Title: Final Report

G050 (s1720557, s1749201, s2110911)

Abstract

The objective of this report is to complete object detection on medical x-rays, and be able to provide bounding boxes and labels to potential abnormalities in chest scans in order to provide assistance to medical professionals. The main focus to determine the feasibility of using modern 'anchor free' approaches in this task. The chosen 'anchor free' model is CornerNet, and a detailed description of its operation is provided. Initial validation set results give a score of around 0.16, however through the removal of duplicate labels this is increased to 0.183. Faster R-CNN as a baseline model for anchor based approaches to act as a comparison for conventional state of the art techniques. This obtains a score of 0.21 on the validation dataset, bettering the score from CornerNet. These models are evaluated on the Vingroup Big Data Institute's Chest X-rays dataset which consists of 18,000 chest x-rays, including 15,000 which are labelled by a panel of radiologist giving the location and type of abnormalities perceived. Our main conclusion from the experiments is that Faster R-CNN (and anchor approaches in general) is better suited to this task, however the training and inference times are greatly increased, with CornerNet training and validation taking 1.33 hours and 0.5 hours respectively, whereas Faster R-CNN takes 3.5 hours and 1 hour respectively. This is important when results from tests are required quickly, and it can perhaps be argued that CornerNet is more applicable for use in the real world due to its superior inference time. We have also concluded that CornerNet is very impacted heavily by class imbalances, with popular classes performing significantly better than less popular classes.

1. Introduction

The field of medical imaging analysis conducted by machine learning models has been growing rapidly over the past few decades, due to its ever increasing accuracy and speed when compared to trained doctors. Coupled with the growing demands on health services in the wake of COVID-19, it is becoming increasingly likely that machine learning will be used to reduce the workloads of trained medical professionals. Dr. Ziad Obermeyer of Harvard Medical School predicts that "In 20 years, radiologists won't exist in anywhere near their current form. They might look more like cyborgs: supervising algorithms reading thousands of studies per minute" (Tedeschi, 2016). For these reasons our project will focus on interpreting chest x-rays - more specifically, we will look to detect and locate multiple potential lung abnormalities within these images. We will be using the Vingroup Big Data Institute's ChestX-rays dataset, as described in Section 3. With this dataset, a model could be produced which is able to provide assistance to medical professionals, and help to identify and locate abnormalities which are easily missed. The motivation is to try and reduce missed diagnosis of overworked medical staff - not to replace, as there would likely be various legal and ethical considerations to make.

Traditionally, the use of deep learning in this area has focused purely on classification, i.e. the model only gives an indication of the presence or absence of certain abnormalities (Wang & Xia, 2018) (Rajaraman et al., 2020). For example a model which can determine whether a person has pneumonia or not. These models are based upon fairly standard CNN models, although sometimes utilise more advanced models such as Faster R-CNN (Rahmat et al., 2020), and provide limited assistance to medical professionals. Our project will focus on (multiple) object detection of abnormalities, which will not only provide a doctor with a list of potential abnormalities, but will also provide bounding boxes for their locations (kag).

We will predominately be considering the applicability of using an anchor-free approach (CornerNet) to object detection, as a modern method of completing this sort of task when compared to more conventional anchor based apporaches.

2. Related Work

Conventional approaches to general object detection can be split into two distinct categories; one stage detectors, and two stage detectors. Two stage detectors have a focus on model performance over anything else, sometimes making their use in real-time applications limited. One stage detectors are optimised for speed of inference, at the cost of accuracy (Jiao et al., 2019).

The most popular family of two stage models are Region Based Convolutional Networks (R-CNNs), with the Faster R-CNN being a prime example of how this family of models functions, and acting as a baseline as evaluated in Section 5.1. Faster R-CNN is based on the approach of splitting the object detection problem into two parts; a region proposal

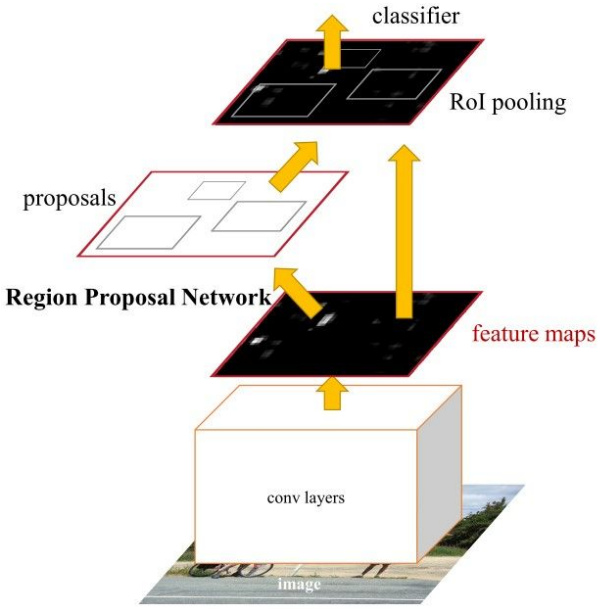


Figure 1. Faster R-CNN Structure

network (RPN), and a Fast R-CNN. The job of the RPN is to recommend regions within images which may contain objects, and to provide potential categories for each of these regions. The Fast R-CNN component then extracts features from these regions and proposes bounding boxes and labels (Ren et al., 2016). A standard workflow is as follows: images are fed into a pre-trained CNN (such as VGG-16) which acts as the backbone of the network. The output features are then input into the RPN which uses anchor boxes in a CNN to make proposal of regions to focus the Fast R-CNNs efforts. The Fast R-CNN takes these proposals, along with the features from the backbone, and they are fed into a region of interest layer (RoI) which classifies images within proposed regions, and fixes a bounding box around them. Figure 1 details a simplified view of this process (Li, 2020).

In general, anchor boxes are defined as a large set of hand-crafted boxes indicating the ideal shape, location, and size of the objects within a dataset. These are used to predict object locations, and classifications, based on the similarity of features to these boxes (Solawetz, 2020; Christiansen, 2021).

The main one shot methods are the YOLO family of models. In YOLO, the input image is divided in to a $n \times n$ grid, where each grid cell is responsible for predicting objects centred on that cell (Zhao et al., 2019). The more mature versions of these models also utilise anchor boxes, which allow for multiple overlapping objects (potentially of different classes) to be detected within the same grid cell. The anchor boxes are of different pre-determined aspect ratios which are manually set.

These anchor based approaches has several drawbacks which can cause problem in certain circumstances. Firstly, as the anchor boxes are hand-crafted, they are not flexible and can struggle to handle objects of varying sizes. This could be particularly harmful to our task as the size of the training bounding boxes varies massively. In such cases, knowledge in the working domain might be required. It is possible to tune the anchor box sizes, however this adds further hyperparameters to the model. Secondly, to achieve high accuracy requires large computational costs as a large number of bounding boxes will be required. This then produces a problem whereby a large proportion of the anchor boxes are covering areas with no classification creating an in-balance when training.

Some recent advancements in object detection have attempted to utilise 'anchor-free' approaches, which look to remove the need to predetermine anchor boxes. Three such examples are CornerNet:Detecting Objects as Paired Keypoints (Law & Deng, 2019), CenterNet: Keypoint Triplets for Object Detection (Duan et al., 2019), and FoveaBox: Beyound Anchor-Based Object Detection (Kong et al., 2020). CornerNet and CentreNet models work similarly through the use of heat maps, whereas FoveaBox learns semantic maps. We evaluate the use of CornerNet in this report to determine its usefulness.

There are limited examples of existing papers which tackle the problem of object detection in lungs/chests. This is likely due to the lack of suitable datasets available until recently. Previous large scale datasets simply labelled each kind of abnormality as present or not, making the localisation problem more difficult. One such dataset is the NIH Chest X-ray dataset (goo).

Two papers which did try to tackle this problem approached it in a very similar way, by looking to combine separate CNNs (such as AlexNet, VGGNet-16 and ResNet-152 etc.) to produce an ensemble prediction; these are "Abnormality Detection and Localization in Chest X-Rays" (Islam et al., 2017) and "Detection and visualization of abnormality in chest radiographs using modality-specific convolutional neural network ensembles" (Rajaraman et al., 2020). These models both look to run a number of separate CNN's, and then combine their feature sets by some metric in order to obtain a heatmap to indicate location, as well as classification results. Although these papers report results, there is very little to compare them against, and they do not determine bounding boxes.

One paper which does tackle the object detection directly, and utilises a standard Faster R-CNN is "Deep Neural Network for Foreign Object Detection in Chest X-rays" (Santosh et al., 2020). The purpose of this paper is to detect circular foreign objects on chest/lung x-rays. This task differs slightly however from ours, as the objects being detected are of a fairly standard shape and size. It does however give confidence that Faster R-CNN will be applicable to chest x-ray applications, and will work as an appropreate baseline. R-CNNs have also been used in related applications, for example "Object Detection in X-ray Images" focuses on the use of R-CNN for object detection on airport baggage scanner x-rays (Wang, 2020).

Anchor based approaches (including both one and two stage

models) have shown to be very diverse, and perform very highly in a number of different domains. As anchor-free approaches are more novel, we would like to determine whether they have a future in medical imaging applications when compared to the more thoroughly tested anchor approaches.

3. Data set and task

The dataset used is the Vingroup Big Data Institute's Chest X-rays. 18,000 individual x-rays are provided in the DICOM format (a standard in medical imgaing) each labelled by radiologists for the presence of 14 different classes of lung abnormalities. These labels also include bounding boxes for each of the identified abnormality represented by coordinates of the four corners. 15,000 of these images are allocated for training (including training set and validation set), while the remaining 3,000 is used for testing. However, as this is a competition dataset, the labels from only 10% of the test set have been made public. There can be more than one class (type of abnormality) per image, thereby posing this problem as a multilabel classification.

The x-ray scans show lungs, surrounded by tissue which consists of skin, bones, muscles etc. When the scan is taken, dense parts of the body absorb more radiation, resulting in a lighter output on the x-ray scan. Conversely, less dense areas appear darker on the scan as little radiation is absorbed. In general, black represents air, white represents bone, and shades of grey represent tissue and fluid (Zahaviguy, 2018).

Lungs are the main focus of this task, and they should generally appear black, as they are not dense, there will however be light areas due to overlapping bone/tissue etc.

In general, the abnormalities which can be observed will show as a differing density to what is considered normal. In some of the abnormalities such as calcification (of bones within the centre of the chest), and cardiomegaly (which appears at the base of the lungs) they are only observed in very specific places within the chest, others can appear more globally or over a wider area.

Figure 2 shows a healthy x-ray, whereas Figure 3 shows an x-rays where various abnormalities have been identified. Some of these alterations are very subtle and not easily observable to the untrained eye, making this a challenging task.

A brief exploration of the data revealed certain properties about the class distribution. Figure 4 illustrates the frequency of classes in the training data. Note that class 14 consist of x-rays with no observable abnormality. As such these images will have no corresponding bounding box and is labelled by only the one class. This class is by far the dominant class in the training data, which could result in challenges arising from class imbalance.

An analysis of the number of bounding boxes per image was also conducted. Note that the labelling for each image was provided by multiple radiologists and therefore

Figure 2. Chest X-Ray with no Abnormalities

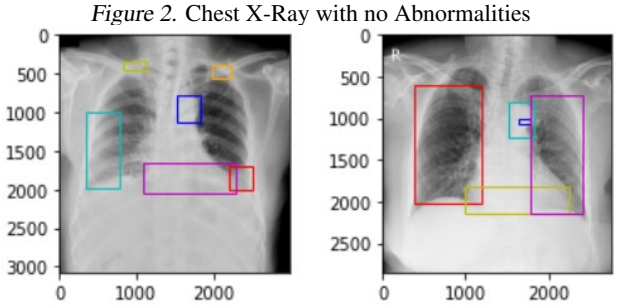


Figure 3. Chest X-Rays with various Abnormalities

there exists overlap between one radiologist's bounding boxes and another. This causes the same abnormality to be labelled multiple times (albeit with slightly different coordinates resulting from the idiosyncratic labelling by each radiologist). This behaviour should not be reflected by the neural network and thus these duplicates need to be removed in preprocessing. Since the data is anonymised, there is no clear way to isolate labels from one radiologist to another. We also cannot assume that an image only contains one instance of a detected class as there is a possibility that the class is detected in two or more separate places (both of which require separate bounding boxes). Therefore, a heuristic approach is employed where different boxes with all four identical corner locations in a image (within a threshold of 200 pixels) are assumed to bound the same object. Hence, only one of these boxes is considered.

Following this preprocessing step, we can plot the distribu-

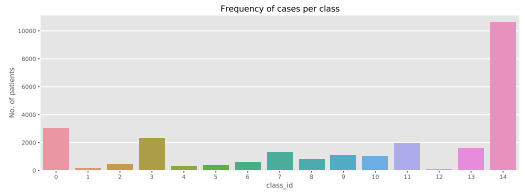


Figure 4. Frequency of classes in the training data

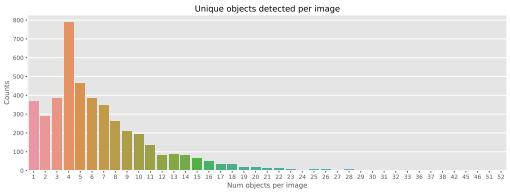


Figure 5. Distribution of unique objects per image

tion of the number of unique bounding boxes per image as shown in figure 5. Note that these counts do not include images in class 14, which will always have zero boxes. The majority of images have 4 boxes detected, while the maximum number or objects detected in a single image was 52.

As mentioned in Section 1, certain one shot algorithms require anchor boxes with predefined as aspect ratios. A good approach is to define these aspect ratios to approximate the aspect ratios of the different classes of objects in the training data. Certain image classification tasks rely on the distinct aspect ratios of different classes to produce these anchor boxes. To see if this is the case for our dataset, a boxplot of the bounding box aspect ratios per class is created as shown in figure 6. Though some classes (3 for example), do show a unique and narrow distribution of aspect ratio, the majority of other classes have largely varying ratios that, as argued in Section 1, could pose a challenge for manual anchor box construction.

This task is evaluated using the mean Average Precision metric (mAP) defined in the PASCAL VOC 2012 dataset competition (Everingham et al., 2010), which is standard for object detection tasks. For each class in the dataset, a predication is considered a true positive if it shares the same label as the ground truth, and the intersection over union (IoU) is greater than 40% (0.4) (Cartucho).

4. Methodology

4.1. Baseline: Faster R-CNN

Faster R-CNN is used as a baseline to our experiments, and is trained on the training dataset described in Section 3. This model architecture is used as Faster R-CNN is the most common and up to date network used for object recognition. Faster R-CNN also has the advantage of being one network unlike some of its alternatives, this gives an advantage that it is simpler to train.

The Faster R-CNN model requires that the background class has class label 0. For our data the background class has label 14, to get around this we swapped the class labels of 0 and 14 before passing it into the model. Then when looking at the outputs of the model we swapped them back.

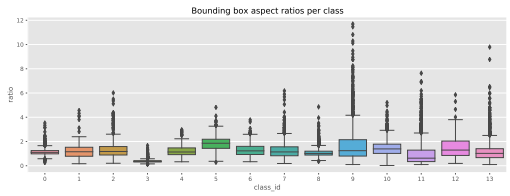


Figure 6. Bounding box aspect ratios

Before the image is passed to the model it is normalised so that each of the colour channels has 0 mean and a standard deviation of 1. The image size is also altered to be a maximum of 255 pixels.

The model that we used as our baseline is the default Faster R-CNN network in the pytorch library. The code for this implementation of the model can be found in the References (tor).

The Faster R-CNN model needs to have a backbone, for this experiment we used a ResNet-50-FPN backbone. It has several hyper-parameters that can be tuned, these include the number of trainable backbone layers, batch size, learning rate, weight decay coefficient and momentum coefficient. The only hyper-parameter here that is not standard is the number of trainable backbone layers. The name of the hyper parameter gives away its role, it allows the model to be more or less flexible depending on the task it is being used for.

4.2. CornerNet

The methodology applied in our experiments follow closely that of the original paper (Law & Deng, 2019). We will nonetheless restate some of key procedures for clarity. Any deviations from the original, such as the choice of hyper-parameters, will be mentioned explicitly. The network is constructed to produce two main outputs, the heatmaps of the top-left and bottom-right corners of the predicted objects. Together these are sufficient to construct the bounding boxes required for the object detection task.

However, the network also requires two additional outputs to supplement these heatmaps, which are the embeddings and offsets vectors. The first of the two is required as the object detection task might necessitate multiple detections of the same class within one image. In this respect, simply outputting the corners for each class is insufficient, as the user would have no concrete information as to which top-left corner is associated with a bottom-left corner and vice-versa. Hence, these embedding vectors are required as outputs for each corner predicted. The key idea is that the distance between these embeddings should be as close as possible for corners that are associated with the same detection and large between corners of different detections. The offset vectors are less crucial, but are present to ensure tighter fits on the bounding boxes. Briefly, these compensate for the rounding effects that occur when the original input image is downsampled to the heatmaps. These therefore provide the necessary offsets for when the model remaps the heatmaps to the original input image size.

These three 'heads' of the network are contained within the prediction module. Before reaching to this stage, the input is processed through the 'backbone' of the network.

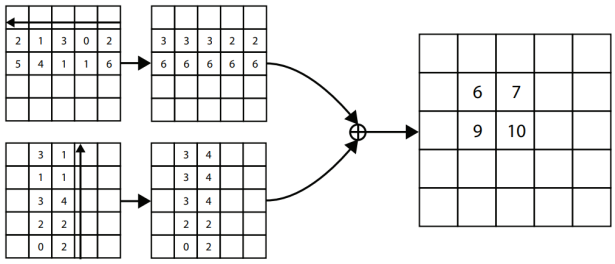


Figure 7. Corner pooling procedure (Law & Deng, 2019)

The backbone architecture utilized consist of two sequential hourglass modules (Newell & Deng, 2017). The term hourglass is in reference to the downsampling and consecutive upsampling design of the module. This architecture is useful in object detections since it captures both global and spatial features in a singular structure. To remain within our page limit, the finer details of the module architecture is omitted from this report but can be referred to in the original paper (Law & Deng, 2019). The output of the hourglass modules is then input in to two separate prediction modules, one for all top-left corner predictions and one for the bottom-right corners. Within each prediction module, a 'corner pooling' layer is present to localize respective corners. This step is unique to CornerNet since the local information at the corners of bounding boxes are limited for the cases when the object is not a rectangular shape, as is the case for almost all objects in our dataset. The step involves a max-pooling across the appropriate axes. For instance, in the case of the upper-left corner detection, for each pixel (i, j) on the input feature map, the corner pooling assigns the maximum of all pixels to the right of (i, j) (within the same row). Similarly, for each (i, j), the pooling assigns a value that is the maximum of all pixels to the bottom of (i, j) (within the same column). These two procedures have the effect of propagating information along the axes to each pixel on the feature map. These two value assignments are then summed to give the location of the corner. An illustration of the process is presented in figure 7. The process is repeated for the bottom-right detection, this propagating information from the left and downwards.

With the corners localized, each prediction module can now output the information required for the bounding boxes. During training, each of the three components has an associated loss function(s) (the embeddings has two loss functions). The primary loss is a variant of the logloss between the target heatmap and the predicted heatmap. There will C total heatmaps associated with each class except the background class (14 in our case). Below is an expression for the focal loss, L_{det}:

$$-\frac{1}{N}\sum_{c=1}^{C}\sum_{i=1}^{H}\sum_{j=1}^{W}\begin{cases}(1-p_{cij})^{\alpha}\log(p_{cij}) & y_{cij}=1\\ (1-y_{cij})^{\beta}(p_{cij})^{\alpha}\log(1-p_{cij}) & otherwise\end{cases} \tag{1}$$

The loss is calculated at every pixel location i, j and every heatmap channel c. p_{cij} is the predicted value while y_{cij} is the ground truth. Two hyperparameters α and β contribution of each point to the loss. The embedding loss is termed as L_{pull} and L_{push} where the first is constructed to train the network to group associated corners together while the second trains the network to increase the distance between embedding vectors for corners of different boxes. The final loss function, L_{off}, is for the offset outputs and is set to be the L1 loss between the predicted and the ground truth. Further details of these loss functions can be referred to in the original paper. The total loss is defined as the weighted sum of these losses as follows:

$$L = L_{det} + \alpha L_{pull} + \beta L_{push} + L_{off} \tag{2}$$

The majority of code for the model is adapted and modified from the following repository https://github.com/zzzxxxttt/pytorch_simple_CornerNet.

5. Experiments

5.1. Baseline: Faster RCNN

For our baseline experiment the number of trainable backbone layers was set to 3. The learning rate was set to 0.005, because during training there were issues with the model producing NaN gradients for the loss. One solution for this problem was to increase the learning rate, which we had originally set at 0.0005. The weight decay coefficient and the momentum coefficient were set to 0.0005 and 0.9 respectively. The optimizer we used for this experiment was Stochastic Gradient Descent.

We opted to use the jpeg version of the data for this experiment. This is because the CornerNet model used in the subsequent experiments required the jpeg data, and we thought it would be best to use the same data for the baseline so that the two methods are comparable. Another reason for making this decision is that the jpeg files are significantly smaller when compared to the Dicom ones. This gives the advantage of taking up less memory, which was already a bottleneck in our system. The Dicom files also contain several pieces of meta data which is unnecessary for the baseline experiment.

The training data was split 80:20 as training set : validation set. This gave 12000 training images and 3000 validation images.

The batch size was set to 2, this is due to memory issues that occurred when training the model. We believe the memory issues were caused by the size of the model and also the size of the images. Because of this small batch size, the model took a long time to train the whole training set.

The training loss graph shown in Figure 8, shows a gradual decreasing starting from a value of 0.51, down to a value of 0.25. At this lower value the losses flatten out fluctuating slightly as the epochs continue. The model appears to be fully trained around 70 epochs, as such this used to evaluate the model with the aim to avoid over-fitting to the data.

mAP is used to evaluate the model as described in Section

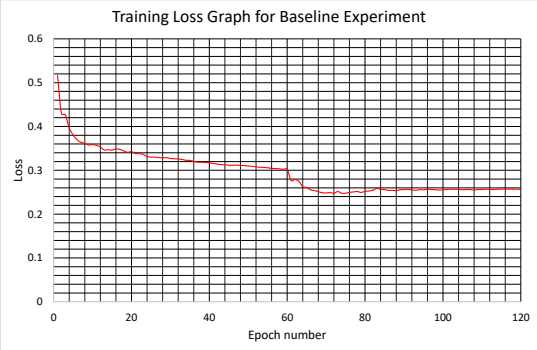


Figure 8. Training loss graph for baseline experiment

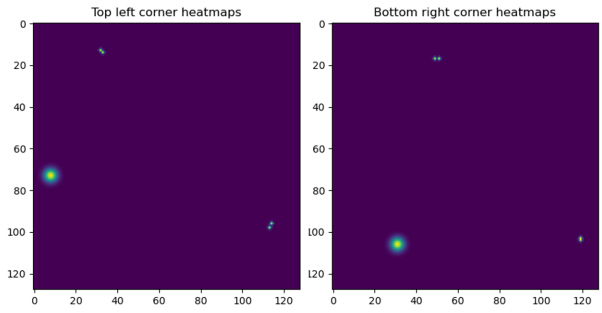


Figure 9. Example ground-truth heatmaps (class 10)

3, with the same IoU as it used on the CornerNet model (0.4). For the final baseline model, the mAP validation dataset score is 0.21. Each training epoch took around 3 hours and 30 minute train, and each validation epoch took around 1 hour.

5.2. CornerNet

In our experiments, the α and β in expression 1 were set to 2 and 4 respectively in accordance with the original paper. Meanwhile the α and β in equation 2 were both set to 0.1. The Adam optimizer was employed in training (Kingma & Ba, 2014) with a minibatch size of 4 (maximum given the input image size and hardware limitations). The complete provided training is split into a 80:20 ratio of training to validation images which equates to 12,000 and 3,000 images allocated for training and validation respectively. All training images were resized to 512x512 and standardised. Image augmentations only include random image flipping. An initial learning rate of 5×10^{-4} was used. Training was performed in batches since the small batch size and complex network architecture meant training time was lengthy. Each batch consisted of 5 epochs. After the third batch the learning rate was reduced to 1×10^{-4} and consequently to 1×10^{-5} after the fifth.

An example of ground-truth heatmaps generated from the top-left and bottom-right bounding box labels is shown in Figure 9. Note that the larger boxes (as observed by a larger separation of the two corners) have corners with larger 'bumps'.

This is due to the way these heatmaps are encoded to allow corner locations to be more flexible for larger boxes. The assumption being that the object can still be captured with minor variations of the corner locations. Hence, there is no need to be strict with the precision of such corners. To implement this, the model utilizes 2D Gaussian 'bumps' of varying radius to mark the corner locations on the heatmaps. The corners of larger boxes will be mapped with Gaussians of larger radius.

The training results of the four losses, as well as the Overall loss, are given in Figure 10. A total of 29 training epochs were performed spanning across 6 training runs.

All four training loss' can be seen to very noisy during training. This is probably due to the small minibatch size causing high variance. This minibatch size cannot be increased futher due to cluster GPU memory constraints. However, the mean losses (averaged over the complete epoch) as shown by the solid curves, are much more stable during training. The mean focal loss shows a positive decline indicating that model is gradually making progress in producing the correct heatmaps. The other losses are more or less stationary (with a slight decline) at a low value throughout training. This appears to suggest that the model has no difficulties in ensuring the correct corners are paired for each box - meaning that the corner pooling layer works well. The push loss is consistently higher than the pull loss during training, an observation that is understandable given the push loss involves measurements between each corner and every other corner i.e. it is easier to ensure the corner embeddings are similar for matching pairs than it is to ensure the embeddings are dissimilar to all other corners. The model also has minimal issues in adjusting the corner locations to account for the downsampling as shown by the low offset loss.

mAP is used for evaluation, a described previously. A validation run was performed at the end of every training epoch. The mAP calculated is across the 14 classes (not including the background class) and the entire 3,000 images in the validation set. A plot of the validation mAP throughout the 29 epochs is shown in figure 11.

Figure 10. CornerNet Training loss

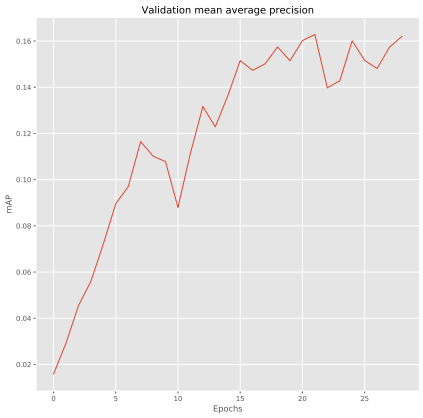


Figure 11. CornerNet Validation mAP

The results show a continuous trend of improvement across training. Since training was performed in batches, we could manually lower the learning rates in response to the mAP scores following each training batch. For instance, the first reduction was performed after epoch 10, where the mAP was seemingly more volatile. Similarly, another reduction was performed at after epoch 20. Close to the 25th epoch, the mAPs were fluctuating around 0.16. Based on this observation and the limitation involving extended training times, it was decided to stop training beyond the 29th epoch. This is to ensure sufficient time for further experiments. An example of the sample prediction bounding boxes against the ground truth bounding boxes is shown in figure 12. One can see from the samples that, the presence of multiple overlapping ground truth boxes (in blue). This is, as mentioned, a property of the training data which is labelled by multiple radiologist, resulting in the same objects being labelled twice or more.

This property is not conducive for the model training, since it is not a behaviour we intend to see replicated in our predictions. In sample 3 of figure 12, we can observe at least 3 ground truth boxes labelled for what is most likely the same detection. This inadvertently causes our model to predict two bounding boxes (in red) for that one detection. The proximity of the two boxes also causes some complications when their corners overlap, risking embedding errors where the model erroneously pairs corners for two different objects. The presence of similar errors in certain tests were also observed in the original paper.

We also believe that is at least partly responsible for the behaviours observed in Figure 14. The red bars of this graph detail the Average Precision (AP) for each class. If the red bar lines considered, there is a close correlation between this graph and Figure 4 which details the class frequencies. The two classes (1 and 12) with the lowest frequency achieve the poorest AP. We believe that a large class imbalance is leading to this poor performance. Class 1 is Atelectasis, and occurs when part of the lung collapses, and increased density is seen on the lungs. Class 12 is a Pneumothorax, and is also when part of the lung collapses. This is seen observed as a light spot where the lung collapsed, and as a dark spot where the collapsed lung fell into (Sakuraandblackcat, 2021). These classes have a lot in common as they are both cases of collapsed lung which can be observed by changes in lung density. Class 4 (Consolidation) is also observed in a similar way, so it is possible that these are difficult to distinguish.

To remedy the issues surrounding the duplicate labels, we propose removing these duplicates from the training data by using the same approach as in section 3. Additionally, we impose a further restriction such that only strongly overlapping boxes of the same class will be removed. We processed our training data accordingly and performed model training with the exact configurations as before to enable a fair comparison. The validation mAP comparison is shown in Figure 13.

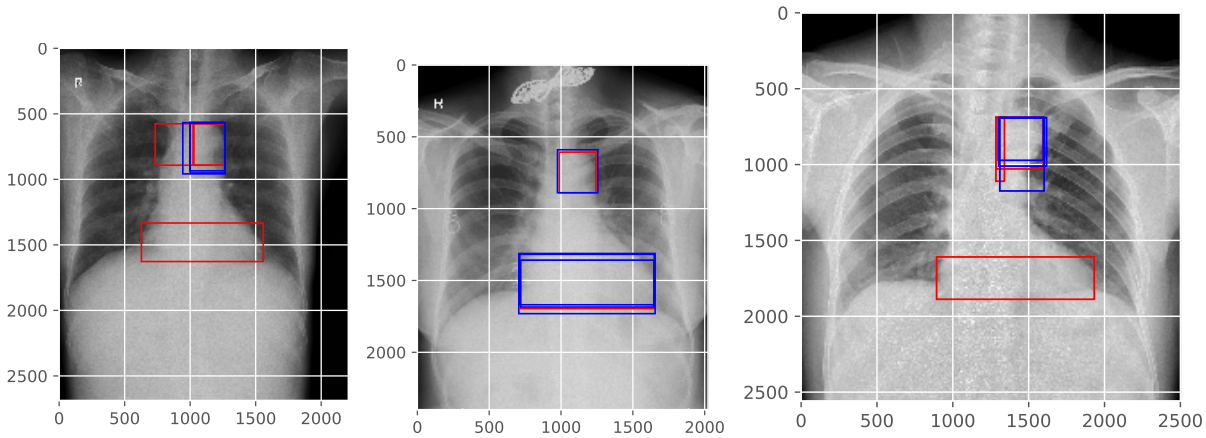
The three training runs spanned across a total of 18 epochs, yielding at the end, a validation mAP of 0.183. At almost all the validation intervals, the model trained on no duplicates dataset obtained better results than that on the original. Its important to note, that while the model hyperparameters and network architecture are kept constant in this comparison, ultimately, the validation set is not, since it too is derived from the training data (and therefore exposed to the same problem of duplicate labelling). Hence, the comparison is not truly be on 'level' terms since, the preprocessing performed on the training set will and should be replicated on the validation. To faithfully evaluate the performance of the two models, we would need a test set, one which we can guarantee that no duplicate labelling exists. This being a Kaggle competition, the provided test set is not accompanied with labels, since these are reserved by the competition organizers for final testing. If we assume that the test labels are devoid of any duplicate labelling, then a comparison can be made with the test mAP. However, even without a level comparison, it is generally in the best interest of the model designer to utilize the best quality data for training.

However, this procedure has had a slightly unexpected impact on the Average Precision (AP) across each class. In general an increase in AP is seen in the popular classes, and a decrease in AP is seen in the unpopular classes, making the problem of class imbalance more prevalent. It is hypothesised that by removing duplicates, there is now not enough variability in the smaller classes to train effectively. Perhaps in future test duplicates should not be removed from the less populous classes in order to provide a better class balance.

6. Conclusions

The main conclusions which can be drawn from the experiments are as follows. On this task, CornerNet performs poorer than the baseline model (mAP of 0.183 vs 0.21).

This is somewhat as expected as these two-shot methods generally deliver greater performance at the cost of greater



(a) Sample 1 (b) Sample 2 (c) Sample 3

Figure 12. Sample predictions (red) vs ground truths (blue)

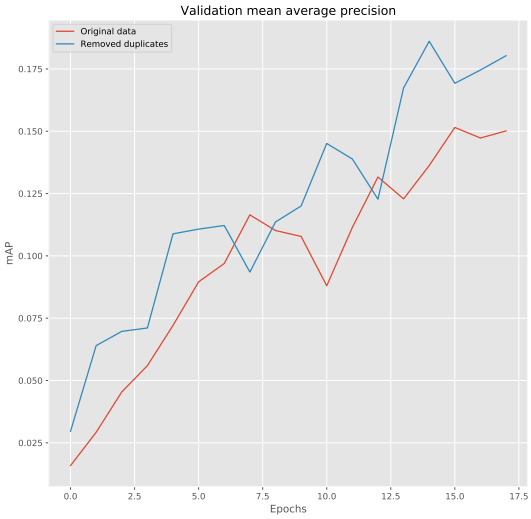


Figure 13. Validation mAP comparison between models trained on original and no duplicates

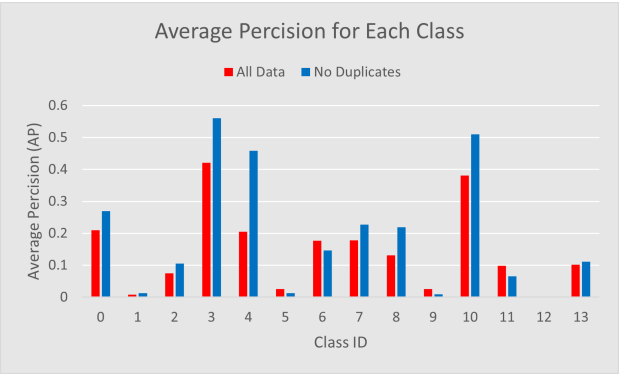


Figure 14. Validation Average Precision (AP) for each Class (Excluding Background) for both with and without duplicate data

computation. As mentioned in the previous section, these models utilize large sets of anchor boxes that are densely spread across the image during training. This is required to ensure sufficient overlap with the ground truth boxes to ultimately give reasonable performance. This process can be considered a brute force approach when compared with anchor-free approaches such as CornerNet. This fact is demonstrated in the drastically different training time between the two models. CornerNet was able to complete 1 epoch of training within an average period of 1 hour and 20 minutes while the baseline Faster-RCNN took on average 3 hours and 30 minutes. Similarly, during the inference phase, the Faster-RCNN model also took more than double the time required to produce the evaluations (1 hour vs less than 30 minutes for CornerNet). Taking these factors into consideration, we can make an argument for CornerNet's better efficiency and practicality when it comes to actually deploying the model for real world use.

We can also conclude that CornerNet's performance is greatly impacted by large class imbalances. Figure 14 and Figure 4 are remarkably similar in their peaks and troughs, with popular classes performing significantly better than unpopular ones. In future research, more consideration would need to be made to ensure that class imbalances are removed possibly through different sampling (oversampling, undersampling etc.) methods.

We also know that it performs poorly when there are a large number of duplicate labels / bounding boxes. When these are removed two things happen, the overall performance of the model improves (due to improved AP of high population classes) and the performance of low population classes deteriorate. This is again most likely related to class imbalances.

Finally, a submission was made on Kaggle for our CornerNet test set predictions. The returned score was an mAP 0.089. Although, this is less than our validation score, its important to note that this value was derived on only 10% of the total test set (300 images vs 3000 for the validation set). Hence, the validation set could be a better representation of our generalisation performance.

References

Nih chest x-ray dataset nbsp;|nbsp; cloud healthcare api nbsp;|nbsp; google cloud. URL https://cloud.google.com/healthcare/docs/resources/public-datasets/nih-chest.

Vinbigdata chest x-ray abnormalities detection. URL https://www.kaggle.com/c/vinbigdata-chest-xray-abnormalities-detection.

Cartucho. map. URL https://github.com/Cartucho/mAP.

Christiansen, Anders. Anchor boxes-the key to quality object detection, Jan 2021. URL https://towardsdatascience.com/anchor-boxes-the-key-to-quality-object-detection-ddf9d612d4f9.

Duan, Kaiwen, Bai, Song, Xie, Lingxi, Qi, Honggang, Huang, Qingming, and Tian, Qi. Centernet: Keypoint triplets for object detection, 2019.

Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., and Zisserman, A. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88 (2):303–338, June 2010.

Islam, Mohammad Tariqul, Aowal, Md Abdul, Minhaz, Ahmed Tahseen, and Ashraf, Khalid. Abnormality detection and localization in chest x-rays using deep convolutional neural networks, 2017.

Jiao, Licheng, Zhang, Fan, Liu, Fang, Yang, Shuyuan, Li, Lingling, Feng, Zhixi, and Qu, Rong. A survey of deep learning-based object detection. *IEEE Access*, 7:128837–128868, 2019. ISSN 2169-3536. doi: 10.1109/access.2019.2939201. URL http://dx.doi.org/10.1109/ACCESS.2019.2939201.

Kingma, Diederik P and Ba, Jimmy. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kong, Tao, Sun, Fuchun, Liu, Huaping, Jiang, Yuning, Li, Lei, and Shi, Jianbo. Foveabox: Beyound anchor-based object detection. *IEEE Transactions on Image Processing*, 29:7389–7398, 2020. ISSN 1941-0042. doi: 10.1109/tip.2020.3002345. URL http://dx.doi.org/10.1109/TIP.2020.3002345.

Law, Hei and Deng, Jia. Cornernet: Detecting objects as paired keypoints, 2019.

Li, Ethan Yanjia. 12 papers you should read to understand object detection in the deep learning era, Aug 2020. URL https://towardsdatascience.com/12-papers-you-should-read-to-understand-object-detection-in-the-deep-learning-era-3390d4a28891.

Newell, Alejandro and Deng, Jia. Pixels to graphs by associative embedding. *arXiv preprint arXiv:1706.07365*, 2017.

Rahmat, Taufik, Ismail, Azlan, and Aliman, Sharifah. Chest x-ray image classification using faster r-cnn. *MALAYSIAN JOURNAL OF COMPUTING*, 5(1), 2020. doi: 10.24191/mjoc.v5i1.

Rajaraman, Sivaramakrishnan, Kim, Incheol, and Antani, Sameer K. Detection and visualization of abnormality in chest radiographs using modality-specific convolutional neural network ensembles. *PeerJ*, 8:e8693, 2020.

Ren, Shaoqing, He, Kaiming, Girshick, Ross, and Sun, Jian. Faster r-cnn: Towards real-time object detection with region proposal networks, 2016.

Sakuraandblackcat. Chest_x – *ray* : *Knowledgesforthe*14*abnormalities*, *Jan*2021. *URL*.

Santosh, K. C., Dhar, M. K., Rajbhandari, R., and Neupane, A. Deep neural network for foreign object detection in chest x-rays. In *2020 IEEE 33rd International Symposium on Computer-Based Medical Systems (CBMS)*, pp. 538–541, 2020. 10.1109/CBMS49503.2020.00107.

Solawetz, Jacob. What are anchor boxes in object detection?, Sep 2020. URL https://blog.roboflow.com/what-is-an-anchor-box/.

Tedeschi, Bob. How machine learning could revolutionize medicine, Oct 2016. URL https://www.statnews.com/2016/10/03/machine-learning-medicine-health/.

Wang, Hongyu and Xia, Yong. Chestnet: A deep neural network for classification of thoracic diseases on chest radiography, 2018.

Wang, Rui. Object detection in x-ray images, Apr 2020. URL https://medium.com/sfu-cspmp/object-detection-in-x-ray-images-414a4fb06dff.

Zahaviguy. What are lung opacities?, Sep 2018. URL https://www.kaggle.com/zahaviguy/what-are-lung-opacities.

Zhao, Zhong-Qiu, Zheng, Peng, Xu, Shou-tao, and Wu, Xindong. Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, 30(11):3212–3232, 2019.