Machine Learning HW12 Report

學號：B06901045 系級：電機三 姓名：曹林熹

1. **請描述你實作的模型架構、方法以及 accuracy 為何。其中你的方法必須為 domain adversarial training 系列（就是你的方法必須要讓輸入 training data & testing data 後的某一層輸出 domain 要相近）。(2%)**

ANS：

我使用的方法主要可以分為兩大部分，第一部分為訓練我的 DANN 模型，第二部分為我用 DANN 使用 semi-supervised 再重新 train 一次 CNN。

第一部分）

改的地方為：

1. Dataloader batch_size

   會這樣改是因為想說每一次 step 能夠考慮的資料數比較多，所以讓 sample code 的 batch_size = 32 改成 batch_size = 64。

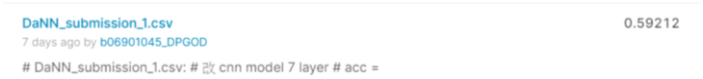
2. DANN model

   這邊是我主要改的部分，label_predictor、domain_classifier 都跟 sample code 一樣，feature_extractor 的話我把 convolution layer 改成 7 層，以利於得到比較好的 feature（最後出來是 1*1），但是我們的 image input 是 32*32 的 image，如果我每一層都使用 maxpooling 的話會不夠用，因此我在前期時單純用 convolution 並沒有 pooling，到第三層才開始用 pooling，最後可以得到我們的特徵圖。

Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 64, 32, 32]	1,664
BatchNorm2d-2	[-1, 64, 32, 32]	128
ReLU-3	[-1, 64, 32, 32]	0
Conv2d-4	[-1, 128, 32, 32]	73,856
BatchNorm2d-5	[-1, 128, 32, 32]	256
ReLU-6	[-1, 128, 32, 32]	0
Conv2d-7	[-1, 256, 32, 32]	295,168
BatchNorm2d-8	[-1, 256, 32, 32]	512
ReLU-9	[-1, 256, 32, 32]	0
MaxPool2d-10	[-1, 256, 16, 16]	0
Conv2d-11	[-1, 256, 16, 16]	590,080
BatchNorm2d-12	[-1, 256, 16, 16]	512
ReLU-13	[-1, 256, 16, 16]	0
MaxPool2d-14	[-1, 256, 8, 8]	0
Conv2d-15	[-1, 512, 8, 8]	1,180,160
BatchNorm2d-16	[-1, 512, 8, 8]	1,024
ReLU-17	[-1, 512, 8, 8]	0
MaxPool2d-18	[-1, 512, 4, 4]	0
Conv2d-19	[-1, 512, 4, 4]	2,359,808
BatchNorm2d-20	[-1, 512, 4, 4]	1,024
ReLU-21	[-1, 512, 4, 4]	0
MaxPool2d-22	[-1, 512, 2, 2]	0
Conv2d-23	[-1, 512, 2, 2]	2,359,808
BatchNorm2d-24	[-1, 512, 2, 2]	1,024
ReLU-25	[-1, 512, 2, 2]	0
MaxPool2d-26	[-1, 512, 1, 1]	0

Total params: 6,865,024
Trainable params: 6,865,024
Non-trainable params: 0

Training epoch 就是 300，得到第一部份的 DANN 後拿去 kaggle，可以得到 0.59212 acc。

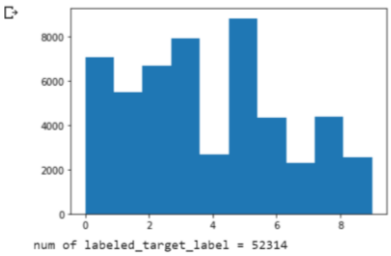


第二部分）

因為準確率還不足，因此我採用 semi-supervised 方式重新 train 一個模型，可以細分為幾個步驟。

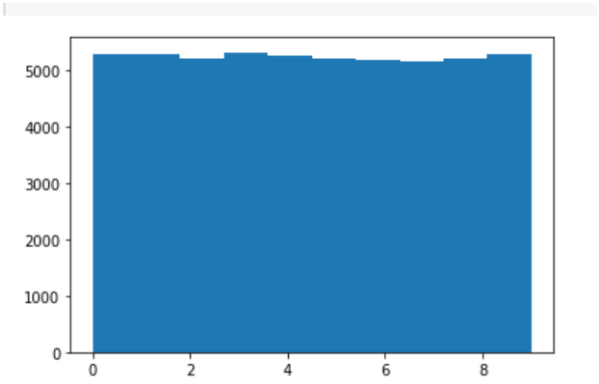
1. pseudo label

   我拿我第一部份的 DANN 去對 target data 做 pseudo label，這時候我使用 softmax 函數去取得我最大的 label，這時設定 threshold =0.999，要大於此閾值我才拿做 semi_train_data。可以看到我們得到的資料共有 52314 筆，但是資料分布相當不平衡。



2. 平衡資料

   因為 training data 資料不平衡的話，我們預測 testing data 時就有高機率都會集中在特定的 class，因此我將資料平衡處理，新的 semi_train_data_loader batch_size = 256，每一種類別資料約有 5000 筆，共 50000 筆左右。



3. CNN model

我拿建立好的 semi data 訓練我們 hw3 的 model，模型部分就是建立一個 CNN。convolution layer 有五層，flatten 的部分有用兩次 Dropout2d，是個簡單的模型。

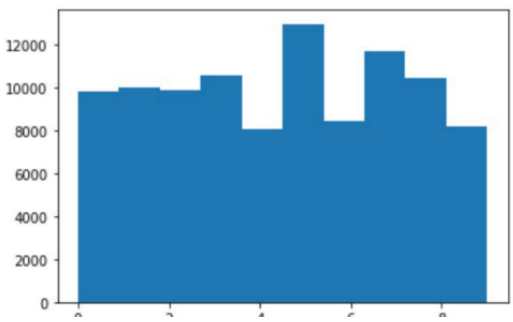
```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1          [-1, 64, 32, 32]             640
       BatchNorm2d-2          [-1, 64, 32, 32]             128
              ReLU-3          [-1, 64, 32, 32]               0
         MaxPool2d-4          [-1, 64, 16, 16]               0
            Conv2d-5         [-1, 128, 16, 16]          73,856
       BatchNorm2d-6         [-1, 128, 16, 16]             256
              ReLU-7         [-1, 128, 16, 16]               0
         MaxPool2d-8           [-1, 128, 8, 8]               0
            Conv2d-9           [-1, 256, 8, 8]         295,168
      BatchNorm2d-10           [-1, 256, 8, 8]             512
             ReLU-11           [-1, 256, 8, 8]               0
        MaxPool2d-12           [-1, 256, 4, 4]               0
           Conv2d-13           [-1, 512, 4, 4]       1,180,160
      BatchNorm2d-14           [-1, 512, 4, 4]           1,024
        MaxPool2d-15           [-1, 512, 2, 2]               0
             ReLU-16           [-1, 512, 2, 2]               0
           Conv2d-17           [-1, 512, 2, 2]       2,359,808
      BatchNorm2d-18           [-1, 512, 2, 2]           1,024
        MaxPool2d-19           [-1, 512, 1, 1]               0
             ReLU-20           [-1, 512, 1, 1]               0
           Linear-21                 [-1, 256]         131,328
             ReLU-22                 [-1, 256]               0
          Dropout2d-23               [-1, 256]               0
           Linear-24                 [-1, 128]          32,896
             ReLU-25                 [-1, 128]               0
          Dropout2d-26               [-1, 128]               0
           Linear-27                  [-1, 10]           1,290
================================================================
Total params: 4,078,090
Trainable params: 4,078,090
Non-trainable params: 0
```

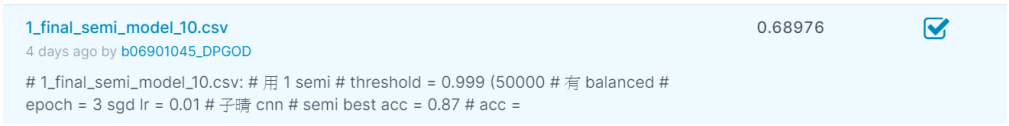
4. training

optimizer 使用 SGD（lr = 0.01、momentum = 0.02），不使用 Adam 是因為經過幾次訓練好發現 SGD 比較好。

Epoch = 3，我發現 epoch 不用訓練太多，很快 acc 就到 90% 左右，為了以防 overfitting 我只 train 三次，train_acc = 0.8734

最後，拿此模型去對我們的 testing 預測，可以看到資料分布有比較平衡了，拿去 kaggle 可以得到 0.68976 成績，比原來多了 10% acc。





2. **請視覺化真實圖片以及手繪圖片通過沒有使用 domain adversarial training 的 feature extractor 的 domain 分布圖。(2%)**

ANS：

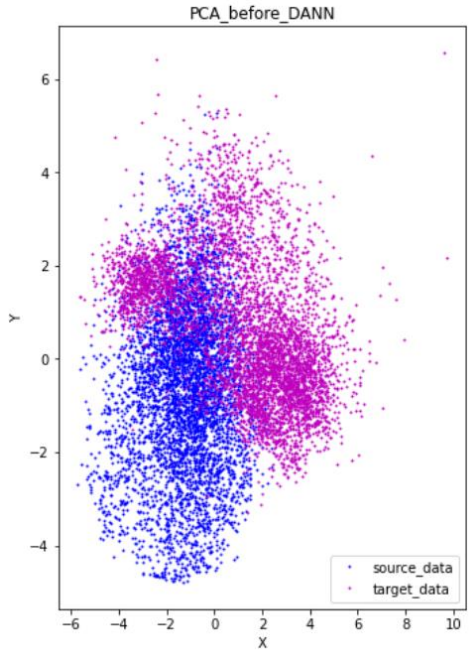
這邊很簡單，我就單純使用 PCA 降維法去對原本的 (1, 32, 32) image 做降維，PCA 模型是拿

```
pca_total = PCA(n_components=2, random_state = 0).fit(source_data + target_data)
```

然後降維的話再各別使用

```
source_projected = pca_total.transform(source_data)
target_projected = pca_total.transform(targe_data)
```

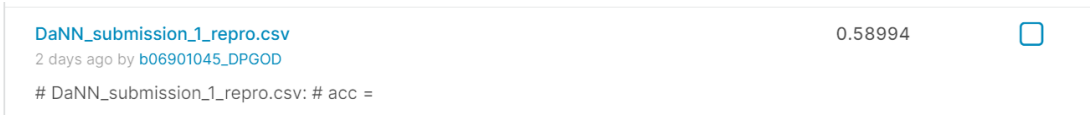
source_data 有 5000 筆，target_data 5056 筆。可以看到少部分點有重疊，但是在外圍部分明顯被散開來。



3. **請視覺化真實圖片以及手繪圖片通過有使用 domain adversarial training 的 feature extractor 的 domain 分布圖。(2%)**

ANS：

這裡我使用的 DANN model 為我在 kaggle 上不使用 semi 前直接預測的 model，那他在 kaggle 上的成績會有 0.58994，此架構也就是在第一題使用的 DANN 模型。



接下來我取得 source_data 與 target_data 經過 feature_extractor 後的 feacture，這些 feature 最後每一個 item 是 (512, 1, 1) 的大小，意即最後是 512 層 1*1 的特徵圖。這邊我的 source_data 有 5000 筆，target_data 有 5056 筆。

降維的話，我一樣採用上題使用的 PCA 降維法，可以發現經過 feature_extractor 後的 feature 在 PCA 降維圖幾乎都混一起，是個成功的結果。