PR #6933 完整报告

verl-project/verl

[megatron] feat: migrate fused logprob/entropy from GPTModel.forward monkey-patch to Megatron output_processor hook

合并时间：2026-08-08 15:12

原文链接：http://prhub.com.cn/verl-project/verl/pull/6933

执行摘要

- 一句话：fused forward 迁移至 Megatron output_processor 钩子
- 推荐动作：值得精读。核心设计是使用签名检查而非版本号做能力探测，实现优雅的自动回退；patch/unpatch 的幂等性处理避免了重复包装；CI 中加入防回退 grep 检测是确保新路径真实生效的务实做法。建议关注后续是否能随 Megatron-Core 全面升级移除 legacy 路径，以及 model_forward_1f1b_overlap.py 是否能同样迁移到 schedule-plan hook。

功能与动机

上游 issue NVIDIA/Megatron-LM#4590 指出：外部 RL 训练循环要使用 selected-token labels、自定义 fused logprob/loss 等，不得不复制或 monkey-patch GPTModel.forward、postprocess、MTP postprocess。verl 之前正是通过 model_forward_fused.py 里 _fused_GPTModel_forward 整个替换 forward 来把 temperature 传进 Megatron 的 _postprocess 边界。Megatron-Core 0.18 随 PR #4686 提供了原生 output_processor 钩子，本 PR 的目标就是改用该契约，消除对 GPTModel.forward 的替换，从而降低与上游 Megatron 升级的冲突成本。PR body 中也明确说明：native path 仅在模型暴露两个钩子参数时自动启用，旧模型自动回退 legacy patch，且无任何公开 API 变更。

实现拆解

1. 能力探测与模式选择：在 verl/models/mcore/model_forward_fused.py 中新增 _supports_output_processor_hook，通过 inspect.signature 检查绑定 GPT 模型的 forward 是否包含 output_processor 与 output_processor_context 参数，而非依赖版本号比较；_resolve_fused_forward_mode 据此返回 hook 或 legacy 模式。模式在模型构建时通过 patch_fused_forward 一次性解析并缓存到 _verl_fused_forward_mode 属性，运行期不再重复检查。
2. 回调实现：新增 FusedOutputProcessorContext dataclass 与 fused_output_processor 回调函数。回调在 Megatron 的 postprocess 边界接收 hidden_states、output_layer、output_weight、labels、context、config 等参数，按 config.sequence_parallel 先做序列并行 gather，再依据 tied/un tied 权重解析 output_weight 或 output_layer.weight，调用 linear_cross_entropy 计算 fused logprobs/entropy，并保留 config-logger 的 payload 记录。

3. 双路径 forward：fused_forward_model_gen 与 fused_forward_model_engine 两个工厂函数在构造 forward_kwargs 后，根据 _use_output_processor_hook(model) 选择路径：hook 路径移除 temperature 参数，改为传入 output_processor=fused_output_processor 与 output_processor_context=FusedOutputProcessorContext(temperature=temperature)；legacy 路径保持原有 temperature=temperature 传参。两条路径共享同一套 THD/FP8/CP 预处理逻辑。
4. 幂等 patch/unpatch：patch_fused_forward 与 unpatch_fused_forward 改为幂等实现。hook 模式下不创建 forward_backup、不替换 forward；legacy 模式下仅在不存在 forward_backup 时保存原 forward，unpatch 时删除 forward_backup。
5. 测试与 CI 配套：新增 tests/models/test_model_forward_fused.py 覆盖能力探测、native/legacy 幂等性、Megatron-Bridge 包装链（Float16Module → DDP → GPTModel）、temperature 传递、weight 解析、config-logger payload 等；.github/workflows/model.yml 将新测试加入 model CI；.github/workflows/e2e_ppo_trainer_megatron_vllm_2.yml 为 Megatron-Bridge GRPO E2E 设置 USE_FUSED_KERNELS=True 与 USE_REMOVE_PADDING=True，并通过 tee 捕获日志、grep 检测 Falling back to non-fused 来防止静默回退。

关键文件：

- verl/models/mcore/model_forward_fused.py（模块 模型前向；类别 source；类型 data-contract；符号 _supports_output_processor_hook, _resolve_fused_forward_mode, _get_fused_forward_mode, _use_output_processor_hook）：核心源码：将 fused logprob/entropy 从替换 GPTModel.forward 的 monkey-patch 迁移到 Megatron 原生 output_processor 钩子，新增能力探测、模式选择、回调实现与幂等 patch/unpatch。
- tests/models/test_model_forward_fused.py（模块 前向测试；类别 test；类型 test-coverage；符号 _new_uninitialized_model, test_mcore_gpt_forward_has_native_output_processor_contract, test_native_hook_selection_preserves_forward_and_is_idempotent, counted_signature）：新增 324 行针对性测试，覆盖 native/legacy 模式选择、幂等性、Megatron-Bridge 包装链、temperature 传递、weight 解析与 config-logger payload，是本 PR 正确性的关键保障。
- .github/workflows/e2e_ppo_trainer_megatron_vllm_2.yml（模块 CI 配置；类别 infra；类型 infrastructure）：E2E CI 启用 USE_FUSED_KERNELS=True 与 USE_REMOVE_PADDING=True，并加入日志 grep 防回退检测，保证新 hook 路径在真实训练中被实际执行。
- .github/workflows/model.yml（模块 CI 配置；类别 infra；类型 infrastructure）：将新增的 tests/models/test_model_forward_fused.py 加入 model CI，确保新的 hook 契约与回退逻辑在标准 CI 中持续验证。

关键符号：_supports_output_processor_hook, _resolve_fused_forward_mode, _get_fused_forward_mode, _use_output_processor_hook, fused_output_processor, FusedOutputProcessorContext, patch_fused_forward, unpatch_fused_forward, fused_forward_model_gen, fused_forward_model_engine

关键源码片段

verl/models/mcore/model_forward_fused.py

核心源码：将 fused logprob/entropy 从替换 GPTModel.forward 的 monkey-patch 迁移到 Megatron 原生 output_processor 钩子，新增能力探测、模式选择、回调实现与幂等 patch/unpatch。

```
# verl/models/mcore/model_forward_fused.py 核心片段
# 模式常量与模型属性名
_FUSED_FORWARD_MODE_ATTR = "_verl_fused_forward_mode"
_HOOK_MODE = "hook"
_LEGACY_MODE = "legacy"

# 通过签名检查判断绑定模型是否支持 Megatron 0.18 的 native hook 契约
# 用签名而非版本号判断，可兼容 backport 与被包装的实现
def _supports_output_processor_hook(patching_model: torch.nn.Module) -> bool:
    parameters = inspect.signature(patching_model.forward).parameters
    return {"output_processor", "output_processor_context"}.issubset(parameters)

def _resolve_fused_forward_mode(patching_model: torch.nn.Module) -> str:
    return _HOOK_MODE if _supports_output_processor_hook(patching_model) else _LEGACY_
    MODE

# 从模型或 language_model 上读取已解析的模式，缺省按 legacy 处理
def _get_fused_forward_mode(model: torch.nn.Module) -> str:
    model = unwrap_model(model)
    mode = getattr(model, _FUSED_FORWARD_MODE_ATTR, None)
    if mode is None and hasattr(model, "language_model"):
        mode = getattr(model.language_model, _FUSED_FORWARD_MODE_ATTR, None)
    return mode if mode in (_HOOK_MODE, _LEGACY_MODE) else _LEGACY_MODE

# 携带 temperature 的 hook 上下文
@dataclass
class FusedOutputProcessorContext:
    temperature: float

# 在 Megatron postprocess 边界执行 fused logprob/entropy 的回调
def fused_output_processor(
    *, hidden_states, output_layer, output_weight, labels, context, config, **_ignored,
):
    output = CausalLMOutputForPPO(
        loss=None, logits=None, past_key_values=None,
        hidden_states=hidden_states, attentions=None,
    )

    if config.sequence_parallel:
        hidden_states = gather_from_sequence_parallel_region(hidden_states)

    # tied 模型用共享 embedding 权重，untied 模型用 output_layer.weight
```

```
    weight = output_weight if output_weight is not None else output_layer.weight

    temperature = context.temperature
    logprobs, entropy = linear_cross_entropy(
        hidden_states, weight, labels, temperature, "none",
        parallel_state.get_tensor_model_parallel_group(),
    )

    if has_config_logger_enabled(config):
        payload = OrderedDict({
            "input_ids": _ignored.get("input_ids"),
            "position_ids": _ignored.get("position_ids"),
            "attention_mask": _ignored.get("attention_mask"),
            "decoder_input": _ignored.get("decoder_input"),
            "logprobs": logprobs,
            "entropy": entropy,
        })
        log_config_to_disk(config, payload, prefix="input_and_logits")

    output.entropy = entropy
    output.log_probs = logprobs
    return output

# patch 与 unpatch 均幂等：hook 模式完全不触碰 forward，legacy 模式只备份一次
def patch_fused_forward(model: torch.nn.Module):
    model = _get_patching_model(model)
    if model is None:
        return

    mode = getattr(model, _FUSED_FORWARD_MODE_ATTR, None)
    if mode is None:
        mode = _resolve_fused_forward_mode(model)
        setattr(model, _FUSED_FORWARD_MODE_ATTR, mode)

    if mode == _HOOK_MODE:
        return

    assert version.parse(mcore.__version__) >= version.parse("0.13.0")
    if not hasattr(model, "forward_backup"):
        model.forward_backup = model.forward
        model.forward = _fused_GPTModel_forward.__get__(model, model.__class__)

def unpatch_fused_forward(model: torch.nn.Module):
    model = _get_patching_model(model)
    if model is None or _get_fused_forward_mode(model) == _HOOK_MODE:
        return
    if hasattr(model, "forward_backup"):
        model.forward = model.forward_backup
        delattr(model, "forward_backup")
```

tests/models/test_model_forward_fused.py

新增 324 行针对性测试，覆盖 native/legacy 模式选择、幂等性、Megatron-Bridge 包装链、temperature 传递、weight 解析与 config-logger payload，是本 PR 正确性的关键保障。

```
# tests/models/test_model_forward_fused.py 关键测试
# 用 object.__new__ 避免触发 GPTModel 的重量级初始化，专注 forward 契约
def _new_uninitialized_model(model_cls=GPTModel):
    model = object.__new__(model_cls)
    torch.nn.Module.__init__(model)
    return model

# 验证当前绑定 Megatron 的 GPTModel.forward 已具备 native hook 契约
def test_mcore_gpt_forward_has_native_output_processor_contract():
    parameters = inspect.signature(GPTModel.forward).parameters
    assert {"output_processor", "output_processor_context"}.issubset(parameters)

# 验证 native 模式下 forward 身份不变、不创建 forward_backup，且签名检查只执行一次
def test_native_hook_selection_preserves_forward_and_is_idempotent(monkeypatch):
    model = _new_uninitialized_model()
    original_forward = model.forward.__func__
    signature_calls = 0
    original_signature = inspect.signature

    def counted_signature(callable_):
        nonlocal signature_calls
        signature_calls += 1
        return original_signature(callable_)

    monkeypatch.setattr(mff.inspect, "signature", counted_signature)

    mff.patch_fused_forward(model)
    mff.patch_fused_forward(model) # 重复 patch 应无副作用
    mff.unpatch_fused_forward(model)
    mff.unpatch_fused_forward(model)
    mff.patch_fused_forward(model)

    assert signature_calls == 1
    assert getattr(model, mff._FUSED_FORWARD_MODE_ATTR) == mff._HOOK_MODE
    assert model.forward.__func__ is original_forward
    assert not hasattr(model, "forward_backup")

# 验证 legacy 回退路径的幂等 patch/unpatch
class LegacyGPTModel(GPTModel):
    def forward(self, input_ids=None):
        return input_ids

def test_legacy_fallback_patch_and_unpatch_are_idempotent():
    model = _new_uninitialized_model(LegacyGPTModel)
```

```
original_forward = model.forward.__func__

mff.patch_fused_forward(model)
backup = model.forward_backup
mff.patch_fused_forward(model)

assert getattr(model, mff._FUSED_FORWARD_MODE_ATTR) == mff._LEGACY_MODE
assert model.forward.__func__ is mff._fused_GPTModel_forward
assert model.forward_backup is backup

mff.unpatch_fused_forward(model)
mff.unpatch_fused_forward(model)
assert model.forward.__func__ is original_forward
assert not hasattr(model, "forward_backup")
```

评论区精华

1. 冗余断言移除：gemini-code-assist[bot] 建议去掉对 weight 是否为 None 的多余断言，因为后续 linear_cross_entropy 调用在 None 时会自然报错。作者已移除，并用单测覆盖 tied / untied 两条权重路径。
2. 文档注明 mcore 版本要求：HollowMan6 指出应在 _supports_output_processor_hook 处说明仅支持 Megatron-Core 0.18.0+，并可在完全迁移后移除 legacy 路径。作者补充了 docstring 与 TODO，同时保留基于签名的能力探测而非版本比较，以兼容 backport。
3. CI 启用 fused kernels：HollowMan6 要求设置 USE_FUSED_KERNELS=True 以真正测试新路径，避免后续回归；作者在 E2E workflow 中为 Megatron-Bridge GRPO 标准用例开启该选项，同时保留 FP8 与 LoRA 用例的默认行为。
4. CI 回退问题修复：HollowMan6 指出首次 CI 运行因未设置 USE_REMOVE_PADDING=True 而回退到 non-fused，作者修改配置为同时设置 USE_FUSED_KERNELS=True 与 USE_REMOVE_PADDING=True，并新增日志 grep 检测，失败即报错，确保 CI 有效性。

- weight 为 None 的冗余断言 (style): 作者已移除该断言，并通过 tied / untied 两条输出权重路径的单元测试覆盖。
- 文档注明 mcore 0.18.0 支持与 legacy 移除计划 (documentation): 作者补充 docstring 与 TODO，并保留基于签名的能力探测以兼容 backport。
- CI 设置 USE_FUSED_KERNELS=True (testing): 作者在 Megatron-Bridge GRPO E2E 标准用例中设置 USE_FUSED_KERNELS=True，并保留 FP8 与 LoRA 用例的默认配置。
- CI 回退到 non-fused 的修复 (testing): 作者同时设置 USE_FUSED_KERNELS=True 与 USE_REMOVE_PADDING=True，并通过 tee/PIPESTATUS 与 grep 检测日志中是否出现 Falling back to non-fused，失败即报错。

风险与影响

- 风险：

1. 核心前向路径变更：model_forward_fused.py 是 Megatron 引擎 fused 前向的核心路径，hook 与 legacy 两条路径并存，任何对 THD/CP/FP8 预处理顺序的调整都可能影响训练正确性。PR 提供了逐位一致（max_abs=0）的 A/B 验证与 5 步 GSM8K 冒烟，但正式 CI 的 TP>=2 覆盖仍需依赖项目流水线确认。
2. 依赖 Megatron hook 契约：新路径依赖 Megatron-Core 0.18 的 output_processor / output_processor_context 契约，若上游后续调整回调签名或 context 传递方式，verl 需要跟进适配；目前通过签名探测自动回退旧版，但回退本身也可能掩盖契约漂移。
3. CI 防回退检测：E2E 步骤依赖 grep 检测 Falling back to non-fused 字符串，若 Megatron 或 verl 改变告警文案，CI 可能误报或漏报；同时 tee 与 PIPESTATUS 的组合在极端日志写入失败时可能掩盖真实训练退出码。
4. 遗留 legacy 路径双倍维护：在移除 legacy 之前，两套路径需持续保持行为一致，任何只改其一的功能（如新增模型参数）都可能造成隐性分叉。 - 影响：对使用 Megatron 引擎且开启 use_fused_kernels 的用户，前向计算路径从 monkey-patch 切换为原生 hook，GPTModel.forward 不再被替换，降低了与 Megatron 升级的冲突风险；行为上输出与 legacy 完全一致，无公开 API 或配置变更。对团队而言，该 PR 消除了对上游 forward 的侵入式修改，为后续移除 legacy patch 和跟进 Megatron 原生扩展点奠定了基础。CI 新增 fused kernels E2E 覆盖，可防止未来回归；测试文件新增 324 行，显著提升该模块的可验证性。 - 风险标记：核心前向路径变更，依赖 Megatron hook 契约稳定性，CI 防回退检测依赖日志文案，双路径需持续保持行为一致

关联脉络

- PR #7101 [docker] feat: upgrade vllm and megatron version, add packages to support DeepSeek-V4: 将 Megatron 升级到 core_v0.18.0，为本 PR 依赖的原生 output_processor 钩子提供了上游基础；PR 作者明确说明本 PR 是在 #7101 合并后 rebase 的。
- PR #7221 [megatron] feat: support contiguous context-parallel layout for DeepSeek V4: 同样修改了 verl/models/mcore/model_forward_fused.py，引入了 cp_layout 等参数处理，与本 PR 的 fused 前向路径同属 Megatron 模型前向演进，需保持兼容。