Deep belief Convolutional Neural Network with Artificial Image Creation by GANs Based Diagnosis of Pneumonia in Radiological Samples of the Pectoralis Major

Tathagat Banerjee, Dhruv Batta and Aditya Jain

December 3, 2020

Deep belief Convolutional Neural Network with Artificial Image Creation by GANs based Diagnosis of Pneumonia in Radiological Samples of the Pectoralis Major

Tathagat Banerjee , Dhruv Batta , Aditya Jain

VIT -AP University , Amravati, India
banerjee.tathagat@vitap.ac.in

Abstract—In this paper, we delineate a comparative classification of Pneumonia using Machine Learning and Deep Learning models. The data used was the dataset of Chest X-Ray Images for Classification made available by (Kermany, 2018) with a total of 5863 images, with 2 classes: normal and pneumonia. For the purpose of correcting the class imbalance between normal and pneumonia images, we use General Adversarial Networks to generate pneumonia ridden images. The comparative classification is built on a final dataset of 19784 images. Logistic regression, SVC, KNN, Random Forest and other machine learning models such as XgBoost and CatBoost are compared with deep learning models such as MobileNet and VGG-16. The machine learning model is based on blob segmentation and detecting the difference between blobs of pneumonia ridden and normal individuals. Finally, a new deep learning model with convolutional and artificial neural networks has been proposed for the classification purpose which increases the accuracy significantly and has a classification report which is best suitable for medical analysis.

Keywords-Neural networks, Machine Learning, ANN, CNN, GAN, F-CNN, Transfer Learning, Artificial Intelligence, Computational Power

I. Introduction

Pneumonia is the single leading cause of mortality in children and is a major cause of child mortality in every region of the world. It is a form of acute respiratory tract infection (ARTI) that affects the lungs. When an individual has pneumonia, the alveoli in the lungs are filled with pus and fluid, which makes breathing painful and limits oxygen intake; this particular stance can be useful in detecting the difference in blobs of a normal person and pneumonia ridden individuals.

The X-Rays and Chest Tomography using CT Scans are cross-sectional images of the body allowing the internal organs to be picturized. The images are further used to identify abnormalities present in a patient.

Plain X-Rays while generating enough information CT Scans provide an edge over since they help detect various blobs in the images. Images of CT Scan and plain X rays have been used from the labeled dataset provided. Images have two classes:-

1. Pneumonia Ridden Patients
2. Normal Healthy Patients

The deep learning model which has been proposed in this paper does the analysis of the CT-Scan with an accuracy of around 98% with real-world data. The model is also not prone to overfitting. This model aims at correctly predicting pneumonia in an image of X-Ray of the chest of the test subject.

II. **Literary Review**

In (Khobragade et al., 2016) proposed that in the detection of pulmonary diseases using chest radiographs was necessary to identify the problem present in the tract. The lung diseases are a major problem to human health and thus for this identification, the referenced paper proposes pulmonary segmentation, extraction of characteristics and classifying it using an Artificial Neural Network for the recognition of the disease. A simple image processing with an intensity-based method was used in conjunction with a method to detect the pulmonary limits and the discontinuities in it. Artificial Neural Networks were used for feed-forward and backpropagation. The problem with this approach is that it uses chest radiographs and the highest infant mortality rate in developing countries due to pneumonia and in these countries, there is little infrastructure and doctors in rural areas to provide the necessary diagnosis.

Many researchers have believed in finding specific patterns for various lung diseases through ultrasound images of the respiratory tract. In (Barrientos et al., 2016) the paper presented a method for automatic diagnostics of pneumonia using ultrasound imaging of the thoracic cavity. The approach presented in the paper is based on the analysis of patterns present in 'rectangular segments' from the sonographic digital images. Utilizing pattern recognition, specific features from the characteristic vectors are obtained and then classified with standard neural networks. A training and testing set of positive and negative vectors were compiled.

The approach is problematic here since they have extracted the image of a single patient to either test or train and obtained a specificity of 100% which points toward a pre-processing conceptual error.

The paper (Rodrigues et al., 2018) presented a way for implementing the Structural Co-occurrence Matrix approach to classify nodules as malignant nodules or benign nodules and also the level of malignancy. The structural co-occurrence matrix technique was applied to extract characteristics of the nodule images and classify them. The SCM was applied in greyscale and images of the Hounsfield unit with four filters, creating eight different configurations. The classification stage used classifiers known as the multilayer perceptron, support vector machine, and k-Nearest Neighbours algorithm which were applied to two tasks: (i) to classify the nodule images as malignant or benign, (ii) to classify the nodules pulmonary lesions at the level of malignancy (1 to 5)

The approach while had a result of 96.7 % for precision and F-score measurements in the first task and 74.5 % accuracy and 53.2 % F-score in the second task but did not achieve the much-required diagnosis of pneumonia pathological change identification and also lapses in terms of accuracy for determining benign and malignant nodules in the respiratory tract.

In (et al A. A. Saraiva) A comparative classification of Pneumonia using Convolutional Neural Networks is proposed in this paper and is an efficient method than the one proposed by Daniel S. Kermany in "Identifying Medical Diagnoses and Treatable Diseases by Image-Based Deep Learning" [5]

The following is their architecture of their model which consists of the inherent problem during the general pre-processing phase as well as lacks the statistical approach in favor of deploying a larger network.

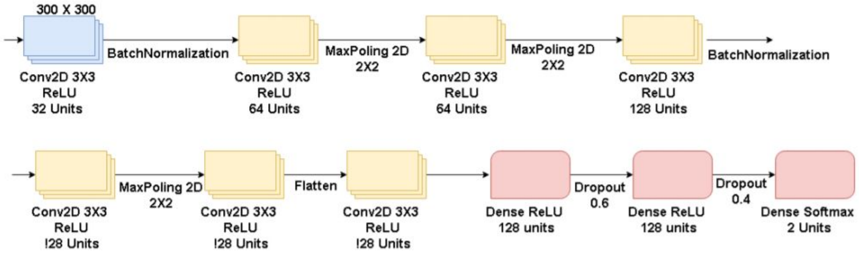
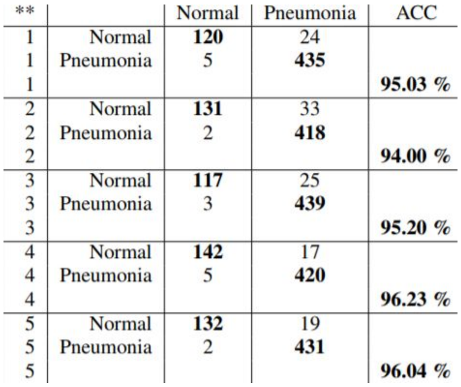


Fig. 1

The images in the training directory are more of the label pneumonia compared to that of normal. Therefore, class balance is not maintained. Hence their accuracy of 95 % is on a dataset with imbalanced classes

Accuracy Matrix of this approach is displayed in the following Fig.2



**		Normal	Pneumonia	ACC
1	Normal	**120**	24	
1	Pneumonia	5	**435**	
1				**95.03 %**
2	Normal	**131**	33	
2	Pneumonia	2	**418**	
2				**94.00 %**
3	Normal	**117**	25	
3	Pneumonia	3	**439**	
3				**95.20 %**
4	Normal	**142**	17	
4	Pneumonia	5	**420**	
4				**96.23 %**
5	Normal	**132**	19	
5	Pneumonia	2	**431**	
5				**96.04 %**

III. **Hardware and Software Requirements**

The Complete experiment has been carried out on an 8 GB, Intel(R) Core(™)i3-6006U CPU @ 2.00GHz 1.99GHz. The operating system is 64-bit Operating System(x64-based processor), Windows 10 Home Single Language.

Python 3.6.1 is used, with Tensorflow 2.0 as software tools for the experimentation. Keras and Scikit Learn are used as dependency libraries for training and getting classification reports. Seaborn and Graphviz is used for plotting and obtaining schematics for the models. OpenCV 2.0 is used to read images and *tdqm* is used for the better graphical user interface.

About 40 to 48 hours of total training time is given to the dataset to achieve good and industrial application level results.

The image quality of the data is set to as 128 x 128 x 3 so that even when unsatisfactory and blurred image data is observed; the quality of results is maintained.

IV. **Transfer Learning Methodology**

Initially, we trained the model with VGG, Mobilenet, and Inception but none of the models gave us expected results when applied on real-time data. The highest accuracy was achieved by the Inception model which was 76 percent on training data but when applied on real-time data(test set) it could only attain an accuracy of 52.3 percent. We have modified the architecture as per our requirement by stacking up a few more layers of deep neural networks which are of type deep belief neural nets, we have named the architecture as Phoenix.

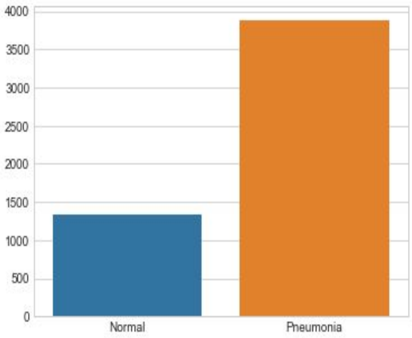
V. **Data**

This dataset contains thousands of validated OCT and Chest X-Ray images.

All images in the dataset (Kermany, 2018) underwent treatment in order to remove all low-quality scans, as well as being classified by two specialist physicians and by a third-party specialist, in order to prevent any misclassification.

As seen in other research on the same dataset, the training dataset lacks images for the 'Normal' label. Therefore, this creates a class imbalance and needs to be solved during the pre-processing.

(Fig of Number of 'Normal' vs 'Pneumonia' Labelled images



In the diagnosis of pneumonia, the alveoli become clogged with secretion and appear as a white spot on the chest radiograph. *Pulmonary consolidation* means that the alveoli in the lungs are filled with inflammatory fluid. In radiography, pulmonary consolidation dovetails to an opacity visible in the tract's radiography, that is, the whitish area. Which can be distinguished in Fig. 3 provided ahead.

Normal *Pneumonia*

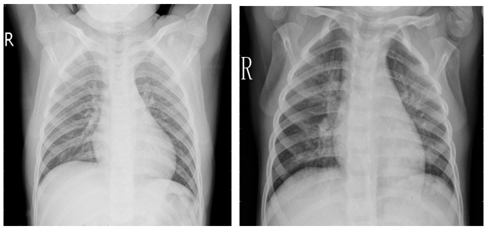


Fig. 3

VI. **Implementation**

a. Machine Learning and Image Analysis

Machine learning has been an important advancement in the field of predictive analysis over the last decade. However, in today's dilemma of predictive analysis both visible (Machine learning) and invisible analysis (Deep Learning) have become part of continuous and prolonged debate. Thus, it is only wise to analyze the given data with both machine learning and deep learning. Our paper ideates the architectural stability and viability of the deep learning and neural network.

Blob segmentation is an important measure that enhances machine learning algorithms to learn. It is important to understand the relics and causes. Basically, blob detection means segmenting the portion of the image which is affected. Using Blob Detection would also create a simpler version of the algorithm to be utilized, whose results are not only generic in nature but simpler and faster to execute as well.

Blob is a group of connected pixels in an image that shares some common property. OpenCV provides a convenient way to detect blobs and filter them based on various characteristics. The algorithm is controlled by using parameters such as Thresholding, Grouping, Merging, Center and radius calculation. In chest tomography and X-Rays of the thoracic cavity, we use parameters set to the specific filter of threshold and inertia. These filters deal with the possibility of the presence of a blob which may be the point of difference between normal and pneumonia images. The thresholding filter is based on the intensity of the blob present while inertia is based on the presence of elongated shape existing in the image which may be a factor. In the following figure, the two images show the blob difference of a healthy person and one suffering from pneumonia.

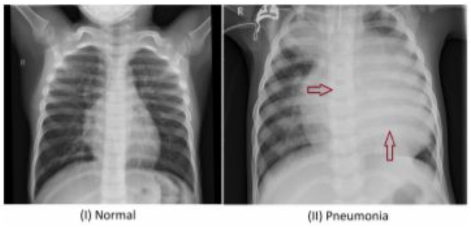


Fig. 4

The main principle mathematical concept of Blob detector is Laplacian Of Gaussian
Given an input image f(x,y), this image is convolved by a Gaussian kernel.

$$g(x, y, t) = \frac{1}{2\pi t} e^{-\frac{x^2+y^2}{2t}}$$

After applying, the laplacian function we get the following equation:-

$$\nabla^2 L = L_{xx} + L_{yy}$$

This provides us with the filter through which our algorithm detects the blobs and the blob segmentation is also pursued through the same mathematical basis.

Even before any pre-processing measure, it is important to build strong and big image data , for which we used deep learning measure of unsupervised learning which is GAN (Generative Adversarial Networks) and VAE (Variational Autoencoders) to prepare artificially equipped and balanced images so, that the medical stability of more 'pneumonia' images does not affect the working ideology and potential of any machine learning algorithm . This way, we not only up-sample the data but also do not create redundancy by using image generation techniques.

The pre-processing phase involves reading the image file names with the help of OpenCV, then labeling all the files in a data frame along with a target variable - Normal, Pneumonia. Finally, shuffling the dataset to remove class consistency problems. Once, the data frame with shuffled, classes are balanced and consistent classes are formed, we then proceed to read the files as per the random shuffled pattern followed in the data frame.After reading the files as images of 128 x 128 x 3 to give all the images dimensionality channels. We have come up with another data frame which consists of 50 thousand columns crossed over to nine thousand rows. Finally, the CSV file accounts for 4.67 GB storage space. Preprocessing stage also deals with normalization, we have used several normalizing techniques which are listed below The techniques used are : -

1. After CSV Building
   a. Standard Scaler Normalization
   b. Normalization by statistical measures
   c. Min - Max normalization
   d. Deca - Norm
2. Before CSV Building
   a. dividing each image pixel by 255
   b. min-max scaling of pixels

In order to reduce the computation, we have used principal dimensionality reduction, the following section describes the approach used.

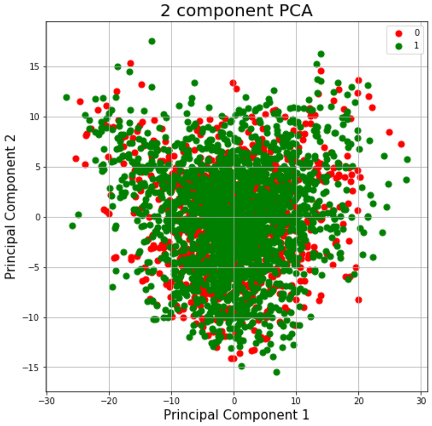
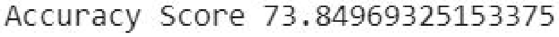


Fig. 5

It can be seen clearly as per the PCA dimensionality approach that both the normal and pneumonia images aren't much separable in two-dimensional spaces. However, it is also an experimental fact that we cannot plot more than three dimensions but on continuous experimentation and evaluation, we have been able to identify the most feasible and computationally efficient dimensional representation which spreads out over 8569 dimensions to find proper variance difference between the two classes.

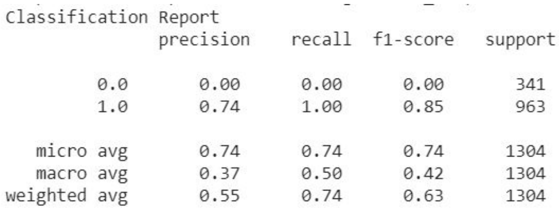
After implementing the Machine Learning algorithm to test our dimensional reduction hypothesis. We have drawn up the confusion matrix, accuracy score, and the Classification Report.



```
Accuracy Score 73.84969325153375
```

Fig. 6

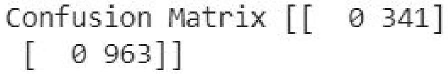
The accuracy seems reasonable for a machine learning algorithm. The classification report discloses the insights of class prediction as well as other parameters like precision, recall, F1 score, etc.



```
Classification Report
              precision    recall  f1-score   support

         0.0       0.00      0.00      0.00       341
         1.0       0.74      1.00      0.85       963

   micro avg       0.74      0.74      0.74      1304
   macro avg       0.37      0.50      0.42      1304
weighted avg       0.55      0.74      0.63      1304
```

Fig. 7

In the above classification report, we can see it conveying the class partiality which the algorithm has shown. The confusion matrix will tentatively show us a nominal prediction for the normal class. .



```
Confusion Matrix [[  0 341]
 [  0 963]]
```

Fig. 8

The above result indicates that even though the dimensionality reduction can be applied on blob segment analysis, the results are highly misleading and cannot be trusted for medical relics. Also, class learning has shown to be a severe problem. Also, it raises questions on the blob segmentation analysis that the blobs detected pertain to which type of blob.

i. Normal
ii. Pneumonia
iii. Any Other (Possibility of some other anomaly existing)

Clustering is used to solve this conundrum as it is the best way to figure which blob segment belongs to which type.

Clustering is a technique used to visualize similar points based on the boundary, statistical measures, and other algorithmic boundaries.

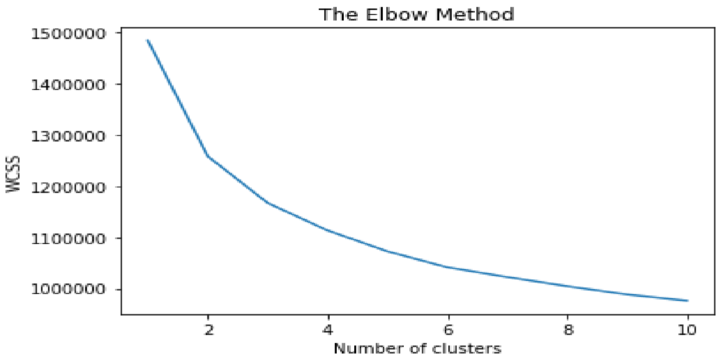


Fig. 9

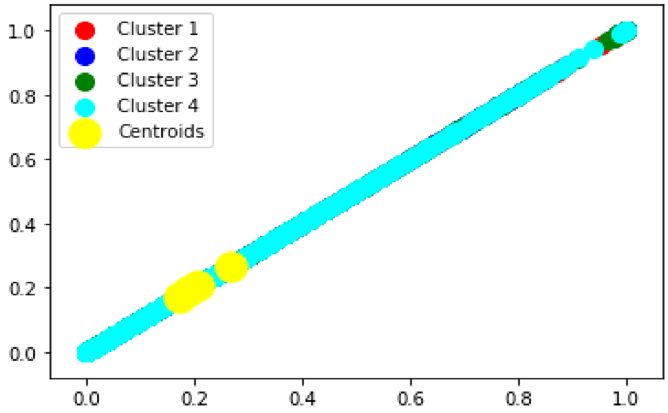
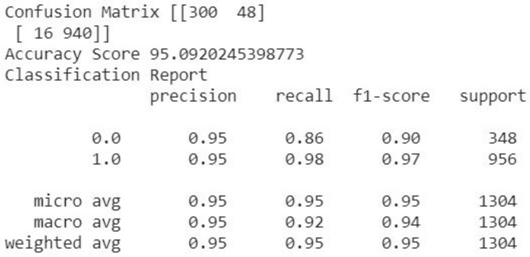


Fig. 10

Again, this seems relatable to the dimensionality PCA plots and also the classification results, the K-Nearest Neighbour clustering centroid showcases a relentless slope wherein all the points lie. So, it is understandable why dimensionality reduction at 8569 was of no use at all. Still, removing all the dimensionality out of the picture would be disastrous to the results.

On further investigation to improve the accuracy and other classification parameters. We have reached 12288 dimensionalities and we can see our results as follows:-

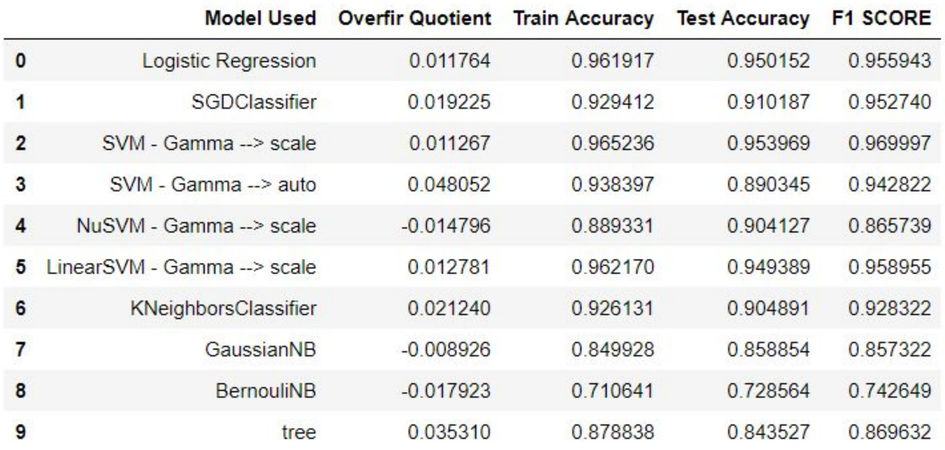


```
Confusion Matrix [[300  48]
 [ 16 940]]
Accuracy Score 95.0920245398773
Classification Report
              precision    recall  f1-score   support

         0.0       0.95      0.86      0.90       348
         1.0       0.95      0.98      0.97       956

   micro avg       0.95      0.95      0.95      1304
   macro avg       0.95      0.92      0.94      1304
weighted avg       0.95      0.95      0.95      1304
```

Fig. 11

Now, we have stuck to these very dimensions not only because of the results but also since further increasing the dimensions; leads to lagging in computational speed, decreasing accuracy and increasing time complexity.

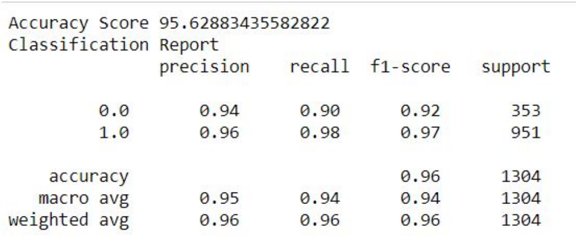
Thus, we have finally trained a number of models on 12288 dimension parameters. Let's have a look at this data frame to do further analysis.



	Model Used	Overfir Quotient	Train Accuracy	Test Accuracy	F1 SCORE
0	Logistic Regression	0.011764	0.961917	0.950152	0.955943
1	SGDClassifier	0.019225	0.929412	0.910187	0.952740
2	SVM - Gamma --> scale	0.011267	0.965236	0.953969	0.969997
3	SVM - Gamma --> auto	0.048052	0.938397	0.890345	0.942822
4	NuSVM - Gamma --> scale	-0.014796	0.889331	0.904127	0.865739
5	LinearSVM - Gamma --> scale	0.012781	0.962170	0.949389	0.958955
6	KNeighborsClassifier	0.021240	0.926131	0.904891	0.928322
7	GaussianNB	-0.008926	0.849928	0.858854	0.857322
8	BernouliNB	-0.017923	0.710641	0.728564	0.742649
9	tree	0.035310	0.878838	0.843527	0.869632

Fig. 12

General algorithms have been used to generalize as well as garner better accuracies. We check with logistic regression and its results.



```
Accuracy Score 95.62883435582822
Classification Report
              precision    recall  f1-score   support

         0.0       0.94      0.90      0.92       353
         1.0       0.96      0.98      0.97       951

    accuracy                           0.96      1304
   macro avg       0.95      0.94      0.94      1304
weighted avg       0.96      0.96      0.96      1304
```

Fig. 13

It is intuitively brilliant to see a simplistic approach algorithm to showcase such good and generic results, The confusion matrix of the above algorithm is given as follows:

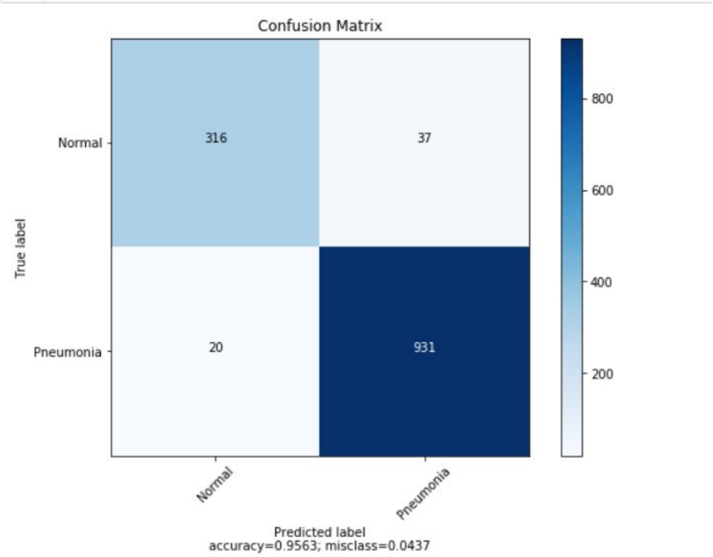
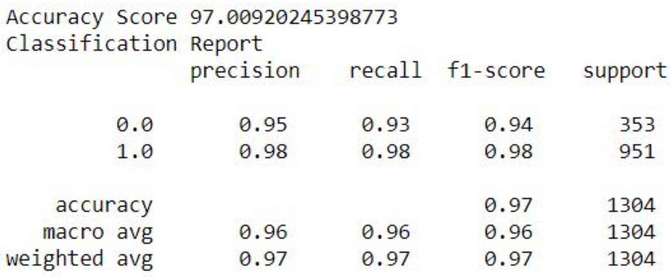


Fig. 14

It showcases that the absence of very less false positives is watered down to 20 and also there is no presence of class partiality any further.

Next, Support Vector Machine(SVM) scaled by gamma. Following is it's classification report:-



```
Accuracy Score 97.00920245398773
Classification Report
              precision    recall  f1-score   support

         0.0       0.95      0.93      0.94       353
         1.0       0.98      0.98      0.98       951

    accuracy                           0.97      1304
   macro avg       0.96      0.96      0.96      1304
weighted avg       0.97      0.97      0.97      1304
```

Fig. 15

This has given an even better result than that of logistic regression. We see an even lower false positive rate than the one in Logistic Regression.

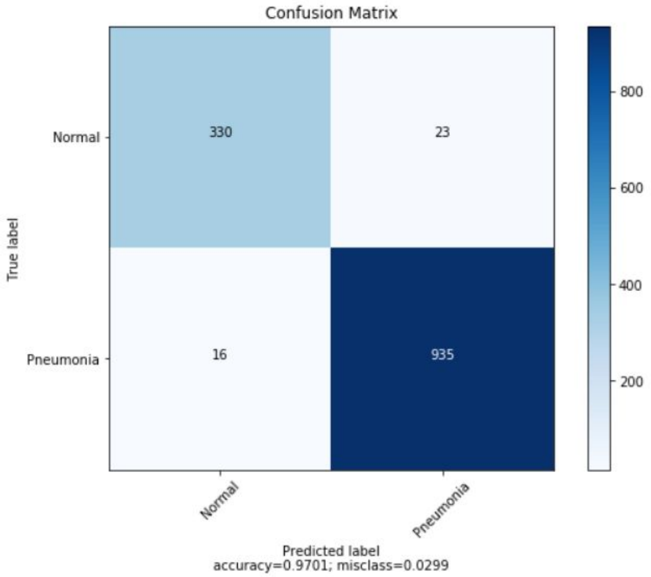
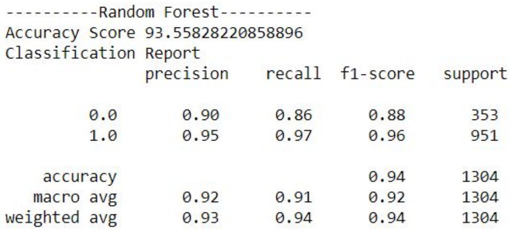


Fig. 16

The false-positive cases are lowered down to 16 and also there is no presence of class partiality further. This is even less than our previous best linear regression.

We would also like to turn to the possibility of gradient boosting algorithms providing better results down; which they have come to showcase in recent years. Their result simulations are listed below for further analysis and understanding.

1. **Random Forest**



```
----------Random Forest----------
Accuracy Score 93.55828220858896
Classification Report
              precision    recall  f1-score   support

         0.0       0.90      0.86      0.88       353
         1.0       0.95      0.97      0.96       951

    accuracy                           0.94      1304
   macro avg       0.92      0.91      0.92      1304
weighted avg       0.93      0.94      0.94      1304
```

Fig. 17

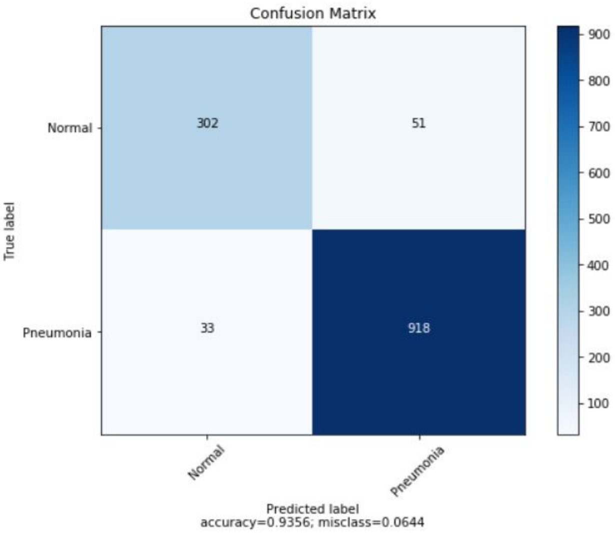
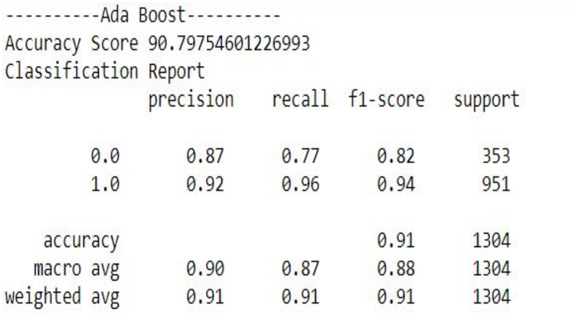


Fig. 18

2. **Ada Boost**



```
----------Ada Boost----------
Accuracy Score 90.79754601226993
Classification Report
              precision    recall  f1-score   support

         0.0       0.87      0.77      0.82       353
         1.0       0.92      0.96      0.94       951

    accuracy                           0.91      1304
   macro avg       0.90      0.87      0.88      1304
weighted avg       0.91      0.91      0.91      1304
```

Fig. 19

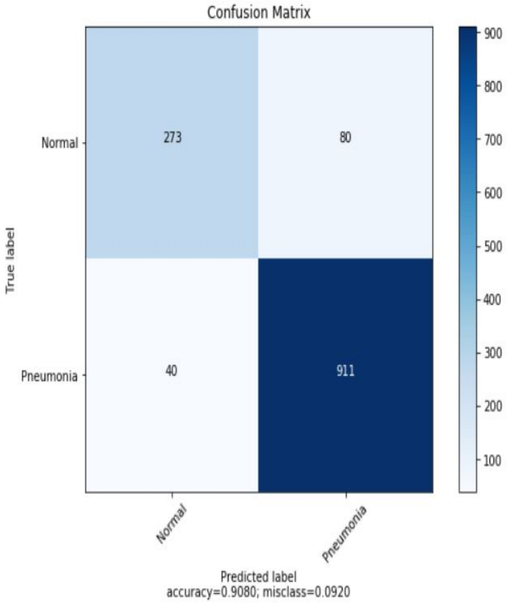
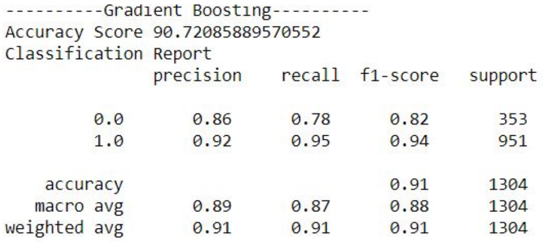


Fig. 20

3. Gradient Boosting



```
----------Gradient Boosting----------
Accuracy Score 90.72085889570552
Classification Report
              precision    recall  f1-score   support

         0.0       0.86      0.78      0.82       353
         1.0       0.92      0.95      0.94       951

    accuracy                           0.91      1304
   macro avg       0.89      0.87      0.88      1304
weighted avg       0.91      0.91      0.91      1304
```

Fig. 21

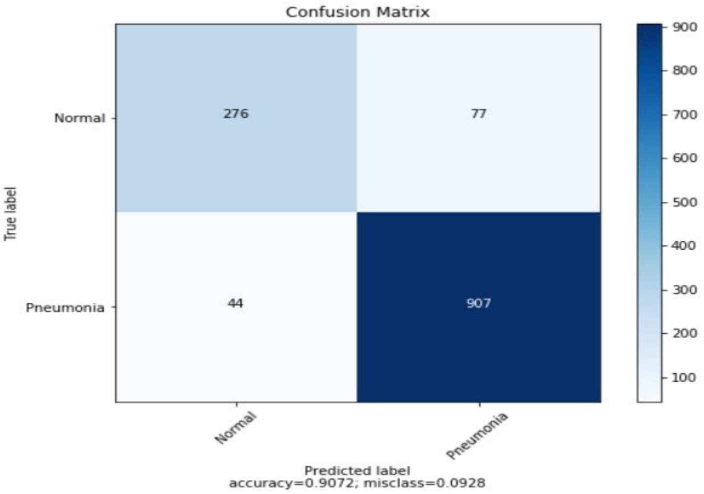


Fig. 22

The gradient boosting algorithms have also shown significant and generic results, the data structure and handling are also the same as described below.

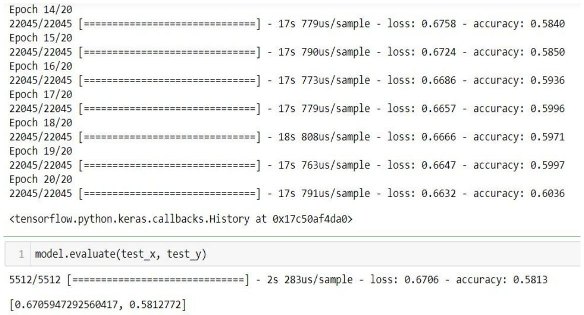
It is important to realize that this complete training and testing has been handled and done in a time period of 29 hours and 37 minutes. The system requirements are the same as described below in the deep learning and further algorithmic section.

b. Transfer Learning and Image Analysis

c.

1. Mobile-net or Google Flow Model

We tried training the data through the Mobile-net/Google Flow model as shown below :



```
Epoch 14/20
22045/22045 [==============================] - 17s 779us/sample - loss: 0.6758 - accuracy: 0.5840
Epoch 15/20
22045/22045 [==============================] - 17s 790us/sample - loss: 0.6724 - accuracy: 0.5850
Epoch 16/20
22045/22045 [==============================] - 17s 773us/sample - loss: 0.6686 - accuracy: 0.5936
Epoch 17/20
22045/22045 [==============================] - 17s 779us/sample - loss: 0.6657 - accuracy: 0.5996
Epoch 18/20
22045/22045 [==============================] - 18s 808us/sample - loss: 0.6666 - accuracy: 0.5971
Epoch 19/20
22045/22045 [==============================] - 17s 763us/sample - loss: 0.6647 - accuracy: 0.5997
Epoch 20/20
22045/22045 [==============================] - 17s 791us/sample - loss: 0.6632 - accuracy: 0.6036

<tensorflow.python.keras.callbacks.History at 0x17c50af4da0>
```

```
model.evaluate(test_x, test_y)
```

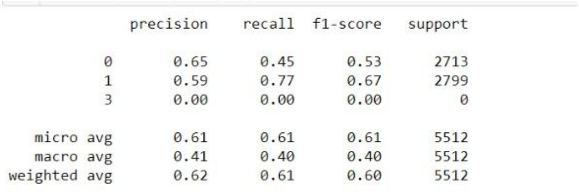
```
5512/5512 [==============================] - 2s 283us/sample - loss: 0.6706 - accuracy: 0.5813

[0.6705947292560417, 0.5812772]
```

Fig. 23

The biggest pitfall of this model is the low accuracy of the provided dataset. The model's low accuracy helps us to lead to the conclusion that this would not be an appropriate solution to the problem presented.

2. VGG - 16

We also implemented the already available VGG-16 model and while it provided greater accuracy than the mobile-net model as can be seen in Fig. 24 below.



```
              precision    recall  f1-score   support

           0       0.65      0.45      0.53      2713
           1       0.59      0.77      0.67      2799
           3       0.00      0.00      0.00         0

   micro avg       0.61      0.61      0.61      5512
   macro avg       0.41      0.40      0.40      5512
weighted avg       0.62      0.61      0.60      5512
```

Fig. 24

Even though the model has an increased accuracy score than the mobile-net model but a better accuracy score can be gained by a specifically designed neural network model which can ensure less variation among the input layer and also the class imbalance is dealt in the model.

d. **Proposed Neural Network and Mode**l

Before the training of the model starts, we realize that the training dataset might cause class imbalance due to a lesser number of images being provided for 'Normal' Label.

To solve this, a **General Adversarial Network** is implemented.

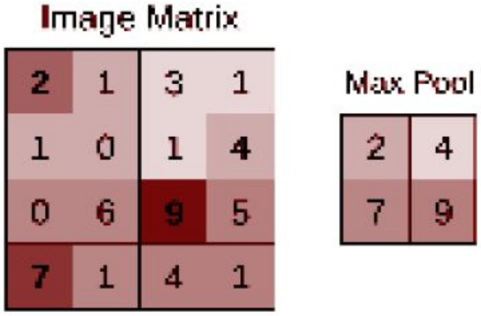
For the first stage of this model, a Convolutional Layer which extracts the features with the help of kernels. These kernels are trained to detect specific features (or Trends) in the image. These kernels move over each part and detects whether the part has the specified resource or not.

If the required feature is present then the function returns a high valued real number otherwise it results in a low valued number being returned.

For a given Two-dimensional image *I*, and a small array, *K* of size h x w (kernel), the convoluted image (*I***K*), is calculated via superimposing the kernel/filter at the top of the image's all possible forms, and recording the sum of the elementary products between the Image *I* and following kernel equation

$$(I * K)_{xy} = \sum_{i=1}^{h} \sum_{j=1}^{w} K_{ij} . I_{x+i-1,y+j-1}$$

The output from this Convolutional Layer is transferred to the pooling layer which is responsible for reducing the spatial size of the feature map, preserving the resources detected into a smaller representation. Although, many alternatives are accessible for pooling. Max Pooling is the most favored. Max Pooling operates by locating the locations in the image that shows the strongest correlation with each resource (the maximum value) are preserved and these values combine to form a smaller space. As shown as the Fig.25 below



2	1	3	1
1	0	1	4
0	6	9	5
7	1	4	1

2	4
7	9

Fig. 25

After going through this process, we progress with **Batch Normalization** as explained in the concepts batch normalization helps with providing a normalized data which ranges from scale 0 to 1 for a data which earlier had a larger range. If in between normalizations are not done there might be one weight dominating others and affecting the results. The layer inputs also, as a result of the normalization, show less variation amongst them.

After Normalizing, the layer inputs which are scaled, are then processed through another Convolutional Neural Network and then through another Max Pooling which again reduces the spatial size of the feature map which is available. Then, we apply **Dropout** because it is a vital step for the enhancement of model which do not overfit , it randomly cancels out nodes in each iteration so that not anyone node is of greater importance than the others to the model, Also so that neither of single input features is heavily weighted, since under each backpropagation iteration the weights are decreased when dropped and failure of output occurs.

Then **Flatten** is used for converting the three-dimensional matrix to the one-dimensional matrix. This is done in order to make the data eligible to be fitted in Artificial neural networks. However, it is of import to note that when training with more than one Convolutional
Layers, flatten must be only used after the last convolutional layer. Once flatten is applied
no more convolutional layers can be employed.

After these 2 dense layers are used. These dense layers directly refer to the Artificial Neural network schema which is 128 nodes in one layer. After the layers are added a Dropout is added and finally, another dense layer is added which has finally 2 nodes. Total Parameters are 331,330 while 331,266 are trainable parameters. For the compilation of the model, we use Sparse Categorical cross-entropy as the loss and 'Adam' as the optimizer.

The **cross-entropy** loss measures the performance of a classification model, with the output being a probability value ranging from 0 to 1.

Hyperparameter Information

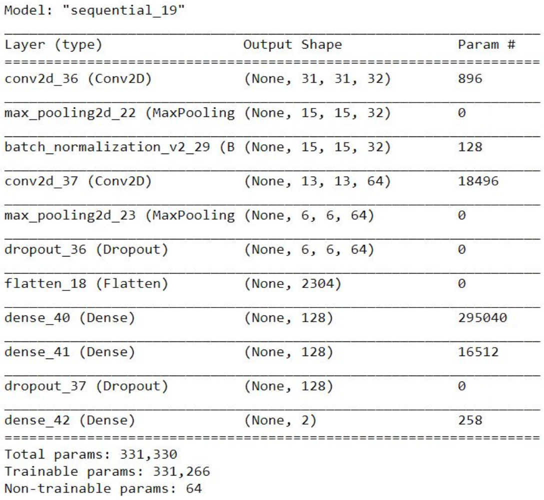
a. All dropouts are 25% that is one fourth so that overfitting can be removed.

b. For regularization, mostly Manhattan distance or L1 regularization is used.

c. The activation function is leaking rectilinear units and rectilinear units.

e. Adam Optimizer function is used to find the global minima and not to be stuck with the local minima.

Model Summary



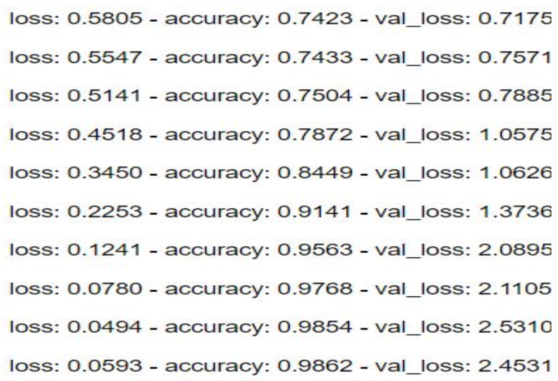
```
Model: "sequential_19"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_36 (Conv2D)           (None, 31, 31, 32)        896
_________________________________________________________________
max_pooling2d_22 (MaxPooling (None, 15, 15, 32)        0
_________________________________________________________________
batch_normalization_v2_29 (B (None, 15, 15, 32)        128
_________________________________________________________________
conv2d_37 (Conv2D)           (None, 13, 13, 64)        18496
_________________________________________________________________
max_pooling2d_23 (MaxPooling (None, 6, 6, 64)          0
_________________________________________________________________
dropout_36 (Dropout)         (None, 6, 6, 64)          0
_________________________________________________________________
flatten_18 (Flatten)         (None, 2304)              0
_________________________________________________________________
dense_40 (Dense)             (None, 128)               295040
_________________________________________________________________
dense_41 (Dense)             (None, 128)               16512
_________________________________________________________________
dropout_37 (Dropout)         (None, 128)               0
_________________________________________________________________
dense_42 (Dense)             (None, 2)                 258
=================================================================
Total params: 331,330
Trainable params: 331,266
Non-trainable params: 64
```

Fig. 26:

The figure provided is a summary of the model we have implemented. It clearly shows the total number of trainable parameters

VII. Results

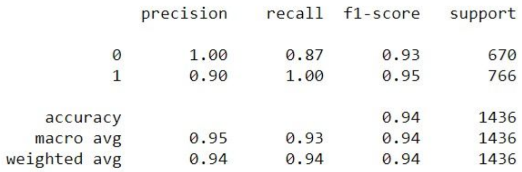
With the model compiled, now it is fitted on the dataset and accuracy of the model by using validation.
The following Fig. 27 shows the training of our model over 10 epochs.



```
loss: 0.5805 - accuracy: 0.7423 - val_loss: 0.7175
loss: 0.5547 - accuracy: 0.7433 - val_loss: 0.7571
loss: 0.5141 - accuracy: 0.7504 - val_loss: 0.7885
loss: 0.4518 - accuracy: 0.7872 - val_loss: 1.0575
loss: 0.3450 - accuracy: 0.8449 - val_loss: 1.0626
loss: 0.2253 - accuracy: 0.9141 - val_loss: 1.3736
loss: 0.1241 - accuracy: 0.9563 - val_loss: 2.0895
loss: 0.0780 - accuracy: 0.9768 - val_loss: 2.1105
loss: 0.0494 - accuracy: 0.9854 - val_loss: 2.5310
loss: 0.0593 - accuracy: 0.9862 - val_loss: 2.4531
```

Fig 27

As we can see the accuracy reaches 98.62%, This is because our model reduces the errors by utilizing the Batch Normalization and Drop Outs at the correct stages of the network. Therefore, we have named this model as 'Phoenix', because as the myth of the phoenix suggests about it rising from the ashes after death, we believe our model arises from the depths of pre-processing errors, models and class imbalances. Our model overcomes all of these and hence provides the most accurate diagnoses while not being overfit.
The outputs of 'Pneumonia' and 'Normal' labels are encoded as 0 and 1 and as we can see the precision for both is quite high which validates our model even further and shows the exactness of our model.
 Now, since we know the Phoenix Model has decent accuracy and is precise enough to be used in Medical Practice. We now compare our activation functions for the layers in our network.
The following figure (Fig.28) is the classification report of our model



```
              precision    recall  f1-score   support

           0       1.00      0.87      0.93       670
           1       0.90      1.00      0.95       766

    accuracy                           0.94      1436
   macro avg       0.95      0.93      0.94      1436
weighted avg       0.94      0.94      0.94      1436
```

Fig 28

In Fig 29 and Fig 30 the plotted graphs showcase 3 activation functions, in comparison to each other plotted between the loss in training of train and validation sets of data respectively

Fig 29

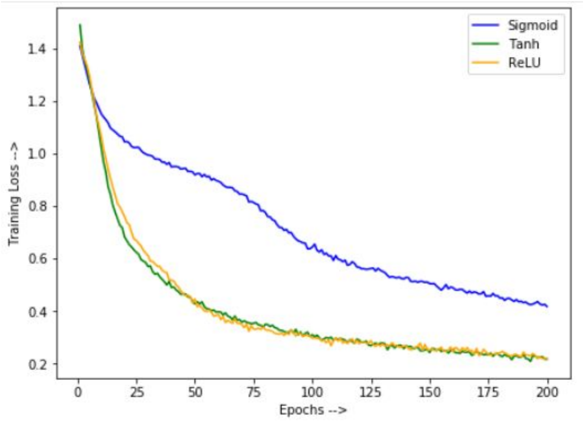
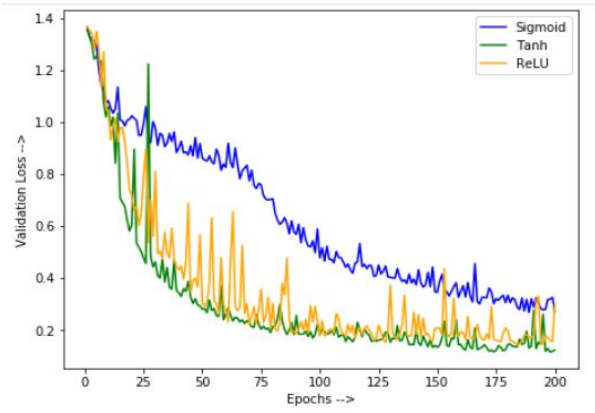


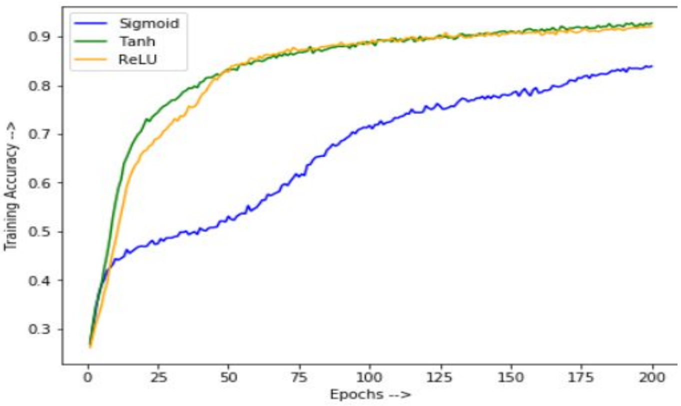
Fig 30

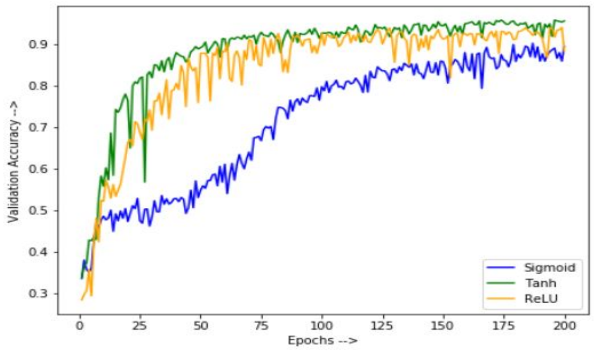


These plots depict clearly that the sigmoid activation function is of no use if the lower number of epochs is to be trained while the losses almost approach zero with the Tanh activation function which again points us to the conceptual error that losses cannot be almost zero since in real-time.

Similar conclusions as above can be made from the plots 3 and 4 of activation functions on accuracy versus epoch graphs on both training and validation sets.

Fig 31





Next, we move on and find the best optimizer for the dataset. We have used 'Adam' optimizer in our model

Again, we plot for training and validation sets loss against the epochs trained. In Fig 33 the respective graphs are present for four optimizers.

Fig 32

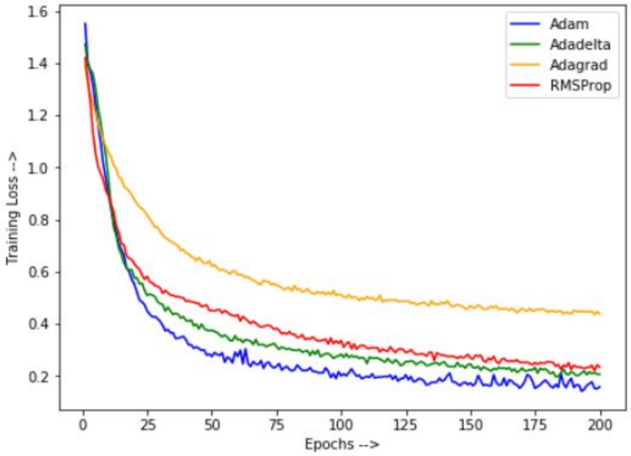
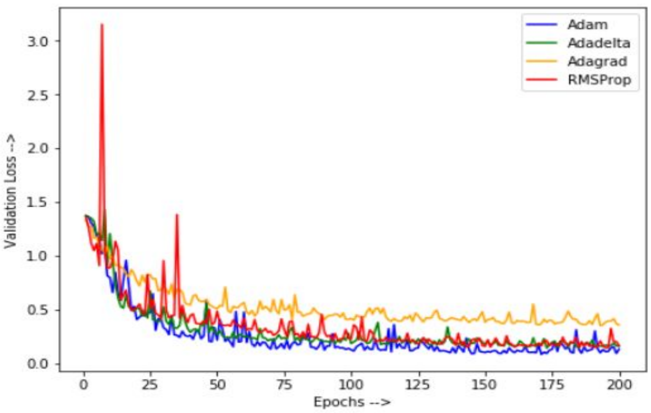


Fig 33



These plots show that 'Adam' as an optimizer has the lowest losses amongst all the other optimizers which in turn shows that 'Adam' is the best choice for the optimizer.

We see a similar trend in the accuracy v/s Epochs plots 6 and 7 for the optimizers based on training and validation set.

Fig 32

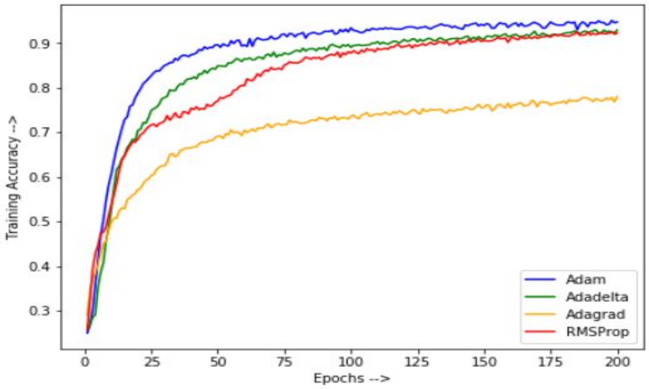
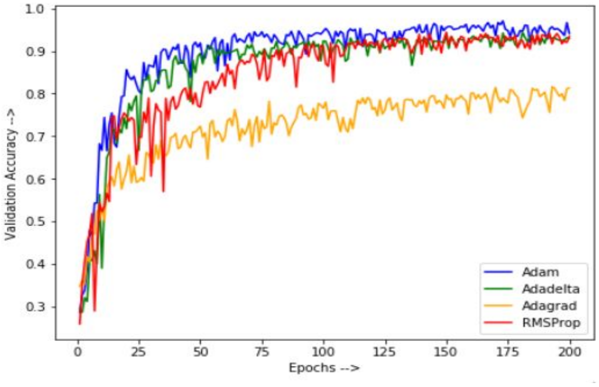


Fig 33



Further, we plot the difference between our train and test accuracy and we showcase that the Phoenix model has high accuracy and comparatively less loss. This is depicted in the following Figs 8 and 9.

Fig 34

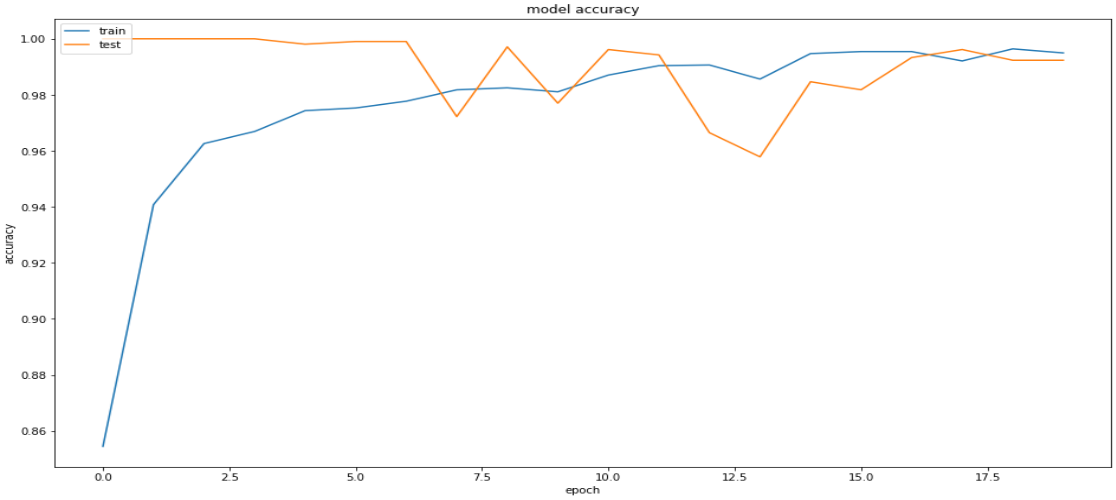
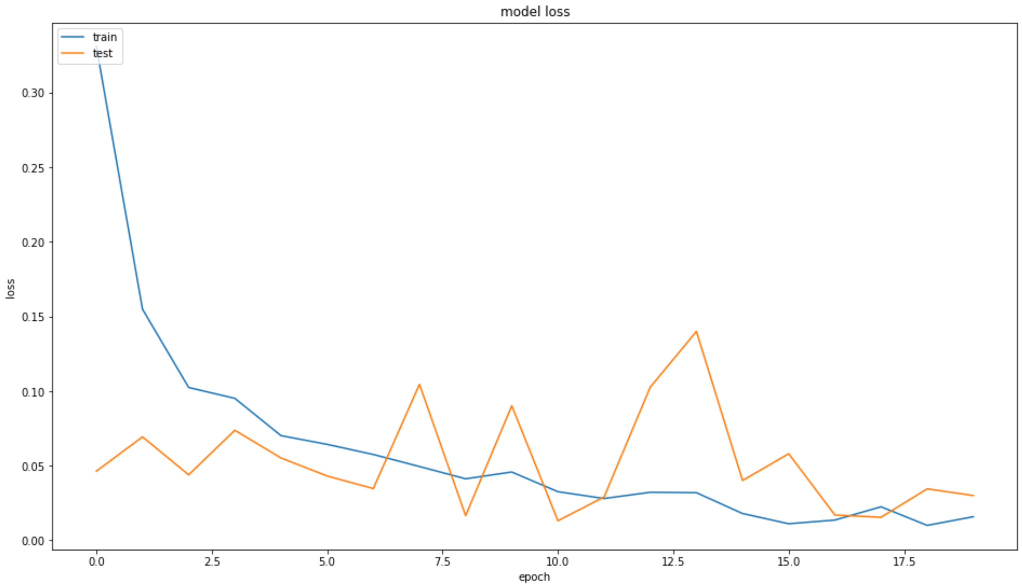


Fig 35



As is clear from these graphs the model's accuracy is high and loss is relatively low. It is due to various factors which we have included in our model such as

a. GANs: which help us address the problems of class imbalance due to the dataset having it inherently
b. Batch Normalization so that layer inputs have a varied distribution.
c. Max Pooling after each Convolutional Neural Network layer.
d. Usage of 'relu' as activation function and 'Adam' as an optimizer.

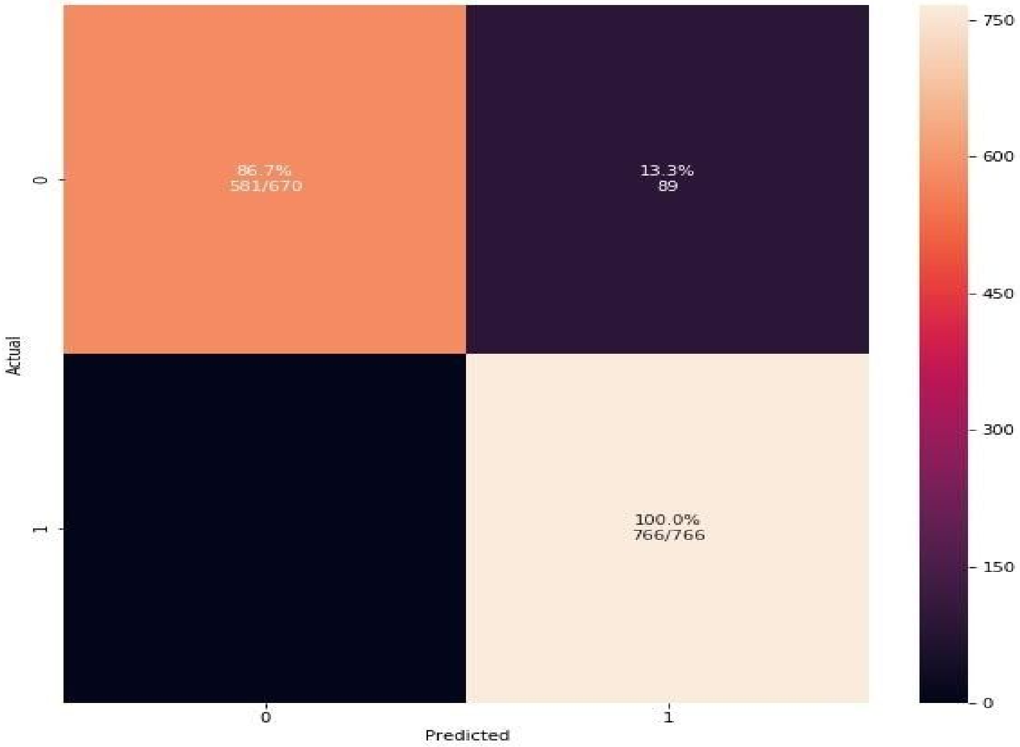
Confusion Matrix

The confusion matrix is an array that contains correct and incorrect predictions of the algorithm and the actual situation.

- True Positive: Number of people who actually have pneumonia according to the algorithm.
- False Negative: Number of people who are actually with pneumonia but categorized as healthy according to the algorithm.
- False Positive: Number of people who are actually healthy, but categorized as pneumonia, according to the algorithm.
- True Negative: Number of people who are really healthy and categorized as healthy according to the algorithm

Fig 36 depicts the confusion matrix for our model Phoenix which shows that the True Negatives and True Positives are classified almost correctly according to the provided.

Fig 36



X. Conclusion

In this paper ‘Phoenix’ Model has been proposed for diagnosing Pneumonia through Chest X-rays and computer vision with a neural network model being trained. We have achieved 98.6 % accuracy on the (Kermany) dataset on which earlier researches have proposed a maximum of average accuracy of 95.30% (et al A. A. Saraiva) and 92.8% (et al Kermany). We believe that our model solves the predicament we found ourselves in earlier while implementing Mobile-net and VGG-16 quite efficiently and has quite a significant rise in terms of accuracy and precision.
In comparison to the machine learning algorithms which tested SVC 97.001% and Logistic regression 95.643%, our model(Phoenix) is better suited for the purposes of the medical industry.

Both the models showcase false positive error of amount 16 and 20 per test sample input cases respectively. As described earlier our version of evaluating the post matrice results was to decrease the false positive so that medical robustness can be achieved. Further strengthening that note we are able to deduce about 0 true negatives using our phoenix model on the given input test sequence. Finally with evidence from both machine learning and deep learning relics of data analytics. Further enhancement and decreasing the false positives and increase of accuracy as well as other parameters of evaluation compared to the earlier models.

References:

[1] Saraiva, Arata & Ferreira, N & Lopes de Sousa, Luciano & Carvalho da Costa, Nator & Sousa, José & Santos, D & Valente, Antonio & Soares, Salviano. (2019). Classification of Images of Childhood Pneumonia using Convolutional Neural Networks. 112-119. 10.5220/0007404301120119.

[2] Barrientos, R., Roman-Gonzalez, A., Barrientos, F., Solis, L., Correa, M., Pajuelo, M., Anticona, C., Lavarello, R., Castaneda, B., Oberhelman, R., Checkley, W.,˜ Gilman, R. H., and Zimic, M. (2016). Automatic detection of pneumonia analyzing ultrasound digital images. In 2016 IEEE 36th Central American and Panama Convention (CONCAPAN XXXVI), pages 1– 4.

[3] Khobragade, S., Tiwari, A., Patil, C. Y., and Narke, V. (2016). Automatic detection of major lung diseases using chest radiographs and classification by feedforward artificial neural network. In 2016 IEEE 1st International Conference on Power Electronics, Intelligent Control and Energy Systems (ICPEICES), pages 1–5.

[4] Rodrigues, M. B., Nobrega, R. V. M. D., Alves, S. S. A.,´ Filho, P. P. R., Duarte, J. B. F., Sangaiah, A. K., and Albuquerque, V. H. C. D. (2018). Health of things algorithms for malignancy level classification of lung nodules. IEEE Access, 6:18592–18601.

[5] Predictive Analysis: https://in.mathworks.com/discovery/predictive-analytics.html

[6] Pneumonia Deaths under the age of five UNICEF: https://www.unicef.org/health/index_91917.html

[7] General Adversarial Nets(2014: Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio in Conference on Advances in neural information processing systems

[8] Guan Sheng Shu , Weiqing Liu , Xiaojie Zheng , Jing Li IF-CNN : Image Aware Inference Ling Shao , Transfer Learning for Visual Categorization : A survey , 2015

[9] Simeng Yue , Seiichi Ozawa , A sequential Multi task learning network with metric based knowledge transfer , 2012

[10] Hai yang Jia , Juan Chen Shin Nowa , He Long Yu Dayou Liu , Soil Fertility with Bayesian Network of Transfer Learning ,2010

[11] Toshiaki Takano , Haruhiko Takase , Hiroharu Kawanaka , Shinji Tsuruoka , Transfer learning for reinforcement learning in the same transition model quick approach and preferential exploration , 2011 12. 12. 10th International conference on Machine Learning and Application and Workshops , Year 2011 Volume 1

[12] Y.M. Hirimuthugoda , Artificial Intelligence Based Approach for Determination of Haematologic Diseases , 2009

[13] Khushal Sethi , Vivek Parmar , Manan Suri Low power Hardware Based Deep Learning Diagnostics Support Case Study , 2018

[14] Aishah khan , Ashraf Khalil , Hassan Hajjdiab Mobile Microscopic Device to detect prasitical Cell related diseases using machine learning

[15] Shoji Kido , Detection of lung abnormality by use of Convolutional Neural Networks and Region CNN's , 2018

[16] Hussain Ketout , MVN_CNN and UBN_CNN for endocardial edge detection , 2011