PR #27106 完整报告

sgl-project/sglang

Make UTs compatible for XPU

合并时间：2026-07-15 12:35

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/27106

执行摘要

- 一句话：使 UT 兼容 XPU 设备
- 推荐动作：该 PR 虽然改动量小，但体现了良好的设备抽象设计：将 cuda 硬编码替换为 get_device() 等通用接口。对于需要跨硬件平台运行的测试和工具代码，这是一个值得采用的模式。建议读者关注 sglang.srt.utils.get_device 和 get_default_distributed_backend 的使用。

功能与动机

PR 说明：This PR make necessary changes required to make UTs compatible to run on XPU. 当前许多测试直接使用 torch.cuda 或 device='cuda'，无法在非 CUDA 设备上执行。通过抽象设备获取方式，使测试可以运行在 XPU 等设备上。

实现拆解

1. 导入通用工具函数：在 4 个文件中新增 from sglang.srt.utils import get_device，提供当前可用设备的统一获取方式。
2. 替换硬编码设备字符串：将 device='cuda'、.cuda()、torch.cuda.memory_allocated() 等具体 CUDA 调用替换为 get_device()、.to(get_device())、torch.get_device_module().memory_allocated()。
3. 动态选择分布式后端：在 test_tensor_dump_forward_hook.py 中，用 get_default_distributed_backend(get_device()) 替代固定的 'nccl'，使后端与设备类型匹配。
4. 处理遗留硬编码：在 dump_comparator.py 中将 x.cuda() 改为 x.to(get_device())，确保 tensor 加载到正确的设备。 这些改动均保持 CUDA 环境下的行为不变，因为 get_device() 在 NVIDIA GPU 上仍返回 'cuda'，且 get_default_distributed_backend('cuda') 返回 'nccl'；同时为 XPU 等设备提供了正确的值。

关键文件：

- python/sglang/srt/debug_utils/dump_comparator.py（模块 调试器；类别 source；类型 dependency-wiring）：唯一的源码修改，将 _load_object 中的 x.cuda() 替换为 x.to(get_device())，引入设备通用化。

- test/registered/kernels/test_fused_topk_deepseek.py（模块 MoE TopK 测试；类别 test；类型 test-coverage）：测试文件，替换了所有 'cuda' 为 get_device()，使测试能在 XPU 上运行。
- test/registered/debug_utils/test_tensor_dump_forward_hook.py（模块 前向钩子测试；类别 test；类型 test-coverage）：测试文件，使用 get_device() 和 get_default_distributed_backend 替代硬编码 CUDA 和后端，是改动最典型的测试文件。
- test/registered/unit/mem_cache/test_radix_cache_unit.py（模块 Radix 缓存测试；类别 test；类型 test-coverage）：测试文件，将 torch.cuda.memory_allocated() 替换为 torch.get_device_module().memory_allocated()，并统一设备设置。

关键符号：_load_object, test_fused_topk_deepseek, test_model_forward_dump, test_memory_allocated

关键源码片段

test/registered/debug_utils/test_tensor_dump_forward_hook.py

测试文件，使用 get_device() 和 get_default_distributed_backend 替代硬编码 CUDA 和后端，是改动最典型的测试文件。

```
from sglang.srt.utils import add_prefix, get_device
from sglang.srt.distributed.parallel_state import get_default_distributed_backend

def test_model_forward_dump(tmp_path):
    set_global_server_args_for_scheduler(ServerArgs(model_path="dummy"))
    device = get_device() # 获取当前设备，支持 cuda, xpu 等
    backend = get_default_distributed_backend(device) # 动态选择分布式后端
    init_distributed_environment(
        backend=backend,
        world_size=1,
        rank=0,
        local_rank=0,
        distributed_init_method="tcp://127.0.0.1:2646",
    )
    initialize_model_parallel()
    model = MockCausalLM()
    model.apply(init_weights)
    model = model.to(device=device, dtype=torch.bfloat16) # 使用 .to() 替代 .cuda()
    dumper = register_forward_hook_for_model(
        model, tmp_path / "sglang_dump", [0], 0, 0, 0
    )
    dir_path = dumper.get_dump_dir()
    inp = torch.randn(4, TEST_HIDDEN_SIZE, dtype=torch.bfloat16) * 0.01
    result = model(inp.to(device)) # 输入也移至正确设备
    data = torch.load(f"{dir_path}/Pass00000.pt")
    assert "model.layernorm" in data
    assert "model.mlp.down_proj" in data
    assert torch.allclose(
        data["model.mlp.down_proj"], result.cpu(), rtol=1e-5, atol=1e-5
```

）

评论区精华

1. 后端选择函数的讨论：siju-samuel 建议使用已有的 get_default_distributed_backend（来自 parallel_state）替代 PR 中新增的 get_distributed_backend 函数，以避免重复。mingfeima 也指出新增函数是重复的。最终 PR 作者采纳了建议，在测试中直接使用 get_default_distributed_backend，并移除了新增函数。
2. XPU CI 注册：关于是否在测试文件中添加 register_xpu_ci，siju-samuel 提出疑问，作者回应暂不加入 CI，因为内部周运行已覆盖。
3. 代码风格：siju-samuel 建议使用 model = model.to(device).bfloat16() 更清晰，但最终代码为 model.to(device=device, dtype=torch.bfloat16)，保留了链式调用。

- 后端选择函数重复 (design): 作者采纳建议，移除新增函数，在测试中直接使用 get_default_distributed_backend(get_device())。
- XPU CI 注册 (testing): ANSHUMAN87 回应暂不需要，因内部周运行已覆盖这些测试用例。
- 代码风格建议 (style): 最终提交采用 model.to(device=device, dtype=torch.bfloat16)，与建议不完全一致但功能一致。

风险与影响

- 风险：变更范围小且逻辑简单，主要风险是： 1) 在 CUDA 环境下，get_device() 和 get_default_distributed_backend 行为与原硬编码一致，回归风险低。 2) 在 XPU 上，这些测试之前不可执行，现在可以运行，但可能发现新的失败，这是预期内的。 3) 对 dump_comparator.py 的工具性改动不影响核心调度路径。总体风险可控，无性能或安全影响。
  - 影响：用户：XPU 用户现在可以运行这些单元测试来验证调试工具、MoE topk 选择、radix cache 等模块的正确性。系统：测试基础设施更具可移植性。团队：Intel/XPU 贡献者可以更顺畅地进行本地测试，减少了在非 CUDA 环境中的适配工作。影响程度中等，只涉及 4 个文件。
  - 风险标记：设备硬编码替换，测试兼容性变更

关联脉络

- PR #28428 [Intel GPU] DeepSeek V4 12/N: use sgl-kernel implementation of silu_and_mul_clamp to run on XPU: 同为 Intel XPU 支持系列，本 PR 为相关测试提供兼容性保障。
- PR #28059 [Intel GPU] DeepSeek V4 11/N: support fp8_paged_mqa_logits_triton from sgl-kernel to run on XPU: 同为 XPU 支持的内核迁移 PR，测试兼容性对其验证至关重要。