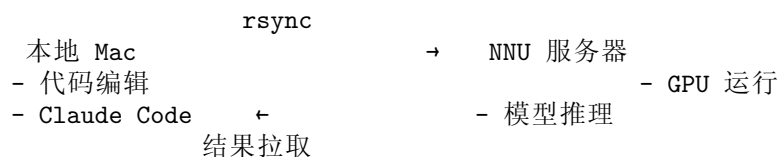


mRAG 项目架构

系统架构



项目结构

```
mRAG/
  main.py          # 测试入口
  Makefile         # 自动化工具
  .exclude        # rsync 排除规则
  .alias          # shell 别名
  requirements.txt # Python 依赖
  environment.yml  # Conda 环境

github/           # 子项目集合
  MRAG-Bench/     # UCLA 基准测试
    eval/
      models/     # 模型评估脚本
        run_model.sh
        llava_one_vision.py
      score.py    # 评分脚本
      utils/     # 工具函数

  LLaVA-NeXT/     # 视觉-语言模型
    llava/
      model/      # 模型实现
      eval/       # 评估工具

  magiclens/      # DeepMind 视觉模型
    model.py
    inference.py
    data/         # 大数据文件（远程专用）
```

数据流

```
HuggingFace (hf-mirror.com)
↓
下载模型/数据集
↓
远程服务器缓存 (/home/user/.cache/huggingface/)
```

↓
MRAG-Bench 评估脚本
↓
LLaVA One Vision 推理
↓
生成结果 JSONL
↓
评分脚本 (score.py)
↓
输出准确率报告

关键组件

1. Makefile 工具链

命令	功能	别名
<code>make sync</code>	本地 → 服务器同步	<code>ms</code>
<code>make cmd</code>	远程命令执行	<code>mc</code>
<code>make config</code>	SSH 免密 + 别名配置	-
<code>make alias</code>	输出别名定义	-

2. 模型推理流程

github/MRAG-Bench/eval/models/llava_one_vision.py

1. 加载 LLaVA One Vision 模型

- `attn_implementation="sdpa"` (避免 flash-attn)

2. 处理 MRAG-Bench 数据集

- 图像 + 文本查询

3. 生成答案

- 多模态理解 + 检索

4. 保存结果

- JSONL 格式输出

3. 环境配置

本地 (Mac): - 代码编辑 - Git 管理 - rsync 同步

远程 (NNU): - Conda 环境: `llava` (Python 3.10) - CUDA 12.1 + PyTorch

2.1.2 - GPU: `CUDA_VISIBLE_DEVICES=0`

网络策略

- HuggingFace: 使用 `hf-mirror.com` 镜像 (国内访问)

- GitHub: 通过 `/etc/hosts` 配置 `140.82.112.4 github.com`
- VPN: Clash 配置 (暂时不可用)

同步策略

rsync 规则 (.exclude)

保护远程大文件

github/magiclens/data

github/magiclens/data/**

排除本地文件

.git/

__pycache__/

*.pyc

.DS_Store

同步命令

```
rsync -avz --delete --exclude-from=.exclude \
./ NNU:~/code/mRAG/
```

扩展点

1. 添加新模型: 在 `github/MRAG-Bench/eval/models/` 下创建新脚本
2. 自定义评估: 修改 `eval/score.py`
3. 数据预处理: 使用 `eval/utils/dataloader.py`