PR #24268 完整报告

sgl-project/sglang

[Kernel] Deprecate DeepGemm in sgl kernel and apply custom wheel sgl-deep-gemm

合并时间：2026-05-07 09:59

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/24268

执行摘要

- 一句话：将 DeepGemm 从 sglang-kernel 剥离为独立 wheel
- 推荐动作：核心架构调整，值得仔细审查。重点关注 nsa_backend.py 中的 _to_2d_context_lens 适配逻辑和 fp8_utils.py 中的 stride 修复，它们直接关系到 DeepSeek 等模型在 FP8 下的正确性。建议在合并后运行完整的 DeepSeek 集成测试以验证无精度退化。

功能与动机

参考 DeepGEMM PR#26 和 sgl-deep-gemm，以及关联 issue #20745。PR body 说明："We will build a single wheel for deepgemm in sglang, rather than compiling it with sglang-kernel"。目的是将 DeepGemm 解耦为独立包，避免随 sglang-kernel 一起编译，便于独立演进和发布。

实现拆解

1. 移除 sglang-kernel 中的 DeepGemm 构建（前置 PR #24457），在 CMakeLists.txt 中删除 DeepGemm 编译目标，并提升 sglang-kernel 版本至 0.4.2.post1（engine.py 中版本检查）。
2. 适配新 wheel 的 API 变化：deep_gemm.get_paged_mqa_logits_metadata 和 fp8_paged_mqa_logits 要求 context_lens 形状为 [batch_size, next_n]。在 nsa_backend.py 中新增 _to_2d_context_lens 辅助函数，将 1D seqlens 转换为 2D，并在 init_forward_metadata 及其 cuda graph 捕获 / 重放方法中应用。同理修改 nsa_indexer.py 中的 _get_topk_paged。
3. 保护非 CUDA/MUSA 平台：在 configurer.py 的 _compute_enable_deep_gemm 开头添加 if not (_is_cuda or _is_musa): return False，避免在非 NVIDIA/AMD GPU 上导入 deep_gemm 时因 _find_cuda_home 断言崩溃。
4. 兼容旧版 DeepGemm 接口：在 compile_utils.py 的 _compile_deep_gemm_one_type_all 中，通过 hasattr 检查 get_compile_mode/set_compile_mode 是否存在，不存在时跳过设置，防止 AttributeError。
5. 修复 DLPack stride 退化：在 fp8_utils.py 的 transform_scale_ue8m0 中，当使用 sgl-deep-gemm 且 sf.shape[-1] == 1 时，恢复 TMA-aligned stride，以满足 DeepGemm 内部断言。

6. 更新 CI 和测试配置：修改 scripts/ci/cuda/ci_install_dependency.sh 安装 sgl-deep-gemm，调整多个分布式测试文件（test_dsa_models_mtp.py、test_deepseek_v32_cp_single_node.py、test_deepseek_v32_fp4_mtp_4gpu.py）中的版本号或标记，以及 warmup_deep_gemm.py 脚本的依赖处理。

关键文件：

- python/sglang/srt/layers/attention/nsa_backend.py（模块 注意力层；类别 source；类型 core-logic；符号 _to_2d_context_lens）：新增 _to_2d_context_lens 将 context_lens 调整为 2D，以匹配 DeepGEMM 0426 API 要求，在 metadata 初始化和 cuda graph 相关路径中使用。
- python/sglang/srt/layers/quantization/fp8_utils.py（模块 量化工具；类别 source；类型 dependency-wiring）：修复 DLPack 转换后 stride 退化，确保 TMA-aligned stride 满足 DeepGemm 内部断言。
- python/sglang/srt/layers/deep_gemm_wrapper/compile_utils.py（模块 编译工具；类别 source；类型 core-logic）：通过 hasattr 保护性检查 get/set_compile_mode API，兼容没有这些接口的 DeepGEMM 版本。
- python/sglang/srt/layers/attention/nsa/nsa_indexer.py（模块 索引器；类别 source；类型 core-logic）：类似 nsa_backend.py，确保 context_lens 形状为 2D 以匹配 q_fp8 在 unsqueeze 后的布局。
- python/sglang/srt/entrypoints/engine.py（模块 引擎入口；类别 source；类型 configuration）：更新 sglang-kernel 版本检查到 0.4.2.post1，确保用户安装正确的 kernel 版本。
- python/sglang/srt/layers/deep_gemm_wrapper/configurer.py（模块 配置器；类别 source；类型 core-logic）：添加非 CUDA/MUSA 平台保护，避免导入 deep_gemm 时因 CUDA 缺失而崩溃。
- scripts/ci/cuda/ci_install_dependency.sh（模块 CI 脚本；类别 infra；类型 infrastructure）：添加 pip install sgl-deep-gemm 到 CI 依赖安装脚本中。
- scripts/ci/cuda/warmup_deep_gemm.py（模块 CI 脚本；类别 infra；类型 infrastructure）：配合新 wheel 调整 warmup 脚本，添加 API 存在性检查。
- test/registered/8-gpu-models/test_dsa_models_mtp.py（模块 测试；类别 test；类型 test-coverage）：更新测试中的版本依赖标记。
- test/registered/cp/test_deepseek_v32_cp_single_node.py（模块 测试；类别 test；类型 test-coverage）：类似更新测试配置。
- test/registered/quant/test_deepseek_v32_fp4_mtp_4gpu.py（模块 测试；类别 test；类型 test-coverage）：类似更新测试配置。
- test/registered/rl/test_weight_checker_e2e.py（模块 测试；类别 test；类型 test-coverage）：微小调整测试分组。

关键符号：_to_2d_context_lens, transform_scale_ue8m0, _compute_enable_deep_gemm, init_forward_metadata, init_forward_metadata_capture_cuda_graph, init_forward_metadata_replay_cuda_graph, _get_topk_paged

关键源码片段

python/sglang/srt/layers/attention/nsa_backend.py

新增 _to_2d_context_lens 将 context_lens 调整为 2D，以匹配 DeepGEMM 0426 API 要求，在 metadata 初始化和 cuda graph 相关路径中使用。

```
# 新增辅助函数，将一维 seq lens 转换为 [batch_size, next_n] 二维张量
def _to_2d_context_lens(seqlens_32: torch.Tensor, batch_size: int) -> torch.Tensor:
    if seqlens_32.dim() == 2:
        return seqlens_32
    n = seqlens_32.numel()
    assert (
        n % batch_size == 0
    ), f"seqlens_32 size {n} is not a multiple of batch_size {batch_size}"
    next_n = n // batch_size
    return seqlens_32.view(batch_size, next_n)

# 在 init_forward_metadata 中使用（省略其他类似处）
seqlens_32_2d = _to_2d_context_lens(seqlens_32, forward_batch.batch_size)
paged_mqa_schedule_metadata = deep_gemm.get_paged_mqa_logits_metadata(
    seqlens_32_2d, 64, deep_gemm.get_num_sms()
)
```

python/sglang/srt/layers/quantization/fp8_utils.py

修复 DLPack 转换后 stride 退化，确保 TMA-aligned stride 满足 DeepGemm 内部断言。

```
# 在 transform_scale_ue8m0 尾部追加的 stride 修复（在 sf 转换之后）
# In sgl-deep-gemm, the C++ deepgemm path returns through DLPack which collapses the
stride
# of size-1 trailing dims to 1 (happens when packed_sf_k == 1, i.e.
# K <= block_k * 4). Restore the TMA-aligned stride so the deepgemm
# assertion sf.stride(-1) == get_tma_aligned_size(mn, element_size) holds.
if not use_torch_impl and sf.shape[-1] == 1:
    from deep_gemm.utils import get_tma_aligned_size

    aligned_mn = get_tma_aligned_size(sf.shape[-2], sf.element_size())
    if sf.stride(-1) != aligned_mn:
        new_stride = list(sf.stride())
        new_stride[-1] = aligned_mn
        sf = sf.as_strided(sf.shape, tuple(new_stride))
return sf
```

python/sglang/srt/layers/attention/nsa/nsa_indexer.py

类似 nsa_backend.py，确保 context_lens 形状为 2D 以匹配 q_fp8 在 unsqueeze 后的布局。

```
# 在 _get_topk_paged 中，调用 DeepGEMM 前将 seqlens_32 转为 2D
# DeepGEMM release-0426 requires context_lens of shape [batch_size, next_n]
# to match q.shape = [batch_size, next_n, heads, head_dim]. The indexer uses
# next_n=1 with batch_size=N_total via q_fp8.unsqueeze(1) below, so mirror
```

```
# that layout here.
if seqlens_32.dim() == 2:
    seqlens_32_2d = seqlens_32
else:
    seqlens_32_2d = seqlens_32.unsqueeze(-1)

if _is_cuda:
    if schedule_metadata is None:
        schedule_metadata = deep_gemm.get_paged_mqa_logits_metadata(
            seqlens_32_2d, blocksize, self.sm_count
        )
# ... 后续 fp8_paged_mqa_logits 也使用 seqlens_32_2d
```

评论区精华

仅有 gemini-code-assist[bot] 的自动审查评论，未提出具体问题。无人工 review 讨论。

- 自动代码审查 (other): 无人工讨论，PR 由作者直接合并。

风险与影响

- 风险：
  - 兼容性风险：用户若未安装 sgl-deep-gemm，则 DeepGemm 相关功能（如 FP8 量化推理）将回退或报错。现有代码中已有 ImportError 捕获，但可能覆盖不全，例如在 transform_scale_ue8m0 中直接 import deep_gemm 未捕获。
  - API 版本依赖：新 wheel 要求 context_lens 为 2D，适配代码假设 deep_gemm.get_paged_mqa_logits_metadata 接受该形状，若未来 API 再次变化需同步维护。
  - 非 CUDA 平台保护：除 configurer.py 和 compile_utils.py 外，其他文件（如 nsa_backend.py）仍有条件导入 deep_gemm，但已包含 is_cuda() 判断，风险较低。
  - 测试覆盖不足：仅更新了少量测试文件的版本号，未新增针对新 wheel 的集成测试，可能遗漏回归。
- 影响：
  - 用户：需要额外运行 pip install sgl-deep-gemm 或通过依赖自动安装；升级 sglang 后若缺少该包将导致 DeepGemm 功能不可用。
  - 系统：sglang-kernel 编译时间减少（不再编译 DeepGemm），安装包体积减小。
  - 团队：需要同时维护 sglang-kernel 和 sgl-deep-gemm 两个包的发布节奏，并在 CI 中确保两者兼容。
  - 风险标记：外部包依赖 sgl-deep-gemm，非 CUDA 平台可能崩溃，API 形状要求严格

关联脉络

- PR #24457 Remove deepgemm in CMakelist.txt of sglang-kernel && Bump kernel version: 前置步骤：从 sglang-kernel 构建中移除 DeepGemm，此 PR 在此基础上集成独立 wheel。

- PR #20745 Issue: 将 DeepGemm 作为独立 wheel 发布：关联 issue，驱动此 PR 的原始动机。