Human Protein Atlas Image Classification

隊名：劉 X 鵬程萬里

電信碩一 詹鈞皓 R06942141

電機碩一 劉兆鵬 R07921052

生機碩二 王凱陞 R06631035

Introduction：

在這個比賽中，我們將要開發能夠在顯微鏡圖像中對蛋白質的混合模式進行分類的模型。而我們為什麼要對蛋白質進行分類呢？蛋白質是人體細胞中的“行動者”，執行許多共同促進生命的功能。 為了完全理解人類細胞的複雜性，我們開發的模型必須在一系列不同的人類細胞中進行分類。通常可視化細胞中蛋白質的圖像是用於生物醫學的研究， 然而，由於顯微鏡的進步，這些圖像的生成速度遠遠超過利用人力資源來給予標籤的速度。 因此，對於使生物醫學圖像分析自動化，以加速對人類細胞和疾病的理解，擁有比以往更大的需求。

關於訓練資料的部份，這個比賽中的 training data 是非常巨量，且由於資料的 data imbalance 頗為嚴重，訓練資料中懸殊的 labels 量會導致 training model 的時候，會因為過度偏向數據較多的一方，而沒有完整得考量到所有 labels 的分佈情形，此時若 testing data 的資料分佈不太相同，則會導致預測結果變得很差。除了 data imbalance 之外，這組資料還是 multiclass 和 multi-labels，在 multi-labels 的情況下，使用 softmax 的話則會導致圖片中可能有出現該類別但因為轉換成機率後沒有超過原先制定的 threshold 則會忽略掉此類別。

所以在這次競賽中，有兩個主要的問題要解決，分別是(1) data imbalance (2) multi-labels，再資料前處理部份，我們分別使用了 oversampling，增加少量類別的資料，和 focal loss 來調整 loss 的計算方式。在 model 部份先後使用了 CNN 和將 ResNet34 當作 pre-trained model 來訓練，最後分別訓練了 4 組 model，每個 model 分別預測不同的類別，之後來作 ensemble 來化解不平衡的問題。

在報告最後，我們放上了我們實驗的數據和結果，雖然 ensemble 的結果不甚理想，但再原本使用 ResNet34 pre-trained 的 model 表現最好，制定 threshold 部份是動態

去調整每一個類別的 threshold，量身打造最適當的門檻，在 Kaggle 上的排名佔全部隊伍的前 16%。

Data Preprocessing：

1. data Imbalance：

   kaggle kernel 中，較為常見的方法有針對少數 labels 的圖片進行 image augmentation 或是使用不同的 loss function，同時選擇正確的 evaluation matrics 也會對於結果有很大的影響 [4]，針對 data imbalance 的探討，Rushi Longadge, etc. [5]整合了一些常用的處理方式，under sampling 和 oversampling 都是值得嘗試的作法，同時亦可以將 under sampling 和 image augmentation 一起使用，將較為少數類別的資料增加，此外 focal loss [6]的作法適用於 data imbalance 的 model training，透過調整計算權重的方式，改善過度專注於訓練多數類別資料的狀況。

   在我們的模型中加入了 focal loss，相較於未加入 focal loss 於 public score 差距就可從未過 strong baseline 到 0.43 左右的效果。

2. data preprocessing ：

   在圖片的預先處理部份，我們參考了幾位不同的 kernel 分享，發現幾乎使用的 image augumentation 相似度都很高，幾乎都是選擇基本的 30 度內的隨機旋轉、或是透過 Dihedral 進行水平或垂直翻轉，以及亮度的微量調整，我們先針對這 Rotate 和亮度調整進行參數調整，但是測試了幾次的結果不論是在 public score 或是 validation set 上都沒有太多的差別，所以最後我們在 Image augumentation 的選擇上，也和大部份 kernel 所採用的相同，將隨機角度設為 30 度內，亮度調整範圍在 0.05 之間，在使用 fastai 實作的狀況，fastai 中的 tfms_from_stats 讓我們可以從 4 個通道預先跑出的平均值和標準差進行 image augumentation，這部份的結果也較於使用 keras 來的更加有效。 在圖片的尺寸上，一開始我們採用 256 x 256 的圖片大小，但經過測試之後發現 512 x 512 的大小會有較好的表現，除此之外，在 kernel 上有許多人提到了可以利用官方釋出的 externel data 增加資料量，然而官方所釋出的資料有許多圖片是有缺通道的，並不是全部的圖片都有 4 個通道，此外也有重複出現圖片的狀況，這部份我們

參考了並使用了[12]所整理的資料，將缺通道和過度重複的圖片都刪掉，將原本的 31071 筆資料加到 80057 筆資料，增加訓練資料，但多增加的資料也和原先的資料一樣有嚴重的 data imbalance 的問題，如 Fig.1，在經過我們測試之後，並沒有將分數有效的提高，這部份我們認為有可能的原因是另外增加的資料始 imbalance 的狀況更加嚴重，且資料量的增加導致我們需要設更多的 epoch 進行訓練，在時間有限下，加入額外的資料為提升訓練次數，以至於不能制定正確的 threshold，使增加資料量不能有效的提高訓練結果。

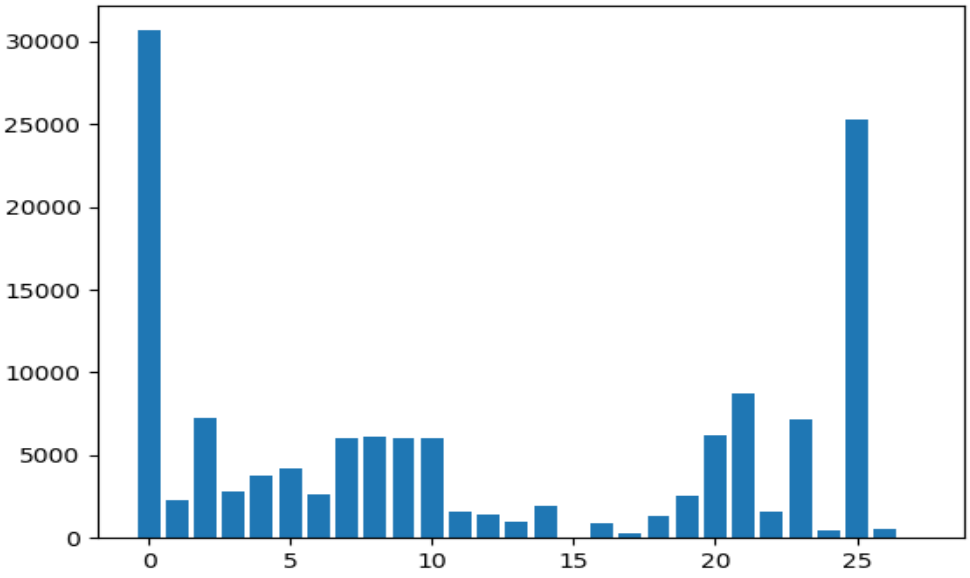

Fig.1 External Data 的類別分佈圖

Model Description:

1. multilabels case :

在傳統的物件分類中，主要都為找出圖片中機率最大的物件為何，而在輸出分類層的部分則會透過 softmax activation function 將各類別所輸出的值根據各類別中所佔的比例轉換為機率值，之後在預測時只需要透過 argmax 則能夠選出最大機率的類別則能夠找出圖片中的物件。所以在 multilabels case 中，我們會將最後一層的 activation function 轉變成使用 sigmoid [7]，將各類別所輸出的值透過 sigmoid 轉換成 0 與 1 之間的值，接著只要將各類別所輸出的跟去我們所定義的 threshold 來判定此圖片中是否包含到此物件。近年來研究者開始往不

同的 multilabel model 研究，如 [8] 透過全卷積層的模型進行圖片的特徵萃取工程，而有別於以往所使用的 sigmoid activation function 將值轉換至 0 與 1 之間的方法，此方法在最後一層時使用 1x1 的 inception unit，將模型的最後一層的輸出 channel 投射到與類別數目相同，也就是將原始圖片中的各類別，透過全卷積層將圖片中各物件都投射到各自的 feature map 中，接著只要透過最後一層中各類別的 feature map 計算各類別的分數，則能夠辨識出此圖片中是否包含著以下的物件。另外，也有研究者使用 sub-concept [9] 的方式來辨別物件中有包含哪些類別，主要方法為使用全卷積層將圖片的特徵進行萃取，接著使用圖片經過卷積層後所降維的 feature map 經由 sub-concept model 將 feature map 進行維度轉換，若圖片中就是包含著固定數量的物件，透過 sub-concept model 就能夠對每一個假設的物件進行分數的計算，進而找出圖片中有包含哪些類別。

2. Model structure :

最一開始我們所採用的模型為基本的 Convolutional neural network（CNN）架構，而有別於一般的 Batch normalization 的部份，我們是針對每一層 channel 的部份進行 normalizaiotn 而非對 batch 維度進行，是因為我們認為此次的圖片為採用 4 個 channel 作為輸入，然而這四個 channel 的資訊並非傳統的光源顏色資訊，而是透過染料顏色所呈現的資訊，所以在原始資料中，圖片的各項 channel 是沒有相關性的，故我們會採用對 channel 的部份先做 normalization 的步驟。此外，因為對於四個顏色的圖片是能夠對於不同物件偵測而成，故我們在模型中也會先分別對四個 channel 的顏色進行 convolutional 的運算，之後再將各 channel 所計算後的 feature map concatenate 起來繼續往下訓練。另外，根據 kaggle kernel 上 lafoss 的教學，使用 pretrained model 的方法能夠獲得更高的成績，原因是因為使用 pretrained model 的 weight 是已經學習過無數張圖片的特徵，所以將之轉換到此任務上也能夠使模型叫快收斂，然而，因為這次的圖片與 pretrained model 所預先訓練的圖片資訊相差甚遠，故在平常一般的 transfer learning 都會將 pretrained weight 設定為 non-trainable 的，但是因為在這次比賽中的圖片較為不同，故我們會將 pretrained weight 設定為可以計算訓練的。使用 pretrained 的目的是只為了讓模型能夠有好的初始值，而透過新的圖片訓練學習讓模型能夠更加的專注在這次比賽中圖片的特徵架構。

然而一般所採用的 pretrained model 都為使用生活中的圖片內容，如 VOC, ImageNet 等圖片集的圖片都是為 3 個 channel (rgb)的圖片，但是我們這次所使用的輸入圖片為 4 個 channel，故直接採用 pretrained model 來訓練是不可行的。故我們將使用 Resnet34 作為 pretrained model ，但我們將此 model 的第一層 CNN layer 的 weight 取出來，並將此層的 CNN 中的 previous layer 改為 4 個 channel，並將原先 pretrained 的 rgb weight 設定自新的 CNN layer 中，如 Fig.2，但因為 pretrained weight 只有三個通道，故於新的 CNN 的 Y channel 中我們會將 pretrained weight 中的 r 與 g 做平均後放入至新的 CNN layer 中的 Y channel ，使得此 Y channel 也能夠在好的初始值下進行訓練。

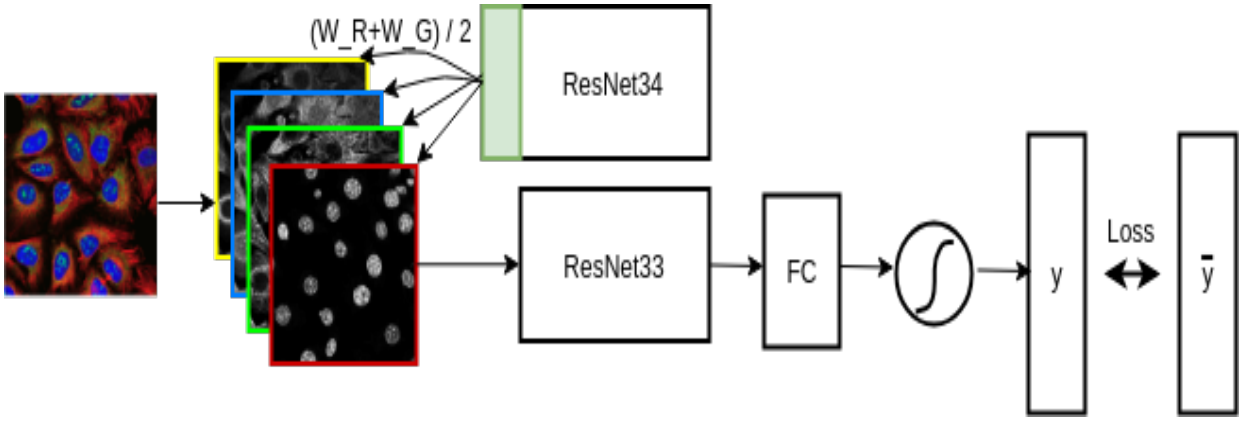

Fig2. ResNet34 架構

更改完 pre-trained model 之後就可以開始 training，我們有更改 ResNet34 中的 loss function，我們更改成 focal loss，然後在 ResNet34 結束之後經過 Flatten 進入 Dense layers 預測，如前文所述，為了因應 multiclass，我們把 activation function 更改成 sigmoid。然後進入 threshold training 的部份。

除了前面兩種方法，我們也試了 ensemble 的方法，因為總共有 28 類別，且 data imbalance 的狀況又很嚴重，所以我們透過[11]中的介面，用肉眼先去分辨細胞中，每個類別的分佈和形狀，我們自己分了 4 個群，再分別去訓練相對應的 model，讓每一個 model 各司其職，最後將這 4 個 model ensemble 起來。但是這個方法的表現並沒有很好。

3. how to define threshold

由於上文提到，我們所訓練出的 model 會面臨 multilabels, multicase 的挑

戰，而為了要提升 classifier 的精確度，這會導致某一不常出現的類別都不出現的情況，所以我們找了討論在 multilabel 和 data imbalnce 的情況下如何選出 threshold 的論文。在研究 Kaggle 上的 kernel 後，定義 threshold 的方法有很多種，像是上述提到最簡單的：定義全部類別的 threshold 為 0.5，或是利用 validaion data 來去微調，測出在哪種 threshold 下所得到的 validation score 會最高。除此之外，另一種方法則是去調整每個類別的 threshold[3]，可以根據每個類別出現機率的分布來制定，又或是單獨依照 R, G, B, Y 四個 channel 分別制定 threshold 再 ensemble 起來得出最終的 threshold 數值。

然而，在 multilabes classifier 下制定 threshold 的領域中，已經有眾多學者做過研究，其中以 Binary Method[1]表現得最好，在多個類別的情況下去針對每個類別設計一個 decision function，並根據此 function 來制定最終的 threshold。

在這個競賽中當我們要做 predict 時，往往會需要用到 Test Time Augmentation(TTA)[10]的技術，他就是在 Validation data 或是 testing data 也隨機作 augmentation 的技術，像是旋轉圖片、翻轉圖片等等，使用這項技術有助於提高我們預測的精確度，既然對預測精確度有影響，當然對於 threshold 的制定也有很大的影響。

我們這組嘗試了統一所有類別的 threshold 數值為 0.5，但效果並不好，我們還去浮動制定 threshold 的數值，並不一定是把 0.5 為分水嶺，但是因為這個競賽要預測的是 multilabes，所以我們覺得對於每項 class 都定義一個專屬於他們的 threshold 是比較合乎邏輯的，所以而我們目前使用制定 threshold 的方法是依照每一個類別的分佈情況來制定的，出現愈多次的類別，相對地 threshold 數值就要比較高，比較難跨越，而出現較少次的類別，就給定較小的 threshold 數值，在應用了這種 distribution 的 thrsehold 後，就有更好的效果。

然而，除了制定靜態的 threshold 之外，我們應該也要能學習動態制定 threshold 的方法，我們使用 focal loss 的技術來調整 loss，並且再最後捨棄 softmax 的 activation function，這改動了我們預測機率的結果，所以訂定 threshold 的方法也要隨之更動。我們在這部份參考了 kaggle kernel 上 lafoss 的最佳化的方法，我們從 training 切了 10%的資料當作 validation data，接下來使用

validation data 預測的結果來最佳化 threshold。首先使用 F1-soft 的方法，其公式如下：

$$x = \sigma(d * (x - 0.5))$$

$$score = \frac{\sum 2(x * y)}{\sum (x + y) + 0.00001}$$

x 為 validation 預測的結果，y 為 labels

在算出 F1-soft 的分數後，最小化所有平方的總和，也就是將 error 最小化，然後將最佳化的 threshold 輸出。

Experiment & Discussion:

1. Training model structure

在這次競賽我們總共實作了 3 種方法，分別為 Convolution Neural Network(CNN)、單純用 pre-trained ResNet34、ResNet34 並更改為 focal loss、和 Ensemble。一開始看到這個圖目就很直覺地要使用 CNN 來做，但是單純只使用 CNN 再這個比賽上是行不通的，因為同時有 multilabel, multiclass 和 data imbalance 的問題要解決，因此我們要使用不一樣的訓練方法。

所以之後再 survey 了各式各樣的方法，選擇使用了 ResNet34 當作 pre-trained model，並且將 loss function 更改為 focal loss 來解決 data imbalance 的問題，而這之中又遇到心的問題，一般的 ResNet34 的輸入是只有 3 個 channel 但是這個競賽的圖片卻有 4 個 channel，分別為 R、G、B、Y，所以我們必須自己更改 ResNet34，讓他能正常地運作，於是我們將此 ResNet34 的第一層 CNN layer 的 weight 取出來，並將此層的 CNN 中的 previous layer 改為 4 個 channel，並將原先 pretrained 的 rgb weight 設定自新的 CNN layer 中，但因為 pretrained weight 只有三個通道，故於新的 CNN 的 Y channel 中我們會將 pretrained weight 中的 r 與 g 做平均後放入至新的 CNN layer 中的 Y channel ，使得此 Y channel 也能夠在好的初始值下進行訓練。 然而，在結果上我們發現 focal loss 的效果還是有極限的。

我們之後又去做了新的嘗試，我們覺得讓 Model 一口氣對 28 種類別會造成太大的負擔，所以我們打算讓 model 分群處理，我們透過”The Cell Atlas”[11]

這個互動式網站上的功能，用肉眼去區分哪些類別一起出現的機率比較大，分佈在細胞的同一區域，進而分群。最後我們分出了 4 群，如下所示：

group1_c = [0,1,2,3,4,5]
group2_c = [6,7,11,12,13,14,15,16,17]
group3_c = [8,9,10,18,19,20,23,27]
group4_c = [21,22,24,25,26]

再訓練完 4 個不同的 model 之後，就將其 ensemble，並且透過 validation data 來這定這組新的 model 的 threshold。但是最後結果確意外地非常差

下表為我們這次實作所有架構在 kaggle leader board 的表現，其中表現最好的就是 pre-trained ResNet34 + focal loss，這組 model 能最有效地解決上述提到的兩個問題，在 Kaggle 的排名也取得不錯的成績。

	public score	private score
CNN	0.272	0.268
ResNet34	0.310	0.306
ResNet34 + focal loss	0.489	0.484
Ensemble	0.1	0.096

2. how to define threshold

要如何制定精確的 threshold 就是一門學問。最簡單制定 threshold 的方法毫無疑問就是將所有類別的 threshold 統一設為 0.5，超過 threshold 的機率就判定為 True，反之則為 False，但在這個競賽中，data imbalance 的影響甚為巨大，這將會使這簡單的機制失去效用，因為在 imbalance 的資料組中，有較不常出現的類別是不太可能超過 threshold 為 0.5 的這堵高牆，像是第 27 類 Rods & rings 僅出現 11 次而已，而第 0,1 類出現的次數都是好幾千倍，這會讓我們的 model 傾向不選擇第 27 類，我們的實驗結果如下表，其反映出將所有類別的 threshold 都設為 0.5 是行不通的。

為了要顧慮到每個類別的出現次數，所以我們對每個類別出現的分佈做了統計，其公式如下：

$$P_i = \frac{C_i}{N}, i \in [0, 1, 2, .., 27]$$

P_i代表的是該類的 threshold 數值，N 為所有類別出現的次數，C_i為第 i 類出現的次數，根據以上公式計算出的 threshold 代表各種類別出現的頻率，出現愈少次的類別會有愈少的 threshold，代表他們要跨過的門檻比較低，反之出現比較多的類別的 threshold 就相對較高，就需要跨過較高的門檻。而從下表我們的實驗結果表示，得到的結果也有顯著的提升。

在這之後，為了因應 data imbalance 的問題，我們使用 focal loss 的技術來調整 loss，並且再最後捨棄 softmax 的 activation function，這改動了我們預測機率的結果，因為訂定 threshold 的方法也要隨之更動。如上述提到，我們去研究找出了一套 threshold learning 的方法，使用這個方法可以使我們的 model 隨著 validation data 來去調整我們 threshold 的數值，並且使用 TTA 的技術可以讓我們的準確率提升很多，但相對應的也增加許多預測的時間。我們目前使用最好的一組 threshold 是：

[0.565, 0.39, 0.55, 0.345, 0.33, 0.39, 0.33, 0.45, 0.38, 0.39, 0.34, 0.42, 0.31, 0.38, 0.49, 0.50, 0.38, 0.43, 0.46, 0.40, 0.39, 0.505, 0.37, 0.47, 0.41, 0.545, 0.32, 0.1]
得到 0.489 的結果。

3. Association rule:

Association rule 的意思常用再資料分析上，他會去計算 association 的規則，像是通常買了鐵鎚的人還會再買釘子，這個規則由兩個條件所制約，分別是 minimal support 和 minimal confidence，minimal support 代表的是這則規則有多有用，在所有的資料中至少要出現一定的次數才會認可這個規則是普遍會出現的，而 minimal confidence 則是代表這個則規則有多少的可信度，他會算出這則規則中若 A 出現，A 和 B 會一起出現的次數，調列出所有的規則。而這次的比賽我們也有對 training data 做分析，因為是 multilabels 的問題，所以一次可能有許多類別一起出現，所以我們就去找看看，什麼類別出現的時候往往會和另一個類別一起出現。我所使用的方法是 Apriori algorithm 和 Eclat algorithm，他們的差別是 vertical mining 和 normal mining，主要目的都是去找 large itemsets，但 Eclat algorithm 的執行速度會比較快。而下表就是我們執行出來的結果，我們設定 minimal support 為 0.06：

Labels	Counts	Labels	Counts
(0, 8, 9, 10, 20)	1302	(9, 10)	6032
(8, 9, 10, 20)	5953	(8, 9)	5953
(0, 9, 10, 20)	1302	(2, 25)	1711
(9, 10, 20)	5953	(0, 19)	933
(9, 10)	6032	(10, 20)	5953

所以我們發現（0, 8, 9, 10, 20）是最常出現再一起的，0, 8, 9, 10,20 這些會互相出現，而其他像是(2,25), (0,19)的類別也是會一起出現的，這可以當作我們再分群時的參考。

Conclusion:

在這次的比賽中，雖然看起來是一般普通的影像分類問題，但是在 data imbalance 和 multilabels, multicase 的情況下，這個問題就變得複雜很多。除此之外，在一開始參加這個競賽的時候都還完全不知道每張圖片要表達的意義，全部都是細胞的圖片，不了解其他意義。在看過 Kaggel 上面的 kernel 和 discussion 後，才漸漸知道要以生物的角度來理解這些圖片，然後從"The Cell Atlas"這個網站才更了解我們要分類的 28 種類別各自代表什麼意義。

然後我們我們花最多時間要解決的就是一開始提到的那兩個問題：data imbalance 和 multilabels，雖然 data imbalance 的問題到最後還是沒有很好的解決，但在做資料分析時並應用 oversampling 技術就有得到一定的成效，然後透過實作 facal loss 再取得進一部的成果。而 multilabels 就依靠不同的 activation function 和 threshold 讓維持每個類別的預測功能。最後我們做出最好的 model 是用 ResNet34 作為 pre-trained model 並修改 focal loss，讓上述提到的兩個問題的影響最小化，然後接著使用最佳化的 threshold 完成預測，在 kaggle leaderboard 上的 public 的成績為 0.489，private 為 0.484，最後排名第 289 名，佔總隊伍前 13%。

Reference:

[1] K. Nigam, A. K. McCallum, S. Thrun, and T. M. Mitchell. Text classification from labeled and unlabeled documents using EM. Machine Learning,https://link.springer.com/article/10.1023/A:1007692713085

[2] Rong-En Fan and Chih-Jen Lin∗, A Study on Threshold Selection for Multi-label Classification, https://www.csie.ntu.edu.tw/~cjlin/papers/threshold.pdf

[3] Iafoss, pretrained ResNet34 with RGBY (0.460 public LB), https://www.kaggle.com/iafoss/pretrained-resnet34-with-rgby-0-460-public-lb

[4] Michal Haltuf. Input preprocessing for fast data loading, https://www.kaggle.com/rejpalcz/best-loss-function-for-f1-score-metric

[5] Rushi Longadge, Snehalata Dongre, Class Imbalance Problem in Data Mining Review, https://arxiv.org/pdf/1305.1707v1.pdf

[6] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, Piotr Dollár. Focal Loss for Dense Object Detection,https://arxiv.org/pdf/1708.02002.pdf

[7] Andre F. T. Martins, Ramon F. Astudillo.From Softmax to Sparsemax: A Sparse Model of Attention and Multi-Label Classification., http://proceedings.mlr.press/v48/martins16.pdf

[8] Yanzhao Zhou, Yi Zhu, Qixiang Ye, Qiang Qiu, Jianbin Jiao.Weakly Supervised Instance Segmentation using Class Peak Response.https://arxiv.org/abs/1804.00880

[9] Ji Feng, Zhi-Hua Zhou.Deep MIML Network.https://cs.nju.edu.cn/zhouzh/zhouzh.files/publication/aaai17deepMIML.pdf

[10] Alexander Liao, Image Augmentation Demo with albumentation https://www.kaggle.com/alexanderliao/image-augmentation-demo-with-albumentation

[11] The Cell Atlas https://www.proteinatlas.org/humanproteome/cell?fbclid=IwAR2I4k1kyUN9L7wNErvg84X8ySGnC2lFNN6Hy9U7OofoFmbl7GwaJU_D2lc

[12] Clear Human Protein Atlas Additional Images and Datahttps://www.kaggle.com/kgeorge/human-protein-atlas-additional-images?fbclid=IwAR07rSLPFSoVMYJFXdvELz0TDKVyiz8uIblB1lW7R3wKaU8BD7j5MOoLhr0