Looking for the tallest tree in Taiwan by Deep Learning

1^{st} ~ 4^{th} pi-run summary (Model-1~4)

2021/06/03

YuanFu Yang, Mabel Hsiao

Outline

- Data Exploration
- Data Preprocess
- Feature Extraction
- Methodology
- Experiment Results
- Conclusion & Future Plan

Data Exploration

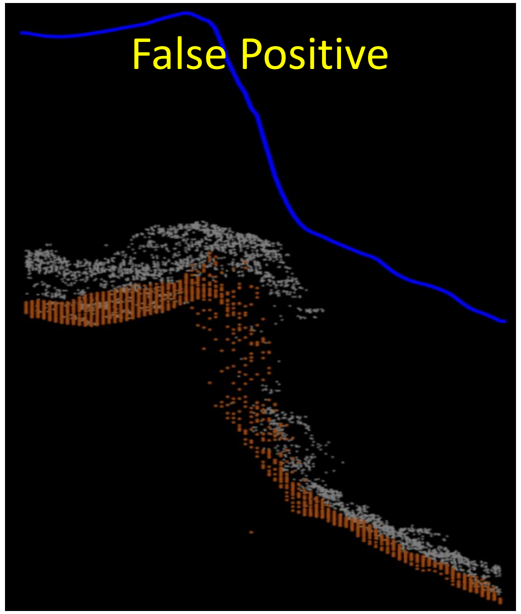
- 目前影像是由群眾外包協助標示座標 (CrowdSourcing)
- **Criteria (Positive rules):**

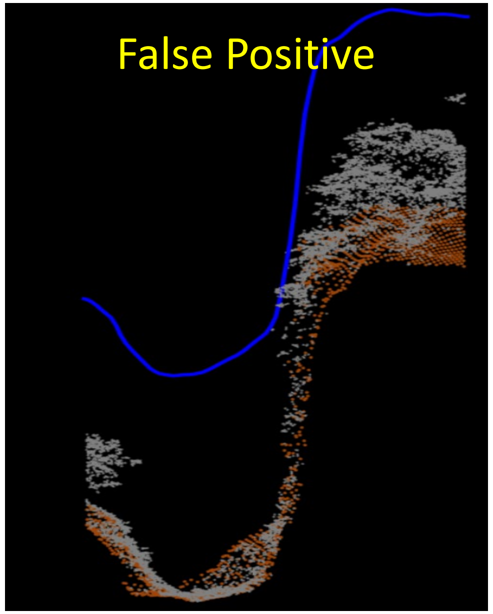
  Valid_count>=3 and Valid_count_zero<=2 ➔ Label “Y”, else “N”

  3個人以上有送出標籤(樹高過基準線, 認為有機會找到高樹), 且2個人以下略過, 沒有送出標籤(認為是電塔等人工建築, 或是認為樹高都很低(低於參考線), 則標籤為“Y”
- **Issue:**

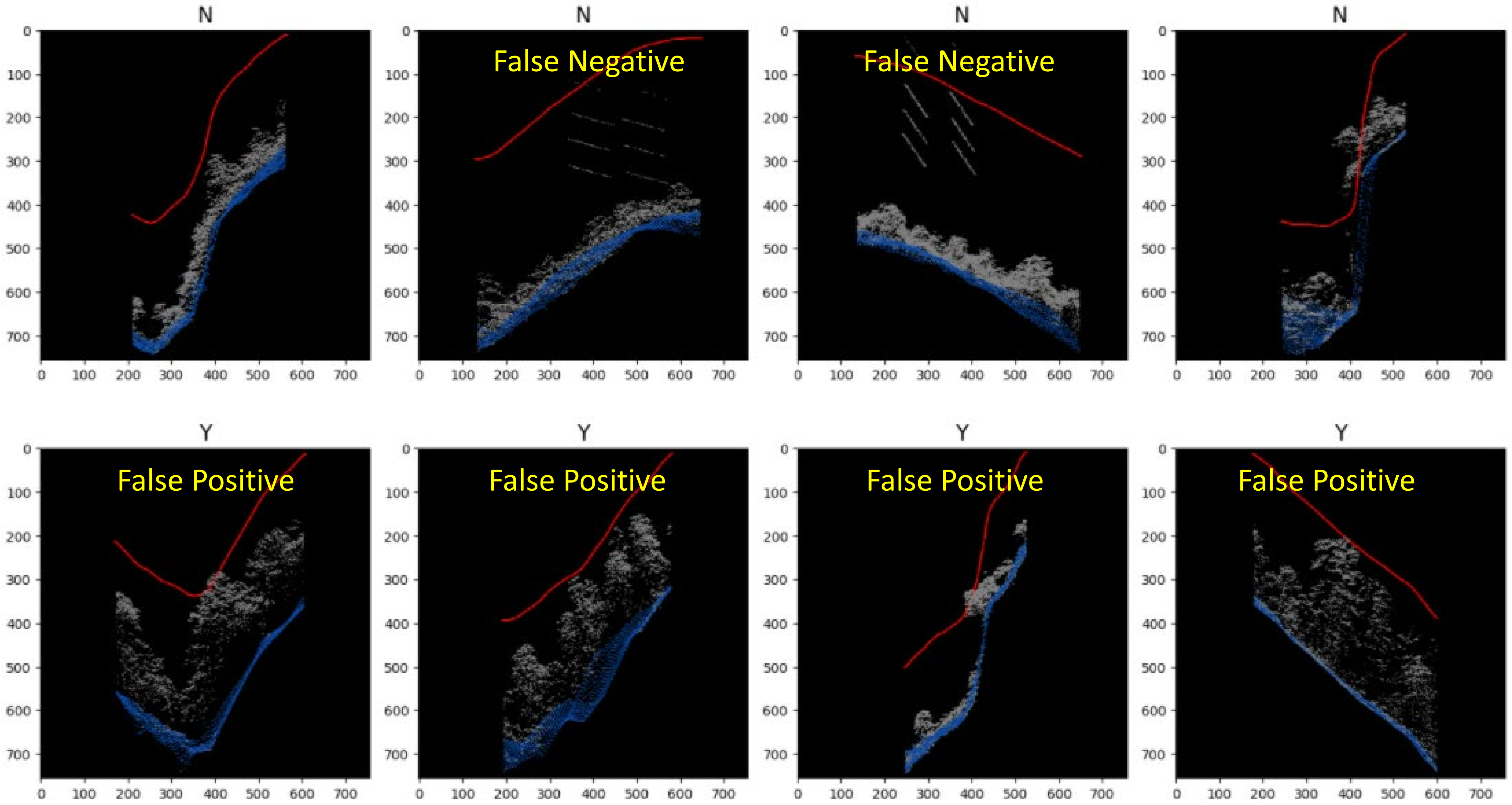
  部分人為標記異常, 如下case, 此圖沒有人工建築干擾, 圖片中只有自然的樹木, 但是沒有標記的比數>有標記的比數. 會被誤Label成 : “ Y “ (有人工建築干擾)

index	id	area	image	valid_ count	valid_ count_zero
41691	51440	4	9520-1-004_001.png	2	3
10845	19418	2	9518-2-062_002.png	3	5





Data Exploration



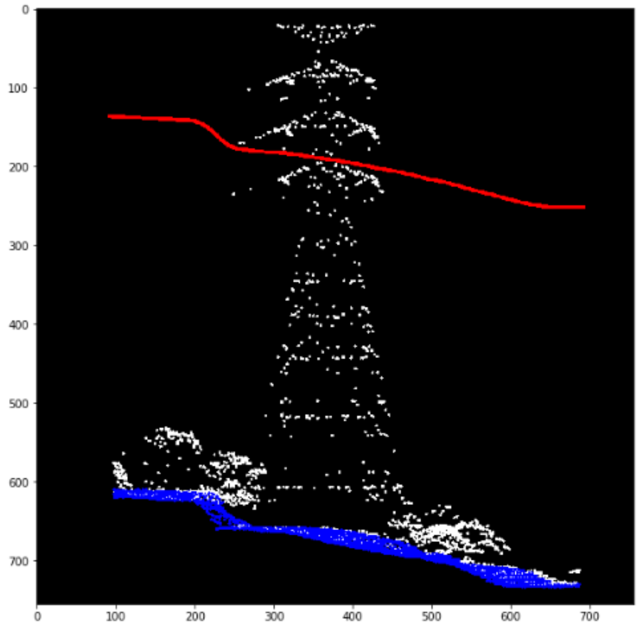
Data Preprocess

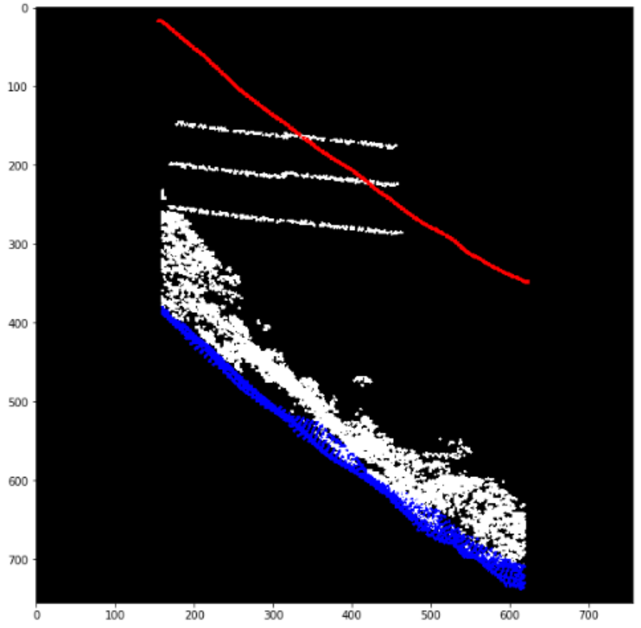
- **Re-Label**:

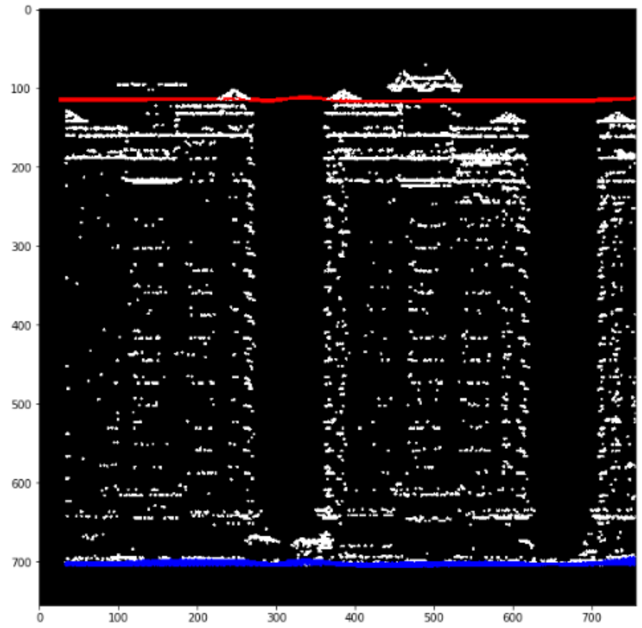
  重新檢視圖片,將人工建築標示出來.

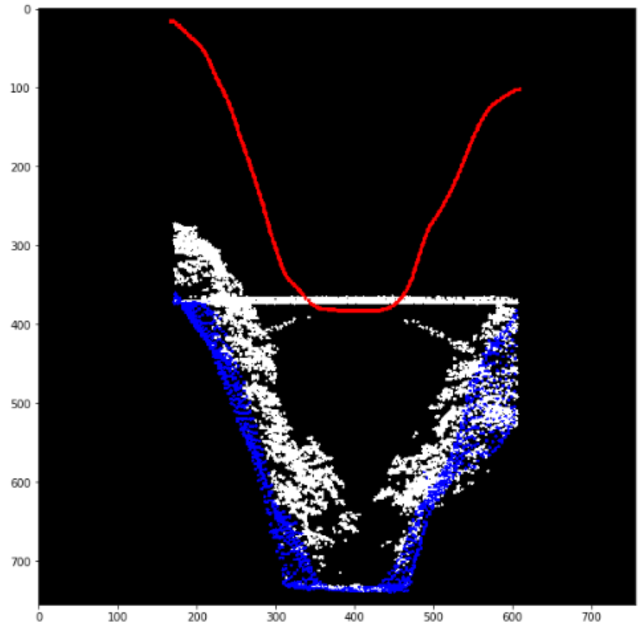
  分類為: 電塔/電纜/大樓/橋梁/風力發電等.

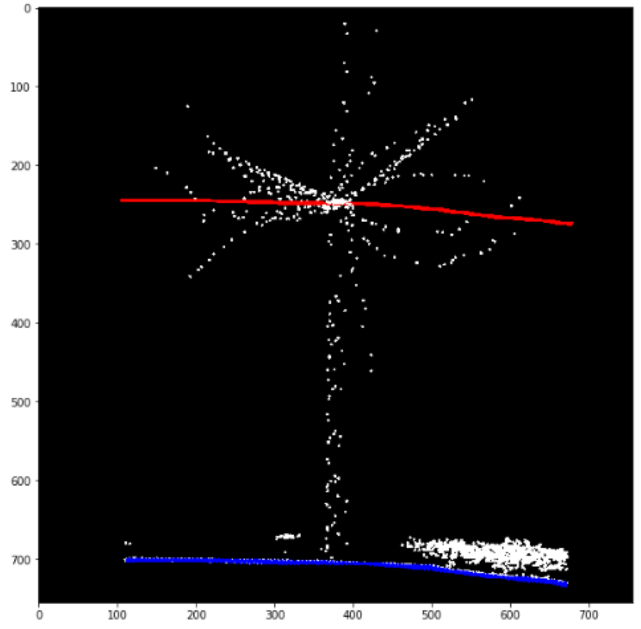
  目前已經完成 25235 張圖片 (46%)











Data Preprocess

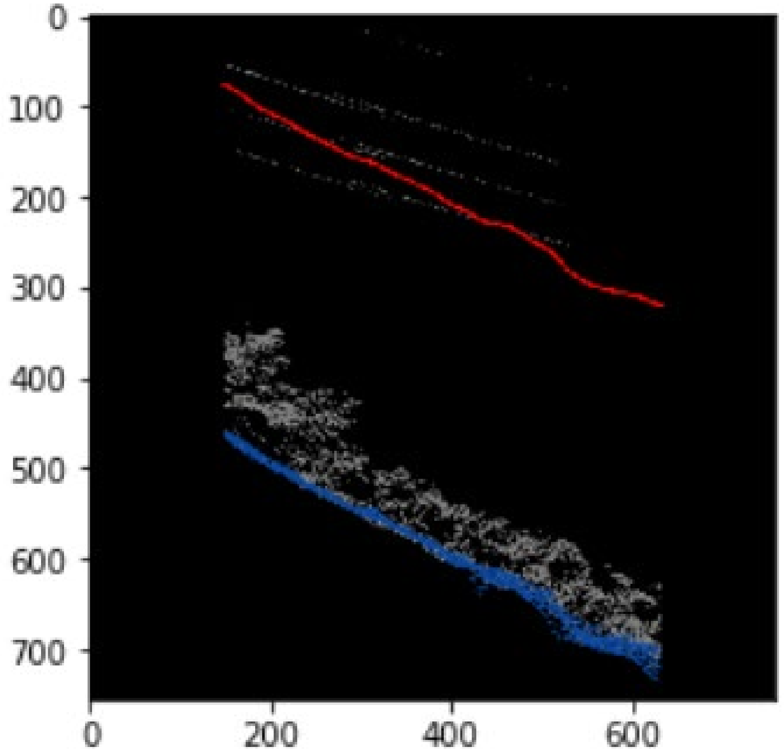
- **Brightness Enhancement**

For DBSCAN, 將Red(天空基線)/Blue(地表基線)/White(物體) 3種顏色進行明亮度增強, 以進行後續DBSCAN.

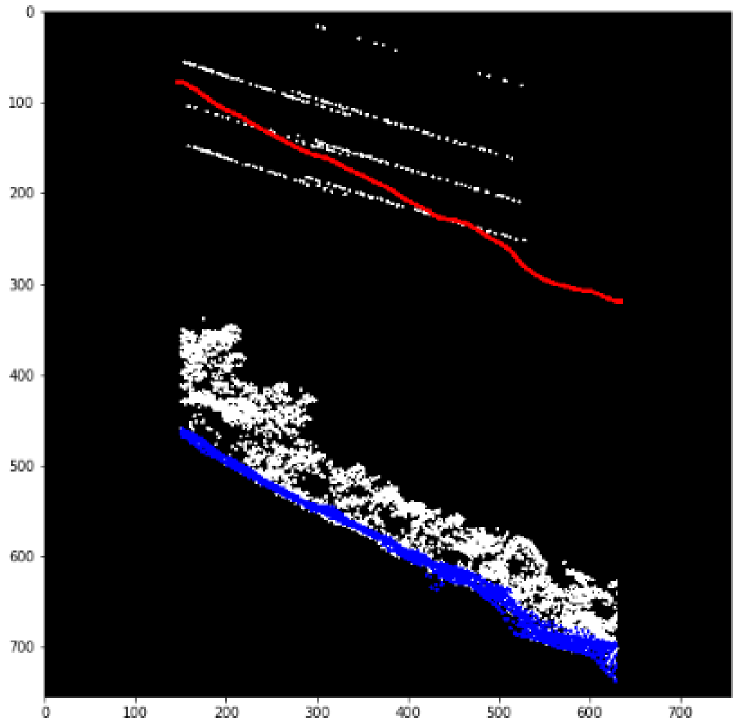
例如: (R/G/B): (101 / 0 / 0) ➔ (255 / 0 / 0)

(0 / 0 / 231) ➔ (0 / 0 / 255)

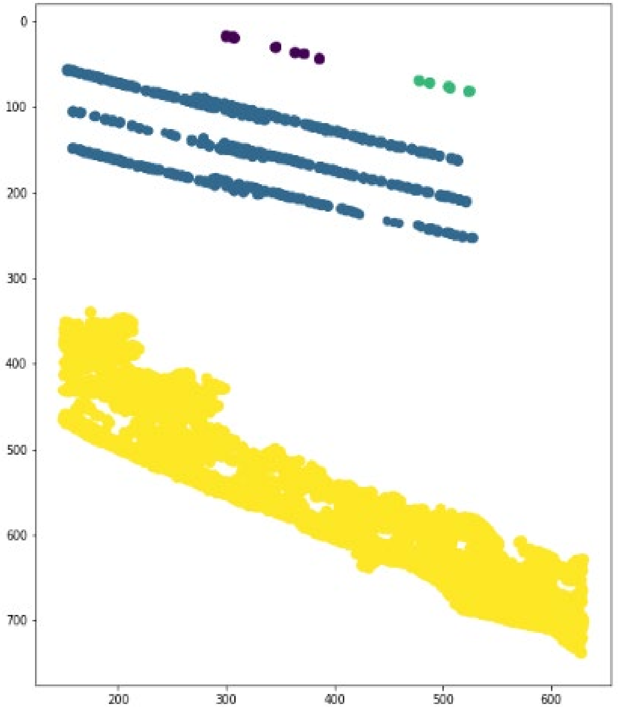
Original Image



Brightness Enhancement



DBSCAN Result

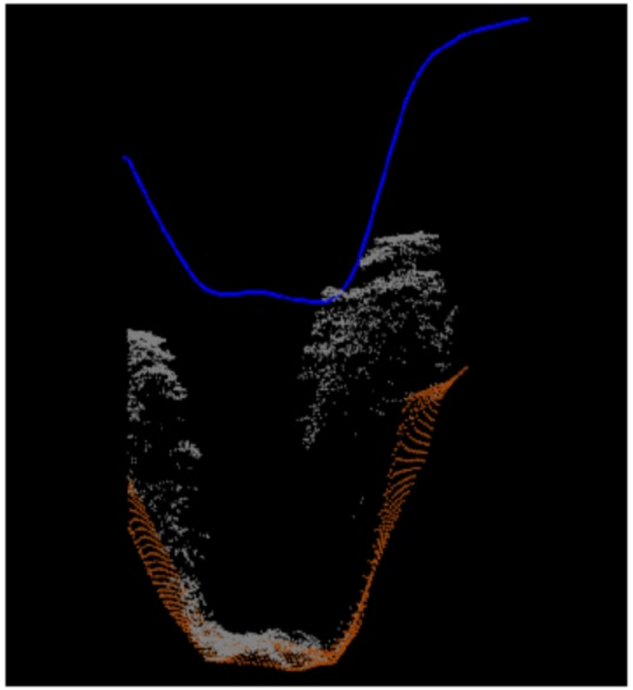


Data Preprocess

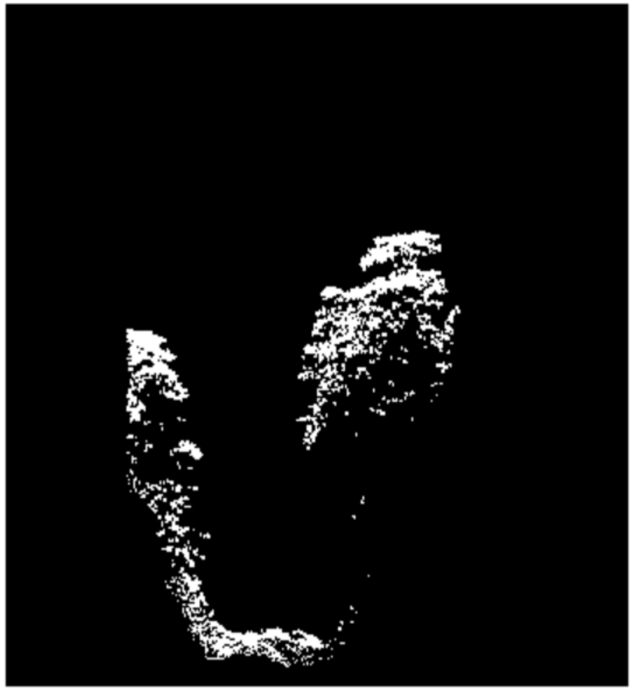
- **Data Preprocess for Deep Learning Input**:

  去除天空與地表基線, 圖片只保留物體. 以輸入Deep Learning Model..

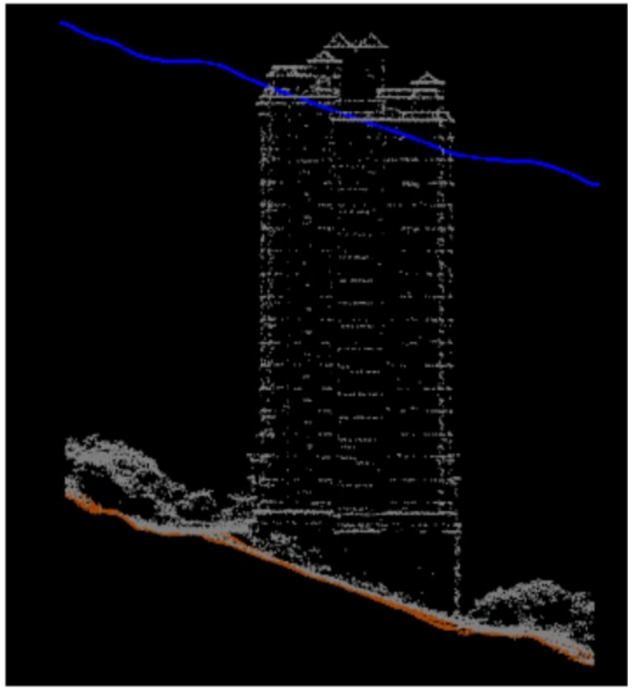
Original Image



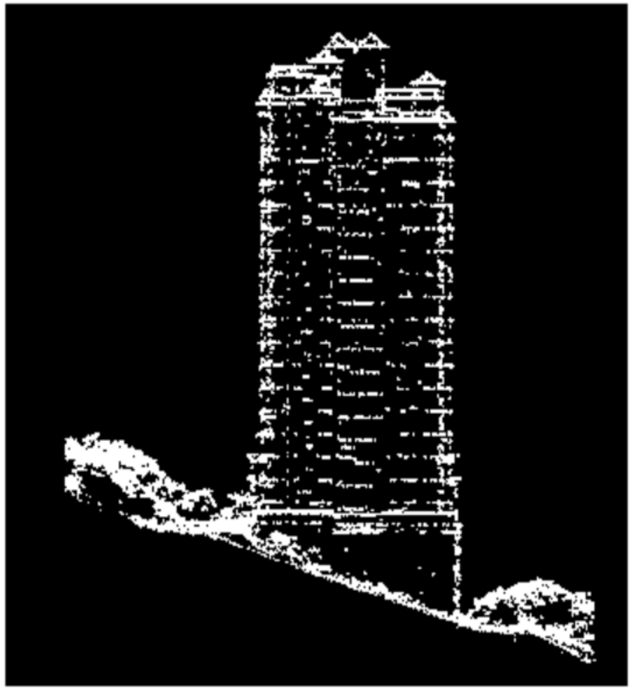
Preprocessed Image



Original Image



Preprocessed Image

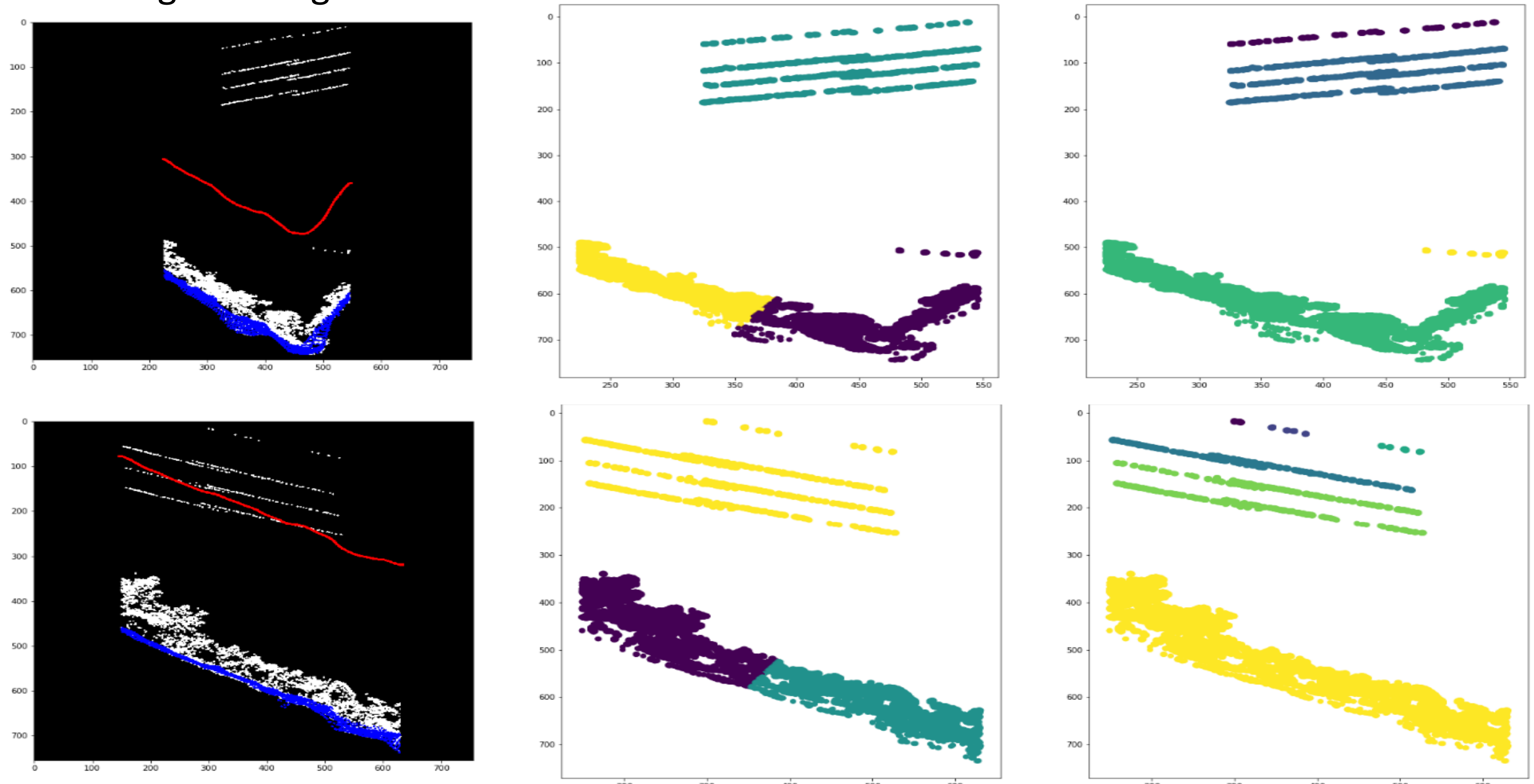


Feature Extraction

- **Unsupervised Learning Study**:

  評估KMean 與 DBSCAN 2種方法, 發現DBSCAN比較適合用來區分電纜與地表物體. 因此採用DBSCAN.

Original Image　　KMean　　DBSCAN Result



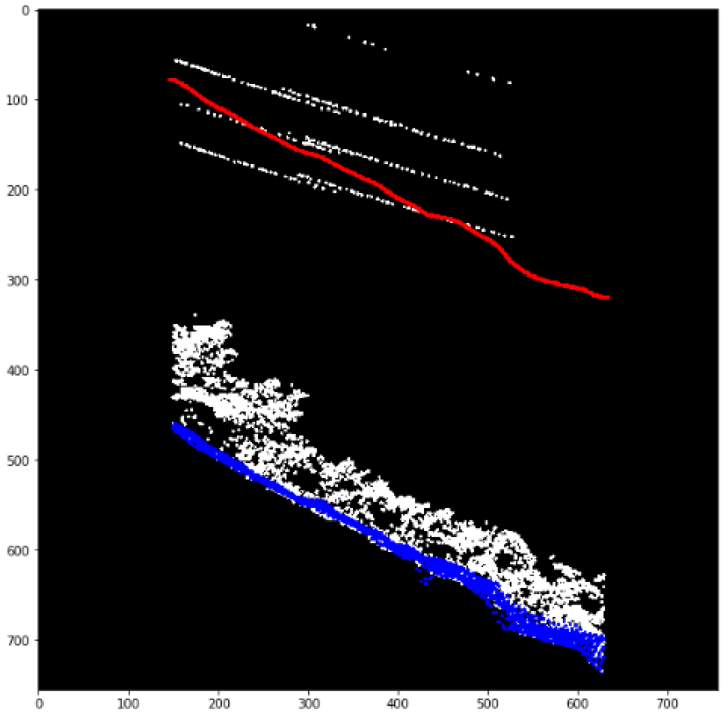
Feature Extraction

- **2-Step DBSCAN**:

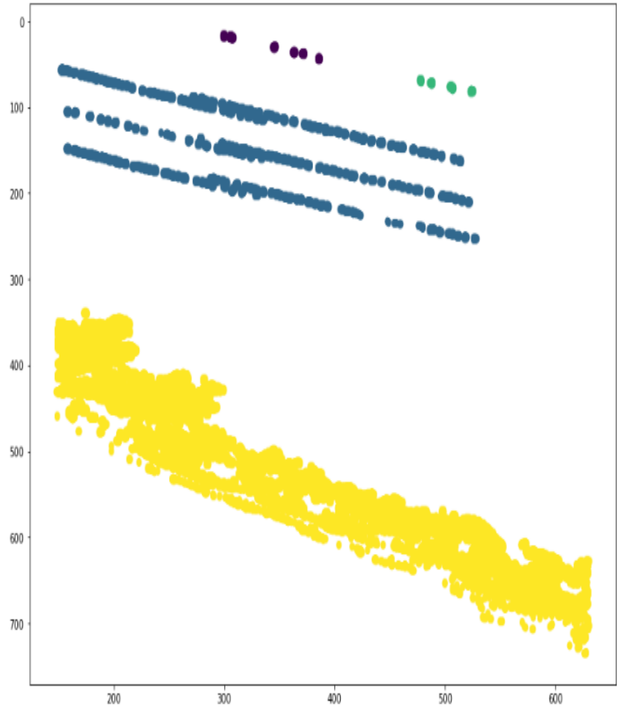
  採用2階段DBSCAN, 第1階段使用較大的eps (粗分), 以過濾地表像素 (如下圖2, 黃色區域) .

  第2階段再針對剩餘群組做分群, 使用較小的eps (細分)(如下圖3).

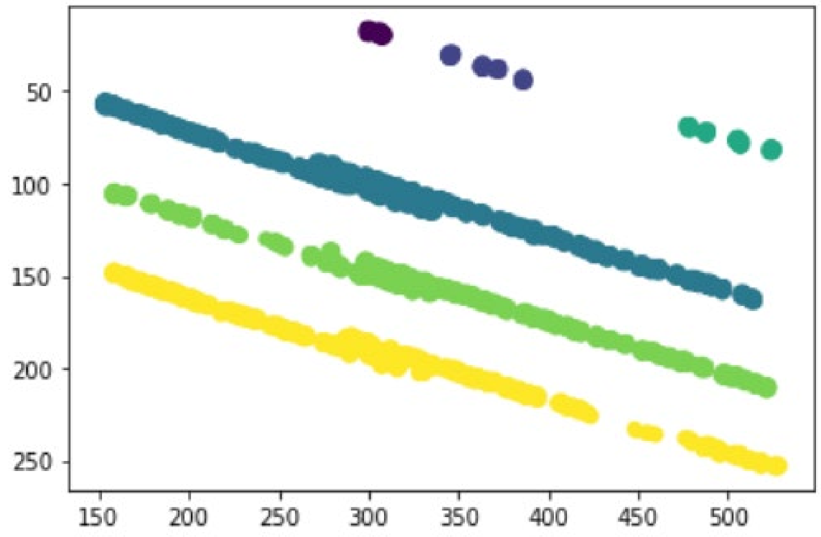
Pic1. Original Image



Pic2. DBSCAN (eps=64)



Pic3. DBSCAN (eps=32)

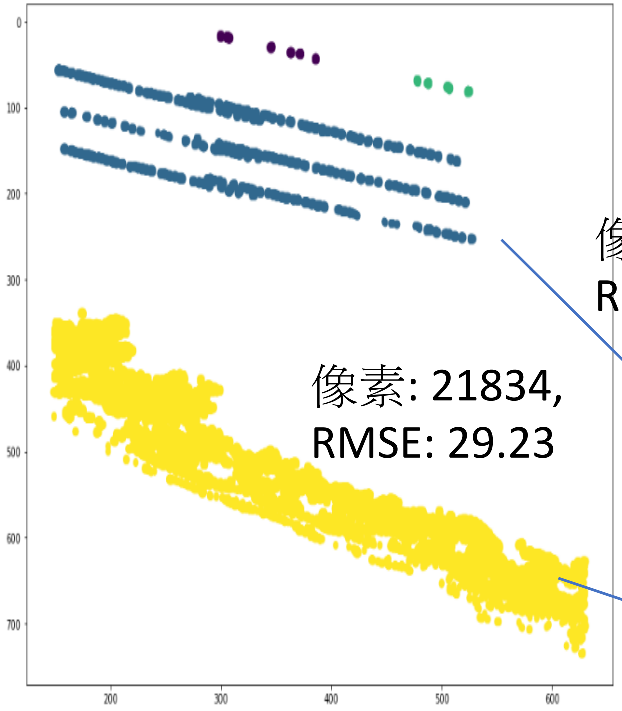


Feature Extraction

- **2-Step DBSCAN**:

使用Linear Regression, 求出第一階段(粗分) 之後各群的RMSE(Root Mean Square Error), 像素數量可以有效用區分地表與空中物體, RMSE則不適合 (如下Case, 地表RMSE=29.32<空中RMSE=37.34).

DBSCAN (eps=64)



像素: 2649,
RMSE: 37.34

```python
L_RMSE=[];LC=[]
for i in set(y_pred):
    Lx=[];Ly=[];count=0
    for j in range(y_pred.shape[0]):
        if(y_pred[j]==i):
            Lx.append(X[j][0]);Ly.append(X[j][1])
            count+=1
    xa=np.array(Lx);ya=np.array(Ly)
    xa=np.reshape(xa,(len(xa),1));ya=np.reshape(ya,(len(ya),1))
    lm = LinearRegression();lm.fit(xa,ya)
    mse=np.mean((lm.predict(xa)-ya)**2)
    rmse = (mse)**0.5
    L_RMSE.append(rmse)
    LC.append(count)
    print(xa.shape[0],rmse)
```

```
76 1.1690828731291767
2649 37.34764691015677
53 1.1128124481487505
21834 29.238288189594307
```

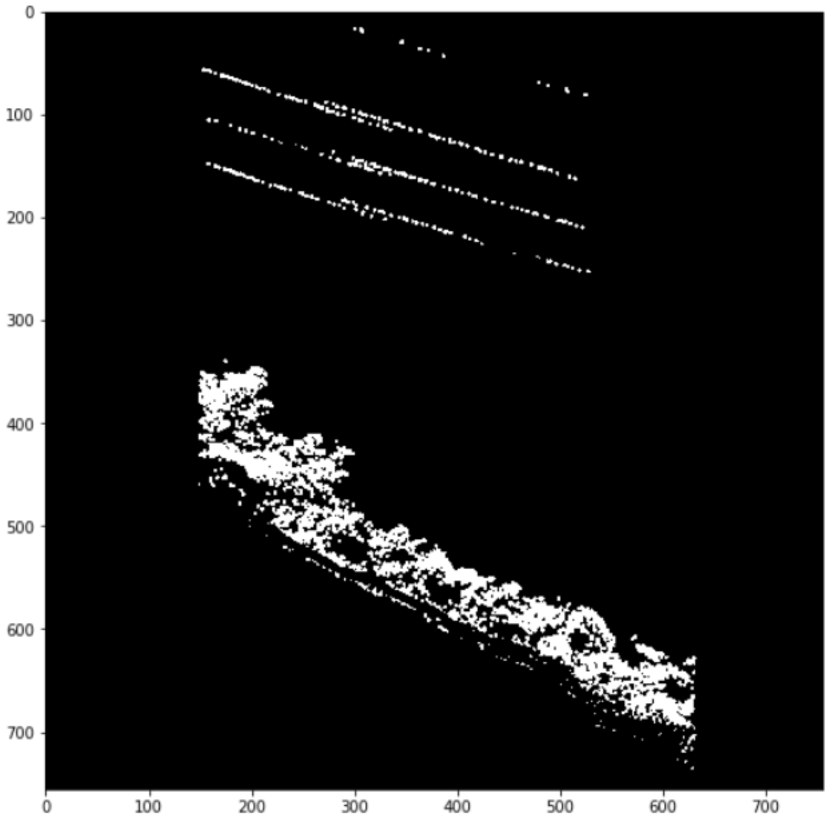
Feature Extraction

- **Hough Transform**

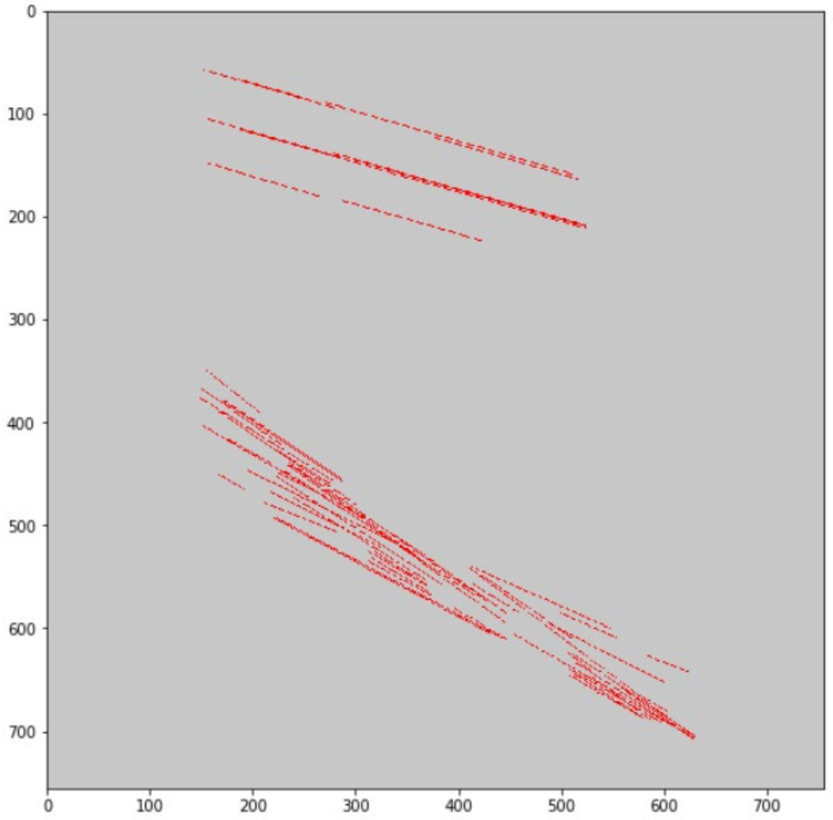
使用Hough Transform進行Line Detection, 無論何種Hyper-parameter, 都會將地表的像素們串聯再一起.

➔ 效果不好, 不予採用.

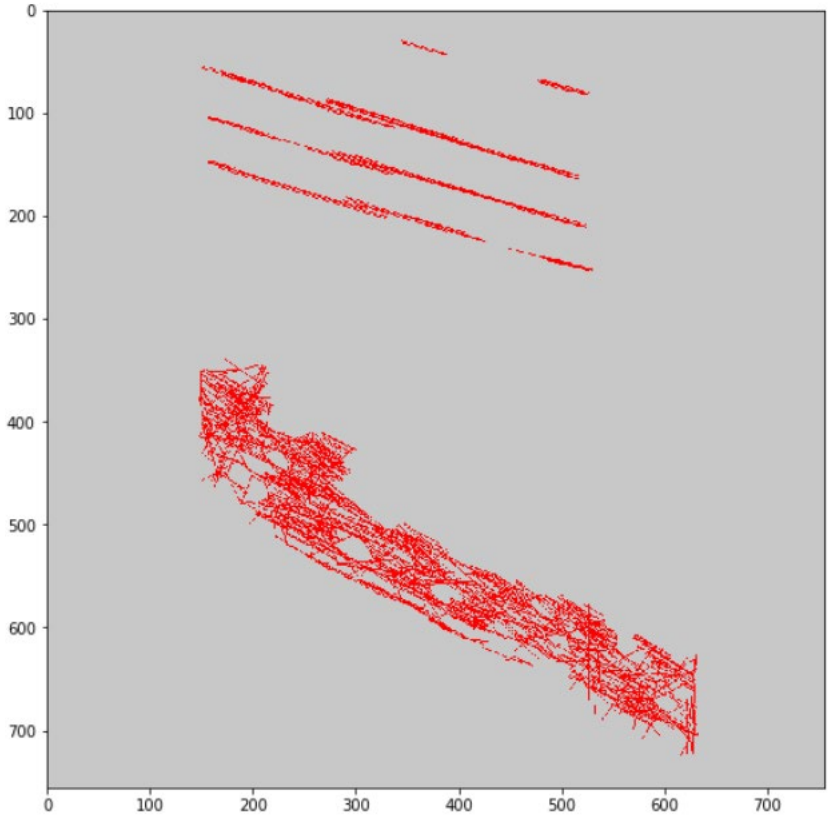
Grayscale



threshold = 10



threshold = 100



Methodology – Model Design

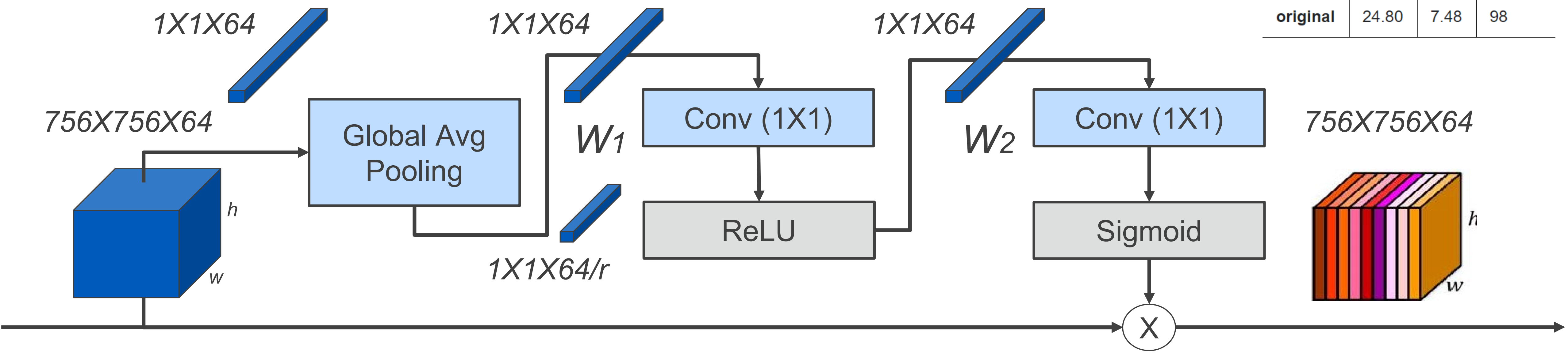
Model	Convolution	Cannel-wise Attention	Self-Attention	Regularization	Activation Function	Flatten Layer	Loss Function	Learning Rate	Optimizer
Model-1/2	Traindiitional Conv. (LeNet)[1]	N/A	N/A	N/A	ReLU (AlexNet)[2]	Fully Connection (LeNet)[1]	Cross-Entropy	Fixed LR	Stochastic Gradient Decent (SGD)
Model-3	Traindiitional Conv. (LeNet)[1]	N/A	N/A	N/A	ReLU (AlexNet)[2]	Fully Connection (LeNet)[1]	Cross-Entropy	LR decay	Adaptive Moment Estimation (Adam)
Model-4	Deep Residual Representation (ResNet)[3]	Squeeze and Excitation (SENet)[4]	N/A	L2 Regularization	ReLU (AlexNet)[2]	Global Average Pooling (NIN)[5]	Cross-Entropy	Cosine LR Anneal [12]	Nesterov Accelerated Gradient (NAG)
Model-5	Self-Proliferating (SPNet)[10]	Squeeze and Excitation (SENet)[4]	N/A	Smooth-L2 Regularization	ReLU6 (MobileNetV1)[7]	Conv. 1X1 (VGG16)[6]	Circle-Loss (CVPR'20)[11]	Cosine LR Anneal [12]	Nesterov Accelerated Gradient (NAG)
Model-6	Self-Proliferating (SP&A Net)	Squeeze and Excitation (SENet)[4]	Global Context Block (GCNet)[9]	Smooth-L2 Regularization	Hard version of Swish (MobileNetV3)[8]	Conv. 1X1 (VGG16)[6]	Circle-Loss (CVPR'20)[11]	Cosine LR Anneal [12]	Nesterov Accelerated Gradient (NAG)

Methodology – Squeeze and Excitation

- **Channel-wise attention** *[4] Squeeze-and-Excitation Networks, Jie Hu et al., CVPR'18*

Ratio r	top-1 err.	top-5 err.	model size (MB)
4	23.21	6.63	137
8	23.19	6.64	117
16	23.29	6.62	108
32	23.40	6.77	103
original	24.80	7.48	98

(a) Context Modeling *(b) Bottleneck Transform* *(c) Fusion*



(a) Context Modeling

$$z_c = \frac{1}{HXW}\sum_{i=1}^{H}\sum_{j=1}^{W} u_c(i,j)$$

(b) Bottleneck Transform

$$s_c = \sigma(W_2 s) = \sigma\big(W_2 \delta(W_1 z)\big)$$

(c) Fusion

$$y_c = F(x_c, s_c) = s_c \cdot x_c$$

Methodology – Cosine LR Anneal

- **Cosine LR Decay**

$$T_{cur} = \min(global\_step, decay\_steps)$$

$$\eta_t = 0.5 \times (1 + \cos(\pi \times \frac{T_{cur}}{T_i}))$$

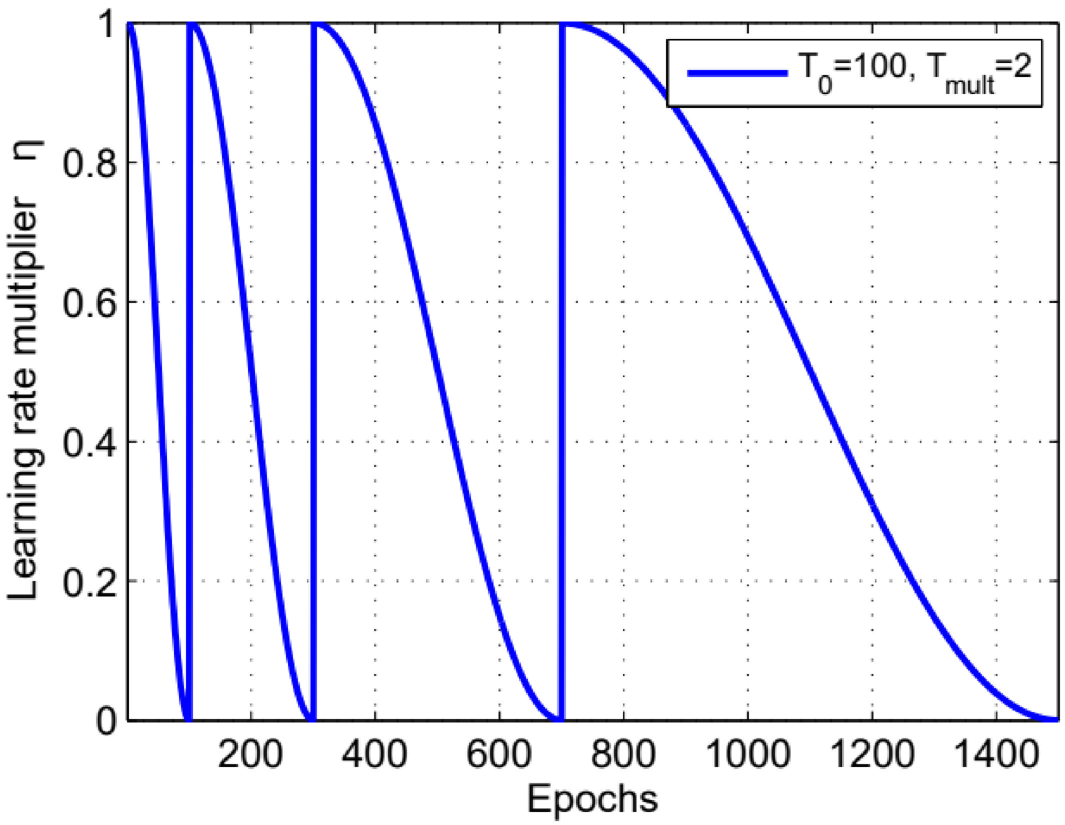
$$decayed = (1 - \alpha) \times \eta_t + \alpha$$

$$decayed\_learning\_rate = \text{learning\_rate} \times decayed$$

$where$:

T_{cur}: $global\ step,\ T_i$: $decay\ step$

η_t: cosine $learning\ rate$ decay



[12] I. Loshchilov, F. Hutter, "SGDR: Stochastic Gradient Descent with Warm Restarts", International Conference on Learning Representations (ICLR), May 2017.

Methodology - Architecture

Ps: Each SE-ResNet has 3 Conv. layers, 2 Batch Normalization and 1 SE Block.

Model-1/2: simple CNN

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 64, 64, 128)	3584
conv2d_2 (Conv2D)	(None, 64, 64, 128)	147584
conv2d_3 (Conv2D)	(None, 64, 64, 128)	147584
max_pooling2d_1 (MaxPooling2	(None, 32, 32, 128)	0
dropout_1 (Dropout)	(None, 32, 32, 128)	0
conv2d_4 (Conv2D)	(None, 32, 32, 256)	295168
conv2d_5 (Conv2D)	(None, 32, 32, 256)	590080
conv2d_6 (Conv2D)	(None, 32, 32, 256)	590080
⋮		
flatten_1 (Flatten)	(None, 32768)	0
dense_1 (Dense)	(None, 256)	8388864
dropout_4 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 128)	32896
dropout_5 (Dropout)	(None, 128)	0
dense_3 (Dense)	(None, 2)	258

Total params: 16,095,874
Trainable params: 16,095,874
Non-trainable params: 0

Model-3: simple CNN

Layer (type)	Output Shape	Param #
conv2d_1 (Conv2D)	(None, 756, 756, 32)	896
conv2d_2 (Conv2D)	(None, 756, 756, 32)	9248
conv2d_3 (Conv2D)	(None, 756, 756, 32)	9248
max_pooling2d_1 (MaxPooling2	(None, 378, 378, 32)	0
dropout_1 (Dropout)	(None, 378, 378, 32)	0
conv2d_4 (Conv2D)	(None, 378, 378, 64)	18496
conv2d_5 (Conv2D)	(None, 378, 378, 64)	36928
conv2d_6 (Conv2D)	(None, 378, 378, 64)	36928
⋮		
flatten_1 (Flatten)	(None, 1131008)	0
dense_1 (Dense)	(None, 256)	289538304
dropout_4 (Dropout)	(None, 256)	0
dense_2 (Dense)	(None, 128)	32896
dropout_5 (Dropout)	(None, 128)	0
dense_3 (Dense)	(None, 1)	129

Total params: 290,052,097
Trainable params: 290,052,097
Non-trainable params: 0

9 Conv. layer + 3 Max-Pooling + 1 Flatten + 2 dense layers

Model-4: SE-ResNet

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	(None, 756, 756, 3)	0	
conv2d_1 (Conv2D)	(None, 378, 378, 64)	9408	input_1[0][0]
max_pooling2d_1 (MaxPooling2D)	(None, 189, 189, 64)	0	conv2d_1[0][0]
batch_normalization_1 (BatchNor	(None, 189, 189, 64)	256	max_pooling2d_1[0][0]
activation_1 (Activation)	(None, 189, 189, 64)	0	batch_normalization_1[0][0]
conv2d_2 (Conv2D)	(None, 189, 189, 64)	36864	activation_1[0][0]
batch_normalization_2 (BatchNor	(None, 189, 189, 64)	256	conv2d_2[0][0]
⋮			
multiply_6 (Multiply)	(None, 24, 24, 512)	0	conv2d_16[0][0] dense_12[0][0]
add_6 (Add)	(None, 24, 24, 512)	0	multiply_6[0][0] add_5[0][0]
batch_normalization_13 (BatchNo	(None, 24, 24, 512)	2048	add_6[0][0]
activation_13 (Activation)	(None, 24, 24, 512)	0	batch_normalization_13[0][0]
global_average_pooling2d_7 (Glo	(None, 512)	0	activation_13[0][0]
dense_13 (Dense)	(None, 1)	512	global_average_pooling2d_7[0][0]

Total params: 9,782,208
Trainable params: 9,775,936
Non-trainable params: 6,272

1 Conv. Layer + 6 SE-ResNet + 1 Global AVG Pooling + 1 dense layers

Experiment Result – Metrics Introduction

Accuracy = (TP+TN) / (TP+TN+FP+FN)

Recall = TP / (TP+FN)

Covid-19: **The sick were misdiagnosed as not sick** ➔ Epidemic out of control

TaiwanTree: The tower or building had not been found ➔ Quality out of control

Precision = TP / (TP+FP)

Covid-19: **No sick were misdiagnosed as sick** ➔ The healthcare system collapsed

TaiwanTree: Only images of trees are mistaken for tower or building ➔ Redundant work

Take harmonic average of Recall and Precision

F1-Score $= 2 \times \frac{Recall \times Precision}{Recall + Precision}$ ➔

Simultaneous control of False Negatives and False Positives
Simultaneous control of Quality and People Productivity

		Predict	
		Positive	Negative
Actual	disease	true positive (TP)	**false negative (FN)**
	no disease	**false positive (FP)**	true negative (TN)

Experiment Result

Test Result

Pi-run	Model-1	Model-2	Model-3	Model-4	Model-6	Model-7
Hardware	GeForce MX100	Tesla P100	Tesla P100	Tesla P100		
Data	All	46% (25235)	46% (25235)	46% (25235)	46% (25235)	100% (54415)
Image壓縮	Y (756X756) to (64X64)	Y (756X756) to (64X64)	N (756X756)	N (756X756)	N (756X756)	N (756X756)
Data Preprocess	Normarlization	Normarlization	Normarlization + Filter ground and sky baseline	Normarlization + Filter ground and sky baseline	Normarlization + Filter ground and sky baseline	Normarlization + Filter ground and sky baseline
Label	CrowdSourcing	DBSCAN+Expert	DBSCAN+Expert	DBSCAN+Expert	DBSCAN+Expert	DBSCAN+Expert
Model	simple-CNN	simple-CNN	simple-CNN	SE-ResNet	SPNet	SP&A Net
#Parameter	16,095,874	16,095,874	290,052,097	9,782,208		
Accuracy	73.75%	95.06%	96.95%	98.06%		
Precision (FP)	38.46%	91.14%	94.27%	96.39%		
Recall (FN)	92.59%	99.77%	99.92%	99.84%		
F-score	54.35%	95.26%	97.02%	98.08%		
Improve Plan		DBSCAN Implement (Martin Ester et al., 1996)	Data Preprocess	ResNet (K. He et al., CVPR'16) + Squeeze and Excitation (J. Hu et al., CVPR'18)	SPNet (Y.F. Yang and M. Sun, ASMC'21)	SP&A Net (Y.F. Yang and M. Sun, 2021)
Detail Action		(1)加入DBSCAN協助偵測高空電纜與其他懸空的人工設施 (2) Re-Label dataset	(1) Deep Learning input 過濾地表基線與空中參考線 (2) 加入Early Stop, 存取最佳模型 (連續10 epoch Val loss沒有降低則停止) (3) Fixed LR改成LR decay (0.0004 per epoch) (4) Optimizer 從SGD改成Adam (5) GPU memory over loading issue, min batch size 32 to 4	(1) 加入Squeeze and Excitation (2) 加入殘差學習 (3) Global AVG Pooling 取代Flatten Layer, 減少參數 (4) 加入Batchnormalization, 移除Drop out (5) 加入L2 Regularization (6) Optimizer 從Adam改成Nadam (7) 改成Cosine LR decay (循環式學習率衰減)		

Training Result

- Model-1 (1st pi-run):

  Training Acc: 83.53%

  Validation Acc: 75.63%

- Model-2 (2nd pi-run):

  Training Acc: 99.05%

  Validation Acc: 98.75%

- Model-3 (3rd pi-run):

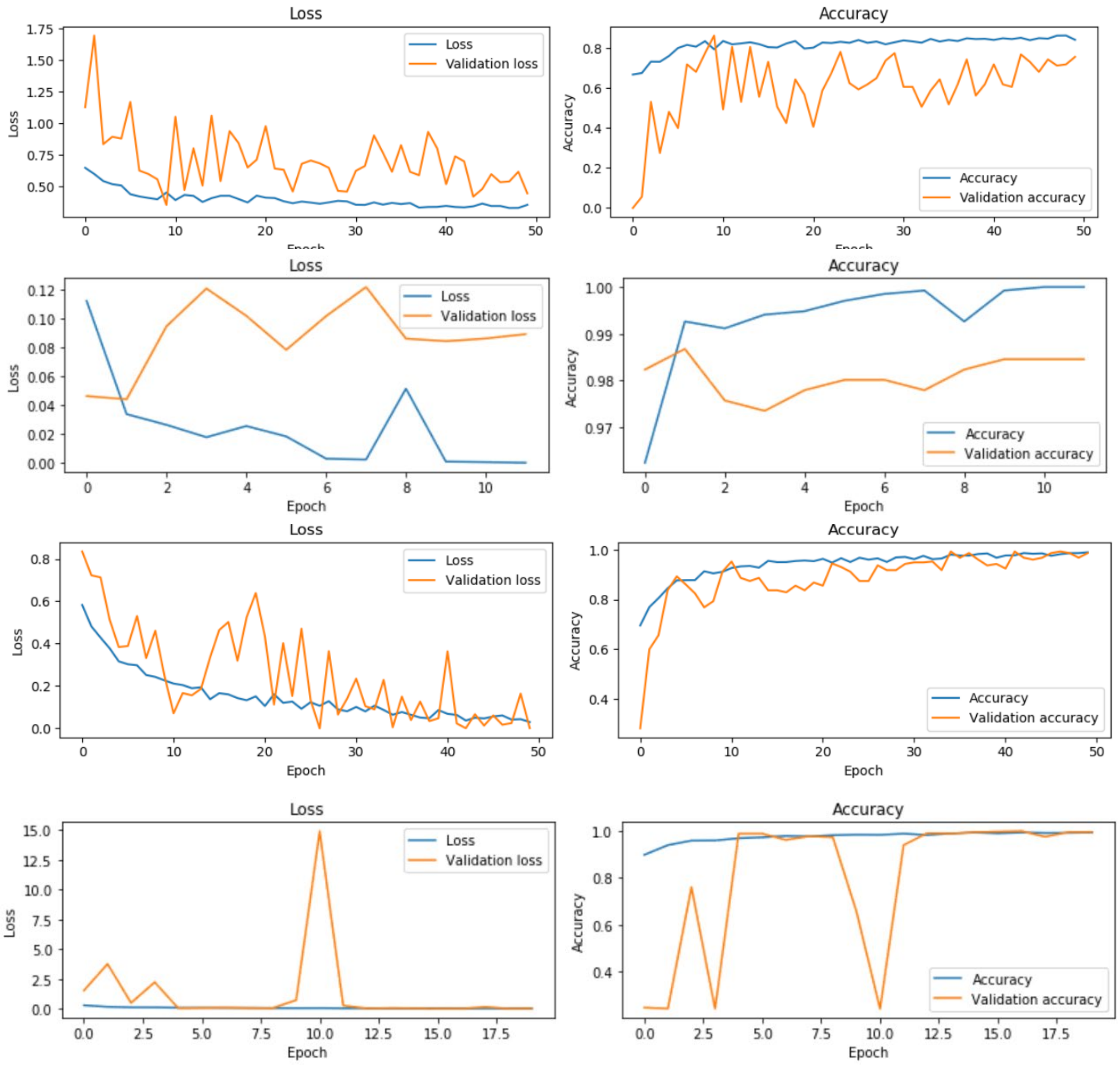
  Training Acc: 100%

  Validation Acc: 98.45%

- Model-4 (4th pi-run):

  Training Acc: 99.19%

  Validation Acc: 99.34%



Test Result: Model-3

		Predict	
		N	Y
Actual	N	Ture Negative	False Positive
	Y	False Negative	True Positive

- DBSCAN Classification Result:

  False Negative: **596**, False Positive: **469**

		Predict	
		N	Y
Actual	N	12253	469
	Y	596	11917

Precision	96.21%
Recall	95.24%
F-Score	95.72%
Accuracy	95.78%

- Deep Learning Classification Result:

  False Negative: **10**, False Positive: **760**

		Predict	
		N	Y
Actual	N	11952	760
	Y	10	12513

Precision	94.27%
Recall	99.92%
F-Score	97.02%
Accuracy	96.95%

Test Result: Model-4

		Predict	
		N	Y
Actual	N	Ture Negative	False Positive
	Y	False Negative	True Positive

- DBSCAN Classification Result:

  False Negative: **596**, False Positive: **469**

		Predict	
		N	Y
Actual	N	12253	469
	Y	596	11917

Precision	96.21%
Recall	95.24%
F-Score	95.72%
Accuracy	95.78%

- Deep Learning Classification Result:

  False Negative: **20**, False Positive: **469**

		Predict	
		N	Y
Actual	N	12233	469
	Y	20	12513

Precision	96.39%
Recall	99.84%
F-Score	98.08%
Accuracy	98.06%

Test Result: Model-4

- False Negative Analysis:

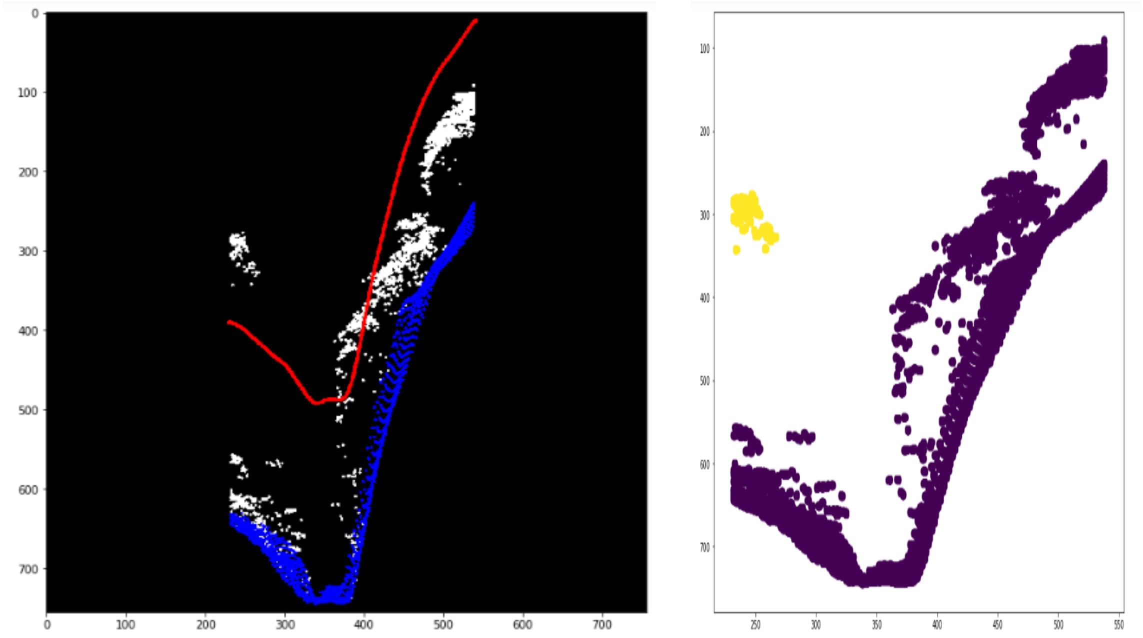
  不明空中物體: **316**, 基準線異常: **153**

Predict

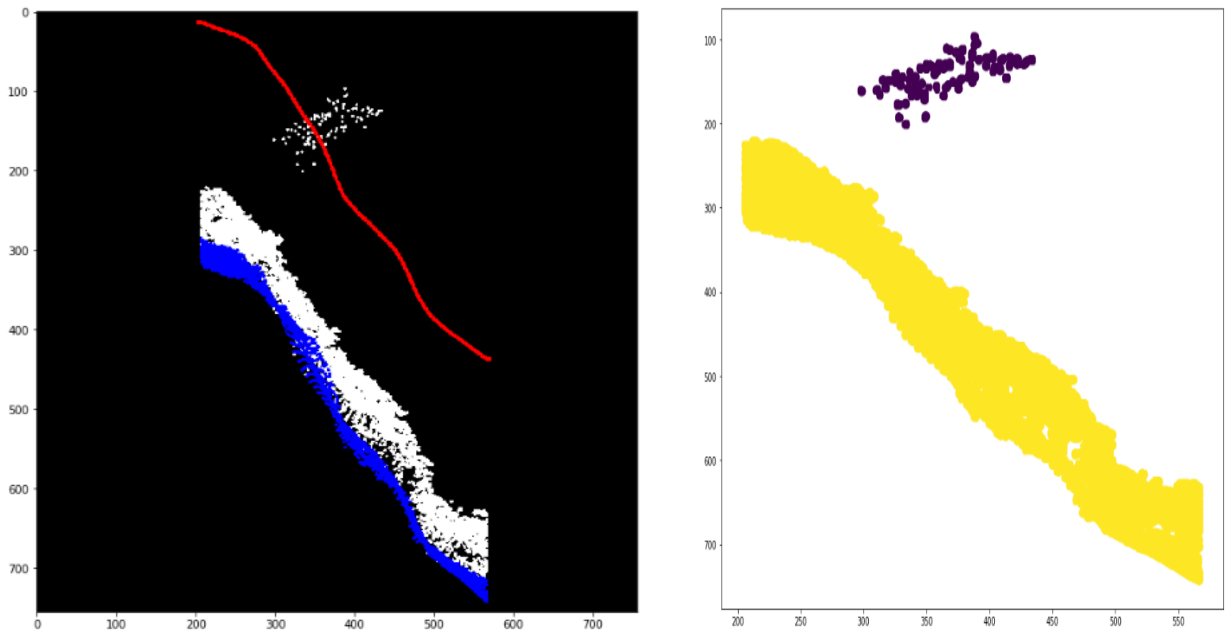
Actual

	N	Y
N	12233	469
Y	20	12513

疑似旁邊橫向生長的樹群:



疑似一群鳥



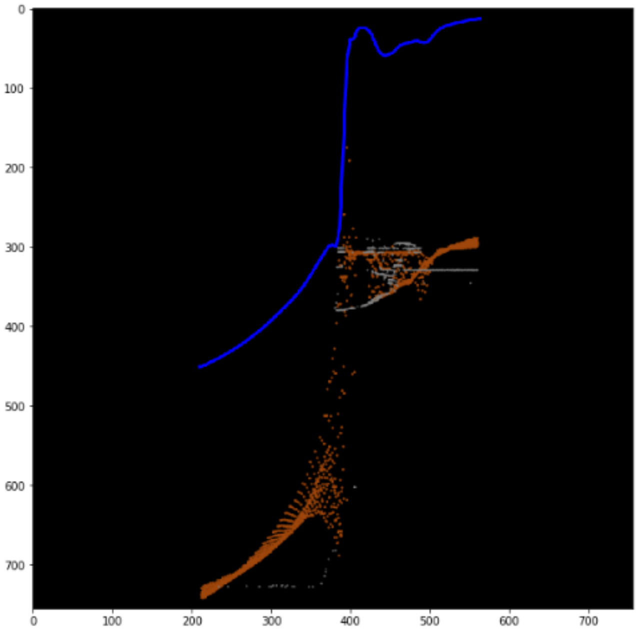
Test Result: Model-4

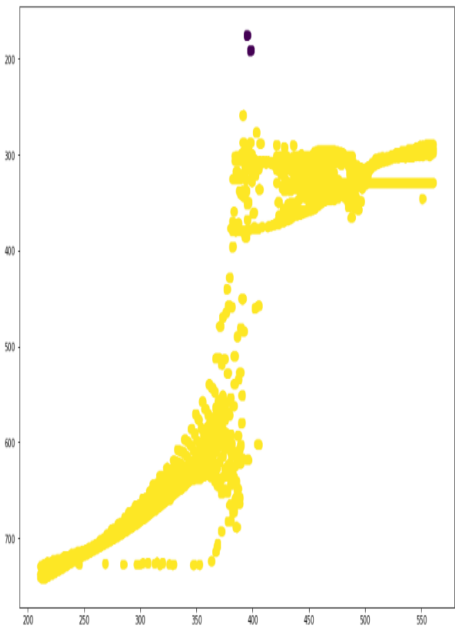
- False Negative Analysis:

  不明空中物體: 316, **基準線異常: 153**

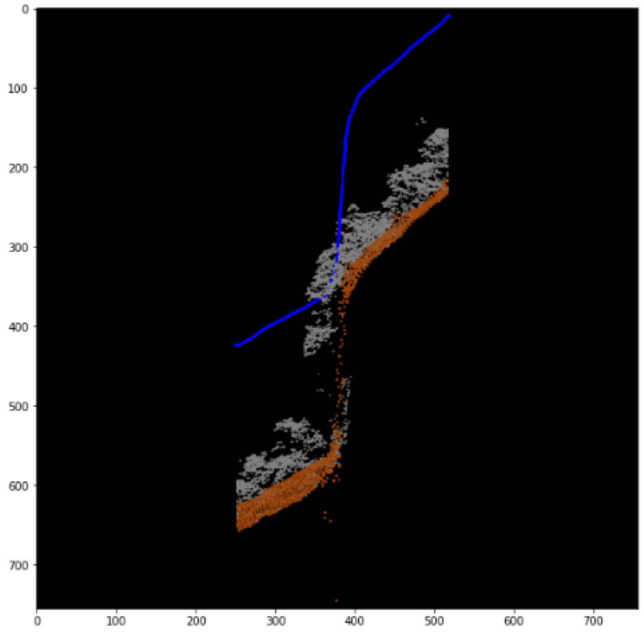
		Predict	
		N	Y
Actual	N	12233	469
	Y	20	12513

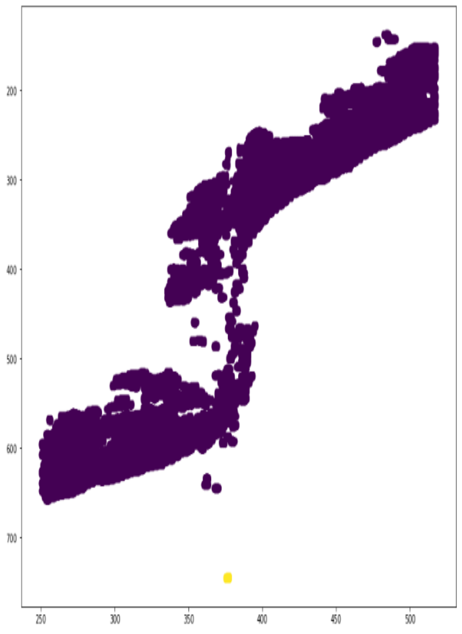
上方飄高異常的地面基準點:





下方異常的地面基準點:





Conclusion & Future Plan

- DBSCAN + Deep Learning 可以有效提高準確度, 在model-2/3/4 中分別降低False Negative 次數至 29/10/20筆, Recall Rate分別提高到 99.77%/99.92%/99.84% (model-1: 92.59%)
- Model-3/4中大幅改善Precision Rate (降低False Positive), Precision Rate 分別提高至94.27%/96.39% (model-1/2: 38.46%/91.14%). 主要改善策略:

(1) 取消圖片壓縮size, 恢復到 756X756 (但訓練時間也會增加1~2倍).

(2) 因應以上, 在model-4中, 平坦層採用Global Average Pooling [5], 以降低模型參數量.

(3) Deep Learning 圖像移除天空基準線與地表基線, 降低不必要的feature map, 避免overfitting.

(4) 採用ResNet 殘差學習 [3], 避免 gradient vanishing problem.

(5) Implement Batch Normalization [13],避免 gradient vanishing problem.

(6) Implement Squeeze and Excitation Block [4], 以選擇重要的feature map, 提高判斷準確度.

(7) Model-3 將Learning Rate 從固定式(fixed RL) 改成 梯度衰減式LR,同時兼顧訓練時間與準確度 .

(8) 承上, Model-4 進一步改成更好的學習方式: 循環式衰減Learning Rate - Cosine Learning Rate Anneal.

(9) Convolution過程中採用L2 Regularization, 加快學習速度與提高精準度.

(10) Optimization 改成 Adaptive Moment Estimation (Adam) 與 Nesterov Accelerated Gradient (NAG), 以加快學習.

Conclusion & Future Plan

- Future Plan:
  (1) 採用Self-Attention [9]: 圖片解析度是756X756, 其幾何結構與各pixel 之間彼此可能有相關性存在, 因此加入Position-wise Attention 會改善模型能力. 另外加入Channel-wise Attention 可以幫助模型學習特徵重要性.
  (2) Implement Self-Proliferator Block [10]: 加入Self-Proliferator 可以降低模型參數量, 改善學習效率.
  (3) Implement Inverted Residual [14]: 參考MobileNetV2, 加入Inverted Residual, 進一步改善梯度消失問題.
  (4) Implement Circle-Loss [11]: 採用Circle-Loss 取代傳統的Cross-Entropy, 以提高準確度.
  (5) Activation Function 改成 ReLU6 [7] 與 Hard version of Swish [8], 改善convolution.
  (6) Flatten Layer 改成 Conv. 1X1 [6], 降低參數量與改善feature extraction.
  (7) 擴充Dataset (46% ➔ 100%, 25235 ➔ 54415).

References

[1] Y. Lecun, L. Bottou, Y. Bengio and P. Haffner, "Gradient-based learning applied to document recognition", in Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, Nov. 1998.

[2] A. Krizhevsky, I. Sutskever, G. E. Hinton, "ImageNet classification with deep convolutional neural networks", Advances in Neural Information Processing Systems (NIPS), 2012.

[3] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition", 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 770-778, Jun. 2016.

[4] J. Hu, L. Shen and G. Sun, "Squeeze-and-Excitation Networks", 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7132-7141, Jun. 2018.

[5] M. Lin, Q. Chen, S. Yan, "Network In Network", International Conference on Learning Representations (ICLR) 2014.

[6] K. Simonyan, A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition", International Conference on Learning Representations(ICLR), 2015.

[7] A. G. Howard, M. Zhu, B. Chen, et al., "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications", arXiv preprint arXiv:1704.04861, Apr 2017.

[8] A. Howard et al., "Searching for MobileNetV3", 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 1314-1324, Nov. 2019.

[9] Y. Cao, J. Xu, S. Lin, F. Wei and H. Hu, "GCNet: Non-Local Networks Meet Squeeze-Excitation Networks and Beyond", 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), pp. 1971-1980, 2019.

[10] Y.F. Yang, M. Sun, "A Novel Deep Learning Architecture for Global Defect Classification: Self-Proliferating Neural Network (SPNet)", 2020 31th Annual SEMI Advanced Semiconductor Manufacturing Conference (ASMC), May 2021.

[11] Y. Sun et al., "Circle Loss: A Unified Perspective of Pair Similarity Optimization", 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6397-6406, Jun. 2020.

[12] I. Loshchilov, F. Hutter, "SGDR: Stochastic Gradient Descent with Warm Restarts", International Conference on Learning Representations (ICLR), May 2017.

[13] S. Ioffe, C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift", International Conference on Machine Learning (ICML), 2015.

[14] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov and L. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks", 2018 IEEE/CVF Computer Vision and Pattern Recognition Conference (CVPR), pp. 4510-4520, Jun. 2018.