DLCV 2021 Fall Hw4 r09942091 謝承軒

Problem 1.

1. **acc**: 44.15 +- 0.83%
   epochs: 80, **each epoch**: 300 batches (80*300 episodes)
   distance function: euclidean, **optimizer**: Adam, **lr:** 1e-4
   data augmentation: RandomResizedCrop, RandomHorizontalFlip, RandomGrayScale
   meta-train: 25-way 1-shot, **meta-val**: 5-way 1-shot
   query: 15 samples / batch

```
Convnet(
  (encoder): Sequential(
    (0): 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()
      (3): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    )
    (1): 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()
      (3): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    )
    (2): 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()
      (3): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    )
    (3): 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()
      (3): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    )
  )
  (MLP): MLP(
    (mlp): Sequential(
      (0): Linear(in_features=1600, out_features=400, bias=True)
      (1): Dropout(p=0.5, inplace=False)
      (2): ReLU()
      (3): Linear(in_features=400, out_features=64, bias=True)
    )
  )
```

2. for parametric distance method:
   I **expand** the **outputs** of the support set and query set to the same shape, and **concatenate** them together.
   It'll get the shape (**support num, query num, 64+64**)
   Then do **MLP** to get the similarity of the instances.

```
(distance_nn): MLP(
  (mlp): Sequential(
    (0): Linear(in_features=128, out_features=32, bias=True)
    (1): Dropout(p=0.5, inplace=False)
    (2): ReLU()
    (3): Linear(in_features=32, out_features=1, bias=True)
  )
)
```

   meta-train: 5-way 1-shot
   meta-test: 5-way 1-shot

distance method	valid acc.
euclidean	43.16 +- 0.82%
cosine	19.88 +- 0.31%
parametric	39.51 +- 0.73%

   I didn't get a good performance in the parametric method, but it was good at my sample method (random sample 300 batches). One possibility is that my sample method is not good enough. The other possibility may be the

3. lr: 1e-4 / epochs: 80 / episodes: 80*300

meta-train/test	distance method	valid acc.
5-way 1-shot	euclidean	43.22
5-way 5-shot	euclidean	63.27
5-way 10-shot	euclidean	67.19

   In my problem 3, the valid samples are based on my sampler.

My 5-way 5/10-shot seems to have a high acc. at the valid set, however, they're not perform well at meta-test in 5-way 1-shot. Can't even reach 40% accuracy.

It seems reasonable to get a higher acc. with more shots in both training and testing with more samples for the support center.

Problem 2.

1. When pretraining the SSL model, I used the default BYOL method to preatrined the resnet50(img size=128).
   I only used the RandomHorizontalFlip(p=20) in augmentation, others are wrapped in BYOL .
   Optimizer: Adam(lr=3e-4)
   Batch: 64
   Batches: 40 (about 24hr)

   Of course, without labels.

2.

setting	Pre-training	Fine-tuning	valid acc.
A	-	full model	37.19
B	TA's	full model	47.29
C	My SSL	full model	40.15
D	TA's	classifier only	24.88
E	MySSL	classifier only	18.97

3. In my problem 2, the setting B gets the highest accuracy of the validation set, however, the loss is high. So it may be a little bit of

overfitting in the office dataset.
The setting D also got a high loss.
The setting A and C have similar loss, but with the pretrained model, the setting C had higher accuracy(3%). So the pretrained model seems helpful.