课程大作业LibFewShot

Course Project LibContinual

吕松霖

lvsl@lamda.nju.edu.cn

2025年5月12日

大纲

- LibFewShot简介
- 环境及作业要求
- 使用及安装指南
- Q & A
- 总结

大纲

- LibFewShot简介
- 环境及作业要求
- 使用及安装指南
- Q & A
- 总结

LibFewShot框架介绍

南京大学 NANJING UNIVERSITY

■ LibFewShot

❑ **系统应用**：开源了统一的**小样本学习算法库LibFewShot**，提供了公平对比平台，GitHub Star数900+

VIG@R&L

Visual Intelligence Group (VIG), Reasoning and Learning Research Group, Nanjing University.

21 followers　Nanjing, China　https://cs.nju.edu.cn/rl/

Pinned

LibFewShot Public

LibFewShot: A Comprehensive Library for Few-shot Learning. TPAMI 2023.

Python　961　182

LibContinual Public

A Framework of Continual Learning

Python　78　10

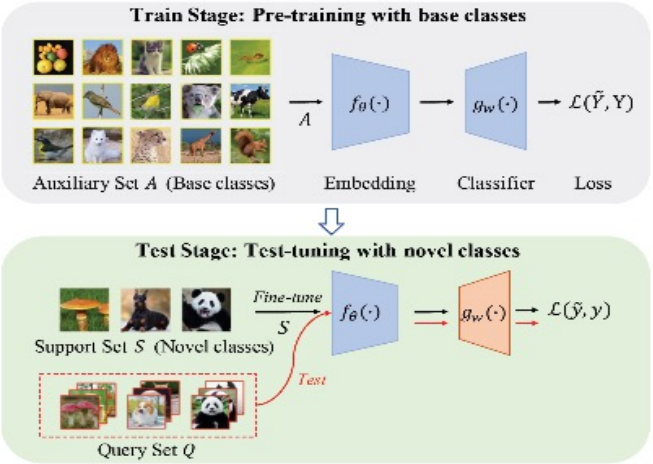
- ❑ **统一的代码架构，保证公平性**
- ❑ **统一的训练设置，保证公平性**
- ❑ **透明的训练Tricks，保证可复现性**
- ❑ **支持多GPU训练、多插曲式训练**
- ❑ **支持YAML配置、框架解耦性高**
- ❑ **提供丰富的数据集和预训练模型库**

LibFewShot: A Comprehensive Library for Few-shot Learning [TPAMI 2023]

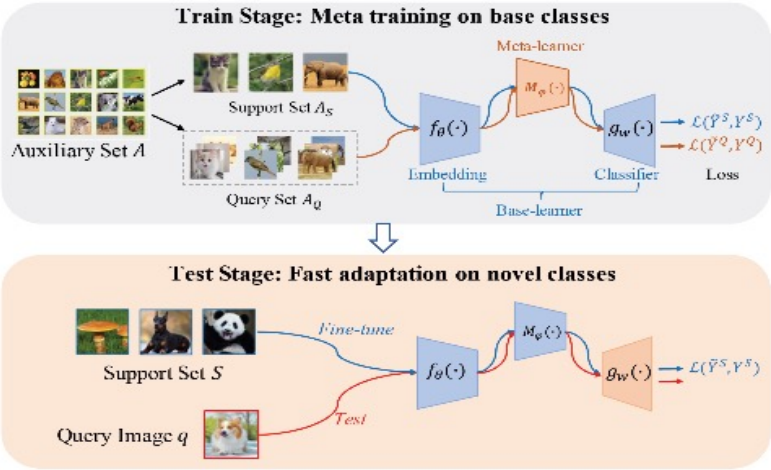
LibFewShot介绍

■ LibFewShot

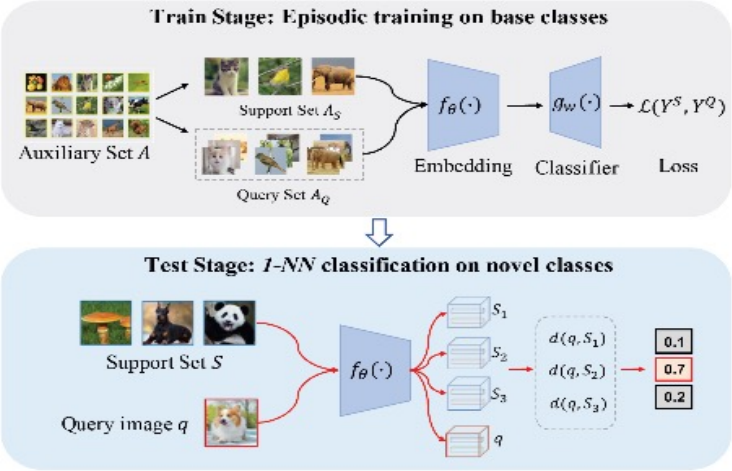
- ❑ 包含基于非插曲式小样本学习算法8种
- ❑ 包含基于元学习的小样本学习算法10种
- ❑ 包含基于度量学习的小样本学习算法14种



(a) Non-episodic based methods



(b) Meta-learning based methods



(c) Metric-learning based methods

- **Baseline** (ICLR 2019)
- **Baseline++** (ICLR 2019)
- **RFS-simple** (ECCV 2020)
- **RFS-distill** (ECCV 2020)
- **S2M2** (WACV 2020)
- **Neg-Cosine** (ECCV 2020)
- **SKD** (BMVC 2021)

- **MAML** (ICML 2017)
- **Versa** (NeurIPS 2018)
- **R2D2** (ICLR 2019)
- **LEO** (ICLR 2019)
- **MTL** (CVPR 2019)
- **ANIL** (ICLR 2020)
- **BOIL** (ICLR 2021)

- **ProtoNet** (NeurIPS 2017)
- **RelationNet** (CVPR 2018)
- **CovaMNet** (AAAI 2019)
- **DN4** (CVPR 2019)
- **CAN** (NeurIPS 2019)
- **FEAT** (CVPR 2020)
- **RENet** (ICCV 2021)
- **DeepBDC** (CVPR 2022)

https://www.computer.org/csdl/journal/tp/5555/01/10239698/1Qck3o85t60

LibFewShot介绍

■ LibFewShot的核心特性包括：

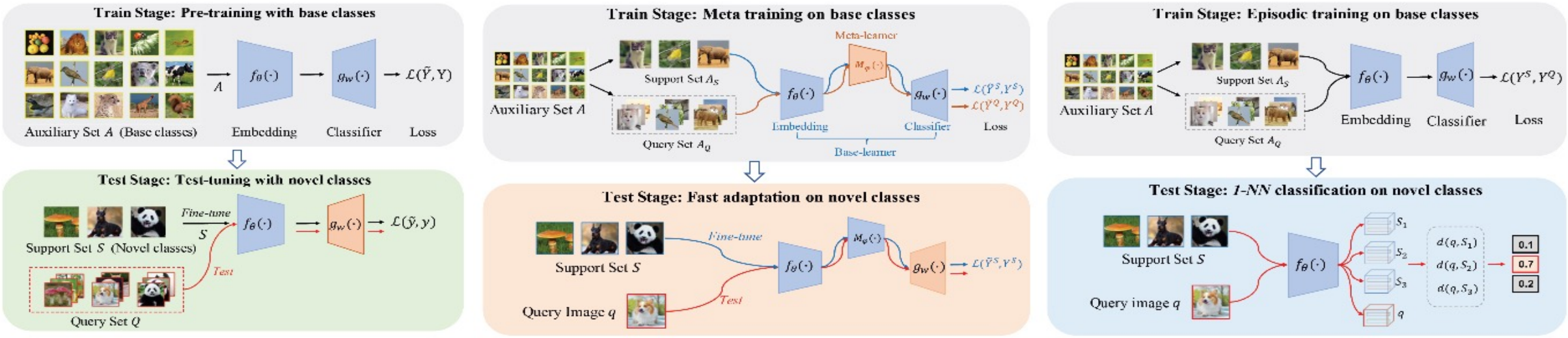
- 模块化设计：项目采用了高度模块化的结构，将数据处理、模型定义、训练过程等关键部分封装为独立的组件，便于扩展和定制。
- 丰富的预训练模型：库中包含了多种流行的深度学习模型，如ResNet, ProtoNet, MatchingNet等，这些模型经过优化，适用于少样本学习场景。
- 多样化数据集支持：支持多个标准的少样本学习数据集，如MiniImageNet, CIFAR-FS, TieredImageNet等，同时也易于添加自定义数据集。
- 高效的训练机制：采用PyTorch作为基础框架，利用其动态图计算的优势，实现快速的模型训练和评估。
- 可复现性保证：提供了详细的文档和示例代码，确保实验结果可重复，便于验证和比较不同方法的效果。

https://www.computer.org/csdl/journal/tp/5555/01/10239698/1Qck3o85t60

LibFewShot介绍

■ LibFewShot的泛用领域：

- 计算机视觉：在分类任务中，当新类别样本稀少时，如图像识别、目标检测等。
- 自然语言处理：在低资源语言的翻译、问答系统等方面。
- 机器人学习：使机器人能在有限的交互中快速学习新的任务。
- 强化学习：在环境变化大或无法获取大量标注数据的情况下。



(a) Non-episodic based methods (b) Meta-learning based methods (c) Metric-learning based methods

https://www.computer.org/csdl/journal/tp/5555/01/10239698/1Qck3o85t60

LibFewShot介绍

■ LibFewShot复现了多篇代表性Fewshot算法，并提供了公平比较的结果：

Method	Embed.	5-way 1-shot		5-way 5-shot	
		Reported	Ours	Reported	Ours
Baseline	Conv64F	42.11	42.34	62.53	62.18
	ResNet18	51.75	51.18	74.27	74.06
Baseline++	Conv64F	48.24	46.21	66.43	65.18
	ResNet18	51.87	53.60	75.68	73.63
RFS-simple	ResNet12	62.02	62.80	79.64	79.57
RFS-distill	ResNet12	64.82	63.44	82.14	80.17
SKD-GEN0	ResNet12	65.93	66.40	83.15	83.06
SKD-GEN1	ResNet12	67.04	67.35	83.54	80.30
NegCos	ResNet12	63.85	63.28	81.57	81.24
MAML	Conv32F	48.70	47.41	63.11	65.24
Versa	Conv64F†	53.40	51.92	67.37	66.26
R2D2	Conv64F	49.50	47.57	65.40	66.68
	Conv64F‡	51.80	55.53	68.40	70.79
ANIL	Conv32F	46.70	48.44	61.50	64.35
BOIL	Conv64F	49.61	48.00	66.45	64.39
	ResNet12**	-	58.87	71.30	72.88
MTL	ResNet12	60.20	60.20	74.30	75.86

Conv64F

Method	Venue	Type	miniImageNet		tieredImageNet	
			1-shot	5-shot	1-shot	5-shot
Baseline	ICLR'19	Non-episodic	44.90	63.96	48.20	68.96
Baseline++	ICML'19	Non-episodic	48.86	63.29	55.94	73.80
RFS-simple	ECCV'20	Non-episodic	47.97	65.88	52.21	71.82
SKD-GEN0	BMVC'20	Non-episodic	48.14	66.36	51.78	70.65
NegCos	ECCV'20	Non-episodic	47.34	65.97	51.21	71.57
MAML	ICML'17	Meta	49.55	64.92	50.98	67.12
Versa	NeurIPS'18	Meta	52.75	67.40	52.28	69.41
R2D2	ICLR'19	Meta	51.19	67.29	52.18	69.19
LEO	ICLR'19	Meta	53.31	67.47	58.15	74.21
MTL	CVPR'19	Meta	40.97	57.12	42.36	64.87
ANIL	ICLR'20	Meta	48.01	63.88	49.05	66.32
BOIL	ICLR'21	Meta	47.92	64.39	50.04	65.51
ProtoNet	NeurIPS'17	Metric	47.05	68.56	46.11	70.07
RelationNet	CVPR'18	Metric	51.52	66.76	54.37	71.93

https://www.computer.org/csdl/journal/tp/5555/01/10239698/1Qck3o85t60

大纲

- ❑ LibFewShot简介
- ❑ 环境及作业要求
- ❑ 使用及安装指南
- ❑ Q & A
- ❑ 总结

环境与数据集要求

- 环境要求
  - 带有CUDA支持的PyTorch环境
  - 安装及环境测试，可以参考https://pytorch.org/get-started/locally/#linux-installation
  - LibFewShot只在Linux+CUDA上测试过，因此不保证在Windows环境下没有使用问题。
- 数据集
  - 小样本分类一般使用miniImageNet、tieredImageNet等数据集
  - 下载链接在：https://github.com/RL-VIG/LibFewShot#datasets，或者使用 https://box.nju.edu.cn/d/7f6c5bd7cfaf4b019c34/
  - 下载完之后解压到你喜欢的目录就可以

作业要求

- 在【腾讯文档】LibFewShot论文复现_2025春季_机器学习导论 填写组队信息中自行填写自己队伍信息，1-2人一队，一篇文章最多两队
- 上述腾讯文档选择一篇小样本论文进行复现，也自行选择近几年顶会经典的小样本学习算法（需与助教商量，基本上小样本分类任务均可整合）
- 需要在LibFewShot框架基础上完成算法复现，与论文中汇报的精度误差在2%以内，代码规范
- 完成实验报告，报告内陈述：对算法的理解，算法复现的难点以及如何解决，复现结果表格，
- 实验报告 无字数要求，提交时将实验报告+训练后模型+增加了所复现代码的代码包三者打包的压缩包（只包含复 现成功的模型和实验结果，如果压缩包太大请检查是否有太多冗余内容）发送至 nju_ml@163.com，
- 一个队伍提交一份 提交内容命名要求: 春2025_libfewshot_队伍编号.rar （邮件主题名去掉.rar）

大纲

- ❑ LibFewShot简介
- ❑ 环境及作业要求
- ❑ 使用及安装指南
- ❑ Q & A
- ❑ 总结

LibFewShot使用指南

■ 中文文档

- 其提供了**专业的中文说明文档**，方便快速入门及整合代码

入门
介绍
贡献者
安装
入门
教程
编写.yaml配置文件
训练/测试LibFewShot中已集成的方法
使用数据集
Transformations
增加一个新的Backbone
增加一个新的分类器
模型库
Model Zoo
贡献
贡献代码

Welcome to LibFewShot's documentation!

入门

- 介绍
- 贡献者
- 安装
  - 获取LibFewShot
  - 配置 `LibFewShot` 环境
  - 测试安装是否正确
  - 后续

https://libfewshot-en.readthedocs.io/zh-cn/latest/index.html

安装LibFewShot

- 下载文件：
  - 执行如下命令：git clone https://github.com/RL-VIG/LibFewShot.git

    在本地电脑上安装LibFewShot库
- 进入相应的目录：
- cd <path-to-LibFewShot>
- 上述代码执行完后，电脑上应有LibFewShot文件夹，效果如下：

```
(base) PS D:\OneDrive\Desktop> cd .\LibFewShot\
(base) PS D:\OneDrive\Desktop\LibFewShot>
```

安装LibFewShot

- 新建一个conda环境，python版本建议大于等于3.9，执行如下代码
  - conda create -n libfewshot python=3.9
  - 后续选项输入y即可自动安装其设定的基础环境
- 注：此步骤需确保你的电脑上存在conda环境，否则自行百度安装

```
(base) PS C:\Users\admin> conda create -n libfewshot python=3.9
Retrieving notices: ...working... done
Channels:
 - defaults
Platform: win-64
Collecting package metadata (repodata.json): done
Solving environment: done

## Package Plan ##

  environment location: C:\Users\admin\.conda\envs\libfewshot

  added / updated specs:
    - python=3.9
```

中间过程输入y(yes)

```
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate libfewshot
#
# To deactivate an active environment, use
#
#     $ conda deactivate

(base) PS C:\Users\admin>
```

配置完成效果

安装LibFewShot

- 执行conda activate libfewshot 进入激活环境

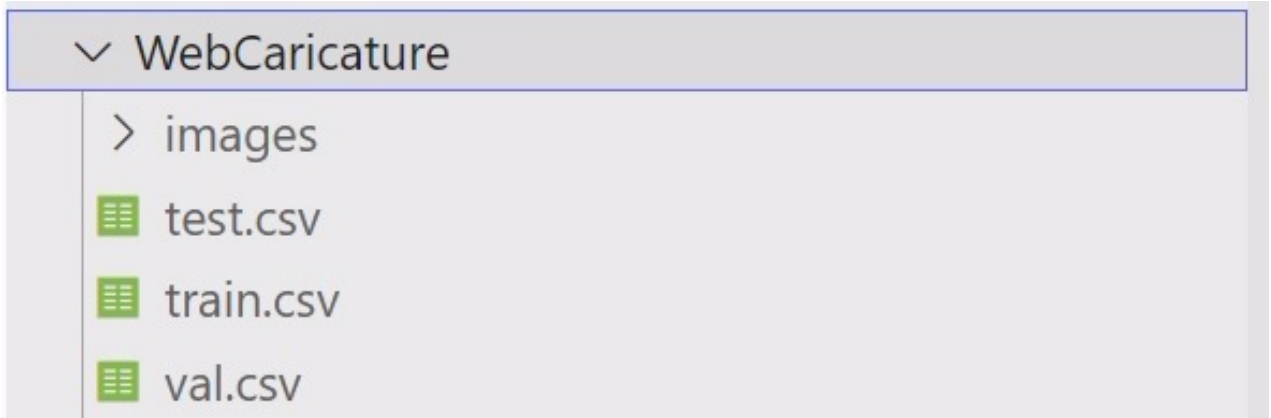
```
(base) PS D:\OneDrive\Desktop\LibFewShot> conda activate libfewshot
(libfewshot) PS D:\OneDrive\Desktop\LibFewShot>
```

- 输入 pip install -r requirements.txt安装相关依赖。
- 注意事项：
  - 务必确保pip指令在LibFewShot文件夹下执行，否则会找不到相关txt文件。
  - 运行此库需安装torch，建议到torch官网自行下载有cuda版本的torch 安装。
  - 为确保重复安装torch，可通过—no-deps参数忽略相关包。

pip install -r requirements.txt --no-deps torch

数据集下载及配置

- 建议直接到LibFewShot库中下载相关数据集，链接如下：
  - Caltech-UCSD Birds-200-2011, Standford Cars, Standford Dogs, *mini*ImageNet, *tiered*ImageNet and WebCaricature are available at Google Drive and 百度网盘(提取码：yr1w).
- 注意：直接从官网下载数据集可能需要二次处理。需检查数据文件中是否含有train.csv文件，若无该文件将无法运行代码。
- 以WebCaricature为例：



测试是否安装成功

- 修改run_trainer.py中config设置: config=Config("./config/test_install.yaml").get_config_dict()

```
# config = Config("./reproduce/Baseline/Baseline-miniImageNet--ravi-Conv64F-5-Reproduce.yaml").get_config_dict()
config = Config("./config/test_install.yaml").get_config_dict()

if config["n_gpu"] > 1:
    os.environ["CUDA_VISIBLE_DEVICES"] = config["device_ids"]
    torch.multiprocessing.spawn(main, nprocs=config["n_gpu"], args=(config,))
else:
```

- 修改config/headers/data.yaml中的data_root为你的数据集路径
- 终端中执行python run_trainer.py
- 若第一个epoch输出正常，则表明LibFewShot已成功安装：

```
  self._data.average[key] = self._data.total[key] / self._data.counts[key]
[04/21/25 23:36:32] INFO     Epoch-(0): [100/100]    Time 0.031 (0.023)      Calc 0.005   trainer.py:372
                             (0.014)      Data 0.021 (0.005)      Loss 4.429 (73.914)
                             Acc@1 41.333 (27.680)
                    INFO      * Acc@1 27.680                                              trainer.py:372
                    INFO     ============ Validation on the val set ============         trainer.py:372
[04/21/25 23:36:33] INFO     Epoch-(0): [100/100]    Time 0.004 (0.011)      Calc 0.002   trainer.py:372
                             (0.002)      Data 0.000 (0.008)      Acc@1 41.333 (35.267)
                    INFO      * Acc@1 35.267 Best acc -inf                                trainer.py:372
                    INFO     ============ Testing on the test set ============           trainer.py:372
[04/21/25 23:36:35] INFO     Epoch-(0): [100/100]    Time 0.004 (0.011)      Calc 0.002 (0.002)      Data 0.001 (0.008)      Acc@1 25.333 (36.987)
                    INFO      * Acc@1 36.987 Best acc -inf
                    INFO      * Time: 0:00:04/0:03:20
```

可能的报错

```
  File "D:\OneDrive\Desktop\LibFewShot\core\trainer.py", line 16, in <module>
    from core.data import get_dataloader
  File "D:\OneDrive\Desktop\LibFewShot\core\data\__init__.py", line 2, in <module>
    from .dataloader import get_dataloader
  File "D:\OneDrive\Desktop\LibFewShot\core\data\dataloader.py", line 8, in <module>
    from .collates import get_collate_function, get_augment_method,get_mean_std
  File "D:\OneDrive\Desktop\LibFewShot\core\data\collates\__init__.py", line 2, in <module>
    from .collate_functions import GeneralCollateFunction, FewShotAugCollateFunction
  File "D:\OneDrive\Desktop\LibFewShot\core\data\collates\collate_functions.py", line 3, in <module>
    from collections import Iterable
ImportError: cannot import name 'Iterable' from 'collections' (D:\ProgramData\anaconda3\Lib\collections
\__init__.py)
```

出现原因：本人在Windows环境下配置，易出现环境版本类的问题。

代码尝试从 collections 模块导入 Iterable 类型，但是失败了。这是因为在 Python 3.3 及以后的版本中，Iterable 类型被移到了 collections.abc 模块中。因此，需要更新你的导入语句以反映这一变化。

解决方案：

- 进入报错的文件将 from collections import Iterable 修改为：from collections.abc import Iterable
- 在Linux环境下配置

Demo

尝试通过修改LibFewShot/config/test_install.yaml配置，训练Resnet12 在WebCaricature 的效果。通过该过程学习LibFewShot的基本使用方法。

可参考：LibFewShot/reproduce/Baseline++/BaselinePlus-tiered_imagenet-resnet12-Table2.yaml文件

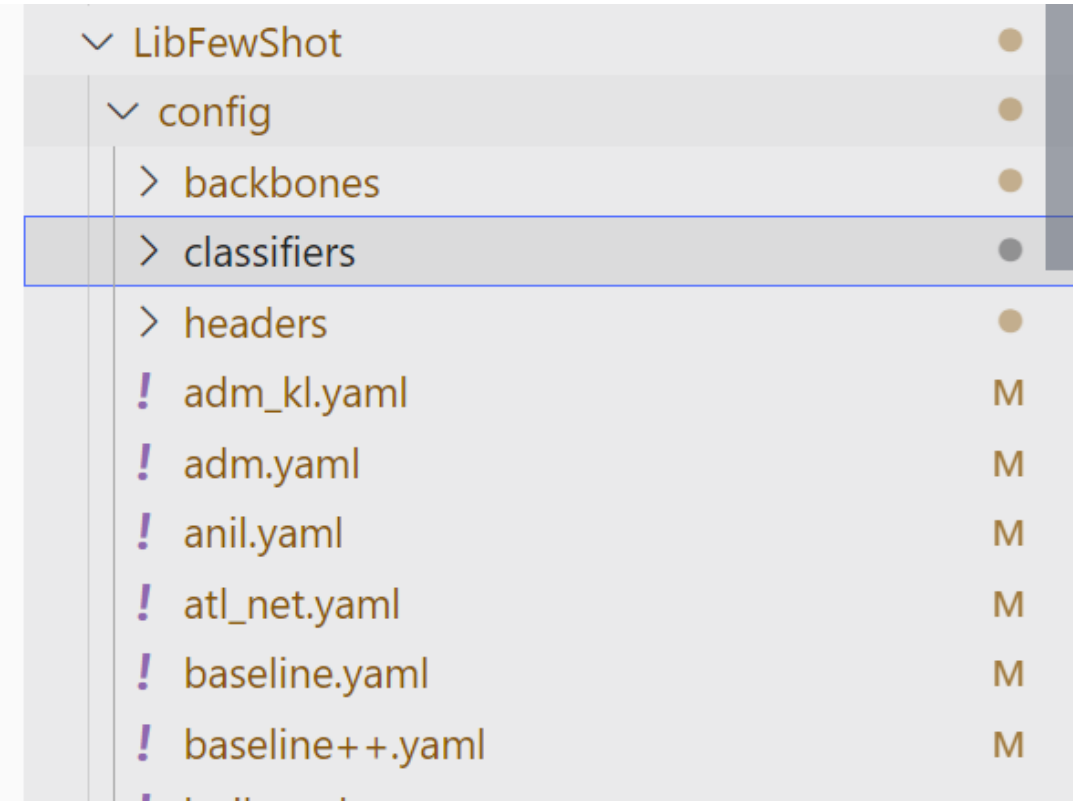
结果如下：

```
                    INFO    ============ Train on the train set ============
                    INFO    learning rate: [0.01]
[05/09/25 21:06:34] INFO    Epoch-(35): [80/80]     Time 0.090 (0.090)      Calc 0.034 (0.034)      Data 0.000 (0.000)      Loss 0.003 (0.004)      Acc@1 100.000 (99.980)
                    INFO     * Acc@1 99.980
                    INFO    ============ Validation on the val set ============
[05/09/25 21:06:56] INFO    Epoch-(35): [100/100]   Time 0.218 (0.219)      Calc 0.217 (0.218)      Data 0.000 (0.000)      Acc@1 82.667 (78.360)
                    INFO     * Acc@1 78.360 Best acc 78.800
                    INFO    ============ Testing on the test set ============
[05/09/25 21:07:19] INFO    Epoch-(35): [100/100]   Time 0.173 (0.227)      Calc 0.171 (0.226)      Data 0.000 (0.000)      Acc@1 80.000 (76.627)
                    INFO     * Acc@1 76.627 Best acc 76.747
                    INFO     * Time: 0:31:50/0:44:12.777778
                    INFO    ============ Testing on the test set ============
[05/09/25 21:18:46] INFO    Epoch-(48): [100/100]   Time 0.219 (0.221)      Calc 0.218 (0.219)      Data 0.000 (0.000)      Acc@1 92.000 (78.427)
                    INFO     * Acc@1 78.427 Best acc 79.947
                    INFO     * Time: 0:43:17/0:44:10
[05/09/25 21:18:47] INFO    ============ Train on the train set ============
                    INFO    learning rate: [0.01]
[05/09/25 21:18:54] INFO    Epoch-(49): [80/80]     Time 0.090 (0.090)      Calc 0.034 (0.035)      Data 0.000 (0.000)      Loss 0.003 (0.005)      Acc@1 100.000 (99.971)
                    INFO     * Acc@1 99.971
                    INFO    ============ Validation on the val set ============
[05/09/25 21:19:17] INFO    Epoch-(49): [100/100]   Time 0.219 (0.234)      Calc 0.218 (0.232)      Data 0.000 (0.000)      Acc@1 70.667 (78.587)
                    INFO     * Acc@1 78.587 Best acc 79.653
                    INFO    ============ Testing on the test set ============
[05/09/25 21:19:39] INFO    Epoch-(49): [100/100]   Time 0.219 (0.219)      Calc 0.218 (0.218)      Data 0.000 (0.000)      Acc@1 88.000 (79.507)
                    INFO     * Acc@1 79.507 Best acc 79.947
                    INFO     * Time: 0:44:10/0:44:10
[05/09/25 21:19:40] INFO    End of experiment, took 0:44:10
                    INFO    Result DIR: ./results/BaselinePlus-resnet12-5-1-May-09-2025-20-35-28
```

LibFewShot框架介绍

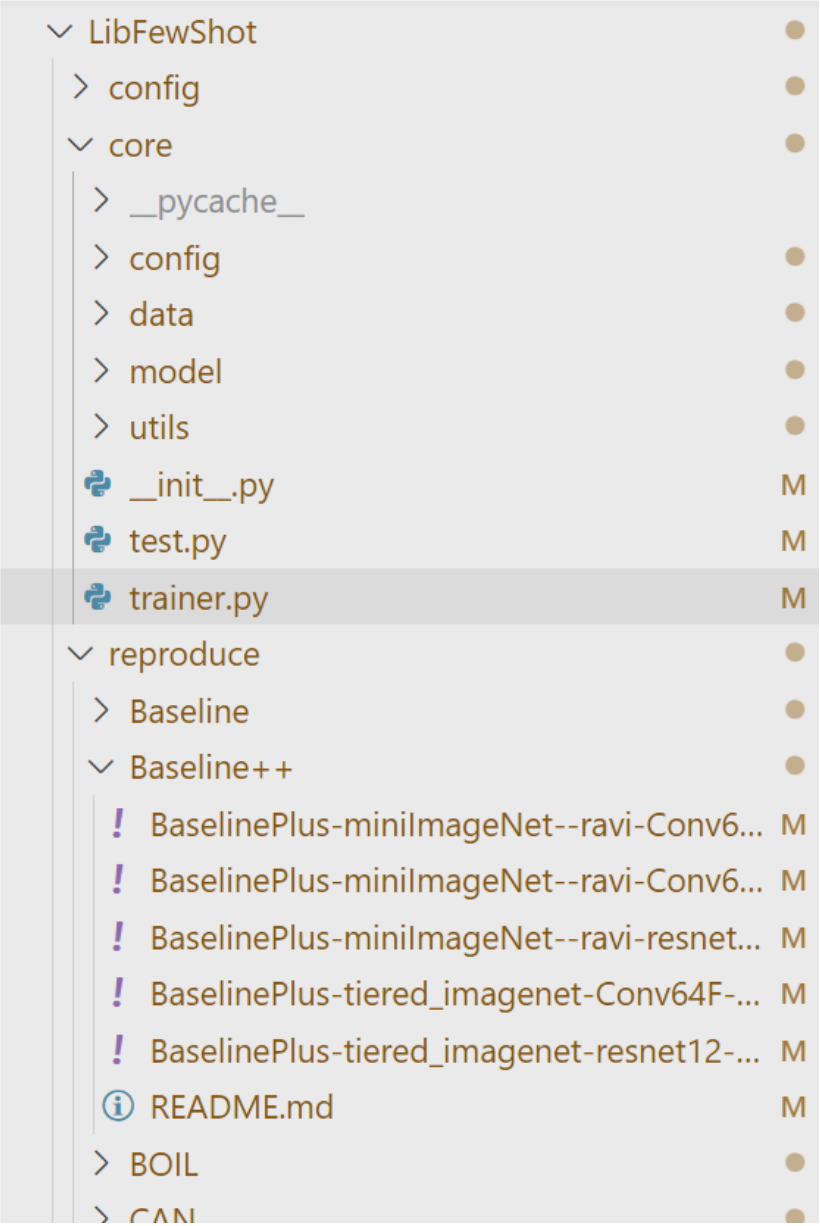
➢ Config文件夹

- ✓ config文件夹主要为LibFewShot内置的模型的初始化模型文件yaml，及模型中的backbone，classifier和header文件。
- ✓ 下属若干文件的yaml中，首先调用yaml文件中罗列的参数，如果没有再去访问includes中包括的data.yaml，device.yaml等文件。




```yaml
includes:
  - headers/data.yaml
  - headers/device.yaml
  - headers/misc.yaml
  - headers/model.yaml
  - headers/optimizer.yaml
  - classifiers/Baseline++.yaml
  - backbones/resnet12.yaml
```

Demo



- core文件夹
  - ✓ 核心模块，实现了模型架构，损失函数和优化器的内部结构。另外有train和test训练所调用的内部类架构。简而言之，core文件实现基本函数，和基本的类，包括损失函数，神经网络，数据集的构建，而config文件夹作为core文件中若干函数，类的参数。
- reproduce文件夹
  - ✓ 该文件夹下的readme.md介绍了不同的神经网络在预训练模型上的训练分数，对比了5-way 1-shot和5-way 5-shot，miniimagenet和tieredimagenet，conv64、resnet12和resnet18在不同网络的分数。其中一般来说，在tieredimagenet训练集上训练的分数高，resnet比conv网络显著提升，resnet12在有些情况下甚至高于resnet18，所以要注意看一下对比实验的训练效果。
  - ✓ 另外，在微调模型上，再加上resnet网络模型，可能会导致显存爆炸，所以需要降低batchsize。
  - ✓ 在reproduce文件夹下的readme中存在若干文件夹，这些是当时训练预训练模型时的参数，可以进行参考。

修改主配置文件

1）如果对于非微调方法的网络，是没有cls和emb路径的，所以不用考虑，对于微调方法来说，如果基于tiercedimage数据集的，在reproduce文件夹下的readme中会介绍这两者的预训练模型，而miniimage数据集没有处理这两者的预训练模型（所以优先考虑使用tiercedimage数据集的预训练模型）。
2）修改神经网络预训练模型为指定路径，这个要么在config/model.yaml，要么文件里已经写出可以直接修改，（要同时修改还是修改一个，记住yaml里罗列的参数优先，如果找不到该参数才会找includes中的yaml文件）
3）修改config/data.yaml文件夹中的数据集路径为所训练数据集路径
4）观察主配置文件夹中的backbone，classfier是否对应预训练模型的要求，若不满足则修改
5）观察config/model.yaml文件夹中的way_num、shot_num、query_num是否满足条件，前两者就是K way-C shot的K和C，query_num是指每次运用支持集时用了多少张测试图片来评判，test_way、test_shot、test_query一般来说跟上面相同即可。

```yaml
augment: true
augment_times: 1
augment_times_query: 1
backbone:
  kwargs:
    avg_pool: true
    is_flatten: true
    keep_prob: 0.0
  name: resnet12
batch_size: 128
classifier:
  kwargs:
    feat_dim: 640
    inner_param:
      inner_batch_size: 4
      inner_optim:
        kwargs:
          dampening: 0.9
          lr: 0.01
          momentum: 0.9
          weight_decay: 0.001
        name: SGD
      inner_train_iter: 100
    num_class: 351
  name: BaselinePlus
data_root: /data/tiered_imagenet
deterministic: true
device_ids: 0
episode_size: 1
epoch: 100
image_size: 84
includes:
```

大纲

- ❑ LibFewShot简介
- ❑ 环境及作业要求
- ❑ 使用及安装指南
- ❑ Q & A
- ❑ 总结

- Q&A 我需要做什么，怎么整合到框架里?

训练/测试LibFewShot中已集成的方法

使用数据集

Transformations

增加一个新的Backbone

增加一个新的分类器

metric based

meta learning

fine tuning

1. 学会测试并使用已经集成的方法
2. 寻找近期的类似的Few-shot Classification文章，将其整理到LibFewShot库中；
3. 测试整理的是否正确以及能否复现论文中的结果
4. 可能需要改动的文件：

   core/model/abstract_model.py core/model/meta/* core/model/metric/* core/model/pretrain/*
5. 若复现良好可贡献相关代码，成为贡献者。
6. 详情请参考使用LibFewShot使用文档。

- Q&A 我没有支持CUDA的电脑/服务器可以用，怎么办?
  - 如果你的电脑有NVIDIA的显卡，那一般是可以安装CUDA的，请参考 https://developer.nvidia.co m/cuda-toolkit;
  - 可以使用Google的Colab或者Azure的免费服务器;
  - 如果你的电脑有AMD的显卡并且你的硬件可以在硬软件支持中找到，那么你可以尝试安装ROCm版本的PyTorch，也可以使用;
  - 如果你使用的是M系列芯片的Macbook，PyTorch已经支持MPS后端，虽然LibFewShot还没有在 该环境下测试过，但应该兼容，可能会有点小问题需要解决;
  - 如果上面集中方法还是比较困难，请参考最后一个Q&A。

Q&A 我找到了一个算法，GitHub已经有他的官方实现/第三方实现了，我可以借鉴吗?

- 可以，这会减轻很多的复现难度，但请注意以下几点： 这份实现是不是正确的？包括但不限于： 1）我使用这个代码，按照对应的配置文件，能不能跑出原文声称的结果？2）仓库的issue列表 里，有没有对复现结果的争议？3）算法实现细节是否和原文描述的一致？
- 这份代码如果是TensorFlow或者MXNet等实现的，那么在参考复现的过程中，需要对比与PyTorch间操作的差异。如果有核心代码的借鉴，需要在所增加的分类头文件的开始，添加原仓库的License或者来源声明。
- 鼓励在原实现上进行精简和优化，例如PyTorch内置了很多奇怪的函数可以快速地解决某些复杂运算。

Q&A 我实现了论文的算法，但是训练完之后精度很低，这是为什么？

请从以下几个方面排查原因：

- 检查训练时载入的参数，例如学习率等，是否和原文/原代码一致？
- 检查训练和测试时数据增广是否和原文一致？
- 原文是否使用了预训练模型？
- 实现上的问题？

Q&A 我发现了LibFewShot的一个BUG！我该怎么反馈？

- 你可以在库的issue里直接提出来，并附上对该bug的描述。如果你还不确定这是不是一个bug， 请看最后一个 Q&A。

Last Q&A 我还有其他的问题，怎么办?

咨询助教！！！

- 助教团队：陈怿飏（522023330016@smail.nju.edu.cn）
  赖　伟（522023330042@smail.nju.edu.cn）
  李　兵（522023330043@smail.nju.edu.cn）
  吴韫琛（652024330030@smail.nju.edu.cn）
  吕松霖（lvsl@lamda.nju.edu.cn）
  刘尚格（lshangge@smail.nju.edu.cn）

大纲

- ❑ LibFewShot简介
- ❑ 环境及作业要求
- ❑ 使用及安装指南
- ❑ Q & A
- ❑ 总结

作业要求

- 在【腾讯文档】LibFewShot论文复现_2025春季_机器学习导论 填写组队信息中自行填写自己队伍信息，1-2人一队，一篇文章最多两队
- 上述腾讯文档选择一篇小样本论文进行复现，也自行选择近几年顶会经典的小样本学习算法（需与助教商量，基本上小样本分类任务均可整合）
- 需要在LibFewShot框架基础上完成算法复现，与论文中汇报的精度误差在2%以内，代码规范
- 完成实验报告，报告内陈述：对算法的理解，算法复现的难点以及如何解决，复现结果表格，
- 实验报告 无字数要求，提交时将实验报告+训练后模型+增加了所复现代码的代码包三者打包的压缩包（只包含复 现成功的模型和实验结果，如果压缩包太大请检查是否有太多冗余内容）发送至 nju_ml@163.com，
- 一个队伍提交一份 提交内容命名要求: 春2025_libfewshot_队伍编号.rar （邮件主题名去掉.rar）

复现成功奖励

- 成功复现LibFewShot中的一篇论文，满足精度达标，代码规范。可以合并进LibFewShot仓库
- 一篇论文成功合并会有额外的Bonus
- 一篇论文仅限一支队伍或一个人，先到先得
- 总共不超过10篇论文
- 希望参与的队伍可以联系助教

LibFewShot总结

■ 特点与优势

- 易用性：API 设计简洁，易于上手，即使对 PyTorch 不熟悉的研究者也能快速入门。
- 研究友好：方便进行新算法的实现和现有算法的比较，加速研究进程。
- 社区活跃：持续更新和完善，积极接受社区贡献，保证项目的活力与质量。

■ 结语

如果你希望利用少量样本数据下解决一些机器学习问题，或者对少样本学习感兴趣，LibFewShot无疑是一个值得尝试的工具。其强大的功能和友好的用户界面，将助力你在科研和应用开发中取得更大的突破。赶快开始你的少样本学习之旅吧！