

```
(Prototypical-Networks-for-Few-shot-Learning-PyTorch-master) manikantab@wells Fargo:~/ML_Course_project/Prototypical-Networks-for-Few-shot-Learning-PyTorch-master/src$ python train.py --eval omniglot -device cuda:6 -cVa 5 -nsVa 1 -exp ../output/1_shot_5_way
== Dataset: Found 33840 items
== Dataset: Found 1692 classes
----- Loaded model,data -----
=== Epoch: 0 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:08<00:00, 12.42it/s]
0.9854666715860367
=== Epoch: 1 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:07<00:00, 12.71it/s]
0.9827333378791809
=== Epoch: 2 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:08<00:00, 11.87it/s]
0.9812888936201731
=== Epoch: 3 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:06<00:00, 15.52it/s]
0.9811000046133995
=== Epoch: 4 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:06<00:00, 14.40it/s]
0.9813333381414413
=== Epoch: 5 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:08<00:00, 11.26it/s]
0.9807777825991313
=== Epoch: 6 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:08<00:00, 11.83it/s]
0.9817714335237231
=== Epoch: 7 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:10<00:00, 9.53it/s]
0.9817000047117471
=== Epoch: 8 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:08<00:00, 11.12it/s]
0.9819407454464171
=== Epoch: 9 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:09<00:00, 11.06it/s]
0.9820133380889893
Test Acc: 0.9820133380889893
---- Test ---- torch.Size([80, 1, 28, 28]) torch.Size([80])
```

Omniglot (5-shot-5-way) :

Trained on 5 classes*5 examples and tested on 5 classes*15 examples

```
(Prototypical-Networks-for-Few-Shot-Learning-PyTorch-master) manikantab@wellsfargo:~/ML_Course_project/PrototypicalNetworks-  
for-Few-shot-Learning-PyTorch-master/src$ python train.py --eval omniglot -device cuda:6 -cVa 5 -nsVa 5 -exp ../output/5_shot_5_way  
== Dataset: Found 33840 items  
== Dataset: Found 1692 classes  
----- Loaded model,data -----  
=== Epoch: 0 ===  
100%|██████████████████████████████████████████████████████████| 100/100 [00:07<00:00, 12.68it/s]  
0.9932000035047531  
=== Epoch: 1 ===  
100%|██████████████████████████████████████████████████████████| 100/100 [00:06<00:00, 16.63it/s]  
0.99393333361387253  
=== Epoch: 2 ===  
100%|██████████████████████████████████████████████████████████| 100/100 [00:08<00:00, 12.44it/s]  
0.9944444446961085  
=== Epoch: 3 ===  
100%|██████████████████████████████████████████████████████████| 100/100 [00:07<00:00, 12.79it/s]  
0.9945000027120113  
=== Epoch: 4 ===  
100%|██████████████████████████████████████████████████████████| 100/100 [00:07<00:00, 14.16it/s]  
0.994880002617836  
=== Epoch: 5 ===  
100%|██████████████████████████████████████████████████████████| 100/100 [00:08<00:00, 11.86it/s]  
0.9951333359877268  
=== Epoch: 6 ===  
100%|██████████████████████████████████████████████████████████| 100/100 [00:08<00:00, 11.58it/s]  
0.9952571455069951  
=== Epoch: 7 ===  
100%|██████████████████████████████████████████████████████████| 100/100 [00:05<00:00, 17.33it/s]  
0.9954000027477741  
=== Epoch: 8 ===  
100%|██████████████████████████████████████████████████████████| 100/100 [00:08<00:00, 11.55it/s]  
0.9956000027391646  
=== Epoch: 9 ===  
100%|██████████████████████████████████████████████████████████| 100/100 [00:05<00:00, 17.26it/s]  
0.9956400028467178  
Test Acc: 0.9956400028467178  
---- Test ---- torch.Size([100, 1, 28, 28]) torch.Size([100])
```

Omniglot (1-shot-20-way)

Trained on 20 classes*1 examples and tested on 20 classes*15 examples

```
(Prototypical-Networks-for-Few-Shot-Learning-PyTorch-master) manikantab@mellsFargo:~/ML_Course_project/Prototypical-Networks-for-Few-shot-Learning-PyTorch-master/src$ python train.py --eval omniglot --device cuda:6 -cVa 20 -nsVa 1 -exp ../output/1_shot_20_way
== Dataset: Found 33840 items
== Dataset: Found 1692 classes
----- Loaded model,data -----
=== Epoch: 0 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:22<00:00, 4.39it/s]
0.9431999981403351
=== Epoch: 1 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:25<00:00, 3.86it/s]
0.9459999993443489
=== Epoch: 2 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:23<00:00, 4.17it/s]
0.946988888780276
=== Epoch: 3 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:22<00:00, 4.51it/s]
0.9467749990522861
=== Epoch: 4 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:29<00:00, 3.42it/s]
0.9471333326101303
=== Epoch: 5 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:26<00:00, 3.78it/s]
0.9467333328723907
=== Epoch: 6 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:26<00:00, 3.84it/s]
0.9468952377353396
=== Epoch: 7 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:24<00:00, 4.10it/s]
0.9467166662961245
=== Epoch: 8 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:26<00:00, 3.80it/s]
0.9470851848522822
=== Epoch: 9 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:26<00:00, 3.74it/s]
0.9473433329463005
Test Acc: 0.9473433329463005
---- Test ---- torch.Size([320, 1, 28, 28]) torch.Size([320])
```

Omniglot (5-shot-20-way)

Trained on 20 classes*5 examples and tested on 20 classes*15 examples

```
(Prototypical-Networks-for-Few-Shot-Learning-PyTorch-master) manikantab@wellsfargo:~/ML_Course_project/Prototypical-Networks-for-Few-shot-Learning-PyTorch-master/src$ python train.py --eval omniglot --device cuda:6 -cVa 20 -nsVa 5 -exp ../output/5_shot_20_way
== Dataset: Found 33840 items
== Dataset: Found 1692 classes
----- Loaded model,data -----
=== Epoch: 0 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:33<00:00, 2.96it/s]
0.9835333448648452
=== Epoch: 1 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:30<00:00, 3.24it/s]
0.98325001090765
=== Epoch: 2 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:30<00:00, 3.33it/s]
0.9834111217657725
=== Epoch: 3 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:27<00:00, 3.64it/s]
0.9839333441853523
=== Epoch: 4 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:26<00:00, 3.71it/s]
0.9836466773748398
=== Epoch: 5 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:22<00:00, 4.45it/s]
0.9837222332755725
=== Epoch: 6 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:16<00:00, 6.11it/s]
0.9840523917334421
=== Epoch: 7 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:19<00:00, 5.11it/s]
0.9840791773796081
=== Epoch: 8 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:17<00:00, 5.65it/s]
0.9842296403646469
=== Epoch: 9 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:14<00:00, 6.84it/s]
0.9843966774344445
Test Acc: 0.9843966774344445
---- Test ---- torch.Size([400, 1, 28, 28]) torch.Size([400])
```

Trained on 5 classes*5 examples and tested on 5 classes*15 examples

```
(Prototypical-Networks-for-Few-shot-Learning-PyTorch-master) manikantab@wells Fargo:~/ML_Course_project/Prototypical-Networks-for-Few-sho  
t-Learning-PyTorch-master/src$ python train.py --eval -device cuda:6 -cVa 5 -nsVa 5 -exp ../output/5_shot_5_way  
----- Loaded model,data -----  
=== Epoch: 0 ===  
100%|███████████████████████████████████████████████████████████████████████| 100/100 [00:10<00:00, 9.97it/s]  
0.8990666687488555  
=== Epoch: 1 ===  
100%|███████████████████████████████████████████████████████████████████████| 100/100 [00:08<00:00, 11.62it/s]  
0.9001333364844322  
=== Epoch: 2 ===  
100%|███████████████████████████████████████████████████████████████████████| 100/100 [00:09<00:00, 10.33it/s]  
0.8989333365360895  
=== Epoch: 3 ===  
100%|███████████████████████████████████████████████████████████████████████| 100/100 [00:09<00:00, 10.87it/s]  
0.8987000036239624  
=== Epoch: 4 ===  
100%|███████████████████████████████████████████████████████████████████████| 100/100 [00:08<00:00, 12.12it/s]  
0.8988266695737839  
=== Epoch: 5 ===  
100%|███████████████████████████████████████████████████████████████████████| 100/100 [00:08<00:00, 11.71it/s]  
0.9002444471915563  
=== Epoch: 6 ===  
100%|███████████████████████████████████████████████████████████████████████| 100/100 [00:08<00:00, 11.54it/s]  
0.8995428597075599  
=== Epoch: 7 ===  
100%|███████████████████████████████████████████████████████████████████████| 100/100 [00:08<00:00, 11.85it/s]  
0.9000000027567148  
=== Epoch: 8 ===  
100%|███████████████████████████████████████████████████████████████████████| 100/100 [00:08<00:00, 12.09it/s]  
0.8993037064870198  
=== Epoch: 9 ===  
100%|███████████████████████████████████████████████████████████████████████| 100/100 [00:08<00:00, 12.33it/s]  
0.9003066692352295  
Test Acc: 0.9003066692352295  
---- Test ---- torch.Size([100, 1, 28, 28]) torch.Size([100])  
(Prototypical-Networks-for-Few-shot-Learning-PyTorch-master) manikantab@wells Fargo:~/ML_Course_project/Prototypical-Networks-for-Few-sho
```

Trained on 5 classes*1 examples and tested on 5 classes*15 examples

```

==== Loaded model,data =====
=== Epoch: 0 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:09<00:00, 10.54it/s]
0.7486666685342789
=== Epoch: 1 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:10<00:00, 9.51it/s]
0.7514666682481765
=== Epoch: 2 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:08<00:00, 11.74it/s]
0.7503555572032928
=== Epoch: 3 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:08<00:00, 11.84it/s]
0.7493000017851591
=== Epoch: 4 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:07<00:00, 12.61it/s]
0.7473066682815552
=== Epoch: 5 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:09<00:00, 10.37it/s]
0.7447777799268563
=== Epoch: 6 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:08<00:00, 11.46it/s]
0.7454285732337407
=== Epoch: 7 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:08<00:00, 11.64it/s]
0.7450333350151778
=== Epoch: 8 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:08<00:00, 12.22it/s]
0.7460888904333115
=== Epoch: 9 ===
100%|██████████████████████████████████████████████████████████████████████████████| 100/100 [00:09<00:00, 10.95it/s]
0.7452133347094059
Test Acc: 0.7452133347094059
---- Test ---- torch.Size([80, 1, 28, 28]) torch.Size([80])

```

MAML : Omniglot dataset

1-shot-5-way

Trained on 5 classes*1 examples and tested on 5 classes*15 examples

```
step: 39050 training acc: [0.19583333 0.96 0.97333333 0.97333333 0.97333333 0.97333333]
step: 39100 training acc: [0.17791667 0.96208333 0.97791667 0.97875 0.97916667 0.97958333]
step: 39150 training acc: [0.22083333 0.97208333 0.98125 0.98166667 0.98166667 0.98208333]
step: 39200 training acc: [0.19041667 0.97833333 0.98708333 0.9875 0.9875 0.9875]
step: 39250 training acc: [0.22708333 0.97375 0.9875 0.9875 0.98791667 0.98791667]
step: 39300 training acc: [0.22958333 0.97208333 0.9775 0.9775 0.9775 0.9775]
step: 39350 training acc: [0.18625 0.97375 0.97458333 0.97541667 0.97583333 0.97666667]
step: 39400 training acc: [0.22083333 0.9575 0.96333333 0.96708333 0.96791667 0.96833333]
step: 39450 training acc: [0.20083333 0.97083333 0.9725 0.97291667 0.97333333 0.97333333]
step: 39500 training acc: [0.19875 0.96333333 0.97416667 0.97458333 0.97541667 0.97541667]
Test acc: [0.206 0.925 0.937 0.9375 0.9375 0.938 0.9385 0.9385 0.939 0.939]
step: 39550 training acc: [0.14791667 0.98416667 0.98958333 0.99041667 0.99083333 0.99083333]
step: 39600 training acc: [0.16916667 0.97541667 0.97833333 0.97916667 0.97916667 0.97916667]
step: 39650 training acc: [0.18791667 0.98208333 0.99083333 0.99083333 0.99083333 0.99125]
step: 39700 training acc: [0.19016667 0.96333333 0.98333333 0.98333333 0.98333333 0.98333333]
step: 39750 training acc: [0.18583333 0.97625 0.98666667 0.98666667 0.98666667 0.98666667]
step: 39800 training acc: [0.19 0.98458333 0.98958333 0.99 0.99 0.99]
step: 39850 training acc: [0.21208333 0.98166667 0.98583333 0.98583333 0.98583333 0.98583333]
step: 39900 training acc: [0.17875 0.97416667 0.98583333 0.98666667 0.98666667 0.98666667]
step: 39950 training acc: [0.17833333 0.97625 0.97791667 0.97791667 0.97791667 0.97791667]
(MAML-Pytorch) manikantab@WellsFargo:~/virtual_env/MAML-Pytorch$ python omniglot_train.py --n_way 5 --k_spt 1 --save_path models/omniglot/1_shot_5_way]
```

```
(MAML-Pytorch) manikantab@WellsFargo:~/virtual_env/MAML-Pytorch$ python omniglot_train.py --eval omniglot --k_spt 1 --model_path models/omniglot/1_shot_5_way/best_model.pth
Namespace(epoch=40000, eval='omniglot', imgc=1, imgsz=28, k_gry=15, k_spt=1, meta_lr=0.001, model_path='models/omniglot/1_shot_5_way/best_model.pth', n_way=5, save_path='/data4/home/manikantab/virtual_env/MAML-Pytorch/models', task_num=32, update_lr=0.4, update_step=5, update_step_test=10)
Loading Pre-trained model at path == models/omniglot/1_shot_5_way/best_model.pth
load from omniglot.npy.
DB: train (1200, 40, 1, 28, 28) test (423, 40, 1, 28, 28)
----- Test -----
x_spt.shape == torch.Size([32, 5, 1, 28, 28])
y_spt.shape == torch.Size([32, 5])
x_gry.shape == torch.Size([32, 75, 1, 28, 28])
y_gry.shape == torch.Size([32, 75])
step: 0 test acc: [0.2258 0.9307 0.9326 0.9326 0.933 0.933 0.933 0.933 0.933 0.9336]
0.933 ]
step: 5 test acc: [0.2103 0.932 0.9365 0.938 0.9385 0.9385 0.9385 0.939 0.939 0.939]
0.939 ]
step: 10 test acc: [0.2063 0.9307 0.936 0.937 0.9375 0.938 0.938 0.938 0.938 0.9385]
0.9385]
step: 15 test acc: [0.2059 0.933 0.939 0.94 0.9404 0.9404 0.9404 0.941 0.941 0.941]
0.941 ]
step: 20 test acc: [0.2062 0.9316 0.938 0.939 0.9395 0.9395 0.9395 0.94 0.94 0.94]
0.9404]
step: 25 test acc: [0.205 0.9316 0.938 0.939 0.939 0.9395 0.9395 0.9395 0.94 0.94]
0.94 ]
step: 30 test acc: [0.2058 0.932 0.938 0.939 0.9395 0.9395 0.94 0.94 0.94 0.9404]
0.9404]
Test Accuracy === [0.2058 0.932 0.938 0.939 0.9395 0.9395 0.94 0.94 0.94 0.9404]
0.9404]
```

5-shot-5-way

Trained on 5 classes*5 examples and tested on 5 classes*15 examples

```
(MAML-Pytorch) manikantab@wellsfargo:~/virtual_env/MAML-Pytorch$ python omniglot_train.py --eval omniglot --k_spt 5 --n_way 5 --model_path models/omniglot/5_shot_5_way/best_model.pth
Namespace(epoch=40000, eval='omniglot', imgc=1, imgsz=28, k_qry=15, k_spt=5, meta_lr=0.001, model_path='models/omniglot/5_shot_5_way/best_model.pth', n_way=5, save_path='/data4/home/manikantab/virtual_env/MAML-Pytorch/models', task_num=32, update_lr=0.4, update_step=5, update_step_test=10)
Loading Pre-trained model at path == models/omniglot/5_shot_5_way/best_model.pth
load from omniglot.npy.
DB: train (1200, 40, 1, 28, 28) test (423, 40, 1, 28, 28)
----- Test -----
x_spt.shape == torch.Size([32, 25, 1, 28, 28])
y_spt.shape == torch.Size([32, 25])
x_qry.shape == torch.Size([32, 75, 1, 28, 28])
y_qry.shape == torch.Size([32, 75])
step: 0      test acc: [0.2308 0.9814 0.982 0.982 0.982 0.9814 0.9814 0.9814 0.9814 0.9814
0.9814]
step: 5      test acc: [0.2146 0.98 0.9805 0.9805 0.9805 0.981 0.981 0.981 0.981 0.981
0.981 ]
step: 10     test acc: [0.203 0.9814 0.982 0.982 0.982 0.982 0.982 0.982 0.982 0.982
0.982 ]
step: 15     test acc: [0.2013 0.9805 0.981 0.9814 0.9814 0.9814 0.9814 0.9814 0.9814 0.9814
0.9814]
step: 20     test acc: [0.2029 0.981 0.9814 0.982 0.982 0.982 0.982 0.982 0.982 0.982
0.9824]
step: 25     test acc: [0.2002 0.981 0.982 0.9824 0.9824 0.9824 0.9824 0.9824 0.9824 0.9824
0.9824]
step: 30     test acc: [0.2019 0.9814 0.9824 0.9824 0.983 0.983 0.983 0.983 0.983 0.983
0.983 ]
Test Accuracy === [0.2019 0.9814 0.9824 0.9824 0.983 0.983 0.983 0.983 0.983 0.983
0.983 ]
(MAML-Pytorch) manikantab@wellsfargo:~/virtual_env/MAML-Pytorch$
```

1-shot-20-way

Trained on 20 classes*1 examples and tested on 20 classes*15 examples

```
(MAML-Pytorch) manikantab@wellsfargo:~/virtual_env/MAML-Pytorch$ python omniglot_train.py --eval omniglot --k_spt 1 --n_way 20 --model_path models/omniglot/1_shot_20_way/best_model.pth
Namespace(epoch=40000, eval='omniglot', imgc=1, imgsz=28, k_qry=15, k_spt=1, meta_lr=0.001, model_path='models/omniglot/1_shot_20_way/best_model.pth', n_way=20, save_path='/data4/home/manikantab/virtual_env/MAML-Pytorch/models', task_num=32, update_lr=0.4, update_step=5, update_step_test=10)
Loading Pre-trained model at path == models/omniglot/1_shot_20_way/best_model.pth
load from omniglot.npy.
DB: train (1200, 40, 1, 28, 28) test (423, 40, 1, 28, 28)
----- Test -----
x_spt.shape == torch.Size([32, 20, 1, 28, 28])
y_spt.shape == torch.Size([32, 20])
x_qry.shape == torch.Size([32, 300, 1, 28, 28])
y_qry.shape == torch.Size([32, 300])
step: 0      test acc: [0.05176 0.792 0.812 0.812 0.8125 0.8125 0.8125 0.8135 0.8135
0.8135 0.8135 ]
step: 5      test acc: [0.05313 0.8135 0.838 0.8384 0.8384 0.839 0.839 0.8394 0.8394
0.8394 0.8394 ]
step: 10     test acc: [0.05157 0.811 0.8345 0.8354 0.8354 0.836 0.836 0.836 0.836
0.8364 0.8364 ]
step: 15     test acc: [0.0504 0.8076 0.832 0.8325 0.8325 0.833 0.833 0.833 0.8335 0.8335
0.8335 ]
step: 20     test acc: [0.04977 0.808 0.833 0.834 0.834 0.834 0.834 0.8345 0.8345 0.8345
0.835 0.835 ]
step: 25     test acc: [0.05087 0.807 0.8325 0.833 0.8335 0.8335 0.8335 0.834 0.834
0.834 0.834 ]
step: 30     test acc: [0.05127 0.807 0.832 0.833 0.833 0.8335 0.8335 0.8335 0.8335
0.834 0.834 ]
Test Accuracy === [0.05127 0.807 0.832 0.833 0.833 0.8335 0.8335 0.8335 0.8335
0.834 0.834 ]
(MAML-Pytorch) manikantab@wellsfargo:~/virtual_env/MAML-Pytorch$
```


5-shot-20-way

Trained on 20 classes*5 examples and tested on 20 classes*15 examples

```
(MAML-Pytorch) manikantab@WellsFargo:~/virtual_env/MAML-Pytorch$ python omniglot_train.py --eval omniglot --k_spt 5 --n_way
20 --model_path models/omniglot/5_shot_20_way/best_model.pth
Namespace(epoch=40000, eval='omniglot', imgc=1, imgsz=28, k_gry=15, k_spt=5, meta_lr=0.001, model_path='models/omniglot/5 s
hot_20_way/best_model.pth', n_way=20, save_path='/data4/home/manikantab/virtual_env/MAML-Pytorch/models', task_num=32, upda
te_lr=0.4, update_step=5, update_step_test=10)
Loading Pre-trained model at path == models/omniglot/5_shot_20_way/best_model.pth
load from omniglot.npy.
DB: train (1200, 40, 1, 28, 28) test (423, 40, 1, 28, 28)
----- Test -----
x_spt.shape == torch.Size([32, 100, 1, 28, 28])
y_spt.shape == torch.Size([32, 100])
x_gry.shape == torch.Size([32, 300, 1, 28, 28])
y_gry.shape == torch.Size([32, 300])
step: 0      test acc: [0.0546 0.9106 0.9214 0.9233 0.9243 0.9253 0.9253 0.926 0.926 0.9263
0.9263]
step: 5      test acc: [0.05365 0.908 0.9204 0.9224 0.923 0.9233 0.924 0.9243 0.925
0.9253 0.9253 ]
step: 10     test acc: [0.05145 0.911 0.9233 0.9253 0.9263 0.927 0.927 0.9272 0.9277
0.9277 0.928 ]
step: 15     test acc: [0.05127 0.912 0.924 0.9253 0.9263 0.927 0.9272 0.9277 0.9277
0.928 0.928 ]
step: 20     test acc: [0.05054 0.9116 0.923 0.9243 0.9253 0.926 0.9263 0.927 0.927
0.9272 0.9272 ]
step: 25     test acc: [0.0508 0.912 0.923 0.925 0.926 0.9263 0.927 0.927 0.9272 0.9272
0.9277]
step: 30     test acc: [0.05066 0.912 0.9233 0.925 0.926 0.9263 0.927 0.9272 0.9272
0.9277 0.9277 ]
Test Accuracy == [0.05066 0.912 0.9233 0.925 0.926 0.9263 0.927 0.9272 0.9272
0.9277 0.9277 ]
(MAML-Pytorch) manikantab@WellsFargo:~/virtual_env/MAML-Pytorch$
```

Adaptation of Omniglot to Mnist (1-shot-5-way)

Trained on 5 classes*1 examples and tested on 5 classes*15 examples

```
(MAML-Pytorch) manikantab@wellsFargo:~/virtual_env/MAML-Pytorch$ python omniglot_train.py --eval mnist --k_spt 1 --model_path models/omniglot/1_shot_5_way/best_model.pth
Namespace(epoch=40000, eval='mnist', imgc=1, imgsz=28, k_gry=15, k_spt=1, meta_lr=0.001, model_path='models/omniglot/1_shot_5_way/best_model.pth', n_way=5, save_path='/data4/home/manikantab/virtual_env/MAML-Pytorch/models', task_num=32, update_lr=0.4, update_step=5, update_step_test=10)
Loading Pre-trained model at path == models/omniglot/1_shot_5_way/best_model.pth
(5421, 1, 28, 28)
(5923, 1, 28, 28)
(5842, 1, 28, 28)
(6742, 1, 28, 28)
(5949, 1, 28, 28)
(5958, 1, 28, 28)
(6131, 1, 28, 28)
(5918, 1, 28, 28)
(6265, 1, 28, 28)
(5851, 1, 28, 28)
data shape: (10, 5421, 1, 28, 28)
DB: test (10, 5421, 1, 28, 28)
----- Test -----
x_spt.shape == torch.Size([32, 5, 1, 28, 28])
y_spt.shape == torch.Size([32, 5])
x_gry.shape == torch.Size([32, 75, 1, 28, 28])
y_gry.shape == torch.Size([32, 75])
step: 0 test acc: [0.1821 0.5625 0.579 0.5796 0.5806 0.58 0.58 0.5786 0.579 0.5796
0.5806]
step: 5 test acc: [0.1989 0.5547 0.576 0.579 0.58 0.581 0.581 0.5815 0.5825 0.5825
0.583 ]
step: 10 test acc: [0.198 0.546 0.574 0.5767 0.577 0.578 0.5786 0.579 0.5796 0.58
0.58 ]
step: 15 test acc: [0.1971 0.548 0.575 0.578 0.579 0.5796 0.58 0.5806 0.581 0.5815
0.582 ]
step: 20 test acc: [0.1973 0.547 0.5737 0.576 0.577 0.5776 0.5786 0.579 0.579 0.5796
0.58 ]
step: 25 test acc: [0.1975 0.548 0.575 0.577 0.578 0.5786 0.5796 0.58 0.5806 0.5806
0.581 ]
step: 30 test acc: [0.1967 0.548 0.5747 0.5767 0.5776 0.578 0.579 0.5796 0.5796 0.58
0.5806]
Test Accuracy === [0.1967 0.548 0.5747 0.5767 0.5776 0.578 0.579 0.5796 0.5796 0.58
0.5806]
```

Adaptation of Omniglot to Mnist (5-shot-5-way)

Trained on 5 classes*5 examples and tested on 5 classes*15 examples

```
(MAML-Pytorch) manikantab@wellsFargo:~/virtual_env/MAML-Pytorch$ python omniglot_train.py --eval mnist --k_spt 5 --n_way 5
--model_path models/omniglot/5_shot_5_way/best_model.pth
Namespace(epoch=40000, eval='mnist', imgc=1, imgsz=28, k_qry=15, k_spt=5, meta_lr=0.001, model_path='models/omniglot/5_shot_5_way/best_model.pth', n_way=5, save_path='/data4/home/manikantab/virtual_env/MAML-Pytorch/models', task_num=32, update_lr=0.4, update_step=5, update_step_test=10)
Loading Pre-trained model at path == models/omniglot/5_shot_5_way/best_model.pth
(5421, 1, 28, 28)
(5923, 1, 28, 28)
(5842, 1, 28, 28)
(6742, 1, 28, 28)
(5949, 1, 28, 28)
(5958, 1, 28, 28)
(6131, 1, 28, 28)
(5918, 1, 28, 28)
(6265, 1, 28, 28)
(5851, 1, 28, 28)
data shape: (10, 5421, 1, 28, 28)
DB: test (10, 5421, 1, 28, 28)
----- Test -----
x_spt.shape == torch.Size([32, 25, 1, 28, 28])
y_spt.shape == torch.Size([32, 25])
x_qry.shape == torch.Size([32, 75, 1, 28, 28])
y_qry.shape == torch.Size([32, 75])
step: 0      test acc: [0.212  0.7627 0.787  0.788  0.789  0.789  0.7896 0.7896 0.7896 0.79
0.79 ]
step: 5      test acc: [0.2009 0.773  0.7915 0.7925 0.793  0.794  0.7944 0.795  0.7954 0.7954
0.7954]
step: 10     test acc: [0.1997 0.7686 0.786  0.787  0.7876 0.788  0.7886 0.789  0.7896 0.79
0.79 ]
step: 15     test acc: [0.1996 0.769  0.7866 0.7876 0.788  0.7886 0.789  0.7896 0.79  0.79
0.7905]
step: 20     test acc: [0.2    0.7705 0.7876 0.7886 0.789  0.7896 0.79  0.7905 0.791  0.7915
0.7915]
step: 25     test acc: [0.1997 0.769  0.787  0.788  0.789  0.7896 0.79  0.7905 0.7905 0.791
0.7915]
step: 30     test acc: [0.1996 0.77  0.788  0.789  0.7896 0.79  0.7905 0.791  0.7915 0.792
0.792 ]
Test Accuracy == [0.1996 0.77  0.788  0.789  0.7896 0.79  0.7905 0.791  0.7915 0.792
0.792 ]
```

Adaptation Omniglot to Mnist (1-shot-20-way)

Trained on 20 classes*1 examples and tested on 20 classes*15 examples

- Since the MNIST dataset has only 10 classes, it was not possible to evaluate 1-shot 20-way model

Adaptation Omniglot to Mnist (5-shot-20-way)

Trained on 20 classes*5 examples and tested on 20 classes*15 examples

- Since the MNIST dataset has only 10 classes, it was not possible to evaluate 5-shot 20-way model

MAML: MinilImageNet

1_shot_5_way :

- update_step = 5
- update_step_test = 15
- best_model_path

```
Val acc: [0.1912 0.405 0.4097 0.412 0.412 0.4146 0.4148 0.416 0.416 0.417
0.4167 0.4167 0.417 0.4172 0.4182 0.4185]
----- Saving the best model -----

step: 2010    training acc: [0.25666667 0.59 0.6 0.6 0.6 0.6 ]
step: 2040    training acc: [0.19333333 0.41 0.43666667 0.44666667 0.45 0.45333333]
step: 2070    training acc: [0.27666667 0.57333333 0.58 0.58 0.58333333 0.58666667]
step: 2100    training acc: [0.14 0.40333333 0.40666667 0.41666667 0.42333333 0.42666667]
step: 2130    training acc: [0.25 0.63333333 0.64666667 0.65 0.65333333 0.65333333]
step: 2160    training acc: [0.21666667 0.58 0.58 0.58 0.58 0.58666667]
step: 2190    training acc: [0.29333333 0.55 0.54333333 0.53333333 0.53333333 0.54 ]
step: 2220    training acc: [0.24 0.50333333 0.50666667 0.50666667 0.50333333 0.50666667]
step: 2250    training acc: [0.26666667 0.59 0.59333333 0.59 0.59666667 0.59333333]
step: 2280    training acc: [0.16666667 0.51666667 0.53333333 0.53666667 0.53333333 0.52666667]
step: 2310    training acc: [0.25333333 0.56666667 0.58333333 0.58 0.59 0.59333333]
step: 2340    training acc: [0.16333333 0.59333333 0.61 0.61333333 0.60666667 0.61333333]
step: 2370    training acc: [0.24666667 0.6 0.58 0.58 0.57333333 0.57666667]
step: 2400    training acc: [0.13666667 0.51666667 0.51 0.51333333 0.51 0.51333333]
step: 2430    training acc: [0.23 0.60333333 0.62333333 0.62666667 0.62333333 0.62333333]
step: 2460    training acc: [0.18333333 0.48666667 0.51666667 0.52333333 0.53333333 0.53333333]
step: 2490    training acc: [0.13333333 0.57666667 0.56666667 0.57333333 0.57333333 0.57666667]
Test acc: [0.2107 0.4348 0.4348 0.4353 0.4353 0.438 0.4385 0.4385 0.439 0.44
0.4397 0.4397 0.4404 0.4404 0.4404 0.4402]
```

5_shot_5_way :

python miniiimagenet_train.py --device cuda:7 --update_step 5 --update_step_test 15 --k_spt 5 --
save_path models/5_shot_5_way

```
Val acc: [0.2139 0.5396 0.553 0.561 0.5625 0.5625 0.5625 0.564 0.5645 0.5635
0.564 0.5645 0.564 0.565 0.565 0.565 ]
----- Saving the best model -----

step: 2010    training acc: [0.20333333 0.63666667 0.63333333 0.67 0.68333333 0.68333333]
step: 2040    training acc: [0.21333333 0.76333333 0.79333333 0.79333333 0.78 0.78666667]
step: 2070    training acc: [0.11666667 0.69333333 0.71 0.72666667 0.73333333 0.74 ]
step: 2100    training acc: [0.24333333 0.69 0.72333333 0.73 0.72333333 0.72333333]
step: 2130    training acc: [0.22 0.64666667 0.64333333 0.65 0.64666667 0.65 ]
step: 2160    training acc: [0.18333333 0.71333333 0.74333333 0.74666667 0.75666667 0.76333333]
step: 2190    training acc: [0.18666667 0.63666667 0.63 0.64666667 0.64333333 0.64 ]
step: 2220    training acc: [0.15 0.67333333 0.69333333 0.71 0.71666667 0.72 ]
step: 2250    training acc: [0.25333333 0.66 0.71666667 0.71 0.71 0.71666667]
step: 2280    training acc: [0.19333333 0.63666667 0.67333333 0.71 0.71666667 0.72 ]
step: 2310    training acc: [0.19 0.68333333 0.72333333 0.71666667 0.72 0.73333333]
step: 2340    training acc: [0.14333333 0.75333333 0.74333333 0.76333333 0.75333333 0.75333333]
step: 2370    training acc: [0.14666667 0.75 0.76333333 0.77 0.76333333 0.77 ]
step: 2400    training acc: [0.22 0.70666667 0.70333333 0.71666667 0.70333333 0.70333333]
step: 2430    training acc: [0.21 0.73333333 0.75666667 0.76333333 0.77333333 0.77333333]
step: 2460    training acc: [0.17333333 0.68666667 0.69666667 0.72 0.72 0.72666667]
step: 2490    training acc: [0.15666667 0.7 0.71666667 0.75333333 0.75333333 0.75333333]
Test acc: [0.2064 0.5625 0.58 0.588 0.5884 0.5913 0.5923 0.5938 0.5938 0.594
0.5947 0.5947 0.594 0.5947 0.595 0.5957]
```

1_shot_20_way :

python miniimagenet_train.py --device cuda:7 --update_step 5 --update_step_test 15 --k_spt 1 --n_way 20 --save_path models/1_shot_20_way

■ Val set not considered(it has only 16 classes)

```
Val acc: [0.04764 0.1458 0.1637 0.167 0.1664 0.167 0.1671 0.1674 0.1672
0.167 0.1671 0.1672 0.1675 0.1675 0.1674 0.1674 ]
----- Saving the best model -----

step: 2010      training acc: [0.0475      0.21666667 0.27833333 0.29166667 0.2925      0.2925      ]
step: 2040      training acc: [0.04      0.27      0.31083333 0.30666667 0.3125      0.305      ]
step: 2070      training acc: [0.04916667 0.275      0.29583333 0.295      0.29666667 0.29916667]
step: 2100      training acc: [0.03583333 0.2525      0.28666667 0.2775      0.27666667 0.27666667]
step: 2130      training acc: [0.03666667 0.2275      0.26666667 0.27083333 0.2675      0.27333333]
step: 2160      training acc: [0.05833333 0.23416667 0.275      0.28583333 0.28666667 0.28833333]
step: 2190      training acc: [0.05416667 0.2925      0.32583333 0.32833333 0.325      0.3225      ]
step: 2220      training acc: [0.04916667 0.25916667 0.29583333 0.31333333 0.3125      0.31      ]
step: 2250      training acc: [0.0575      0.28916667 0.31      0.31166667 0.31166667 0.30916667]
step: 2280      training acc: [0.06      0.24583333 0.27      0.2725      0.27666667 0.2775      ]
step: 2310      training acc: [0.03083333 0.22166667 0.27416667 0.28416667 0.2875      0.285      ]
step: 2340      training acc: [0.05166667 0.26166667 0.31666667 0.32916667 0.32833333 0.33083333]
step: 2370      training acc: [0.04      0.2825      0.33833333 0.345      0.3475      0.34333333]
step: 2400      training acc: [0.05916667 0.2475      0.30083333 0.30083333 0.30666667 0.2975      ]
step: 2430      training acc: [0.05083333 0.2725      0.31416667 0.3225      0.315      0.31      ]
step: 2460      training acc: [0.05333333 0.25416667 0.26916667 0.28      0.28      0.29      ]
step: 2490      training acc: [0.065      0.28083333 0.3275      0.3225      0.31916667 0.32333333]
Test acc: [0.05045 0.1433 0.1603 0.1647 0.165 0.1649 0.1644 0.1648 0.1649
0.165 0.1649 0.1652 0.1649 0.165 0.1653 0.1654 ]
```

5_shot_20_way :

python miniimagenet_train.py --device cuda:7 --update_step 5 --update_step_test 15 --k_spt 5 --n_way 20 --save_path models/5_shot_20_way

- Val set not considered(it has only 16 classes)

```
Val acc: [0.0472 0.1808 0.24    0.2477 0.2578 0.2615 0.2622 0.263  0.2627 0.263
0.2632 0.2632 0.2632 0.2637 0.2634 0.2634]
----- Saving the best model -----

step: 2010      training acc: [0.0475    0.37916667 0.45    0.465    0.485    0.49083333]
step: 2040      training acc: [0.07416667 0.33333333 0.44583333 0.45916667 0.47833333 0.4775    ]
step: 2070      training acc: [0.04416667 0.355    0.44833333 0.46916667 0.48333333 0.48333333]
step: 2100      training acc: [0.01416667 0.35083333 0.445    0.45916667 0.47416667 0.48    ]
step: 2130      training acc: [0.03166667 0.36833333 0.4275    0.46666667 0.46416667 0.46416667]
step: 2160      training acc: [0.04916667 0.36083333 0.43666667 0.46166667 0.47166667 0.4975    ]
step: 2190      training acc: [0.04333333 0.36833333 0.45083333 0.475    0.47    0.49083333]
step: 2220      training acc: [0.04833333 0.37    0.46416667 0.49416667 0.495    0.51416667]
step: 2250      training acc: [0.06583333 0.37916667 0.465    0.47416667 0.4825    0.49166667]
step: 2280      training acc: [0.04333333 0.36166667 0.46333333 0.49833333 0.4975    0.50666667]
step: 2310      training acc: [0.05333333 0.32583333 0.40666667 0.4375    0.445    0.455    ]
step: 2340      training acc: [0.05666667 0.3625    0.43    0.46833333 0.49916667 0.515    ]
step: 2370      training acc: [0.0625    0.295    0.41    0.43083333 0.44    0.44583333]
step: 2400      training acc: [0.05916667 0.33416667 0.41416667 0.42583333 0.45583333 0.4725    ]
step: 2430      training acc: [0.06166667 0.34416667 0.43666667 0.46666667 0.46916667 0.47416667]
step: 2460      training acc: [0.01916667 0.36333333 0.4675    0.49916667 0.50916667 0.5125    ]
step: 2490      training acc: [0.0625    0.33416667 0.42666667 0.43916667 0.46    0.46166667]
Test acc: [0.0474 0.1825 0.2369 0.2512 0.2605 0.2605 0.2637 0.264 0.264 0.264
0.2646 0.2644 0.2642 0.2644 0.2646 0.2646]
```