Chest X-ray Abnormalities Detection

Chaoran Huang
University of California, Irvine
Irvine, United States of America
chaorah1@uci.edu

***Abstract*—Chest X-ray Abnormalities Diagnosis remains a challenging and critical health problem. The interpretation of radiology study is not a simple classification problem; the diagnostic report involves numerous parameters. Doctors sometimes also have trouble to correctly distinguish the chest abnormalities from x-ray due to the complexity or carelessness. Therefore, we designed a software to assist doctors diagnose chest abnormalities through neural networks: YOLO and Faster RCNN.**

***Index Terms*—YOLO, Faster R-CNN, x-ray, abnormalities, diagnosis**

I. Introduction

According to the article "Discrepancy and Error in Radiology: Concepts, Causes and Consequences", the author argues that "the level of error for clinically significant or major error in radiology is in the range 2-20% and varies depending on the radiological investigation". [1] During exploratory data analysis, our team discovered that different radiologists tend to arrive at different diagnostic results of a certain chest x-ray image. To ensure consistent and correct diagnosis, developing computer-aided detection and diagnosis systems are essential to radiological practitioners. In this paper, we aim to research this topic and develop a system based on neural network model with diagnosis suggestions intended to detect and locate the abnormality in chest x-ray images so that can help both doctors and patients avoid misdiagnosis on chest diseases.

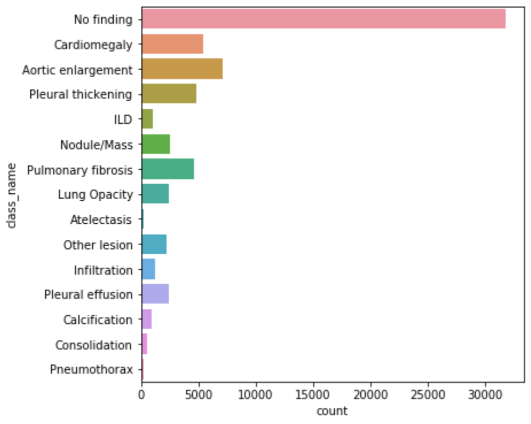
II. Background and Related Work

Machine Learning (ML) and Artificial Intelligence (AI) are widely used in many fields. Medical and radiology fields are also considered as a potential field to apply ML and AI technologies. Especially in primary care and suspicious illness, due to the contribution of large data and ML/AI tools, doctors will have more confidence in diagnose marginal cases which are rare and difficult to diagnose. We collect a high-quality dataset from Kaggle "VinBigData Chest X-ray Abnormalities". [2]

Table 1: Data Collection and Preliminary Analysis

	Data Statistics
# Of Images	15,000
Size of All DICOM Images	160 GiB
# Of Ground Truth Including No Finding	67,913
# Of Ground Truth Excluding No Finding	36,096

According to Kaggle's description, "The dataset used in this competition was created by assembling de-identified Chest X-ray studies provided by two hospitals in Vietnam: The Hospital 108 and the Hanoi Medical University Hospital". The data is in DICOM type, which is a standard file type in medical communication and management. All the DICOM files are annotated by multiple radiologists and labeled with 15 classes as findings, including 14 abnormality labels and the "No finding" label. The "training.csv" (shown in figure. training.csv file labels example) describes one object with image name, class label, and bounding box coordinates. Since one image has been annotated by multiple radiologists, several bounding boxes are possibly assigned to the same abnormalities.



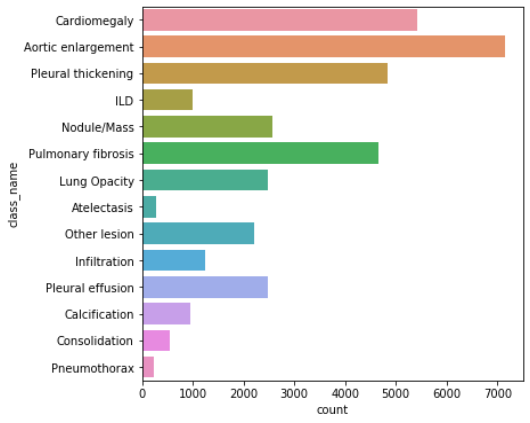


Figure 1. Dataset Overview

Data Processing and Augmentation

During our exploration of the dataset, we discover a large portion of images labeled "no finding". By removing them, the dataset size shrinks to 4394 images. Those "no finding" images contain little information and thus will not facilitate our training process. In addition, most mainstream neural networks do not accept DICOM files as input, so it is necessary to convert the DICOM file into a NumPy array. The pydicom class can complete this goal. To fit the data into the model and analyze all parts of data at the same weight, padding and resize operation are applied to images. The padding operation converts non-square images to square ones by introducing new pixels to edges. These images will later be resized to 640 by 640.

To reduce the overfitting and improve the robustness of our model, some data augmentations are inevitable. After some runs, flip with a probability of 0.5, a scale by an additional 0.5 image size, and sheer with 1 degree are adopted to the dataset. Translation in our augmentation is also embraced. From the heatmap generated by ground truth in the previous section, it is clearly to discovered that most abnormalities are located at relatively fixed positions. Therefore, I use a small value of 0.1 image size in our translation. This technique will improve the model's object detection ability on the dataset.

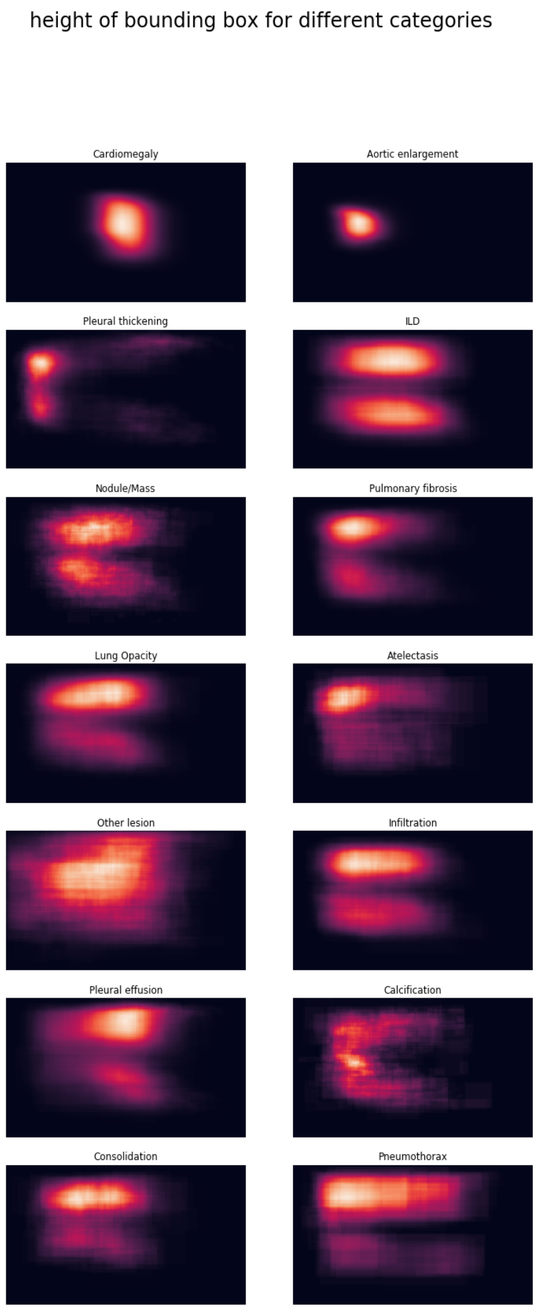


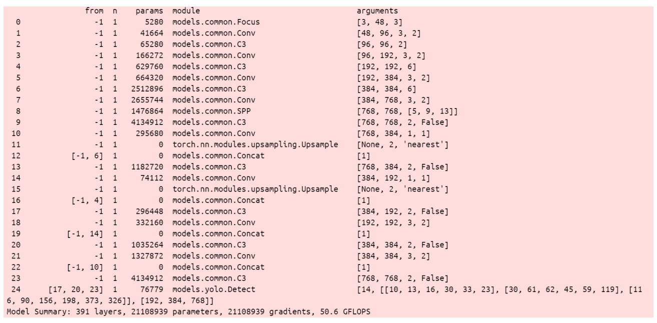
Figure 2. Heatmap of all 14 abnormalities

We have separated the train data into train and validate sets; the ratio is 0.8 to 0.2. The table below illustrates the label distribution in the training set and the validation set.

III. PROPOSE METHOD

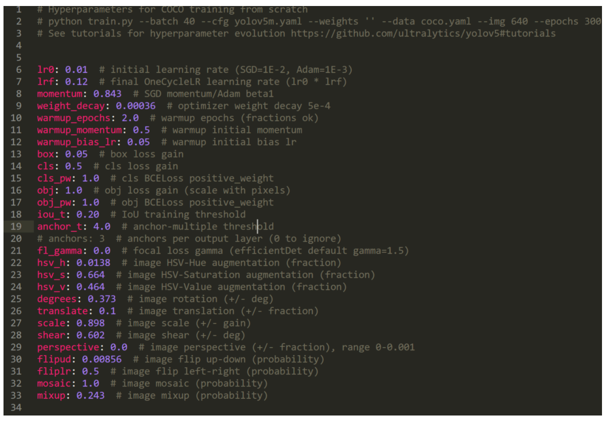
We proposed two models to approach the problem. The first one is through YOLOv5, a PyTorch-based one-stage detector.

The author did not publish the paper for YOLOv5. But from the source code, I find that the network architecture is similar to YOLOv4 [3]. They both use CSPDarknet53 as the backbone, PANET, and SPP as the neck and YOLOv3's head. Also, the sub-model YOLOv5 s/m/l/x share the same architecture. The only difference is that the model uses *depth_multiple* and *width_multiple* to control the depth of the model and the number of filters in the channel. In the following training, different models on YOLOv5 are tested to collect an inclusive result.



```
                 from  n    params  module                                  arguments
  0                -1  1      5280  models.common.Focus                     [3, 48, 3]
  1                -1  1     41664  models.common.Conv                      [48, 96, 3, 2]
  2                -1  1     65280  models.common.C3                        [96, 96, 2]
  3                -1  1    166272  models.common.Conv                      [96, 192, 3, 2]
  4                -1  1    629760  models.common.C3                        [192, 192, 6]
  5                -1  1    664320  models.common.Conv                      [192, 384, 3, 2]
  6                -1  1   2512896  models.common.C3                        [384, 384, 6]
  7                -1  1   2655744  models.common.Conv                      [384, 768, 3, 2]
  8                -1  1   1476864  models.common.SPP                       [768, 768, [5, 9, 13]]
  9                -1  1   4134912  models.common.C3                        [768, 768, 2, False]
 10                -1  1    295680  models.common.Conv                      [768, 384, 1, 1]
 11                -1  1         0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
 12           [-1, 6]  1         0  models.common.Concat                    [1]
 13                -1  1   1182720  models.common.C3                        [768, 384, 2, False]
 14                -1  1     74112  models.common.Conv                      [384, 192, 1, 1]
 15                -1  1         0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
 16           [-1, 4]  1         0  models.common.Concat                    [1]
 17                -1  1    296448  models.common.C3                        [384, 192, 2, False]
 18                -1  1    332160  models.common.Conv                      [192, 192, 3, 2]
 19          [-1, 14]  1         0  models.common.Concat                    [1]
 20                -1  1   1035264  models.common.C3                        [384, 384, 2, False]
 21                -1  1   1327872  models.common.Conv                      [384, 384, 3, 2]
 22          [-1, 10]  1         0  models.common.Concat                    [1]
 23                -1  1   4134912  models.common.C3                        [768, 768, 2, False]
 24      [17, 20, 23]  1     76779  models.yolo.Detect                      [14, [[10, 13, 16, 30, 33, 23], [30, 61, 62, 45, 59, 119], [11
6, 90, 156, 198, 373, 326]], [192, 384, 768]]
Model Summary: 391 layers, 21108939 parameters, 21108939 gradients, 50.6 GFLOPS
```

Figure 3. YOLOv5 Architecture



```
# Hyperparameters for COCO training from scratch
# python train.py --batch 40 --cfg yolov5m.yaml --weights '' --data coco.yaml --img 640 --epochs 300
# See tutorials for hyperparameter evolution https://github.com/ultralytics/yolov5#tutorials


lr0: 0.01  # initial learning rate (SGD=1E-2, Adam=1E-3)
lrf: 0.12  # final OneCycleLR learning rate (lr0 * lrf)
momentum: 0.843  # SGD momentum/Adam beta1
weight_decay: 0.00036  # optimizer weight decay 5e-4
warmup_epochs: 2.0  # warmup epochs (fractions ok)
warmup_momentum: 0.5  # warmup initial momentum
warmup_bias_lr: 0.05  # warmup initial bias lr
box: 0.05  # box loss gain
cls: 0.5  # cls loss gain
cls_pw: 1.0  # cls BCELoss positive_weight
obj: 1.0  # obj loss gain (scale with pixels)
obj_pw: 1.0  # obj BCELoss positive_weight
iou_t: 0.20  # IoU training threshold
anchor_t: 4.0  # anchor-multiple threshold
# anchors: 3  # anchors per output layer (0 to ignore)
fl_gamma: 0.0  # focal loss gamma (efficientDet default gamma=1.5)
hsv_h: 0.0138  # image HSV-Hue augmentation (fraction)
hsv_s: 0.664  # image HSV-Saturation augmentation (fraction)
hsv_v: 0.464  # image HSV-Value augmentation (fraction)
degrees: 0.373  # image rotation (+/- deg)
translate: 0.1  # image translation (+/- fraction)
scale: 0.898  # image scale (+/- gain)
shear: 0.602  # image shear (+/- deg)
perspective: 0.0  # image perspective (+/- fraction), range 0-0.001
flipud: 0.00856  # image flip up-down (probability)
fliplr: 0.5  # image flip left-right (probability)
mosaic: 1.0  # image mosaic (probability)
mixup: 0.243  # image mixup (probability)

```

Figure 4. YOLOv5 Hyperparameters

During the training process of YOLOv5, there are three decisions to make, including optimizers and hyperparameters. Firstly, the two built-in optimizers for YOLOv5 are SGD and ADAM. SGD is selected as our optimizer because of its performance in optimizing models of large datasets. Secondly, YOLOv5 provides users with various hyperparameters. One of them is the learning rate. We determine the initial learning rate for all models to be 0.01 because it is recommended by the YOLOv5 developers.

Our second model is "Faster R-CNN"

In comparison to other "RCNN" family models, Faster R-CNN outperforms other models in efficiency and performance. This model architecture consists of four building blocks. The first one is a region proposal algorithm that generates bounding boxes of objects in the image. The second one is a feature extraction algorithm that extracts features from these objects. The third one is a classification layer that predicts what classes the object belongs to. The last one is a regression layer which makes the coordinates of all the bounding boxes more precise.

Faster R-CNN is like a pipeline that uses an RPN (Regional Proposal Network) to decide the region and then a Fast R-CNN to detect what those regions are. RPN improves vastly on the time efficiency of the model. By introducing RPN, the region proposal part now only needs approximately 10ms to scan over one image and some layers can still be used in the later object detection stage. Thus, there is a huge improvement in Faster R-CNN.

Also, we applied the technique of placing a set of "Anchors" on the input image for each location on the output feature map from the backbone network. These anchors indicate possible objects in various sizes and aspect ratios at this location.

IV. EXPERIEMNTAL RESULT

A. Evalutaion

To evaluate our model, criteria including precision, recall, and mAP@0.5 are most important values. The mAP derives from the precision-recall graph and the IoU threshold is set to 0.5.

Equation 1 mAP@0.5

$$mAP@0.5 = \frac{\sum_i^{\#\_of\_classes} \int_0^1 P_i(R_i)dR_i}{\#\_of\_classes} \mid IoU_{threshold} = 0.$$

Equation 2 Precision

$$Precision = \frac{TP}{TP+FP}$$

Equation 3 Recall

$$Recall = \frac{TP}{TP+FN}$$

Table 2. Evaluation

	Relevant	*Non-Relevant*
Retrieved	True Positive	False Positive
Not Retrieved	False Negative	True Positive

In real life context, if radiologists are to deploy an object detection network like mine, they would expect the network to identify correct abnormalities from an X-ray image. This expectation coincides with the precision score which describes the ability of a neural network to output as many correct bounding boxes as possible. Therefore, our goal is to produce models that output high precision scores. However, the model still requires a satisfactory recall score to output a minimal amount of false-negative bounding boxes. Thus, splitting into groups of two and train the two selected models respectively is reasonable.

B. Results

YOLOv5: The developers of PyTorch-based YOLOv5 release four major variants of their model. From the chart below, as the size of the model increases, the performance increases. The next step of the training process is to apply augmentation to our images and choose the model with optimal performance. To select the model for the next stage of training, because YOLOv5x can output the highest PR and mAP scores, we select YOLOv5xl in the next stage of training.

Table 3. YOLOv5 results

Best mAP@0.5 From 100 Epochs				
	Precision	*Recall*	*mAP@.5*	*mAP@.5:.95*
YOLOv5s (Epoch 35)	.3917	.3418	.3173	.1353
YOLOv5m (Epoch 21)	.4268	.3477	.3212	.1316
YOLOv5l (Epoch 28)	.4120	.3502	.3256	.1423
YOLOv5x (Epoch 20)	.4215	.3594	.3324	.1461
YOLOv5x w/ augmentation (Epoch 61)	.4661	.34	.3408	.1533

According to our experiment, the model with augmentation can reduce more validation loss and converge much later than the one without. With augmentation, although requiring more epochs, the model gains precision and mAP scores at the expense of recall score. This result coincides with our goal described earlier, that of a high precision score and satisfactory recall score.

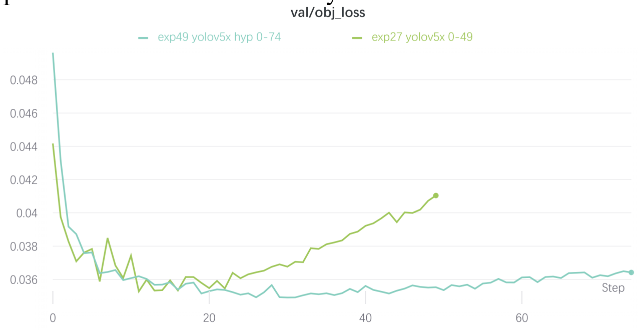


Figure 5. The Object loss of validation of YOLOv5x with augmentation and without

Also, data augmentation increases the number of class labels with a few images (Atelectasis, Pneumothorax), so it improves the mAP score a lot on detection of labels with relatively few images. The mAP and PR score of every class is shown in figure 4.

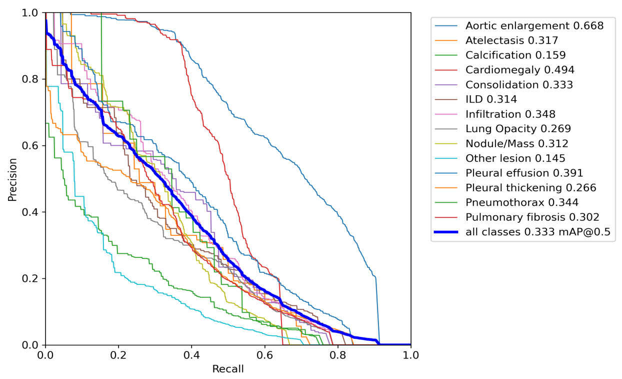


Figure 6. The Precision-Recall Curve of YOLOv5x with augmentation

Faster R-CNN: Under the abnormality detection circumstance, false-negative means that the model failed to identify chest abnormalities, which is an unwanted outcome. Meanwhile, we would rather have more false positives than false negatives which may cause patients to delay their treatments. Here, the goal of our abnormality detection model is to increase the precision value as much as possible, even at the expense of a part of the recall score. We believe our model achieves the goal.

Table 4. Faster R-CNN results

Faster R-CNN results			
Precision(macro)	***Precision(micro)***	***Precision(weighted)***	***recall***
.1580	.7763	.7956	.1548

C. Conclusion on models

Systematic evaluation the performance of different models on the dataset is made; the first evaluation is between several models from YOLOv5. There are 5 different networks under YOLOv5. Under experiments, it is found that all models converge before 50 epochs and start overfitting. YOLOv5x takes the most time to train and perform the best. After applying data augmentation to YOLOv5x, the degree of overfitting has been reduced. The models gained an obvious improvement in the detection of abnormalities with few labels. Second, we evaluate the performance of Faster R-CNN. The trained model is characterized by a high precision score but a low recall score. According to the result above, YOLOv5 possesses a high recall score but less precision score on the given dataset while Faster R-CNN has a high precision score but less recall score. Both models require further training to achieve higher accuracy, possibly through hyperparameter evolution. Although the two models are not able to output highly precise diagnoses, they can assist practitioners to determine chest abnormalities, saving time and effort.

V. DISCUSSION

In the ground truth images, we found that there are many small bounding boxes. However, due to the hardware restriction, we can only afford training models with images of 640 by 640. As our hardware situation improves, we will be able to use images of 1024 by 1024 to perform more training and examine the influence on PR and mAP scores.

When we explore the prediction of YOLOv5x, we discover two intriguing situations. Some labels predicted by our model with high confidence are not labeled by radiologists in ground truth images. For example, our model predicts "aortic enlargement" with a confidence level of 0.42 in a case. However, even though a similar pattern was labeled as "aortic enlargement" in the ground truth, this pattern does not appear in the ground truth image. The practitioners might overlook this abnormality when they labeled images. Also, unlike well-labeled datasets like COCO datasets, the same abnormalities are labeled by radiologist's multiple times (see figure 6), possibly because 14 radiologists participated in the labeling process. These labeled bounding boxes do not overlap correctly, so they may negatively affect the accuracy of the model. To ensure each abnormality to have one bounding box, we can reduce the number of bounding boxes. If most radiologists agree on a label, then replacing the wrong label of that bounding box with the right one be helpful. In addition, although both two models cannot achieve an extremely high accuracy to persuade doctors and patients completely believe on the system, we can help them in the other way by using a recommendation system. From the training/validation process, we collected confidence of each x-ray image. When a new x-ray image is detected by the system, the system will also provide several x-ray images with similarities so that users can compare their x-ray image with other x-ray images to assist their diagnosis.

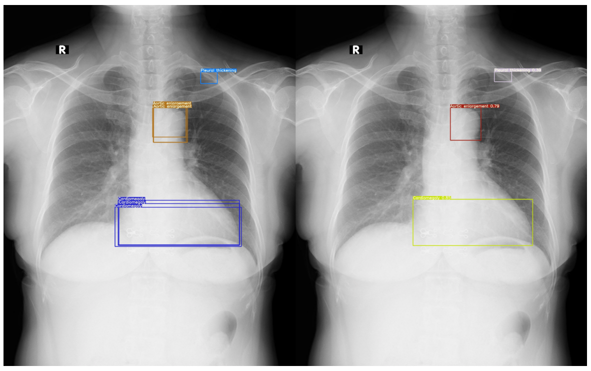


Figure 7. A sample of ground truth prediction of YOLOv5x

VI. CONCLUSION

In conclusion, this work about chest X-ray abnormalities detection will be potential and helpful to healthcare professionals like pulmonologists, and endocrinologists. Although our system is still at the first stage, it already proved the possibility of using neural network techniques to assist human beings in diagnosing diseases. It also helps people by giving similar X-ray images instead of jumping to a conclusion. We evaluated these two proposed neural network models in a reasonable way and our result confirmed the feasibility of detecting chest abnormalities through it. We also proposed a diagnosis suggestion which is provide radiologists similar x-ray images to perfect the detection and decrease mis-diagnose possibilities. Therefore, it evidently shows that it will be an innovative system with plenty of room to improve and will be beneficial to ourselves.

ACKNOWLEDGMENT

We gratefully knowledge the support of Kaggle.com of dataset VinBigData Chest X-ray Abnormalities.

REFERENCES

[1] VinBigData Chest X-ray Abnormality Detection: Automatically localize and classify thoracic abnormalities from chest radiographs: https://www.kaggle.com/c/vinbigdata-chest-xray-abnormalities-detection

[2] Brady, Adrian, Risteárd Ó Laoide, Peter McCarthy, and Ronan McDermott. "Discrepancy and error in radiology: concepts, causes and consequences." The Ulster medical journal vol. 81,1 (2012): 3-9.

[3] Bochkovskiy, Alexey, Chien-Yao Wang, and Hong-Yuan Mark Liao., "YOLOv4: Optimal Speed and Accuracy of Object Detection." Cornell Computer Science, 2004.