HW3

1. **Supervised learning**:

一開始是直接將 labeled data (RGB image)拿去 train，test data set predict 的結果，accuracy 在 public set 為 0.38 而在 private set 為 0.39，但是若將 image 先做灰階後再丟去 train，結果在 public set 為 0.46 而在 private set 為 0.47，提升了不少，CNN 的架構由下圖程式碼表示 (使用灰階 image)，loss function 採用 category crossentropy，而 optimizer 使用 adam。

```
model=Sequential()
model.add(Convolution2D(128,5,5,activation='relu',border_mode='same',input_shape=(1,32,32)))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.2))
model.add(Convolution2D(64,5,5,activation='relu',border_mode='same'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Convolution2D(64,3,3,activation='relu',border_mode='same'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(output_dim=64,init='normal',W_regularizer=l2(0.03)))
model.add(Activation('relu'))
model.add(Dropout(0.5))
model.add(Dense(output_dim=10,W_regularizer=l2(0.01)))
model.add(Activation('linear'))
model.add(Dropout(0.2))
model.add(Dense(nclass,init='normal'))
model.add(Activation('softmax'))
```

2. **Semi-supervised learning (1)**

第一個方法使用的是 self-training，先 train labeled data 得到 model，再來拿這 model 去 predict unlabeled data 得到的每筆 unlabeled data 屬於各個 class 的機率並計算 entropy，將 entropy 小於 0.8 的 unlabeled data 進行 label 後與原本 labeled data 一起再丟回 model 去 train，得到新的 model 後再拿剩下的 unlabeled data (entropy>0.8) 去 predict，若 predict 出來的結果 entropy 小於 0.8 的 unlabeled data 不到 2000 筆，則此 model 為最終 model，若大於 2000 筆則繼續重覆此流程。

3. **Semi-supervised learning (2)**

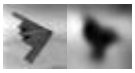
第二個方法使用的是 cluster by autoencoder，下圖是 convolutional autoencoder 的結構，loss function 採用 category crossentropy，而 optimizer 使用 adam。

```
input_img = Input(shape=(1, 32, 32))

x = Convolution2D(64, 5, 5, activation='relu',border_mode='valid')(input_img)
x = Convolution2D(64, 5, 5, activation='relu', border_mode='valid')(x)
x = MaxPooling2D((2, 2))(x)
x = Convolution2D(64, 5, 5, activation='relu', border_mode='valid')(x)
x = MaxPooling2D((2, 2))(x)
x = Flatten()(x)
x = Dense(output_dim = 512,activation = 'relu')(x)
encoded = Dense(output_dim=256,activation='relu')(x)

x = Dense(output_dim=512 , activation = 'relu')(encoded)
x = Dense(output_dim = 1024,activation = 'relu')(x)
x = Reshape((64, 8, 8))(x)
x = UpSampling2D((2, 2))(x)
x = Deconvolution2D(64, 5, 5, output_shape=(None, 64, 12, 12), activation='relu', border_mode='valid')(x)
x = UpSampling2D((2, 2))(x)
x = Deconvolution2D(64, 5, 5, output_shape=(None, 64, 28, 28), activation='relu', border_mode='valid')(x)
x = Deconvolution2D(64, 5, 5, output_shape=(None, 64, 32, 32), activation='relu', border_mode='valid')(x)
decoded = Convolution2D(1, 3, 3, activation='sigmoid', border_mode='same')(x)
```

將 labeled data 與 unlabeled data 共 50000 筆 data，先做完灰階化 (維度變成(1,32,32))，再都丟進此 mode 進行 training，但由於實在 train 得太慢所以 loss 值到達 0.5772 時我就將其暫停，而此時 predict 的結果大概由下面兩張圖所示，第一張為 input 的 image，而第二張為 predict 的 image，可見還相差許多，但 predict 的 image 中物件的輪廓已相當明顯。



再來，將所有 data 做 encode 時，得到長度為 256 的 code，發現每筆 data 的 code 中大約有 1/2 為 0，這實在非常奇妙，所以我再將所有 data 的 code 用 PCA 做降維成長度為 128 的 code，並將所有 data 依照這 128 去做 kmeans clustering，而 n_cluster 設為 10，執行後發現，5000 筆 labeled data 有絕大部分都被分在同一群，之後又再試試 Spectral Clustering，發現 class 5 的 label data 在三個群組裡，相較於其他 class 的 label data，是出現最多次的，由此可見 autoencoder 的 training 還不夠完美，使得 encode 的 code 之識別能力還不夠強。由於最後無法完美的 clustering，所以第二個方法沒有做完，所以沒有上傳 method2 的 code。

4. Compare and analyze your results

在 self-training 方法中，直接使用 RGB image 去做的話，得到的 model 於 test data set predict 的結果，accuracy 在 public set 為 0.5 而在 private set 也為 0.5，但若以灰階的 image 做的話，public set 為 0.46 而 private set 也為 0.46，反而沒有 improve 原本 supervised 的 model，可能是 entropy 的 threshold 設太小了，使沒有足夠的 unlabeled data 來 improve model。而 github 上傳的為使用 RGB 的 code。