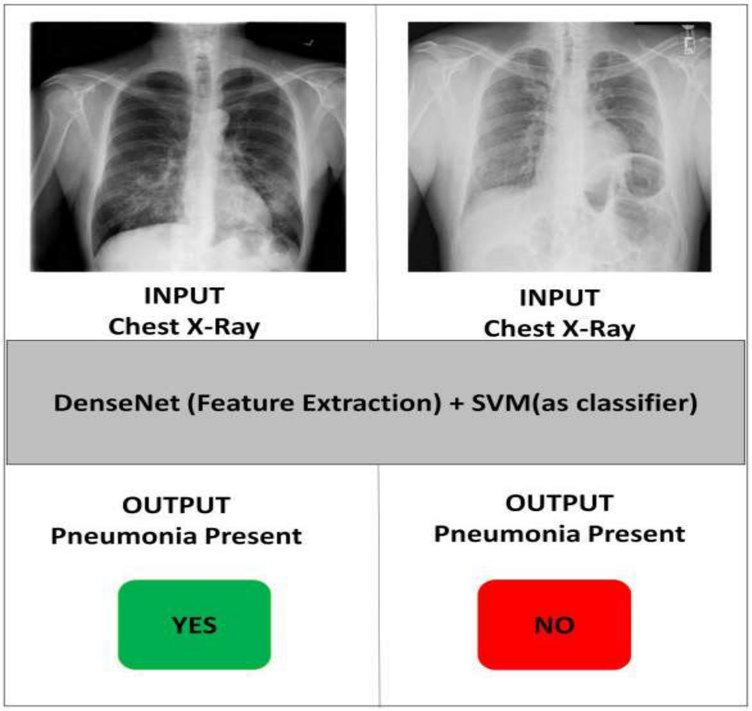
CHEST X-RAY IMAGES FOR PNEUMONIA DETECTION

Mr. Abhimanyu Singh (11800740)
Department of Computer Science & Engineering
Lovely Professional University, Phagwara
abhimanyusinghchauhan98@gmail.com

Mr. Sanjay Kumar Singh (15745)
Department of Computer Science & Engineering
Lovely Professional University, Phagwara

INTRODUCTION –

The risk of pneumonia is immense for many, especially in developing nations where billions face energy poverty and rely on polluting forms of energy. The WHO estimates that over 4 million premature deaths occur annually from household air pollution-related diseases including pneumonia. Over 150 million people get infected with pneumonia on an annual basis especially children under 5 years old. In such regions, the problem can be further aggravated due to the dearth of medical resources and personnel. For example, in Africa's 57 nations, a gap of 2.3 million doctors and nurses exists. For these populations, accurate and fast diagnosis means everything. It can guarantee timely access to treatment and save much needed time and money for those already experiencing poverty. Over the recent years, Computer Aided Designs (CAD) have become the major research domain in machine learning. The subsisting CAD systems have already been proved to facilitate the medical area primarily in detection of breast cancer, mammograms, lung nodules etc. In the procedure of employing Machine Learning (ML) techniques to medical images, significant features are of uppermost importance. For this reason, most of the previous algorithms used hand crafted features for developing CAD systems based on examining images. However, the hand-crafted features with limitations varying according to tasks were not capable of supplying much meaningful features. Employment of Deep Learning (DL) models particularly Convolutional Neural Networks (CNNs) revealed their self-potential of extracting useful features in image classification tasks. This process of feature-extraction demands transfer learning methods where pre-trained CNN models learn the generic features on largescale datasets like ImageNet which are later on transferred to the required task. Availableness of pre-trained CNN models like DenseNet highly aid in procedure of significant feature extraction. In addition, the classification used with high-rich extracted features exhibit improved performance in classifying images. Chest screening subroutines which are mainly used for sensing lung nodules can also be used to diagnose other illnesses such as pneumonia, effusion, cardiomegaly etc. Among these, pneumonia is an infectious and deadly disease which strikes over millions of people, majorly those who are aged above 65 and suffering from chronic diseases like asthma or diabetes. In the procedure of diagnosing pneumonia, chest X-Rays are considered as the most effective method to determine the extent and location of the septic region in the lungs. However, examining chest radio-graphs is not a leisurely task for radiotherapists. In chest X-ray images, appearance of pneumonia can be hazy and can be misapprehended with other diagnoses. The evaluation of chest X-Ray specifically in case of Pneumonia can be misleading because many other problems like congestive heart failure, lung scarring etc. can mimic a Pneumonia. This is the main reason behind the misclassification of the X-ray images in the dataset.



DATASET DESCRIPTION -

The dataset used is ChestX-ray14 released by Wang et al. (2017) also publicly available on the Kaggle [21] platform which consists of 112,120 frontal chest X-ray images from 30,085 patients. Each radiographic image in the dataset is labeled with one or more out of different 14 thoracic diseases. These labels were concluded through Natural Language Processing (NLP) by text-mining disease-classification from the associated radiological reports and are expected to be more than 90% accurate. For the sake of this work, following the approaches from the past, we treat the labels as ground truth for the purpose of pneumonia detection. Prior to the release of this dataset, the largest publicly available dataset of chest radio-graphs was Openi which consisted of roughly 4,143 X-ray images. All the radio-graph images in the dataset are of 1024 by 1024 resolution. Out of these 112,120 images, 1431 images are found to be labeled with pneumonia. In order to balance the dataset for binary classification, 1431 normal X-ray images (labeled with ’No Findings’) have been selected from the dataset. Altogether, the final dataset used for the classification task is the subset of the original dataset which consists of 1431 positive image samples (images labeled with ’Pneumonia’) and 1431 negative image samples (images labeled with ’No Findings’). After that, the dataset was divided into two parts out of which for the testing, 573 images were randomly selected from the whole final dataset. The images were downscaled from 1024 by 1024 resolution to 224 by 224 resolution before they were given input to the network.

METHODOLOGY OF PROPOSED MODEL –

This section deals with the detailed description of the applied methodology. The proposed pneumonia detection system using the ’Densely Connected Convolutional Neural Network’ (DenseNet-169) is described in Figure 2. The architecture of the proposed model has been divided into three different stages - the preprocessing stage, the feature-extraction stage and the classification stage.

A. The Pre-Processing Stage: -

The primary goal of using Convolutional Neural Network in most of the image classification tasks is to reduce the computational complexity of the model which is likely to increase if the input are images. The original 3-channel images were resized from 1024×1024 into 224×224 pixels to reduce the heavy computation and for faster processing. All of the further techniques have been applied over these downsized images.

B. The Feature-Extraction Stage: -

Although, the features were extracted with different variants of pre-trained CNN models the statistical results obtained proposed DenseNet-169 as the optimal model for the feature extraction stage. Therefore, this stage deals with the description of DenseNet-169 model architecture and its contribution in feature extraction.

C. The Classification Stage: -

After feature extraction, different classifiers such as Random Forest, Support Vector Machine etc. were used for the classification task. But the best results were found to be attained when Support vector Machine was used as classifier for the problem. So, in the best proposed model features extracted from DenseNet-169 were used with SVM classifier to accomplish better results. The description of the parameters and Kernel used with SVM is as follows: Let us suppose a set of training data as (x1,y1),(x2,y2).......(xn,yn) and the data needs to be separated into two set of classes where x_i ε F^d is the feature vector and y^i ε (0,1) represents the label class. A Support Vector machine used for binary classification is able to find the best hyperplane for the above training data presented i.e the one with the maximum margin between the classes and is capable of separating the data points of one class with the other. The performance of SVM highly depends on the selection of the kernel and parameters. We used the Gaussian ’radial basis function’ kernel (rbf). The gamma and C parameters of RBF kernel highly affects the performance of SVM. Intuitively, the gamma parameter is used to define amount of influence that a single training example should go to in which lesser value implies ’far’ and larger value implies ’close’. So the gamma parameter shows the inverse of radius of the influence of samples that were selected as support vectors by the model. On the other hand, the C parameter compensates the misclassification of training samples. A low C provides a smooth surface where as a high C tries to classify all training samples correctly by providing the model exemption to select more samples as support vectors.

WORK DONE THROUGH CODE-

- **Load libraries:**

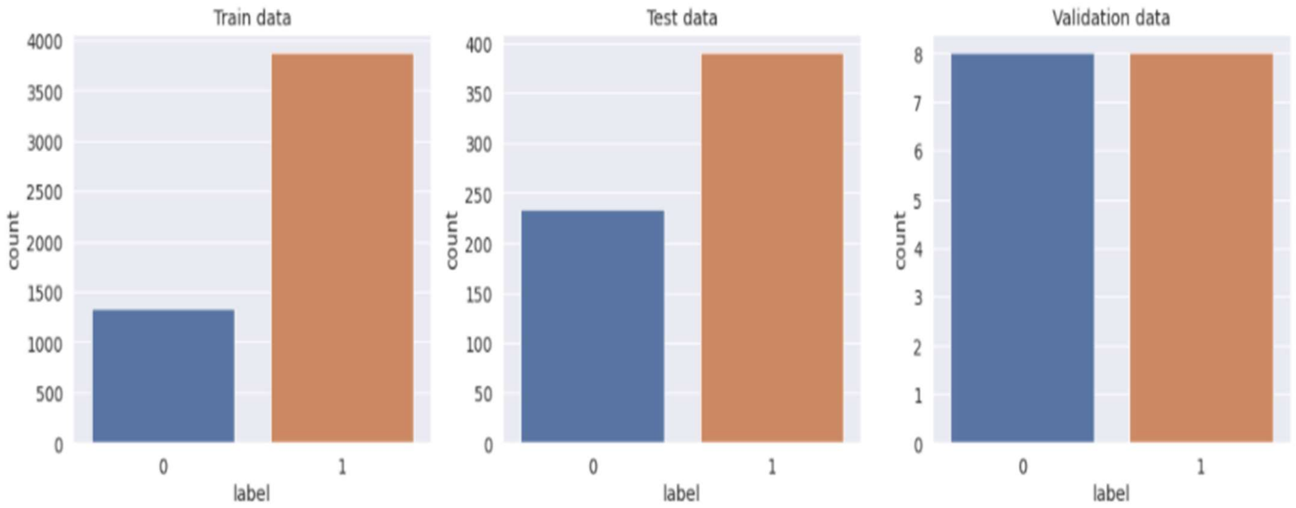
We will use standard "data science" libraries including numpy, pandas, tensorflow, matplotlib and seaborn. Some additional libraries like glob and cv2 are used.

- **Load data:**

Our data consists of 3 folders - train, test, val. Train & test are used for modeling, validation will be used to check performance of model. Size of validation set is very small (16 cases). We will also create data frames so we can visualize distribution of cases in each set. This kind of load may not work well for large dataset as you will run out of memory. Data generator with flow from directory could be used, but it's really slow.

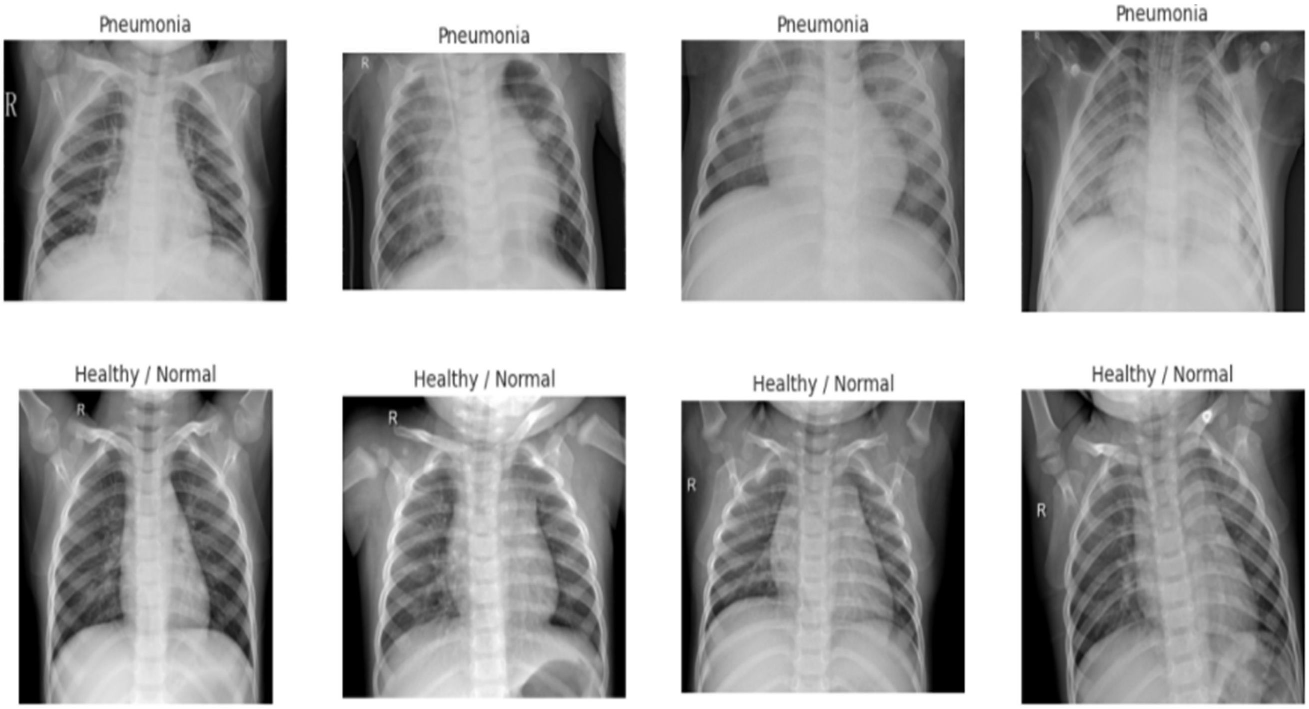
- **Visualize distribution of cases**

Now let's draw how are our datasets imbalanced. We will find out, that training dataset is highly imbalanced, testing dataset is slightly imbalanced and validation dataset is balanced.



- **Few samples of each case:**

In this step we simply want to see few cases of pneumonia and few cases of healthy people. For uninterested person it may not be easy to identify pneumonia on image. We will also notice that images does not have same dimension and must be rescaled to same width & height.



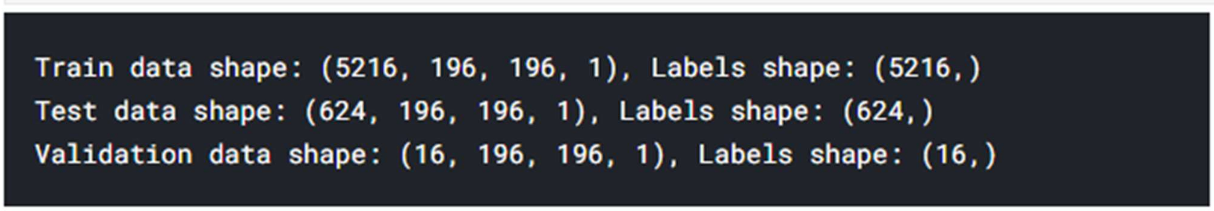
- **Data Preprocessing:**

In following section, we are going to prepare our data for modeling. Down-size of this technique is that you may run out of memory very quickly... but it's really fast!

To make it simple, define 2 functions.

- **Process data** - load image, resize it, convert to grayscale, normalize and reshape to dimension required for tensorflow.
- **Compose dataset** - loop through images, generating 2 numpy arrays. First contains image itself as matrix, second contains label.

Using our functions let's prepare our train, test and validation arrays from dataframes.



```
Train data shape: (5216, 196, 196, 1), Labels shape: (5216,)
Test data shape: (624, 196, 196, 1), Labels shape: (624,)
Validation data shape: (16, 196, 196, 1), Labels shape: (16,)
```

Image augmentation is very important to make our model robust to unseen data. It takes each image and modify it slightly so simply said in each epoch, different image (generated from same) is sent to model for training.

It may be redundant step, but we are going to convert our 1D array of target labels into 2D array, changing classification from binary to categorical. During my tests, binary classification was slower and less accurate then categorical.

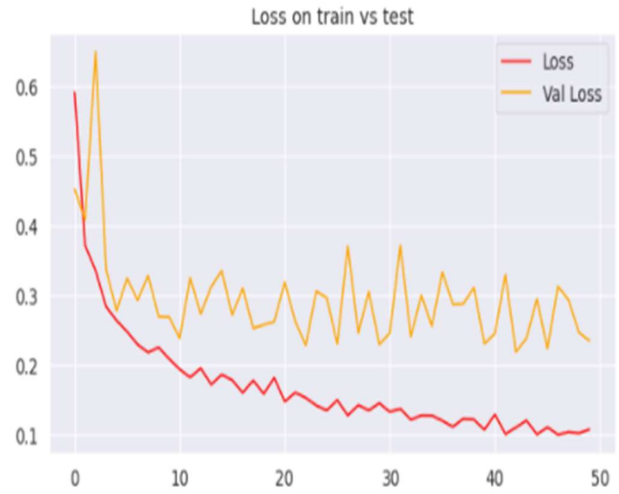
- **Modelling:**

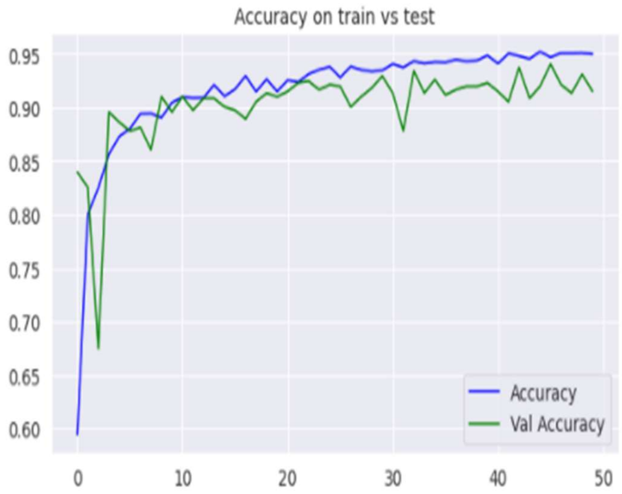
Simple sequential model is used, starting with 2 convolutional networks of kernel size (7,7) and max pooling with pool size (3,3), followed by 2 convolutional networks of kernel size (7,7) and same pool size and finalized by several repeating sets of 2 convolutional networks of kernel size (3,3) with max pooling and pool size (2,2)

Model bit slower, but getting above 92% accuracy that's great result!

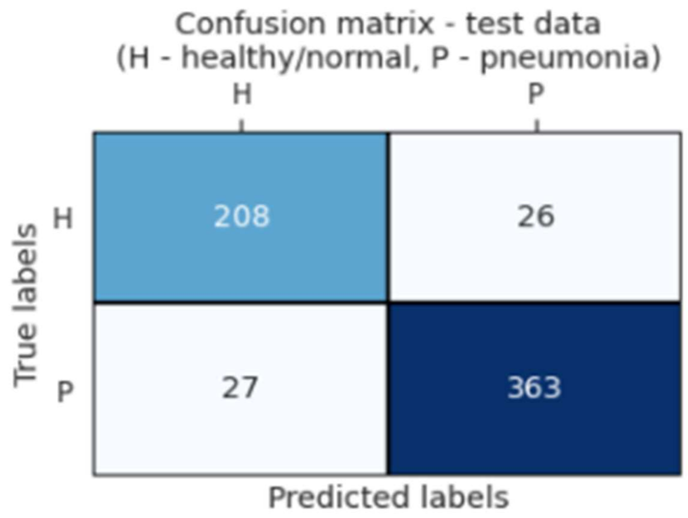
- **Evaluation:**

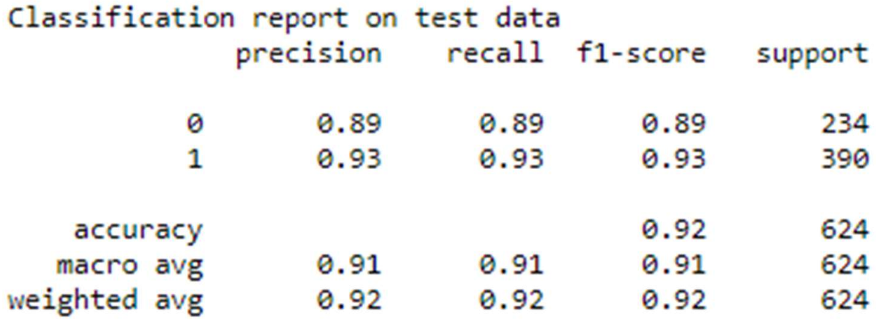
First, we will quickly check evolution of loss and accuracy over epochs and then draw confusion matrix on test data. Then how our validation set (16 cases) will work with trained model and compare real vs predicted label. loss on train vs test data:





Result:





```
Classification report on test data
              precision    recall  f1-score   support

           0       0.89      0.89      0.89       234
           1       0.93      0.93      0.93       390

    accuracy                           0.92       624
   macro avg       0.91      0.91      0.91       624
weighted avg       0.92      0.92      0.92       624
```