

PR #34556 完整报告

sgl-project/sglang

Inference Support Mamba 2 and 1

合并时间: 2026-09-12 20:51

原文链接: <http://prhub.com.cn/sgl-project/sglang/pull/34556>

执行摘要

- 一句话: 新增 Mamba-1/2 纯 SSM 模型推理支持
- 推荐动作: 值得精读, 尤其是 Mamba-1 如何在不新增 state cache 实现的前提下复用 Mamba2 后端。建议按以下顺序阅读: 先看 configs/mamba.py 与 configs/mamba_utils.py 的 Mamba2StateShape.create_full_rank (理解 num_heads = intermediate_size、head_dim = 1 的状态布局映射), 再看 layers/attention/mamba/mamba1.py 的 _ssm_params/forward (理解三元返回契约与 radix/ 投机断言的由来), 然后是 models/falcon_mamba.py 的 use_bc_dt_rms 参数化与统一 load_weights (review 驱动的复用设计), 最后看 configs/model_config.py 的 SSM 分支与 utils/hf_transformers/config.py 的 _try_load_raw_mamba_config。生产落地前建议确认两点: 一是长上下文场景是否接受当前 Mamba-1 prefill 吞吐 (等待 Triton scan kernel), 二是运维侧能否接受 --disable-radix-cache 这一约束; 此外可在非 XPU 设备上补一条 runtime 冒烟测试以扩大覆盖。

功能与动机

PR body 明确指出, Mamba 家族 (Mamba2 的 mistralai/Mamba-Codestral-7B-v0.1, Mamba-1 的 tiuuai/falcon-mamba-7b、state-spaces/mamba-130m-hf 及 raw 系列) 此前在 SGLang 上完全无法服务——「they crashed at startup or fell through to attention code paths they don't fit」。作者归纳了四类根因: 这些 config 没有 num_attention_heads/head_dim, 形状推导在任何权重加载之前就已失败; 纯 SSM 路径被错误地接到 paged decoder KV cache 上 (SSM 本身不保留 KV); Mamba2 checkpoint 使用 backbone.* 命名且没有 max_position_embeddings; raw state-spaces config 既无 model_type 也无 architectures, AutoConfig 直接拒绝。动机即补齐 SGLang 的模型支持矩阵并让这些 checkpoint 能按原样被服务。

实现拆解

1. 新增模型入口 (全部由 SSM 分支门控): python/sglang/srt/models/mamba2.py 新增 Mamba2ForCausalLM (pre-norm + MambaMixer2, 无 MLP 子层); python/sglang/srt/models/falcon_mamba.py 新增 FalconMambaForCausalLM (pre-norm + MambaMixer1, 带 use_bc_dt_rms 开关); python/sglang/srt/models/mamba.py 仅 33 行, 继承 FalconMambaForCausalLM 并把 use_bc_dt_rms 置为 False。三者都在 decoder layer 里通过

get_attn_backend().forward(self.mixer, ...) 调用后端，并显式传
use_triton_causal_conv=True 让 conv 在非 CUDA (XPU) 后端也可用；EntryClass 承接
模型注册，既有 MHA/MLA 分支未被动到。

2. 新增 Mamba-1 混合器：python/sglang/srt/layers/attention/mamba/mamba1.py 的
MambaMixer1 按 HF 权重名 (in_proj/conv1d/x_proj/dt_proj/A_log/D/out_proj) 组织，
rms_normalize 复刻 Falcon 对 B/C/dt 的无权 RMSNorm，_ssm_params 负责 dt/B/C
的切分与归一化；forward 按 varlen 约定把 token 切成「前 prefill、后 decode」，
_forward_prefill 走纯 torch 逐 token selective scan，_forward_decode 复用共享
selective_state_update kernel。两条前置断言 (is_target_verify、
has_mamba_track_mask) 把不支持的场景变成 fail fast。
3. 复用 Mamba2 后端与显存池：python/sglang/srt/configs/mamba_utils.py 新增
Mamba2StateShape.create_full_rank (review 中从 create_mamba1 改名)，以
num_heads = intermediate_size、head_dim = 1 表达 full-rank 状态，使 Mamba-1 直接
骑上 Mamba2CacheParams 与 Mamba2AttnBackend；configs/mamba.py 的
_mamba1_cache_params 与 configs/mamba2.py 的
Mamba2Config.mamba2_cache_params 分别产出 cache 参数 (后者复用原
Mamba2StateShape.create)。
4. 配置与接线 (additive)：configs/model_config.py 新增 AttentionArch.SSM、
PURE_SSM_ARCHITECTURES、PURE_MAMBA1_ARCHITECTURES，在
_derive_model_shapes 中把纯 SSM 的 head_dim 设为 0、attention_arch 设为 SSM，
get_total_num_kv_heads/get_num_attention_heads 对 SSM 返回 0；
model_executor/model_runner_components/attention_backend_setup.py 增加 SSM 分
支直接构造 Mamba2AttnBackend (init_cublas() 上移到早返回之前，
Mamba2AttnBackend 改惰性导入)；configs/hybrid_arch.py 改为对三个新 config 类做
isinstance 匹配。
5. raw checkpoint 支持与配套：utils/hf_transformers/config.py 新增
_try_load_raw_mamba_config，在 AutoConfig.from_pretrained 之前用
PretrainedConfig.get_config_dict 识别只有 d_model/n_layer/ssm_cfg 的 state-spaces
配置并构造 MambaConfig (含 pad_vocab_size_multiple 的 vocab 对齐)，并在 parse
阶段对已是 SGLang config 的实例跳过 registry 重载。测试与文档同步：
test/srt/models/test_mamba2.py (CPU，覆盖权重名重映射、inv_freq 跳过、未匹配键不
致命)、test/registered/e2e/xpu/test_xpu_mamba1_runtime.py (XPU 真机
prefill→decode 与贪心确定性，守护 mixer/ 后端三元契约)、
docs/docs/supported-models/generative_models.mdx 增加条目。

关键文件：

- python/sglang/srt/layers/attention/mamba/mamba1.py (模块 状态空间层；类别 source
；类型 core-logic；符号 rms_normalize, MambaMixer1, _ssm_params, forward)：本
PR 的技术核心：新增 Mamba-1 selective-scan 混合器，承载 full-rank A、可选 B/C/dt
RMSNorm、prefill 纯 torch 扫描与 decode 复用共享 kernel，并定义了与
Mamba2AttnBackend 的三元返回契约和两条 fail-fast 断言。

- python/sglang/srt/models/falcon_mamba.py (模块 模型复用; 类别 source; 类型 data-contract; 符号 FalconMambaDecoderLayer, FalconMambaModel, FalconMambaForCausalLM, use_bc_dt_rms) : Falcon-Mamba (Mamba-1) 模型定义, 同时是 state-spaces 纯 Mamba 的基类: review 后把 use_bc_dt_rms 参数化、LM head 按 config.tie_word_embeddings 绑定、权重加载统一处理 backbone.*embeddings./embedding./norm_f 重映射。
- python/sglang/srt/configs/model_config.py (模块 模型配置; 类别 source; 类型 data-contract; 符号 AttentionArch.SSM, PURE_SSM_ARCHITECTURES, PURE_MAMBA1_ARCHITECTURES, _derive_model_shapes) : 全局模型配置入口的 SSM 接线: 新增 AttentionArch.SSM 与 PURE_SSM_ARCHITECTURES, 在 _derive_model_shapes 中为纯 SSM 设置 head_dim = 0 与 attention_arch = SSM, 并在 KV heads / attention heads 相关方法上对 SSM 返回 0。
- python/sglang/srt/configs/mamba.py (模块 配置钩子; 类别 source; 类型 dependency-wiring; 符号 _mamba1_cache_params, MambaConfig, FalconMambaConfig, full_attention_layer_ids) : 新增 Mamba-1 与 Falcon-Mamba 的 SGLang config 类, 按 NemotronHConfig 的模式把 mamba_chunk_size、full_attention_layer_ids、mamba2_cache_params 收敛为配置自身属性, 正是 review 中 roikoren755 要求的重构落点。
- python/sglang/srt/configs/mamba2.py (模块 配置基类; 类别 source; 类型 dependency-wiring; 符号 Mamba2Config, full_attention_layer_ids, mamba2_cache_params) : Mamba2 的 SGLang config 类, 把 mamba_chunk_size 别名与 mamba2_cache_params 从原本计划在 hybrid_arch 里的注入逻辑搬进配置类, 并复用 Mamba2StateShape.create。
- python/sglang/srt/utils/hf_transformers/config.py (模块 配置加载; 类别 source; 类型 core-logic; 符号 _try_load_raw_mamba_config) : 新增 _try_load_raw_mamba_config 预 AutoConfig 钩子, 解决 raw state-spaces checkpoint 无 model_type/architectures 导致 AutoConfig 直接拒绝的问题, 并在 parse 阶段跳过对 SGLang config 实例的 registry 重载。
- python/sglang/srt/models/mamba2.py (模块 模型定义; 类别 source; 类型 data-contract; 符号 Mamba2DecoderLayer, Mamba2Model, Mamba2ForCausalLM, load_weights) : Mamba2 主模型文件: Mamba2ForCausalLM 用 pre-norm + MambaMixer2 搭出无 MLP 的 SSM decoder, 权重加载把 backbone.*embeddings./norm_f/A_log 重映射到 SGLang 模块, 是本 PR 最早落地的模型入口。
- python/sglang/srt/models/mamba.py (模块 模型入口; 类别 source; 类型 data-contract; 符号 MambaForCausalLM) : state-spaces 纯 Mamba 的入口文件; review (ispobock) 要求复用后它只有 33 行, 通过继承 FalconMambaForCausalLM 并翻转 use_bc_dt_rms 表达差异, 是“模型文件互相 import”惯例的落地样例。
- python/sglang/srt/model_executor/model_runner_components/attention_backend_setup.py (模块 后端装配; 类别 source; 类型 dependency-wiring; 符号 build_attention_backends) : attention 后端装配的 SSM 分支: 直接构造 Mamba2AttnBackend 并返回 AttentionBackends; review 后 init_cublas() 上移到早返回

之前、Mamba 专属导入改为分支内惰性加载。

- `python/sglang/srt/configs/mamba_utils.py` (模块 缓存形状; 类别 source; 类型 core-logic; 符号 `create_full_rank`): 新增 `Mamba2StateShape.create_full_rank` (review 中由 `create_mamba1` 改名), 是把 Mamba-1 full-rank 状态塞进 Mamba2 head 布局的关键工具, 也是整套复用策略的基石。
- `test/registered/e2e/xpu/test_xpu_mamba1_runtime.py` (模块 端到端测试; 类别 test; 类型 test-coverage; 符号 `TestXPUMamba1Runtime`, `test_prefill_to_decode`, `test_greedy_is_deterministic`, `register_xpu_ci`): 回应 alphabtc1“补 runtime 测试”的要求: 启动真实 server (state-spaces/mamba-130m-hf) 验证 `prefill`→`decode` 生成与贪心确定性, 专门守护 `MambaMixer1` 与 `Mamba2AttnBackend` 的返回契约和 SSM state cache。
- `test/srt/models/test_mamba2.py` (模块 单元测试; 类别 test; 类型 test-coverage; 符号 `_param`, `_FakeMamba2`, `TestMamba2WeightRemap`, `test_backbone_names_are_remapped_and_loaded`): CPU 单测, 覆盖 `Mamba2ForCausalLM.load_weights` 的名称重映射 (`backbone./embeddings./norm_f/A_log`)、`inv_freq` 跳过与未匹配权重不致命; 不构造完整模型, 可在无 GPU 环境回归加载逻辑。
- `python/sglang/srt/configs/hybrid_arch.py` (模块 架构分发; 类别 source; 类型 core-logic; 符号 `mamba2_config`): review 明确要求把 `getattr(config, "_is_pure_mamba2", False)` 风格的标志判断改成对新 config 类的 `isinstance` 匹配, 这里只保留一行分流逻辑。
- `docs/docs/supported-models/generative_models.mdx` (模块 文档; 类别 docs; 类型 documentation): 文档侧登记新增支持的 Mamba 家族模型, 保证用户在支持矩阵中能查到这些 checkpoint。

关键符号: `MambaMixer1.forward`, `MambaMixer1._ssm_params`, `MambaMixer1._forward_prefill`, `MambaMixer1._selective_scan`, `MambaMixer1._forward_decode`, `rms_normalize`, `Mamba2ForCausalLM.load_weights`, `FalconMambaForCausalLM.load_weights`, `Mamba2StateShape.create_full_rank`, `ModelConfig._derive_model_shapes`, `ModelConfig.get_num_attention_heads`, `build_attention_backends`, `_try_load_raw_mamba_config`

关键源码片段

`python/sglang/srt/models/falcon_mamba.py`

Falcon-Mamba (Mamba-1) 模型定义, 同时是 state-spaces 纯 Mamba 的基类: review 后把 `use_bc_dt_rms` 参数化、LM head 按 `config.tie_word_embeddings` 绑定、权重加载统一处理 `backbone.*/embeddings./embedding./norm_f` 重映射。

```
class FalconMambaForCausalLM(nn.Module):
    """Falcon-Mamba (Mamba-1) 模型 + LM head.
```

同时作为 state-spaces 纯 Mamba 的基类 (`models/mamba.py` 只翻转 `use_bc_dt_rms`); LM head 是否 tie 取决于 `config.tie_word_embeddings`, 因此一个类覆盖两类 checkpoint。

"""

Falcon-Mamba 对 B/C/dt 做无权重 RMSNorm; 纯 Mamba 置 False
use_bc_dt_rms: bool = True

```
def __init__(self, config, quant_config=None, cache_config=None):
    super().__init__()
    self.config = config
    self.quant_config = quant_config

    self.model = FalconMambaModel(
        config=config,
        quant_config=quant_config,
        prefix="model",
        use_bc_dt_rms=self.use_bc_dt_rms,
    )
    self.lm_head = ParallelLMHead(
        config.vocab_size, config.hidden_size, prefix="lm_head"
    )
    # state-spaces 的 checkpoint 没有独立 lm_head, 需要与输入 embedding 共享权重;
    # Falcon-Mamba 自身 tie_word_embeddings=False, 走不到这个分支。
    if config.tie_word_embeddings: # 直读 config, 不做 getattr 防御式兜底
        self.lm_head = self.lm_head.tie_weights(self.model.embed_tokens)
    self.logits_processor = LogitsProcessor(config)
```

```
def load_weights(self, weights: Iterable[Tuple[str, torch.Tensor]]):
    params_dict = dict(self.named_parameters())
    loaded_params: Set[str] = set()
```

```
    for name, loaded_weight in weights:
        # 统一重映射: backbone.* -> model.*; embeddings. (-hf) 与 embedding. (raw
        # state-spaces) 都归到 embed_tokens.; norm_f -> norm。A_log 保持原名,
        # mixer 内部按  $A = -\exp(A\_log)$  计算。
        if name.startswith("backbone."):
            name = "model." + name[len("backbone.") :]
            name = name.replace("embeddings.", "embed_tokens.")
            name = name.replace("embedding.", "embed_tokens.")
            name = name.replace("norm_f.", "norm.")

        if name not in params_dict:
            logger.warning(f"Skipping parameter {name} - not found in model")
            continue

        param = params_dict[name]
        weight_loader = getattr(param, "weight_loader", default_weight_loader)
        weight_loader(param, loaded_weight)
        loaded_params.add(name)
```

```
unloaded_params = set(params_dict.keys()) - loaded_params
```

```

# tie 的 lm_head 天然不在 checkpoint 中，不应算作漏载参数
unloaded_params = {p for p in unloaded_params if not p.startswith("lm_head")}
if unloaded_params:
    logger.warning(
        f"The following parameters were not loaded: {unloaded_params}"
    )
return loaded_params

```

EntryClass = FalconMambaForCausalLM

python/sglang/srt/configs/model_config.py

全局模型配置入口的 SSM 接线：新增 AttentionArch.SSM 与 PURE_SSM_ARCHITECTURES，在 `_derive_model_shapes` 中为纯 SSM 设置 `head_dim = 0` 与 `attention_arch = SSM`，并在 KV heads / attention heads 相关方法上对 SSM 返回 0。

```

class AttentionArch(IntEnum):
    MLA = auto()
    MHA = auto()
    SSM = auto() # State Space Models (Mamba / Mamba2)，无注意力

```

```

# 纯 Mamba-1 (selective-scan) 架构；mixer 与状态布局一致，差异在各自模型文件里处理
PURE_MAMBA1_ARCHITECTURES = ("FalconMambaForCausalLM", "MambaForCausalLM")
# 纯 SSM 因果 LM：没有注意力，因此 HF config 里也没有 num_attention_heads / head_dim
PURE_SSM_ARCHITECTURES = ("Mamba2ForCausalLM",)

```

```

def _derive_model_shapes(self):
    # 纯 SSM 模型没有注意力头：把 head_dim 设为 0，让 KV cell 尺寸为 0，
    # 而不是对缺失的 num_attention_heads 做除法。
    is_pure_ssm = any(
        arch in self.hf_config.architectures for arch in PURE_SSM_ARCHITECTURES
    )
    self.head_dim = getattr(self.hf_text_config, "head_dim", None)
    if self.head_dim is None:
        if is_pure_ssm:
            self.head_dim = 0
        else:
            self.head_dim = (
                self.hf_text_config.hidden_size
                // self.hf_text_config.num_attention_heads
            )
    setattr(self.hf_text_config, "head_dim", self.head_dim)

```

... 其余上下文长度 / v_head_dim 推导保持原逻辑 ...

```

# 注意力架构判定：纯 SSM 直接落到 SSM 分支 (head_dim 已为 0)，其余保持 MHA
if is_pure_ssm:

```

```

        self.attention_arch = AttentionArch.SSM
    else:
        self.attention_arch = AttentionArch.MHA
    # Mamba2 没有 num_attention_heads, 这里保留 None, 由下面的取值方法兜底
    self.num_attention_heads = getattr(
        self.hf_text_config, "num_attention_heads", None
    )

def get_num_attention_heads(self, tensor_parallel_size) -> int:
    # 纯 SSM 没有注意力头 (num_attention_heads 为 None) ; 用 0 兜底,
    # 避免 MFU 指标上报时出现 None // tp_size。
    total_num_attention_heads = self.num_attention_heads or 0
    return max(1, total_num_attention_heads // tensor_parallel_size)

def get_total_num_kv_heads(self) -> int:
    # ... 其它分支 ...
    # Mamba SSM 没有注意力, 也就没有 KV heads
    if self.attention_arch == AttentionArch.SSM:
        return 0
    # 非 GQA 模型下 KV heads 等于注意力头数
    return self.hf_text_config.num_attention_heads

```

评论区精华

1. 配置组织方式被要求重构 (roikoren755) : 评论指出 `_is_pure_mamba2/_is_pure_mamba1` 注入标志“aren't really needed”, 并建议像 `NemotronHConfig` 一样用属性“initialized inside the config's init”; `hybrid_arch.py` 里也被要求改成 `isinstance` 检查。作者随后新增 `configs/mamba.py/configs/mamba2.py` 两个 `SGLang config` 类承载 `mamba_chunk_size`、`full_attention_layer_ids`、`mamba2_cache_params`, 删掉标志位。
2. 模型文件重复被要求复用 (ispobock) : “This file and `falcon_mamba.py` differ only in this flag and the tied `lm_head.141` of `219` model files import from another one. Reuse instead of copying?”——最终 `MambaForCausalLM` 变成 `FalconMambaForCausalLM` 的 33 行子类, 只翻转 `use_bc_dt_rms`; 同时按 `claude/rules/no-getattr-defensive.md` 要求把 `config.tie_word_embeddings` 改为直读而非 `getattr` 防御。
3. `mixer` 返回契约回归被查出 (alphabetc1) : “`Mamba2AttnBackend.forward()` expects `mixer.forward()` to return three values, but this returns only two.”作者确认这是后端契约改三元返回 (#37836) 后的回归, 改为 `return mixer_out, None, None`, 并补了 XPU runtime 测试来守护。
4. `radix tracking` 未真正禁用 (alphabetc1) : 评论质疑“documented as unsupported but not actually disabled... leave tracked prefix states unwritten while the backend still allocates tracking destinations?”作者改为断言 `has_mamba_track_mask` 未设置并要求 `--disable-radix-cache`。

5. prefill 逐 token 循环的性能质疑 (alphabeta1) : 要求提供 1K/4K prefill benchmark; 作者给出 falcon-mamba-7b (XPU、tp1) 实测 prefill 1024 = 8.9 s / 115 tok/s、4096 = 22.1 s / 185 tok/s, 承认 scan 是瓶颈, Triton kernel 留作 follow-up。
6. 若干细节被逐一纠正: eager_runner.py 的三处 attn_backend is not None 守卫被 ispobock 指出多余 (“attn_backend is never None here”) 后全部删除并回到 upstream ; attention_backend_setup.py 的 SSM 早返回跳过 init_cublas() warmup、以及 Mamba2AttnBackend 顶层导入被要求移入 SSM 分支; _MAMBA1_CHUNK_SIZE 被要求改名为 _MAMBA1_CACHE_CHUNK_SIZE; metrics 侧 None // tp_size 通过 get_num_attention_heads 返回 0 修复。
- MambaMixer1.forward 返回值与 Mamba2AttnBackend 三元契约不匹配 (correctness): 改为 return mixer_out, None, None, 并新增 test/registered/e2e/xpu/test_xpu_mamba_1_runtime.py 的 prefill→decode 用例守护该契约; 该测试注释明确写到“Before the 3-tuple fix this raised 'not enough values to unpack'”。
- radix mamba-state tracking 只是文档标注不支持, 并未真正禁用 (correctness): 增加 assert not metadata.has_mamba_track_mask, fail fast 并要求以 --disable-radix-cache 启动; XPU runtime 测试也相应加上该参数。
- Mamba-1 prefill 逐 token Python 循环的性能与 benchmark 要求 (performance): 本 PR 接受该限制作为已知代价 (换取设备无关性), Triton Mamba-1 scan kernel 作为明确 follow-up; PR body 也写明“prefill throughput is low”。
- 配置组织: 应使用 SGLang config 类而不是注入 _is_pure_mamba* 标志 (design): 新增 configs/mamba.py (MambaConfig/FalconMambaConfig) 与 configs/mamba2.py (Mamba2Config), 删除注入标志与 hybrid_arch 内的 inline 构建; create_mamba1 重命名为 create_full_rank。
- mamba.py 与 falcon_mamba.py 高度重复, 应复用而非复制 (design): MambaForCausalLM 改为继承 FalconMambaForCausalLM 并只翻转 use_bc_dt_rms (mamba.py 缩到 33 行); falcon_mamba.py 把开关参数化、LM head 按 tie_word_embeddings 绑定、权重加载器统一。
- 纯 SSM 的注意力头 / KV 头 fallback 会掩盖新模型问题 (correctness): get_total_num_kv_heads 改为 AttentionArch.SSM 时显式 return 0, 其余仍直读 num_attention_heads; get_num_attention_heads 用 self.num_attention_heads or 0 兜底, metrics 上报不再崩溃。
- SSM 早返回跳过 init_cublas warmup 与 Mamba 依赖的导入位置 (performance): init_cublas() 上移到 SSM 早返回之前; Mamba2AttnBackend 改为分支内惰性导入, AttentionArch 保留在顶部 (每次都要做架构判断) 。
- eager_runner 中多余的 attn_backend is not None 守卫 (design): 三处守卫全部删除, eager_runner.py 回到 upstream 状态, SSM 不再有专门的 runner 改动。
- 命名细节: _MAMBA1_CHUNK_SIZE 误导、以及常量语义 (style): 重命名为 _MAMBA1_CACHE_CHUNK_SIZE, 并在注释中说明 Mamba-1 本身没有 chunk size。

风险与影响

- 风险:

1. Mamba-1 prefill 性能是明确短板 (`layers/attention/mamba/mamba1.py` 的 `_forward_prefill/_selective_scan`) : 多 token prefill 用纯 torch 逐 token 循环, 每步多次设备算子; 作者实测 falcon-mamba-7b (XPU、tp1) 1024 token 需 8.9 s (115 tok/s)、4096 token 22.1 s (185 tok/s)。长 prompt 首 token 延迟会非常明显, Triton Mamba-1 scan kernel 是尚未落地的 follow-up。
2. 使用约束带有隐性成本: Mamba-1 必须在 `--disable-radix-cache` 下运行, 否则 `has_mamba_track_mask` 断言直接失败; 同时不支持投机解码 (`is_target_verify` 断言)。这两点只在断言 / 文档中体现, 用户若沿用默认启动参数会直接报错。
3. 全局配置路径被触碰: `configs/model_config.py` 的 `head_dim` 推导与 `attention_arch` 判定、`attention_backend_setup.build_attention_backends` 都是所有模型共享的路径; 虽然由 `is_pure_ssm / AttentionArch.SSM` 严格门控, 但 `get_total_num_kv_heads` 返回 0、`get_num_attention_heads` 的 or 0 兜底会影响 KV 内存估算与 MFU 指标, 需要回归验证非 SSM 模型。
4. 权重加载采用字符串替换式重映射: `load_weights` 里 `name.replace("embeddings.", "embed_tokens.")`、`name.replace("embedding.", "embed_tokens.")`、`name.replace("norm_f.", "norm.")` 属于子串替换, 理论上可能误伤包含同样子串的键; CPU 单测 (`test/srt/models/test_mamba2.py`) 只覆盖 Mamba2 的少量样例, FalconMambaForCausalLM.load_weights (含 `embedding.` 单数分支与 `tied lm_head` 过滤) 没有对应单测。
5. 测试覆盖偏窄: runtime 覆盖只有 `test/registered/e2e/xpu/test_xpu_mamba1_runtime.py` (注册在 `stage-b-test-1-gpu-xpu`), CUDA/ROCm 上 Mamba 家族的运行时路径无 CI 覆盖; Mamba2 模型本身也只有 CPU 权重重映射测试, 没有端到端运行测试。
6. 实现假设类风险: Mamba-1 借用常量 `_MAMBA1_CACHE_CHUNK_SIZE = 256` 仅用于约束 conv window; `dt_bias_zero` 以零 buffer 规避 kernel 的 `dt_bias is None` 分支。这些是脆弱耦合点, 后续若后端 kernel 改动需要同步。- 影响: 用户侧: SGLang 可首次服务 Mamba 家族模型, 包括 Mamba-Codestral-7B-v0.1、falcon-mamba-7b/instruct、state-spaces/mamba-130m-hf 以及 raw 的 mamba-130m/790m/2.8b (raw 需要额外的 `--tokenizer-path` EleutherAI/gpt-neox-20b)。系统侧: 纯 SSM 模型的 KV cache 为 0 字节, 并发上限不再由 KV 显存决定, 而由 `--max-total-tokens` 与 Mamba state-cache 槽位约束, 容量规划模型与 attention 模型完全不同; Mamba-1 需要关闭 radix cache, 影响前缀复用带来的调度收益。生态与团队侧: 这是 SGLang 模型矩阵的一次扩展, 并新增 `AttentionArch.SSM` 这一架构维度, 为后续纯 SSM / 混合 SSM 模型 (NemotronHConfig 模式) 提供了可复用的接线范式; 验证主要在 Intel XPU (2x Arc Pro B60) 上完成, 对 Intel 后端生态是正向投入。对既有模型: 所有改动由 SSM 架构 / 标志门控, MHA/MLA 模型保持原有代码路径与性能, 风险面局限在被门控的配置推导与后端构造分支。- 风险标记: Mamba-1 prefill 逐 token 扫描性能差, Mamba-1 需 `--disable-radix-cache`, 暂不支持投机解码, 运行时测试仅覆盖 XPU, 字符串替换式权重重映射未全量覆盖, 触碰全局 `model_config` 推导路径

关联脉络

- PR #37818 [Bugfix] Track DFlash Mamba state at checkpoint boundaries: 同属 Mamba/SSM 状态缓存语义的改动：该 PR 修 DFlash 在 Mamba 边界漏检 checkpoint 状态，而本 PR 则为 Mamba-1 增加 radix 状态跟踪的断言与 `--disable-radix-cache` 约束，二者共同界定了 SGLang 中 SSM 状态的正确使用边界。
- PR #37584 [Unified Tree] Port SWA Branching-Point Caching to the Rust TreeCore: 同样涉及 mamba 组件 (`rust/sglang-radix-tree/src/components/mamba.rs`) 与 `mem_cache` 的状态复用路径，是本 PR 中新引入的 mamba state cache 约束在统一树 / 前缀缓存层面的相邻工作。
- PR #37565 [NPU] Support DFlash speculative decoding for MiMo-V2.5-Pro (mxfp4): 同为跨硬件后端上打通新模型能力的特性 PR：该 PR 在 NPU 上解锁投机解码，而本 PR 明确断言 Mamba-1 暂不支持投机解码 (`is_target_verify`)，说明新模型支持与投机解码能力在不同后端上的推进节奏存在并行与互补关系。