

PR #52191 完整报告

vllm-project/vllm

[CPU] Support FP16/BF16 persisted GDN state on AMX

合并时间: 2026-08-31 16:32

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/52191>

执行摘要

- 一句话: CPU AMX 上 GDN state 支持 FP16/BF16 持久化, prefix caching 吞吐 +35%
- 推荐动作: 值得精读, 尤其是三类读者: 1) 负责 CPU/Mamba/GDN 后端的工程师——`_cpu_mamba_backend` 的契约式分发 (`resolve_model_cls + get_mamba_state_dtype_from_config`) 取代硬编码白名单, 是本次最有复用价值的设计决策; 2) 内核开发者——`state_t` 模板参数 + `store_state_vec2` 的「算 FP32、存低精度」模式, 以及线程私有 `state_tmp` 展开缓冲的开销取舍; 3) 关注 hybrid prefix caching 的人——`state dtype` 如何影响统一 block size 与吞吐的因果链。阅读时建议重点关注两处: 一是低精度 state 每个 chunk 写回即舍入的语义是否与目标模型训练 / 推理预期一致, 二是在自家长上下文负载上复测精度, 因为现有质量证据只有 gsm8k 前 100 样本。

功能与动机

PR body 明确指出动机: BF16/FP16 GDN state reduces the aligned block size used by hybrid prefix caching and therefore reduce the prefix-caching glassjaw on CPU AMX backends.。CPU 上 hybrid 模型 (注意力 KV + 循环 state) 必须使用统一 block size, state 元素越宽, 对齐后 block 越大, prefix 命中粒度越粗; 此前 `vllm/platforms/cpu.py` 只要检测到 AVX-512BF16 就把 `mamba_ssm_cache_dtype` 无条件重置为 float32, 用户显式请求的 bf16/fp16 直接被丢弃。作者给出的对照实验中, GDN block 从 auto 的 1152 tokens 降到 640, fp8 KV 下从 2176 降到 1152, 对应吞吐几何平均提升 35.18% / 37.51%。

实现拆解

1. 入口: 平台配置改为条件化决策 (`vllm/platforms/cpu.py`, `CpuPlatform.check_and_update_config`)。原逻辑是「检测到 AVX-512BF16 且请求 dtype 非 float32 → 直接改写为 float32 并告警」; 现在先调用新增的 `_cpu_mamba_backend(vllm_config)` 判断模型是否真的跑在加速 GDN 后端上, 仅当请求 dtype 属于 ("float16", "bfloat16") 且后端名为 `gdn` 时才保留低精度并打 info 日志, 否则维持原有 float32 回退并在告警里带上后端名, 便于定位是哪个模型触发了重置。
2. 后端识别: 白名单改为模型契约。初版 `_cpu_mamba_backend` 用 "qwen3_5" in `model_type` or "qwen3_next" in `model_type` or "olmo_hybrid" in ... 的硬编码白名单 (review 中被指出不可持续), 最终版本改为: 读 `model_config.hf_text_config.layer_type` s 判断是否存在 `linear_attention`, 再通过 `model_config.registry.resolve_model_cls()` 找到模型类并调用 `get_mamba_state_dtype_from_config(vllm_config)`, 把返回元组的第二

个元素 (SSM state dtype) 与用户请求的 cache dtype 比对, 一致才返回 "gdn"。三处 `try/except Exception` 与返回值类型校验保证任何一步失败都退回到 `model_type/architecture` 派生的 fallback 名, 语义上等价于「不认领低精度」。

3. 内核泛型化: 新增 `state_t` 模板参数 (`csrc/cpu/sgl-kernels/fla.cpp`)。
`chunk_gated_delta_rule_fwd_inter_kernel_impl`、
`fused_sigmoid_gating_delta_rule_update_kernel_impl`、
`fused_sigmoid_gating_delta_rule_update_spec_kernel_impl` 的 `state/state_ptr` 参数类型由 `float*` 变为 `state_t*`; 新增模板助手 `store_state_vec2<state_t>` 统一处理「FP32 向量 → 窄精度向量」的写回。chunk 间递推内核额外申请 `state_tmp` ($D \times D$ 个 float, 线程私有) 用于展开低精度 state, chunk 结束再从 FP32 缓冲向量化写回槽位; 解码内核的向量化路径改调 `store_state_vec2`, 标量路径显式 `static_cast<float>` 读、`static_cast<state_t>` 写; 投机解码内核的 src/dst 指针同步泛化。
4. 数值语义: 算 FP32、存低精度。所有 GEMM (`brgemm/blas_gemm`)、`exp(g_last)` 缩放、累加都在 FP32 缓冲上进行, 只有 state 槽位的读写走 `state_t`, 因此每个 chunk 写回即发生一次舍入——这与 GPU 侧状态 dtype 语义保持一致, 也是测试参考实现需要模拟的量化点。
5. 测试与验证配套。新增 `tests/platforms/test_cpu.py` (+152 行) 用 `_FakeModel / _FakeRegistry / _cpu_config` 构造最小 `VllmConfig`, 参数化 8 个用例覆盖 qwen3_5 (bf16/fp16/auto 均保留) 与 nemotron_h (mamba 层, 一律回退 float32), 并 monkeypatch `torch.cpu._is_avx512_bf16_supported`;
`tests/kernels/mamba/cpu/test_cpu_gdn_ops.py` 新增 `DECODE_DTYPE_CASES / PREFILL_DTYPE_CASES` (fp32 全矩阵 + bf16/fp16 代表用例), 参考实现 `ref_gated_delta_rule` 扩展 `state_dtype` 与 `state_write_interval` 以模拟按 chunk 舍入, `two_call_split` 用例也参数化到三种 state dtype, 断言统一 `.float()` 后比较 (atol 1e-2)。作者另在 issue 评论中补充 gsm8k 前 100 样本质量验证: auto(fp32)/bf16/fp16 三种设置均得 0.9。

关键文件:

- `csrc/cpu/sgl-kernels/fla.cpp` (模块 状态内核; 类别 source; 类型 core-logic; 符号 `store_state_vec2`, `chunk_gated_delta_rule_fwd_inter_kernel_impl`, `fused_sigmoid_gating_delta_rule_update_kernel_impl`, `fused_sigmoid_gating_delta_rule_update_spec_kernel_impl`): 核心内核改动: 三个 GDN 内核按 `state_t` 泛型化, 新增 `store_state_vec2` 向量化助手, chunk 间递推内核引入 `state_tmp` FP32 展开缓冲并按 chunk 写回低精度 state, 是本次吞吐 / 精度权衡的落点。
- `vllm/platforms/cpu.py` (模块 平台配置; 类别 source; 类型 core-logic; 符号 `_cpu_mamba_backend`, `CpuPlatform.check_and_update_config`): 策略入口: 新增 `_cpu_mamba_backend` 以模型 state dtype 契约判定 GDN 后端, 并把原来无条件重置 `mamba_ssm_cache_dtype=float32` 改为条件保留 bf16/fp16; 是白名单争议收敛后的最终形态。
- `tests/platforms/test_cpu.py` (模块 平台测试; 类别 test; 类型 test-coverage; 符号 `_FakeModel`, `get_mamba_state_dtype_from_config`, `_FakeRegistry`, `resolve_model_cls`): 新增平台策略单测: 用 `_FakeModel/_FakeRegistry` 隔离 registry 依赖, 参数化 8 个用例

验证「契约匹配保留低精度 / 非 GDN 回退 float32」这一新策略。

- tests/kernels/mamba/cpu/test_cpu_gdn_ops.py (模块 内核测试; 类别 test; 类型 test-coverage; 符号 ref_gated_delta_rule, test_fused_sigmoid_gating_delta_rule_update_cpu, test_chunk_gated_delta_rule_cpu, test_chunk_gated_delta_rule_cpu_two_call_split) : 内核数值验证: 新增 DECODE/PREFILL dtype 参数化用例, 参考实现扩展 state_dtype 与 state_write_interval 以模拟按 chunk 舍入, 覆盖 decode/prefill/ 两次调用状态传递。

关键符号: _cpu_mamba_backend, CpuPlatform.check_and_update_config, store_state_vec2, chunk_gated_delta_rule_fwd_inter_kernel_impl, fused_sigmoid_gating_delta_rule_update_kernel_impl, fused_sigmoid_gating_delta_rule_update_spec_kernel_impl, get_mamba_state_dtype_from_config, ref_gated_delta_rule, test_cpu_accelerated_gdn_dtype_policy

关键源码片段

vllm/platforms/cpu.py

策略入口: 新增 _cpu_mamba_backend 以模型 state dtype 契约判定 GDN 后端, 并把原来无条件重置 mamba_ssm_cache_dtype=float32 改为条件保留 bf16/fp16; 是白名单争议收敛后的最终形态。

```
def _cpu_mamba_backend(vllm_config: VllmConfig) -> str:
    """返回由模型配置决定的 CPU 状态后端名。"""
    model_config = vllm_config.model_config
    if model_config is None:
        return "none"

    hf_config = getattr(model_config, "hf_text_config", None)
    model_type = str(getattr(hf_config, "model_type", "")).lower()
    architecture = str(getattr(model_config, "architecture", "")).lower()
    layer_types = getattr(hf_config, "layer_types", None)
    # 是否含线性注意力层, 是切到 GDN 路径的前提
    has_linear_attention = isinstance(layer_types, (list, tuple)) and (
        "linear_attention" in layer_types
    )

    try:
        has_inner_state = bool(model_config.has_inner_state)
    except (AttributeError, RuntimeError):
        has_inner_state = False

    # 既无 inner state 也无线性注意力层: 该模型不使用任何 SSM 后端
    if not (has_inner_state or has_linear_attention):
        return "none"

    fallback_backend = model_type or architecture or "unknown"
```

```

if not has_linear_attention:
    return fallback_backend

# 早期版本在此处硬编码 qwen3_5 / qwen3_next / olmo_hybrid 白名单,
# review 后改为向模型类索取 persisted state dtype 契约
try:
    model_cls, _ = model_config.registry.resolve_model_cls(
        model_config.architecture,
        model_config=model_config,
    )
except Exception:
    return fallback_backend

try:
    state_dtypes = model_cls.get_mamba_state_dtype_from_config(vllm_config)
except Exception:
    return fallback_backend

if not isinstance(state_dtypes, tuple) or len(state_dtypes) != 2:
    return fallback_backend

if vllm_config.cache_config.mamba_ssm_cache_dtype == "float16":
    cache_dtype = torch.float16
elif vllm_config.cache_config.mamba_ssm_cache_dtype == "bfloat16":
    cache_dtype = torch.bfloat16
else:
    return fallback_backend

# 用户请求的低精度与模型声明的 SSM state dtype (元组第二项) 一致时,
# 才认领 gdn 后端, 从而放行低精度持久化
if state_dtypes[1] == cache_dtype:
    return "gdn"

return fallback_backend

```

```

class CpuPlatform(Platform):
    @classmethod
    def check_and_update_config(cls, vllm_config: VllmConfig) -> None:
        # ... block_size / MLA 相关逻辑略 ...

        # Accelerated GDN uses AMX tiles or AVX-512BF16 VDPBF16PS.
        if (
            torch.cpu._is_avx512_bf16_supported()
            and cache_config.mamba_ssm_cache_dtype != "float32"
        ):
            mamba_backend = _cpu_mamba_backend(vllm_config)
            if (
                cache_config.mamba_ssm_cache_dtype in ("float16", "bfloat16")

```

```

        and mamba_backend == "gdn"
    ):
        # 契约匹配: 保留用户请求的 bf16/fp16, 缩小 hybrid prefix caching
        # 的统一 block size
        logger.info(
            "Using %s SSM state storage for the CPU accelerated GDN backend.",
            cache_config.mamba_ssm_cache_dtype,
        )
    else:
        # 非 GDN 模型或契约不匹配: 沿用原来的 float32 回退,
        # 告警里带上后端名, 便于定位是哪类模型触发重置
        cache_config.mamba_ssm_cache_dtype = "float32"
        logger.warning(
            "Reset SSM cache type to float32 for accelerated GDN mamba "
            "backend '%s'.",
            mamba_backend,
        )

```

tests/platforms/test_cpu.py

新增平台策略单测: 用 `_FakeModel/_FakeRegistry` 隔离 registry 依赖, 参数化 8 个用例验证「契约匹配保留低精度 / 非 GDN 回退 float32」这一新策略。

```

class _FakeModel:
    """用假模型类声明 persisted state dtype 契约, 隔离真实 registry 依赖。"""

    @classmethod
    def get_mamba_state_dtype_from_config(
        cls, vllm_config: VllmConfig
    ) -> tuple[torch.dtype, torch.dtype]:
        # 约定: 返回值第二位才是 SSM state 的持久化 dtype
        state_dtype = {
            "float16": torch.float16,
            "bfloat16": torch.bfloat16,
        }.get(vllm_config.cache_config.mamba_ssm_cache_dtype, torch.float32)
        return torch.float32, state_dtype


class _FakeRegistry:
    """替换 model_config.registry, 避免测试真的去导入模型实现。"""

    @staticmethod
    def resolve_model_cls(architecture, model_config):
        return _FakeModel, architecture


@pytest.mark.parametrize(
    "model_type,requested_dtype,resolved_dtype,layer_types,expected_dtype",
    [
        # linear_attention + 契约匹配: 保留用户显式请求的低精度

```

```

    pytest.param(
        "qwen3_5", "bfloat16", "bfloat16", ("linear_attention",), "bfloat16",
        id="gdn-explicit-bf16",
    ),
    pytest.param(
        "qwen3_5", "float16", "float16", ("linear_attention",), "float16",
        id="gdn-explicit-fp16",
    ),
    # 仅有 mamba 层（非 GDN）：无论显式还是 auto，一律回退 float32
    pytest.param(
        "nemotron_h", "bfloat16", "bfloat16", ("mamba",), "float32",
        id="unsupported-explicit-bf16",
    ),
    pytest.param(
        "nemotron_h", "float16", "float16", ("mamba",), "float32",
        id="unsupported-explicit-fp16",
    ),
],
)
def test_cpu_accelerated_gdn_dtype_policy(
    monkeypatch: pytest.MonkeyPatch,
    model_type: str,
    requested_dtype: str,
    resolved_dtype: str,
    layer_types: tuple[str, ...],
    expected_dtype: str,
) -> None:
    # 强制走 AVX-512BF16 分支，使低精度策略生效
    monkeypatch.setattr("torch.cpu._is_avx512_bf16_supported", lambda: True)
    cache_config = CacheConfig(mamba_ssm_cache_dtype=requested_dtype)
    config = _cpu_config(
        cache_config,
        model_type=model_type,
        resolved_dtype=resolved_dtype,
        layer_types=layer_types,
    )

    CpuPlatform.check_and_update_config(config)
    assert cache_config.mamba_ssm_cache_dtype == expected_dtype

```

评论区精华

三条 reviewer 意见（bigPYJ1151）都直接改变了实现：

1. 设计层面最重的一条针对 `_cpu_mamba_backend` 的白名单实现——Maintain a white list is not a long-term solution. Maybe we should consider a better way to check the GDN usage, 促成了 commit 24cab2d7 「cpu: detect GDN from model state dtype contract」，即现在的 `resolve_model_cls + get_mamba_state_dtype_from_config` 契约判定，这也让新增的模型只要实现该接口即可自动获得低精度收益。

2. 两条性能意见要求向量化 D*D 的 state 转换循环 (Maybe this loop can be vectorized. D*D is not a small number. / Same here) , 最后一个 commit e41b09f8 「cpu: vectorize GDN state conversions」引入 store_state_vec2 与 load_float_vec2, 仅保留标量尾巴。
3. 精度层面: Did you test the accuracy when use BF16 state?, 作者以 gsm8k 前 100 样本 (auto/bf16/fp16 全部 0.9) 回应, 随后 review 转为 APPROVED。另外 claude[bot] 因 PR 来自 fork 未做自动审查, 最终由 bigPYJ1151 人工批准合并。
- GDN 后端判定不应依赖硬编码白名单 (design): 改为契约式判定: model_config.registry.resolve_model_cls 找到模型类后调用 get_mamba_state_dtype_from_config, 用返回的 SSM state dtype 与用户请求的 cache dtype 比对, 一致才返回 "gdn"; 失败一律回退到 model_type/architecture 派生的 fallback 名。对应 commit 24cab2d7, 并补充 8 个平台单测固定该行为。
- 低精度 state 转换循环需要向量化 (performance): 最后一个 commit e41b09f8 「cpu: vectorize GDN state conversions」引入 store_state_vec2<state_t> 与 load_float_vec2, 读入与写回都改为向量化循环 + 标量尾巴, 并统一了 decode 内核的存储路径。
- BF16 state 的模型级精度是否可接受 (testing): 作者在 PR 评论中给出 gsm8k 前 100 样本结果: auto (fp32), bf16 and fp16 all get 0.9, 随后 review 给出 APPROVED 并合并。样本规模有限, 缺少长上下文与多轮 prefix 复用场景的质量回归。

风险与影响

- 风险:
 1. CPU 默认行为变更: 此前 AVX-512BF16 机器上 mamba_ssm_cache_dtype 必被重置为 float32, 现在 GDN 模型会保留 bf16/fp16, 属于数值路径的默认变更; 影响范围限于 CPU AMX + 声明了 state dtype 契约的 GDN 模型, 非 GDN 的 mamba 模型 (如 nemotron_h) 仍回退 float32, 回归面可控。
 2. 精度漂移: state 首次进入低精度存储, 长序列、多次 prefix 复用或多次 chunk 写回会累积舍入误差; 现有证据是 gsm8k 前 100 样本 (采样规模偏小) 与内核单测 atol 1e-2 的对比, 缺少长上下文、高并发、多轮 prefix 复用下的质量回归。
 3. 异常静默回退: _cpu_mamba_backend 对 resolve_model_cls、get_mamba_state_dtype_from_config 均使用裸 except Exception 且只在回退分支写日志, 若模型契约实现本身有 bug, 表现是「悄悄变慢」而非报错, 排查成本较高; 同时它假设返回元组第二项即 SSM state dtype, 顺序约定变化会静默失效。
 4. 初始化期副作用: check_and_update_config 在引擎初始化阶段调用 resolve_model_cls, 会触发模型模块解析 / 导入, 属于平台配置层新增的隐式依赖, 若模型导入较慢会拖长启动; 异常虽被吞掉但副作用仍发生。
 5. 内核额外开销: 低精度路径每 chunk 多一次 D*D 展开与一次写回 (state_tmp 线程私有 FP32 缓冲), brgemm_supported()==false 的标量路径下 pack_vnni2 会就地缩放 state_tmp, 正确性依赖「先快照、后缩放」的顺序, 注释已显式说明, 但该组合路径 (非 AVX-512 平台 + 低精度 state) 缺少针对性测试。- 影响: 用户: 在 Intel AMX (SPR/EMR/GNR) 上部署 Qwen3.5/Qwen3.6、Qwen3-Next、OLMo-Hybrid 等 GDN 混合模型的用户, 开启 prefix caching 后可获得约 35%-38% 的输出吞吐提升, 同时循环

state 显存 / 内存占用减半, hybrid prefix caching 的有效 block size 从 1152/2176 降到 640/1152 tokens, 前缀命中粒度更细。系统: CPU 平台的 SSM state dtype 决策从「硬件能力驱动」变为「模型契约驱动」, _cpu_mamba_backend 成为 CPU 后续 Mamba/GDN 后端选择的事实上的扩展点; 内核侧 state_t 泛型化为未来其他低精度 state (如 FP8) 留出了模板位置。团队: 新增模型若想吃到该优化, 需实现 get_mamba_state_dtype_from_config 并声明 layer_types 含 linear_attention, 这是一条需要写进模型接入规范的隐性契约; 测试侧新增了 tests/platforms/test_cpu.py 这一平台策略单测入口。无 GPU 侧影响, 不涉及 API/schema 变更。- 风险标记: CPU 默认为行为变更, 低精度状态数值漂移, 异常静默回退, 缺少长上下文精度验证, 非 AVX-512 组合路径覆盖不足

关联脉络

- PR #54934 [CI][CPU] Add speculative-decoding coverage to CPU CI: 本 PR 同时把投机解码路径的内核 fused_sigmoid_gating_delta_rule_update_spec_kernel_impl 泛化到 state_t, 而该历史 PR 正是为 CPU 平台补上投机解码 E2E/Smoke 覆盖, 两者共同构成 CPU GDN 投机解码链路的实现与验证。
- PR #56773 [Bugfix][CPU] Fix DeepSeek-R1 (FP8 MLA + MoE) correctness on CPU backend: 同属 CPU 后端的精度 / 内核正确性修复线 (scaled_mm/cpu.py、experts/cpu_moe.py), 与本 PR 在 FP8/ 低精度路径上对 CPU 内核语义的谨慎处理方向一致, 可对照其验证与回退策略。
- PR #50894 [Bugfix] Scale KV page size for hidden states extraction with TP: 同样围绕 hybrid 模型 (注意力 KV + 循环 / 隐藏状态) 的页大小与对齐逻辑修改 vllm/v1/core/kv_cache_utils.py, 与本 PR 动机中「GDN state 宽度决定 hybrid prefix caching 对齐 block size」是同一类问题的不同侧面。
- PR #56791 [Bugfix] Trim stale consequence claims from unannotated-eagle warning: 改动同样落在 vllm/v1/core/kv_cache_utils.py 的 hybrid/mamba 状态告警文案上, 说明平台层对 hybrid state dtype 的提示逻辑近期是活跃修改区, 本 PR 的 float32 回退告警措辞变更属于同一演进方向。