機器學習與深度學習導論期末報告

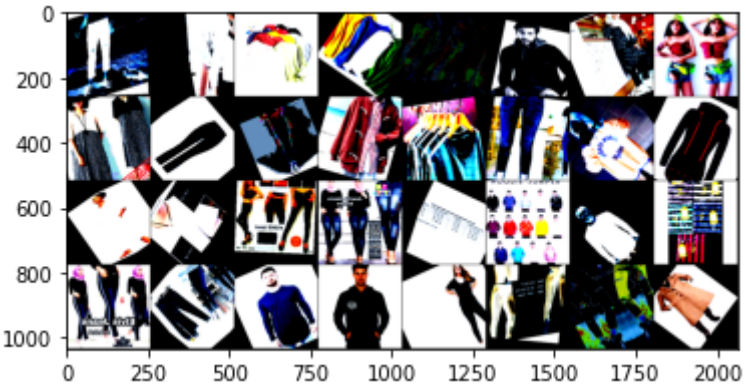
b06502027 羅恩至 r10521525 蕭仲綱

1. 前置作業

首先將 training data 和 testing data 下載到 colab 上。training data 一共9279張，其中我們從中隨機抽取8279張用於training，1000張用於validation。我們是使用pytorch框架來做訓練。在pytorch中需要先做出一個關於dataset的class，在這個class中有三個方法，分別是初始化、取得資料(檔名和label)和回傳長度(圖片數量)，之後再將這個包好的dataset用DataLoader下載下來。

2. Data augmentation

將資料集中的圖片，統一大小為256*256、RGB格式，並進行平移、水平翻轉、旋轉、縮放、改變明度對比度等方式來擴增資料集容量，並將所有圖片轉換為tesnor的形式儲存，且對於每張圖片均有做normalization處理。而根據以上的這些處理方式，依照不同的參數(如：進行水平翻轉的機率、旋轉的角度、縮放比例等)，總共設置了5種變化的組合。資料集的容量經過處理後會放大五倍，所以最後用來訓練的圖片數量共有 8279*5 = 41395 張，用來validation的圖片則有5000張。



圖(一) 進行Data augmentation後的圖片

3. Pretrained model

一開始曾試著使用手刻的CNN模型訓練，但是因為資料量較大，每個epoch即需要大量的訓練時間，而因免費版的colab有每日使用時間限制、且不時有斷線風險，因此無法訓練過久，故無法有效提高訓練準確度，實測下來20epoch的accuracy約0.57左右。

因為Pytorch網站上附有一些預訓練過的模型，因此後來都是嘗試使用不同的pretrain model，將之下載下來後，稍加修改輸出的feature數(1000改4)來做測試。嘗試過的模型包括Resnet、Densenet、Efficientnet、Googlenet、Squeezenet、Ghostnet等等，至於預測結果以Efficientnet的準確度最佳(稍微訓練幾個epoch即可通過strong baseline)、Densenet次之，GoogleNet則是private score遠比public score好上許多(約3%左右)。

表1 各種模型的測試結果(public score/private score)

Model	acc(1種Data augmentation)	acc(5種Data augmentation)	acc(pseudo labeling)
densenet121	0.74357/0.75000	0.82000/0.82642	0.83357/0.82214
resnet50	未測試	0.80357/0.80714	0.79428/0.81000
mobilenet_v2	未測試	0.81285/0.81285	未測試
efficientnet_b3 (efficientnet_b4)	未測試	0.85000/0.84285	0.84357/**0.85928**
GoogleNet	未測試	0.80785/0.83071	未測試
SqueezeNet	未測試	0.71071/0.74928	未測試

4. Pseudo labeling

除了有label的圖片外，助教也有提供5277張沒有標籤的資料(其中一張毀損故有5276張)。這次期末專題也有嘗試過使用semi supervising的技巧，若模型的validation準確度夠高，變先對這些圖片做預測，當結果具有極高機率都預測為某類標籤時(設定0.98或0.99)，便把這些圖片丟進訓練資料中，再行訓練。然而在實作pseudo labeling時，雖然可以稍微提高準確度(約0.83~0.84)，但colab記憶體也會十分容易爆炸，無法訓練太多的epoch，因此最後幾次訓練時並未使用pseudo label。

不過值得一提的是，在kaggle比賽截止公布private score成績後，發現有一組使用efficeintnet_b4加上pseudo labeling的private score高達0.85928，但是因為public score不高(0.84357)所以未勾選此次結果，實為可惜。但也為此得知，若是模型訓練到一定的精準程度的話，使用pseudo labeling確實能夠提高整體的testing準確度。

5. Training hyperparameters

在超參數的使用方面，Optimizer使用Adam、learning rate0.0001、weight decay 10e-05，epoch數訂為1~5左右，1個epoch的訓練時間依pretrained的模型種類及圖片數量、大小而有所不同，大部分落在約10~40分鐘左右。

6. Ensemble

後來也有嘗試使用pytorch的torchensemble套件中的VotingClassifier來訓練，在此只要設定estimator及epoch的值，便能對不同的模型進行訓練。雖然訓練時間會較久，1個epoch約1~2小時(依estimator的數量而定)，但此舉可以稍微提升準確度(public score 0.85357,private score 0.85500)。

而最後也有另外撰寫了一個小程式，將幾個public score最高的訓練結果進行投票(用取眾數的方式)，可以稍微提高準確度(但若已經使用torchensemble套件訓練的話，則因為模型已經有做過voting，所以效果不是很好)。

表2 進行Ensemble後的測試結果(模型均使用efficientnet_b4)

method	public score	private score
torchensemble	0.85357	0.85500
torchensemble + voting	0.85357	0.85500
voting(public score 0.84以上的結果做投票)	0.85285	0.85357