP, SupportsEagle3):
    model_cls = DeepseekV4Model
```

评论区精华

- torch.zeros 改为 torch.empty：reviewer @wuxun-zhang 指出 dspark.py 中未使用的 dummy_q 张量使用 torch.zeros 浪费清零，建议改为 torch.empty，作者 @majian4work 表示接受。
- MHC 自定义算子集成：reviewer @yma11 提问是否应直接使用 XPU 自有的 MHC 自定义算子而非当前方案。作者答复称该算子尚未发布，计划在发布后由算子所有者进行集成。
- 参数 always False 问题：reviewer @jikunshang 询问某个选项是否始终为 False（XPU 上）。作者确认默认 False，并询问未来是否支持，另一 reviewer @xinyu-intel 表示合理。
  - torch.zeros 性能问题 (performance): 修改为 torch.empty。
  - MHC 自定义算子选择 (design): 当前使用 HCHeadOp/MHCPostOp 临时方案，后续将切换至正式 XPU MHC 算子。
  - 参数 always False 问题 (question): 保持默认 False，未来可能支持。

风险与影响

- 风险：
  - 性能风险：dspark.py 中部分张量使用 torch.zeros 而非 torch.empty，会增加不必要的清零开销（已提出修改建议）。
  - 依赖风险：依赖未正式发布的 HCHeadOp/MHCPostOp 自定义算子，若算子接口或行为变动，可能导致功能失效。
  - 测试缺失：本次变更未包含任何自动化测试，无法保证功能正确性和性能预期，需手动验证。
  - 平台限定变更：仅影响 XPU 平台，不会对其他硬件造成回归。
  - 影响：用户影响：XPU 用户现在可以尝试 DeepSeek-V4 的 DSpark 投机解码加速，但功能处于早期支持阶段，可能需要额外配置和手动测试。系统影响：新增约 450 行代码，在主模型路径中增加了 EagleModelMixin 和 SupportsEagle3 的耦合，未来其他平台需同步适配。团队影响：需要对接 XPU kernel 的发布和测试，建议后续补充 CI 测试。
  - 风险标记：缺少测试覆盖，依赖未发布 XPU kernel，平台限定变更，性能隐患（zeros 初始化）

关联脉络

- PR #47463 [Perf] Optimize fused_topk_bias for DSv4, 1.5~2x kernel performance improvement: 同为 DeepSeek-V4 的性能优化，共同提升推理效率。
- PR #47718 [ROCm][Perf] DSv4 two-stage compressor kernel for HCA prefill: 另一平台（ROCm）对 DeepSeek-V4 的优化，体现多平台适配趋势。

- PR #48167 [Bugfix] Fix FlashInfer non-causal draft attention (DFlash/DSpark) on Blackwell: DSpark 相关 bugfix，共享相同的投机解码设计。