PR #30654 完整报告

sgl-project/sglang

Support per-regex diff-threshold predicates in the tensor comparator

合并时间：2026-07-09 20:15

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/30654

执行摘要

- 一句话：为张量比较器添加 per-regex predicate 规则，替代单一浮点阈值
- 推荐动作：值得精读，特别是 threshold_dsl.py 中安全 eval 的设计模式（空 builtins、dummy 环境预校验）和 per-regex 规则解析的 fail-closed 策略，可作为类似工具 DSL 实现的参考。

功能与动机

RL 训练转储中包含接近零的张量（如 MoE 专家的梯度），其相对差异无意义但绝对差异可忽略；单一的 rel 阈值无法表达 'rel OR max_abs' 的 rescue 机制而不放松其他检查。

实现拆解

1. 新建 DSL 模块(threshold_dsl.py)：定义 DiffThresholdRule 冻结值对象，parse_diff_threshold_rules 解析 CLI 令牌（兼容旧浮点简写或 (regex, predicate) 对），resolve_predicate 实现 first-fullmatch-wins 解析，parse_predicate 编译并校验谓词表达式（限制命名空间为 rel、max_abs、mean_abs，执行时空 builtins），evaluate_predicate 安全评估编译后的代码。
2. 改造比较器核心(comparator.py)：compare_tensor_pair 和 compute_diff 函数将 diff_threshold: float 参数替换为 predicate: str，默认谓词 'rel <= 0.001' 保持旧行为。compute_diff 最终通过 evaluate_predicate 决定 passed，不再比较 rel_diff 与阈值。
3. CLI 入口改造(entrypoint.py)：--diff-threshold 参数改为 nargs='*'，接受单个浮点简写或 (regex, predicate) 对，通过 parse_diff_threshold_rules 转换为规则列表，传入后续比较流程。
4. 联动类型与格式化：DiffInfo.diff_threshold 改为 predicate 字符串，记录每个张量使用的判据；格式化器 (formatter.py) 使用 diff.passed 而非重算。
5. 测试全覆盖：新增 test_threshold_dsl.py 覆盖 DSL 解析、求值、边界条件；test_entrypoint.py 和 test_comparator.py 增加端到端 predicate 行为测试；test_formatter.py 和 test_output_types.py 补充自定义修饰。同时修复了直接脚本执行时的导入问题。

关键文件：

- python/sglang/srt/debug_utils/comparator/threshold_dsl.py（模块 比较器；类别 source；类型 dependency-wiring；符号 DiffThresholdRule, parse_diff_threshold_rules,

resolve_predicate, parse_predicate）：新增 DSL 模块，定义了规则数据结构、解析逻辑和谓词安全求值，是本次变更的核心。

- test/registered/debug_utils/comparator/test_threshold_dsl.py（模块 DSL 测试；类别 test；类型 test-coverage；符号 _ev, TestParsePredicate, test_valid_predicates_parse, test_invalid_predicates_raise）：DSL 模块的全面测试，覆盖解析、求值、边界条件、错误消息。
- test/registered/debug_utils/comparator/test_entrypoint.py（模块 入口测试；类别 test；类型 test-coverage；符号 TestDiffThresholdCliParsing, test_parse_args_collects_diff_threshold_tokens, TestDiffThresholdPredicateExitCode, _dump_near_zero_pair）：端到端测试，验证 CLI 参数解析和 predicate 对 exit code 的影响。
- test/registered/debug_utils/comparator/tensor_comparator/test_comparator.py（模块 比较器测试；类别 test；类型 test-coverage；符号 TestComputeDiffPredicate, _near_zero_pair, test_default_predicate, test_predicate_rescues_near_zero_via_max_abs）：比较器核心的 predicate 单元测试，验证 compute_diff 的 predicate 行为。
- python/sglang/srt/debug_utils/comparator/tensor_comparator/comparator.py（模块 比较器；类别 source；类型 dependency-wiring）：比较器核心逻辑修改：用 predicate 替换 diff_threshold 参数，并集成 evaluate_predicate。
- python/sglang/srt/debug_utils/comparator/entrypoint.py（模块 入口；类别 source；类型 dependency-wiring）：CLI 参数变更的关键入口，添加 predicate 解析和传递逻辑。
- python/sglang/srt/debug_utils/comparator/bundle_comparator.py（模块 比较器；类别 source；类型 dependency-wiring）：中间层比较器，传递 diff_threshold_rules 参数。

关键符号：parse_diff_threshold_rules, resolve_predicate, parse_predicate, evaluate_predicate, compare_tensor_pair, compute_diff, run, _compare_bundle_pairs, parse_args

关键源码片段

test/registered/debug_utils/comparator/test_threshold_dsl.py

DSL 模块的全面测试，覆盖解析、求值、边界条件、错误消息。

```
# 测试辅助：编译并评估表达式，返回 bool
def _ev(
    expr: str, *, rel: float = 0.0, max_abs: float = 0.0, mean_abs: float = 0.0
) -> bool:
    return evaluate_predicate(
        parse_predicate(expr), rel=rel, max_abs=max_abs, mean_abs=mean_abs
    )


class TestParsePredicate:
    @pytest.mark.parametrize(
        "expr",
        [
            "rel <= 0.0085",
```

```
            "rel < 1",
            "max_abs > 0",
            "mean_abs >= 1e-5",
            "rel <= 0.01 or max_abs <= 1e-4",
            "rel <= 0.01 and max_abs <= 1e-4",
            "(rel <= 0.01 and max_abs <= 1e-4) or mean_abs <= 1e-5",
            "0 <= rel < 1",
            "rel <= -0.0",
            "rel <= 0",
        ],
    )
    def test_valid_predicates_parse(self, expr: str) -> None:
        """所有支持的表达式形式应无错误地通过 parse_predicate。"""
        parse_predicate(expr)

    @pytest.mark.parametrize(
        "expr",
        [
            "abs(rel) < 1", # 函数调用不允许
            "rel.x < 1", # 属性访问不允许
            "foo < 1", # 未知变量
            "rel < 'x'", # 类型比较不允许
            "", # 空字符串
            "rel <", # 语法错误
        ],
    )
    def test_invalid_predicates_raise(self, expr: str) -> None:
        """非法表达式应抛出 ValueError。"""
        with pytest.raises(ValueError):
            parse_predicate(expr)

    def test_unknown_name_message_lists_allowed(self) -> None:
        """错误消息应列出所有允许的变量名。"""
        with pytest.raises(ValueError, match="rel.*max_abs.*mean_abs"):
            parse_predicate("foo < 1")

class TestEvaluatePredicate:
    def test_rel_only_true_and_false(self) -> None:
        """纯 rel 谓词仅使用 rel 值。"""
        assert _ev("rel <= 0.01", rel=0.005) is True
        assert _ev("rel <= 0.01", rel=0.02) is False

    def test_le_boundary_inclusive(self) -> None:
        """<= 包含边界值，< 不包含。"""
        assert _ev("rel <= 0.01", rel=0.01) is True
        assert _ev("rel < 0.01", rel=0.01) is False

    def test_or_short_circuit_semantics(self) -> None:
        """or 在左侧为真时短路返回 True（近零 rescue 模式）。"""
```

```
        assert _ev("rel <= 0.0085 or max_abs <= 1e-3", rel=2.0, max_abs=2e-5) is True
        assert _ev("rel <= 0.0085 or max_abs <= 1e-3", rel=2.0, max_abs=0.5) is False
```

test/registered/debug_utils/comparator/test_entrypoint.py

端到端测试，验证 CLI 参数解析和 predicate 对 exit code 的影响。

```
class TestDiffThresholdPredicateExitCode:
    """端到端：per-regex --diff-threshold predicate 驱动 per-tensor 判决和 exit code。"""

    @staticmethod
    def _dump_near_zero_pair(tmp_path: Path) -> tuple[Path, Path]:
        """创建符号翻转的近零张量对（rel_diff = 2.0, max_abs = 2e-5）。"""
        baseline_t = torch.tensor([[1e-5, -1e-5], [1e-5, -1e-5]])
        baseline = _create_rank_dump(tmp_path / "baseline", rank=0, name="g", tensor=baseline_t)
        target = _create_rank_dump(tmp_path / "target", rank=0, name="g", tensor=-baseline_t)
        return baseline, target

    def test_default_predicate_fails_near_zero(self, tmp_path, capsys) -> None:
        """默认 'rel <= X' predicate 使近零对失败（exit code 1）。"""
        baseline, target = self._dump_near_zero_pair(tmp_path)
        argv = _make_argv(baseline, target, diff_threshold=0.0085)
        records, exit_code = _run_and_parse(argv, capsys)
        tensors = [r for r in records if isinstance(r, ComparisonTensorRecord)]
        assert len(tensors) == 1
        assert tensors[0].diff is not None and tensors[0].diff.passed is False
        assert tensors[0].diff.predicate == "rel <= 0.0085"
        assert exit_code == 1

    def test_predicate_passes_near_zero(self, tmp_path, capsys) -> None:
        """'rel or max_abs' predicate 使近零对通过（exit code 0）。"""
        baseline, target = self._dump_near_zero_pair(tmp_path)
        argv = _make_argv(
            baseline, target,
            diff_thresholds=[(".*", "rel <= 0.0085 or max_abs <= 1e-4")],
        )
        records, exit_code = _run_and_parse(argv, capsys)
        tensors = [r for r in records if isinstance(r, ComparisonTensorRecord)]
        assert len(tensors) == 1
        assert tensors[0].diff.passed is True
        assert tensors[0].diff.predicate == "rel <= 0.0085 or max_abs <= 1e-4"
        assert exit_code == 0

    def test_predicate_does_not_rescue_real_magnitude_failure(self) -> None:
        """真正的大 diff 应同时 fail rel 和 max_abs 条件。"""
        # 张量 [1,1,1,1,1,1,1,1,1,2] -> max_abs=1, rel_diff ~0.043
        # 两个条件均不满足，应 fail
```

test/registered/debug_utils/comparator/tensor_comparator/test_comparator.py

比较器核心的 predicate 单元测试，验证 compute_diff 的 predicate 行为。

```
class TestComputeDiffPredicate:
    @staticmethod
    def _near_zero_pair() -> tuple[torch.Tensor, torch.Tensor]:
        """符号翻转的近零对：rel_diff = 2.0, max_abs/mean_abs = 2e-5。"""
        x = torch.tensor([1e-5, -1e-5, 1e-5, -1e-5])
        return x, -x

    def test_default_predicate(self) -> None:
        """无 predicate 时使用默认值 'rel <= 0.001'；近零对 fail 并记录 predicate 字符串。"""
        x, y = self._near_zero_pair()
        diff = compute_diff(x_baseline=x, x_target=y)
        assert diff.rel_diff == pytest.approx(2.0, abs=1e-4)
        assert diff.max_abs_diff == pytest.approx(2e-5, abs=1e-7)
        assert diff.predicate == "rel <= 0.001"
        assert diff.passed is False

    def test_predicate_rescues_near_zero_via_max_abs(self) -> None:
        """'rel or max_abs' predicate 使近零对通过（即使 rel 失败）。"""
        x, y = self._near_zero_pair()
        diff = compute_diff(
            x_baseline=x, x_target=y, predicate="rel <= 0.0085 or max_abs <= 1e-4"
        )
        assert diff.rel_diff > 1.0 # 相对项仍然失败
        assert diff.passed is True
        assert diff.predicate == "rel <= 0.0085 or max_abs <= 1e-4"

    def test_predicate_does_not_rescue_real_magnitude_diff(self) -> None:
        """真实幅度 diff 应使 'rel or max_abs' 两个条件均失败。"""
        x = torch.ones(10)
        y = x.clone()
        y[0] = 2.0 # max_abs_diff = 1.0, rel_diff ~0.043
        diff = compute_diff(
            x_baseline=x, x_target=y, predicate="rel <= 0.0085 or max_abs <= 1e-4"
        )
        assert diff.passed is False
```

评论区精华

该 PR 无 review 讨论，只有一个机器人评论提示 quota 限制。

- 暂无高价值评论线程

风险与影响

- 风险：
  1. 向后兼容风险：CLI --diff-threshold 从单个 float 改为 nargs='*'，但兼容旧用法（单个浮点简写解析为 'rel <= 值'）；默认值行为保持不变。但旧脚本传递 --diff-threshold

0.001 仍有效，不会 break。

2. 安全风险：*evaluate_predicate* 使用 *eval*，但限制了 __builtins__ 为空，只允许 rel、max_abs、mean_abs 三个变量，且谓词在解析时通过 dummy 环境验证，防止注入。风险低。
3. 性能风险：谓词编译结果通过 @lru_cache 缓存，重复评估开销小。但每次张量比较都需调用 *resolve_predicate* 进行正则匹配，正则数量增加可能影响性能。默认规则列表通常很小，影响可忽略。
4. 逻辑风险：*resolve_predicate* 在张量名不匹配任何模式时抛出 ValueError（fail-closed），新规则可能遗漏某些张量导致工具退出，但提示清晰要求添加 *catch-all* 规则。用户需注意规则顺序。
5. 测试覆盖风险：测试覆盖了 DSL 解析、评估边界、*end-to-end exit code*，但未覆盖大规模正则列表场景。 - 影响：用户影响：使用 --diff-threshold 的用户升级后无需修改命令（兼容旧用法）；新用户可利用 *per-regex* 规则精细控制不同张量的通过标准。DiffInfo 中 *predicate* 字段代替 diff_threshold，可能影响下游解析工具，但 JSON 序列化向后兼容（字段名改变）。系统影响：仅影响 *debug_utils.comparator* 子系统，无其他模块依赖。团队影响：降低了调试 RL 训练 dump 时的假阳性报警，提高诊断效率。

- 风险标记：安全 *eval* 但限制空 builtins, fail-closed 可能导致工具退出 , 正则匹配性能（小规则集可忽略）, diff_threshold 字段名变更影响下游解析

关联脉络

- PR #30656 *Cap diagnostic detail computation for failing tensors*: 同一比较器模块的性能优化，与本次 *predicate* 功能在同一代码路径。
- PR #30655 Fix *rel_diff being nan for bitwise-identical tensors*: 同一比较器的 bug 修复，与本次变更同属 *debug_utils* 的持续改进。
- PR #30657 Support *grad injection and step override in the dumper's model dump*: *dumper* 的新功能，与比较器配合构成 *debug_utils* 工具链。