Pneumonia Detection – RSNA Dataset 2020

OCTOBER 4

Capstone Project Group 9 Oct (Computer Vision)

Authored by:

Utkarsh Jain, Ananth Manoj , Tej, Jagnya

Pneumonia Detection - RSNA Dataset - Abstract

Using computer Vision / Deep Learning

Pneumonia is a deadly lung disease known as silent killer and it is one among top-10 causes of death in some countries. The most common diagnostic tool for pneumonia is Chest X-rays. However, due to several other medical conditions in the lungs, diagnosis of pneumonia using chest X-rays is very complicated and need trained specialist intervention for confirming the case. Therefore, there is a direct need for computer-aided diagnosis systems to assist clinicians in making better decisions. This work proposes some approaches to reach the solution. We experimented with basic convolutional neural network, ResNet and Mask-RCNN. And Mask-RCNN turned out to perform better among them.

CHEST X-RAY BASICS AND WHAT OPACITY MEANS?

In the process of taking the image, an X-ray passes through the body and reaches a detector on the other side. Tissues with sparse material, such as lungs which are full of air, do not absorb the X-rays and appear black in the image. Dense tissues such as bones absorb the X-rays and appear white in the image. In short -

- Black = Air
- White = Bone
- Grey = Tissue or Fluid

The left side of the subject is on the right side of the screen by convention. we can also see the small L at the top of the right corner. In a normal image we see the lungs as black, but they have different projections on them - mainly the rib cage bones, main airways, blood vessels and the heart.

Any area in the chest radiograph that is more white than it should be can term as Opacity

Usually the lungs are full of air. When someone has pneumonia, the air in the lungs is replaced by other material - fluids, bacteria, immune system cells, etc. That's why areas of opacities are areas that are grey but should be more black. When we see them we understand that the lung tissue in that area is probably not healthy.

So in short,

- Balck Area in lungs --> Healthy
- white / Grey in the lungs --> Unhealthy or Have Opacity

PROBLEM FACED

There are some problems that are faced in hospitals. Some of them are jotted bellow:

- Identifying the pneumonia needs specialized and trained doctors, as to find the minute areas in X-Rays along with other symptoms. And examining the X-Rays would take time for any such doctors.
- Lack of Specialized doctors in remote places of the world makes it difficult to identify such diseases.

So, how can we overcome this problem will be concern, which drives to our project identification of Pneumonia.

OBJECTIVE

Our objective is to build a model which takes a CXR image as input and attempts to find potentially infected-area and predicts the position of box/boxes bounding the infected area. Automating Pneumonia screening in chest radiographs, providing affected area details through bounding box.Assist physicians to make better clinical decisions or even replace human judgment in certain functional areas of healthcare (eg, radiology). Guided by relevant clinical questions, powerful AI techniques can unlock clinically relevant information hidden in the massive amount of data, which in turn can assist clinical decision making.

ABOUT DATA SET:

To improve the efficiency and reach of diagnostic services, the Radiological Society of North America (RSNA®) has collaborated with the US National Institutes of Health, The Society of Thoracic Radiology, MD.ai to develop a rich dataset with thousands of DICOM images or Chest X-Rays(CXR) and marked CXR with Boxing the pneumonia area, which can help us in train and test our Models.

EXPLORING THE DATA:

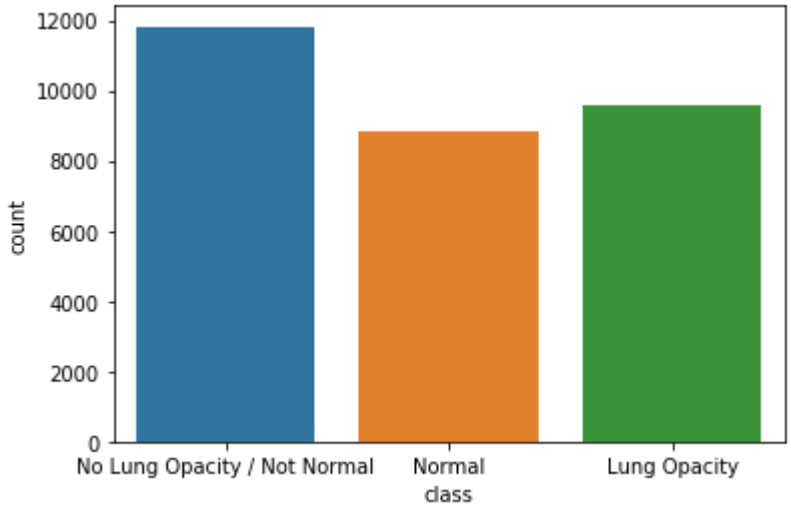
Data provided have following:

- stage_2_train_images → folder with CXR DICOM images with marking Pneumonia
- stage_2_test_images → folder with CXR DICOM images without marking Pneumonia
- stage_2_detailed_class_info.csv contains information about the positive and negative classes in the training set
- stage_2_train_labels.csv contains information about pneumonia boxes along for train dataset images

The training data is provided as a set of patientIds and bounding boxes. Bounding boxes are defined as follows: `x-min y-min width height`
There is also a binary target column, `Target`, indicating pneumonia or non-pneumonia.
There may be multiple rows per `patientId`

EDA WITH PYTHON:

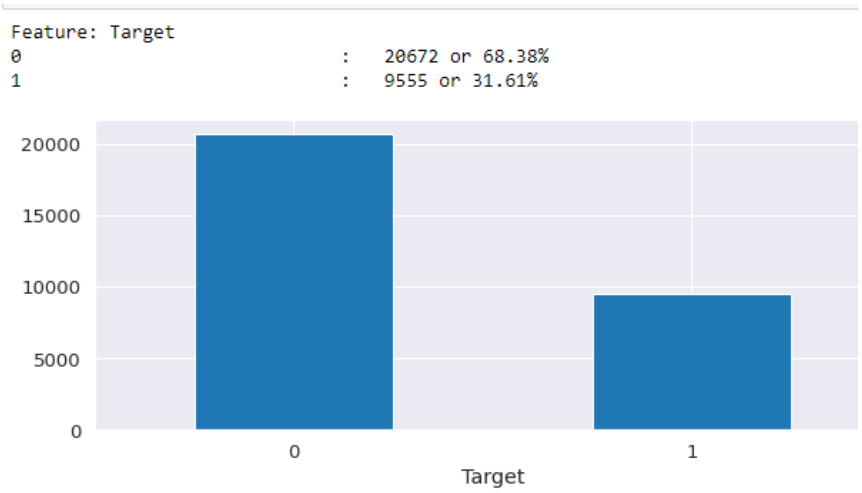
There are three classes of Patients: No Lung Opacity / Not Normal, Normal and Lung Opacity, dataset is dominated with class 'No Lung Opacity / Not Normal'.



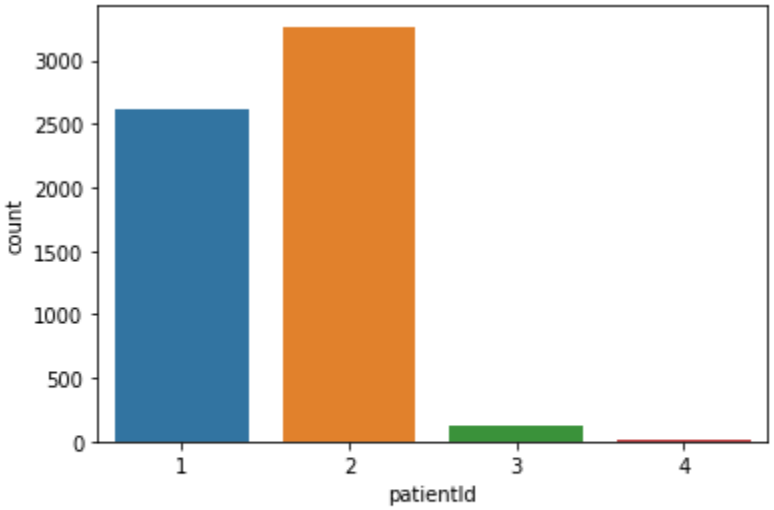
68.38% or 20672 of data labeled as No Pneumonia (Target / label as 0)

31.61% or 9555 of data labeled as Pneumonia (Target / label as 0)

Data is set is highly non uniform.

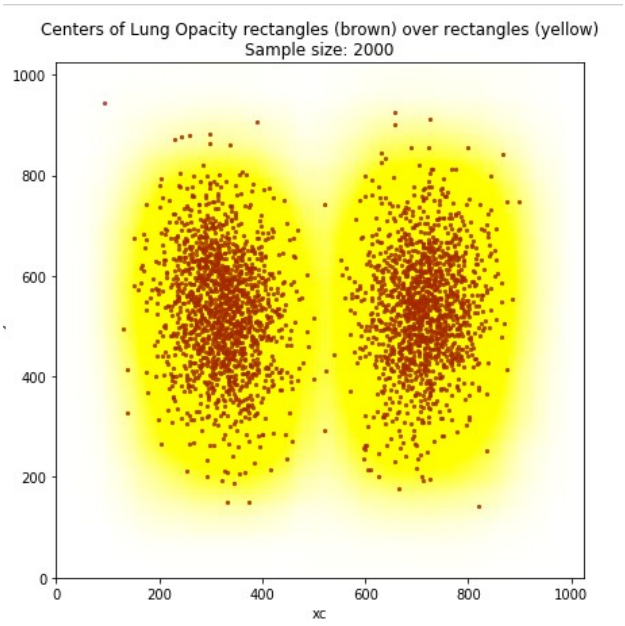


We observed that there are multiple records for each patient and total data is of 26684 Unique patients and number of infected patients 6012 while number of records in class-1 is 9555, the difference is because a patient may have more than 1 infected area and thus having more than 1 record per patient. From the below graph we can see there are few patients with more than one infected area.



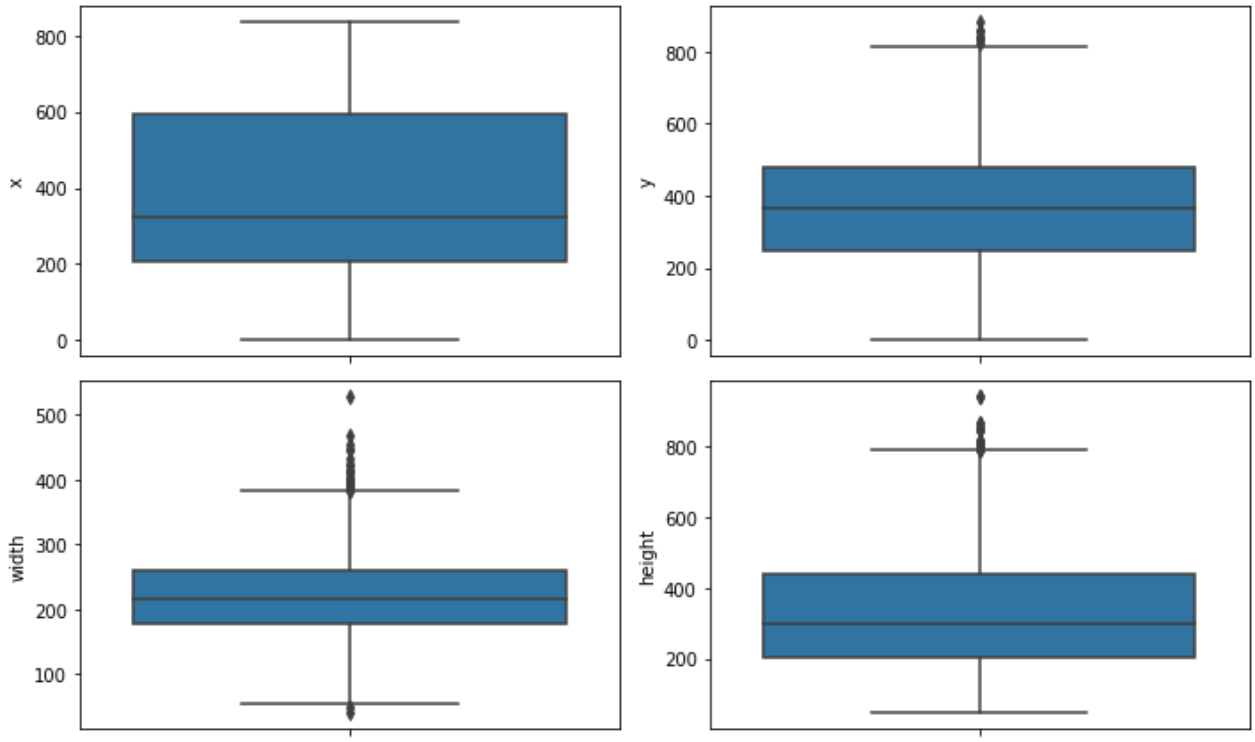
- Almost all people have 1 or 2 infected areas
- Very few people with 3 or 4 infected areas

Let's visualize center of lung Opacity (Brown) over the rectangular infected areas (Yellow)

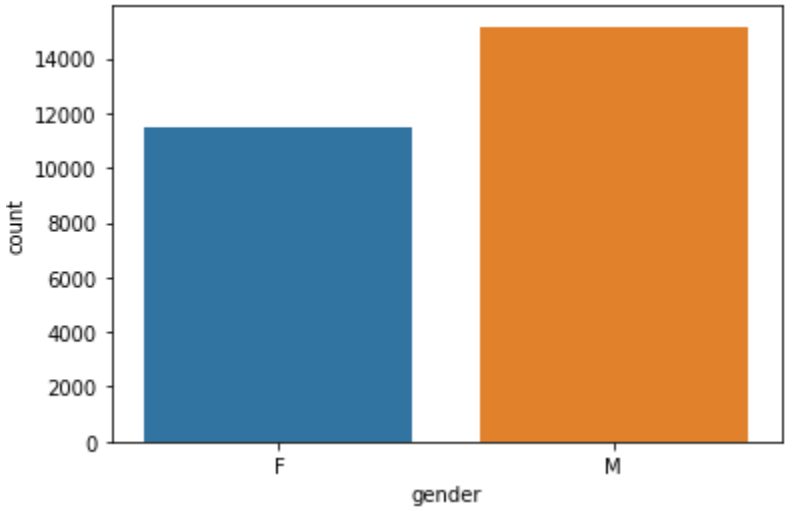


Above shows that for most of the cases center of infected region is Middle of the lungs.

There are no Null values in the dataset given that's really a good news ☺ and we don't need to make any assumptions in that perspective. But there are some outliers for width and height.



Below image shows Gender distribution of data and looks like more Male centered data.



STEP-BY-STEP WALK THROUGH THE SOLUTION:

We created three models and experimented each against multiple factors:

1. Base CNN model from scratch with ___ layers
2. Residual Neural Network (ResNet) CNN from Scratch with depth of 4

3 Mask RCNN Model

Base CNN model:

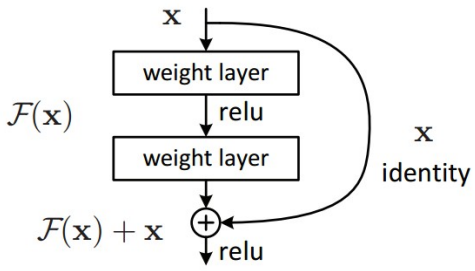
The Basic CNN model which was built from scratch includes combination of 2D convolution layers, Maxpooling layers and Conv2dTranspose layers, having final depth of 8 such combinations. The model uses relu as activation function, having 3x3 srtides , takes 256x256 sized image and uses Adam optimiser while compiling. The generator has been used to get image from dicom file , its respective position and size of the box (x,y,width, height) which then is created as a mask. This is done to get train set and validate set. Using to compile and

- At first we ran the model for 15 epochs keeping decent learning rate of 0.001 and with depth of 3 we observed how well would our model learn from the input by keeping an eye on loss and mean_iou. This helped us to set bench mark.
- Seeing the accuracy we increased the depth of the model to 8, introducing dropouts in middle of the layers and increased the epochs to 20, the model did not respond quite well.
- Keeping in mind the response to previous execution, we further increased the epochs to 30 , removing the dropouts and advancing the learning rate to 0.00001. This significantly increased the mean IOU to 60% and the prediction was quite stunning.
- Further fine tuning the model to 50 epochs and using cosine annealing along with learning rate increased the mean IOU and dice coefficient
- Once more trying to fine tune the model did not improve the model as expected.

Thus we agreed to switch our model to Resnet and check out the efficiency of that model by fine tuning it.

Residual Neural Network (ResNet) CNN:

There are many variants of ResNet architecture i.e. same concept but with a different number of layers and we use ResNet50. The main idea of ResNets is to add the activation of layer l to the output of layer l+2 before applying non-linear activation. Thus, the activation of layer l can follow a shortcut to later layers in addition to going through layer l+1. This would allow us to train much deeper neural networks without suffering from gradient vanishing and exploding problems. This method of bypassing the data from one layer to another is called as shortcut connections or skip connections. This approach allows the data to flow easily between the layers without hampering the learning ability of the deep learning model. The advantage of adding this type of skip connection is that if any layer hurts the performance of the model, it will be skipped.



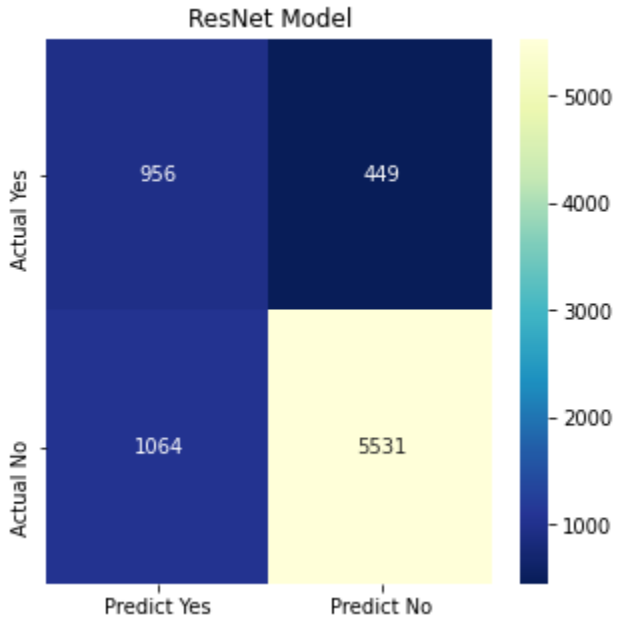
Our ResNet model consists of a

- Down sampling block (consisting of Batch normalization, leaky ReLU, Convolutional 2D and max pool layer).

- ResNet block (consisting of batch normalization, leaky ReLU and convolutional 2D layers).
- The downsampling and resnet blocks are repeated based on configuration parameters of n_blocks and depth.

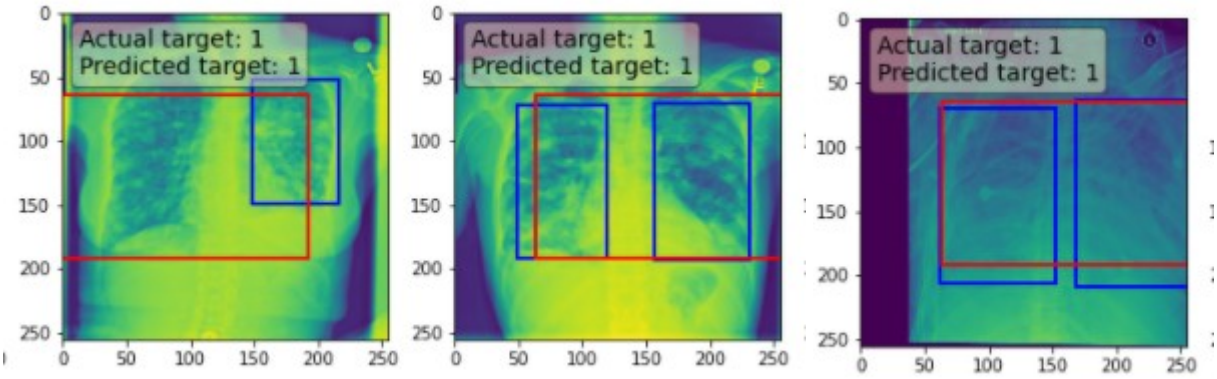
In the end we upsample the data points to the same size as before in order to get the target mask. Given data is spilt into 80:20 for training and validation purpose. We have tried with 70:30 ratio of training and validation as well but 80:20 yielded good results. Input for the model is training CXR images Pixel Array with bounding box(es) coordinates (if present) and output for the model is predicted Bounding boxes coordinates for the validation data set.

With configurations Depth = 4 ; Batch_size=32 ; Image Dimensions reduced to 512; Optimizer Adam ResNet Model mean IOU accuracy of is ***72.81%.*** And below is confusion matrix.



- Validation Accuracy: 0.81
- Validation precision: 0.47
- Validation Recall: 0.68

And some of predictions by ResNet model are show here (Blue box is ground truth and Red Box is Prediction) images are:



Mask-RCNN:

Next up, we decided to use a pre-designed Mask-RCNN model for this task:

Trainning approach:

There are several ways to train MaskRCNN -

- One is to train only the 'head' layers, which is whats called Tranfer Learni ng wherein we start with pre-trained weights and train only top layers (hi ger-level). We tried that and found that `loss` would start converging with out learning enough, this is perhaps because the pre-trained model is train ed on real-life, data-to-day object's images which are vastly different from the images we have here which are X-ray images.
- We can also include more layers to train.
- However, since these images belongs to a vastly different category, we de cided to not to use pre-trained model and train all the layers.

We require to create a model-configuration file, which defines various paramete rs and settings like, Learning-rate, epoch-size, etc.

Confiuration:

- **IMAGES_PER_GPU** should be as high as possible. We had access to atm ost 12 GB of GPU RAM which allowed us to set it to atmost 8.

- The role of Backbone is to extract features which coukd then furher be pas sed to CNN-layers. Two options are available here: '**resnet50**' and '**resnet101**', the latter one being a relatively deep network. We set `**BACKB ONE**` to '**resnet50**' to reduce memory load and for faster trainning.
- **DETECTION_NMS_THRESHOLD**:
  It may so happen that the model predicts overlapping bounding-boxes, re ndering some predictions to be useless. This is exactly what we noticed in some initial results. We changed it's value from 0.3 (default) to 0.01 after e poch-80.
- **TRAIN_ROIS_PER_IMAGE** - We set it to 128. It may be lowered down t o increase trainning speed with the expense of performance.
- **LEARNING_RATE**
- - It had been set to 0.001 throughout the trainning as there had been no pl ateauing of loss. It might need to be lowered down if trained further.
- Images have been rescaled to 512x512 dimensions to increase trainning sp eed

We trained it for a total of 260 epochs while keeling an eye on validation MAP to assure that the model is indeed learning and not overfitting.
Each epoch on google-colab took 242 seconds and that amounts to a total of 17 hours of training.

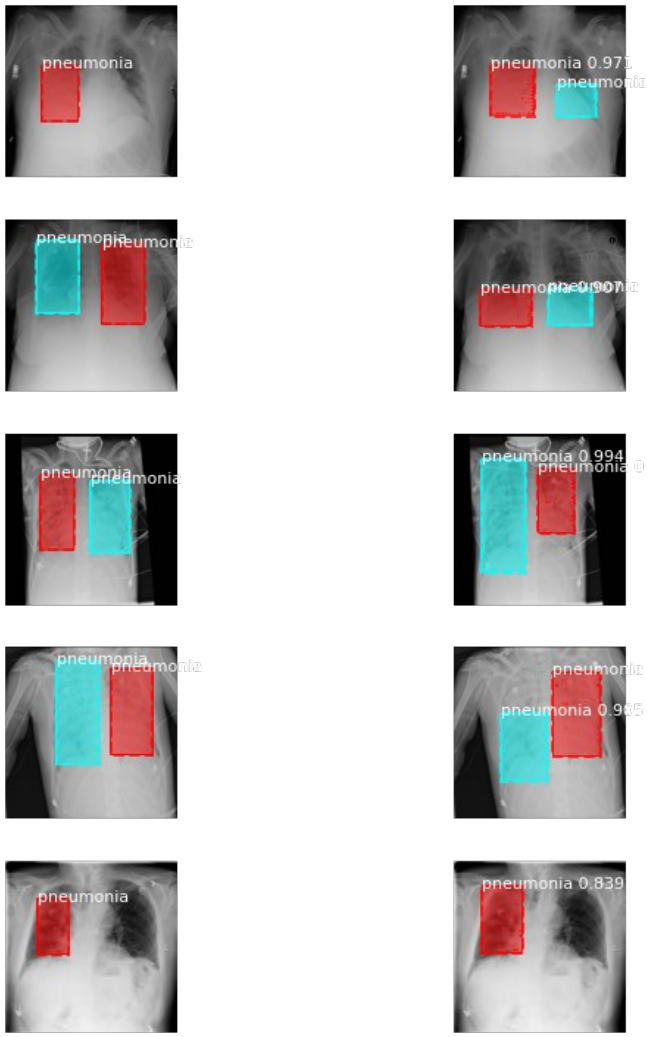
The evaluation metric used is MAP (Mean Average Precision), maximum can be 1 and minimum be 0 .
We have gotten an MAP of 0.12. To put that into perspective, the best model list ed on kaggle has gotten an MAP of 0.25.

Confusion Matrix:

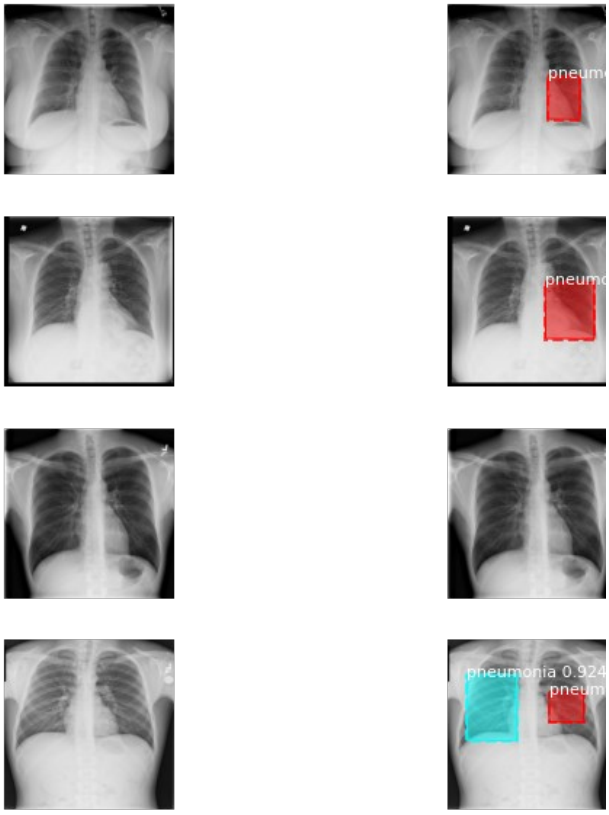
Classification accuracy:
validation precision: 0.66
validation recall: 0.95

Visualizations of predictions vs actual:

When image belongs to class 1



When image belongs to class 0



Limitations:
As can be seen the model performed fairly-well in predicting bbox in case if the patient is infected but performed badly in case of non-infected patients.
Putting in other way, there are many of false-positives.

To improve upon this:

- One may employ some data augmentation techniques and train for more epochs.
- Another way could be to train a classifier, which is presumably not a very hard problem, to first classify an image into class-1 or class-0 and then predict bounding box only if it belongs to class 1. This way many false positive would be eliminated at the classifier model.

MODEL EVALUATION:

The objective of the model is to predict the boxes (location of inflammation) as accurate as possible.

The prominent parameters that we have noticed here are,

- learning rate
- depth of the model
- dropouts
- number of epochs
- batch size

To evaluate the model, some good metrics are used such as:

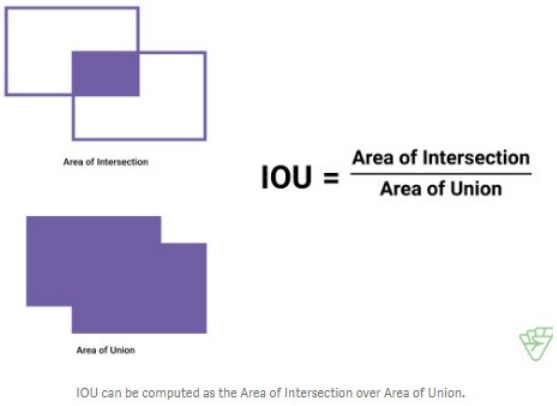
MAP:

The mean average precision (mAP) or sometimes simply just referred to as AP is a popular metric used to measure the performance of models doing document/information retrival and object detection tasks.

$$\text{MAP} = \frac{\sum_{q=1}^{Q} \text{AveP}(q)}{Q}$$

IOU:

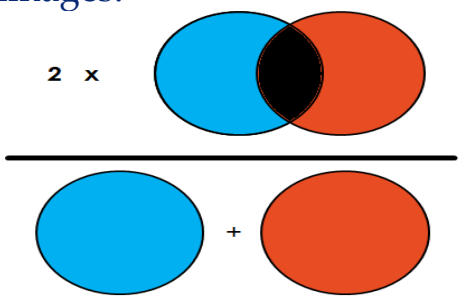
IOU is an evaluation metric used to measure the accuracy of an annotation on a particular task! It comes in handy when we are measuring how close an annotation or test output lines up with the ground truth. As a ratio of the areas of intersection and union, it works on annotations of all shapes and size.



IOU can be computed as the Area of Intersection over Area of Union.

Dice Coefficient:

The Dice Coefficient is 2 * the Area of Overlap divided by the total number of pixels in both images.



Comparision To Benchmark:

At initial stage keeping the model built from scratch with 15 epochs, learning rate as 0.001, layer depth as 3, we got the Mean iou as 57%. so keeping this in mind we set the bench mark 65-70% mean IOU.

We improved on the benchmark that was set to **68%** mean IOU by fine tuning the model. Making the model get trained to 50 epochs with learning rate of 0.00001 an cosine annealing and increasing the depth of the model to 8 layers. Increasing the layers of the model helped in recognising the features from the images. (to be continued....)

Comparison of Models.

As we have tried 3 different models of this project, lets check what are the major difference between them. And also it is best to compare these models on different aspects to select a model that gives good outcome.
Here are some of the aspects on which we have compared the model.

- **Structure and input:**

- All the models uses 256x 256 dimensional images except, Mask-RCNN which had been trained on 512x512 dimensional images.
- The base model only uses combination of 2d convolution layer along with maxpooling of size 2x2 and conv2d transpose layers with activation function of relu and sigmoid.
- Whereas, the resnet CNN we have used Batch normalization, leaky relu and upsampling 2D along with the previously used layers.
- And coming to Mask RCNN, we implemented it from from mrcnn package.

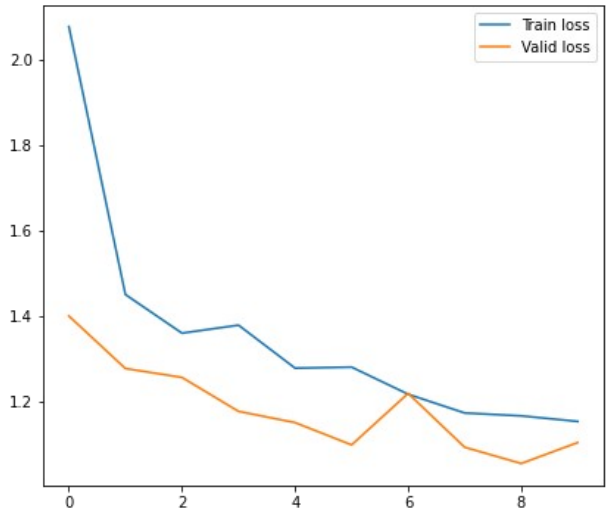
- **Training the model:**

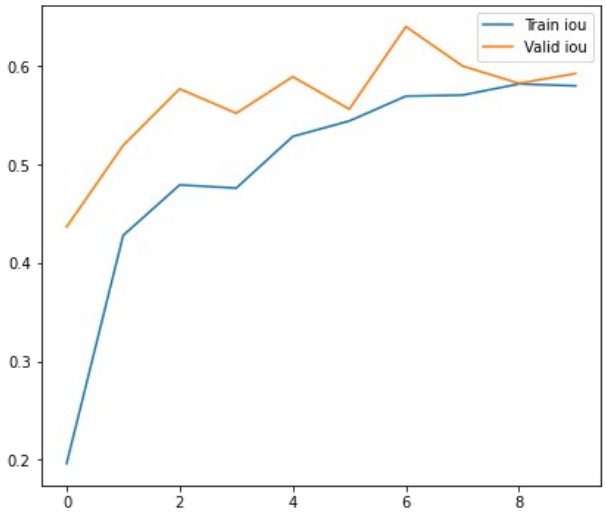
- We have trained the base model with learning rate of 0.00001 and optimizer as adam for 60 epochs.
- Whereas ResNet CNN uses cosine annealing and trained for 10 epochs.
- And in Mask RCNN, It had been set to 0.001 throughout the trainning as there had been no plateauing of loss. It might need to be lowered down if trained further.
- The time taken for the base model and Resnet to run is approchs 712 secs for each epochs.
- Each epoch on google-colab took 242 seconds and that amounts to a total of 17 hours of training.

- **Loss and accuracy matrics:**

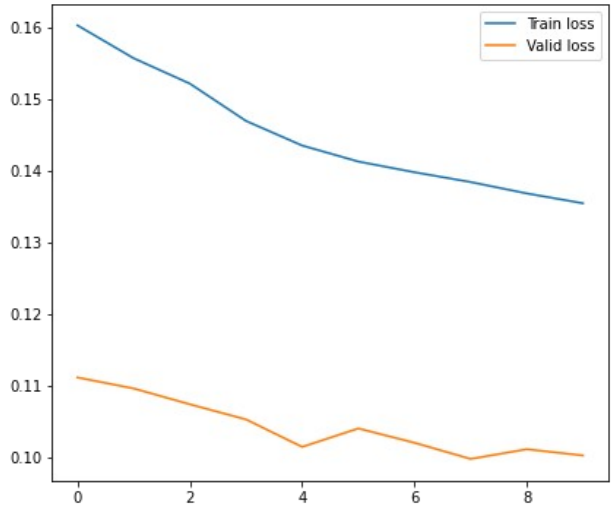
- After training the models for such a long time, we have got a mean IOU of 67% percent in Base model with loss % of 19.
- Whereas in ResNet we have gotten reletively good outcome of 72.81% Mean IOU, and loss of 10%
- Classification results:
  - validation-recall: 0.68
  - validation-precision: 0.47
- And in Mask RCNN the predictions are reletively better, having the Mean Average Precision of .12 (To put that in perspectiive the best model in kaggle got .25 MAP).
- Classification results:
  - validation-recall: 0.95
  - validation-precision: 0.84

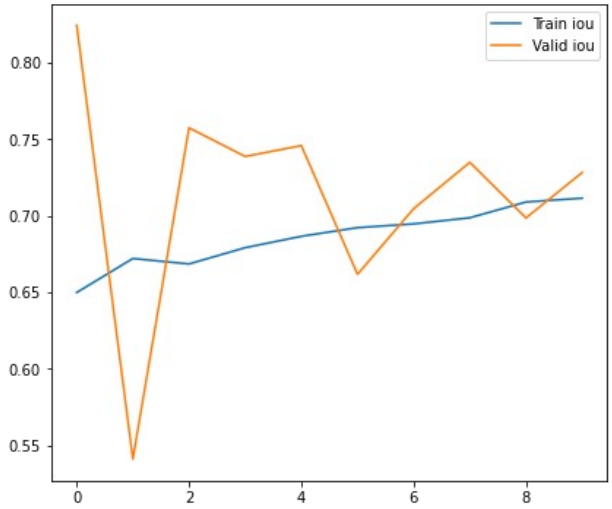
Variation of loss and mean_iou for training and validation data of Base model.





Below is the representation of variation of loss and mean_iou for training and validation data of ResNet RCNN.





In Brief :

	Base Model	ResNet CNN	Mask RCNN
Image dimension	256x256	256x256	512x512
Structure	Simple	Moderate	Complex
Depth	8	4	50 + more
Number of epochs	60	10	260
Time taken to train (per epoch)	712 sec	712 sec	242 sec
Mean IOU	67%	72.87%	-
MAP	-	-	.12

Validation precision	-	0.47	0.84
Validation Recall	-	0.68	0.94

Implications:

Given that humans have limited ability to detect patterns in individual images, accurate diagnosis can be a problem for even medical professionals. In order to minimize the number of errors and unintended consequences, computer programs based on neural networks and deep learning principles are increasingly used as assistant tools in medicine. The aim of this study was to develop a model of an intelligent system that receives x-ray image of the lungs as an input parameter and, based on the processed image, returns the possibility of pneumonia as an output. The implementation of this functionality was implemented through methodologies based on convolution neural network architectures.

Our models, as we aimed for predicts the location of the inflammation in the image if the image belongs to the diseased patient. By doing so providing affected area details through bounding box. Assist physicians to make better clinical decisions or even replace human judgement in certain functional areas of healthcare (eg, radiology). Guided by relevant clinical questions, powerful AI techniques can unlock clinically relevant information hidden in the massive amount of data, which in turn can assist clinical decision making.

Limitations

Some of the limitation that we noticed are as follows,

- As this project is of medical use, the model used here should be highly accurate and well coherent, which we lack a bit in our model.
- Going with the dataset, the data collected majorly were of non pneumonia patients compared to patient having that symtoms, which is / maybe one of the reason for the model to not getting trained well.
- We have some false positives and few false negatives while testing the prediction, which is major obstacle in real world life. This may result in creating a problem rather than solving one.
- The model is not accurate enough to detect exact location in some cases

To **enhance the solution,** experimentation on fine tuning the model's hyperparameter, which is a very time consuming process as we found, can greatly improve results. Improvement may be achieved by adding extra layers that concentrate more on extracting features from image, training model for longer. Training on higher-dimensional images may will probably get us better results with the expense of time taken and memory consumption. Training the model with complex model may also help.
Availability of more data is always beneficial.

Closing Reflections

There are quite a plenty of thing that we have learnt from this process. Some of them are jotted down below,

- When taken a project, the first aim is to understand the model properly rather than directly jumping to model building or EDA.

- Exploratory Data Analysis is also important even though we find it useless in the initial stages.
- There are more than one ways to solve the problem, no matter how complex the problem is.
- Machine Learning is not one man's show, sometimes it takes a good team effort to build the suitable model.

Coming to the model, we have learnt,

- How to convert data into required format (i.e here reducing the size of the image as required for the model and the idea of converting the position and size to mask)
- Analysing which type of models and layers ca be used to tackle the image dependent problems
- Tackling the overfitting problem by using dropouts and underfitting by using extra layers and learning rate.
- Also learnt different type of loss and matrics that can used while compiling the model such as dice coefficent, MAP, Mean IOU etc.

Next time when chance has been get, we want to explore more on the models that can be used. Concentrate on fine-tuning. Learning and working on some parameters which we couldn't guess how it may effect output and implementing it in a good order to get good results and much more...