繁體中文場景文字辨識競賽

初階：場景文字檢測

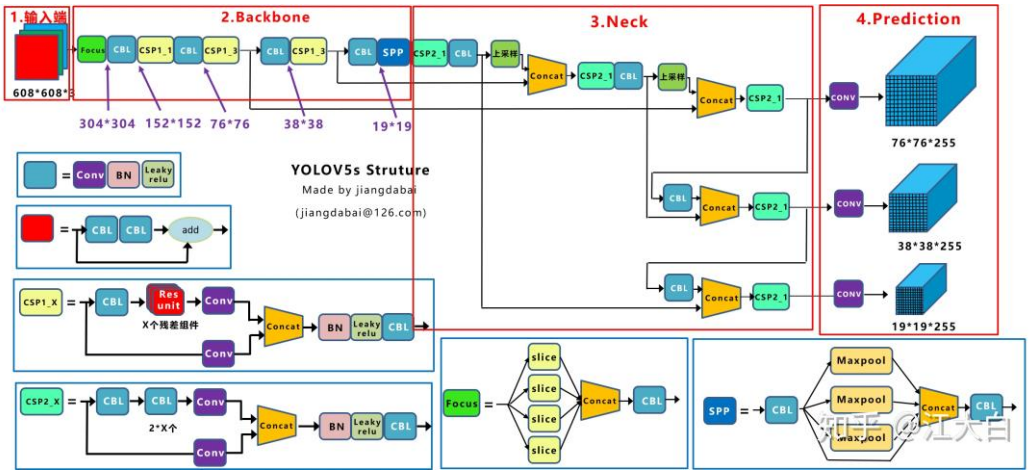
報告說明文件

壹、 環境

- 作業系統：Ubuntu 20.04.2 LTS
- 作業系統核心：GNU/Linux 5.4.0-74-generic x86_64
- 語言：Python 3.8
- 套件：(詳見 requirement.txt)
  - matplotlib
  - numpy
  - opencv-python
  - Pillow
  - PyYAML
  - scipy
  - torch
  - torchvision
  - tqdm
  - tensorboard
  - seaborn
  - pandas
  - scikit-learn
  - thop
  - pycocotools
- 預訓練模型：Yolov5x6，來源：
  https://github.com/ultralytics/yolov5/releases/download/v5.0/yolov5x6.pt
- 額外資料集：無

貳、　演算方法與模型架構

此次團隊使用的模型為 Yolov5x，以下是 Yolov5 的模型架構圖：



圖一、Yolov5 模型架構圖[1]

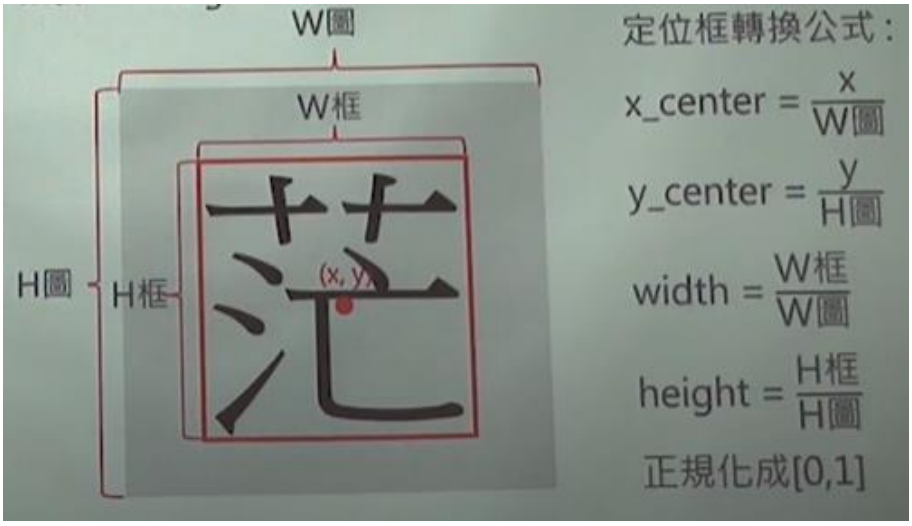
Yolov5 模型如同前幾代的 Yolo 模型，如 Yolov3、Yolov4 使用 one-stage 結構，分為輸入端、Backbone、Neck 和 Prediction 四個部份。在輸入端採用了和 Yolov4 一樣的 Mosaic 數據增強的方式，而且加入了自動錨框計算、自動圖片縮放。在 Backbone 階段中，增加 Focus 結構達到切片操作，也加入 CSP 結構達到減少了計算量的同時可以保證準確率。在 Neck 階段中，加入 FPN 及 PAN 結構，且採用借鑒 CSPnet 設計的 CSP2 結構，加強網路特徵融合的能力。在 Prediction 階段中採用 GIOU_Loss 做 Bounding box 的損失函数。而在 Yolov5 網路中團隊採用了 Yolov5x 網路，也是其中最深的網路，以增加網路特徵提取和特徵融合的能力。其中模型參數如下: 773 layers, 141119664 parameters, 141119664 gradients, 221.6 GFLOPS。

```
                from  n    params  module                                  arguments
  0                -1  1      8800  models.common.Focus                     [3, 80, 3]
  1                -1  1    115520  models.common.Conv                      [80, 160, 3, 2]
  2                -1  1    309120  models.common.C3                        [160, 160, 4]
  3                -1  1    461440  models.common.Conv                      [160, 320, 3, 2]
  4                -1  1   3285760  models.common.C3                        [320, 320, 12]
  5                -1  1   1844480  models.common.Conv                      [320, 640, 3, 2]
  6                -1  1  13125120  models.common.C3                        [640, 640, 12]
  7                -1  1   5531520  models.common.Conv                      [640, 960, 3, 2]
  8                -1  1  11070720  models.common.C3                        [960, 960, 4]
  9                -1  1  11061760  models.common.Conv                      [960, 1280, 3, 2]
 10                -1  1   4099840  models.common.SPP                       [1280, 1280, [3, 5, 7]]
 11                -1  1  19676160  models.common.C3                        [1280, 1280, 4, False]
 12                -1  1   1230720  models.common.Conv                      [1280, 960, 1, 1]
 13                -1  1         0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
 14          [-1, 8]  1         0  models.common.Concat                    [1]
 15                -1  1  11992320  models.common.C3                        [1920, 960, 4, False]
 16                -1  1    615680  models.common.Conv                      [960, 640, 1, 1]
 17                -1  1         0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
 18          [-1, 6]  1         0  models.common.Concat                    [1]
 19                -1  1   5332480  models.common.C3                        [1280, 640, 4, False]
 20                -1  1    205440  models.common.Conv                      [640, 320, 1, 1]
 21                -1  1         0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
 22          [-1, 4]  1         0  models.common.Concat                    [1]
 23                -1  1   1335040  models.common.C3                        [640, 320, 4, False]
 24                -1  1    922240  models.common.Conv                      [320, 320, 3, 2]
 25         [-1, 20]  1         0  models.common.Concat                    [1]
 26                -1  1   4922880  models.common.C3                        [640, 640, 4, False]
 27                -1  1   3687680  models.common.Conv                      [640, 640, 3, 2]
 28         [-1, 16]  1         0  models.common.Concat                    [1]
 29                -1  1  11377920  models.common.C3                        [1280, 960, 4, False]
 30                -1  1   8296320  models.common.Conv                      [960, 960, 3, 2]
 31         [-1, 12]  1         0  models.common.Concat                    [1]
 32                -1  1  20495360  models.common.C3                        [1920, 1280, 4, False]
 33  [23, 26, 29, 32]  1    115344  models.yolo.Detect                      [7, [[19, 27, 44, 40, 38, 94], [96, 68, 86, 152, 180, 137],
[140, 301, 303, 264, 238, 542], [436, 615, 739, 380, 925, 792]], [320, 640, 960, 1280]]
Model Summary: 773 layers, 141119664 parameters, 141119664 gradients, 221.6 GFLOPS
```

圖二、Yolov5x6 訓練模型架構

參、 資料處理

團隊使用 80%訓練資料搭配 20%驗證資料來切分 Public Training Dataset，並將其各自放置於 train/資料夾以及 val/資料夾，接著藉由主辦單位給定的 label ground truth，將其從 json 轉換為 YOLO txt 檔，且 ground truth 格式為 <class> <x_center> <y_center> <width> <height>，並依據下圖規則進行計算：



圖三、ground truth 格式轉換公式

到了最後的步驟，團隊將訓練及驗證資料的照片相對位置寫於 train.txt 和 val.txt，另外加入 setting.yaml 檔進行基本的設定，如:標籤分類種類數、每個 label 數字代表意義的設定等，並將訓練資料集和驗證資料集放置的位置指向上面所述的兩個文字檔。而因為在本次的訓練中並未加入其他的資料集，因此到這裡資料的處理已然完成。下圖呈現檔案位置放置方式:

```
+---annotations
|           setting.yaml
|           train.txt
|           val.txt
|
+---images
|     +---train
|     \---val
\---labels
       +---train
\---val
```

圖四、檔案位置放置方式

肆、 訓練方式

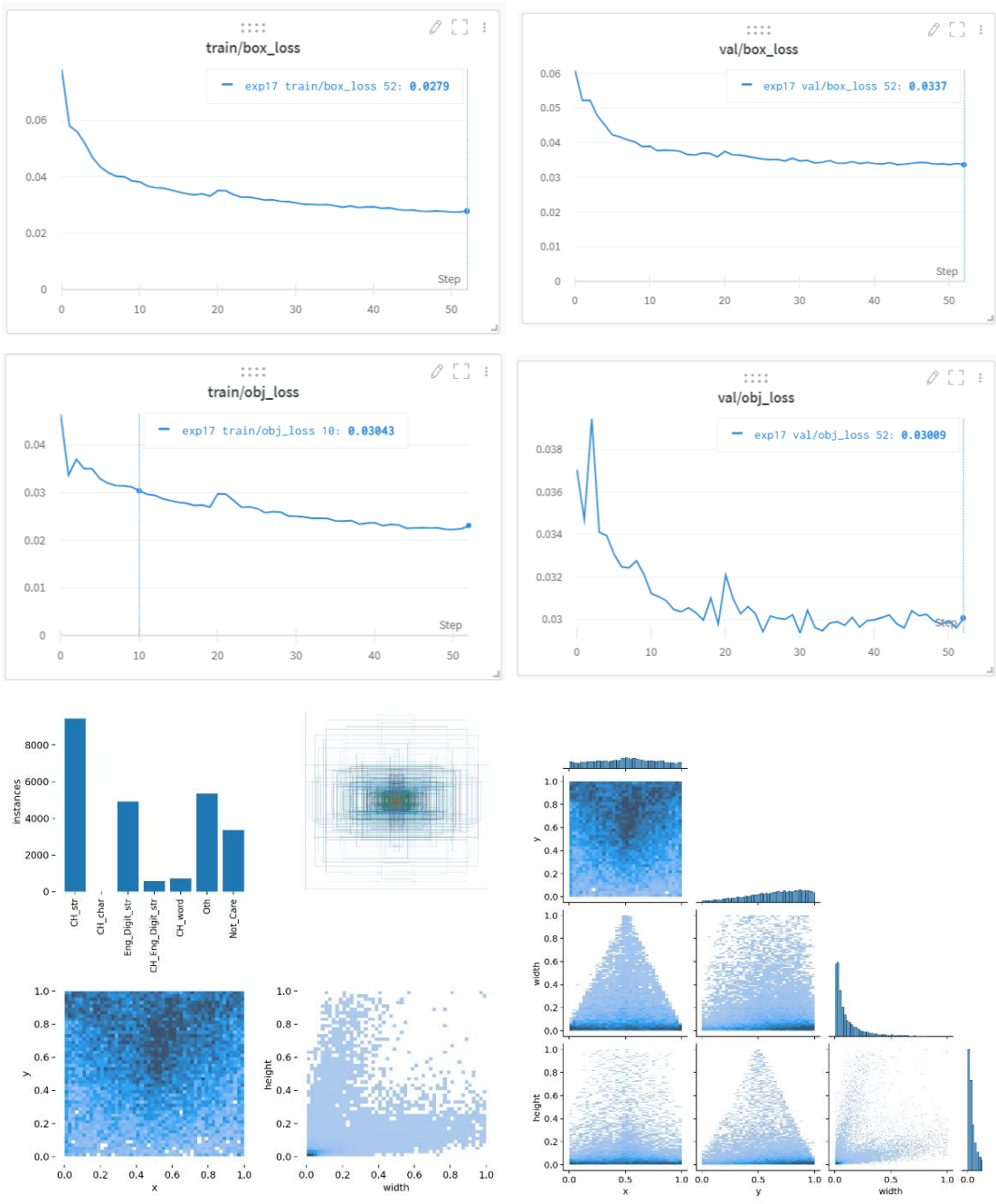
團隊採用 Yolov5x6 作為預訓練的模型，在 Image size 的設定上採用 1365，其為訓練圖片中的最長邊，而根據程式運作狀態可發現此設定必須為 64 之倍數，因此系統會自動將此變數設為 1408，並且加入「*--rect*」參數，以利於訓練矩形圖片，訓練的 epochs 團隊採用 300 epochs(根據 Yolov5「官方文檔所述，建議從 300 個 epoch 開始訓練，如果這造成過度擬合，那麼就可以提早結束。如果 300 個 epochs 後沒有發生過擬合，則可訓練更長時間，即 600、1200 等)，Batch Size 部分因為硬體設備的不足只能採用「*--batch-size 4*」，而官方建議將其至少設為 16 以上避免 batchnorm statistics 問題的發生。在 Hyperparameters 的部分則可參考我的 github 文檔(https://github.com/angelowen/2021_AICUP_ObjDectect/blob/master/yolov5/data/hyp.scratch.yaml)，其中基本設置包括了 warmup、learning rate 等等。

- learning rate 採用 OneCycle learning rate 方法將 initial learning rate 設為 0.01，並在每個 epochs 後乘上 0.2 以動態調整
- optimizer: SGD
- momentum: 0.937
- weight_decay: 0.0005
- warmup_epochs: 3.0
- warmup_momentum: 0.8
- IoU training threshold: 0.20

在 Data Augmentation 則採用 mosaic、Hsv-Hue、rotation 數據增強等方法:

- hsv_h: 0.015 # image HSV-Hue augmentation (fraction)
- hsv_s: 0.7 # image HSV-Saturation augmentation (fraction)
- hsv_v: 0.4 # image HSV-Value augmentation (fraction)
- degrees: 0.0 # image rotation (+/- deg)
- translate: 0.1 # image translation (+/- fraction)
- scale: 0.5 # image scale (+/- gain)
- shear: 0.0 # image shear (+/- deg)
- perspective: 0.0 # image perspective (+/- fraction), range 0-0.001
- flipud: 0.0 # image flip up-down (probability)
- fliplr: 0.5 # image flip left-right (probability)
- mosaic: 1.0 # image mosaic (probability)
- mixup: 0.0 # image mixup (probability)

另外 label smoothing 也是訓練技巧之一。為了緩解由 label 不夠 soft 而容易導致過擬合的問題，團隊採用此方法來讓模型對預測 less confident，訓練過程中亦採用 Weights & Biases 方法紀錄訓練的過程，一邊觀察訓練過程中是否過擬合並定時查看各項數值的表現，部分記錄請見下圖：



圖五、訓練紀錄圖表

伍、 分析與結論

對於此次使用的模型，經過多次偵測後查看所抓取的物件可以發現一些共同特性，抓取的物件含有許多過度微小的細節，就連人眼都難以辨認，包括部分招牌倒影以及冷氣上的標籤等，因此應統計分析哪些是容易抓錯的物件進行訓練資料的改進。本團隊在此次的比賽過程中，為了解決此項問題採用的方法是為每個類別的輸出設定一個特定的閾值，若低於此閾值則不將其寫入答案中，並且排除抓取物件中面積較小的 bounding box，將其視為模型的誤差值。此外也判斷當高度或寬度低於 22 pixels 時，因抓取到的物件常為錯誤圖像因此須將其排除在外。另外，在我們的模型中可發現，Not care 的類別經常會判斷錯誤，而在比賽中此類別並不計分，因此若偵測到此類別之物件也將予以排除。再者，若物件分類為英文字串，但其 bounding box 呈現直長型(高度遠大於寬度)，則必為偵測錯誤。依據上面幾項情況將錯誤狀況排除可大幅提升準確率。在偵測過程中也可發現特定物體會抓不到目標，如：字體左右交錯的招牌、亮度過高或傾斜的字體，甚至在圖片中明顯可見但機器卻不將其視為字串的圖像。再者，基於硬體設備的缺乏團隊只能採用 batch size 4，未來若能將其增大，準確率必定大大提升。

此外團隊期望使用 Hyperparameter Evolution 找出最佳參數，因為找出一組最佳的超參數一直都是一個挑戰，須在高維的搜索空間找尋維度之間未知的相關性，而在每個點上評估 fitness 的代價很高，也因此現階段我們無法達成，未來若有更充足的運算資源，此方法值得期待。最後，提升訓練資料量以提升準確率是最基礎的方法，一方面可以避免模型 overfitting，另一方面也增進了模型的穩定性，使其能夠更輕易應用於各場景的圖像。

在 inference 的過程中團隊採用 Model Ensembling 的技術，而這也是此次能達成高準確率的重要方法之一，藉由利用多個 model 合作的方式篩選出應該輸出的物件，達到多個學習加總的效果。另外也採用 Test Time Augmentation 方法，它將圖像大小增加約 30% 並左右翻轉以 3 種不同的解析度進行處理以改善結果，雖然需要耗時正常時間的 2~3 倍，但能抓出許多先前漏掉的文字。

我們的模型整體上來說可以抓取到大部分的物件，但會有抓取到非字體的物件以及重複抓取的問題導致整體的準確率不佳，以下點出幾項模型的失敗範例。

問題一、水平及垂直的 bounding box

圖六、 Img_123.jpg、Img_257.jpg、Img_424.jpg 失敗範例

在圖六中明顯可見,藍色矩形所抓取的範圍因為 yolo 模型只能創造水平及垂直的 bounding box 而導致有所偏差。此問題主要是因為文本通常有方向性，但 Yolov5 模型為 anchor-based，而 anchor-based 的檢測通常是水平和垂直方向的 bounding box，所以如果要改善因文本有角度的狀況，可以用 EAST(Efficient and Accuracy Scene Text detection pipeline) 模型改善，EAST 模型支持旋轉 bounding box、任意四邊形兩種文本區域標注形式，對於旋轉 bounding box 標注，模型執行時會對特徵圖中每一像素預測其到 bounding box 四邊的距離以及 bounding box 的方向[2]。

問題二、弧形字體無法完整框出

圖七、對於弧形字體無法完整框出其位置的範例圖

在一些圓形或橢圓形的招牌中，會有弧形字體的設計。對於弧形字體無法完整框出其位置，可以參考 *Real-time Scene Text Detection with Differentiable Binarization* [3]，該篇論文提出了一種即時任意形狀的場景文本檢測器，名為 Differentiable Binarization (DB)的模型，可以在網絡中執行二值化(binarization)。

問題三、抓取錯誤

圖八、抓取錯誤

如圖八中所示，將冷氣機識別為中文字串(左)，以及許多過小字體和倒影都被框列(右)。該問題可以透過訓練資料篩選解決，把在訓練資料中會造成辨識錯誤的圖片篩選出來，經統計分析且重新 label 後，再進行訓練。

問題四、縱行橫列重複抓取

圖九、縱行橫列重複抓取

如圖所示，有時候會發生縱行、橫列重複抓取的狀況，解決方法只需在辨識後，加入判斷 bounding box 重疊情況的機制即可解決。

問題五、長型招牌無法正確抓取文字

圖十、此類長形招牌經常無法正確抓取文字

此問題只能透過增加訓練資料量解決，透過學習更多類型的招牌，解決抓取失敗的問題。

陸、　程式碼

本次團隊所使用的程式碼已上傳 Github 網站，請至以下連結取得我們的程式碼：https://github.com/angelowen/2021_AICUP_ObjDectect，或是在終端機環境執行

git clone https://github.com/angelowen/2021_AICUP_ObjDectect.git。

柒、 使用的外部資源與參考文獻

[1] 江大白 (2020).*深入浅出 Yolo 系列之 Yolov5 核心基础知识完整讲解*

[2] 拨浪鼓儿 (2018).*文本检测模型综述*

[3] Liao, M., Wan, Z., Yao, C., Chen, K., & Bai, X. (2020, April). Real-time scene text detection with differentiable binarization. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 34, No. 07, pp. 11474-11481).

[4] ultralytics. *yolov5*. Retrieved from https://github.com/ultralytics/yolov5

[5] Micikevicius, P. (2011). Multi-GPU programming. GPU Computing Webinars, NVIDIA.

[6] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S. (2020, August). End-to-end object detection with transformers. In European Conference on Computer Vision (pp. 213-229). Springer, Cham.

[7] Moshkov, N., Mathe, B., Kertesz-Farkas, A., Hollandi, R., & Horvath, P. (2020). Test-time augmentation for deep learning-based cell segmentation on microscopy images. Scientific reports, 10(1), 1-7.

[8] Ganaie, M. A., & Hu, M. (2021). Ensemble deep learning: A review. arXiv preprint arXiv:2104.02395.

[9] YOLOv5 Documentation. Retrieved from https://docs.ultralytics.com/

[10] Bodla, N., Singh, B., Chellappa, R., & Davis, L. S. (2017). Soft-NMS--improving object detection with one line of code. In Proceedings of the IEEE international conference on computer vision (pp. 5561-5569).

[11] Hu, H., Gu, J., Zhang, Z., Dai, J., & Wei, Y. (2018). Relation networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 3588-3597)..

[12] Lin, T. Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., ... & Zitnick, C. L. (2014, September). Microsoft coco: Common objects in context. In European conference on computer vision (pp. 740-755). Springer, Cham.

[13] Zhang, C., Huang, Z., Guo, H., Qin, L., Ang Jr, M. H., & Rus, D. (2021). SMART-Rain: A Degradation Evaluation Dataset for Autonomous Driving in Rain.

[14] Lian, J., Yin, Y., Li, L., Wang, Z., & Zhou, Y. (2021). Small object detection in traffic scenes based on attention feature fusion. Sensors, 21(9), 3031.

[15] Hogan, M., Rondao, D., Aouf, N., & Dubois-Matra, O. (2021). Using Convolutional Neural Networks for Relative Pose Estimation of a Non-Cooperative Spacecraft with Thermal Infrared Imagery. arXiv preprint arXiv:2105.13789.

[16] Vishwakarma, R., & Vennelakanti, R. (2020, December). CNN Model & Tuning for Global Road Damage Detection. In 2020 IEEE International Conference on Big Data (Big Data) (pp. 5609-5615). IEEE.

[17] Yaw Jou, S., & Yen Su, C. (2021). A Novel lightweight Convolutional Neural Network, ExquisiteNetV2. arXiv e-prints, arXiv-2105.

[18] Zhou, P., Kortoci, P., Yau, Y. P., Braud, T., Wang, X., Finley, B., ... & Hui, P. (2021). Augmented informative cooperative perception. arXiv preprint arXiv:2101.05508.

[19] Yap, M. H., Hachiuma, R., Alavi, A., Brungel, R., Goyal, M., Zhu, H., ... & Frank, E. (2020). Deep Learning in Diabetic Foot Ulcers Detection: A Comprehensive Evaluation. arXiv preprint arXiv:2010.03341.

[20] Zheng, Z., Wang, P., Ren, D., Liu, W., Ye, R., Hu, Q., & Zuo, W. (2020). Enhancing geometric factors in model learning and inference for object detection and instance segmentation. arXiv preprint arXiv:2005.03572.

捌、 聯絡資料

- 隊伍

隊伍名稱	Private leaderboard 成績	Private leaderboard 名次
CCUML_木木丘山大岩壁	**0.655208**	**6**

- 隊員

姓名(中英皆需填寫)	學校系所	電話	E-mail
林岳 LIN, YUEH	中正大學 資訊工程學系	0932877274	jeff4209@gmail.com
温彥博 Yen Po Wen	成功大學 資訊工程研究所	0979736145	angeloewn@csie.io
陳威樺 Wei Hua Chen	中正大學 資訊工程學系	0966702995	tim.win.win@gmail.com
謝侑融 Hsieh Yu-Jung	中正大學 資訊工程學系	0930989102	eric1999880814@gmail.com
洪晟瑞 Cheng-Jui Hung	中正大學 資訊工程學系	0908070237	eritup45@gmail.com

- 指導教授

教授姓名	課程	課號	學校系所	E-mail
江振國	機器學習	4105931	中正大學 資訊工程學系	ckchiang@ccu.edu.tw