PR #45993 完整报告

vllm-project/vllm

[Model] Remove MiniMaxText01, MiniMaxVL01, MiniMaxForCausalLM

合并时间：2026-06-22 15:20

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/45993

执行摘要

- 一句话：移除已淘汰的 MiniMax Text-01 和 VL-01 模型架构
- 推荐动作：该 PR 值得精读，展示了如何安全地淘汰旧模型架构：通过 _PREVIOUSLY_SUPPORTED_MODELS 提供清晰错误提示替代静默失败，同时保留共享组件避免影响其他模型。审查过程也体现了跨团队协作：作者、MoE 专家、工具调用工程师共同确认删除范围。建议团队后续淘汰其他旧架构时参考此流程。

功能与动机

MiniMax Text-01 和 VL-01 已被 MiniMax M2 和 M3 全面取代，继续维护旧架构会增加代码维护负担和混淆风险。本次清理移除了不再使用的模型定义、工具解析器与相关测试，使代码库更精简。同时通过_PREVIOUSLY_SUPPORTED_MODELS机制保留了向后兼容的提示信息。

实现拆解

1. 删除核心模型代码：移除 vllm/model_executor/models/minimax_text_01.py 和 minimax_vl_01.py，包含所有模型类（MiniMaxText01MLP、MiniMaxText01MoE、MiniMaxVL01MultiModalProjector 等）。
2. 删除 MiniMax M1 工具解析器：根据审查者建议，移除 vllm/tool_parsers/minimax_tool_parser.py 及其相关的工具调用模板（tool_chat_template_minimax_m1.jinja）和 Rust 快照测试。
3. 更新模型注册表：在 vllm/model_executor/models/registry.py 中将四个旧架构添加到 _PREVIOUSLY_SUPPORTED_MODELS，并移除其对应的模型字典条目。
4. 清理示例与测试：从 examples/generate/multimodal/vision_language_offline.py 中删除 run_minimax_vl_01 函数；移除 tests/models/multimodal/processing/test_minimax_vl_01.py、tests/tool_parsers/test_minimax_tool_parser.py 等测试文件；在 tests/models/multimodal/generation/test_common.py、tests/models/multimodal/generation/vlm_utils/model_utils.py 中删除相关的测试条目。
5. 修复残留引用：更新 docs/contributing/model/basic.md 中过时的内部链接，修正 test_attention_backends_selection.py 中错误导入 MiniMaxText01LinearAttention 的位置。

关键文件：

- vllm/model_executor/models/minimax_text_01.py（模块 模型推理；类别 source；类型 deletion；符号 replace_weight_name, weight_loader_with_alias, wrapper, inner_func）：删除全部 1000 行代码，是本次 PR 的核心删除文件。包含旧版 Text-01 模型的所有实现（MLP、MoE、LinearAttention 等）。
- vllm/model_executor/models/minimax_vl_01.py（模块 模型推理；类别 source；类型 deletion；符号 MiniMaxVL01ImagePixelInputs, MiniMaxVL01ImageEmbeddingInputs, MiniMaxVL01MultiModalProjector, init）：删除 VL-01 视觉语言模型定义（385 行），包含多模态投影器、输入类型定义和处理器类。
- vllm/tool_parsers/minimax_tool_parser.py（模块 工具解析器；类别 source；类型 deletion；符号 MinimaxToolParser, init, preprocess_model_output, remove_tool_calls_from_think）：应审查者要求额外删除的 MiniMax M1 工具解析器（852 行），包含流式状态追踪、thinking tag 处理、JSON 清理等逻辑。
- tests/tool_parsers/test_minimax_tool_parser.py（模块 测试；类别 test；类型 deletion；符号 minimax_tokenizer, minimax_tool_parser, sample_tools, assert_tool_calls）：删除 MiniMax 工具解析器的配套测试（1227 行），包含 parametrize 测试用例。
- vllm/model_executor/models/registry.py（模块 模型注册表；类别 source；类型 data-contract）：修改模型注册表：移除旧架构的字典条目，新增到 _PREVIOUSLY_SUPPORTED_MODELS（+4/-7）。
- examples/generate/multimodal/vision_language_offline.py（模块 示例；类别 source；类型 core-logic；符号 run_minimax_vl_01）：删除 run_minimax_vl_01 函数及其在路由字典中的注册。
- tests/models/multimodal/processing/test_minimax_vl_01.py（模块 测试；类别 test；类型 deletion；符号 test_processor_override, _validate_image_prompt_replacements_one, _test_image_prompt_replacements, test_processor_prompt_replacements_regression）：删除 VL-01 的多模态处理器测试文件（113 行）。

关键符号：replace_weight_name, weight_loader_with_alias, wrapper, inner_func, MiniMaxText01MLP.init, MiniMaxText01MLP.forward, MiniMaxText01MoE.init, MiniMaxVL01MultiModalProjector.init, MiniMaxVL01MultiModalProjector.forward, MinimaxToolParser.extract_tool_calls, MinimaxToolParser.preprocess_model_output, run_minimax_vl_01

关键源码片段

vllm/model_executor/models/minimax_text_01.py

删除全部 1000 行代码，是本次 PR 的核心删除文件。包含旧版 Text-01 模型的所有实现（MLP、MoE、LinearAttention 等）。

```
# This is a representative snippet from the deleted file (base version).
# The class shows standard vLLM MLP pattern with parallel linear layers.

class MiniMaxText01MLP(nn.Module):
    """Standard MLP used by Text-01. 采用 MergedColumnParallelLinear + RowParallelLinear
    结构"""
```

```
    def __init__(
        self,
        hidden_size: int,
        intermediate_size: int,
        quant_config: QuantizationConfig | None = None,
        layer_idx: int = None,
        prefix: str = "mlp",
    ) -> None:
        super().__init__()
        self.layer_idx = layer_idx
        # 合并门控和上投影：两个输出通道
        self.gate_up_proj = MergedColumnParallelLinear(
            hidden_size,
            [intermediate_size] * 2,
            bias=False,
            quant_config=quant_config,
            prefix=f"{prefix}.gate_up_proj",
        )
        self.down_proj = RowParallelLinear(
            intermediate_size,
            hidden_size,
            bias=False,
            quant_config=quant_config,
            prefix=f"{prefix}.down_proj",
        )
        self.act_fn = SiluAndMul()

    def forward(self, x: torch.Tensor) -> torch.Tensor:
        gate_up, _ = self.gate_up_proj(x)
        x = self.act_fn(gate_up)
        x, _ = self.down_proj(x)
        return x
```

vllm/model_executor/models/minimax_vl_01.py

删除 VL-01 视觉语言模型定义（385 行），包含多模态投影器、输入类型定义和处理器类。

```
# 多模态投影器：将视觉特征映射到文本空间
class MiniMaxVL01MultiModalProjector(nn.Module):
    """两层 MLP 投影器，与 LLaVA-NeXT 结构类似"""
    def __init__(
        self,
        vision_hidden_size: int,
        text_hidden_size: int,
        projector_hidden_act: str,
        multimodal_projector_bias: bool,
        quant_config: QuantizationConfig | None = None,
        prefix: str = "",
    ):
        super().__init__()
```

```
        # 第一层：视觉隐层 -> 文本隐层（列并行）
        self.linear_1 = ColumnParallelLinear(
            vision_hidden_size,
            text_hidden_size,
            bias=multimodal_projector_bias,
            quant_config=quant_config,
            prefix=f"{prefix}.linear_1",
        )
        self.act = get_act_fn(projector_hidden_act)
        # 第二层：文本隐层 -> 文本隐层（行并行）
        self.linear_2 = RowParallelLinear(
            text_hidden_size,
            text_hidden_size,
            bias=multimodal_projector_bias,
            quant_config=quant_config,
            prefix=f"{prefix}.linear_2",
        )

    def forward(self, image_features: torch.Tensor) -> torch.Tensor:
        hidden_states, _ = self.linear_1(image_features)
        hidden_states = self.act(hidden_states)
        hidden_states, _ = self.linear_2(hidden_states)
        return hidden_states
```

vllm/tool_parsers/minimax_tool_parser.py

应审查者要求额外删除的 MiniMax M1 工具解析器（852 行），包含流式状态追踪、thinking tag 处理、JSON 清理等逻辑。

```
# MinimaxToolParser 核心构造逻辑
class MinimaxToolParser(ToolParser):
    """MiniMax M1 模型专用的工具解析器，处理 <tool_calls> 标签流"""
    def __init__(self, tokenizer: TokenizerLike, tools: list[Tool] | None = None):
        super().__init__(tokenizer, tools)
        # 流式状态：追踪当前工具进度
        self.streaming_state: dict[str, Any] = {
            "current_tool_index": -1,
            "tool_ids": [],
            "sent_tools": [],
        }
        # 工具调用的起始 / 结束标签
        self.tool_call_start_token = "<tool_calls>"
        self.tool_call_end_token = "</tool_calls>"
        self.tool_call_regex = re.compile(
            r"<tool_calls>(.*?)</tool_calls>|<tool_calls>(.*)", re.DOTALL
        )
        # 尝试从 tokenizer 中获取 token ID，若失败则回退到字符串匹配
        self.tool_call_start_token_id = self.vocab.get(self.tool_call_start_token)
        self.tool_call_end_token_id = self.vocab.get(self.tool_call_end_token)
        if self.tool_call_start_token_id is None or self.tool_call_end_token_id is None:
```

```
logger.warning(
    "Minimax Tool parser could not locate tool call start/end "
    "tokens in the tokenizer. Falling back to string matching."
)
```

评论区精华

审查者 sfeng33 要求额外移除 PR#20297 引入的 MiniMax M1 专用工具解析器，作者在第三个 commit 中落实。tdoublep 询问线性注意力 (minimax_linear_attn.py) 是否还被 M2 和 M3 使用，ZJY0516 确认仍被 Ling 2.5 模型使用，因此该共享模块未被删除。PR 触发了 pre-commit 和 merge 冲突的自动检查，均已解决。

- 删除 M1 工具解析器 (design): PR 作者在第三个 commit 中删除了 vllm/tool_parsers/minimax_tool_parser.py 及其相关测试和模板。
- 线性注意力是否仍被 M2/M3 使用 (question): ZJY0516 回应确认仍被 Ling 2.5 模型使用，因此共享模块被保留。

风险与影响

- 风险：主要风险在于遗漏对共享组件（如 minimax_linear_attn.py）的依赖检查，若其他模型依赖被删除的模块可能导致运行时错误。通过第二次 commit 修复了 test_attention_backends_selection.py 中的导入问题并更新了文档链接，降低了风险。工具解析器的删除不会影响 M2/M3 模型，因为它们的解析器是独立的。整体风险较低，因为所有被删除的架构均为已淘汰的旧版本。
- 影响：对用户的影响：使用 MiniMax M1 模型的用户无法再加载旧版检查点，需降级到 vLLM v0.23.0 或更早版本。对系统的影响：代码库大小减少约 3881 行，维护成本降低。对团队的影响：清理后模型注册表更清晰，减少了维护旧模型带来的混淆。没有对现有 M2/M3 模型的功能影响。
- 风险标记：核心模型删除，共享组件保留，工具解析器清理

关联脉络

- 暂无明显关联 PR