Training Tutorial：

This document will use the VGG network as an example to introduce the training process of icCNN for single-category and multi-category classification tasks. （https://github.com/ada-shen/icCNN）

Dataset

After downloading the code, modify the corresponding address of the dataset in **vgg_ori_train.py**, **vgg_iccnn_train.py** and **vgg_iccnn_multi_train.py**, respectively, as shown below:

```
cub_file = '/data/sw/dataset/frac_dataset'
voc_file = '/data/sw/dataset/VOCdevkit/VOC2010/voc2010_crop'
celeb_file = '/home/user05/fjq/dataset/CelebA/'
```

where **cub_file**, **voc_file**, **celeb_file** correspond to the CUB, PascalVOC and CelebA datasets, respectively.

Training step

Single-category classification

1. Train the original model

```
python3 train_all.py -type ori -is_multi 0 -model vgg
```

You can use the above command to train the original model

The model will be trained according to the specific parameters in **vgg_ori_train.py**, which are shown below:

```
IS_TRAIN = 0          # 0/1
IS_MULTI = 0
LAYERS = '13'
DATANAME = 'bird'
NUM_CLASSES = 6 if IS_MULTI else 2
if DATANAME == 'celeb':
    NUM_CLASSES = 80
```

where for single-category classification task training, let **IS_TRAIN = 1**, **IS_MULTI = 0**

In addition, the parameters **BATCHSIZE**, **LR**, and **EPOCH** required for network training are set at the following locations:

```
BATCHSIZE = 1
LR = 0.000001
EPOCH = 200
```

The model file will be saved in the following location

```
log_path = '/data/fjq/iccnn/vgg/' # for model
```

2. Generate feature maps of the original model

```
python3 train_all.py -type ori -is_multi 0 -model vgg
```

Let **IS_TRAIN = 0** and **IS_MULTI = 0** in **vgg_ori_train.py**, and input the above command to generate the feature map of the original VGG model using the model on the **pretrain_model** address shown below.

```
log_path = log_path + dataset + '/'
pretrain_model = log_path + 'model_2000.pth'
```

Its obtained feature map and the accuracy will be saved to the following location:

3. Train the corresponding ICCNN model

After completing the training of the original model in Step 1, the address of the original model of the ICCNN model used and corresponding to the training is written to the following location in the **vgg_iccnn_train.py** code: `pretrain_model = log_path + 'model_2000.pth'`

Input the following command to train the corresponding ICCNN model

```
python3 train_all.py -type iccnn -is_multi 0 -model vgg
```

The network parameters need to be written in advance in **vgg_iccnn_train.py** at the same location as the corresponding location in **vgg_ori_train.py** and let **IS_TRAIN = 1**.

In addition, when training the ICCNN model, there are additional parameters as shown below

```
center_num = 5
lam = 0.1
T = 2 # T = 2 ===> do sc each epoch
F_MAP_SIZE = 196
STOP_CLUSTERING = 200
if LAYERS == '13':
    CHANNEL_NUM = 512
elif LAYERS == '16':
    CHANNEL_NUM = 512
```

where **center_num** denotes the number of convolutional kernel groups, **lam** denotes the weight before similarity loss, **T** denotes the T round for one clustering, **STOP_CLUSTERING** denotes the epoch position to stop clustering, **F_MAP_SIZE** denotes the size of the feature map, **CHANNEL_NUM** denotes the number of channels of the feature map

Multi-category classification

1. Train the original model

```
python3 train_all.py -type ori -is_multi 1 -model vgg
```

You can use the above command to train the original model

The model will be trained according to the specific parameters in **vgg_ori_train.py**, which are shown below:

```
IS_TRAIN = 0          # 0/1
IS_MULTI = 0
LAYERS = '13'
DATANAME = 'bird'
NUM_CLASSES = 6 if IS_MULTI else 2
if DATANAME == 'celeb':
    NUM_CLASSES = 80
```

where for multi-category classification task training, make **IS_TRAIN = 1**, **IS_MULTI = 1**

In addition, the parameters **BATCHSIZE**,**LR**, **EPOCH**, and the model storage address required for network training are located in the same location as the single-category classification task in the code **vgg_ori_train.py**

2. Generate feature maps of the original model

```
python3 train_all.py -type ori -is_multi 1 -model vgg
```

Let **IS_TRAIN = 0** and **IS_MULTI = 1** in **vgg_ori_train.py**, and input the above command to generate the feature map of the original VGG model using the model on the **pretrain_model** address shown below.

```
log_path = log_path + dataset + '/'
pretrain_model = log_path + 'model_2000.pth'
```

Its obtained feature map and the accuracy will be saved to the following location:

```
save_path = '/data/fjq/iccnn/basic_fmap/vgg/'  # for get_feature
acc_path = '/data/fjq/iccnn/basic_fmap/vgg/acc/'
```

3. Train the corresponding ICCNN model

After completing the training of the original model in Step 1, the address of the original model of the ICCNN model used and corresponding to the training is written to the following location in the **vgg_iccnn_multi_train.py** code:

```
pretrain_model = log_path + 'model_2000.pth'
```

Input the following command to train the corresponding ICCNN model

```
python3 train_all.py -type iccnn -is_multi 1 -model vgg
```

The network parameters need to be written in advance in **vgg_iccnn_multi_train.py** at the same location as the corresponding location in **vgg_ori_train.py**, and let **IS_TRAIN = 1**.

In addition, when training the ICCNN model, there are additional parameters as shown below

```
center_num = 16
lam1 = 0.1
lam2 = 0.1
T = 2 # T = 2 ===> do sc each epoch
F_MAP_SIZE = 196
STOP_CLUSTERING = 200
if LAYERS == '13':
    CHANNEL_NUM = 512
elif LAYERS == '16':
    CHANNEL_NUM = 512
elif LAYERS == '19':
    CHANNEL_NUM = 512
```

where **center_num** denotes the number of convolutional kernel groups, **lam1** denotes the weight before the similarity loss function and **lam2** denotes the weight before the multi-category loss function, **T** denotes the T round for one clustering, **STOP_CLUSTERING** denotes the epoch position to stop clustering, **F_MAP_SIZE** denotes the size of the feature map, **CHANNEL_NUM** denotes the number of channels of the feature map.