

# RSNA Pneumonia Detection Challenge

309553018 林孟學 310553027 姚淨云 309513080 楊亞澍

Project repository: <https://github.com/love124356/RSNA-Pneumonia-Detection-Challenge>

---

## Introduction

In this competition, we had to build an algorithm to detect the visual signal of pneumonia in medical images. That is, we need to detect lung opacities on the X-ray images. Pneumonia consistently ranks high on the list of the top 10 causes of death in the United States and third leading causes of death in Taiwan after 2016. Although it is a very common disease, accurate diagnosis of pneumonia is a daunting task as it requires a review of a chest radiograph (CXR) by a trained specialist and confirmation by clinical history, vital signs, and laboratory tests. Moreover, there are many other complex conditions on CXR that can lead to misinterpretation of the results by experts. Therefore, it is important to find out how to assist the expert to determine more accurately whether the image has pneumonia by the algorithm.

We have tried many modern object detection neural networks such as Faster R-CNN, Mask R-CNN, YOLOv5, and RetinaNet, and fine-tuned to this problem. To deal with the small amount and large variety of datasets, we also do many tries about data preprocessing and data augmentation.

## Related work

### Two-Stage methods

- Faster R-CNN[1]

The Faster R-CNN uses a Region Proposal Network (RPN), which takes the image feature map as input and generates a set of object proposals includes rectangular regions and objectness scores. Rather than Selective Search, which is used in Fast R-CNN to generate region proposals, RPN shares partial convolutional computation with the CNN of feature extraction and has a massive speedup compared to Fast R-CNN. After this, Faster R-CNN uses RoI pooling to compile the features of each proposal for object detection.

- Mask R-CNN[2]

Mask R-CNN added Mask Prediction Branch on the Faster R-CNN which is used for segmentation and proposes RoI Align, which improves RoI Pooling by fixing pixel misalignment problems which has a large effect on object segmentation.

In the two-stage models, the RPN of the first stage can filter out a large part of negative samples, and the final detection module of the second stage only needs to process a small number of candidate frames. Its accuracy is better than the one-stage methods. However, its detection speed is lower.

## One-Stage methods

- YOLOv5[3]

YOLOv5 is a family of compound-scaled object detection models trained on the COCO dataset and includes simple functionality for Test Time Augmentation (TTA), model ensembling, hyperparameter evolution, and so on. It also provides mosaic data enhancement, automatic anchor size calculation to train the model.

- RetinaNet[4]

RetinaNet uses many excellent architectures such as residual networks, Feature Pyramid Network (FPN), etc. They are good at extracting features, or multi-scale problems. It also proposed focal loss which is useful in one-stage detection strategies by preventing a large number of easy negatives from overwhelming the detector during training.

These one-stage methods have been tuned for speed to face many real-time tasks, but their accuracy trails that of two-stage methods.

## Classification

- Swim Transformer[5]

Swim Transformer is a new vision Transformer, which cleverly solves the problems that previous transformers often have a large computation complexity because of the high resolution of pictures and then serves as a general-purpose backbone for computer vision region.

## Other methods

- K-Fold Cross Validation

This method split the training dataset into K equal-sized subsamples, taking one subsample as the validation set and others as training set each time. Through this schedule, we can get k weights finally. Combining these models and predictions gives us the best result.

- Ensemble Learning

Ensemble learning is a good machine learning technique that combines many good models (prediction results) and gets a better result than training only one large and well-performed model.

# Proposed approach

## Classification

We used swin-transformer for classification. The swin-large is applied to our classification model and its basic architecture is shown in Fig. 1. Models were trained on either 2 classes (opacity vs. not) or 3 classes (opacity vs. not normal/no opacity vs. normal).

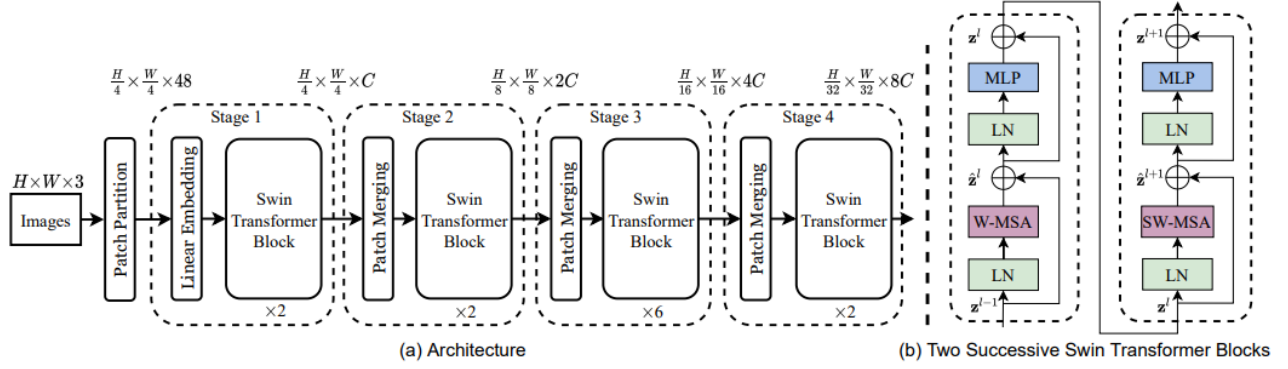


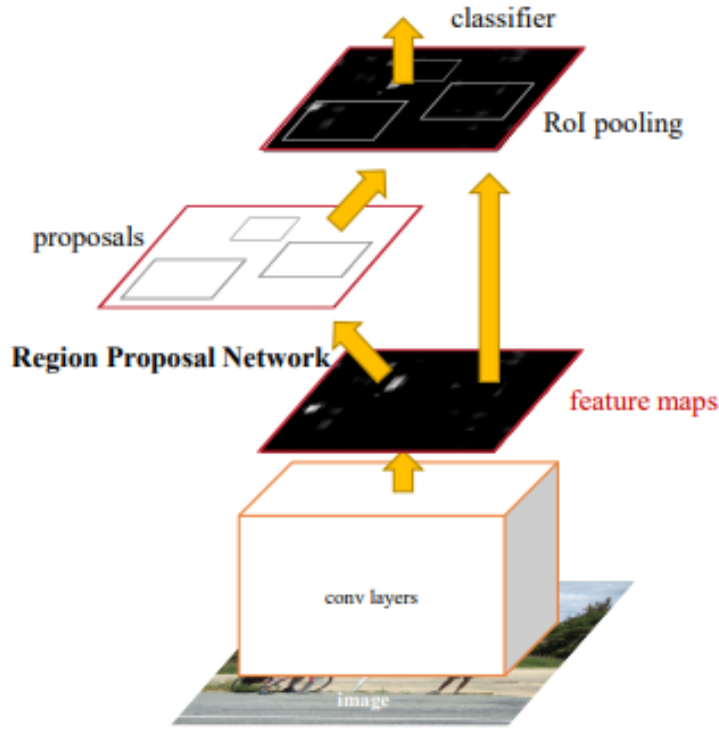
Fig. 1. (a) The architecture of a Swin Transformer (Swin-T); (b) two successive Swin Transformer Blocks . W-MSA and SW-MSA are multi-head self attention modules with regular and shifted windowing configurations, respectively.

We used the swin-transformer pre-trained networks on the ImageNet-22K. Then we fine-tuned those weights on the pneumonia dataset. We were getting about 0.90 - 0.92 accuracy across our folds.

## Detection

Our object detection system, Faster R-CNN with Feature Pyramid Network(FPN), is composed of two modules. The first module is a deep fully convolutional network that proposes regions, and the second module is the Fast R-CNN detector that uses the proposed regions. The entire system is a single, unified network for object detection (Fig. 2). Using the ‘attention’ mechanisms, the RPN module tells the Fast R-CNN module where to look. The illustration of FPN is shown in Fig. 3.

For further comparison, we also applied YOLOv5, RetinaNet and Mask R-CNN for reference results. The architecture of YOLOv5 and RetinaNet is shown in Fig. 4-5, respectively.



$$L(\{p_i\}, \{t_i\}) = \frac{1}{N_{cls}} \sum_i L_{cls}(p_i, p_i^*) + \lambda \frac{1}{N_{reg}} \sum_i p_i^* L_{reg}(t_i, t_i^*).$$

- cls layer predicts an anchor as an object (proposal) or non-object
- reg layer regresses the bounding box location

Fig. 2. Faster R-CNN is a single, unified network for object detection. The RPN module serves as the ‘attention’ of this unified network.

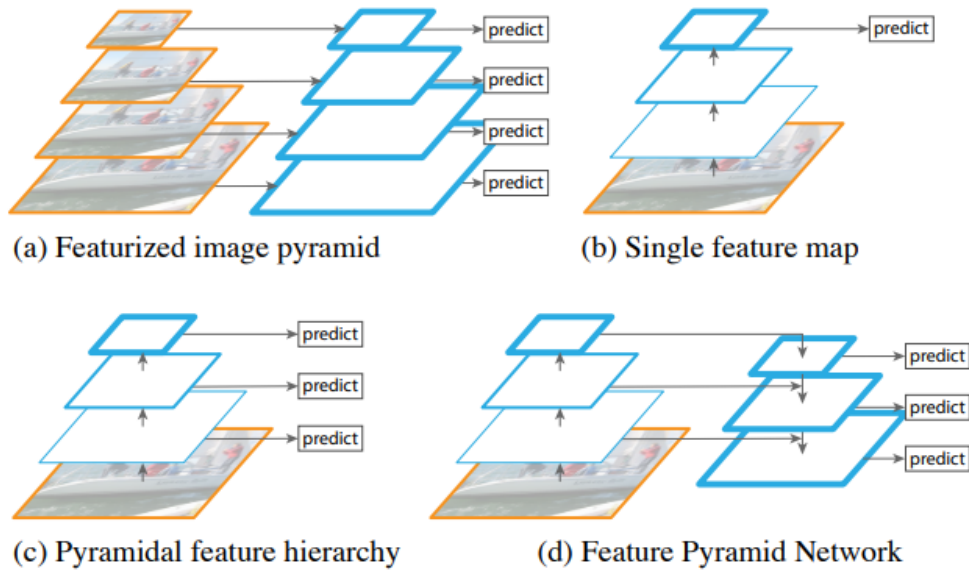
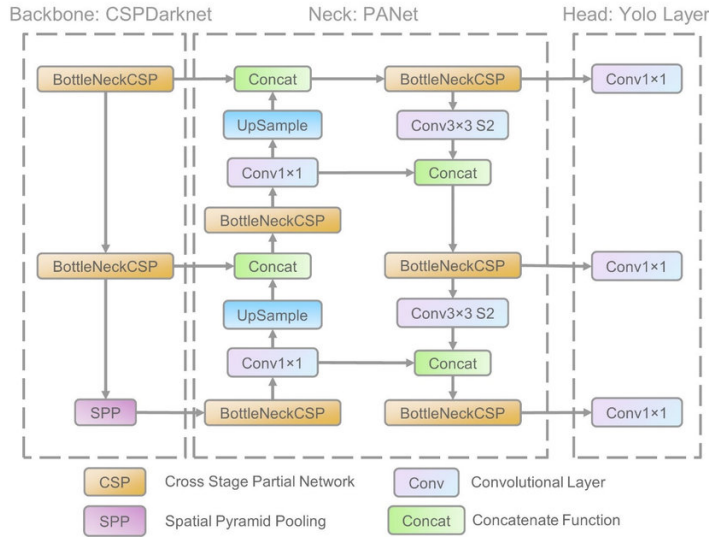


Fig. 3. (a) Using an image pyramid to build a feature pyramid. Features are computed on each of the image scales independently, which is slow. (b) Recent detection systems have opted to use only single scale features for faster detection. (c) An alternative is to reuse the pyramidal feature hierarchy computed by a ConvNet as if it were a featurized image pyramid. (d) Our proposed Feature Pyramid Network (FPN) is fast like (b) and (c), but more accurate. In this figure, feature maps are indicated by blue outlines and thicker outlines denote semantically stronger features.

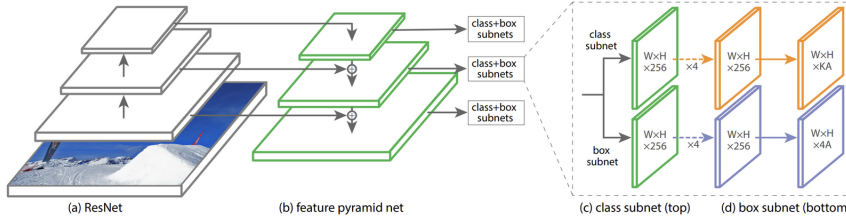


- The BECLogits loss function calculates the loss of the objectness score
- The class probability score uses the cross-entropy loss function (BCEclsloss), and the bounding box uses the GIoU Loss

$$GIoU = IoU - \frac{|A_c - U|}{|A_c|}$$

Fig. 4. The architecture of YOLOv5.

1. YOLOv5 Backbone: It employs CSPDarknet as the backbone for feature extraction from images consisting of cross-stage partial networks.
2. YOLOv5 Neck: It uses PANet to generate a feature pyramids network to perform aggregation on the features and pass it to Head for prediction.
3. YOLOv5 Head: Layers that generate predictions from the anchor boxes for object detection.



$$FL(p_t) = -\alpha_t(1 - p_t)^\gamma \log(p_t).$$

Fig. 5. The architecture of RetinaNet. (a) Bottom-up Pathway - The backbone network (e.g. ResNet) calculates the feature maps at different scales, irrespective of the input image size or the backbone. (b) Top-down pathway and Lateral connections - The top-down pathway upsamples the spatially coarser feature maps from higher pyramid levels, and the lateral connections merge the top-down layers and the bottom-up layers with the same spatial size. (c) Classification subnetwork - It predicts the probability of an object being present at each spatial location for each anchor box and object class. (d) Regression subnetwork - It regresses the offset for the bounding boxes from the anchor boxes for each ground-truth object.

RetinaNet Focal loss uses  $\gamma$  (focusing parameter) and  $\alpha$ -balanced variant to solve the large class imbalance problems which are encountered during training of dense detectors overwhelms the cross-entropy loss

# Experiment results

## Dataset

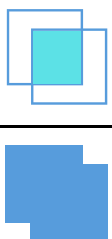
The dataset of the RSNA Pneumonia Detection Challenge is from the NIH CXR14 dataset using their original labels which were derived from radiology reports. The dataset has 26684 training images, of which only 6012 images contain 9555 positive instances for training and 3000 test images for testing. It contains the following classes:

- Not Normal/No Lung Opacity - No lung opacity refers to no opacity suspicious for pneumonia.
- Lung Opacity - A finding on chest radiograph that a patient with cough and fever has a high likelihood of being pneumonia. This class will be labeled with bounding boxes.
- Normal

## Evaluation metrics

We evaluated by the following metrics:

- Intersection over Union (IoU) - The measure is the degree of overlap between the two areas, which is the ratio of the area of the overlap to the area of the union of the two areas. By using IoU with a threshold (e.g. 0.5) to determine if the object is correctly detected, and we can determine whether a detection result (Positive) is correct (True) or incorrect (False), e.g., if  $\text{IoU} > 0.5$ , it is considered True Positive, otherwise it is considered False Positive.

$$\text{IoU} = \frac{\text{Area of the overlap}}{\text{Area of the union}}$$


- Average Precision (AP) - The average precision over thresholds. We also calculate false-negative if it indicates a ground truth box had no associated predicted box.

$$\text{precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives} + \text{False Negative}}$$

- Mean Average Precision (mAP) - The mean of APs over classes.

## Comparison

In the following section\*, we compare models with different hyper-parameters and architectures. We also try to implement some techniques that are often used in Kaggle competitions such as model ensemble, K-fold cross-validation.

\* All the experiments apply data augmentation (YOLOv5 - Mosaic, Faster R-CNN - flip, resize, rotate, blur with probability, Mask R-CNN - affine transform, brightness, blur with probability).

As shown in Tab. 1, Faster R-CNN has better performance. We have only tuned the hyper-parameters for YOLOv5 and Faster R-CNN to get the final results. We also found that in general, Mask R-CNN is derived from Faster R-CNN and should perform better, but there is a significant difference in this task. This may be due to the fact that we have not tuned the hyper-parameters of Mask R-CNN. If we turned the hyper-parameters, the mAP will be similar.

Table 1. Comparison with related work.

	backbone	mAP
<i>One-stage methods</i>		
YOLOv5[3]	CSPDarkNet	13.948
RetinaNet[4]	ResNet-50	11.173
<i>Two-stage methods</i>		
Mask R-CNN[2]	ResNet-50	11.647
<b>Faster R-CNN[1]</b>	ResNet-50-FPN	<b>15.627</b>

As shown in Tab. 2, we modified model architecture, epochs, batch size, image size, confidence threshold, and training and validation data for YOLOv5. Also, we modified epochs, batch size, image size, learning rate, and confidence threshold for Faster R-CNN.

We can find that compared to Faster R-CNN, YOLOv5 needs to set a larger batch size and a lower confidence threshold to have better performance. Also, YOLOv5 with larger model architecture has almost the same performance compared with other smaller model architecture. We even found a more interesting thing: as soon as the training and validation data were modified, there was a sudden increase in mAP.

Table 2. Models with different hyper-parameters: only YOLOv5 and Faster R-CNN

Model	Hyper-parameters	mAP
<i>One-stage methods</i>		
1. YOLOv5l[3]	100 epochs, 16 batch size, 640 x 640 image size, 0.25 conf. threshold	9.357
2. YOLOv5l6[3]	300 epochs, 16 batch size, 608 x 608 image size, 0.3 conf. threshold	12.124
3. YOLOv5l6[3]	500 epochs, 32 batch size, 608 x 608 image size, 0.3 conf. threshold	11.808
4. YOLOv5l6[3]	300 epochs, 4 batch size, 1024 x 1024 image size, 0.25 conf. threshold	11.884
5. YOLOv5m6[3]	300 epochs, 32 batch size, 608 x 608 image size, 0.3 conf. threshold	11.891
6. YOLOv5s6[3]	300 epochs, 32 batch size, 608 x 608 image size, 0.3 conf. threshold	12.261

7. YOLOv5n6[3]	300 epochs, 32 batch size, 608 x 608 image size, 0.3 conf. threshold	11.831
8. YOLOv5l6[3]	300 epochs, 32 batch size, 608 x 608 image size, 0.3 conf. threshold	12.109
9. YOLOv5l6[3]	300 epochs, 32 batch size, 608 x 608 image size, 0.3 conf. threshold, 5-fold cross-validation (best mAP)	10.223
10. YOLOv5l6[3]	300 epochs, 32 batch size, 608 x 608 image size, 0.3 conf. threshold, other training and validation data	<b>13.129</b>
<i>Two-stage methods</i>		
11. Faster R-CNN[1]	50 epochs, 8 batch size, 300 x 300 image size, 1e-4 lr, 0.8 conf. threshold	12.845
12. Faster R-CNN[1]	50 epochs, 8 batch size, 300 x 300 image size, 1e-3 lr, 0.8 conf. threshold	13.114
13. Faster R-CNN[1]	50 epochs, 8 batch size, 512 x 512 image size, 1e-3 lr, 0.8 conf. threshold	12.633
14. Faster R-CNN[1]	50 epochs, 4 batch size, 300 x 300 image size, 1e-3 lr, 0.8 conf. threshold	13.807
15. Faster R-CNN[1]	50 epochs, 2 batch size, 300 x 300 image size, 1e-3 lr, 0.8 conf. threshold	13.627
16. Faster R-CNN[1]	100 epochs, 4 batch size, 300 x 300 image size, 1e-3 lr, 0.8 conf. threshold	14.054
17. Faster R-CNN[1]	100 epochs, 4 batch size, 256 x 256 image size, 1e-3 lr, 0.8 conf. threshold	13.388
18. Faster R-CNN[1]	100 epochs, 4 batch size, 256 x 256 image size, 1e-3 lr, 0.7 conf. threshold	12.772
19. Faster R-CNN[1]	100 epochs, 4 batch size, 256 x 256 image size, 1e-3 lr, 0.9 conf. threshold	13.112
20. Faster R-CNN[1]	100 epochs, 4 batch size, 300 x 300 image size, 1e-3 lr, 0.8 conf. threshold, other training and validation data, 5-fold cross-validation (best mAP)	<b>15.627</b>



As shown in Tab. 3 and Tab. 4, we tried to ensemble different models. We tried to do an ensemble method for different or the same models and found that Faster R-CNN performed better than the others. We think if we use different models (Mask R-CNN, Faster R-CNN...), we may get better performance than the current. However, we don't have too much time to tune the hyperparameters and train the models.

Table 3. Ensemble models

Ensemble models	mAP
combine 2, 10	12.089 (-1.104)
combine 2, 5, 6, 8, 10	11.286 (-1.843)
No. 9, 5-fold cross validation	8.517 (-1.706)
No. 20, 5-fold cross validation	<b>16.336</b> (+0.709)

Table 4. Ablation study: Faster R-CNN using 5-fold cross validation and ensemble method

	mAP
Fold 0	15.627
Fold 1	14.590
Fold 2	14.587
Fold 3	14.756
Fold 4	8.831
+ ensemble	<b>16.336</b> (+0.709)

As shown in Tab. 5, we tried to use a classifier to combine model predictions. We averaged the classification scores with the predicted confidence values of the bounding box to get a new score.

Obviously, this method is helpful for YOLOv5. Our reasoning is that this method will improve YOLOv5 because of its low confidence threshold. However, Faster R-CNN has an output confidence threshold as high as 0.8, so this method is not effective for it. We also tried to lower the output confidence threshold, but there was no improvement either.

Table 5. Use classifier to combine model predictions

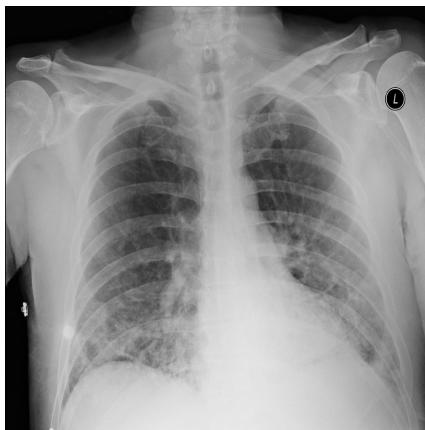
Number of Tab. 2	classifier threshold	mAP
10	0.2	13.984 (+0.855)
14	0.5	12.371 (-1.436)

## Conclusion

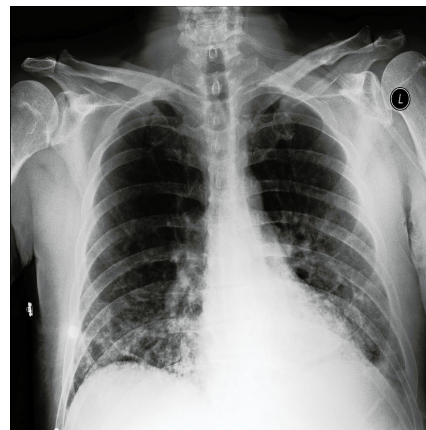
In this work, we use many techniques such as data augmentation, k-fold cross-validation, ensemble method, combine with classification, try many object detection models including one-stage and two-stage methods, and finally perform a result **16.336** mAP in Pneumonia Detection Challenge with Faster R-CNN[1] and ensemble method. And we find that a larger image size may not be that much useful in this task, because the pneumonia region is large enough for normal detectors.

## Interesting findings

- Histogram Equalization: A paper, *X-Ray Image based COVID-19 Detection using Pre-trained Deep Learning Models*[6], applies histogram equalization to every images as one of their preprocessing methods, in order to minimize the effect of sampling bias while collecting these X-ray images with unknown number and variety of X-Ray machines, exposure parameters and operator behaviour. However, we get a worse result with this preprocessing methods. This probably because of that the histogram equalization method will break the foggy places in original pictures which is exactly the Pneumonia region.



*original*



*after equalization*

- Ensemble: Ensemble is absolutely a helpful approach in machine learning, although combining a bunch of bad models still gives a bad performance.

## Model performance

Submission and Description	Private Score	Public Score	Use for Final Score
<a href="#">ensemble.csv</a> a few seconds ago by <a href="#">cy</a>	0.16336	0.08024	<input type="checkbox"/>

## Contribution

Tasks	Contributors (%)
Literature survey	309553018(30%), 310553027(40%), 309513080(30%)
Approach design	309553018(33%), 310553027(33%), 309513080(33%)
Approach implementation (experiment)	309553018(33%), 310553027(33%), 309513080(33%)
Report writing	309553018(25%), 310553027(50%), 309513080(25%)
Slide making and oral presentation	309553018(33%), 310553027(33%), 309513080(33%)

## Reference

- [1] S. Ren, K. He, R. Girshick and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137-1149, 1 June 2017, doi: 10.1109/TPAMI.2016.2577031.
- [2] K. He, G. Gkioxari, P. Dollár and R. Girshick, "Mask R-CNN," 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2980-2988, doi: 10.1109/ICCV.2017.322.
- [3] GitHub. YOLOv5. Available online: <https://github.com/ultralytics/yolov5> (accessed on 3 January 2022)
- [4] T. -Y. Lin, P. Goyal, R. Girshick, K. He and P. Dollár, "Focal Loss for Dense Object Detection," 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 2999-3007, doi: 10.1109/ICCV.2017.324.
- [5] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, Baining Guo; Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 10012-10022
- [6] Michael J Horry<sup>1, 3</sup>, Subrata Chakraborty<sup>1 \*</sup>, Manoranjan Paul<sup>2</sup>, Anwaar Ulhaq<sup>2</sup>, Biswajeet Pradhan<sup>1</sup>, Manash Saha<sup>4</sup>, Nagesh Shukla<sup>1</sup>, "X-Ray Image based COVID-19 Detection using Pre-trained Deep Learning Models"