PR #36070 完整报告

sgl-project/sglang

[Diffusion] Load pruned MiniMax H3 components natively

合并时间：2026-08-24 16:39

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/36070

执行摘要

- 一句话：原生加载剪枝 MiniMax H3 检查点，LoRA 层新增常量偏移机制
- 推荐动作：值得精读。 1) MiniMaxH3DiTModel.prepare_lora_adapter 展示了如何在不改变运行时的情况下，通过仿射基投影兼容剪枝权重与外部 LoRA，数学变换与错误防御（宽度统一校验、零基拒绝）都值得借鉴。 2) LoRA 层的 lora_output_offset 是通用扩展，merge / FSDP / commit-as-base 三种边界条件的处理可作为工程样板。 3) 建议结合 test_lora_commit_as_base.py 的用例理解动态与合并模式的偏移缩放语义。

功能与动机

PR body 明确说明目标：support multimodalart/MiniMax-H3-Pruned without introducing a non-native model runtime or a separate loader。剪枝版检查点用 rank-8 曲线表替代了原 time_embedder MLP，且 AdaLN 投影以 folded_bias 形式存储；与此同时，面向完整模型训练发布的 AdaLN LoRA 仍以 2688 维仿射空间为输入。若直接加载会导致维度错配或语义缺失，因此需要在配置层、模型参数层与 LoRA 层同时做适配，保证外部检查点与官方 LoRA 都能零转换使用。

实现拆解

1. 识别剪枝 schema 并建立配置映射

python/sglang/multimodal_gen/configs/models/dits/minimax_h3.py 的 MiniMaxH3DiTConfig.update_model_arch 识别 _class_name == "MiniMaxH3PrunedTransformer3DModel"，将源配置的 time_embed_dim（完整 2688 维）保存到新字段 adaln_affine_input_dim，用 adaln_rank（8）覆盖内部 time_embed_dim，用 time_table_size 设置 adaln_curve_grid（如 1025）。MiniMaxH3DiTArchConfig.param_names_mapping 新增三条规则：time_embedder.table → adaln_t_table、norm_out.folded_bias → final_layer.adaln_proj.linear.bias、transformer_blocks.N.adaln_proj.folded_bias → blocks.N.adaln_proj.linear.bias，把剪枝检查点的曲线表与折叠偏差接入原生命名空间。transformer_loader.py 中 checkpoint_uses_diffusers_layout 的判断从“等于 MiniMaxH3Transformer3DModel”改为“不等于原生类名”，使 pruned 类名也走 diffusers 布局转换。

2. 注册仿射元数据参数

python/sglang/multimodal_gen/runtime/models/dits/minimax_h3.py 的 MiniMaxH3DiTModel 在 _aliases 中增加 MiniMaxH3PrunedTransformer3DModel 让加载器解析到原生实现；当 adaln_affine_input_dim 非空时注册 adaln_basis（[time_embed_dim, adaln_affine_input_dim]）与 adaln_mean（[adaln_affine_input_dim]）两个 fp32 参数，作为仿射投影锚点。post_load_weights 将二者纳入 fp32 保持校验，并在 adaln_t_table 存在时把这两项追加进 fp32 参数白名单。

3. LoRA 层新增常量输出偏移机制

python/sglang/multimodal_gen/runtime/layers/lora/linear.py 将 LoRAWeightEntry 从 6 元组扩展为 7 元组（末尾追加 output_offset）。BaseLayerWithLoRA 新增 lora_output_offset / has_lora_output_offset 状态与三个方法：_scaled_lora_output_offset（按 strength 与 alpha/rank 缩放偏移）、_active_lora_output_offset（区分动态 / 合并模式取偏移）、_add_lora_output_offset（forward 统一出口叠加偏移）。所有 LoRA 线性层（BaseLayerWithLoRA、ColumnParallelLinearWithLoRA、RowParallelLinearWithLoRA、ReplicatedLinearWithLoRA）的 forward 末尾都改经 _add_lora_output_offset；merged 分支条件从 self.merged or self.disable_lora 改为 self.disable_lora or (self.merged and not self.has_lora_output_offset)，保证带偏移的适配器在 merge 后仍能补上常量项。同时限制带偏移的适配器不能 commit_merged_as_base，FSDP 分片权重下不能 merge。

4. 模型侧 LoRA 投影钩子与管线接线

models/dits/base.py 的 BaseDiT 新增 prepare_lora_adapter 默认透传实现作为可扩展钩子；minimax_h3.py 覆写该方法：对宽度为 adaln_affine_input_dim 的 AdaLN LoRA A 因子执行 a @ basis.T 投影到内部 rank-8 曲线空间，并把均值项 b @ (a @ mean) 生成为 lora_output_offset 常量偏移，全程在 float64 下计算后转回 float32。pipelines_core/lora/pipeline.py 在 _apply_lora_to_layers 中把 name + ".lora_output_offset" 传给 set_lora_weights，在 load_lora_adapter 名称归一化完成后调用 transformer.prepare_lora_adapter 做模型特定变换。

5. 测试配套

test_minimax_h3_dit_contract.py 新增 test_pruned_adaln_lora_projection_preserves_affine_term，用 SimpleNamespace 验证投影数学与偏移生成，并覆盖 folded_bias / time_embedder.table 映射与 pruned 配置解析；test_lora_commit_as_base.py 新增 test_lora_output_offset_tracks_dynamic_and_merged_scale，验证动态与合并模式下偏移缩放一致，且带偏移的适配器不允许 commit 为权重基座。

关键文件：

- python/sglang/multimodal_gen/runtime/layers/lora/linear.py（模块 LoRA 层；类别 source；类型 core-logic；符号 _scaled_lora_output_offset, _active_lora_output_offset, _add_lora_output_offset, set_lora_weights）：LoRA 层核心逻辑：扩展 LoRAWeightEntry 为 7 元组，新增 lora_output_offset 常量偏移机制，统一所有 LoRA 线性层 forward 出口，并处理 merge / FSDP / commit-as-base 边界。
- python/sglang/multimodal_gen/runtime/models/dits/minimax_h3.py（模块 H3 模型；类别 source；类型 data-contract；符号 prepare_lora_adapter）：模型契约核心：新增

MiniMaxH3PrunedTransformer3DModel 别名、adaln_basis / adaln_mean 参数注册，以及 AdaLN LoRA 仿射投影的 prepare_lora_adapter 实现。

- python/sglang/multimodal_gen/runtime/pipelines_core/lora/pipeline.py（模块 LoRA 管线；类别 source；类型 dependency-wiring）：LoRA 管线接线：set_lora_weights 传递 lora_output_offset，load_lora_adapter 后调用 prepare_lora_adapter 钩子。
- python/sglang/multimodal_gen/configs/models/dits/minimax_h3.py（模块 模型配置；类别 source；类型 data-contract）：数据契约：识别 MiniMaxH3PrunedTransformer3DModel schema，新增 adaln_affine_input_dim 字段与 folded_bias / table 映射规则。
- python/sglang/multimodal_gen/runtime/models/dits/base.py（模块 模型基类；类别 source；类型 data-contract；符号 prepare_lora_adapter）：基类扩展：BaseDiT 新增 prepare_lora_adapter 默认透传实现，作为所有 DiT 模型的 LoRA 预处理钩子。
- python/sglang/multimodal_gen/test/unit/test_minimax_h3_dit_contract.py（模块 契约测试；类别 test；类型 test-coverage；符号 test_pruned_adaln_lora_projection_preserves_affine_term）：契约测试：验证 AdaLN LoRA 投影数学、偏移生成、folded_bias 映射与 pruned 配置解析。
- python/sglang/multimodal_gen/test/unit/test_lora_commit_as_base.py（模块 LoRA 测试；类别 test；类型 test-coverage；符号 test_lora_output_offset_tracks_dynamic_and_merged_scale）：LoRA 测试：验证动态与合并模式下 output_offset 缩放一致，且带偏移适配器拒绝 commit-as-base。
- python/sglang/multimodal_gen/runtime/loader/component_loaders/transformer_loader.py（模块 模型加载；类别 source；类型 core-logic）：加载逻辑关键一行：checkpoint_uses_diffusers_layout 的判断从指定类名改为非原生类名，使 pruned 类名走 diffusers 布局转换。

关键符号：MiniMaxH3DiTModel.prepare_lora_adapter, BaseDiT.prepare_lora_adapter, BaseLayerWithLoRA._scaled_lora_output_offset, BaseLayerWithLoRA._active_lora_output_offset, BaseLayerWithLoRA._add_lora_output_offset, BaseLayerWithLoRA.set_lora_weights, BaseLayerWithLoRA.commit_merged_as_base, MiniMaxH3DiTConfig.update_model_arch, BaseLayerWithLoRA.forward

关键源码片段

python/sglang/multimodal_gen/runtime/layers/lora/linear.py

LoRA 层核心逻辑：扩展 LoRAWeightEntry 为 7 元组，新增 lora_output_offset 常量偏移机制，统一所有 LoRA 线性层 forward 出口，并处理 merge / FSDP / commit-as-base 边界。

```
def _scaled_lora_output_offset(
    self,
    offset: torch.Tensor | None,
    strength: float,
    rank: int | None,
    alpha: int | None,
```

```
) -> torch.Tensor | None:
    # 常量输出偏移同样要经过 strength 与 alpha/rank 缩放，
    # 使 merge 前后、动态与静态路径的语义保持一致。
    if offset is None:
        return None
    offset = self.slice_lora_b_weights(offset.unsqueeze(-1)).squeeze(-1)
    scale = strength
    if rank is not None and alpha is not None and rank != alpha:
        scale *= alpha / rank
    return offset if scale == 1.0 else offset * scale


def _active_lora_output_offset(self) -> torch.Tensor | None:
    # disable 或从未设置偏移时直接返回 None，保证老路径零额外开销。
    if self.disable_lora or not self.has_lora_output_offset:
        return None
    if not self.merged:
        # 动态模式：取当前适配器的偏移。
        return self._scaled_lora_output_offset(
            self.lora_output_offset,
            self.strength,
            self.lora_rank,
            self.lora_alpha,
        )
    # 合并模式：多适配器的偏移必须逐项缩放后累加。
    combined = None
    for _, _, _, strength, rank, alpha, offset in self.lora_weights_list:
        scaled = self._scaled_lora_output_offset(offset, strength, rank, alpha)
        if scaled is not None:
            combined = scaled if combined is None else combined + scaled
    return combined


def _add_lora_output_offset(self, output: torch.Tensor) -> torch.Tensor:
    # 所有 LoRA 线性层 forward 的统一出口：在最终输出上补齐常量偏移。
    offset = self._active_lora_output_offset()
    if offset is None:
        return output
    return output + offset.to(device=output.device, dtype=output.dtype)
```

python/sglang/multimodal_gen/runtime/models/dits/minimax_h3.py

模型契约核心：新增 MiniMaxH3PrunedTransformer3DModel 别名、adaln_basis / adaln_mean 参数注册，以及 AdaLN LoRA 仿射投影的 prepare_lora_adapter 实现。

```
def prepare_lora_adapter(
    self, adapter: dict[str, torch.Tensor]
) -> dict[str, torch.Tensor]:
    """Project released-checkpoint AdaLN LoRAs onto pruned coordinates."""
    # 剪枝模型把完整 2688 维 AdaLN 条件压缩到 rank-8 曲线空间；
```

```
# 未启用剪枝路径（adaln_affine_input_dim 为 None）时直接透传适配器。
full_width = self.arch.adaln_affine_input_dim
if full_width is None:
    return adapter

# 只关心 AdaLN 投影的 LoRA A 因子，attention / FFN 的 LoRA 不受维度变化影响。
suffix = ".adaln_proj.linear.lora_A"
a_keys = sorted(key for key in adapter if key.endswith(suffix))
if not a_keys:
    return adapter

# 输入宽度必须全局统一：已等于内部曲线宽度则直接可用，
# 等于完整仿射宽度才需要投影，否则拒绝加载以避免静默错配。
widths = {int(adapter[key].shape[-1]) for key in a_keys}
if widths == {self.arch.time_embed_dim}:
    return adapter
if widths != {full_width}:
    raise ValueError(
        "MiniMax H3 pruned AdaLN LoRA inputs must be uniformly "
        f"{self.arch.time_embed_dim} or {full_width} wide, got "
        f"{sorted(widths)}."
    )

# 仿射基与均值来自组件 checkpoint，是整个投影的数学锚点。
basis = self.adaln_basis
mean = self.adaln_mean
assert basis is not None and mean is not None
if isinstance(basis, DTensor):
    basis = basis.full_tensor()
    mean = mean.full_tensor()
if torch.count_nonzero(basis).item() == 0:
    raise ValueError(
        "MiniMax H3 pruned LoRA projection requires adaln_basis and "
        "adaln_mean from the component checkpoint."
    )

projected = dict(adapter)
work_device = adapter[a_keys[0]].device
# 全程在 float64 下计算，避免 float32 矩阵乘累积误差；
work_basis = basis.to(device=work_device, dtype=torch.float64)
work_mean = mean.to(device=work_device, dtype=torch.float64)
for a_key in a_keys:
    b_key = a_key[: -len("lora_A")] + "lora_B"
    if b_key not in adapter:
        raise ValueError(f"MiniMax H3 AdaLN LoRA is missing {b_key!r}.")
    a = adapter[a_key]
    b = adapter[b_key]
    a64 = a.to(torch.float64)
    b64 = b.to(device=work_device, dtype=torch.float64)
```

```
            # 关键变换 1：LoRA A 从完整仿射空间经 basis.T 投影到内部 rank-8 曲线空间。
            projected[a_key] = (a64 @ work_basis.T).to(torch.float32)
            # 关键变换 2：输入均值项 x @ mean 无法被线性投影吸收，
            # 拆成常量输出偏移（lora_output_offset），由 LoRA 层在输出端以加法补偿。
            projected[a_key[: -len("lora_A")] + "lora_output_offset"] = (
                b64 @ (a64 @ work_mean)
            ).to(torch.float32)

        logger.info(
            "Projected %d MiniMax H3 AdaLN LoRA modules from width %d to %d",
            len(a_keys),
            full_width,
            self.arch.time_embed_dim,
        )
        return projected
```

评论区精华

本 PR 无 review 评论记录（comments_count 与 review_comments_count 均为 0），无法提取讨论线程。不过从两次 commit 的演进可以观察到一条重要信息：首个提交只完成检查点 schema 加载，第二个提交标题为 "fix: project minimax h3 lora onto pruned adaln (#36075)"，说明 AdaLN LoRA 投影与 output_offset 机制是在后续修正中补齐的——这也解释了为何需要在 LoRA 层引入通用常量偏移，而非在 H3 模型内部做特判。

- 暂无高价值评论线程

风险与影响

- 风险：
  1. LoRA 元组契约变更风险：LoRAWeightEntry 从 6 元组扩展为 7 元组，仓库内所有解包点（_merge_lora_into_data、_should_merge_in_fp32、merge_lora_weights、_active_lora_output_offset）必须同步更新；本 PR 已覆盖，但外部扩展代码若直接构造该元组会破坏。
  2. forward 语义变化影响所有 LoRA 路径：ColumnParallelLinearWithLoRA / RowParallelLinearWithLoRA 的 merged 分支条件改变，带偏移的适配器 merge 后仍走动态分支，会引入额外的 _active_lora_output_offset 调用与张量加法；无偏移的常规 LoRA 仅为纯分支判断，开销可忽略。
  3. 数值精度依赖：投影计算依赖 adaln_basis / adaln_mean 的 fp32 精度与完整性，若 checkpoint 中基为零或参数精度被下游工具链降为 fp16/bf16，加载会直接报错（有意的安全设计，但强依赖 checkpoint 质量）。
  4. Extra CI 失败未排除：PR Test (Extra) 显示失败（Run #32685261449），PR 未附失败原因或结论，合并前未见排除记录。
- 影响：
  - 用户侧：可直接加载 multimodalart/MiniMax-H3-Pruned 检查点并直接使用面向完整模型发布的官方 LoRA，无需手动转换权重。

- 系统侧：LoRA 层新增通用常量偏移机制，所有 diffusion LoRA 线性层的 forward 出口统一经过 _add_lora_output_offset，为其他模型复用该机制提供了基础；同时 prepare_lora_adapter 进入 BaseDiT 基类，成为所有 DiT 模型的 LoRA 预处理入口。
- 团队侧：影响范围集中在 multimodal_gen 子系统的 LoRA 层与 H3 模型契约，不涉及 SRT 调度器、注意力内核等核心推理路径；LoRA 数据结构的扩展属于向后兼容设计（新增字段带默认值）。
- 风险标记：LoRA 通用路径变更，元组契约解包遗漏风险，FP32 精度依赖，Extra CI 未通过，新参数加载依赖

关联脉络

- PR #36067 [Diffusion] Load Diffusers MiniMax H3 components natively: 本 PR 的直接承接者：36067 建立了 Diffusers MiniMax H3 原生加载路径，本 PR 在其基础上扩展 pruned schema 与 AdaLN LoRA 投影。
- PR #36075 [Diffusion] fix: project minimax h3 lora onto pruned adaln: 第二个 commit 的提交信息中显式引用 #36075，对应 AdaLN LoRA 投影修正，与本 PR 的 prepare_lora_adapter 直接相关。
- PR #36076 [Diffusion] Support compact Qwen3-VL conditioning for MiniMax H3: 同一 MiniMax H3 功能线，围绕 H3 条件编码与加载能力的连续增强。
- PR #36080 [Diffusion] Support hybrid MiniMax H3 conditioning: 同一 MiniMax H3 功能线，扩展 H3 的 conditioning 能力，与本 PR 同属该模型在 multimodal_gen 中的系列演进。