ProtoNet复现实验报告

201220138 白志欣

> 注：与原文结果相差不大的实验结果在复现任务的三、四以及其它代码复现的结果分析当中。

复现结果：

zldscr0/ProtoNet_result (github.com)

中可查看该实验报告中的几个实验结果和最优模型。

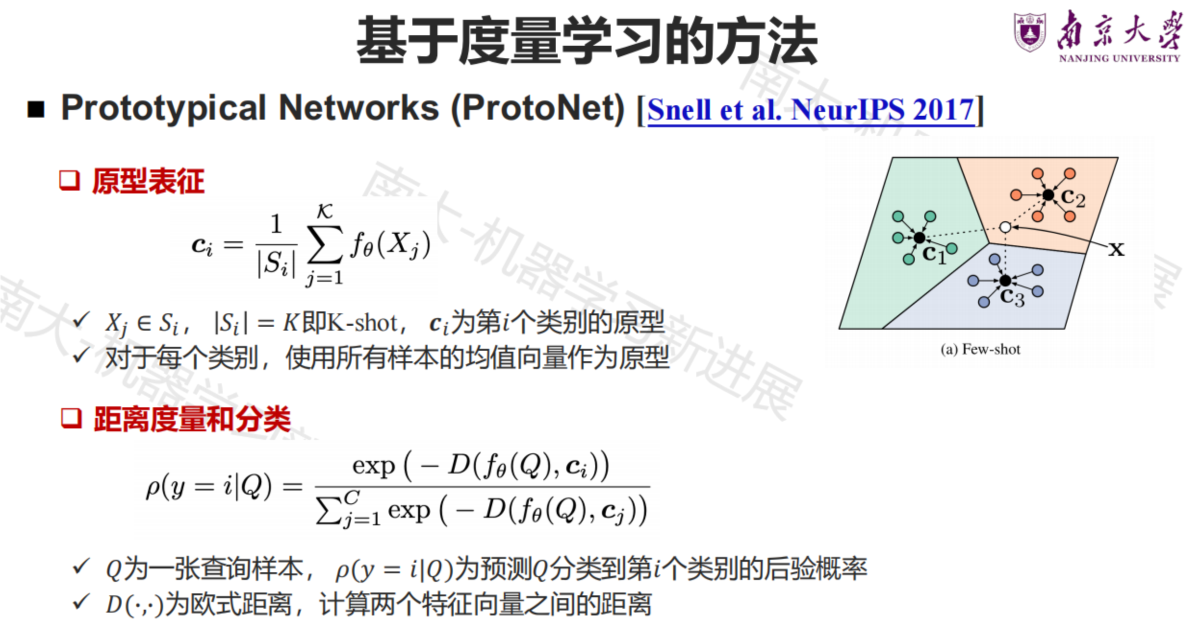
简介

Paper：1703.05175.pdf (arxiv.org)

Code：jakesnell/prototypical-networks: Code for the NeurIPS 2017 Paper "Prototypical Networks for Few-shot Learning" (github.com)

[RL-VIG/LibFewShot: LibFewShot: A Comprehensive Library for Few-shot Learning, TPAMI 2023. (github.com)](https://github.com/RL-VIG/LibFewShot)

在该论文所提出的原型网络方法中，需要将样本投影到一个度量空间，且在这个空间中同类样本距离较近，异类样本的距离较远。下图为这个投影空间的示意图，假如在这个投影空间中，存在三个类别的样本，且相同类别的样本间距离较近。为了给一个未标注样本x进行标注，则将样本x投影至这个空间并计算x与各个类别的原型距离，离得近的就认为x属于哪个类别。



基于度量学习的方法

■ Prototypical Networks (ProtoNet) [Snell et al. NeurIPS 2017]

❑ 原型表征

$$c_i = \frac{1}{|S_i|}\sum_{j=1}^{K} f_\theta(X_j)$$

✓ $X_j \in S_i$， $|S_i| = K$即K-shot， c_i为第i个类别的原型

✓ 对于每个类别，使用所有样本的均值向量作为原型

❑ 距离度量和分类

$$\rho(y=i|Q) = \frac{\exp\left(-D(f_\theta(Q), c_i)\right)}{\sum_{j=1}^{C}\exp\left(-D(f_\theta(Q), c_j)\right)}$$

✓ Q为一张查询样本， $\rho(y=i|Q)$为预测Q分类到第i个类别的后验概率

✓ $D(\cdot,\cdot)$为欧式距离，计算两个特征向量之间的距离

实验环境

```
Python 3.9

Cuda-11.6
```

数据集下载

miniImageNet：

https://drive.google.com/drive/u/1/folders/1SEoARH5rADckI-_gZSQRkLclrunL-yb0

[南大云盘 NJU Box]()

原论文效果

数据集：miniInageNet

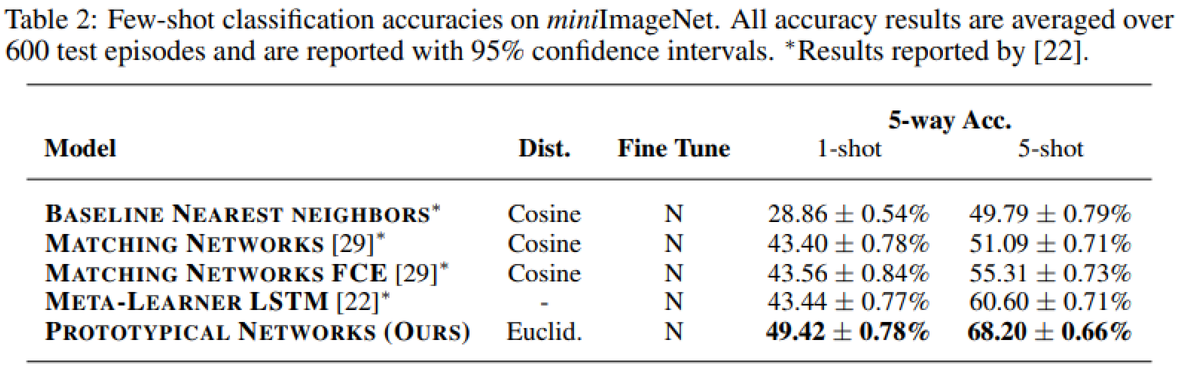


Table 2: Few-shot classification accuracies on *mini*ImageNet. All accuracy results are averaged over 600 test episodes and are reported with 95% confidence intervals. *Results reported by [22].

Model	Dist.	Fine Tune	5-way Acc.	
			1-shot	5-shot
BASELINE NEAREST NEIGHBORS*	Cosine	N	28.86 ± 0.54%	49.79 ± 0.79%
MATCHING NETWORKS [29]*	Cosine	N	43.40 ± 0.78%	51.09 ± 0.71%
MATCHING NETWORKS FCE [29]*	Cosine	N	43.56 ± 0.84%	55.31 ± 0.73%
META-LEARNER LSTM [22]*	-	N	43.44 ± 0.77%	60.60 ± 0.71%
PROTOTYPICAL NETWORKS (OURS)	Euclid.	N	**49.42 ± 0.78%**	**68.20 ± 0.66%**

LibFewShot

安装依赖

https://libfewshot-en.readthedocs.io/en/latest/install.html

```
git clone https://github.com/RL-VIG/LibFewShot.git
cd <path-to-LibFewShot> # cd in `LibFewShot` directory
pip install -r requirements.txt
```

参数设置(示例)

- `epoch`: The number of `epoch` during training.
- `test_epoch`: The number of `epoch` during testing.
- `pretrain_path`: The path of the pre-training weights. At the beginning of the training, this setting will be first checked. If it is not empty, the pre-trained weights of the target path will be loaded into the `backbone` of the current training.
- `resume`: If set to True, the training status is read from the default address to support continual training.
- `way_num`: The number of `way` during training.
- `shot_num`: The number of `shot` during training.
- `query_num`: The number of `query` during training.
- `test_way`: The number of `way` during testing. If not specified, the `way_num` is assigned to the `test_way`.
- `test_shot`: The number of `shot` during testing. If not specified, the `shot_num` is assigned to the `test_way`.
- `test_query`: The number of `query` during testing. If not specified, the `query_num` is assigned to the `test_way`。
- `episode_size`: The number of tasks/episodes used for the network training at each time.
- `batch_size`: The `batch size` used when the `pre-training` model is `pre-trained`. In some kinds of methods, this property is useless.
- `train_episode`: The number of tasks per `epoch` during training.
- `test_episode`: The number of tasks per `epoch` during testing.

```
epoch: 50
test_epoch: 5
```

```
pretrain_path: ~
resume: False

way_num: 5
shot_num: 5
query_num: 15
test_way: ~
test_shot: ~
test_query: ~
episode_size: 1
# batch_size only works in pre-train
batch_size: 128
train_episode: 10000
test_episode: 1000
```

训练模型

1.Modify the config file

本复现任务当中直接使用了`config/proto.yaml`进行修改。

2.Run

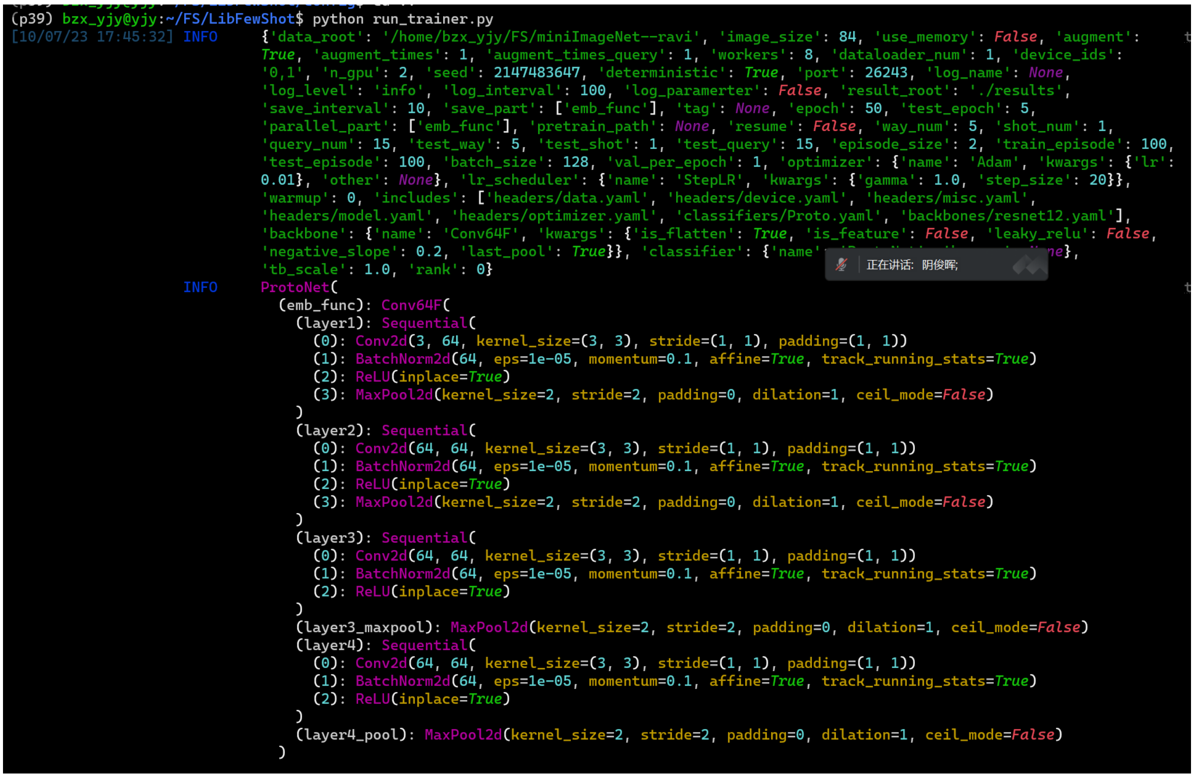
set the `config` as follows in `run_trainer.py`:

```
config = Config("./config/proto.yaml").get_config_dict()
```

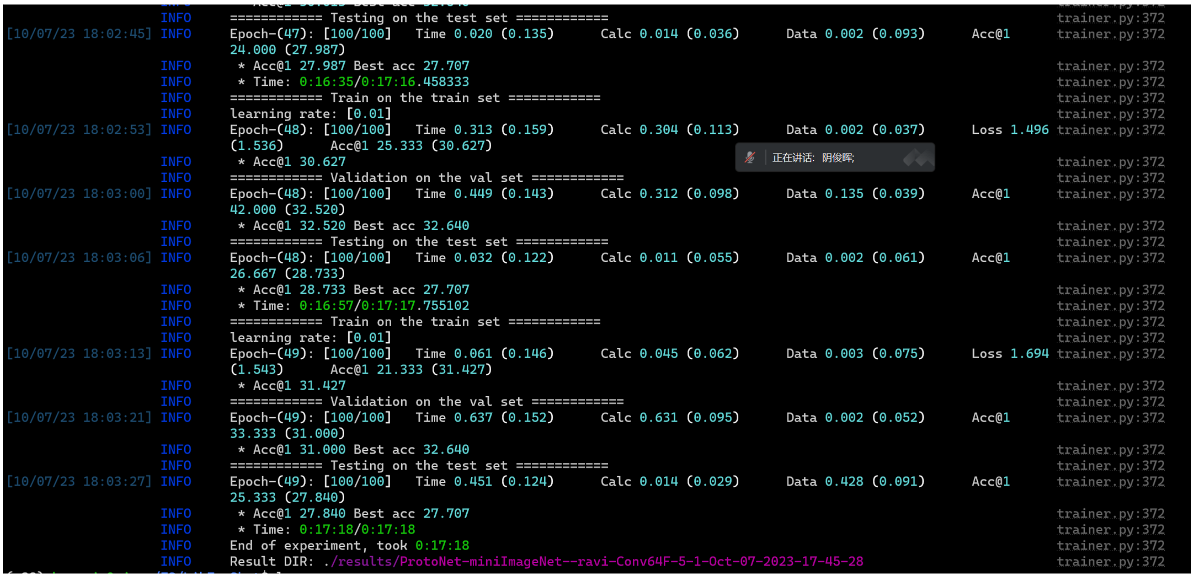
2.train with the console command:

```
python run_trainer.py
```

部分训练过程截图:



```
(p39) bzx_yjy@yjy:~/FS/LibFewShot$ python run_trainer.py
[10/07/23 17:45:32] INFO     {'data_root': '/home/bzx_yjy/FS/miniImageNet--ravi', 'image_size': 84, 'use_memory': False, 'augment':
                             True, 'augment_times': 1, 'augment_times_query': 1, 'workers': 8, 'dataloader_num': 1, 'device_ids':
                             '0,1', 'n_gpu': 2, 'seed': 2147483647, 'deterministic': True, 'port': 26243, 'log_name': None,
                             'log_level': 'info', 'log_interval': 100, 'log_paramerter': False, 'result_root': './results',
                             'save_interval': 10, 'save_part': ['emb_func'], 'tag': None, 'epoch': 50, 'test_epoch': 5,
                             'parallel_part': ['emb_func'], 'pretrain_path': None, 'resume': False, 'way_num': 5, 'shot_num': 1,
                             'query_num': 15, 'test_way': 5, 'test_shot': 1, 'test_query': 15, 'episode_size': 2, 'train_episode': 100,
                             'test_episode': 100, 'batch_size': 128, 'val_per_epoch': 1, 'optimizer': {'name': 'Adam', 'kwargs': {'lr':
                             0.01}, 'other': None}, 'lr_scheduler': {'name': 'StepLR', 'kwargs': {'gamma': 1.0, 'step_size': 20}},
                             'warmup': 0, 'includes': ['headers/data.yaml', 'headers/device.yaml', 'headers/misc.yaml',
                             'headers/model.yaml', 'headers/optimizer.yaml', 'classifiers/Proto.yaml', 'backbones/resnet12.yaml'],
                             'backbone': {'name': 'Conv64F', 'kwargs': {'is_flatten': True, 'is_feature': False, 'leaky_relu': False,
                             'negative_slope': 0.2, 'last_pool': True}}, 'classifier': {'name' [illegible] ne},
                             'tb_scale': 1.0, 'rank': 0}
                    INFO     ProtoNet(
                               (emb_func): Conv64F(
                                 (layer1): Sequential(
                                   (0): Conv2d(3, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
                                   (1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
                                   (2): ReLU(inplace=True)
                                   (3): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
                                 )
                                 (layer2): Sequential(
                                   (0): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
                                   (1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
                                   (2): ReLU(inplace=True)
                                   (3): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
                                 )
                                 (layer3): Sequential(
                                   (0): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
                                   (1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
                                   (2): ReLU(inplace=True)
                                 )
                                 (layer3_maxpool): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
                                 (layer4): Sequential(
                                   (0): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
                                   (1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
                                   (2): ReLU(inplace=True)
                                 )
                                 (layer4_pool): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
                               )
```



```
                    INFO      ============ Testing on the test set ============                                     trainer.py:372
[10/07/23 18:02:45] INFO      Epoch-(47): [100/100]   Time 0.020 (0.135)      Calc 0.014 (0.036)      Data 0.002 (0.093)      Acc@1    trainer.py:372
                              24.000 (27.987)
                    INFO       * Acc@1 27.987 Best acc 27.707                                                       trainer.py:372
                    INFO       * Time: 0:16:35/0:17:16.458333                                                       trainer.py:372
                    INFO      ============ Train on the train set ============                                      trainer.py:372
                    INFO      learning rate: [0.01]                                                                 trainer.py:372
[10/07/23 18:02:53] INFO      Epoch-(48): [100/100]   Time 0.313 (0.159)      Calc 0.304 (0.113)      Data 0.002 (0.037)      Loss 1.496 trainer.py:372
                              (1.536)      Acc@1 25.333 (30.627)
                    INFO       * Acc@1 30.627                                                                       trainer.py:372
                    INFO      ============ Validation on the val set ============                                   trainer.py:372
[10/07/23 18:03:00] INFO      Epoch-(48): [100/100]   Time 0.449 (0.143)      Calc 0.312 (0.098)      Data 0.135 (0.039)      Acc@1    trainer.py:372
                              42.000 (32.520)
                    INFO       * Acc@1 32.520 Best acc 32.640                                                       trainer.py:372
                    INFO      ============ Testing on the test set ============                                     trainer.py:372
[10/07/23 18:03:06] INFO      Epoch-(48): [100/100]   Time 0.032 (0.122)      Calc 0.011 (0.055)      Data 0.002 (0.061)      Acc@1    trainer.py:372
                              26.667 (28.733)
                    INFO       * Acc@1 28.733 Best acc 27.707                                                       trainer.py:372
                    INFO       * Time: 0:16:57/0:17:17.755102                                                       trainer.py:372
                    INFO      ============ Train on the train set ============                                      trainer.py:372
                    INFO      learning rate: [0.01]                                                                 trainer.py:372
[10/07/23 18:03:13] INFO      Epoch-(49): [100/100]   Time 0.061 (0.146)      Calc 0.045 (0.062)      Data 0.003 (0.075)      Loss 1.694 trainer.py:372
                              (1.543)      Acc@1 21.333 (31.427)
                    INFO       * Acc@1 31.427                                                                       trainer.py:372
                    INFO      ============ Validation on the val set ============                                   trainer.py:372
[10/07/23 18:03:21] INFO      Epoch-(49): [100/100]   Time 0.637 (0.152)      Calc 0.631 (0.095)      Data 0.002 (0.052)      Acc@1    trainer.py:372
                              33.333 (31.000)
                    INFO       * Acc@1 31.000 Best acc 32.640                                                       trainer.py:372
                    INFO      ============ Testing on the test set ============                                     trainer.py:372
[10/07/23 18:03:27] INFO      Epoch-(49): [100/100]   Time 0.451 (0.124)      Calc 0.014 (0.029)      Data 0.428 (0.091)      Acc@1    trainer.py:372
                              25.333 (27.840)
                    INFO       * Acc@1 27.840 Best acc 27.707                                                       trainer.py:372
                    INFO       * Time: 0:17:18/0:17:18                                                              trainer.py:372
                    INFO      End of experiment, took 0:17:18                                                       trainer.py:372
                    INFO      Result DIR: ./results/ProtoNet-miniImageNet--ravi-Conv64F-5-1-Oct-07-2023-17-45-28    trainer.py:372
```

测试模型

1.Modify the PATH

修改`run_test.py`中的最优模型路径：

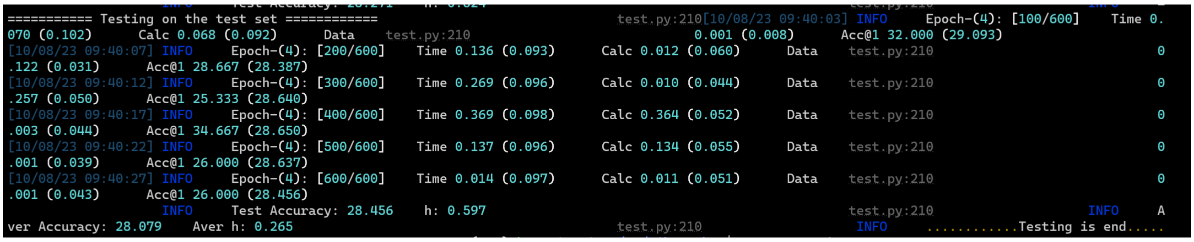
```
PATH = "./results/ProtoNet-miniImageNet--ravi-Conv64F-5-1-Oct-07-2023-17-45-28"
```

2.Run

test with the console command:

```
python run_test.py
```

部分测试过程截图：



```
============ Testing on the test set ============                   test.py:210[10/08/23 09:40:03] INFO      Epoch-(4): [100/600]    Time 0.
070 (0.102)      Calc 0.068 (0.092)      Data    test.py:210                  0.001 (0.008)      Acc@1 32.000 (29.093)
[10/08/23 09:40:07] INFO      Epoch-(4): [200/600]    Time 0.136 (0.093)      Calc 0.012 (0.060)      Data    test.py:210                  0
.122 (0.031)      Acc@1 28.667 (28.387)
[10/08/23 09:40:12] INFO      Epoch-(4): [300/600]    Time 0.269 (0.096)      Calc 0.010 (0.044)      Data    test.py:210                  0
.257 (0.050)      Acc@1 25.333 (28.640)
[10/08/23 09:40:17] INFO      Epoch-(4): [400/600]    Time 0.369 (0.098)      Calc 0.364 (0.052)      Data    test.py:210                  0
.003 (0.044)      Acc@1 34.667 (28.650)
[10/08/23 09:40:22] INFO      Epoch-(4): [500/600]    Time 0.137 (0.096)      Calc 0.134 (0.055)      Data    test.py:210                  0
.001 (0.039)      Acc@1 26.000 (28.637)
[10/08/23 09:40:27] INFO      Epoch-(4): [600/600]    Time 0.014 (0.097)      Calc 0.011 (0.051)      Data    test.py:210                  0
.001 (0.043)      Acc@1 26.000 (28.456)
                    INFO      Test Accuracy: 28.456    h: 0.597                               test.py:210                  INFO      A
ver Accuracy: 28.079    Aver h: 0.265                      test.py:210                  INFO      ............Testing is end.....
```

复现任务

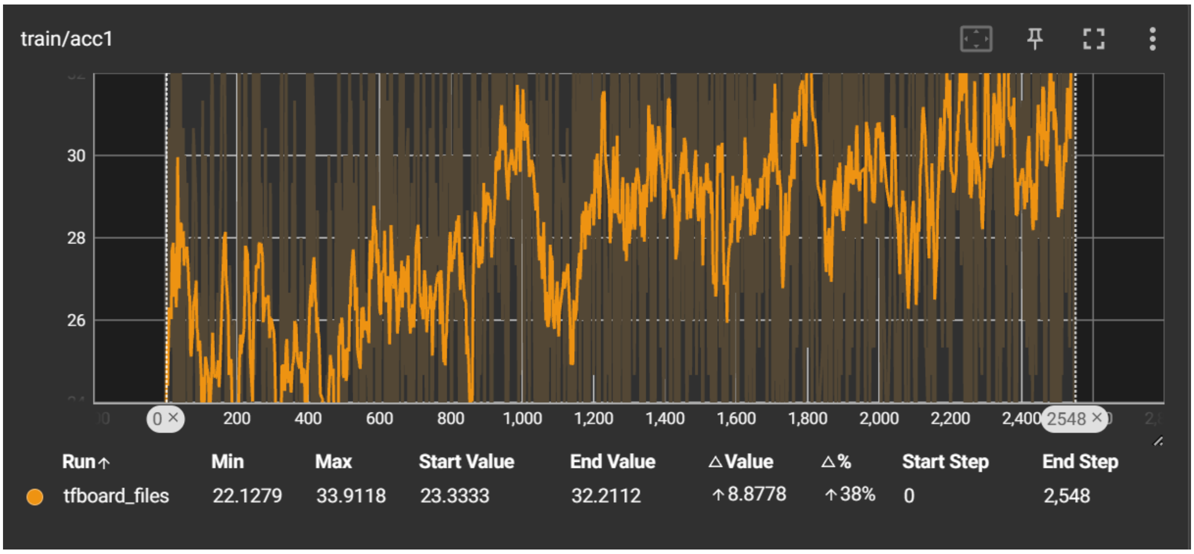
为接近原文效果，本次复现多次调节参数进行训练，主要做了以下几个训练任务：

一.5-way-1-shot（Adam episode_size: 2 train_episode: 100 test_episode: 600）

首先采用了默认的参数进行训练，大概20分钟左右就能完成训练，训练完成后从`results`中查看tensorboard图：

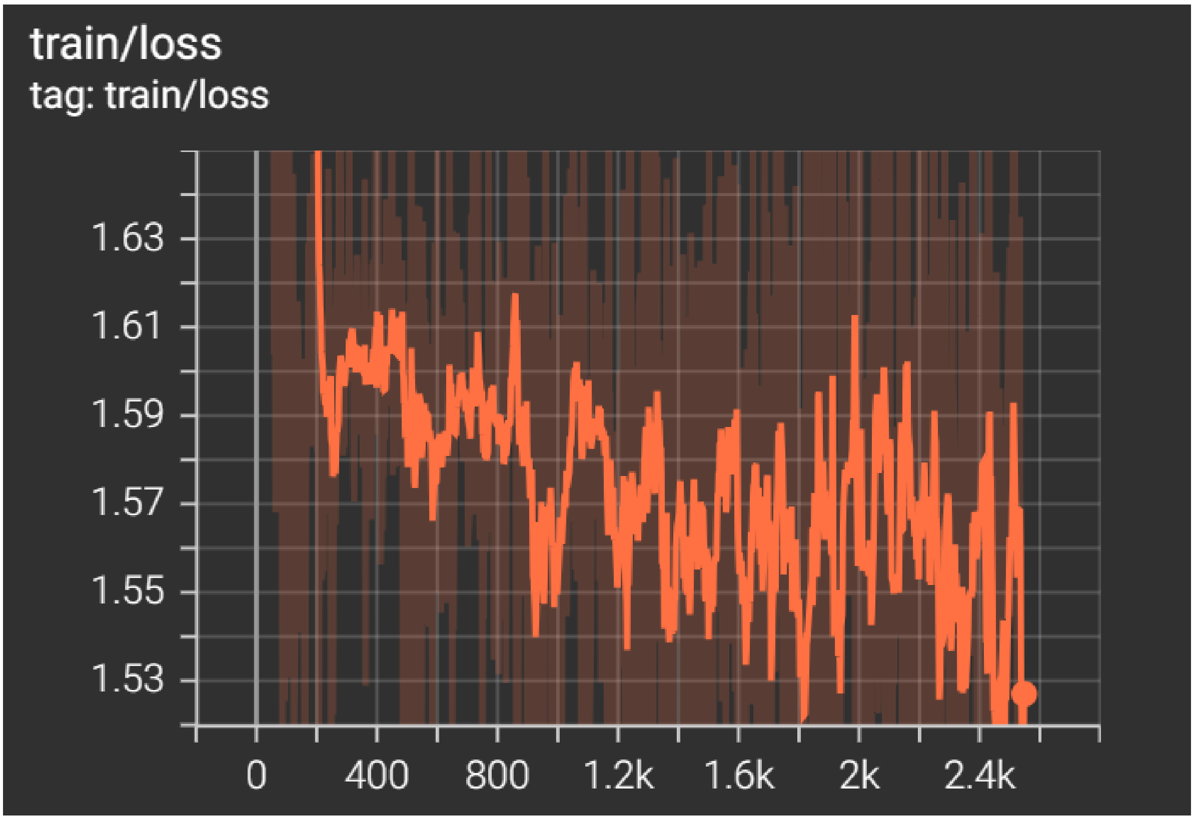
存储位置：`results/ProtoNet-miniImageNet--ravi-Conv64F-5-1-Oct-07-2023-17-45-28`

train acc

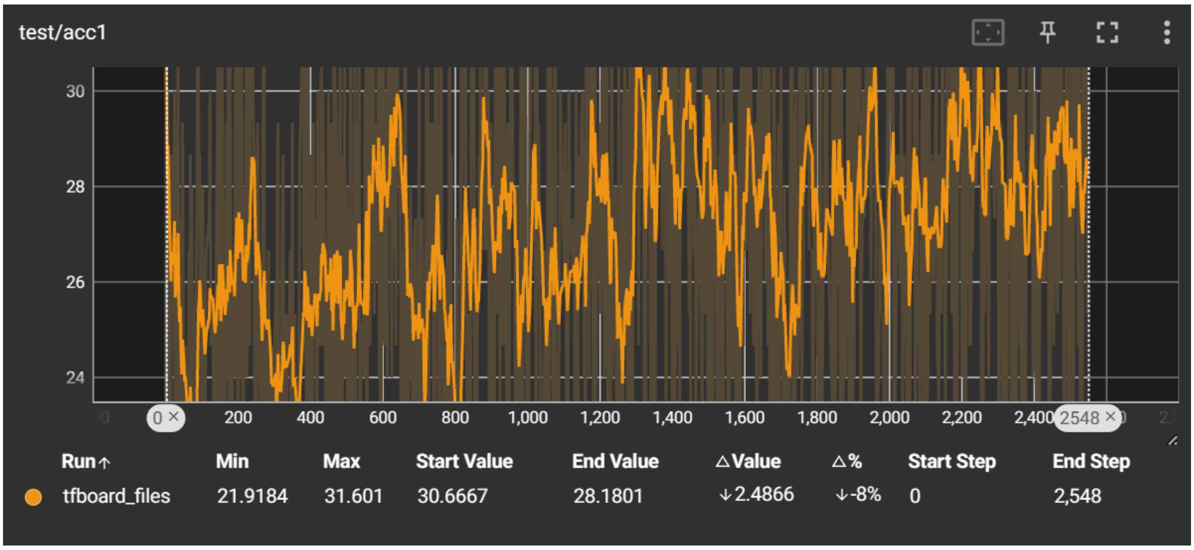


Run↑	Min	Max	Start Value	End Value	△Value	△%	Start Step	End Step
● tfboard_files	22.1279	33.9118	23.3333	32.2112	↑8.8778	↑38%	0	2,548

训练Loss曲线

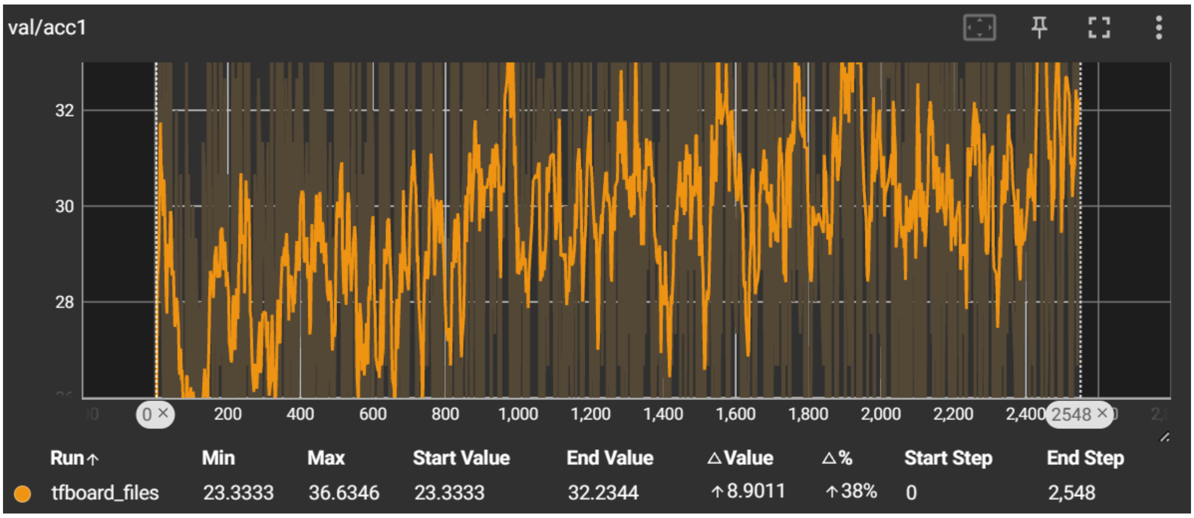


test acc



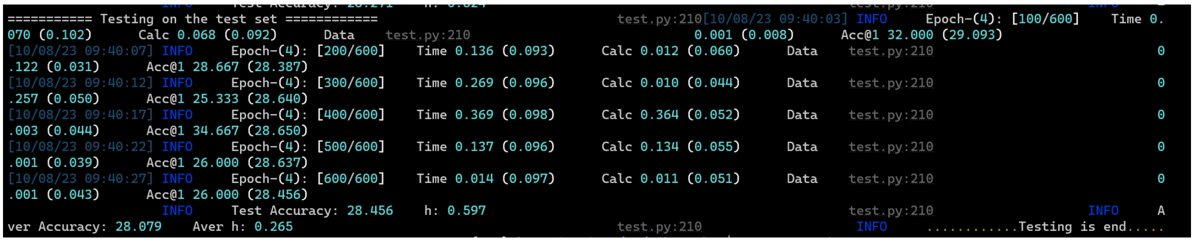
Run↑	Min	Max	Start Value	End Value	△Value	△%	Start Step	End Step
● tfboard_files	21.9184	31.601	30.6667	28.1801	↓2.4866	↓-8%	0	2,548

val acc



Run↑	Min	Max	Start Value	End Value	△Value	△%	Start Step	End Step
● tfboard_files	23.3333	36.6346	23.3333	32.2344	↑8.9011	↑38%	0	2,548

task	本实验精度	参考文献精度
5-way-1-shot	28.079+-0.265	49.42+-0.78



```
=========== Testing on the test set ============                   test.py:210[10/08/23 09:40:03] INFO     Epoch-(4): [100/600]    Time 0.
070 (0.102)      Calc 0.068 (0.092)      Data    test.py:210                     0.001 (0.008)      Acc@1 32.000 (29.093)
[10/08/23 09:40:07] INFO     Epoch-(4): [200/600]    Time 0.136 (0.093)      Calc 0.012 (0.060)      Data    test.py:210                  0
.122 (0.031)      Acc@1 28.667 (28.387)
[10/08/23 09:40:12] INFO     Epoch-(4): [300/600]    Time 0.269 (0.096)      Calc 0.010 (0.044)      Data    test.py:210                  0
.257 (0.050)      Acc@1 25.333 (28.640)
[10/08/23 09:40:17] INFO     Epoch-(4): [400/600]    Time 0.369 (0.098)      Calc 0.364 (0.052)      Data    test.py:210                  0
.003 (0.044)      Acc@1 34.667 (28.650)
[10/08/23 09:40:22] INFO     Epoch-(4): [500/600]    Time 0.137 (0.096)      Calc 0.134 (0.055)      Data    test.py:210                  0
.001 (0.039)      Acc@1 26.000 (28.637)
[10/08/23 09:40:27] INFO     Epoch-(4): [600/600]    Time 0.014 (0.097)      Calc 0.011 (0.051)      Data    test.py:210                  0
.001 (0.043)      Acc@1 26.000 (28.456)
                    INFO     Test Accuracy: 28.456    h: 0.597                                 test.py:210                  INFO     A
ver Accuracy: 28.079    Aver h: 0.265                              test.py:210                  INFO     .............Testing is end.....
```

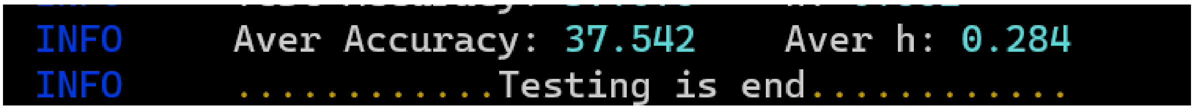
可看出在这样的参数设置下，离原文的实验精度还较远，还做了一个补充实验，将Adam换成了SGD，但结果相差不大。

二.5-way-5-shot（SGD episode_size: 2 train_episode: 100 test_episode: 600）

结果存储地址：`results/ProtoNet-miniImageNet--ravi-Conv64F-5-5-Oct-08-2023-09-21-03`

以相同的参数设置，仅改变了`shot`数量，进行5-way-5-shot的实验，得出的结果如下面的表格所示：

task	本实验精度	参考文献精度
5-way-5-shot	37.542+-0.284	68.20+-0.66

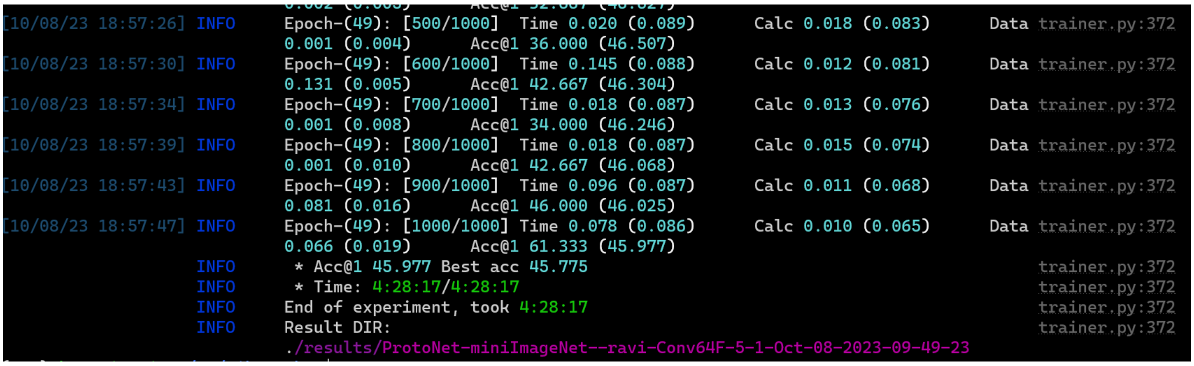


```
INFO     Aver Accuracy: 37.542    Aver h: 0.284
INFO     .............Testing is end.............
```

三.5-way-1-shot(SGD episode_size: 2 train_episode: 10000 test_episode: 600)

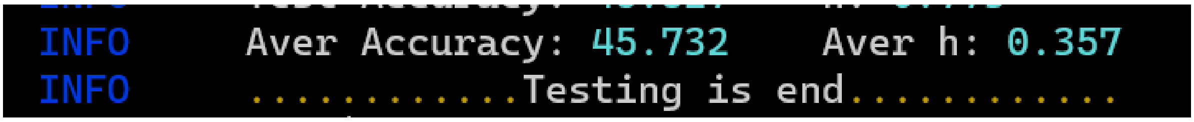
考虑到每个epoch的训练episode数太少，导致实验精度不够，第三四个实验增大了train_episode，本实验训练50轮次所需时间大约为10个小时（两个GPU），

结果和最优模型保存位置为：`./results/ProtoNet-miniImageNet--ravi-Conv64F-5-1-Oct-08-2023-09-49-23`



```
[10/08/23 18:57:26] INFO     Epoch-(49): [500/1000]  Time 0.020 (0.089)      Calc 0.018 (0.083)      Data trainer.py:372
                             0.001 (0.004)       Acc@1 36.000 (46.507)
[10/08/23 18:57:30] INFO     Epoch-(49): [600/1000]  Time 0.145 (0.088)      Calc 0.012 (0.081)      Data trainer.py:372
                             0.131 (0.005)       Acc@1 42.667 (46.304)
[10/08/23 18:57:34] INFO     Epoch-(49): [700/1000]  Time 0.018 (0.087)      Calc 0.013 (0.076)      Data trainer.py:372
                             0.001 (0.008)       Acc@1 34.000 (46.246)
[10/08/23 18:57:39] INFO     Epoch-(49): [800/1000]  Time 0.018 (0.087)      Calc 0.015 (0.074)      Data trainer.py:372
                             0.001 (0.010)       Acc@1 42.667 (46.068)
[10/08/23 18:57:43] INFO     Epoch-(49): [900/1000]  Time 0.096 (0.087)      Calc 0.011 (0.068)      Data trainer.py:372
                             0.081 (0.016)       Acc@1 46.000 (46.025)
[10/08/23 18:57:47] INFO     Epoch-(49): [1000/1000] Time 0.078 (0.086)      Calc 0.010 (0.065)      Data trainer.py:372
                             0.066 (0.019)       Acc@1 61.333 (45.977)
                    INFO      * Acc@1 45.977 Best acc 45.775                                              trainer.py:372
                    INFO      * Time: 4:28:17/4:28:17                                                     trainer.py:372
                    INFO     End of experiment, took 4:28:17                                              trainer.py:372
                    INFO     Result DIR:                                                                  trainer.py:372
                             ./results/ProtoNet-miniImageNet--ravi-Conv64F-5-1-Oct-08-2023-09-49-23
```

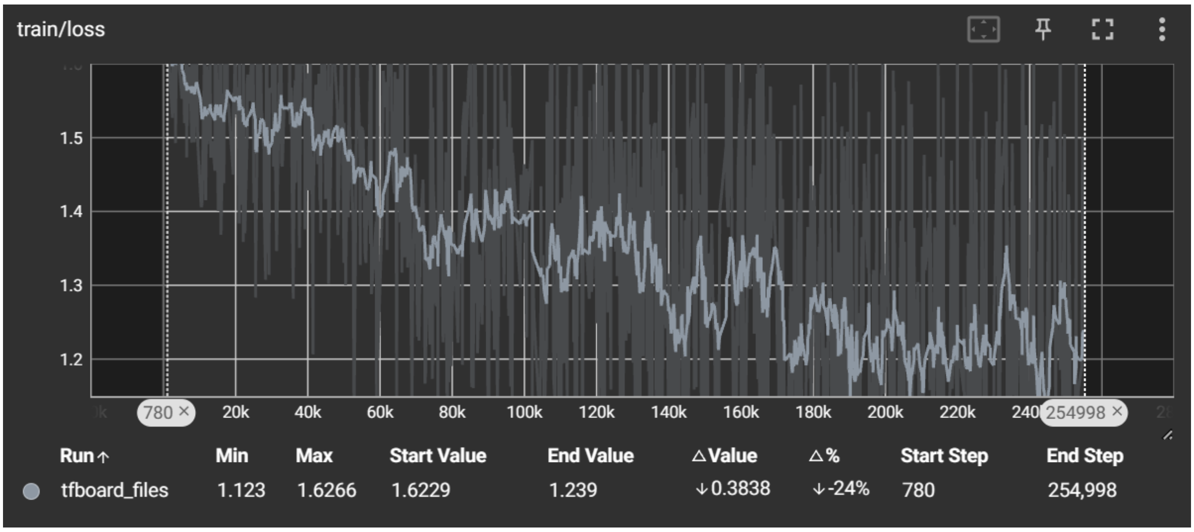
task	本实验精度	参考文献精度
5-way-1-shot	45.732+-0.357	49.42+-0.78



```
INFO      Aver Accuracy: 45.732     Aver h: 0.357
INFO      .............Testing is end.............
```

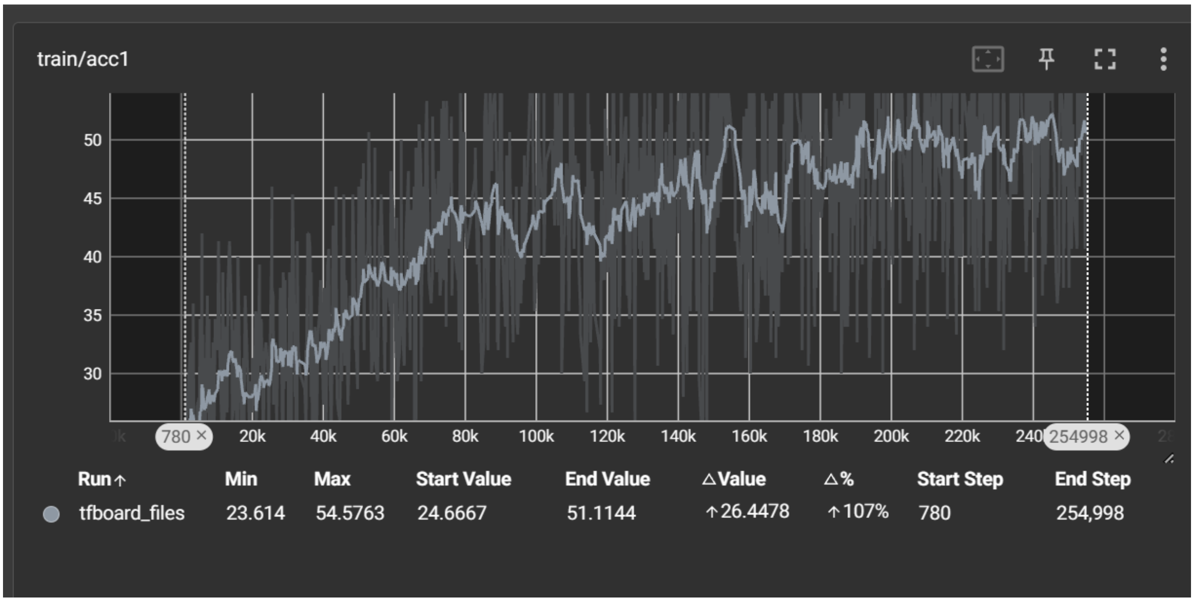
可见本实验精度与原文当中的精度相差不是很大，说明增大训练的episode数是有效的，但是仍有一点差距，由于时间限制，没有再多修改一些参数进行实验。

Loss图



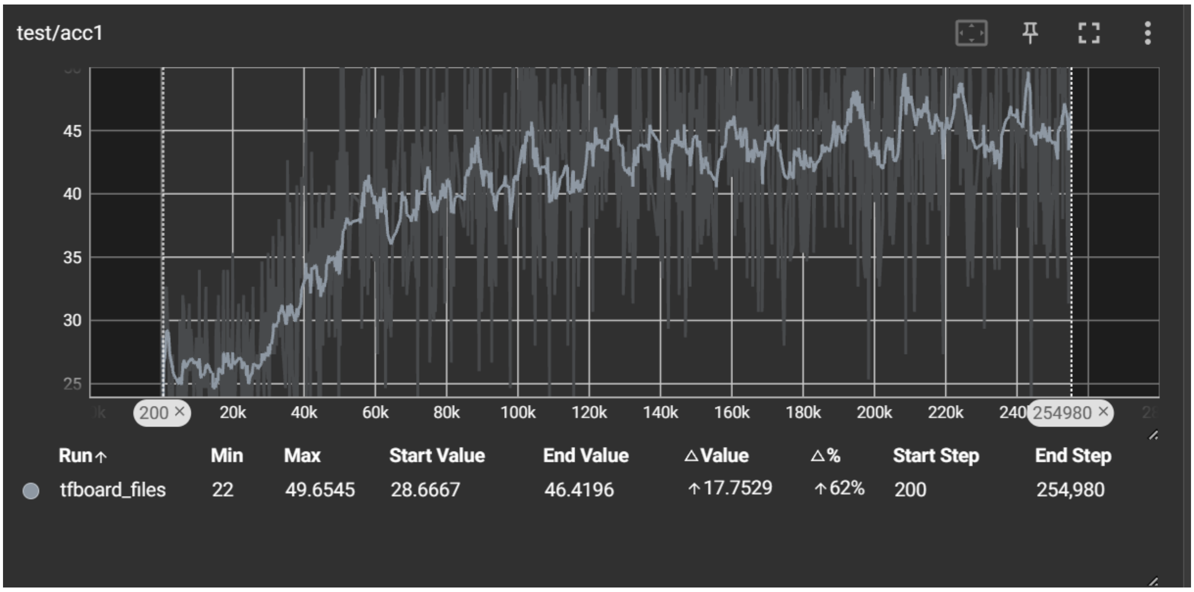
Run↑	Min	Max	Start Value	End Value	△Value	△%	Start Step	End Step
● tfboard_files	1.123	1.6266	1.6229	1.239	↓0.3838	↓-24%	780	254,998

train acc



Run↑	Min	Max	Start Value	End Value	△Value	△%	Start Step	End Step
● tfboard_files	23.614	54.5763	24.6667	51.1144	↑26.4478	↑107%	780	254,998

test acc

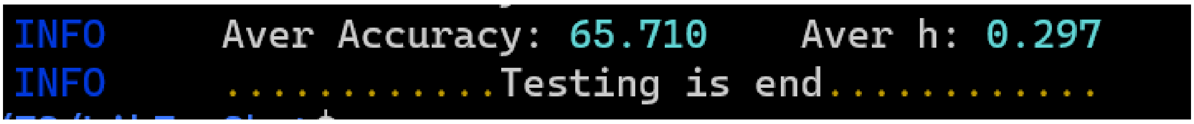


Run↑	Min	Max	Start Value	End Value	△Value	△%	Start Step	End Step
tfboard_files	22	49.6545	28.6667	46.4196	↑17.7529	↑62%	200	254,980

四.5-way-5-shot(episode_size: 2 train_episode: 10000 test_episode: 600)

由于50轮次训练时间过长，该实验的测试结果是训练了30个轮次的最优模型的测试结果，结果和最优模型保存位置为：`results/ProtoNet-miniImageNet--ravi-Conv64F-5-5-Oct-08-2023-14-33-47`

后续将在代码库里更新训练了50个轮次的结果。



```
INFO     Aver Accuracy: 65.710     Aver h: 0.297
INFO     ............Testing is end............
```

task	本实验精度	参考文献精度
5-way-5-shot	65.710+-0.297	68.20+-0.66

其它代码复现

除了LibFewShot中的代码，本次对ProtoNet的复现还使用了其它代码

```
https://github.com/hrdwsong/ProtoNet-Paddle.git
```

环境配置

安装GPU版本的paddlepaddle

```
python3 -m pip install paddlepaddle-gpu==2.5.1.post116 -f
https://www.paddlepaddle.org.cn/whl/linux/cudnnin/stable.html
```

验证是否安装成功

```
Python 3.9.18 | packaged by conda-forge | (main, Aug 30 2023, 03:49:32)
[GCC 12.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import paddle
>>> print(paddle.fluid.core.is_compiled_with_cuda())
True
>>> quit()
```

超参数设置

5-way-1-shot超参数配置如下：

超参数名	设置值
data.way	30
data.shot	1
data.query	15
data.test_way	5
data.test_shot	1
data.test_query	15
lr	0.001

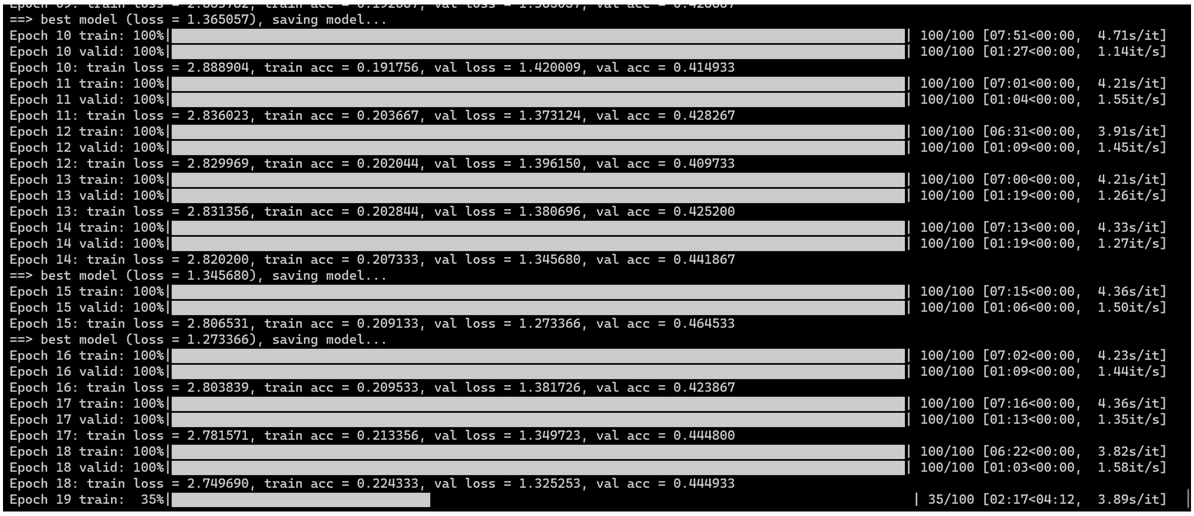
5-way-5-shot超参数配置如下：

超参数名	设置值
data.way	20
data.shot	5
data.query	15
data.test_way	5
data.test_shot	5
data.test_query	15
lr	0.001

训练(以5-way-1-shot为例)

```
python run_train.py --data.way 30 --data.shot 1 --data.query 15 --data.test_way
5 --data.test_shot 1 --data.test_query 15 --data_root ../miniImageNet--ravi
```

训练过程截图：



```
==> best model (loss = 1.365057), saving model...
Epoch 10 train: 100%|████████████████████████████████████████████████████████████| 100/100 [07:51<00:00,  4.71s/it]
Epoch 10 valid: 100%|████████████████████████████████████████████████████████████| 100/100 [01:27<00:00,  1.14it/s]
Epoch 10: train loss = 2.888904, train acc = 0.191756, val loss = 1.420009, val acc = 0.414933
Epoch 11 train: 100%|████████████████████████████████████████████████████████████| 100/100 [07:01<00:00,  4.21s/it]
Epoch 11 valid: 100%|████████████████████████████████████████████████████████████| 100/100 [01:04<00:00,  1.55it/s]
Epoch 11: train loss = 2.836023, train acc = 0.203667, val loss = 1.373124, val acc = 0.428267
Epoch 12 train: 100%|████████████████████████████████████████████████████████████| 100/100 [06:31<00:00,  3.91s/it]
Epoch 12 valid: 100%|████████████████████████████████████████████████████████████| 100/100 [01:09<00:00,  1.45it/s]
Epoch 12: train loss = 2.829969, train acc = 0.202044, val loss = 1.396150, val acc = 0.409733
Epoch 13 train: 100%|████████████████████████████████████████████████████████████| 100/100 [07:00<00:00,  4.21s/it]
Epoch 13 valid: 100%|████████████████████████████████████████████████████████████| 100/100 [01:19<00:00,  1.26it/s]
Epoch 13: train loss = 2.831356, train acc = 0.202844, val loss = 1.380696, val acc = 0.425200
Epoch 14 train: 100%|████████████████████████████████████████████████████████████| 100/100 [07:13<00:00,  4.33s/it]
Epoch 14 valid: 100%|████████████████████████████████████████████████████████████| 100/100 [01:19<00:00,  1.27it/s]
Epoch 14: train loss = 2.820200, train acc = 0.207333, val loss = 1.345680, val acc = 0.441867
==> best model (loss = 1.345680), saving model...
Epoch 15 train: 100%|████████████████████████████████████████████████████████████| 100/100 [07:15<00:00,  4.36s/it]
Epoch 15 valid: 100%|████████████████████████████████████████████████████████████| 100/100 [01:06<00:00,  1.50it/s]
Epoch 15: train loss = 2.806531, train acc = 0.209133, val loss = 1.273366, val acc = 0.464533
==> best model (loss = 1.273366), saving model...
Epoch 16 train: 100%|████████████████████████████████████████████████████████████| 100/100 [07:02<00:00,  4.23s/it]
Epoch 16 valid: 100%|████████████████████████████████████████████████████████████| 100/100 [01:09<00:00,  1.44it/s]
Epoch 16: train loss = 2.803839, train acc = 0.209533, val loss = 1.381726, val acc = 0.423867
Epoch 17 train: 100%|████████████████████████████████████████████████████████████| 100/100 [07:16<00:00,  4.36s/it]
Epoch 17 valid: 100%|████████████████████████████████████████████████████████████| 100/100 [01:13<00:00,  1.35it/s]
Epoch 17: train loss = 2.781571, train acc = 0.213356, val loss = 1.349723, val acc = 0.444800
Epoch 18 train: 100%|████████████████████████████████████████████████████████████| 100/100 [06:22<00:00,  3.82s/it]
Epoch 18 valid: 100%|████████████████████████████████████████████████████████████| 100/100 [01:03<00:00,  1.58it/s]
Epoch 18: train loss = 2.749690, train acc = 0.224333, val loss = 1.325253, val acc = 0.444933
Epoch 19 train:  35%|█████████████████████                                        | 35/100 [02:17<04:12,  3.89s/it]
```

测试

```
python run_eval.py --model.model_path ./results/5w1s/best_model.pdparams --data.test_way 5 --data.test_shot 1 --data.test_query 15 --data_root ../miniImageNet--ravi
```

结果分析

由于训练轮次高达400次，每轮训练较慢，与库中的已有的训练log（同一数据集）对照了前20次的训练结果，训练结果几乎相同，因此直接采用了该库的结果作分析。

task	本项目精度	参考文献精度
5-way-1-shot	50.16+-0.80	49.42+-0.78
5-way-5-shot	68.3+-0.65	68.20+-0.66

参考资料

LibFewShot教程：https://libfewshot-en.readthedocs.io/en/latest/introduction.html

在远程服务器下运行tensorboard，并在本地浏览器下查看

https://www.paddlepaddle.org.cn/install/quick?docurl=/documentation/docs/zh/install/pip/linux-pip.html

https://github.com/hrdwsong/ProtoNet-Paddle/tree/main