PR #41699 完整报告

vllm-project/vllm

[Model] Use AutoWeightsLoader for Plamo2

合并时间：2026-05-05 16:56

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/41699

执行摘要

- 一句话：Plamo2 权重加载迁移至 AutoWeightsLoader
- 推荐动作：该 PR 是 vLLM 权重加载标准化大图景中的一环，代码量中等但逻辑清晰。推荐开发者阅读，以理解 AutoWeightsLoader 的使用模式、从 *ForCausalLM 到 *Model 的迁移步骤，以及 tie_word_embeddings、旋转位置编码等特殊场景的处理方式。对于希望贡献类似迁移的贡献者，这是一个很好的参考范例。

功能与动机

推进 #15697 全模型加载标准化，让 Plamo2 的权重加载方式与其他语言模型（如 Qwen2、CohereMoe）保持一致，消除 Composite 模型中手动编写权重加载逻辑的需要，为后续多模态模型复用 Plamo2 backbone 铺平道路。

实现拆解

实现分为两步：

1. 将加载逻辑下移到 Plamo2Model：原 Plamo2ForCausalLM.load_weights 的主体被完整移至 Plamo2Model.load_weights，并修改三处细节：移除 tie_word_embeddings 分支、移除旋转编码跳过、将 model.norm.weight 子串检查改为 norm.weight 精确匹配。
2. 重新实现 Plamo2ForCausalLM.load_weights：通过 AutoWeightsLoader 递归加载子模块，传递 skip_prefixes=["lm_head."] 以处理 tie_word_embeddings，并返回 set[str] 满足契约。 配套验证：通过 py_compile、模型注册表加载测试和 lm-eval 回归（gsm8k）确保功能等价。

关键文件：

- vllm/model_executor/models/plamo2.py（模块 模型加载；类别 source；类型 refactor；符号 load_weights）：唯一变更文件。将权重加载逻辑从 Plamo2ForCausalLM 下移到 Plamo2Model，并让外层使用 AutoWeightsLoader 委托。包含名称映射、in_proj 重塑、RMSNorm 偏置等核心逻辑的迁移，以及三处调整以适配 AutoWeightsLoader 契约。

关键符号：Plamo2Model.load_weights, Plamo2ForCausalLM.load_weights

关键源码片段

vllm/model_executor/models/plamo2.py

唯一变更文件。将权重加载逻辑从 Plamo2ForCausalLM 下移到 Plamo2Model，并让外层使用 AutoWeightsLoader 委托。包含名称映射、in_proj 重塑、RMSNorm 偏置等核心逻辑的迁移，以及三处调整以适配 AutoWeightsLoader 契约。

```
# vllm/model_executor/models/plamo2.py
# Plamo2Model 新增的 load_weights 方法，接管了原 Plamo2ForCausalLM 中的加载逻辑
def load_weights(self, weights: Iterable[tuple[str, torch.Tensor]]) -> set[str]:
    params_dict = dict(self.named_parameters())
    loaded_params: set[str] = set()
    for name, loaded_weight in weights:
        # 1. 名称替换：将 HuggingFace 权重名映射为 vLLM 内部格式
        # 例如 .A_log -> .A, .B_norm_weight -> .B_norm.weight
        replacements = {
            ".A_log": ".A",
            ".B_norm_weight": ".B_norm.weight",
            ".C_norm_weight": ".C_norm.weight",
            ".dt_norm_weight": ".dt_norm.weight",
            ".q_weight": ".q_norm.weight",
            ".k_weight": ".k_norm.weight",
        }
        for old, new in replacements.items():
            if old in name:
                name = name.replace(old, new)

        # 2. 处理 mixer.in_proj 权重：需要转置 + 重塑 + 切分合并
        # 以适配 MergedColumnParallelLinear 的期望形状
        if (
            ".mixer.in_proj.weight" in name
            or "mixer.in_proj.qweight" in name
            or "mixer.in_proj.scales" in name
            or "mixer.in_proj.qzeros" in name
        ):
            if "mixer.in_proj.weight" in name:
                loaded_weight = loaded_weight.transpose(0, 1)
            # 重塑为 [hidden, num_heads, per_head*2] 再拆分为 gate 和 hidden
            loaded_weight = loaded_weight.reshape(
                loaded_weight.shape[0], self.config.mamba_num_heads, -1
            )
            gate_weight, hidden_states_weight = loaded_weight.chunk(2, dim=-1)
            gate_weight = gate_weight.reshape(loaded_weight.shape[0], -1)
            hidden_states_weight = hidden_states_weight.reshape(
                loaded_weight.shape[0], -1
            )
            loaded_weight = torch.cat([gate_weight, hidden_states_weight], dim=-1)
            if "mixer.in_proj.weight" in name:
                loaded_weight = loaded_weight.transpose(0, 1)

        # 3. RMSNorm 偏移：vLLM 的 RMSNorm 实现不自动加偏置
        if ".pre_mixer_norm" in name:
```

```
            loaded_weight += 1.0
        elif ".post_mixer_norm" in name:
            loaded_weight += 1.0 / 5
        elif ".pre_mlp_norm" in name:
            loaded_weight += 1.0
        elif ".post_mlp_norm" in name:
            loaded_weight += 1.0 / (5**1.5)
        elif name == "norm.weight": # 精确匹配，避免误匹配子模块 norm
            loaded_weight += 1.0

        # 4. 跳过 pipeline parallelism 中不属于当前设备的参数
        if is_pp_missing_parameter(name, self):
            continue

        # 5. 执行权重加载
        param = params_dict[name]
        weight_loader = getattr(param, "weight_loader", default_weight_loader)
        weight_loader(param, loaded_weight)
        loaded_params.add(name)

    return loaded_params # 返回已加载参数集合，满足 AutoWeightsLoader 契约
```

评论区精华

gemini-code-assist[bot] 提出了两条关于权重名称匹配健壮性的建议：使用 endswith 替代 in 进行子串检查。PR 作者已在 norm.weight 上采用精确相等，但保留了其他位置的 in 检查，理由是原始代码同样使用 in 且经过测试。最终 PR 被 DarkLight1337 批准合并，该讨论未引起修改。

- 权重名称匹配的鲁棒性 (design): PR 作者没有直接回应，但在 norm.weight 上已经使用了精确相等（==）。其他位置的 in 检查得以保留，理由是原始代码同样使用此模式且经过回归测试无问题。PR 最终被批准合并。

风险与影响

- 风险：风险极低。变更仅涉及权重加载路径的重构，所有现有逻辑均原样迁移。lm-eval 对比显示数值完全一致。不过，没有新增单元测试，仅依赖回归测试；如果未来 AutoWeightsLoader 的行为发生变化，可能影响兼容性。另外，in 子串匹配的细微风险依然存在（如可能的误匹配），但原始代码已有相同模式且未见问题。
- 影响：对用户无功能影响。对系统：Plamo2 的加载路径与其他模型统一，降低维护成本，为未来多模态模型中复用 Plamo2 backbone 提供基础。影响范围仅限于 Plamo2 模型及后续参考此模式迁移的开发者。
- 风险标记：低风险重构，缺少测试覆盖，权重匹配采用旧模式但已验证

关联脉络

- PR #41690 [Model] Use AutoWeightsLoader for CohereMoe: 同一系列重构，将 CohereMoe 也迁移至 AutoWeightsLoader，参考了类似模式。
- PR #15697 [Feature]: Composite model loading using AutoWeightsLoader for all models: 本 PR 是此 issue 的推进步骤之一，目标是所有模型标准化加载方式。