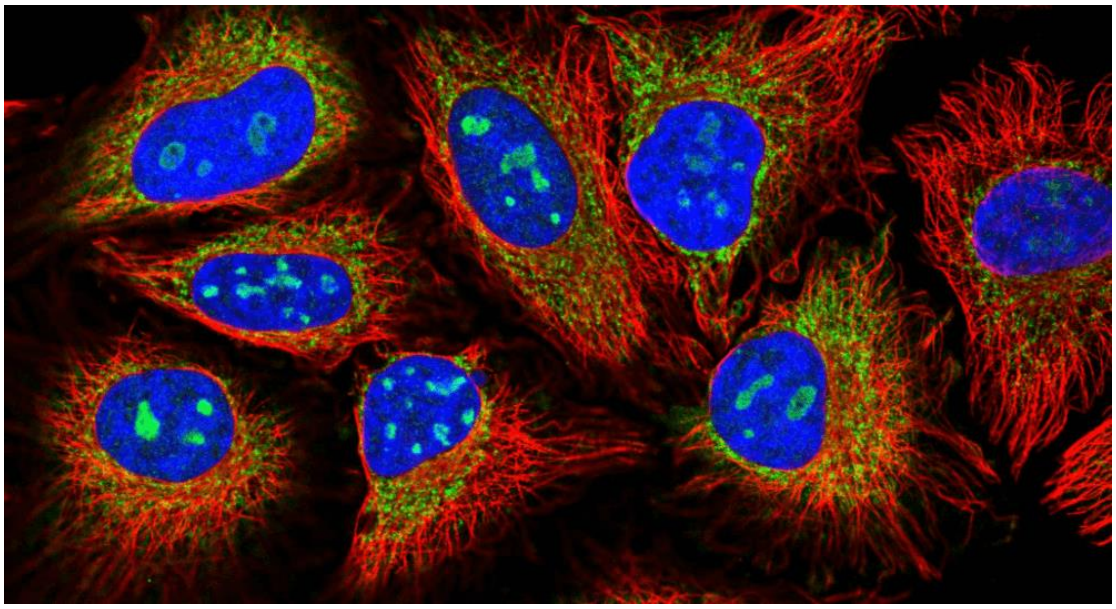
2018 Fall Machine Learning
Final Project Report

Human Protein Atlas Image Classification



Team: NTU_b04505025_下面應該沒人了吧

Leaderboard: Top12% - 250th/2173 (public)

Top13% - 280th/2173 (private)

Member: 電機四 b04505025 陳在賢

電機四 b04901064 林祐葳

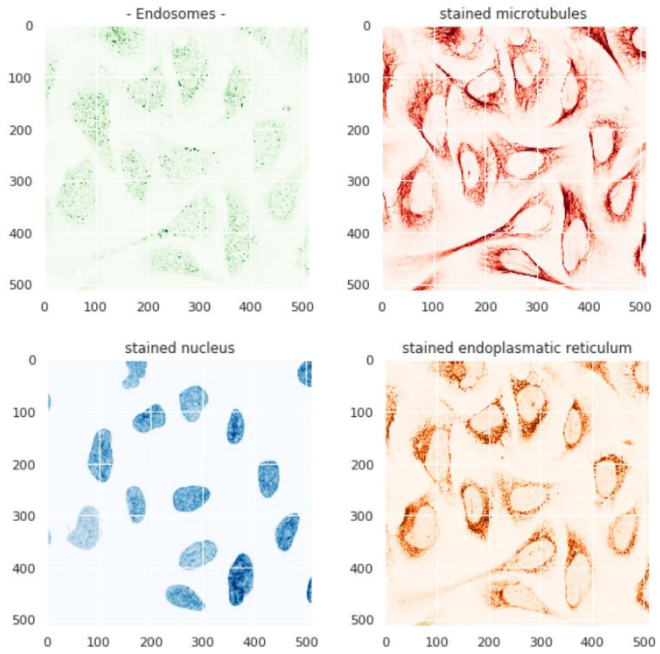
電機四 b04901065 游博翔 （依學號排序）

2019/1/14

1. Introduction & Motivation : 0.5%

本次專題是參加 Kaggle 上名為「Human Protein Atlas Image Classification」的公開競賽，競賽內容是要完成顯微鏡下照片的蛋白質樣本分類。

具題來說，會輸入含有 4 個 channel 的 512*512 顯微鏡下照片（4 個 channel 分別為胞內體、微管、細胞核、內質網的顯影圖），如下圖所示：



之後，要輸出此張照片中所含有的蛋白質樣本種類（共有 28 種蛋白質樣本且每張照片可能含有多於一種的蛋白質樣本）。

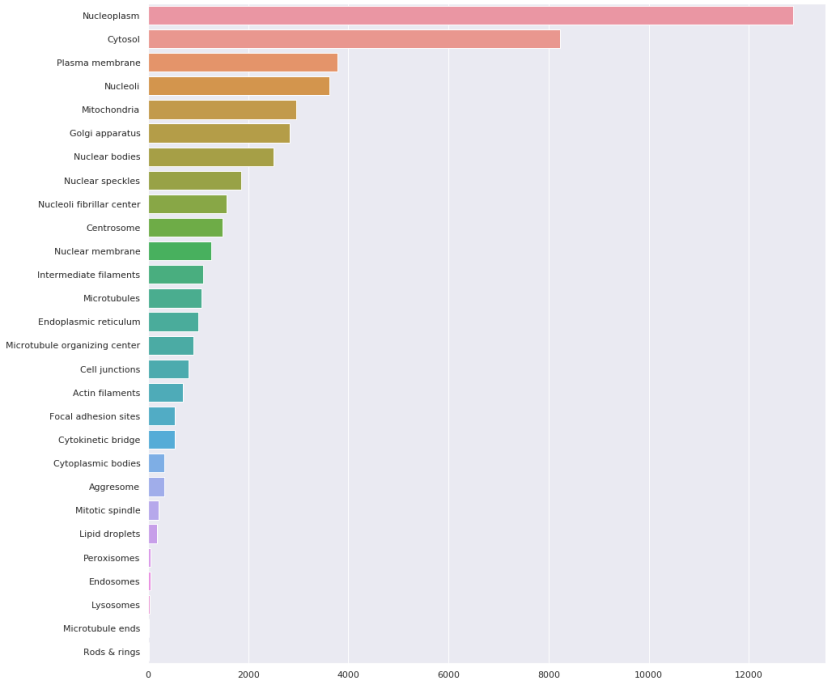
至於以此為專題題目的動機，是因為這是在 Kaggle 上當前進行的競賽，會有相當多機器學習相關領域的專家在上面做交流。而我們也想藉這個機會，觀摩這些專家的討論甚至了解他們的做法，從中獲取更為深刻的學習。

2. Data Preprocessing/Feature Engineering : 1.5%

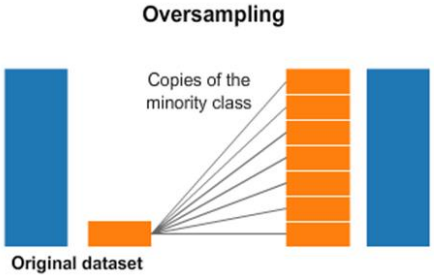
關於資料前處理部分，我們有使用了三個技巧：oversample、downsample、augment：

(1) Oversample

由於競賽提供的 data 有很嚴重的 class unbalanced 問題，亦即某些蛋白質樣本所對應的照片非常少，下統計圖展示了 30000 萬張 training image 中含各蛋白質樣本的 image 數：



對於照片數少的蛋白質樣本（其中有 4 個 class 甚至少到不到 100 張），我們很難要求模型在僅看到少數圖像下就學會對該樣本的辨別。因此就決定使用 oversample，該技巧會將含少量照片的 class 的 data 多複製幾次，如下圖所示：



但給 network 重複訓練「相同」照片會對準確度有幫助嗎？畢竟沒有新的照片就意謂著沒有新資訊。不過以我們實作的結果發現，此技巧甚至能大幅提升準確度。

事後討論原因，我們認為是因為此技巧相當於，讓模型在此些照片中多學幾個 epoch，以提升模型對這些 training data 的辨別正確率，在沒發生 overfitting 的情形下，也就能間接提升模型對這些樣本正確判讀的機率。

(2) Downsample

此步單純是將原本 512*512 的照片降至 256*256，以解決讀入圖像太大而使得 GPU 記憶體不足 (Out-of-Memory) 的問題。

(3) Augment

此步是對輸入圖片做特殊幾何變形（我們用了 rotate、shear、flip），並假設新圖對應的 label 不變，而以此做為新的 training data，送入模型做訓練。對輸入為照片的機器學習問題中，此為相當常用的技巧。

3. Model Description (At least two different models) : 2.5%

我們有用過兩種 model：一為最初使用的手刻 CNN、另一為後來用的 Batch-Normalization Inception (BN-Inception) network 且經 ImageNet pretrain 過。

關於手刻 CNN，是因為這次專題與 hw3 同為圖像輸入的 task。因此，起初我們先引用 hw3 中用 Keras 所疊出的 CNN network（為由 Convolution 3D layer、Max Pooling、Batch Normalization、Dropout 以及 flatten 後所接的多層 Dense layers 所組成的架構）。在參考很多相關論文以及多個已被廣泛使用的 network 後（如 resnet、GoogleNet 等），我們花了相當多時間刻出屬於自己的架構。當時在 public 上就已經有 0.325 的成績（已超過 simple baseline 的 0.28）。

然而，若在可使用各式套件的 final project 上，還用自己手刻的架構顯然會落人一大截。在參考網路上多篇分享文後，後來決定改用 PyTorch 引入現有的 network，搭配以其他 dataset pretrained 過的 model 為初始模型。而由於 PyTorch 已支援上述行為，因此僅需幾行 code 即可完成上述目的，如下圖所示：

```
def get_net():
    model = bninception(pretrained="imagenet")
    model.global_pool = nn.AdaptiveAvgPool2d(1)
    model.conv1_7x7_s2 = nn.Conv2d(config.channels, 64, kernel_size=(7, 7), stride=(2, 2), padding=(3, 3))
    model.last_linear = nn.Sequential(
                nn.BatchNorm1d(1024),
                nn.Dropout(0.5),
                nn.Linear(1024, config.num_classes),
            )
    return model
```

換上新的 model 後，public 就大幅進步至 0.425 了！

至於為什麼要用 BN-Inception network？這其實只是網路上許多參賽者共同試出來的結果，發現 Inception network 在此 task 很有效果而已，也很難說出他能優於其他 network 的具體原因。

又問，是否需要以其他 dataset 對模型進行 pretrain？畢竟能辨別狗、貓或其他物件的模型，在蛋白質的辨識還是一竅不通。不過以我們實測的結果來看，做過 pretrain 的模型確實表現得較佳。我們猜測原因在於，雖然 pretrain 對物件的辨識沒有直接的幫助，但能讓模型更快抓出圖像中出現的幾何圖形（如圓形、線條、邊界等）。因為這些是圖像中共同的重要特徵，在 pretrain 時可以學得到，而實際應用在其他 dataset 時也能有所幫助。

4. Experiment and Discussion : 4%

除了上述嘗試使用各種模型外，我們也另外嘗試過多個的實驗，而最終對準確度有明顯進步的，包含下列兩項：

(1) 改變輸入圖像的大小

前頁有說明過，由於 GPU 記憶體不夠，因此我們在資料前處理時，有將照片由 512*512 downsample 至 256*256。而關於解析度對準確度的影響，我們起初認為由於樣本分類只須依蛋白質的形狀做判定，因此降低解析度也不會對準確度有很大的影響。但為了驗證此事，我們又將照片 downsample 至 128*128 做 training，卻意外得發現準確率有明顯退步。

這也讓我們思考是否在資料前處理時做的 downsample，其實也降低了準確度。不過若要使用原始解析度做 training，就得先解決記憶體不夠的問題。於是我們用下述方法解決此問題：不要一次把所有 training data 讀入記憶體，而是在當前 batch 再把需要用到的資料讀入。在網路上查訊多筆資料後，知道 PyTorch 已內建此功能，只需搭配 Dataloader 函式就可完成上述目標，如下圖所示：

```
233     # load dataset
234     train_gen = HumanDataset(train_data_list,config.train_data,mode="train")
235     train_loader = DataLoader(train_gen,batch_size=config.batch_size,shuffle=True,pin_memory=True,num_workers=4)
251         train_metrics = train(train_loader,model,criterion,optimizer,epoch,val_metrics,best_results,start)
```

主要分為四步：

1) 幫 dataset 寫一個型態為 torch.utils.data.Dataset 的 class（此例中命名為 HumanDataset）並在裏頭修改必要的內建函式，之後 training 過程會自動根據這些函式讀檔
2) 生成一個型態為前步 class 的 data generator
3) 生成一個型態為 torch.utils.data.DataLoader 的 data loader，裏頭會傳入前步所生成 data generator
4) 最後將 data loader 輸入做 training 即可

在我們將資料以原始解析度做 training 後，也確實大幅提模型的準確度，讓我們在 public 的成績從 0.415 大幅進步至 0.477（已遠遠超越 strong baseline 的 0.42）。至於為何解析度在對蛋白質分類有這麼大的影響，這就牽扯到生醫部分的專業知識，我們也就沒細究其原因了。

(2) 使用 external data

在搜尋資料的過程中，發現主辦單位有在他們的官方網站上提供額外的 training data（Kaggle 上未提供的）。而大量數據是增進模型準確度萬年不變的法則，因此我們把 external data 載下來後加入原有 data 一起做 training。但令我們意外的是，這樣做卻反而讓模型的準確度降低！

思考多日後發現可能的原因在於，我們已對原始數據做 oversample（將少量樣本數的照片多複製幾次）。而若直接加入 external data，反而會讓資料更偏向為未做 oversample 的原始數據。因此我們後來假設照片數多的蛋白質樣本，早已被模型學習透徹而不需要再額外加入 external data 學；相反得，對 image 少的蛋白質樣本就要加入 external data，以增進模型在這些樣本的辨別正確度（此法稱為 undersample，與 oversample 有類似效果。只是前者是去除 data 數多的 class，後者是複製 data 數少的 class）。

使用了 external data 後，又讓我們在 public 的成績從 0.477 大幅進步至 0.529，甚至讓我們這組暫時進入銅牌行列中 (Top10%)。

除了上述兩個實驗外，我們還嘗試有用其他技巧，例如：用 ensemble 處理 Unbalanced data 的問題，具體內容為訓練一些只看過少量樣本數的模型，而讓模型不被擁有大量樣本的蛋白質主導，並將其與原模型做 ensemble。不過這些技巧都沒有對準確率有明顯的提升，因此就不做額外討論了。

5. Conclusion : 1%

我們這隊在 Kaggle 上 leaderboard 的排名為 Top13%，且在 NTU 組別中的排名為第 1 (public)、3 (private) 名。我們的模型之所以有不錯的表現，應該是因為包含以下三個主要特色：

(1) 引入現有 network，搭配以其他 dataset pretrained 過的 model 為初始模型

選擇強而有力的 network 可以讓訓練過程，有更高的機率抓出關鍵特徵而達更高的準確度，因此應選用前人辛苦研究的 network。同時以其他同為相片的 dataset 對模型進行 pretrain，可以讓模型更容易抓出相片中的特殊幾何圖形，使訓練過程更為迅速。

(2) 提升 training data 的解析度

本次 dataset 對照片解析度很敏感，提升解析度能明顯增加準確度。因此在用 PyTorch 的 Dataset 及 Dataloader 功能，解決記憶體不足的問題後，就能對模型輸入未經 downsample 的照片做訓練，而達到更佳的學習效果。

(3) 以 oversample/undersample 的方式提升照片少的樣本的辨別正確率

對照片少的樣本，我們很難要求模型在僅看過少量 data 就學會辨別該樣本。因此可透過對照片少的樣本的 data 多複製幾次 (oversample)，或在 external data 中去除照片多的樣本的 data (downsample)。以避免模型被照片多的樣本主導，同時強化對照片少的樣本的學習。

另外，若要在 Kaggle 上的 leaderboard 繼續往上衝，根據之前比賽排名相當前面的參賽者的分享，ensemble 是相當重要的一環，也就是要多試幾個模型以及參數設定。而假設各個模型都有其優、缺點，則將多個模型的預測結果做加權總合，就容易組出一個更為強大的模型。不過，這就得花相當多時間去嘗試，同時硬體設備的升級就顯得必要。而我們這組的模型目前仍為 single-model 的版本，因此未來還可以朝 ensemble 的技巧繼續發展。

6. Reference : 0.5%

[1] Protein Atlas - Exploration and Baseline
https://www.kaggle.com/allunia/protein-atlas-exploration-and-baseline?fbclid=IwAR2yriVvis8Tk7wbGCu_RJz4Duv4uAgkKRHXFhrEZIj9gINDOlrTk1Ptu0E

[2] Some information that may help you reach 0.59+
https://www.kaggle.com/c/human-protein-atlas-image-classification/discussion/75691

[3] summary of external dataset and leak
https://www.kaggle.com/c/human-protein-atlas-image-classification/discussion/75783