PR #6305 完整报告

verl-project/verl

[data] fix: forward apply_chat_template_kwargs to system prompt measurement

合并时间：2026-05-12 21:24

原文链接：http://prhub.com.cn/verl-project/verl/pull/6305

执行摘要

- 一句话：修复自定义 chat template 参数未转发导致 loss mask 偏移
- 推荐动作：建议所有使用 apply_chat_template_kwargs 的 SFT 用户升级此修复。值得关注的设计决策包括参数转发模式和 reviewer 提出的改进建议。可以在后续 PR 中参照 reviewer 建议，迁移到 apply_chat_template wrapper 和字典合并模式以增强健壮性。

功能与动机

What does this PR do? Fixes a bug where extract_system_prompt_and_generation and initialize_system_prompt in verl/utils/chat_template.py did not forward apply_chat_template_kwargs to their internal apply_chat_template() calls. This caused the system prompt length to be measured using default kwargs while per-turn tokenization used custom kwargs, resulting in incorrect token stripping and a shifted loss mask. Any SFT run using apply_chat_template_kwargs with a model whose chat template embeds configurable fields in the system prompt (e.g., model_identity on openai/gpt-oss-20b) would trigger the sanity_check assertion error. Bypassing it via ignore_input_ids_mismatch=True would result in training on silently malformed data (shifted loss mask, missing role markers).

实现拆解

1. 修改 chat template 核心函数：在 verl/utils/chat_template.py 中，为 initialize_system_prompt 和 extract_system_prompt_and_generation 添加 **apply_chat_template_kwargs 参数，并在三个内部的 tokenizer.apply_chat_template() 调用中透传这些参数，确保系统提示长度计算使用与每轮 tokenization 一致的 kwargs。
2. 调整数据集调用处：在 verl/utils/dataset/multiturn_sft_dataset.py 的 __init__ 方法中，将 self.apply_chat_template_kwargs 作为关键字参数传递给 extract_system_prompt_and_generation，保证初始化时系统提示长度的测量与后续数据处理（_process_single_message）对齐。

3. 新增测试用例确保回归：在 tests/utils/dataset/test_multiturn_sft_dataset_on_cpu.py 中新增 test_multiturn_sft_dataset_with_chat_template_kwargs 测试函数，使用 openai/gpt-oss-20b 模型并传入 model_identity 参数，验证 assistant 消息内容正确位于 loss mask 为 1 的区域，user 和 system 消息位于 loss mask 为 0 的区域。该测试在修复前因 sanity check 失败而无法通过。

关键文件：

- verl/utils/chat_template.py（模块 模板工具；类别 source；类型 core-logic；符号 initialize_system_prompt, extract_system_prompt_and_generation）：核心修复文件：修改 initialize_system_prompt 和 extract_system_prompt_and_generation 函数，添加 **apply_chat_template_kwargs 参数并传递给内部的 apply_chat_template 调用，解决系统提示长度测量与每轮 tokenization 参数不一致的问题。
- verl/utils/dataset/multiturn_sft_dataset.py（模块 数据集；类别 source；类型 core-logic）：调用处修改：将 self.apply_chat_template_kwargs 传递给 extract_system_prompt_and_generation，确保系统提示长度测量使用与 _process_single_message 一致的参数。
- tests/utils/dataset/test_multiturn_sft_dataset_on_cpu.py（模块 测试覆盖；类别 test；类型 test-coverage；符号 test_multiturn_sft_dataset_with_chat_template_kwargs）：新增测试用例 test_multiturn_sft_dataset_with_chat_template_kwargs，使用 openai/gpt-oss-20b 和自定义 model_identity 参数，验证 loss mask 正确覆盖 assistant 内容，确保修复有效。

关键符号：initialize_system_prompt, extract_system_prompt_and_generation, test_multiturn_sft_dataset_with_chat_template_kwargs

关键源码片段

verl/utils/chat_template.py

核心修复文件：修改 initialize_system_prompt 和 extract_system_prompt_and_generation 函数，添加 **apply_chat_template_kwargs 参数并传递给内部的 apply_chat_template 调用，解决系统提示长度测量与每轮 tokenization 参数不一致的问题。

```
# verl/utils/chat_template.py

def initialize_system_prompt(tokenizer, **apply_chat_template_kwargs) -> list[int]:
    """
    Initialize system prompt tokens for chat templates that support them.
    Args:
        tokenizer: The tokenizer with a chat template
        **apply_chat_template_kwargs: Additional arguments for apply_chat_template
    Returns:
        List of token IDs for the system prompt, or empty list if not supported
    """
    # 注意：将 **apply_chat_template_kwargs 传递给 apply_chat_template，
    # 保证系统提示长度测量与每轮 tokenization 使用相同的自定义参数
    token1 = normalize_token_ids(
```

```
        tokenizer.apply_chat_template(
            [{"role": "user", "content": ""}], add_generation_prompt=False, tokenize=True, **apply_
            chat_template_kwargs
        )
    )
    token2 = normalize_token_ids(
        tokenizer.apply_chat_template(
            [{"role": "user", "content": ""}] * 2,
            add_generation_prompt=False,
            tokenize=True,
            **apply_chat_template_kwargs,
        )
    )
    system_prompt = token1[: -(len(token2) - len(token1))]
    return system_prompt


def extract_system_prompt_and_generation(tokenizer, **apply_chat_template_kwargs):
    """Extract system prompt and generation prompt tokens."""
    # 三个内部调用都需要转发 kwargs，确保测量一致性
    token1 = normalize_token_ids(
        tokenizer.apply_chat_template(
            [{"role": "user", "content": ""}], add_generation_prompt=False, tokenize=True, **apply_
            chat_template_kwargs
        )
    )
    token2 = normalize_token_ids(
        tokenizer.apply_chat_template(
            [{"role": "user", "content": ""}] * 2,
            add_generation_prompt=False,
            tokenize=True,
            **apply_chat_template_kwargs,
        )
    )
    system_prompt = token1[: -(len(token2) - len(token1))]

    token3 = normalize_token_ids(
        tokenizer.apply_chat_template(
            [{"role": "user", "content": ""}], add_generation_prompt=True, tokenize=True, **apply_
            chat_template_kwargs
        )
    )
    generate_prompt = token3[len(token1):]

    return system_prompt, generate_prompt
```

评论区精华

Review 中 gemini-code-assist[bot] 提出了两点高优先级建议：

- 使用本地 apply_chat_template wrapper 和字典合并模式：建议用模块内定义的 apply_chat_template 函数代替直接调用 tokenizer.apply_chat_template，以保持一致性并支持需特殊处理的模型（如 Qwen3.5）。同时使用 **{**apply_chat_template_kwargs, 'add_generation_prompt': False, 'tokenize': True} 模式避免用户 kwargs 包含 tokenize 等参数时导致 TypeError。
- 在系统提示初始化时使用 processor 代替 tokenizer：建议在 MultiTurnSFTDataset.__init__ 中调用 extract_system_prompt_and_generation 时优先使用 self.processor（当可用时），以保证与 _process_single_message 中 tokenization 逻辑一致，尤其是对多模态模型。

这些建议未被当前 PR 采纳，但审核者 wuxibin89 批准了合并，认为这些增强可以在后续 PR 中完成。

- 使用本地 apply_chat_template wrapper 和字典合并模式避免参数冲突 (design)：当前 PR 未采纳，仍使用直接调用方式；wuxibin89 审批通过，认为可后续改进。
- 在系统提示初始化时使用 processor 代替 tokenizer (correctness)：当前 PR 未修改，仍使用 tokenizer；wuxibin89 审批通过。

风险与影响

- 风险：
  1. 参数冲突风险：当前实现直接将 **apply_chat_template_kwargs 与显式参数（如 tokenize=True）一起展开，如果用户配置的 kwargs 中包含这些键名，会引发 TypeError。建议采用字典合并模式规避。
  2. 多模态兼容性风险：未使用 processor 可能导致多模态模型（如 VLM）在 tokenization 时产生不一致。
  3. 回归风险：核心数据路径变更可能影响未使用自定义 kwargs 的现有流程，但已有测试覆盖基础场景。
  4. 安全风险：无直接安全风险。 - 影响：影响范围：所有在 SFT 训练中使用 apply_chat_template_kwargs 配置且模型系统提示包含可配置字段（如 model_identity）的用户。修复后训练数据的 loss mask 正确，不再触发断言错误或静默错误。对未使用自定义 kwargs 的用户无影响。影响程度中等，涉及数据正确性而非性能或可用性。无 API 破坏，配置向后兼容。 - 风险标记：参数冲突风险，多模态兼容性风险，核心数据路径变更

关联脉络

- 暂无明显关联 PR