PR #33282 完整报告

sgl-project/sglang

docs(diffusion): update skills for MiniMax-H3

合并时间：2026-08-03 12:44

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/33282

执行摘要

- 一句话：更新 diffusion skills，新增 MiniMax-H3 T2VA 基准 preset
- 推荐动作：值得精读的文件为 bench_diffusion_denoise.py 的 force_eager 与 artifact_dir 设计，以及 existing-fast-paths.md 对 H3 数值契约的约束记录；其余为文档知识固化，可 skim。建议后续为该 preset 增加轻量单测或 CI 断言，避免文档与实际命令漂移。

功能与动机

PR body 明确指出核心目标是 为 MiniMax-H3 补齐 source-tracked 基准 preset、把 H3 部署 / 拓扑约束、联合视频 / 音频正确性检查、profiler 归因和既有快速路径写成文档，供 agent 技能消费。MiniMax-H3 是刚合入的原生音视频生成模型（#33275），其 task/target 契约与常规 diffusion 模型不同，需要专门的技能与基准配置，避免后续优化在缺乏一致性守卫的情况下破坏 eager BF16/FP32 基准。

实现拆解

1. 新增 H3 基准 preset（脚本核心）：在 python/sglang/multimodal_gen/.claude/skills/sglang-diffusion-benchmark-profile/scripts/bench_diffusion_denoise.py 的 MODELS 中新增 minimax-h3-t2va，请求字段（task/conditions/target/flow_shift/audio_flow_shift/num_inference_steps）全部放入 config_overrides 以生成独立 JSON 配置，extra_args 固定 4 GPU、TP2 + Ulysses2 拓扑；同时新增 force_eager 标志，使 build_sglang_cmd 在全局默认开启 torch.compile 时也强制跳过 --enable-torch-compile。
2. 生成配置路径下沉：build_sglang_cmd 新增 artifact_dir 参数，run_benchmark_once 将 --output-dir 传入，使 generated_configs/ 落在用户指定的输出目录下，运行可自包含复现。
3. 基准 / 性能技能文档扩展：benchmark-and-profile.md 增加 H3 部署约束（Ulysses 非 Ring、禁 CFG parallel、保留 tiled video-VAE decode）、手工命令示例、serving 基准驱动与联合视频 / 音频校验指标（帧级 PSNR/SSIM、32 kHz 音频波形与 log-mel 指标）；sglang-diffusion-performance/SKILL.md 增加 H200/H100/B200 拓扑快速配方与 --model-variant fl2va/ref2va 说明。
4. 既有快速路径记录：existing-fast-paths.md 补记 H3 的 indexed_modulation.py、ulysses_qkv.py、usp_relayout.py、fused QK norm + RoPE 等既有内核及其数值契约，要求优化前先证明这些守卫为何未命中。

5. 新增 Native Task-Contract 风格与量化边界：sglang-diffusion-add-model/SKILL.md 新增第三种 pipeline 风格并以 MiniMax-H3 为参考实现，补充对应 checklist；testing-and-accuracy.md 强调联合模态输出校验；sglang-diffusion-modelopt-quant/SKILL.md 明确 H3 在线 FP8 与已验证 ModelOpt PTQ/export 分离，避免误入支持矩阵。
6. 验证配套：PR body 列出的验证包括 py_compile、preset 断言、--list-models、--validate-nightly-alignment、quick_validate.py 与 Markdown 链接校验；GPU 单测因本机 transformers 环境不兼容未运行，仓库内也没有新增测试文件。

关键文件：

- python/sglang/multimodal_gen/.claude/skills/sglang-diffusion-benchmark-profile/scripts/bench_diffusion_denoise.py（模块 基准脚本；类别 source；类型 core-logic）：新增 minimax-h3-t2va preset、force_eager 标志及 artifact_dir 参数，是本 PR 唯一源码改动，定义了 H3 可复用基准命令与 eager 一致性守卫。
- python/sglang/multimodal_gen/.claude/skills/sglang-diffusion-benchmark-profile/benchmark-and-profile.md（模块 基准文档；类别 docs；类型 documentation）：H3 部署 / 拓扑约束、手工命令、serving 基准与联合视频 / 音频校验指标的权威文档，是本次技能更新的主体。
- python/sglang/multimodal_gen/.claude/skills/sglang-diffusion-add-model/SKILL.md（模块 接入文档；类别 docs；类型 documentation）：新增 Third Pipeline Style（Native Task-Contract），以 MiniMax-H3 为参考，规范后续多模态强耦合模型的接入方式。
- python/sglang/multimodal_gen/.claude/skills/sglang-diffusion-performance/SKILL.md（模块 性能文档；类别 docs；类型 documentation）：给出 H3 在不同 GPU 拓扑下的推荐启动命令与限制说明，是性能优化的第一入口。
- python/sglang/multimodal_gen/.claude/skills/sglang-diffusion-benchmark-profile/existing-fast-paths.md（模块 快速路径；类别 docs；类型 documentation）：记录 H3 的 indexed modulation、packed Ulysses QKV、fused QK norm + RoPE 等既有快速路径与数值契约，防止重复造轮子。
- python/sglang/multimodal_gen/.claude/skills/sglang-diffusion-add-model/references/testing-and-accuracy.md（模块 测试文档；类别 docs；类型 documentation）：强调联合模态输出验证要求，明确 H3 大拓扑 smoke case 的定位与配对测试。
- python/sglang/multimodal_gen/.claude/skills/sglang-diffusion-modelopt-quant/SKILL.md（模块 量化文档；类别 docs；类型 documentation）：明确 H3 在线 FP8 与 ModelOpt PTQ 的分界，防止误将 H3 加入已验证量化矩阵。
- python/sglang/multimodal_gen/.claude/skills/sglang-diffusion-benchmark-profile/SKILL.md（模块 技能简介；类别 docs；类型 documentation）：技能入口文档同步提及 H3 preset 与快速路径，保证代理能快速导航到详情页。

关键符号：build_sglang_cmd, run_benchmark_once

关键源码片段

python/sglang/multimodal_gen/.claude/skills/sglang-diffusion-benchmark-profile/scripts/bench_diffusion_denoise.py

新增 minimax-h3-t2va preset、force_eager 标志及 artifact_dir 参数，是本 PR 唯一源码改动，定义了 H3 可复用基准命令与 eager 一致性守卫。

```
# MiniMax-H3 通过 target.duration_seconds 拥有时间画布，因此模型专属采样字段
# 走 --config 生成的 JSON，而不是通用的 --width/--height/--num-frames 标志。
"minimax-h3-t2va": {
    "path": "MiniMaxAI/MiniMax-H3",
    "prompt": "At night, while their owner sleeps in a bedroom, three cats march in loudly
              playing tiny brass instruments, then abruptly file out.",
    "seed": 1101,
    "config_overrides": {
        "task": "t2va",
        "conditions": [],
        "target": {
            "short_edge": 768,
            "aspect_ratio": "16:9",
            "duration_seconds": 5.0,
        },
        "audio_flow_shift": 3.0,
        "flow_shift": 12.0,
        "num_inference_steps": 50,
    },
    "extra_args": [
        "--model-variant=fl2va",
        "--num-gpus=4",
        "--tp-size=2",
        "--ulysses-degree=2",
        "--performance-mode=speed",
        "--enable-torch-compile=false",
    ],
    # H3 eager BF16/FP32 是一致性基准；当前 torch.compile 会改变数值输出，
    # 因此该 preset 永远不加全局 helper 默认的 --enable-torch-compile。
    "force_eager": True,
},


def build_sglang_cmd(
    model_key: str,
    perf_dump_path: Optional[str] = None,
    warmup: bool = True,
    torch_compile: bool = True,
    seed: int = 42,
    save_output: bool = True,
    artifact_dir: Optional[Path] = None,
) -> list[str]:
    """构造 sglang generate 命令；与 benchmark-and-profile.md 保持精确一致。"""
    cfg = MODELS[model_key]

    cmd = [
```

```
        "sglang", "generate", "--backend=sglang",
        f"--model-path={cfg['path']}", f"--prompt={cfg['prompt']}",
    ]

    # 有 preset 专属 seed 时优先使用，保证跨次运行可复现
    effective_seed = cfg.get("seed", seed)
    if effective_seed is not None:
        cmd.append(f"--seed={effective_seed}")

    if "negative_prompt" in cfg:
        cmd.append(f"--negative-prompt={cfg['negative_prompt']}")
    if "image_path" in cfg:
        cmd.append(f"--image-path={cfg['image_path']}")

    # 将模型专属请求字段写入 generated_configs 下的独立 JSON，再以 --config 传入；
    # artifact_dir 存在时配置落盘到用户指定输出目录，否则使用默认 benchmark 目录。
    if "config_overrides" in cfg:
        config_root = (
            Path(artifact_dir)
            if artifact_dir is not None
            else get_output_dir("benchmarks", REPO_ROOT)
        )
        config_dir = ensure_dir(config_root / "generated_configs")
        config_path = config_dir / f"{model_key}.json"
        with open(config_path, "w") as f:
            json.dump(cfg["config_overrides"], f, indent=2, sort_keys=True)
        cmd.append(f"--config={config_path}")

    cmd.extend(cfg["extra_args"])

    # force_eager 优先于调用方全局 torch_compile 参数：H3 只能在 eager 模式下
    # 作为一致性基准，compile 会改变数值结果，此处显式禁止追加该标志。
    if save_output:
        cmd.append("--save-output")
    if warmup:
        cmd.append("--warmup")
    if torch_compile and not cfg.get("force_eager", False):
        cmd.append("--enable-torch-compile")
    if perf_dump_path:
        cmd.extend(["--perf-dump-path", perf_dump_path])

    return cmd
```

评论区精华

该 PR 无实际 review 评论与讨论线程：评论区仅有 gemini-code-assist[bot] 的停用提示（两次环境残留）和作者 BBuf 的 /tag-and-rerun-ci 触发指令。CI 状态（PR Test 与 Extra 均为失败徽章）缺少人工讨论记录，具体失败原因需结合 CI 日志确认。

- 暂无高价值评论线程

风险与影响

- 风险：
  1. 文档与代码漂移风险：bench_diffusion_denoise.py 中硬编码的 H3 config_overrides 与未来 H3 API/ 字段演进可能脱节，且未纳入自动化测试（PR body 承认 GPU 单测未跑，CI 也显示失败），新增分支 artifact_dir 与 force_eager 缺少直接单测覆盖。
  2. force_eager 全局语义：build_sglang_cmd 中 torch_compile and not cfg.get("force_eager", False) 会覆盖调用方显式传入的 torch_compile=True，若未来出现需要强制 compile 的模型而无对应对冲逻辑，可能被误伤；目前仅影响 H3 预设，运行时行为零影响。
  3. 配置路径变更：artifact_dir 使生成的 H3 配置文件移入 --output-dir/generated_configs/，依赖旧默认位置（get_output_dir("benchmarks", ...)）的自动化脚本需同步调整。 - 影响：影响范围集中在 diffusion 开发工具链与 agent 技能库：开发者在基准、性能优化、模型接入与量化流程中会读取这些 skill 文档，H3 基准可一键复现并强制 eager 一致性模式；对线上服务与推理运行时无任何代码路径影响。团队层面降低了 H3 后续优化破坏 eager 数值契约的概率，属于中等偏低影响、长期正向维护价值。 - 风险标记：文档与代码漂移风险，新增分支缺少测试覆盖，force_eager 全局逻辑扩展性，CI 测试显示失败

关联脉络

- PR #33275 [diffusion] model: support minimax-h3: 本 PR 的技能文档与基准 preset 完全围绕 #33275 合入的 MiniMax-H3 原生音视频管线编写，属于同一功能线的后续知识固化。
- PR #33251 docs: drop unreachable inkling LoRA benchmark entry: 同属 diffusion/skill 文档维护系列，体现持续清理与对齐文档的趋势。