# 项目开发复盘笔记（个人回顾用）

**项目**：Qwen3-4B 微调实验
**时间**：2025年12月
**用途**：个人复盘，记录思考过程和问题解决经验

---

# 一、拿到需求后的分析过程

## 1.1 需求理解

拿到 Project Spec 后，首先提取关键信息：

```
核心任务：
1. 对 Qwen3-4B 进行 SFT 微调
2. 在 SFT 基础上进行 DPO/RL 训练
3. 在 9 个 benchmark 上评估
4. 24 小时训练时间限制
5. 硬件：4×H100 GPU
```

## 1.2 技术方案确定

**问题**：4B 模型 + 4×H100，直接全量微调可行吗?

**分析**：

- Qwen3-4B 参数量：~4B × 4 bytes (fp32) = 16GB
- 训练时需要梯度 + 优化器状态 = 3-4x 参数量 = 48-64GB
- 4×H100 (80GB each) = 320GB，理论可行

**但考虑到**：

- 想在 24 小时内完成两阶段训练
- 需要留余量给 DPO（需要同时加载 policy 和 reference 模型）

**最终决策**：使用 **QLoRA**（4-bit 量化 + LoRA）

- 显存占用降低到 ~10GB
- 训练速度更快
- 效果接近全量微调

## 1.3 数据选择

**SFT 数据**：

- 考虑过：ShareGPT、OpenOrca、Alpaca
- 选择 Alpaca：规模适中（52K），格式规范，社区验证

**DPO 数据**：

- 考虑过：OpenAssistant、Anthropic HH-RLHF
- 选择 UltraFeedback：偏好数据质量高，与 Qwen 系列兼容

---

# 二、实现过程中的关键决策

## 2.1 训练配置选择

**学习率**：

- 初始较大 (2e-4)，让 LoRA adapter 快速适应
- DPO 阶段降低 (5e-5)，避免过拟合

**Batch Size 策略**：

```
实际 batch = per_device_batch × gradient_accumulation × num_gpus
           = 4 × 8 × 4 = 128
```

选择有效批次 128 的原因：

- 太小 (32)：训练不稳定
- 太大 (256)：显存压力大，且对小数据集可能过拟合

## 2.2 注意力实现选择

**遇到的问题**：Flash Attention 2 安装失败

**分析**：

- H100 支持 FA2，但编译依赖复杂
- 服务器环境可能缺少某些编译工具

**解决方案**：改用 `sdpa`（Scaled Dot-Product Attention）

- PyTorch 2.0+ 内置
- 性能接近 FA2
- 零安装成本

```
attn_implementation: "sdpa"  # 而不是 "flash_attention_2"
```

## 2.3 分布式训练配置

**遇到的问题**：DDP + LoRA + Gradient Checkpointing 冲突

**症状**：

```
RuntimeError: Expected to mark a variable ready only once.
This error is caused by one of the following reasons...
```

**分析**：

- Gradient checkpointing 会重新计算前向传播
- LoRA adapter 的梯度可能被标记多次
- DDP 的 bucket 机制与此冲突

**解决方案**：禁用 gradient checkpointing

```
gradient_checkpointing: false
```

虽然会增加显存占用，但 H100 80GB 够用。

---

# 三、遇到的问题及解决思路

## 3.1 数据加载问题

**问题**：本地数据格式识别失败

**调试过程**：

1. 检查目录结构 → 发现有 `dataset_dict.json`
2. 检查代码逻辑 → 发现只处理了 `dataset_info.json` 格式
3. 添加多格式检测逻辑

**修复**：

```
if os.path.exists(os.path.join(local_path, "dataset_dict.json")):
    dataset = load_from_disk(local_path)
elif os.path.exists(os.path.join(local_path, "train")):
    dataset = load_from_disk(local_path)
# ... 更多格式
```

## 3.2 DPO 训练 OOM

**问题**：

```
CUDA out of memory. Tried to allocate 2.00 GiB
```

**分析**：

- DPO 需要同时加载 policy 和 reference 模型
- 显存是普通训练的 2 倍

**解决方案**：

```
per_device_train_batch_size: 2 → 1
gradient_accumulation_steps: 16 → 32
```

保持有效批次不变（128），但单步显存减半。

## 3.3 评估阶段的一系列问题

### 问题 1：PermissionError 锁文件

```
PermissionError: Permission denied: 'xxx.lock'
```

**原因**：HF datasets 在共享缓存目录创建锁文件，没有写权限

**解决**：将 HF_HOME 改为用户可写目录

```
export HF_HOME=/data/250010131/cache/huggingface
```

### 问题 2：MMLU 配置不兼容

```
ValueError: Couldn't find cache for cais/mmlu for config 'world_religions'
```

**原因**：lm-eval 需要各学科单独配置，但缓存只有 'all'

**解决**：从评估任务中移除 mmlu

### 问题 3：SFT/DPO 模型无法加载

```
ValueError: Unrecognized model. Should have `model_type` key
```

**原因**：保存的是 LoRA adapter，不是完整模型

**解决**：评估前先合并 adapter

```
python scripts/merge_lora.py --model outputs/sft/final --output
outputs/sft/merged
```

### 问题 4：多卡并行评估锁冲突

```
PermissionError: ... builder.lock
```

**原因**：多个进程同时访问同一缓存目录

**最终方案**：放弃多卡并行，改用单卡顺序评估

- 虽然慢一些，但稳定可靠
- H100 单卡性能足够

---

# 四、经验教训总结

## 4.1 技术选型原则

1. **优先选择成熟稳定的方案**
   - Flash Attention → SDPA（更稳定）
   - 多卡并行 → 单卡顺序（更可控）
2. **显存管理要保守**
   - 始终预留 20-30% 余量
   - 遇到 OOM 时优先减小 batch size
3. **分布式训练的坑很多**
   - DDP + LoRA + GradientCheckpointing 组合要小心
   - 先单卡调通，再尝试多卡

## 4.2 调试技巧

1. **先看完整错误栈**
   - 不要只看最后一行
   - 关注 "The above exception was the direct cause of..."
2. **逐步排查**
   - 遇到问题时，先用最简配置跑通
   - 然后逐个添加功能，定位问题点
3. **善用日志**
   - `print` 关键变量
   - 检查张量形状和设备

## 4.3 项目管理

1. **时间分配**
   - 环境配置：2-3 小时
   - SFT 训练：8-10 小时
   - DPO 训练：10-12 小时
   - 评估分析：2-4 小时
2. **风险管理**
   - 先跑小规模实验验证可行性
   - 重要训练前保存 checkpoint
   - 评估时先用简单 benchmark 测试

---

# 五、代码/脚本备忘

## 5.1 常用命令

```
# 激活环境
source activate /data/250010131/conda_envs/mlsys_project

# SFT 训练
bash scripts/run_sft_train.sh

# DPO 训练
bash scripts/run_dpo_train.sh

# 评估
bash scripts/run_full_evaluation.sh

# 结果分析
python scripts/analyze_results.py
python scripts/visualize_results.py
```

## 5.2 关键配置文件

| 文件 | 用途 |
| --- | --- |
| `configs/sft_config.yaml` | SFT 训练配置 |
| `configs/dpo_config.yaml` | DPO 训练配置 |
| `configs/ds_config.json` | DeepSpeed 配置 |

## 5.3 输出目录

| 目录 | 内容 |
| --- | --- |
| `outputs/sft/final/` | SFT LoRA adapter |
| `outputs/sft/merged/` | SFT 合并模型 |
| `outputs/dpo/final/` | DPO LoRA adapter |
| `outputs/dpo/merged/` | DPO 合并模型 |
| `eval_results/` | 评估结果 |
| `report/` | 报告和图表 |

---

# 六、下次可以改进的地方

1. **数据方面**
   - 尝试更大规模数据（ShareGPT 100K+）
   - 数据清洗和去重
2. **训练方面**
   - 尝试更大的 LoRA rank (128)
   - 探索学习率调度策略
   - 尝试 warmup ratio 调优
3. **评估方面**
   - 提前准备好所有 benchmark 缓存
   - 使用自定义评估脚本避免 lm-eval 的各种问题
4. **工程方面**
   - 更完善的错误处理和日志
   - 训练过程可视化 (TensorBoard)
   - 自动化脚本减少手动操作