PR #6831 完整报告

verl-project/verl

[ci] chore: add some NPU's UT/ST

合并时间：2026-07-03 14:12

原文链接：http://prhub.com.cn/verl-project/verl/pull/6831

执行摘要

- 一句话：新增 NPU 单元测试 / 系统测试及 CI 配置
- 推荐动作：建议合并。该 PR 填补了 NPU 平台测试空白，是基础设施的重要完善。值得关注的设计决策是使用 get_device_name 工具函数统一设备抽象，该模式值得在后续测试和核心代码中推广。

功能与动机

PR body 指出目标为 'Use cases benchmarking GPUs，add some NPU's UT/ST'，即复用 GPU 已有的测试用例体系，为 NPU 平台补齐单元测试和系统测试，确保持续集成中 NPU 功能正确性。

实现拆解

1. 新增 veomni 测试脚本：在 tests/special_npu/nightly_ci_ascend/run_grpo_qwen3_30b_veomni_fsdp.sh 中添加 Qwen3-30B GRPO 训练测试脚本，配置 veomni 引擎 + FSDP，设置 NPU 特定环境变量（如 HCCL_OP_EXPANSION_MODE、VLLM_USE_V1 等），并支持 device 自动检测（gpu/npu）。
2. 修改 engine 测试以兼容 NPU：在 tests/models/test_engine.py 中导入 get_device_name、get_torch_device 等工具函数，将 torch.cuda.device_count() 替换为 get_torch_device().device_count()，将硬编码 'cuda' 替换为 device_name，使测试在 NPU 上也能正确执行。同时调整 assert_close 参数（check_dtype=False）以适应精度差异。
3. 更新 nightly CI 配置：在 nightly_ascend.yml 中新增 nightlyCI_grpo-qwen3-30b-veomni-fsdp_ascend job，配置 a3-16 容器，安装 veomni 依赖，运行新增的测试脚本。同时修改之前 job 的安装方式（从 pip install --no-deps -e . 改为 pip install -r requirements-npu.txt && pip install -v -e .）以符合 NPU 环境。
4. 增加 model 和 unit test CI：在 model_ascend.yml 中添加 model_engine_ascend step，运行 pytest tests/models/test_engine.py；在 npu_unit_tests.yml 中添加 checkpoint_engine 的 NPU 和 CPU 单元测试。
5. 其他 CI 调整：vllm_ascend.yml 中将 test_vllm_abort 替换为 test_bucketed_weight_transfer；e2e_ascend.yml 中调整安装命令。

关键文件：

- tests/special_npu/nightly_ci_ascend/run_grpo_qwen3_30b_veomni_fsdp.sh（模块 测试脚本；类别 test；类型 test-coverage）：新增 veomni+FSDP 的 GRPO 训练测试脚本，是本次 NPU 系统测试的核心内容，配置了 NPU 专属环境变量和 vLLM 参数。
- tests/models/test_engine.py（模块 测试引擎；类别 test；类型 test-coverage；符号 get_device_name, get_torch_device, test_actor_engine, test_critic_engine）：修改了 engine 测试用例，使其在 NPU 上也能运行。通过引入设备抽象工具替代硬编码 CUDA，是本次 UT 兼容改造的关键文件。
- .github/workflows/nightly_ascend.yml（模块 CI 配置；类别 infra；类型 infrastructure）：添加了运行 veomni 测试的 nightly CI job，并修改了安装方式以确保 NPU 环境正确配置，是 CI 配置的核心变更。
- .github/workflows/model_ascend.yml（模块 CI 配置；类别 infra；类型 infrastructure）：新增 model_engine_ascend 测试步骤，用于运行 engine 单元测试，覆盖 NPU 环境下的模型引擎验证。
- .github/workflows/npu_unit_tests.yml（模块 CI 配置；类别 infra；类型 infrastructure）：新增 checkpoint_engine 的 NPU 和 CPU 单元测试步骤，提升 checkpoint 相关代码的测试覆盖。

关键符号：get_device_name, get_torch_device, get_test_language_model, create_training_config, test_actor_engine, test_critic_engine

关键源码片段

tests/special_npu/nightly_ci_ascend/run_grpo_qwen3_30b_veomni_fsdp.sh

新增 veomni+FSDP 的 GRPO 训练测试脚本，是本次 NPU 系统测试的核心内容，配置了 NPU 专属环境变量和 vLLM 参数。

```
# 文件：tests/special_npu/nightly_ci_ascend/run_grpo_qwen3_30b_veomni_fsdp.sh
# 主要部分：设备检测与环境配置，然后启动 GRPO 训练

set -x
ENGINE=${1:-vllm}
DEVICE=${DEVICE:-$(python3 -c 'import torch_npu' 2>/dev/null && echo npu || echo gpu)}

MODEL_ID=${MODEL_ID:-Qwen/Qwen3-30B-A3B-Instruct-2507}
MODEL_PATH=${MODEL_PATH:-${HOME}/.cache/models/${MODEL_ID}}
TRAIN_FILE=$HOME/data/gsm8k/train.parquet
TEST_FILE=$HOME/data/gsm8k/test.parquet
max_prompt_length=$((1024 * 2))
max_response_length=$((1024 * 2))
rollout_max_num_seqs=$((128))
n_devices_per_node=$((8))
actor_ppo_max_token_len=$(((max_prompt_length + max_response_length) * 1))
infer_ppo_max_token_len=$(((max_prompt_length + max_response_length) * 3))

# 日志目录
SCRIPT_NAME="$(basename -- "${BASH_SOURCE[0]}" .sh)"
```

```
LOG_DIR=/root/.cache/nightly_log/$SCRIPT_NAME
mkdir -p $LOG_DIR

ROLLOUT_FILE=$LOG_DIR/$(date "+%Y%m%d_%H%M%S")

# NPU 特定环境变量
case "${DEVICE}" in
    npu)
        export TASK_QUEUE_ENABLE=1
        export HCCL_OP_EXPANSION_MODE="AIV"
        export VLLM_USE_V1=1
        export VLLM_VERSION=0.18.0
        export VLLM_ASCEND_ENABLE_NZ=0
        export HCCL_BUFFSIZE=610
        export CKPT_DIR="./c0kpt30b"
        export PYTORCH_NPU_ALLOC_CONF=max_split_size_mb:1024
        export CUDA_DEVICE_MAX_CONNECTIONS=1
        n_devices_per_node=16
        ;;
    *)
        echo "Unsupported DEVICE=${DEVICE}. Expected 'gpu' or 'npu'." >&2
        exit 1
        ;;
esac

# ... 后续配置 actor, rollout, ref, trainer 参数，并调用 verl.trainer.main_ppo
```

tests/models/test_engine.py

修改了 engine 测试用例，使其在 NPU 上也能运行。通过引入设备抽象工具替代硬编码 CUDA，是本次 UT 兼容改造的关键文件。

```
## 文件：tests/models/test_engine.py
# 关键改动：使用设备抽象工具替代硬编码 'cuda'

from verl.utils.device import get_device_name, get_nccl_backend, get_torch_device

# 获取当前设备名称（如 'cuda' 或 'npu'）
device_name = get_device_name()


def test_actor_engine(strategy):
    ray.init()
    # 替换 torch.cuda.device_count() -> get_torch_device().device_count()
    device_count = get_torch_device().device_count()
    config = create_training_config(
        model_type="language_model",
        strategy=strategy,
        device_count=device_count,
        model=get_test_language_model(device_count),
```

```
    )
    # ... 后续保持不变


def test_critic_engine(strategy):
    # 同样替换
    device_count = get_torch_device().device_count()
    # ...
    # 使用 device_name 替代硬编码 'cuda'
    with torch.device(device_name), torch.autocast(device_type=device_name, dtype=torch.
    bfloat16):
        hf_model = AutoModelForTokenClassification.from_pretrained(
            value_model_path, torch_dtype=torch.float32, attn_implementation="flash_attention_2"
        )
        hf_output = hf_model(
            input_ids.to(device=device_name),
            attention_mask=attention_mask.to(device=device_name)
        )
    # ...
    # 调整 assert_close 精度
    torch.testing.assert_close(hf_values_mean, mcore_values_mean, atol=1e-3, rtol=1e-2, check_
    dtype=False)
```

评论区精华

- wucong25在 test_engine.py review 中建议使用已有的 get_device_name 工具函数取代手动 if device_name == 'cuda' 分支，实现设备抽象统一。作者后续采纳并修改。
- wucong25还指出 from verl.utils.device import * 应改为按需导入，避免污染命名空间。作者同样修正为明确导入 get_device_name, get_nccl_backend, get_torch_device。
- daikang6对 bot 自动评论中的代码路径错误进行反馈（无关实际变更）。
- 使用 get_device_name 统一设备判断 (design): 作者采纳，修改为导入并使用 get_device_name 和 get_torch_device。
- 按需导入避免 import * (style): 作者修正为具体导入三个函数。

风险与影响

- 风险：
  - 环境依赖风险：新增的 veomni 测试脚本依赖特定版本的 veomni（0.1.11）和 transformers（5.3.0），若上游包更新可能造成兼容性问题。
  - CI 稳定性风险：新增的 nightly CI job 运行时间较长（timeout 180min），可能因资源不足或环境不稳定频繁失败，增加维护负担。
  - GPU 回归风险：test_engine.py 的设备抽象修改可能引入 GPU 端的回归（如 get_torch_device().device_count() 行为是否一致），但风险较低（已在 GPU 上运行过）。

    - 测试覆盖不足：新增脚本只覆盖 Qwen3-30B veomni 一种配置，缺乏其他模型规模和后端的测试。
- 影响：
    - 用户影响：无直接用户影响，均为 CI 和测试改进。
    - 系统影响：显著提升 NPU 平台的自动化测试覆盖，包括 engine、checkpoint、训练流程等关键组件，有助于早期发现问题。
    - 团队影响：新增 CI job 会增加维护成本，但通过统一设备抽象降低了后续测试编写难度。
    - 风险标记：新增测试依赖特定包版本，CI job 超时风险，设备抽象改动可能影响 GPU 端

关联脉络

- PR #6900 [doc] refactor: update ascend quick_start, add quickstart scripts for 4 training-inference backend combinations: 同为 NPU CI 改进 PR，更新了快速开始脚本和 CI 配置，与本次 PR 的 CI 改动属同一方向。
- PR #6877 [ci] fix: Add more test cases in e2e_ppo_trainer_megatron_sglang_ascend.yml: 同样在 Ascend CI 中添加测试用例，是本次 PR 的前置或关联工作。