PR #34524 完整报告

sgl-project/sglang

Fix DFlash sliding attention causality defaults

合并时间：2026-08-12 14:05

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/34524

执行摘要

- 一句话：修复 DFlash 滑动注意力因果默认值回归
- 推荐动作：值得精读。这是一个小而完整的回归修复，展示了『显式声明优先 + 历史默认值分层回退』的设计模式，并附带清晰的 CI 验证数据。对维护 DFlash 推测解码、draft 模型接入或类似『配置默认值契约』的同学有参考价值。重点看 _get_dflash_attention_type 的默认值传递与 MuseGlimmerAssistantConfig.is_causal 的配套声明。

功能与动机

PR #34262 在引入 Muse Glimmer 支持时，将 DFlash 注意力类型统一为『checkpoint 未声明 is_causal 则默认双向』，但历史 DFlash checkpoint 并不包含该字段。z-lab/gemma-4-31B-it-DFlash 有 4 个 sliding-attention 层且未声明 is_causal，导致平均 speculative accept length 从约 5.62 回退至 5.27，未通过 test_gemma4_dflash_31b_extra.py 的 5.4 阈值。作者在 PR body 中明确指出『This change restores compatibility for existing DFlash checkpoints while retaining Muse Glimmer's intended bidirectional draft attention.』

实现拆解

1. 定位回归根因：python/sglang/srt/models/dflash.py 中 _get_dflash_attention_type 原先在 checkpoint 未声明 is_causal 时一律返回 ENCODER_ONLY（双向），导致 z-lab/gemma-4-31B-it-DFlash 这类历史 checkpoint 的 4 个 sliding-attention 层从 causal 变为双向，spec accept length 从 5.62 掉到 5.27。
2. 引入默认值参数：将 _get_dflash_attention_type 改为接收 default: AttentionType 关键字参数，先通过 config.get_text_config() 解析 text config；仅当 is_causal 取值为 None（未声明）时才返回调用方传入的默认值，否则按 True/False 分别映射为 DECODER/ENCODER_ONLY，保证显式声明始终优先。
3. 按层类型恢复历史默认：在 _get_dflash_layer_attention_params 中，sliding_attention 层传入默认 DECODER（因果），full_attention 层传入默认 ENCODER_ONLY（双向），恢复 PR #34262 之前的 legacy 行为。
4. 显式声明 Muse Glimmer 配置：在 python/sglang/srt/configs/muse_glimmer.py 的 MuseGlimmerAssistantConfig 中增加类属性 is_causal = False，使 Muse Glimmer draft 模型显式声明双向注意力，不受 sliding 层新默认值影响。

5. 验证与配套：未新增单元测试文件；作者在 2x NVIDIA H100（TP=2）本地运行 test/registered/spec/test_gemma4_dflash_31b_extra.py，avg_spec_accept_length 恢复至 5.6333，同时 CI rerun 中 Gemma-4 与 Muse Glimmer 两个 acceptance 测试均通过。

关键文件：

- python/sglang/srt/models/dflash.py（模块 推测解码；类别 source；类型 data-contract；符号 _get_dflash_attention_type, _get_dflash_layer_attention_params）：回归修复的核心文件：_get_dflash_attention_type 新增默认值参数并改为显式声明优先，_get_dflash_layer_attention_params 按层类型回传历史默认值，是本次变更的主逻辑。
- python/sglang/srt/configs/muse_glimmer.py（模块 模型配置；类别 source；类型 core-logic；符号 MuseGlimmerAssistantConfig）：为 MuseGlimmerAssistantConfig 显式声明 is_causal = False，确保 Muse Glimmer 的双向 draft 注意力不随 sliding 层默认值恢复而变化。

关键符号：_get_dflash_attention_type, _get_dflash_layer_attention_params, MuseGlimmerAssistantConfig

关键源码片段

python/sglang/srt/models/dflash.py

回归修复的核心文件：_get_dflash_attention_type 新增默认值参数并改为显式声明优先，_get_dflash_layer_attention_params 按层类型回传历史默认值，是本次变更的主逻辑。

```
def _get_dflash_attention_type(config, *, default: AttentionType) -> AttentionType:
    """优先尊重 checkpoint 显式声明的 is_causal，否则回退到 legacy 层默认值。"""
    # 使用 config API 统一解析 text_config（兼容嵌套的 text_config 结构）
    text_config = config.get_text_config()
    is_causal = getattr(text_config, "is_causal", None)
    # is_causal 未声明时使用调用方传入的历史默认值，避免 PR #34262 引入的行为漂移
    if is_causal is None:
        return default
    # 显式声明优先：True 映射 DECODER（因果），False 映射 ENCODER_ONLY（双向）
    return AttentionType.DECODER if is_causal else AttentionType.ENCODER_ONLY


def _get_dflash_layer_attention_params(config, layer_id: int) -> Tuple[int, AttentionType]:
    layer_types = get_dflash_layer_types(config)
    if layer_types is None:
        return -1, AttentionType.ENCODER_ONLY
    if layer_id >= len(layer_types):
        raise ValueError(
            "DFLASH config.layer_types must contain one entry per draft layer. "
            f"Got {len(layer_types)} entries, layer_id={layer_id}."
        )

    layer_type = layer_types[layer_id]
```

```
    if layer_type == "full_attention":
        # 历史默认：full-attention 层为双向注意力
        return -1, _get_dflash_attention_type(
            config, default=AttentionType.ENCODER_ONLY
        )
    if layer_type == "sliding_attention":
        # 历史默认：sliding-attention 层为因果注意力；
        # 窗口掩码与因果性正交（mask 条件为 p1 - p0 >= window）
        sliding_window_size = get_dflash_attention_sliding_window_size(config)
        assert sliding_window_size is not None
        return sliding_window_size, _get_dflash_attention_type(
            config, default=AttentionType.DECODER
        )
    raise ValueError(
        "Unsupported DFLASH draft layer type. "
        f"layer_types[{layer_id}]={layer_type!r}."
    )
```

python/sglang/srt/configs/muse_glimmer.py

为 MuseGlimmerAssistantConfig 显式声明 is_causal = False，确保 Muse Glimmer 的双向 draft 注意力不随 sliding 层默认值恢复而变化。

```
class MuseGlimmerAssistantConfig(PretrainedConfig):

    model_type = "muse_glimmer_assistant"
    # 显式声明 DFlash draft 为双向注意力，覆盖 sliding 层的 legacy causal 默认值，
    # 以保留 Muse Glimmer 预期的双向 draft 行为
    is_causal = False
    # DFlash draft 没有独立输出头；draft_worker_common 复用目标模型的 head
    vocab_size = None
```

评论区精华

本 PR 没有实质性的 review 评论，维护者 hnyls2002 直接 APPROVED。作者在 issue 评论中请求 rerun test_gemma4_dflash_31b_extra.py 与 test_muse_glimmer_dflash_assistant_gsm8k.py，github-actions 在 2-gpu-h100 与 1-gpu-h100 两个环境分别执行并全部通过。设计上没有出现争议，核心权衡（显式声明优先 + legacy 层默认值回退）在 PR body 中有清晰说明。

- CI 回归验证结果 (testing): 两个测试均通过，Gemma-4 DFlash 恢复 5.63 的接受长度，Muse Glimmer 显式双向行为得到端到端确认。

风险与影响

- 风险：
  1. 兼容性：config.get_text_config() 是 PretrainedConfig 提供的 API；若未来传入不继承 PretrainedConfig 的自定义配置对象，可能因缺少该方法而抛 AttributeError。当前 DFlash 均走标准配置路径，风险较低。

2. 行为契约：默认值依赖 is_causal is None 判空；若某个 checkpoint 显式声明 is_causal=False 而历史实现按因果处理，行为会随显式声明改变（这是预期结果，但混合声明场景需留意）。
3. 测试覆盖：没有为默认值逻辑新增独立单元测试，回归保护仅依赖两个 acceptance 测试；若运行环境缺少对应 checkpoint，常规 CI 无法捕获同类回归。
4. 影响面：仅影响 DFlash 推测解码 draft 模型的注意力类型选择，不触碰推理热路径、调度或 kernel，无性能与安全影响。 - 影响：用户侧：修复 gemma-4-31B-it-DFlash 用户的推测解码质量回归，接受长度恢复到历史 5.61-5.62 区间；Muse Glimmer 用户双向 draft 行为保持不变。系统侧：改动集中在模型配置与注意力类型解析，不在每 token 前向热路径内，无额外开销。团队侧：确立了 DFlash 注意力类型『显式声明优先、未声明时按层类型回退』的默认值契约，后续接入新 DFlash 变体时需要显式声明 is_causal 或依赖该契约。 - 风险标记：核心路径变更，无新增单测，默认值契约依赖 is_causal 判空

关联脉络

- PR #34262 [Feature] Add Muse Glimmer model support: 该 PR 引入 _get_dflash_attention_type 的双向默认行为，导致 gemma-4 DFlash 未声明 is_causal 时 sliding 层从 causal 变为双向，本 PR 是对其回归的修复。