

|  |
| --- |
| **Pneumonia Detection Challenge** |
| **Project Report** |

|  |
| --- |
|  |
|  |
|  |

**Submitted By: Group 13**

|  |  |
| --- | --- |
| PRESENTED TO | Great Learning |
| PRESENTED BY | Group 13 – Aug C – Great Learning |
| DATE | 24 July 2020 |
| VERSION | 1.0 |

## Group 13 – Contacts

|  |  |  |
| --- | --- | --- |
| Name | Phone | Email |
| Sonal Sharma | +91 88102 64764 | [Sonal\_1996@yahoo.com](mailto:Sonal_1996@yahoo.com) |
| Chandrashekar C | +91 98867 94586 | [chandrashekar.c@gmail.com](mailto:chandrashekar.c@gmail.com) |
| Susanta Mondal | +91 70440 55951 | Susanta.221285@gmail.com |
| Gopinath Bailur | +91 99400 78996 | [gbailur@gmail.com](mailto:gbailur@gmail.com) |

|  |  |  |  |
| --- | --- | --- | --- |
| CHANGE LOG | | | |
| DATE | VERSION | AUTHOR | CHANGE DESCRIPTION |
| 24th July 2020 | 1.0 | Group 13 | Initial Draft |
| 27th July 2020 | 1.1 | Group 13 | Updated based on the feedback with the mentor |

**Table of Contents**

[1 Overview 4](#_Toc46762525)

[2 Abstract 4](#_Toc46762531)

[3 Project Objective 5](#_Toc46762532)

[4 EDA Inference and Data pre-processing 6](#_Toc46762533)

[5 Current Approach on Model building 16](#_Toc46762534)

[6 Next steps and improvements. 20](#_Toc46762535)

# Overview

# Pneumonia is a form of acute respiratory infection that affects the lungs. The lungs are made up of small sacs called alveoli, which fill with air when a healthy person breathes. When an individual has pneumonia, the alveoli are filled with pus and fluid, which makes breathing painful and limits oxygen intake.

# Following are some of the key facts about Pnuemonia which needs at most attention to address the problem proactively.

* Pneumonia accounts for 15% of all deaths of children under 5 years old, killing 808 694 children in 2017.
* Pneumonia can be caused by viruses, bacteria, or fungi.
* Pneumonia can be prevented by immunization, adequate nutrition, and by addressing environmental factors.
* Pneumonia caused by bacteria can be treated with antibiotics, but only one third of children with pneumonia receive the antibiotics they need.

# Per WHO, Pneumonia is the single largest infectious cause of death in children worldwide. Pneumonia killed 808 694 children under the age of 5 in 2017, accounting for 15% of all deaths of children under five years old. Pneumonia affects children and families everywhere, but is most prevalent in South Asia and sub-Saharan Africa. Children can be protected from pneumonia, it can be prevented with simple interventions, and treated with low-cost, low-tech medication and care.

# The WHO and UNICEF integrated Global action plan for pneumonia and diarrhoea (GAPPD) aims to accelerate pneumonia control with a combination of interventions to protect, prevent, and treat pneumonia in children with actions to:

* protect children from pneumonia including promoting exclusive breastfeeding and adequate complementary feeding.
* prevent pneumonia with vaccinations, hand washing with soap, reducing household air pollution, HIV prevention and cotrimoxazole prophylaxis for HIV-infected and exposed children.
* treat pneumonia focusing on making sure that every sick child has access to the right kind of care -- either from a community-based health worker, or in a health facility if the disease is severe -- and can get the antibiotics and oxygen they need to get well;

# Based on research we have made an attempt to look at the Chest Radiography to identify the Lung Opacity and quickly help Clinical specialists to take right decisions to drive proactive measure to cure and help avoid the spread of this decease to larger extent.

# **Abstract**

This project is aimed at detecting Pneumonia by locating the lung opacities on the Chest radiographs. This process can help identify the problem at an early stage as well as helps the Clinical analysis much faster and drives better decision making. This process of Pneumonia detection will be done by looking at several thousand images of Chest radiographs taken from past wherein the analysis and desired results were identified by the specialists. These past datapoints will become the indicators and these images will be processed through the Computer Vision Technology of deep learning to capture every details by which a Deep Learning Algorithm will be built.

The new patients data will be fed to this model which detects the Lung Opacity indication along with its location such that Clinical specialists will be able to confirm diagnosis quickly and can help in taking respective decisions quickly to move forward with the next steps of the treatment.

Deep neural networks models have conventionally been designed and experiments were performed upon them by human experts in a continuing trial and error method. This process demands enormous time, knowhow and resources. To overcome this problem, a novel but simple model is introduced to automatically perform optimal classification tasks with deep neural network architecture .

The Neural network architecture was specifically designed for Pneumonia image classification tasks. The proposed technique is based on the CNN algorithm, utilizing set of neurons to convolve on a given sample images to extract relevant features from them. This is demonstrated through validating the accuracy of the detection along with the objective to reduce the loss while the network is learning the details.

As part of this project, we will be demonstrating the outcome with 4 different model architecture along with their outcome in each of the model and provide the commentary for each of the models developed.

# Project Objective

The objective of this capstone project is to build a Pneumonia detection system to locate the position of the inflammation in a Chest Radiography.

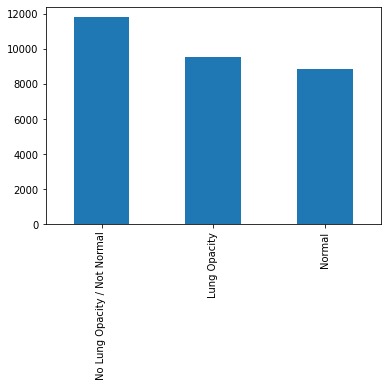
Based on the dataset provided, we will have to do the following steps to work towards building the final model and validate and results.

* Validate the images and respective bounding box coordinates.
* Extract all the features from the images and build the csv file for processing further.
* Perform Exploratory Data Analysis to validate all the data points and build insights
* Based on the project objective – isolate the dataset and accordingly images as well which are fully unique by removing the duplicate records in the dataset based on target variable.
* Build the model with different architectures and showcase the accuracy and showcase the right model to approach the problem description

# EDA Inference and Data pre-processing

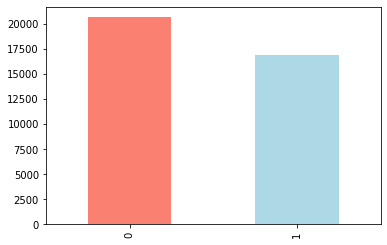
|  |
| --- |
|  |

1. Based on the sample dataset provided – we see that there are 3 class values present..



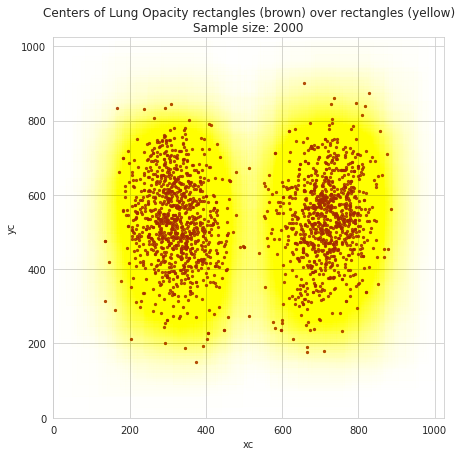
Our objective would be look at images having Lung Opacity and identify the right bounding box to locate the area of inflammation, hence we may need to club them into Lung Opacity and others as another single category

1. Post merging both the documents – class information with the labels document we see the following distribution on the “Target” variable



We observe there are more records not having lung opacity issues, which may lead to bias.

1. We observe the following heat map making the visualization where in the Target = 1



The scatter plot with brown dots shows the areas of inflammation highlighted which needs to be learnt to identify the issues.

1. Following are set of Metadata that we are able to notice from each of the chest X-ray’s provided..

Dataset.file\_meta -------------------------------

(0002, 0000) File Meta Information Group Length UL: 202

(0002, 0001) File Meta Information Version OB: b'\x00\x01'

(0002, 0002) Media Storage SOP Class UID UI: Secondary Capture Image Storage

(0002, 0003) Media Storage SOP Instance UID UI: 1.2.276.0.7230010.3.1.4.8323329.28530.1517874485.775526

(0002, 0010) Transfer Syntax UID UI: JPEG Baseline (Process 1)

(0002, 0012) Implementation Class UID UI: 1.2.276.0.7230010.3.0.3.6.0

(0002, 0013) Implementation Version Name SH: 'OFFIS\_DCMTK\_360'

-------------------------------------------------

(0008, 0005) Specific Character Set CS: 'ISO\_IR 100'

(0008, 0016) SOP Class UID UI: Secondary Capture Image Storage

(0008, 0018) SOP Instance UID UI: 1.2.276.0.7230010.3.1.4.8323329.28530.1517874485.775526

(0008, 0020) Study Date DA: '19010101'

(0008, 0030) Study Time TM: '000000.00'

(0008, 0050) Accession Number SH: ''

(0008, 0060) Modality CS: 'CR'

(0008, 0064) Conversion Type CS: 'WSD'

(0008, 0090) Referring Physician's Name PN: ''

(0008, 103e) Series Description LO: 'view: PA'

(0010, 0010) Patient's Name PN: '0004cfab-14fd-4e49-80ba-63a80b6bddd6'

(0010, 0020) Patient ID LO: '0004cfab-14fd-4e49-80ba-63a80b6bddd6'

(0010, 0030) Patient's Birth Date DA: ''

(0010, 0040) Patient's Sex CS: 'F'

(0010, 1010) Patient's Age AS: '51'

(0018, 0015) Body Part Examined CS: 'CHEST'

(0018, 5101) View Position CS: 'PA'

(0020, 000d) Study Instance UID UI: 1.2.276.0.7230010.3.1.2.8323329.28530.1517874485.775525

(0020, 000e) Series Instance UID UI: 1.2.276.0.7230010.3.1.3.8323329.28530.1517874485.775524

(0020, 0010) Study ID SH: ''

(0020, 0011) Series Number IS: "1"

(0020, 0013) Instance Number IS: "1"

(0020, 0020) Patient Orientation CS: ''

(0028, 0002) Samples per Pixel US: 1

(0028, 0004) Photometric Interpretation CS: 'MONOCHROME2'

(0028, 0010) Rows US: 1024

(0028, 0011) Columns US: 1024

(0028, 0030) Pixel Spacing DS: [0.14300000000000002, 0.14300000000000002]

(0028, 0100) Bits Allocated US: 8

(0028, 0101) Bits Stored US: 8

(0028, 0102) High Bit US: 7

(0028, 0103) Pixel Representation US: 0

(0028, 2110) Lossy Image Compression CS: '01'

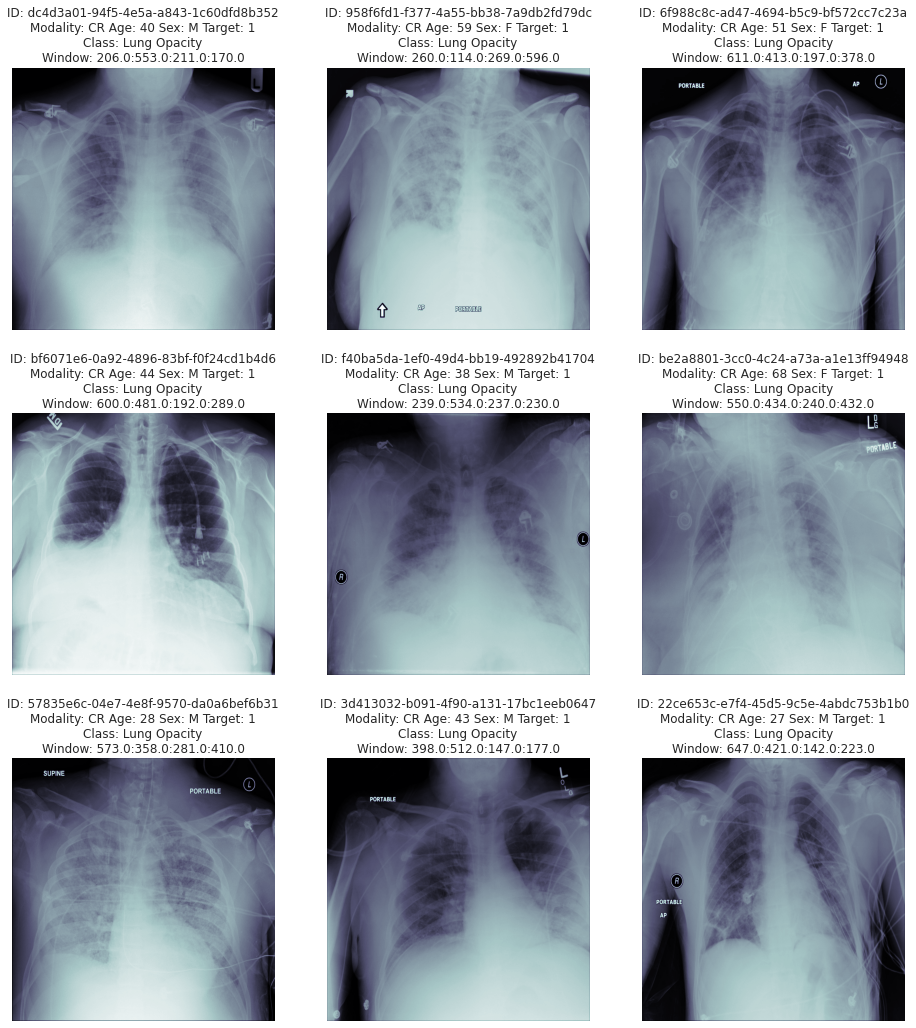
(0028, 2114) Lossy Image Compression Method CS: 'ISO\_10918\_1'

(7fe0, 0010) Pixel Data OB: Array of 142006 elements

1. Of the above metadata – basis the observation of key fields names and its values, we decided to consider following metadata to form the CSV file for validating and processing further..

* 'patientId'
* 'Modality'
* 'PatientAge'
* 'PatientSex'
* 'BodyPartExamined'
* 'ViewPosition'
* 'ConversionType'
* 'Rows'
* 'Columns'
* 'PixelSpacing'

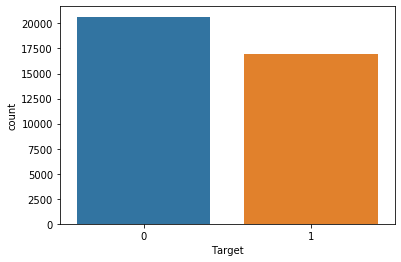
1. Processing the dicom images, we see the following



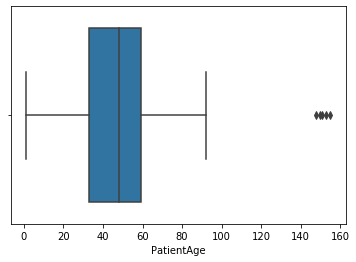
We observe all the images are 1024/1024 shape and it needs to be reshaped before feeding them into the model.

Based on the processed CSV file from the metadata – EDA was performed on this and below are the findings and also pre-processing steps taken to perform the data analysis

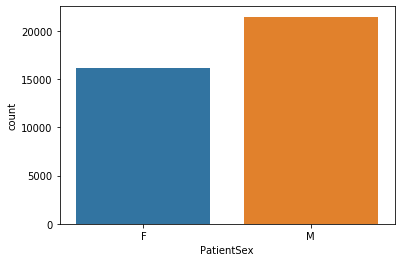
* EDA performed on the data taken out from each of parameters extracted from the images and clubbed against their patient ID's to validate them for its accuracy, impact based on availability of information overall. Below are some of the key findings.
* There are totally 37,629 records available in the dataset where in 16 fields were extracted for validation
* For Pnuemonia detection - "Target" Field provides the classification. 0 : No Lung Opacity, 1 : Lung Opacity



* "class" Field provides 3 groups. However when its seen against Target classification and respective coordinates availability - its doesn't seems to isolate between Normal and Not Normal cases. There are may be other problems, for the purpose of this exercise this observation will be ignored
* "Target" has 20,672 records/images which are Not having Lung Opacity and 16,957 having lung Opacity. We observe 55% of the images doesn't have lung opacity and only 45% having Lung Opacity - This may create imbalance in prediction tending towards not having Lung Opacity - Need to observe this furhter for duplicate records and plan for data augmentation
* "PatientAge" - We observe 5 records having ages above 100 - which should be dropped



* "PatientSex" - We observe 55% of male records in the total count. This may not be a factor for recognition hence decided to continue even though there is imbalance..

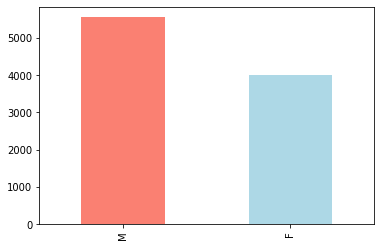


* We observe 9555 records are unique - which needs to be considered for model building
* For processing further – we have taken the ROWS having Target = 1, we found that there are 16K records
* Post processing for Duplicate records – we see that the record count comes to 9555 records..

1 9555

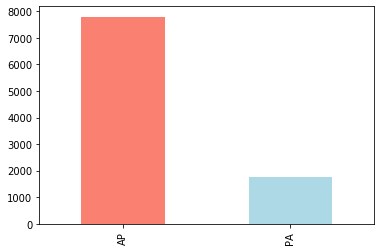
Name: Target, dtype: int64

* Of the 9555 records – below is the distribution on M / F



Since Male records are more – this may create data imbalance even though it may not be crucial at this time this needs to be validated.

* With the column “ViewPosition” we notice the below findings..



### With respect to Column "ViewPosition" correlated with the Target value of 1 - There is high imbalance that we observe with the above information having AP (Anterior-Posterior). One way to interpret this target unbalance is that patients that are imaged in an AP position are those that are more ill, and therefore more likely to have contracted pneumonia. Note that the absolute split between AP and PA images is about 50-50, so the above consideration is extremely significant

* Validating the target with age group – we observe the below pattern.

PatientAge count

0 0-10 219

1 11-20 602

2 21-30 1308

3 31-40 1591

4 41-50 1675

5 51-60 2226

6 61-70 1313

7 71-80 518

8 81-90 100

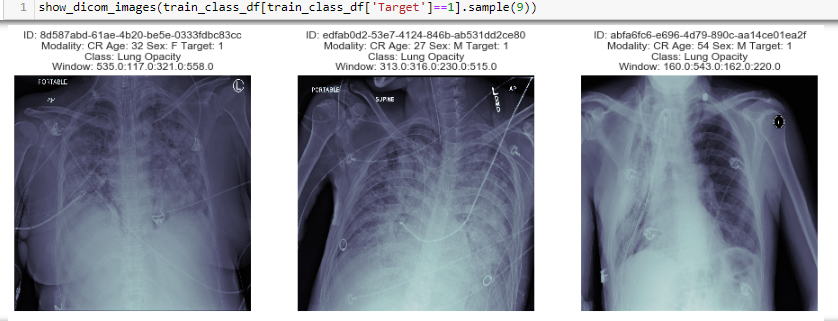
9 91-100 3

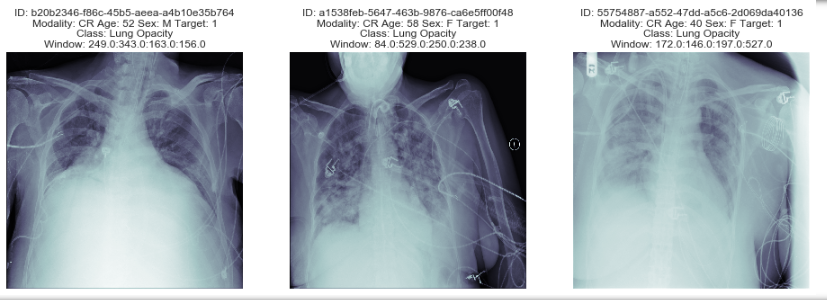
10 more than 100 0

Between age from 21 to 70 – The problem is highly detected and specifically 51 to 60 has more occurrence. We also observe some of the outlier which may needs to be dropped.

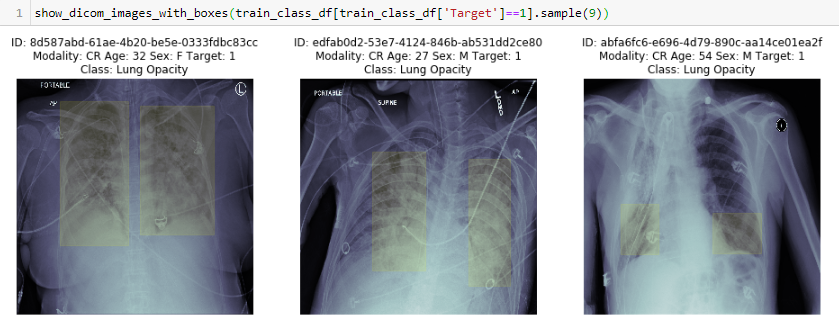
* We were able to extract and visualize the diacom images provided and visual representation are detailed below.

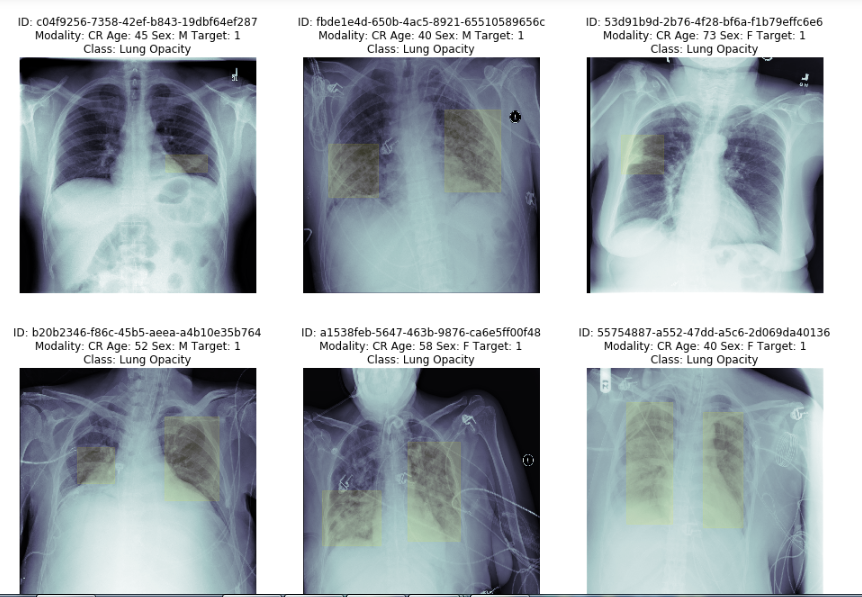
**RAW Images:**





**RAW Images along with its bounding boxes (for the same samples above):**





* Post this process as part of pre-processing we also associated the proper labelling for the respective images identified for processing further, we also validated this against the total records of 9555 to make sure that images along with associated labels are properly matched.

# Interim Approach on Model building and Model Selection

Deep neural networks models have conventionally been designed and experiments were performed upon them by human experts in a continuing trial and error method. This process demands enormous time, knowhow and resources. To overcome this problem, a novel but simple model is introduced to automatically through CNN models.

We also noted that constructing and training a complex deep learning model from scratch is mostly infeasible due to the lack of hardware infrastructure. Therefore, we decided to exploits the idea of transfer learning which is the improvement of learning in a new prediction task through the transfer of knowledge from a related prediction task that has already been learned. This will improve the current computer vision methods based on the use of deep learning to diagnose X-rays images more effectively. By utilizing convolutional neural networks re-trained with our obtained data, we would like to experiment them to achieve the greater classification accuracy.

We decided to develop the following models as part of our experiment and present our findings with details.

* MobileNet
* YOLO
* SSD
* Mask R-CNN

Following section details the MobileNet Model implementation details:

As we begin our MobileNet implementation by adopting to the Transfer learning operation, we selected the model architecture through Tensor Hub which can be referenced in this URL <https://tfhub.dev/google/imagenet/mobilenet_v2_140_224/feature_vector/4>

TensorFlow Hub 2.0 allows the easier way to select the appropriate model based on our requirement and provides us the directional inputs on using that further in our model building exercise. We wanted to attempt this as a new feature to explore during this project cycle and selected Version 4 for our development purposes.

Some of the key steps / decisions taken as part of model development is detailed as below.

**Pre-processing data:**

1. Pre-process the image by converting them from Dicom to JPG images
2. Convert the images into Tensors
3. Resize the image from 1024/1024 to 224 / 224 as expected by MobileNet architecture

The above steps are defined as functions so that it returns the appropriate images.

**Turning our data into batches:**

To make the model development efficient and faster to visualise we set the batch size to 32. Say, we are process 1000+ images they all may not fit into the memory, hence decided to set this to 32 per batch and this can be modified based on trials.

Also, in order to use TensorFlow effectively, we need our data in the form for Tensor Tuples which looks like (image, lable). For which we have defined a function “get\_image\_label” which does all the steps of image conversion, processing, reshaping and returns the images and corresponding labels.

**Building the model:**

Below steps were followed to build the model..

# Setup input shape to the model

INPUT\_SHAPE = [None, IMG\_SIZE, IMG\_SIZE, 3] # batch, height, widhth, color channelsl

# Setup output shape of our model

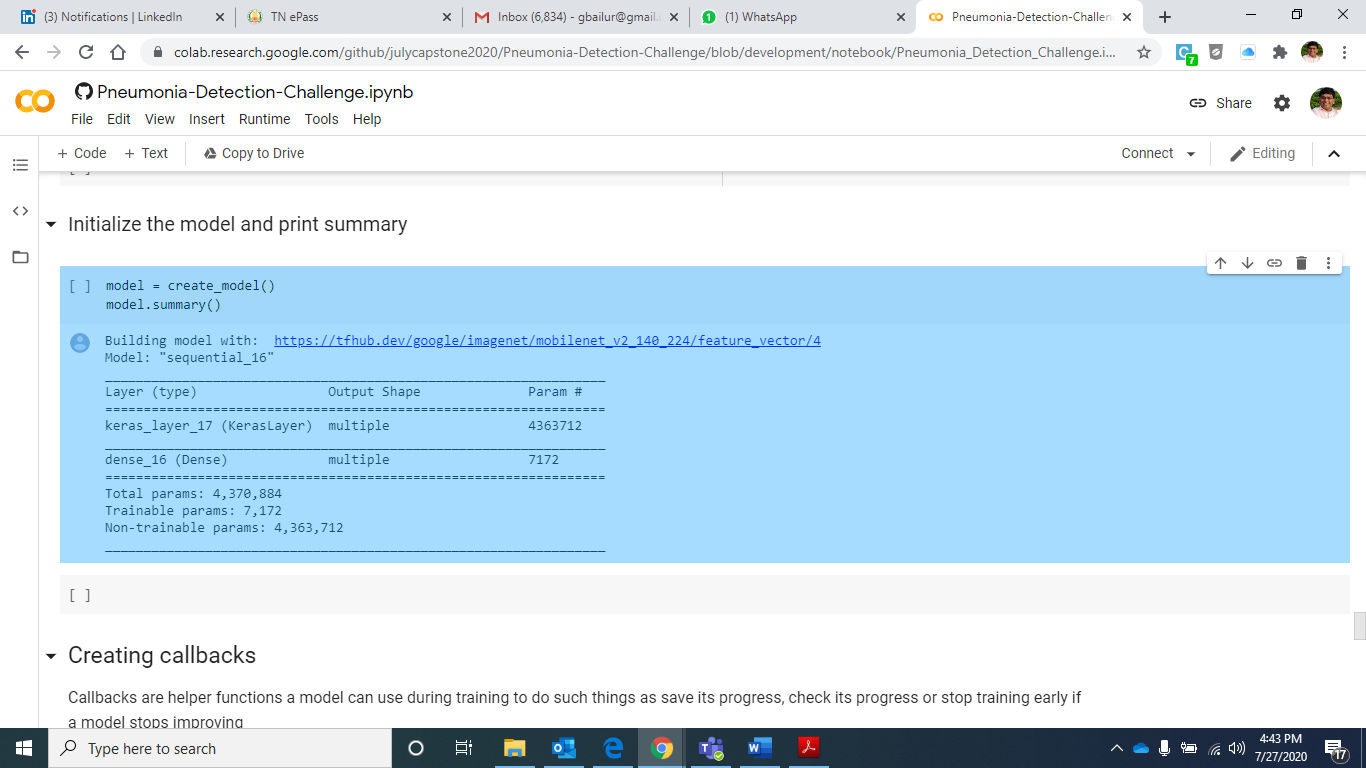
OUTPUT\_SHAPE = 4

# Setup model URL from TensorFlow Hub

MODEL\_URL = "https://tfhub.dev/google/imagenet/mobilenet\_v2\_140\_224/feature\_vector/4"# @param ["https://tfhub.dev/google/imagenet/mobilenet\_v2\_140\_224/feature\_vector/4"]

NOTE: Model URL was provided to direct to the mobilenet architecture that we decided to use going forward..

**The model summary is defined below:**



**Defining callback:**

We have setup the Tensorboard feature to define the call back and below steps are taken to address them.

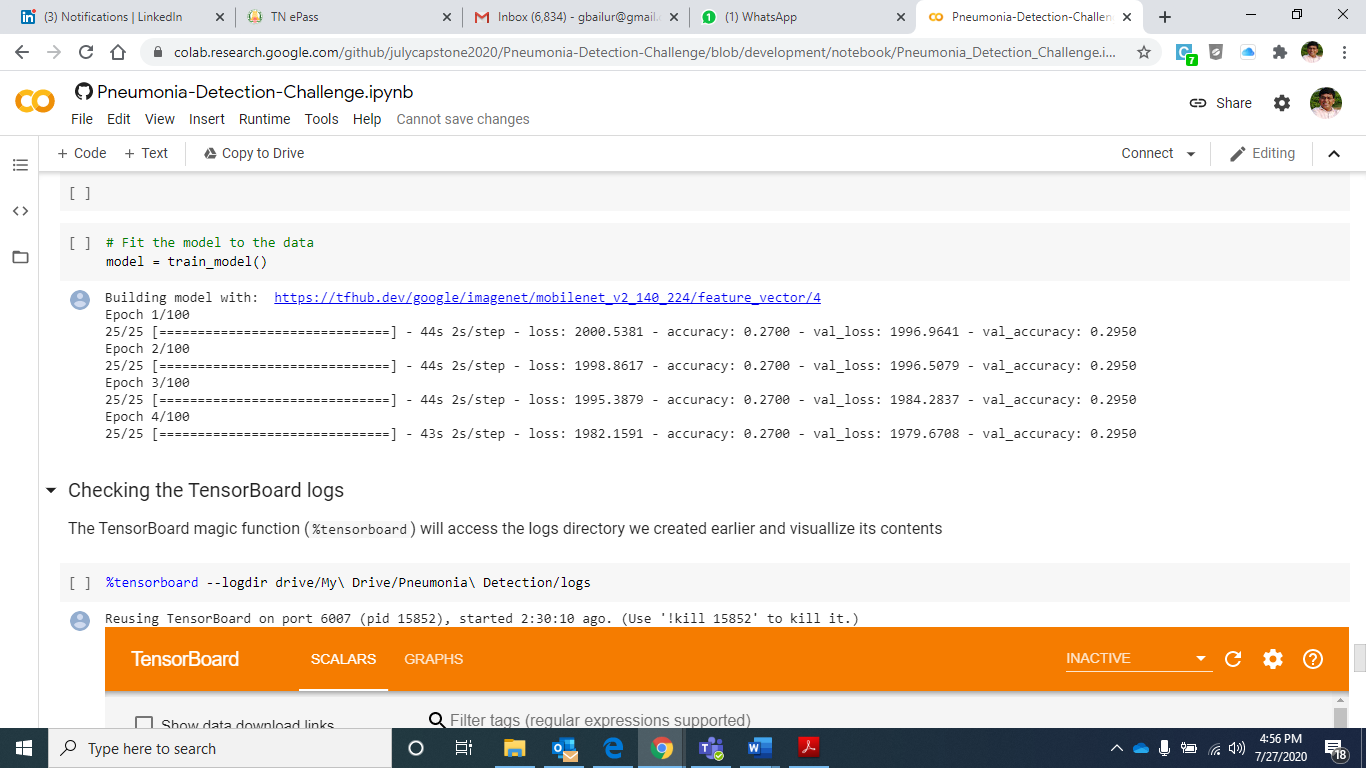
* Load the tensorboard notebook extension
* Create tensortboard callback – by which we will able to save logs to a directory and pass it to our model during fit
* To visualize our training logs we will use %tensorboard magic function

**Defining Early stopping:**

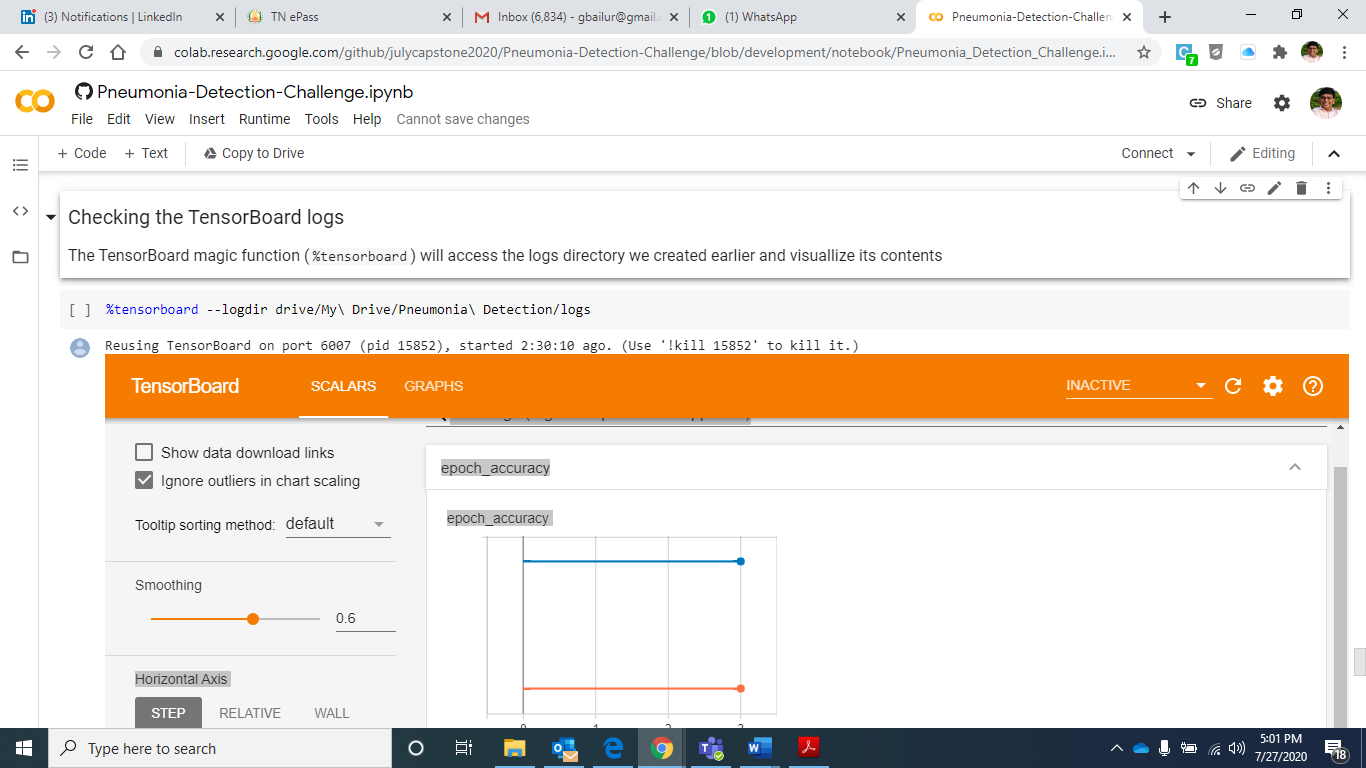
We have also defined early stopping to ensure we stop the process if there is a overfit scenario and/or loss value is not improving much. This way we will able to validate and re-run the model by changing right hyper parameter for better accuracy outcome.

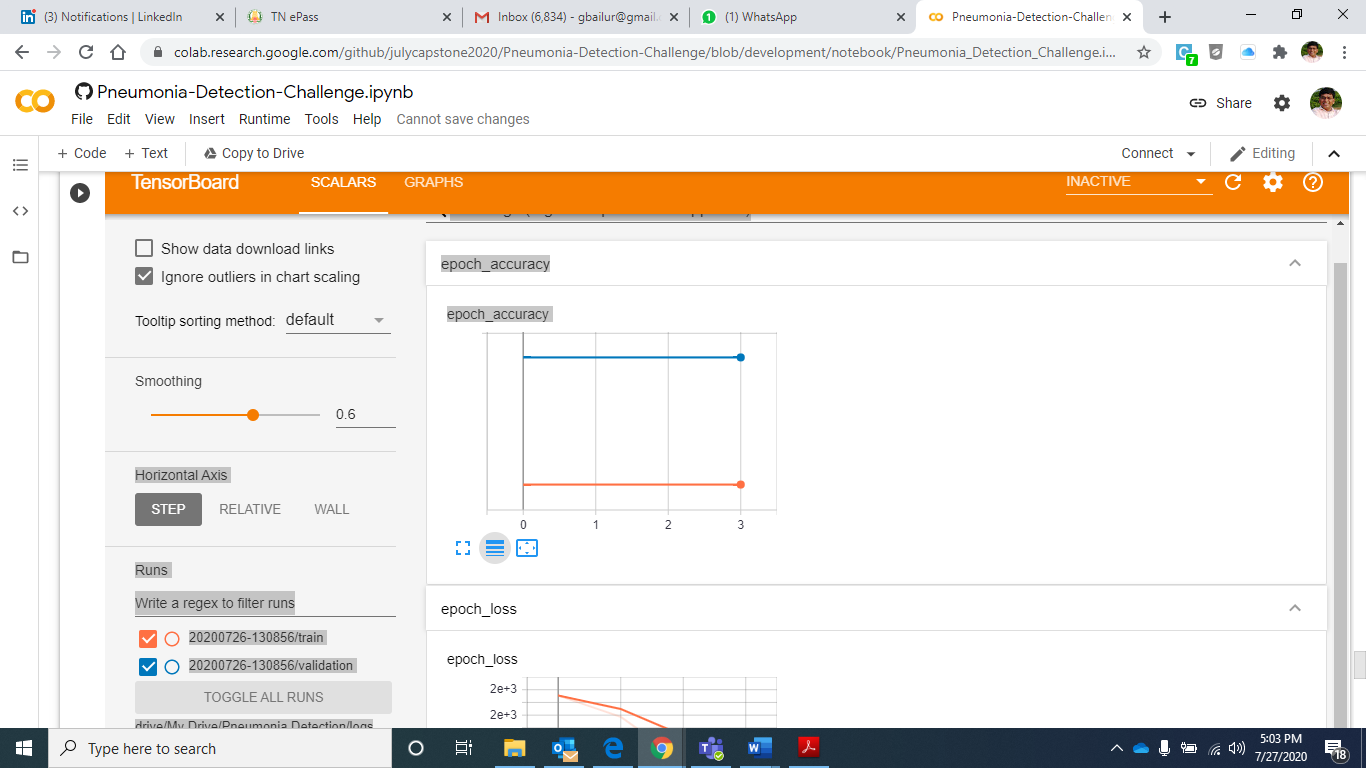
**Running the model:**

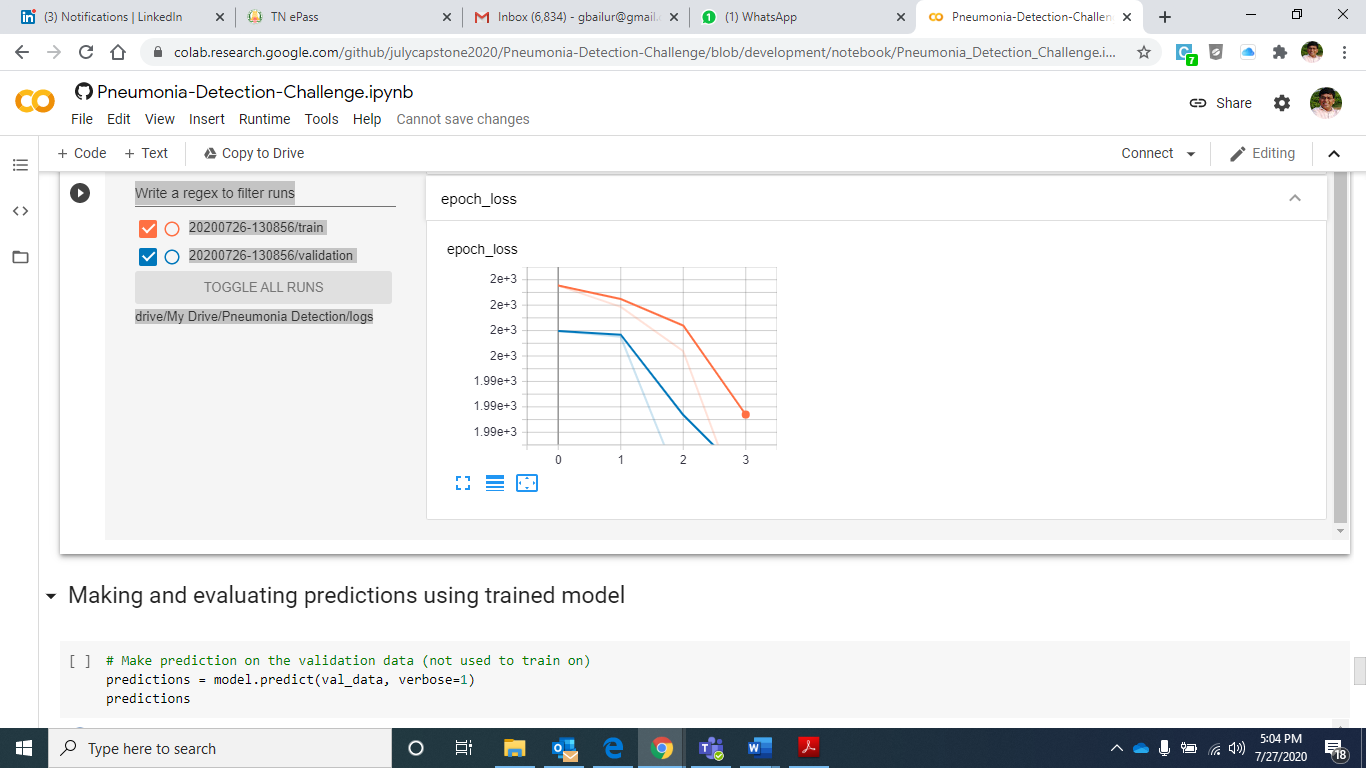
We have set the epochs to 100 to start with and called the fit function to execute the model.. Below is our INTERIM Observation..



**Validating the logs:**







We did get into some of the challenges in progressing with the image translation and hence we observe high loss and very low accuracy to start with. We will continue further to address the challenges with the above approach to drive a very high accuracy when we finally conclude on this.

# Model Development

# 

# We continued the effort to build the model further and this section details model building process and associated outcome.

# 

# Mobilenet

# Mobilnet represent the efficient models for embedded computer vision applications. MobileNets are based on a streamlined architecture that uses depth-wise separable convolutions to build light weight deep neural networks. Mobilenets are built primarily from depthwise separable convolutions and subsequently used in inception models to reduce the computation in first few layers.

# 

# Mobilenet Application reference

# The code implementation path:

# <https://github.com/julycapstone2020/Pneumonia-Detection-Challenge/blob/development/notebook/Pneumonia_Detection_Challenge_5000.ipynb>

# Model was trained on 5000 images for this purpose with following parameters

# Train / test split of 80:20

# Batch size of 32

# Loss function: Categorical cross entropy

# Metrics: Accuracy

# Optimizer: Adam

# Activation function: Softmax

# Sample image representations

# **Visualizing the first 25 images from the training batch:**

show\_25\_images(train\_images, train\_labels)

# 

# **Visualizing the first 25 images from the validation batch:**

show\_25\_images(val\_images, val\_labels)

# 

# Model Architecture

# 

# This section details the core layers that Mobilenet is built on which are depthwise separable filters. This is a form of factorized convolutions which considers a standard convolution into a depthwise convolution and a 1x1convolution called a pointwise convolution.

# 

# Mobilenet Body Architecture is as depicted below

# 

# 

# 

# **Current implementation of Model Summary:**

# 

**Building model with:** [**https://tfhub.dev/google/imagenet/mobilenet\_v2\_140\_224/feature\_vector/4**](https://tfhub.dev/google/imagenet/mobilenet_v2_140_224/feature_vector/4)

**Model: "sequential"**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Layer (type) Output Shape Param #**

**==============================================================**

**keras\_layer (KerasLayer) multiple 4363712**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**dense (Dense) multiple 7172**

**===============================================================**

**Total params: 4,370,884**

**Trainable params: 7,172**

**Non-trainable params: 4,363,712**

# 

# Results and conclusion

# Model was executed for 100 epochs, however we saw that model exited since there is not much improvement in accuracy.

# 

# We leveraged Tensorboard to depict the accuracy results and below are representations of them.

# **Epoch Accuracy:**

# 

# **Epoch Loss:**

# 

# Based on the results, Mobilenet architecture may not be suitable model to rightly understand for the bounding box scenario’s. Hence we moved on implementing the relevant model to validate the outcomes for the specific tasks.

# Yolo

The YOLO framework (You Only Look Once) deals with object detection in a different way. It takes the entire image in a single instance and predicts the bounding box coordinates and class probabilities for these boxes. The biggest advantage of using YOLO is its superb speed – it’s incredibly fast and can process 45 frames per second. YOLO also understands generalized object representation. We present a residual learning framework to ease the training of networks that are substantially deeper than those used previously. We explicitly reformulate the layers as learning residual functions with reference to the layer inputs, instead of learning unreferenced functions. We provide comprehensive empirical evidence showing that these residual networks are easier to optimize, and can gain accuracy from considerably increased depth. On the ImageNet dataset we evaluate residual nets with a depth of up to 152 layers---8x deeper than VGG nets but still having lower complexity.

YOLO is considered best for real time detection.

# YOLO Application Reference

**Code implementation path:**

<https://github.com/julycapstone2020/Pneumonia-Detection-Challenge/blob/master/notebook/yolo%20(2).ipynb>

# For YOLO implementation we explored Transfer learning option and finalized to use Darknet implementation and leverage the benefits of this deployment to drive better accuracy results. As part of that, we decided to close the darknet package and modify the configurations according to our model requirement which needs to be implemented. With some of the basic due diligence tests performed by testing Yolov3 and Yolov4, we finalized to adopt to use Yolov4 algorithm for our Pneumonia deployment perspective.

# 

# For Darknet model configuration – we referred to the deployment guide in the link

# <https://colab.research.google.com/drive/1lTGZsfMaGUpBG4inDIQwIJVW476ibXk_#scrollTo=wkzMqLZV-rF5>

# This below reference diagram was referred to build the clone of the project and develop the project specific directory structure along the specific parameters that we adopted to make our model suit the requirement.

# 

As part of that for transfer learning we leveraged **yolov4-tiny.conv.29** weights while executing the model on all the images.

For processing them further in the model. Per Yolo’s recommendation, we also decided to use metrics of Average IOU (Intersection Over Union) Metrics to measure the success of the model.

# Below diagram shows the accuracy measures of Average IOU scores for the model developed.

# Sample Image representations

# Based on the preformed tests below are the ground truth representations that we have in the dataset against which we will be doing the predictions

# 

# The below images represents the actual predictions for the tests performed along with its bounding boxes. Only few samples are represented as below.

# 

# Model Architecture

ResNet has won several competitions and its architecture allows for better learning in deeper networks. I’ve used the Keras implementation with weights of ResNet50 and modified the code to have the YOLO classifier at the end.

# 

The YOLO detector is broken into three main pieces.

* **YOLO Backbone —** The YOLO backbone is a convolutional neural network that pools image pixels to form features at different granularities. The Backbone is typically pretrained on a classification dataset, typically ImageNet.
* **YOLO Neck —**The YOLO neck (FPN is chosen above) combines and mixes the ConvNet layer representations before passing on to the prediction head.
* **YOLO Head** — This is the part of the network that makes the bounding box and class prediction. It is guided by the three YOLO loss functions for class, box, and objectness.

# Result and conclusions

# Based on the test results plotted we see the below prediction trend for with foreground and background outcome.

# 

# With respect to the above outcome of high IOU coverage in prediction we consider YOLO to be one of the best model to adopt for such predictions.

# Mask RCNN

# Mask R-CNN is been a deep neural network which is aimed to solve instance segmentation problem in Computer Vision and Machine Learning. This can separate different objects in an image or video by providing object bounding boxes, classes and masks associated with the image or video. Mask R-CNN first generates proposals about the regions where there might be an object and it predicts the class of the object. In this process it refines the bounding boxes and generates a mask at pixel level of the object based on the proposal created.

# Mask-RCNN Application references

# Code implementation path:

# <https://github.com/julycapstone2020/Pneumonia-Detection-Challenge/blob/master/notebook/maskRCNN.ipynb>

# For this specific implementation – we referenced the “[matterport](https://github.com/matterport/Mask_RCNN)” package implementation of maskr-cnn and cloned it to use it in our project. Details of this is available in this [link](https://github.com/matterport/Mask_RCNN)

# The reference that cloned includes the following;

# Source code was built on Mask R-CNN with FPN and ResNet101

# Pretrained weights for MS COCO dataset

# ParallelModel class for multi-GPU training.

# Since this model also requires annotations on the images, we also explored options by referring to the details on Annotator project provided by MD.ai and this can be referenced in this [link](https://public.md.ai/annotator/project/LxR6zdR2/workspace.)

# Below image refers to the code block where we are leveraging the MD.ai annotations..

# 

# 

# Below section represents the key configuration items for the model that we developed by leveraging Resnet50 model.

**Configurations:**

BACKBONE resnet50

BACKBONE\_STRIDES [4, 8, 16, 32, 64]

BATCH\_SIZE 16

BBOX\_STD\_DEV [0.1 0.1 0.2 0.2]

COMPUTE\_BACKBONE\_SHAPE None

DETECTION\_MAX\_INSTANCES 3

DETECTION\_MIN\_CONFIDENCE 0.9

DETECTION\_NMS\_THRESHOLD 0.1

FPN\_CLASSIF\_FC\_LAYERS\_SIZE 1024

GPU\_COUNT 1

GRADIENT\_CLIP\_NORM 5.0

IMAGES\_PER\_GPU 16

IMAGE\_CHANNEL\_COUNT 3

IMAGE\_MAX\_DIM 64

IMAGE\_META\_SIZE 14

IMAGE\_MIN\_DIM 64

IMAGE\_MIN\_SCALE 0

IMAGE\_RESIZE\_MODE square

IMAGE\_SHAPE [64 64 3]

LEARNING\_MOMENTUM 0.9

LEARNING\_RATE 0.001

LOSS\_WEIGHTS {'rpn\_class\_loss': 1.0, 'rpn\_bbox\_loss': 1.0, 'mrcnn\_class\_loss': 1.0, 'mrcnn\_bbox\_loss': 1.0, 'mrcnn\_mask\_loss': 1.0}

MASK\_POOL\_SIZE 14

MASK\_SHAPE [28, 28]

MAX\_GT\_INSTANCES 3

MEAN\_PIXEL [123.7 116.8 103.9]

MINI\_MASK\_SHAPE (56, 56)

NAME pneumonia

NUM\_CLASSES 2

POOL\_SIZE 7

POST\_NMS\_ROIS\_INFERENCE 1000

POST\_NMS\_ROIS\_TRAINING 200

PRE\_NMS\_LIMIT 6000

ROI\_POSITIVE\_RATIO 0.33

RPN\_ANCHOR\_RATIOS [0.5, 1, 2]

RPN\_ANCHOR\_SCALES (32, 64, 128, 256, 512)

RPN\_ANCHOR\_STRIDE 1

RPN\_BBOX\_STD\_DEV [0.1 0.1 0.2 0.2]

RPN\_NMS\_THRESHOLD 0.7

RPN\_TRAIN\_ANCHORS\_PER\_IMAGE 16

STEPS\_PER\_EPOCH 100

TOP\_DOWN\_PYRAMID\_SIZE 32

TRAIN\_BN False

TRAIN\_ROIS\_PER\_IMAGE 16

USE\_MINI\_MASK True

USE\_RPN\_ROIS True

VALIDATION\_STEPS 50

WEIGHT\_DECAY 0.0001

As part of transfer learning we leveraged the weigh file provided by the model development and training.

**mask\_rcnn\_pneumonia\_0006.h5**

# 

# Sample image representations

# While developing the model, initially we started with training for 1000 images samples and then extended further.

# Below represents one such reference having the bounding box and masks applied to it.

# 

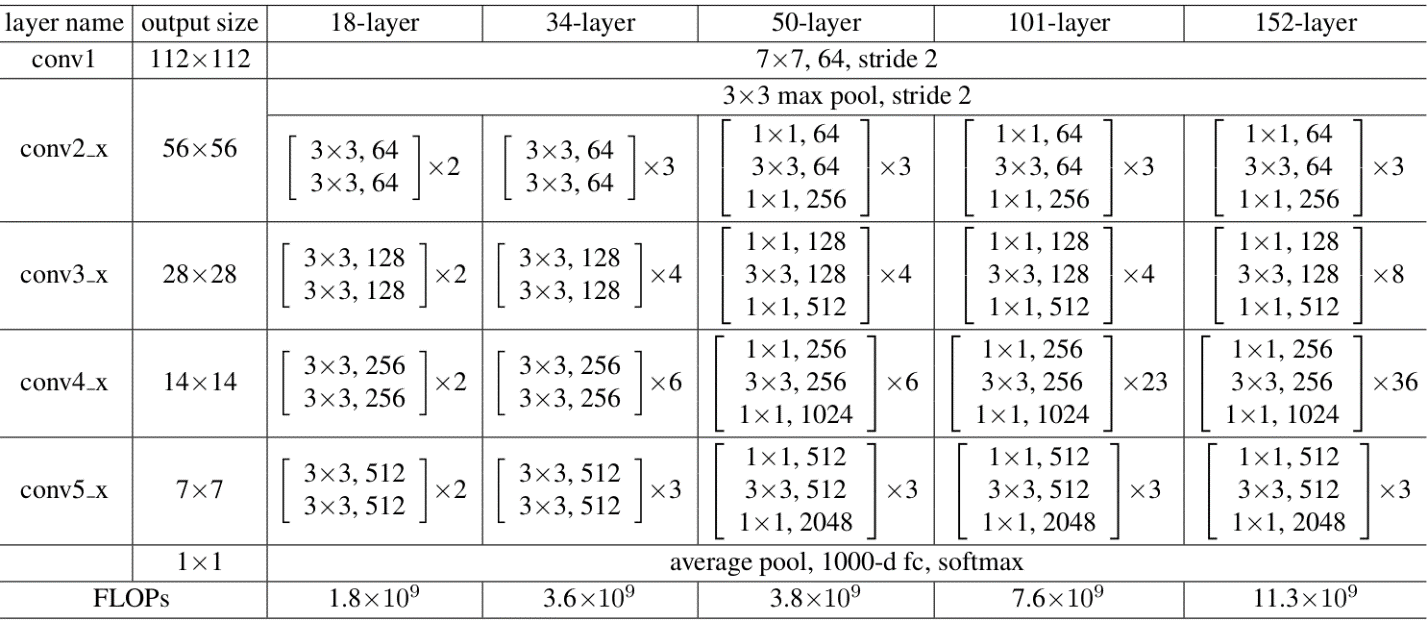
# Model Architecture

# Mask R-CNN used Resnet 101 architecture to extract features from the images. However for this model development we leveraged the Resnet50 Architecture.

# 

## Base architecture of Resnet50 is detailed below.

## ResNet50 Architecture



# Results and Conclusion

# Here are the details of steps that we followed to train the model and results in each phases.

# With Epoch set to 1, the results are as below.

# 

# With Epoch set to 5:

# 

# As we progressed further, below are actual results that we have noted.

# 

# 

# 

# 

# SSD

# SSD Application references

# Sample image representations

# Model Architecture

# Results and Conclusion

# Next Steps and conclusion

As we continue further, our objective is to build the model with 4 different architectures by adopting the transfer learning techniques and we will continue to try on YOLO, SSD and Mask R-CNN along with Mobilenet..

Based on accuracy level and understanding the model behaviour further for each of them, In order to tune these models further – we will follow the below steps.

* Adding a layer in addition to what the model provides
* Validating the hyperparameters based on the results to drive the improvement in accuracy
* Trying different loss function and optimizers to see where things can be improved.