

Report of 2021 VRDL Final Project

Brief introduction

For this task, we are given some chest X-ray radiographs and are asked to come up with a model to output pneumonia regions (bounding boxes) for each graph if they exist. In essence, this is a structured learning problem in the field of machine learning, or, specifically, object detection in computer vision and visual recognition.

A CXR (Chest X-Ray) is comprised of white, gray and black. Metal and bones, which are substantially made of Calcium, appears white on the graph, while air is shown black on the graph. For such liquid/fluid parts as tissues and immune system cells, they are mostly in gray.

Healthy lungs are majorly full of air, so black should be the main color in such regions on the graph. However, if a patient has pneumonia, his/her lungs would be full of bacteria and immune system cells, which should appear gray on the graph. That is, we should find such gray regions that should originally be black. Such gray/whiter regions are called *lung opacities*.

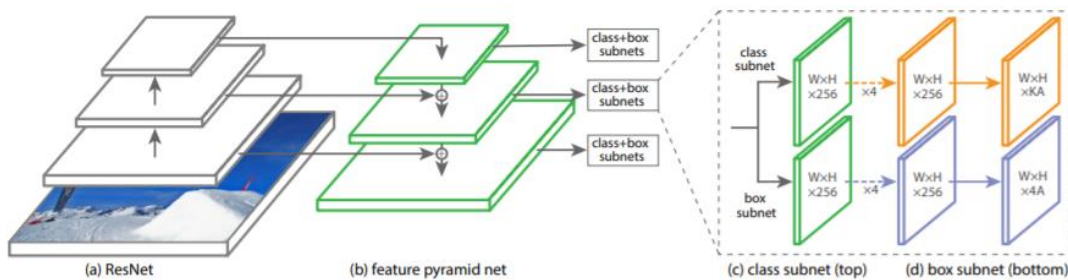
Although it seems that we are left with a simple task of reporting gray regions supposed to be black, there might be many issues out there. For the first thing, the qualities of CXRs are not guaranteed, which is pretty normal in practical and should definitely be overcome in the real world. By quality, there are some aspects of quality, e.g. positioning, angle, and contrast of black and white. These qualities vary from graph to graph. Another possible issue is that we may mistake a tumor for pneumonia. They are quite similar on the CXRs yet still distinguishable under some observations. Tumors usually have clear rounded shapes, whereas pneumonia regions usually appear hazy and blurry. There surely exist some solutions for the issues mentioned above, but due to time limitation, we did not put those into practice. Also, we argue that it's better to do lung localization first then perform object detection, since by doing so, we can limit the region of interest to some more specific parts of a graph.

Methodology

Introduction of RetinaNet

There are four components in a RetinaNet model architecture:

- a) Residual Network(ResNet) - which calculates the feature maps at different scales, irrespective of the input image size or the backbone.
- b) Feature Pyramid Network - a feature extractor generates multiple feature map layers
- c) Class subnet - It predicts the probability of an object being present at each spatial location for each anchor box and object class.
- d) Box subnet - It regresses the offset for the bounding boxes from the anchor boxes for each ground-truth object.



Besides the model architecture, the most important work is that the authors propose a new loss function-Focal loss.

One stage detectors generate 10000 - 100000 candidate boxes per image but only a few boxes contain objects. This imbalance causes two problems: (1)training is inefficient as most locations are easy negatives that contribute no useful learning signal.(2)the easy negatives can overwhelm training and lead to degenerate models. The authors think the extreme class imbalance is the central cause that one stage detectors have worse accuracy than two-stage detectors, and they claim that the focal loss can deal with this issue.

The cross entropy (CE) loss for binary classification is as follow:

$$CE(p, y) = CE(p_t) = -\log(p_t).$$

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{otherwise,} \end{cases}$$

where

The authors define the focal loss as:

$$\text{FL}(p_t) = -(1 - p_t)^\gamma \log(p_t).$$

Or in practice we use an α -balanced variant of the focal loss:

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

The idea is that if a sample is already classified correctly, we can significantly decrease or down weigh its contribution to the loss. This allows the model to incorporate the losses made on the misclassifications arising only from the very low probabilities.

Data pre-process

We resize image to 224*224, and divide the 25684 training images into a large training set (24399 images, 95%) and a small validation set (1285 images, 5%)

Model architecture

In this final project, we trained two RetinaNets and their backbones are resnet-50 and resnet-101 respectively. For the FPN in RetinaNet, we use linear interpolation to upsample the features, and use 2D convolution to produce P3~P7 features, which are defined in the paper of RetinaNet. The classification subnet and box regression subnet are composed of five convolutional layers.

Hyperparameters

We use SGD with learning rate = 0.01, momentum = 0.9, decay = 0.0001 as our optimizer. We trained both models for 75 epochs with 2500 steps per epoch and batch size of 8. During training, we use augmentations, such as rotation, translation, scaling, and horizontal flipping.

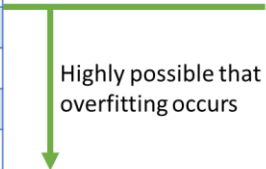
We use non-maximum suppression=0 to eliminate any overlapping bounding boxes in each network's output. We take weighted averages of overlapping bounding boxes from both trained neural networks, using the scores for the boxes from each network as the weights. For bounding boxes that did not overlap between the two neural networks, we used a separate higher threshold value(=0.15) to decide whether the solitary box should be retained. We also shrink all final bounding boxes by a factor of 0.83.

Experimental results

YOLO_v5

First, we try YOLOv5 and record the results at different numbers of epochs and different bounding box thresholds as below:

| model | # of epochs | Bbox_th | Private score | Public score |
|---------|-------------|---------|---------------|--------------|
| YOLOv5s | 10 | 1e-1 | 0.12432 | 0.02160 |
| | | 5e-2 | 0.11700 | 0.03395 |
| | | 15e-2 | 0.11157 | 0.00793 |
| | 20 | 1e-1 | 0.15505 | 0.08730 |
| | | 5e-2 | 0.12931 | 0.03240 |
| | | 15e-2 | 0.13562 | 0.08730 |
| | 30 | 1e-1 | 0.13893 | 0.08641 |
| | | 5e-2 | 0.12057 | 0.06134 |
| | | 15e-2 | 0.13076 | 0.02380 |
| | 40 | 1e-1 | 0.13858 | 0.04761 |



RetinaNet

Then we build RetinaNet with Resnet50 and Resnet101, and experiment different bounding box thresholds as well:

| Model | BBox_th | Private score | Public score |
|-----------|---------|---------------|--------------|
| ResNet50 | 0.05 | 0.15871 | 0.05709 |
| | 0.1 | 0.17121 | 0.07341 |
| | 0.15 | 0.16844 | 0.07341 |
| | 0.2 | 0.15372 | 0.07341 |
| ResNet101 | 0.05 | 0.16476 | 0.04166 |
| | 0.1 | 0.16257 | 0.04861 |
| | 0.15 | 0.14555 | 0.05555 |
| | 0.2 | 0.13139 | 0.02380 |

We choose the best results from two models, ensemble them, and experiment different NMS thresholds:

| Model | NMS_th | Private score | Public score |
|----------|--------|---------------|--------------|
| Ensemble | 0 | 0.17532 | 0.05753 |
| | 0.1 | 0.17548 | 0.05753 |
| | 0.2 | 0.17585 | 0.05753 |
| | 0.3 | 0.17133 | 0.05753 |
| | 0.4 | 0.16598 | 0.05753 |

Although we have tried different combination on three thresholds, we found that the best score we got is (ResNet50_BBox_th, ResNet101_BBox_th, Ensemble_NMS_th) = (0.04, 0.05, 0). Noted that the threshold of the ResNet50 is not the one that produce best private score. The result is shown as below:

| Model | BBox_th | NMS_th | Private score | Public score |
|-----------|---------|--------|---------------|--------------|
| Resnet50 | 0.04 | - | 0.14698 | 0.04282 |
| ResNet101 | 0.05 | - | 0.16476 | 0.04166 |
| Ensemble | - | 0 | 0.17916 | 0.05034 |

Summary

In this object detection task, we try multiple models like YOLOv5, RetinaNet with Resnet50 and Resnet101 as backbones. Our best model is an ensemble model with two RetinaNet, and our private score is 0.17916.

According to our experiment results, we think that choosing proper bounding boxes and NMS thresholds is important. Also, we find that ensemble model may improve the performance when single model is not good enough.

GitHub link of our code

https://github.com/Lucas-Kuo/VR_DL_Final

https://github.com/yahan1011/VRDL_FP

Reference

1. <https://medium.com/@kimujan/retinanet-%E8%AE%80%E5%BE%8C%E6%95%B4%E7%90%86-af821dcd4558>
2. <https://arxiv.org/pdf/1708.02002.pdf>
3. <https://github.com/pmcheng/rsna-pneumonia>
4. https://www.researchgate.net/figure/The-network-architecture-of-Yolov5-It-consists-of-three-parts-1-Backbone-CSPDarknet_fig1_349299852
5. <https://jonathan-hui.medium.com/understanding-feature-pyramid-networks-for-object-detection-fpn-45b227b9106c>
6. <https://medium.com/image-processing-and-ml-note/yolo-v5-explained-and-demystified-22f041ad316e>
7. <https://medium.com/swlh/focal-loss-what-why-and-how-df6735f26616>
8. <https://www.kaggle.com/zahaviguy/what-are-lung-opacities/notebook>

Screenshot of our rank

| | | |
|---------------------------------|---------|---------|
| stage2_test.csv | 0.17916 | 0.05034 |
| 3 days ago by yhchang1011 | | |

Teamwork of each task

| Tasks | contributors (%) |
|--------------------------------------|---|
| Literature survey | 309657009 (33%), 310553004 (33%), 109550162 (33%) |
| Approach design | 309657009 (33%), 310553004 (33%), 109550162 (33%) |
| Approach implementation (experiment) | 309657009 (33%), 310553004 (33%), 109550162 (33%) |
| Report writing | 309657009 (33%), 310553004 (33%), 109550162 (33%) |
| Slide making and oral presentation | 309657009 (33%), 310553004 (33%), 109550162 (33%) |