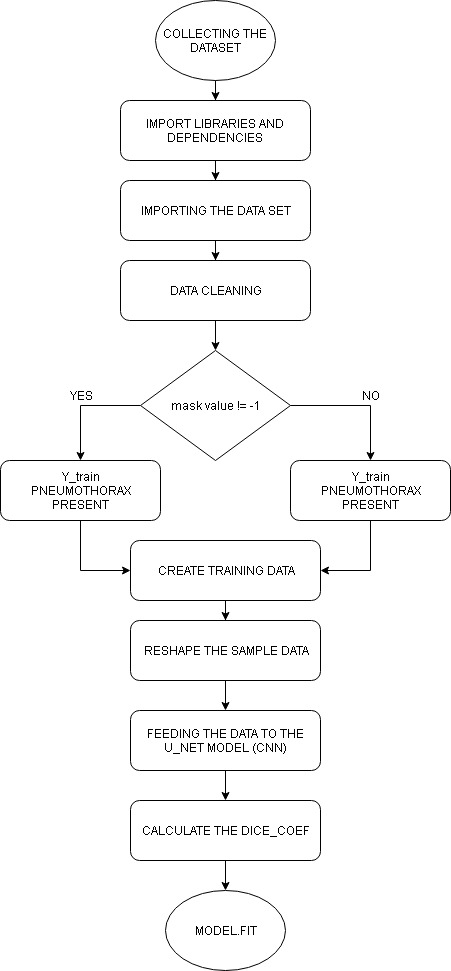
|  |
| --- |
| Identifying  Pneumothorax  Deep Learning on Python |
| Kunal Gehlot  Gehlotkunal4@outlook.com |



## Flow Chart of Training Process

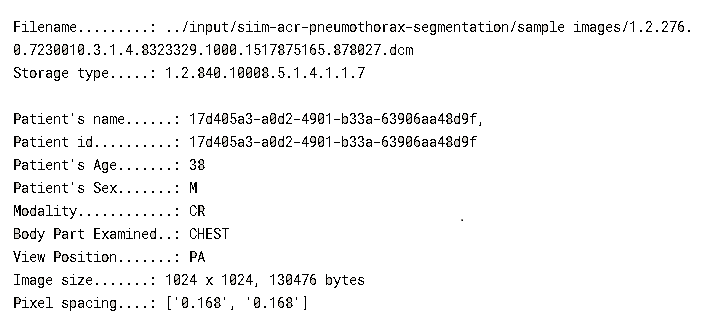


## Collecting the Database

The data base for the problem contains images and annotations. The data set was made available for the purpose of early recognition of **pneumothorax**.

A sample submission file was also provided as to classify what will be the data stored within the **submission.csv** file. The data is comprised of images in **DICOM** format and annotations in the form of image **IDs** and **run-length-encoded (RLE)** masks. Some of the images contain instances of **pneumothorax**, which are indicated by encoded masks in the annotations.

Information stored in a **DICOM** file



This files also contains the **RLE** values which are used to find the presence of pneumothorax. If the value of **RLE** is **-1** then **pneumothorax** is not present but if the value of **RLE** is not **-1** and is an **array of pixels** then **pneumothorax** is present. The **pixel array** is used to find the location of the area in which **pneumothorax** is present and with that we can create a mask to better display the affected area.

## Import and Cleaning the Data

The data contains **dicom-images-train**, **dicom-images-test, train-rle.csv** and **train-rle-sample.csv** which is used for the prediction. The total number of **files** in the training dataset are 10712 and the number of **files** in test data set are 1377 which will be used for making the predictions.

The imported data set is then sorted. We will **read train-rle.csv** file. This file contains the **RLE** for the respected file. The number of entries in the **train-rle file** are 11582.

We have 10712 number of **files** and 11582 number of **RLEs** for the files. The **RLE** values is then stored inside a list. The next step is finding that how many of the images are annotated.

The process of data cleaning involves removing the duplicate **files**. In the data set some files doesn’t have any **RLE** values which were also dropped in the data cleaning process. After the data cleaning process two data are created **ds1** and **ds2** which stores values of **RLE** and **indices (contains ImageId and EncodedPixels)** respectively.

This data cleaning process is performed for improving the overall accuracy of the model and improve the prediction.

A small part of the data set is printed to visualize the format of the dataset.

Next step is checking the **RLE** and printing the images of the lungs without **pneumothorax** images which have **pneumothorax** overplayed with mask.

**RESHAPE THE SAMPLE DATA**

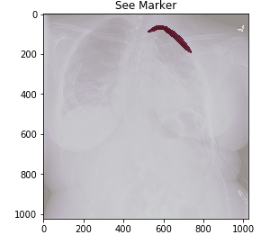
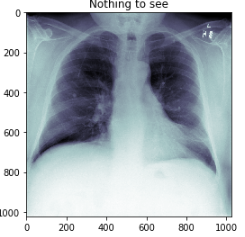
The **X\_train** and **Y\_train** dataset is then reshaped for better processing and change the size of the training and validation samples.

## Create Training Data

The imported files from the data base will be used for training the model.

**X\_train** and **Y\_train** two files will be trained in this process which will then be used for predicting the loss and the **dice\_coeff** of the images in the dataset. **X\_train** is produced using the pixel array of the files in the data set. **Y\_train** is also produced which will store the information of the **RLEs** of the files. The **RLE** of the file contains two types of value. The first value represents the healthy lungs or (lungs with no **pneumothorax**) and the second value defines the affected area (in which **pneumothorax** is present)**.** For that we will call **rle2mask** function which will convert the **RLE** files and to **mask** and will overlap them on the images.

The images that doesn’t have **pneumothorax** have a **RLE** value of -1 and if the **RLE** is in the form of **pixel** **array** then the lungs contains **pneumothorax** disease.



## Creating the Convolution Neural Network (CNN)



The above image shows a **Convolution Neural Network**. It is divided into three layers **Input layer**, **Hidden layer**, **Classification layer**. The hidden layer is the layer which involves the processing part. The **hidden layer** in a **CNN** contains **multiple convolution** and **pooling layers**. Then the images is send to **classification layer**. The number of layers of **CNN** can be increased or decreased to change the total no of parameters on which to train. These parameters are then fed to the model for calculation the loss percentage. The **dice\_coeff** is also calculated in this process.

Formula for calculating the **dice\_coeff**

The smooth in the above formula is used for smoothing the predicted value.

The activation function used is **RELU** and padding is also used for maintaining the original size of the image. The loss is calculated using **binary\_crossentropy.**

The model take **inputs** and **outputs** variables as inputs for training.

The last step in Training process is training the model. The trained model is saved as **model.h5** and will be used in testing phase.