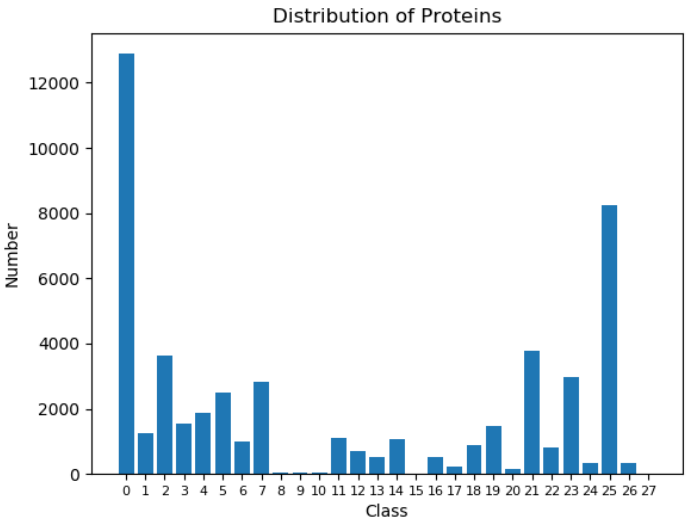
Final Project Proposal
Human Atlas Protein Image Classification

NTU_b04505025_下面應該沒人了吧
B04505025 陳在賢
B04901064 林祐葳
B04901065 游博翔

Problem Study

Imbalanced Data

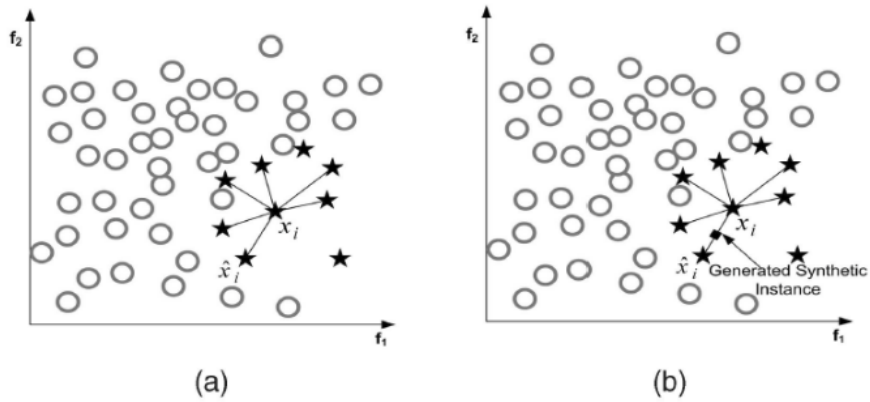
所謂的 Imbalanced Data 指的是在 training set 中，每一類的訓練資料量分佈不均，且有一定的差距，也就是 class-imbalanced。舉例來說，在一個二元分類的問題當中假設屬於 1 的有 995 筆資料，而 0 的只有 5 筆，對於分類器而言不用進行分類直接猜結果為 1 具有 99%以上的正確率(loss function 以 accuracy 來看)，也就是說分類器無法學習到 0 的特徵。下圖所示為本次 task 中，training data 在 28 種 protein cell 的分佈，從肉眼即可判斷資料分佈嚴重的不均衡，訓練資料在 0 和 25 明顯較多，而在 8,9,10 則是明顯的少於平均。可以推測在 8,9,10 的圖片上會較分辨不出來。



要解決資料不平均的問題，我們根據以下三個大方向進行研究：包括資料的 sampling、class weights 以及 loss function 的選擇。

1.SMOTE:Synthetic Minority Over-sampling Technique

sampling 是解決資量不平均最直接的方法，其中包括對於資料量少的進行 over-sampling，或是對於資料量多的進行 under-sampling。我們不想使用 under-sampling 放棄已經獲得的 training data。而我們找到的這篇論文是 2002 年的，有 6000 多個 citations。使用的方法是 synthetic 的方式進行 over-sampling，下方圖以二元分類為例進行說明。



對於資料量較少的 data，我們對於任兩個資料點的 features 進行 linear combination，而使用的兩個係數和為 1，也就是說新的資料點會在選取的兩個資料點的連心線上，以此原理增加新的資料，並且也能新增的資料對於該類資料也是相對合理的。目前我們正想辦法應用此技巧於 image 的 preprocessing。

2.ADASYN: Adaptive Synthetic Sampling for Imbalanced Learning

ADASYN 與上述的 SMOTE 原理十分類似，是於 2008 年的論文。同樣使用 synthetic(如上圖所示)的方法，然而產生的點是以非該類別而鄰近的點進行 sampling，主要目標是要針對 outliers，減少 outliers 對於該類別的影響，但我們認為這可能對分辨率的提昇很有限。

3. Loss Function

(1)Class Weights:

對於不同資料量的類別我們可以在計算 loss function 時給予不同的 weights，使得資料量雖然少但對 loss 影響的 weights 較大，也能因此提昇該類別 data 的影響力。可與下方所述的 Macro-F1 Score 進行整合，使用加權平均的概念來表示 loss function。

(2)Macro-F1 Score:

首先為什麼要使用 F1 Score 的原因其實在前面已經有講過原因了，假如只使用準確率來判斷的話會導致分類器過於傾向資料量龐大的那一類，而使得表面上 training loss 很小，實際上已經 overfitting 了。

而由於此次 task 是 multi-label，因此必須使用 Macro-F1 Score 也就是將每一類的 F1 Score 進行平均。如下圖：

$$Macro - F1 = \frac{1}{c}\sum_{i=1}^{c}\frac{2TP_i}{2TP_i + FN_i + FP_i}$$

Related Task Deep Dive

由於是分辨人體蛋白質，因此選擇此篇論文做探討

Automated Analysis Of Human Protein Atlas Immunofluorescence Images

此篇 paper 所研究的主題也是人類蛋白質的影像辨識和特徵選取，因為和此次我們 final project 的主題很接近，因此便想閱讀和稍微研究一下，了解他們是使用什麼方法來訓練模型，處理資料，並希望藉機學習一下他們的方法來改善我們的分類效果。

論文中，作者主要嘗試了兩種方法，以下分別簡單介紹：

1.採用 **RBF 核的支援向量機**分類

使用 N-fold 交叉驗證，並在每一組 Fold 中採用 **stepwise discriminant analysis(SDA)**來選出最有代秉性的特徵。而 SVM 的 kernel parameter 和 slack penalty 選 64 和 0.25，另外為了降低 imbalance dataset 對結果的影響，**class weighting** 的方法也有採用。上述的模型使用 **LIBSVM(http://www.csie.ntu.edu.tw/ cjlin/libsvm/)** 這個 toolbox 來實現。

2.Random Forest 分類器

作者使用 500 棵「樹」然後在每個節點使用 11 個特徵來實做，使用 version 4.5-30 of the randomForest package (http://cran.rproject.org/web/packages/randomForest/index.html)來實現。

而結果的部份(Table1)，總體來說 Random Forest 的效果較好，但若資料未受污染(作者定義為清晰沒有污點的影像)，則 SVM 亦可獲得很好的成果。另外，使用 SDA 和 RF 兩種方法取出的 feature 大致雷同，因此可推論這些 feature 對訓練會有較大的影響。

Classes	#field img.	#cell img.	SVM field	SVM cell	SVM voting	RF field	RF cell	RF voting
Centrosomes	40	289	30.0	11.4	7.5	45.0	29.4	40.0
Cytoskeleton	260	2035	61.2	58.2	67.8	67.3	66.5	70.3
ER	226	1818	77.9	71.6	81.3	81.9	78.9	85.5
Golgi apparatus	218	1800	66.1	57.8	74.5	68.3	66.4	75.5
Punctate patterns	180	1497	76.1	60.3	73.1	70.0	68.6	72.6
Mitochondria	612	4905	86.4	80.0	91.6	91.2	87.4	96.2
Nucleoli	247	2114	70.9	65.4	72.1	76.9	74.1	81.2
Nucleus	1720	14183	96.3	95.7	98.3	98.0	98.0	99.2
Plasma membrane	54	458	53.7	27.3	28.8	46.3	46.5	40.0
Overall accuracy	—	—	84.8	80.6	87.4	87.5	85.8	89.8

RF	SDA	Feature name
1	1	Corr. between prot. and nuc. channels
2	2	Corr. between prot. and ER channels
3	64	Prot. and ER obj. overlap int. ratio
4	30	Threshold adjacency statistic #13
5	29	Prot. and nuc. obj. int. overlap ratio
6	11	Threshold adjacency statistic #12
7	7	Prot. and nuc. obj. area overlap ratio
8	10	Prot. and tub. obj. int. overlap ratio
9	4	Non-object fluorescence
10	3	Texture #16: corr. (4x downsampling)
11	37	Variance of # of pixels per object
12	35	Texture #3: corr. (4x downsampling)
13	5	Corr. between prot. and tub. channels
14	16	Info. between prot. and nuc. channels
15	12	Info, between prot. and ER channels
16	28	Ratio of largest to smallest obj. size

透過研讀此篇 paper，我們認為可以嘗試將 SVM 或是 RF 的架構應用在我們的模型上，因為目前我們只是用單純的 CNN 的架構來作訓練。

Proposed Method

以上的論文閱讀提供我們模型日後改進和 fine tune 的方向，而做為第一版最原始的模型，我們先簡單的設計一個基本的 Deep 的 CNN 架構，其中卷積層的 filter size 以 2 的次方指數增長逐層加大，採用 ReLU 激勵函數，並在每一連接層中都加入 BatchNormalization 層，並適時的加上 DropOut 和 MaxPooling 層。

值得注意的是由於此次的 task 中，一張影像可能會包含多種蛋白質，為一 multilabel classification 的題目，若和一般情況一樣，在最後 Fully-Connected Network 的採用 softmax 激勵函數，會導致 label 偏向單一化，因此我們改用 sigmoid function 做為

輸出層，並且透過調整門檻，將機率超過門檻的 dimension 都視為可能的 label，這樣會讓結果更為準確。

之後預計進行的方向有：

1. 採用 pretrained model 來進行網路前段的訓練
2. 使用 imgaug 套件來做影像資料前處理
3. 改使用 SVM 或 RF 等不同的網路建構方法，而不是使用單純的 CNN
4. 加入 SMOTE 算法來改善
5. 進行不同 learning rate scheduling 實驗
6. 針對 multi-label 的 scenario 採用 stratification 方法分類
7. 加入 cross-validation ensemble 機制，結合不同模型的訓練結果
8. 調整超參數，進行多次實驗

References

1. Newberg J, Li J, Rao A, Pontén F, Uhlén M, Lundberg E, Murphy R. Automated analysis of human protein atlas immunofluorescence images. In: Biomedical Imaging: From Nano to Macro, 2009. ISBI'09. IEEE International Symposium On. IEEE: 2009. p. 1023–1026.

2. Nitesh V. Chawla , Kevin W. Bowyer , Lawrence O. Hall , W. Philip Kegelmeyer, SMOTE: synthetic minority over-sampling technique, Journal of Artificial Intelligence Research, v.16 n.1, p.321-357, January 2002

3. He, H., Bai, Y., Garcia, E.A., Li, S.: Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In: Proceedings of the International Joint Conference on Neural Networks, 2008, part of the IEEE World Congress on Computational Intelligence, 2008, Hong Kong, China, June 1–6, 2008, pp. 1322–1328 (2008)

Appendix

Current Model

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	(None, 192, 192, 3)	0	
batch_normalization_1 (BatchNor	(None, 192, 192, 3)	12	input_1[0][0]
conv2d_1 (Conv2D)	(None, 190, 190, 8)	224	batch_normalization_1[0][0]
activation_1 (Activation)	(None, 190, 190, 8)	0	conv2d_1[0][0]
batch_normalization_2 (BatchNor	(None, 190, 190, 8)	32	activation_1[0][0]
conv2d_2 (Conv2D)	(None, 188, 188, 8)	584	batch_normalization_2[0][0]
activation_2 (Activation)	(None, 188, 188, 8)	0	conv2d_2[0][0]
batch_normalization_3 (BatchNor	(None, 188, 188, 8)	32	activation_2[0][0]
conv2d_3 (Conv2D)	(None, 186, 186, 16)	1168	batch_normalization_3[0][0]
activation_3 (Activation)	(None, 186, 186, 16)	0	conv2d_3[0][0]
batch_normalization_4 (BatchNor	(None, 186, 186, 16)	64	activation_3[0][0]
max_pooling2d_1 (MaxPooling2D)	(None, 93, 93, 16)	0	batch_normalization_4[0][0]
dropout_1 (Dropout)	(None, 93, 93, 16)	0	max_pooling2d_1[0][0]
conv2d_4 (Conv2D)	(None, 93, 93, 16)	2320	dropout_1[0][0]
conv2d_5 (Conv2D)	(None, 93, 93, 16)	6416	dropout_1[0][0]
conv2d_6 (Conv2D)	(None, 93, 93, 16)	272	dropout_1[0][0]
conv2d_7 (Conv2D)	(None, 93, 93, 16)	12560	dropout_1[0][0]
activation_4 (Activation)	(None, 93, 93, 16)	0	conv2d_4[0][0]
activation_5 (Activation)	(None, 93, 93, 16)	0	conv2d_5[0][0]
activation_6 (Activation)	(None, 93, 93, 16)	0	conv2d_6[0][0]
activation_7 (Activation)	(None, 93, 93, 16)	0	conv2d_7[0][0]
concatenate_1 (Concatenate)	(None, 93, 93, 64)	0	activation_4[0][0] activation_5[0][0] activation_6[0][0] activation_7[0][0]
batch_normalization_5 (BatchNor	(None, 93, 93, 64)	256	concatenate_1[0][0]
max_pooling2d_2 (MaxPooling2D)	(None, 46, 46, 64)	0	batch_normalization_5[0][0]
dropout_2 (Dropout)	(None, 46, 46, 64)	0	max_pooling2d_2[0][0]
conv2d_8 (Conv2D)	(None, 44, 44, 32)	18464	dropout_2[0][0]
activation_8 (Activation)	(None, 44, 44, 32)	0	conv2d_8[0][0]
batch_normalization_6 (BatchNor	(None, 44, 44, 32)	128	activation_8[0][0]
max_pooling2d_3 (MaxPooling2D)	(None, 22, 22, 32)	0	batch_normalization_6[0][0]
dropout_3 (Dropout)	(None, 22, 22, 32)	0	max_pooling2d_3[0][0]
conv2d_9 (Conv2D)	(None, 20, 20, 64)	18496	dropout_3[0][0]
activation_9 (Activation)	(None, 20, 20, 64)	0	conv2d_9[0][0]
batch_normalization_7 (BatchNor	(None, 20, 20, 64)	256	activation_9[0][0]

```
max_pooling2d_4 (MaxPooling2D)  (None, 10, 10, 64)   0           batch_normalization_7[0][0]
_________________________________________________________________________________________________
dropout_4 (Dropout)             (None, 10, 10, 64)   0           max_pooling2d_4[0][0]
_________________________________________________________________________________________________
conv2d_10 (Conv2D)              (None, 8, 8, 128)    73856       dropout_4[0][0]
_________________________________________________________________________________________________
activation_10 (Activation)      (None, 8, 8, 128)    0           conv2d_10[0][0]
_________________________________________________________________________________________________
batch_normalization_8 (BatchNor (None, 8, 8, 128)    512         activation_10[0][0]
_________________________________________________________________________________________________
max_pooling2d_5 (MaxPooling2D)  (None, 4, 4, 128)    0           batch_normalization_8[0][0]
_________________________________________________________________________________________________
dropout_5 (Dropout)             (None, 4, 4, 128)    0           max_pooling2d_5[0][0]
_________________________________________________________________________________________________
flatten_1 (Flatten)             (None, 2048)         0           dropout_5[0][0]
_________________________________________________________________________________________________
dropout_6 (Dropout)             (None, 2048)         0           flatten_1[0][0]
_________________________________________________________________________________________________
dense_1 (Dense)                 (None, 28)           57372       dropout_6[0][0]
_________________________________________________________________________________________________
activation_11 (Activation)      (None, 28)           0           dense_1[0][0]
_________________________________________________________________________________________________
batch_normalization_9 (BatchNor (None, 28)           112         activation_11[0][0]
_________________________________________________________________________________________________
dropout_7 (Dropout)             (None, 28)           0           batch_normalization_9[0][0]
_________________________________________________________________________________________________
dense_2 (Dense)                 (None, 28)           812         dropout_7[0][0]
_________________________________________________________________________________________________
activation_12 (Activation)      (None, 28)           0           dense_2[0][0]
=================================================================================================
Total params: 193,948
Trainable params: 193,246
Non-trainable params: 702
```