

科研能力提升培训

作者：唐健凯 Jack Tang tjk24@mails.tsinghua.edu.cn
更新日期：2025年2月

0. 摘要

本文主要介绍了在人机交互和普适计算领域如何提升科研能力的相关教程，包括如何检索阅读文献、如何复现文章算法、如何调试硬件、如何进行用户实验、如何撰写论文、如何正确提问等。

1. 如何检索阅读文献

1.1 去哪检索文献

- 中国知网**：中国知网是一个综合性的学术文献数据库，包含了中国学术期刊、学位论文、会议论文等，适合初步了解某个领域的研究现状，如需深入研究，建议使用如下的国际学术数据库。
- Google Scholar**：谷歌学术是一个免费的学术搜索引擎，可以搜索出全球范围内的学术文献。还可以批量导出文献的Bibtex格式。
- IEEE Xplore**：IEEE Xplore是IEEE出版的学术文献数据库，包含了IEEE出版的所有期刊、会议论文和标准。
- ACM Digital Library**：ACM Digital Library是ACM出版的学术文献数据库，包含了ACM出版的所有期刊、会议论文和标准。
- Springer**：Springer是一个全球性的出版集团，提供了大量的学术文献。

1.2 如何检索文献

- 关键词检索**：在Google Scholar、IEEE Xplore、ACM Digital Library、Springer等数据库中，输入关键词，可以检索到相关的学术文献。
- 引用检索**：在Google Scholar中，可以通过检索某篇文章的引用，找到引用了这篇文章的其他文章。
- 高级检索**：可以通过高级检索，设置检索条件，精确检索到相关的学术文献。例如需要检索“传感器”和“人体健康监测”的相关文献，可以设置检索条件为“传感器 AND 人体健康监测”。还可以设置检索条件为“传感器 OR 人体健康监测”，检索到包含“传感器”或“人体健康监测”的相关文献。

1.3 如何筛选文献

- 优先选择**相关性高、引用量高**的文献，可以通过Google Scholar查看文献的引用量。
- 优先选择**顶级学术会议/期刊**的文章，具体可以参考SCI和CCF分区：人机交互与普适计算领域的顶会有**CHI、UIST、Ubicomp、NeurIPS、MobiCOM**等，顶刊有**IMWUT、TOCHI、IJHCS、Pervasive**等；生物医学领域的顶会有**MICCAI、EMBC、ISBI**等，顶刊有**IEEE TBME、IEEE TMI、IEEE JBHI**等。当然如果是在**Nature、Science、Cell**及其子刊等顶级期刊发表的文章，也是非常值得阅读的。
- 优先选择**近5年发表**的文献，以保证文献的新颖性和前沿性。
- 优先选择**开源代码**的文献，可以通过GitHub查看文献的代码是否开源。

1.4 如何阅读文献

阅读文献时，可以按照以下步骤进行：

- 泛读标题、摘要**：先阅读文献的标题和摘要，了解文献的主要内容。
- 略看引言、图表**：然后略看文献的引言、图表，了解文献的背景、实验结果和相较于前人的创新点。
- 深读方法、实验**：如果确认文章是和自己研究契合，接着深读文献的方法和实验部分，了解文献的具体实验设计和实验结果。
- 整理提炼**：读论文的目的通常不仅仅是了解单篇文献，而是服务后续的研究和论文写作，因此在阅读的过程中，往往需要整理提炼文献的关键信息和创新点。这里给了一个示例的阅读表格，经验上如果能阅读并填写30篇文献，就能对该领域有一个比较全面的了解。

题目	Bibtex	会议/期刊	链接	年份	*总结	*任务
Mmpd: multi-domain mobile	@inproceedings{tang2023mmpd, title={Mmpd: multi-domain mobile video physiology dataset},	EMBC 2023}	链接	2023	提出了一个33人的跨肤色跨环境rPPG数据集MMPD，并使用4种算法系统评估了数据集的可靠性	前置摄像头测心率

题目	Bibtex	会议/ 期刊	链接	年份	*总结	*任务
video physiology dataset	author={Tang, Jiankai and Chen, Kequan and Wang, Yuntao and Shi, Yuanchun and Patel, Shwetak and McDuff, Daniel and Liu, Xin}, booktitle={2023 45th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)}, pages={1--5}, year={2023}, organization={IEEE}					

2. 如何复现文章算法

2.1 什么样的算法值得复现

选择具备以下特征的算法优先复现：

- **代码完整度**：作者开源了核心代码和预训练模型,并且提供了环境配置文件和数据集下载链接。GitHub社区有一定的讨论和已回复的Issues。
- **高影响力**：被顶会/顶刊收录且引用量高（如CVPR/NeurIPS论文），在GitHub的Star数量多。

示例：复现rPPG-Toolbox时，作者提供了完整的代码、数据集和环境配置，且论文被NeurIPS收录，GitHub Star数超过500。

2.2 前期准备

2.2.1 服务器连接

- **命令行配置免密登录**：ssh-keygen -t rsa 生成密钥对，将公钥 id_rsa.pub 发送给管理员添加到服务器的 ~/.ssh/authorized_keys 文件
- **命令行连接**：ssh user@ip，无需输入密码应该连接成功。
- ****VSCode连接**：**安装Remote-SSH插件，输入 ssh user@ip 连接服务器。需要配置私钥文件路径： ~/.ssh/id_rsa 。VSCODE配置文件如下：

```
Host 10.1.1.1 (服务器ip)
  User root (服务器用户名)
  HostName 10.1.1.1 (服务器ip)
  IdentityFile ~/.ssh/id_rsa (私钥文件路径)
```

- **验证GPU状态**：nvidia-smi 查看显存占用，同时推荐安装 nvidia-smi 查看GPU使用情况。
- **创建个人文件夹**：mkdir -p /mnt/data/user（user为用户名，具体的路径请联系管理员，传统的路径为 /home/user，可能会有权限问题和存储空间问题）

示例：通过 ssh user@10.1.1.1（具体ip请联系管理员）连接实验室服务器

2.2.2 环境配置

- **使用Conda创建虚拟环境**：conda create -n repro python=3.8
- **安装基础依赖**：pip install -r requirements.txt 或 conda create -f environment.yml
- ****特殊依赖处理**：**CUDA版本需与PyTorch匹配(请到Pytorch官网：<https://pytorch.org/get-started/previous-versions/> 查看对应版本)

示例：安装PyTorch时指定版本 pip install torch==1.12.1+cu113

2.2.3 代码下载

- **从GitHub克隆仓库**：git clone https://github.com/ubicomplab/rPPG-Toolbox
- ****检查分支版本**：**切换到论文对应commit

示例：使用 git checkout a1b2c3d 回退到实验版本

2.2.4 数据集下载

- **官方渠道获取**：论文附录/项目官网
- **数据集预处理**：注意标注格式转换、数据集划分

示例：下载UBFC-rPPG: <https://drive.google.com/drive/folders/1o0XU4gTlo46YfwaWjlgbtCncc-oF44Xk>

2.2.5 硬盘挂载

如果你的数据集较大且无法直接下载到服务器，可以通过硬盘挂载的方式将数据集传输到服务器上。

- **查看硬盘**：`sudo fdisk -l`
- **查看分区**：`ls /dev/sd*` 或 `lsblk`
- **挂载硬盘**：`sudo mount /dev/sdb1 /mnt/data`（sdb1为硬盘分区，/mnt/data为挂载点）
- **取消挂载**：`sudo umount /mnt/data`
- **配置开机自动挂载**：`sudo vim /etc/fstab`，添加 `/dev/sdb1 /mnt/data ext4 defaults 0 0`（ext4为硬盘格式，可根据实际情况修改，部分硬盘可能不支持自动挂载，需要提前确认）

2.3 复现步骤

2.3.1 环境检验

- **Python版本检验**：

```
python --version
```

- **验证CUDA可用性**：

```
import torch
print(torch.cuda.is_available())
```

输出 True 表示GPU可用。

2.3.2 数据预处理

- **数据集加载**：从原始文件中读取数据，通常需要写一个 `data_loader.py` 文件,如果有多个数据集，可以写一个类，其他数据集继承这个类。
- **数据预处理**：对数据进行预处理，包括对坏数据的清洗和标注，以及对数据的格式转换和归一化处理，和如数据增强、标准化等。
- **数据集划分**：将数据集划分为训练集、验证集和测试集，通常按照**6:2:2**的比例划分。注意，划分时要确保是**跨样本划分**，即同一个样本不会同时出现在训练集、验证集和测试集中。
- **数据可视化**：对数据进行可视化，查看数据的分布和特征，以便更好地理解数据。

示例：对UBFC-rPPG数据集进行数据预处理，包括读取数据、数据格式转换、数据归一化、数据集划分和数据可视化。可以参考代码库：

https://github.com/ubicomplab/rPPG-Toolbox/tree/main/dataset/data_loader

2.3.3 模型训练

- **模型选择**：选择合适的模型，通常可以参考论文中的模型结构和超参数设置，通过命令行或者配置文件设置模型的超参数。
- **损失函数**：选择合适的损失函数，通常可以参考论文中的损失函数设置，也可以根据实际情况选择适合的损失函数，一般分类问题使用交叉熵损失函数（PyTorch中为 `nn.CrossEntropyLoss`），回归问题使用均方误差损失函数（PyTorch中为 `nn.MSELoss`）。
- **优化器**：选择合适的优化器，通常可以参考论文中的优化器设置，也可以根据实际情况选择适合的优化器，一般可以使用**Adam优化器**（PyTorch中为 `torch.optim.Adam`）。
- **训练过程记录**：在训练过程中，建议记录训练损失、验证损失、训练准确率、验证准确率等指标，同时绘制随轮次变化的训练损失和验证损失曲线，以便更好地了解模型的训练情况。
- **模型保存**：在训练过程中，可以保存模型的参数，以便后续的模型评估和模型预测。通常会保存模型的最佳参数或每隔一定轮次保存一次模型参数，避免训练过程中模型参数的丢失和占用过多的存储空间。
- **模型评估**：在训练结束后，可以对模型进行评估，计算模型在测试集上的准确率、精确率、召回率、F1值等指标，以便更好地了解模型的性能。
- **模型对比**：可以将复现的模型与原文中的模型进行对比，计算两者的性能差距，以便更好地了解模型的复现情况。

示例：使用PyTorch训练PPG相关的多个模型，设置模型的超参数、损失函数、优化器，记录训练过程，保存模型参数，评估模型性能，对比原文模型的性能。可以参考代码库：https://github.com/ubicomplab/rPPG-Toolbox/tree/main/neural_methods

2.3.5 结果分析

- **结果可视化：** 对模型的评估结果进行可视化，绘制**混淆矩阵**、**ROC曲线**、**PR曲线**等图表，回归任务可以绘制预测值和真实值的**散点图**，**Bland-Altman图**等。
- **结果解释：** 对模型的评估结果进行解释，纵向对比不同模型的输出结果分布，以便更好地理解模型的性能。重点可以关注bad case，找出模型预测错误的样本，分析错误的原因，提出改进的方案。
- **结果总结：** 对模型的评估结果进行总结，确保控制**单一变量**，研究是何种因素导致模型性能的提升或下降，并确保结果的可复现性。

2.4 如何确保你的工作可以被复现

你是否经常遇到这样的情况：在复现论文算法时，遇到了各种各样的问题，比如环境配置、代码写作规范、实验记录可视化、一站式config文件、环境配置文件等。有时候过了一段时间，你再次打开代码，发现自己已经忘记了当时的环境配置和实验记录，导致无法复现当时的工作。这些问题不仅会影响你的工作效率，还会影响你的科研能力提升。因此，我们需要确保我们的工作可以被复现，以便更好地提升我们的科研能力。

2.4.1 GitHub版本管理

不要使用命令的方式如 `main_v1.py` 这样的方式来管理代码版本，而是使用GitHub来管理代码版本，以便更好地追踪代码的修改历史。

- **创建代码仓库：** 在GitHub上创建一个代码仓库，将你的代码上传到GitHub上，以便更好地管理你的代码。命令行操作如下：

```
git init
git add .
git commit -m "first commit"
git branch -M main
git remote add origin
```

- **版本控制：** 在GitHub上使用版本控制，每次修改代码后，都要提交代码到GitHub上，以便更好地追踪代码的修改历史。命令行操作如下：

```
git add .
git commit -m "update code"
git push -u origin main
```

- **分支管理：** GitHub上使用分支管理，可以创建不同的分支，分别用于开发、测试、发布等不同的环境，以便更好地管理你的代码。命令行操作如下：

```
git branch dev
git checkout dev
git push origin dev
```

- **Pull Request：** 在GitHub上使用Pull Request，可以将你的代码提交到主分支，让其他人进行代码审查，以便更好地提升代码的质量。命令行操作如下：

```
git checkout main
git pull
git checkout dev
git merge main
git push
```

- **GitHub Readme：** 一定要在GitHub上使用Readme，详细将你的项目介绍、使用方法、贡献者等信息用Markdown语言写在Readme上，以便更好地展示你的项目，否则别人可能无法理解你的项目。命令行操作如下：

```
echo "# project" >> README.md
git add README.md
git commit -m "add README"
git push -u origin main
```

- **Issues管理：** 在GitHub上使用Issues管理，可以将你的问题和建议提交到Issues上，让其他人进行讨论和解决，以便更好地提升你的工作效率。命令行操作如下：

```
git checkout -b issue1
git add .
git commit -m "fix issue1"
git push origin issue1
```

- **GitHub 网站搭建：** 在GitHub上使用GitHub Pages，可以将你的项目网站托管在GitHub上，以便更好地展示你的项目。命令行操作如下：
示例：使用GitHub Pages搭建项目网站，可以参考GitHub Pages官方文档：<https://docs.github.com/en/pages>

2.4.2 代码写作规范

- **代码注释：** 在代码中添加注释，解释代码的功能、输入输出、实现原理等信息，以便更好地理解代码。命令行操作如下：

```
# This is a comment
You can also write a paragraph of comments to introduce the function of the code. You can start with `"""` and end with `"""`.
```

2.4.3 实验记录可视化

- **实验输入记录：** 模型训练前，可以通过批量抽样的方式，将训练数据的输入和标签保存为图片预览，以便更好地了解数据的分布和特征。
- **训练过程记录：** 实验过程中，可以使用 TensorBoard 和 wandb 进行实验记录可视化，记录训练损失、验证损失、训练准确率、验证准确率等指标，同时绘制随轮次变化的训练损失和验证损失曲线，以便更好地了解模型的训练情况。
- **实验结果记录：** 实验结束后，自动化地保存实验结果，包括模型参数、模型评估结果、模型预测结果等信息，同时自动化绘制混淆矩阵、ROC曲线、PR曲线等图表。

2.4.4 一站式config文件

- **配置文件：** 在代码中使用配置文件，将模型的超参数、损失函数、优化器等信息保存在配置文件中，以便更好地管理模型的超参数，同时可以通过修改配置文件来调整模型的超参数；将数据集的路径、数据集的格式、数据集的划分等信息保存在数据集配置文件中，以便更好地管理数据集的配置，同时可以通过数据集配置文件来配置数据集，提高代码的可复现性。
示例：使用 `yaml` `json` `toml` 等配置文件，将数据集、实验设置、模型超参数等信息保存在配置文件中，以便更好地管理实验配置，提高代码的可复现性。参考代码库：https://github.com/ubicomplab/rPPG-Toolbox/tree/main/configs/train_configs

2.4.5 环境配置文件

- **环境配置文件：** 在代码中使用 `requirements.txt` 或 `environment.yml` 文件，将代码的依赖包、Python版本、CUDA版本等信息保存在环境配置文件中，以便更好地配置代码的运行环境，同时可以通过环境配置文件来配置代码的运行环境，提高代码的可复现性。

TODO

- ☐ 如何调试硬件
- ☐ 如何进行用户实验
- ☐ 如何撰写论文
- ☐ 如何正确提问