Machine Learning Engineer Nanodegree

Capstone Proposal

Zhenxiong Yang
December 12, 2018

Proposal

I. Definition

Project Overview

Pneumonia is a big killer in Healthcare area for human-beings. Pneumonia ranks third in twenty leading principal hospital discharge groups in United States in 2015, only next to heart disease and chest pain[1]. There are about 544,000 visits in hospital taking up 4.4% distribution[1]. And it causes more than 50,000 deaths which nearly half of them are 85 years and over[2]. X-ray is the best way to diagnose pneumonia[3]. However, usually diagnosing pneumonia precisely needs experienced doctors checking carefully. So it is vital to confirm the pneumonia accurately and quickly through X-ray photographs. This can help doctors to make effective analysis and decisions, and also save patients' waiting time in hospital. This a Kaggle project.

Problem Statement

Pneumonia detection can be divided into two parts. The first is to classify whether it shows pneumonia or non-pneumonia. This is binary classification problem. The second is to localize which regions of pictures show pneumonia if it is pneumonia. Every region is presented in rectangle described by a left-top coordinate and width and height. There may be more than one region in a picture. This is the object detection problem. I will explore an algorithm to classify the images and predict the regions to approach the ground truth as possible.

Metrics

The Evaluation Metrics is according to Kaggle Evaludation. The score is evaluated on the mean average precision at different intersection over union (IoU) thresholds. The IoU of a set of predicted bounding boxes and ground truth bounding boxes is calculated as:

$$IoU(A, B) = \frac{A \bigcap B}{A \bigcup B}$$

The metric sweeps over a range of IoU thresholds, at each point calculating an average precision value. The threshold values range from 0.4 to 0.75 with a step size of 0.05: (0.4, 0.45, 0.5, 0.55, 0.6, 0.65, 0.7, 0.75). At each threshold value t, a precision value is calculated based on the number of true positives (TP), false negatives (FN), and false positives (FP) resulting from comparing the predicted object to all ground truth objects: $\frac{TP(t)}{TP(t)+FP(t)+FN(t)}$. The average precision of a single image is calculated as the mean of the above precision values at each IoU threshold. In nearly all cases confidence will have no impact on scoring.

II. Analysis

Data Exploration

The data is from Kaggle Data and consists of several files. The detailed_class_info.csv maps patientId to result class.The train images and test images files show patients' X-ray original photographs and also attach the personal information, like patientId, sex, age and so on. I will use train pictures as X_data, and the train_labels as y_label according to corresponding 'patientId'.

There are three classes here, "No Lung Opacity/Not Normal", "Normal", "Lung Opacity". Both "Normal" and "No Lung Opacity/Not Normal" images are non-pneumonia targeted as 0, and "Lung Opacity" images are pneumonia targeted as 1.

The train_labels.csv illustrate patients' results, including 'patientId', 'x-min', 'y-min', 'width', 'height', 'target'. The 'x-min' and 'y-min' is the coordinate of the region. 'width' and 'height' describe the region size. The 'target' is 1 if it is pneumonia and 0 if not. They are consistent with the classes in detailed_class_info.csv. We can see some labels data in Table 1.

Table 1: Label Example

	patientId	x	y	width	height	Target
0	0004cfab-14fd-4e49-80ba-63a80b6bddd6	NaN	NaN	NaN	NaN	0
1	00313ee0-9eaa-42f4-b0ab-c148ed3241cd	NaN	NaN	NaN	NaN	0
2	00322d4d-1c29-4943-afc9-b6754be640eb	NaN	NaN	NaN	NaN	0
3	003d8fa0-6bf1-40ed-b54c-ac657f8495c5	NaN	NaN	NaN	NaN	0
4	00436515-870c-4b36-a041-de91049b9ab4	264.0	152.0	213. 0	379.0	1

We can also know there are 9555 pneumonia boxes in 6012 pneumonia images in total 26684 images.

Exploratory Visualization

I plot an pneumonia image in Figure1. It shows clearly there are two pneumonia regions.
First I count the pneumonia images and pneumonia images. About 1/3 are pneumonia.
We can see the details in a bar graph. I also do the statistics about the numbers of boxes in a pneumonia image. Figure 3 shows most pneumonia images have one box, and then two boxes. Few images have more than two pneumonia regions. No images have over four regions.

Algorithms and Techniques

As is discussed above, this problem contains two sections, classification and detection. There are several excellent models to solve this kinds of problems, like Faster-RCNN[4] or Mask-RCNN[5], Yolo[6] and SSD[7]. Fast-RCNN processes

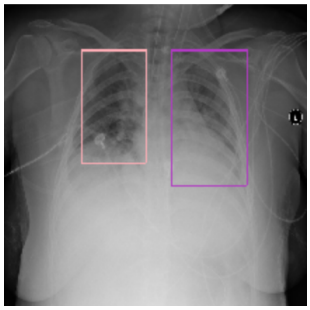


Figure 1: Pneumonia

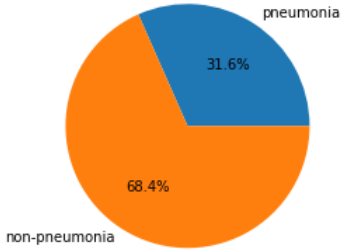


Figure 2: Percentage of pneummonia

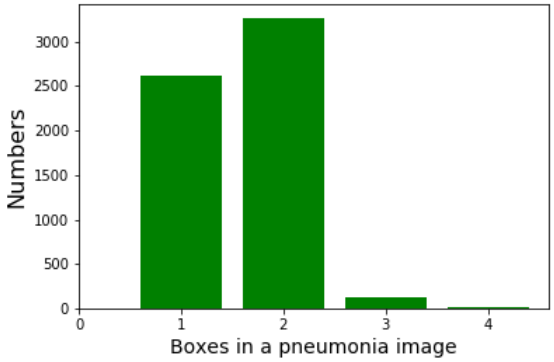


Figure 3: Numbers of boxes

accurately but slow. SSD has better accuracy and speed. Yolo is faster but less accurate. But Yolov3[8] has improved a lot and surpasses in both mAP-50 and time aspect. In my training SSD is slow convergence or even no convergence in this project. Maybe my parameters are not good. So I choose Yolov3. Yolov3 divides the image into n by n cells. If the center of an object is in a cell, then this cell is responsible for the object. The model creates several anchors based on this cell. These anchors has several sizes and ratios. The number of anchors is the number of size by the number of ratios. For each anchor, its dimension in width and height is specified by p_w, p_h. The cell responsible for detection position is specified by the top-left coordinate c_x, c_y. After some convolutional layers, the model outputs five parameters $\hat{t}_x$, $\hat{t}_y$, $\hat{t}_w$, $\hat{t}_h$ and $\hat{t}_o$. The last $\hat{t}_o$ specified the confidence that is the object probability by IOU between prediction box and truth bounding box. $\hat{t}_x$, $\hat{t}_y$ are relative to the cell top-left coordinate. $\hat{t}_w$, $\hat{t}_h$ are relative to the anchor size. Then it predicts a region also named proposal which is $\hat{b}_x$, $\hat{b}_y$, $\hat{b}_w$, $\hat{b}_h$. The relationship is as follows:

$$\begin{cases} \hat{b}_x = \sigma(\hat{t}_x) + c_x \\ \hat{b}_y = \sigma(\hat{t}_y) + c_y \\ \hat{b}_w = p_w e^{\hat{t}_w} \\ \hat{b}_h = p_h e^{\hat{t}_h} \end{cases}$$

σ is the sigmoid function: $\sigma(x) = 1/(1+e^{-x})$. It is unstable to calculate the loss directly between prediction bounding box($\hat{b}_x$, $\hat{b}_y$, $\hat{b}_w$, $\hat{b}_h$) and ground-truth bounding box(b_x, b_y, b_w, b_h). Therefore this model makes a inverse transformation for the ground-truth values to relative values as before to get t_x, t_y, t_w, t_h. The loss roughly contains two parts. One is the classification loss and the other is detection region loss like Faster-RCNN[**?**]

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

If one object is positive, the probability p is 1, otherwise it is 0. The t is the transformed ground-truth coordinate t_x, t_y, t_w, t_h. The t^* is the prediction coordinate $\hat{t}_x$, $\hat{t}_y$, $\hat{t}_w$, $\hat{t}_h$. When the object is negative which is background, we don't calculate the region loss but only the classification loss. The detailed formula is complicated and can refer source code. There are some parameters can be tuned to optimize: learning rate, batch, subdivisions, and so on.

Benchmark

As a project from Kaggle, the top score in Private Leaderboard is considered as the benchmark model. The private leaderboard is calculated with approximately 99% of the test data. I will strain every nerve to approach or even exceed it.

III. Methodology

DataPreprocessing

The coordinate format in a original data is 'xmin', 'ymin', 'width', 'height'. But Yolov3 requires relative center coordinate and relative width and height by

image_size. So I need to preprocess the input data to relative center format and postprocess the predict data back to 'xmin', 'ymin'. We have know the pneumonia and non-pneumonia proportion. I choose the same number of negative images to improve the precision. For the label data, I create every label .txt file for every image and they have the same name which is patientId. If the target is 1, the label text is target, relative center x, relative center y, relative width, relative height. If there are multiple boxes in a images, there will be multiple lines corresponding to it. If the target is 0, the label text is empty. The image format is 'dcm', which Yolov3 doesn't support. So I convert it to 'jpg' format. During this process, I also add a dimension to RGB three dimensions.

Implementation

To implement Yolov3 to this project, I need to make some files and do some adaptive changes. First, I train model on GPU 1080Ti, so I change GPU=1, CUDNN=1, OPENCV=1 in Makefile and then make in command.
Second, I split the images to train part and validation part in 9: 1 proportion. As I have created label txt file in preprocessing, here I create train.txt and val.txt files where the contents are the address of the corresponding images.
Third, I create rsna.cfg, rsna.names, rsna.data files. The rsna.cfg is copied from yolov3.cfg. I need to do change some parameters. In this project, there is only one object which is pneumonia. The other images can be considered backgrounds and we ignore them. So I edit classes=1 and filters=18 at the some time in yolo layer. The number of filters is $(classes + 1) \times 3$. To improve the accuracy, I change the width and height to 1024 that is the image size. The rsna.names file contains the class names. There is only one class here. So I edit pneumonia. The rsna.data file contains class numbers, train file path, valid file path, names file path and backup directory which is used to save the weights in training. Now the preparation for yolov3 is finished.

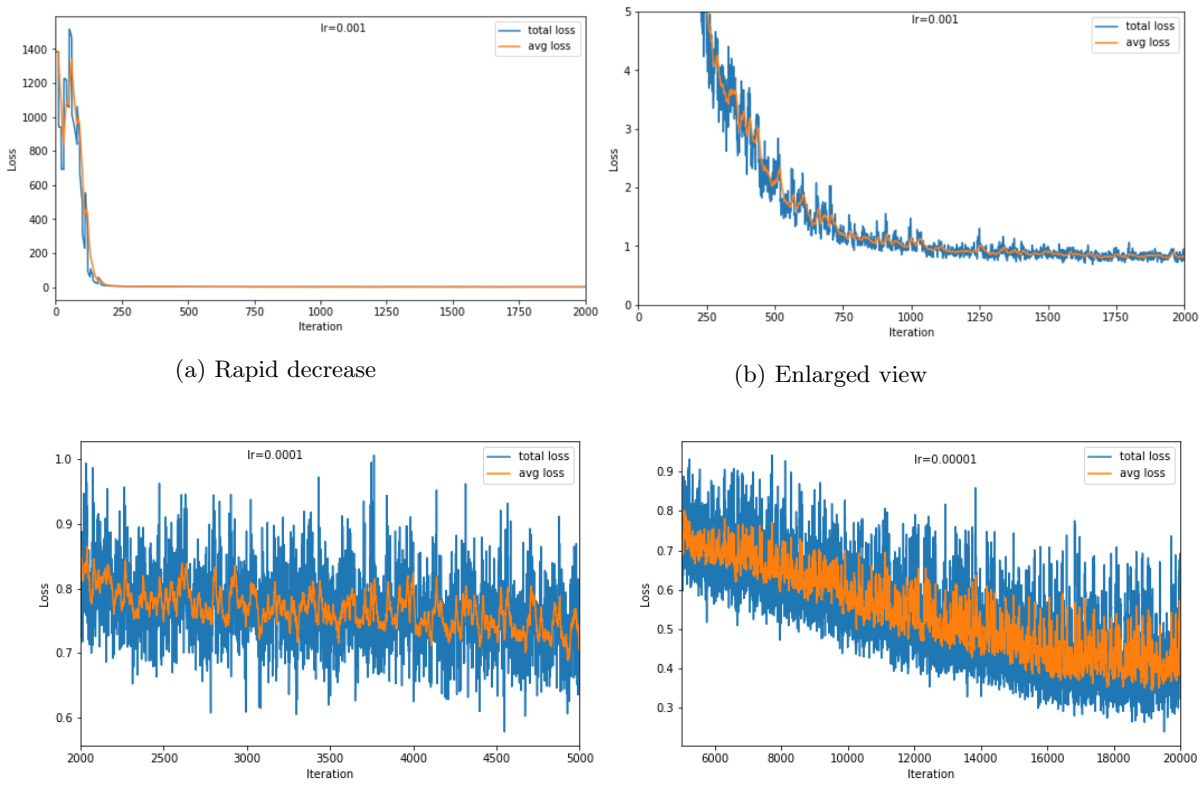
Refinement

At beginning, I have tried SSD and found the loss fluctuated. So I attempt to yolov3. At first I set learning rate to 0.001. The loss decreased fast. At about 2000 iterations, it decreased very slow and saturated to about 0.6. So I decrease the learning rate to 0.0001. The loss decreases to about 0.3 at 20000 iterations and didn't decrease any more. To get a better the model, I have made such improvement. First the positive and negative ratio is 1:3 in the data. Too many negative images are not good for the model to learn. I add the negative images as many as positive images in training.

IV. Results

Model Evaluation and Validation

For image detection, Faster-RCNN, SSD and Yolov3 are good candidate models. But Faster-RCNN trains too slow. For SSD, I have tried learning rate from 0.02, 0.01, 0.001, 0.0001 to 0.00001 for this project. The losses all fluctuates from 500 to 1200. Beside adam, I have also tried sgd, rmsProp optimizer. There is no



(a) Rapid decrease

(b) Enlarged view

(c) Stabilization

(d) Saturation

Figure 4: Train loss

trend to decrease. Maybe the parameters are not appropriate. Therefore I convert to Yolov3. The loss decreases rapidly in two hundreds of iterations and gradually slows down as in Figure 4a-4b.

And after 2000 iterations, I decrease the learning rate from 0.001 to 0.0001 in three thousand iterations and find the loss is nearly stable and decrease a little from roughly 0.8 to 0.7. And then I continue to decrease the learning rate to 0.00001 from 5000 iterations to 20000 iterations. After 20000 iterations, the loss stops to decease. We can see this in Figure 4c-4d.

In training, we can see the loss decreasing with the number of iterations increasing. After training, we need to validate the model. We use the score in metric and the AP(Average Precision) to check when the model starts to overfit and then choose the best weight. Score is $TP/(TP+FP+FN)$ and average precision is the expectation precision $TP/(TP+TF)$ on 11 recall values from 0 to 1 by every 0.1 interval. Usually, the smaller the recall, the higher the precision.

From Figure 5, we can see that the score and AP have the same trend in general. When the iteration is 18000, the score is highest up to 0.2 and AP is also very high around to 70%. After 18000 iterations, the model start to overfit.

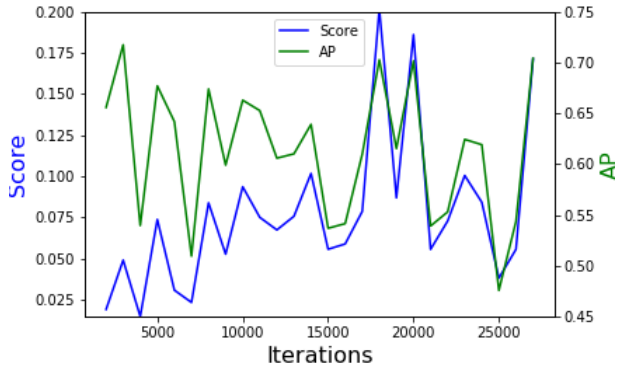


Figure 5: Score and AP

Justification

So I choose the weight in 18000 iterations to do the predictions and get the score. My score is 0.05347. I choose the Kaggle top one score in private leaderboard as the benchmark. The best score is 0.25475. There are still some gaps. This weight in validation images is the best one. The detailed parameters are: $AP = 70.24\%$, $TP = 201$, $FP = 68$, $FN = 728$.
There are maybe some parameters not good enough or limitation for this model. For positive prediction I think the final solution can be a reference for a doctor. But it has a lot of FN, which means it misses many pneumonia patients.

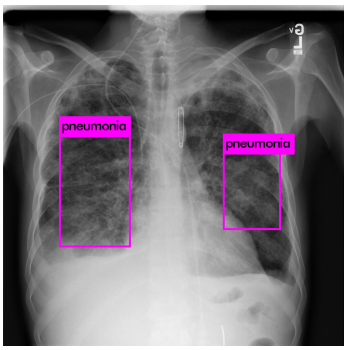
V. Conclusion

Free-Form Visualization

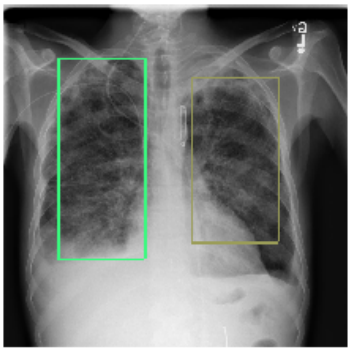
And let's see some right predictions in Figure 6. These are validation images. Left part are predictions and the right part are the truth. The prediction region is close to the truth boxes even though there are some deviations. In most cases, the prediction regions is smaller than the truth boxes.
We can see the wrong predictions either predicting wrong region or missing the region in Figure 7. For prediction3, the model makes mistakes in the left region. It is a wire not pneumonia. And the right region is not accuracy which is too small. For prediction4, the right prediction is good but the left part is missed.

Reflection

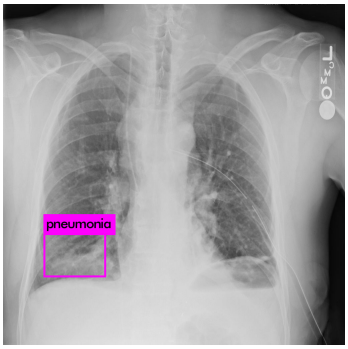
For the project, we can know it is image detection problem. We need to both classify the image and predict its regions. Here there is only one class and most regions are two or only one and no more than four. The dataset in labels is x, y, w, h format and we preprocess it to relative rx, ry, rw, rh format in order to calculate the loss for convenience.
At first I think this project is fascinating that a model could predict or judge the the pnemonia just by historical images that trend to do the doctors' job. However, when I devoted myself to this work, I find it is not easy to rival doctors.



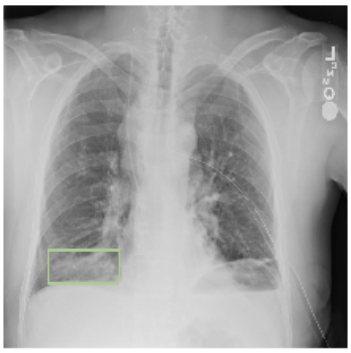
(a) Prediction1



(b) Truth1

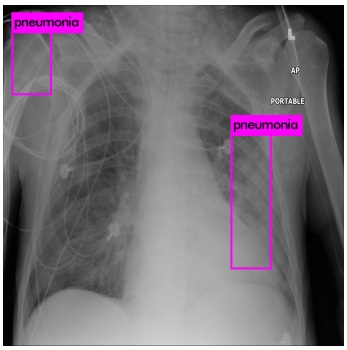


(c) Prediction2

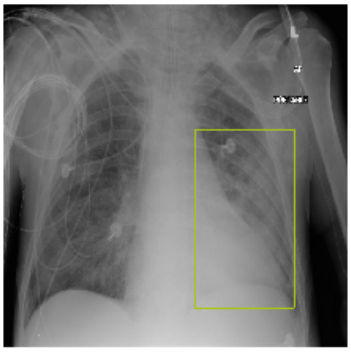


(d) Truth2

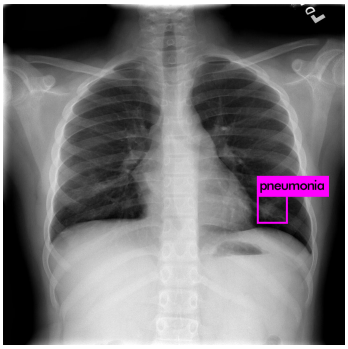
Figure 6: Right Predictions



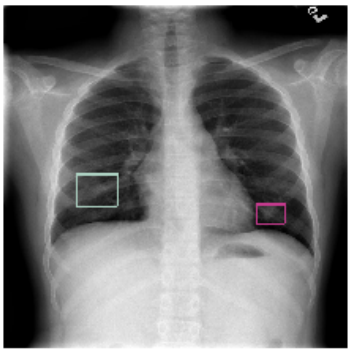
(a) Prediction3



(b) Truth3



(c) Prediction4



(d) Truth4

Figure 7: Wrong Predictions

The result fits my expectation just a little. There are too many FP and the predict region is not accuracy enough. And that's the difficulty here, on both accuracy and comprehensive aspect. The mAP is 70% in validation which is also AP because of only one class. Maybe there are some parameters needing to improve. I am new in this field but I am willing to be engaged in it.

Improvement

For further improvements, I may try to SSD to fine-tune with better parameters or Mask-RCNN. A better solution must exist. But we don't know yet.

References

[1] Rui P, Kang K, *National Ambulatory Medical Care Survey: 2015 Emergency Department Summary Tables*. Table 27. www.cdc.gov/nchs/data/nhamcs/web_tables/2015_ed_web_tables.pdf

[2] *Deaths: Final Data for 2015. Supplemental Tables*. Tables I-21, I-22. www.cdc.gov/nchs/data/nvsr/nvsr66/nvsr66_06_tables.pdf

[3] *How Is Pneumonia Diagnosed?*. NHLBI. 1 March 2011. https://www.nhlbi.nih.gov/health-topics/pneumonia

[4] Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun, *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*. https://arXiv:1506.01497v3

[5] Kaiming He Georgia Gkioxari Piotr Dollar Ross Girshick, *Mask R-CNN*. https://1703.06870v3

[6] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi, *You Only Look Once: Unified, Real-Time Object Detection*. https://arXiv:1506.02640v5

[7] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, Alexander C. Berg, *SSD: Single Shot MultiBox Detector*. https://arXiv:1512.02325v5

[8] Joseph Redmon, Ali Farhadi, *YOLOv3: An Incremental Improvement*. https://arXiv:1804.02767v1