Classification and prediction of Common Thorax Diseases.

01FB15ECS213	**Praveen C Naik**
01FB15ECS198	**Omkar D**
01FB15ECS209	**Prajwal B**
01FB15ECS196	**Nithin G**

Table of contents :

Machine learning techniques :

Supervised Machine Learning Algorithms:

Machine learning algorithms that make predictions on given set of samples. Supervised machine learning algorithm searches for patterns within the value labels assigned to data points.
The majority of practical machine learning uses supervised learning.

Supervised learning is where you have input variables (x) and an output variable (Y) and you use an algorithm to learn the mapping function from the input to the output.

$$Y = f(X)$$

The goal is to approximate the mapping function so well that when you have new input data (x) that you can predict the output variables (Y) for that data.

It is called supervised learning because the process of an algorithm learning from the training dataset can be thought of as a teacher supervising the learning process. We know the correct answers, the algorithm iteratively makes predictions on the training data and is corrected by the teacher. Learning stops when the algorithm achieves an acceptable level of performance.

Supervised learning problems can be further grouped into regression and classification problems.

- Classification: A classification problem is when the output variable is a category, such as “red” or “blue” or “disease” and “no disease”.
- Regression: A regression problem is when the output variable is a real value, such as “dollars” or “weight”.

Some common types of problems built on top of classification and regression include recommendation and time series prediction respectively.

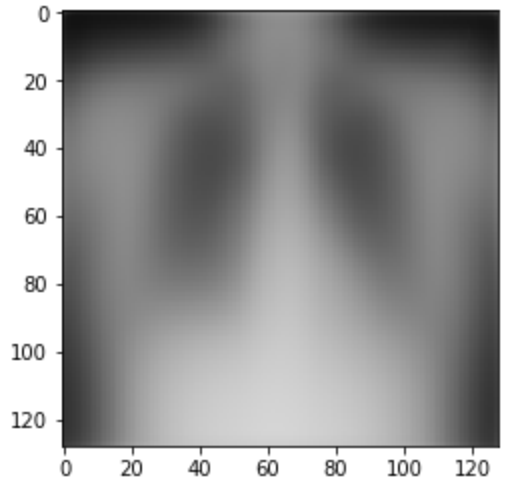
Dataset: NIH Xray dataset

1. Chest X-ray dataset comprises 112,120 frontal view X-ray images of 30,805 unique patients with the text mined fourteen disease image labels (where each image can have multi labels), mined from the associated radiological reports using natural language processing. Fourteen common thoracic pathologies include Atelectasis, Consolidation, Infiltration, Pneumothorax, Edema, Emphysema, Fibrosis, Effusion, Pneumonia, Pleural_thickening, Cardiomegaly, Nodule, Mass and Hernia, which is an extension of the 8 common disease patterns listed in CVPR 2017 paper. The text-mined disease labels are expected to have accuracy >90%.
2. **Source :** https://nihcc.app.box.com/v/ChestXray-NIHCC
3. **Instance count :** 18,577
4. **Attribute list** : Image Index, Finding Labels, #, Patient ID, Patient Age, Patient Gender, View Position, Original Image Size and Original Image, Pixel Spacing
   But chosen attributes are #, Patient Age, Gender, View position and image name.
5. **Attribute count : 5**
6. **Class count: 14**
7. **Class list :**

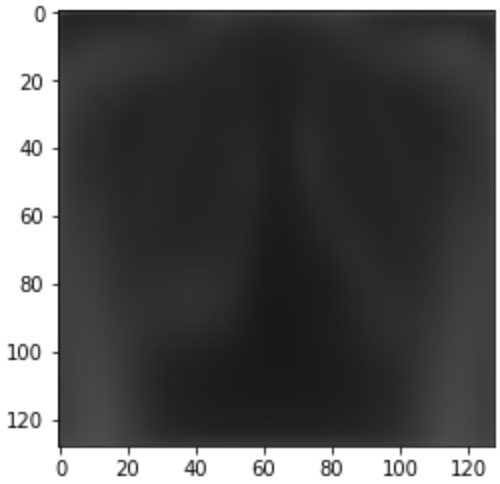
Atelectasis	Emphysema
Consolidation	Fibrosis
Infiltration	Effusion
Pneumothorax	Pneumonia
Edema	Pleural_thickening
Mass	Cardiomegaly
Hernia	Nodule

8. **Preprocessing of Data** : It is not possible to view the entire 18000 images at once. Thus we use the mean image to observe an average of the entire dataset. The mean image is calculated by taking the mean values for each pixel across all training examples. The image roughly represents the thorax.

Mean

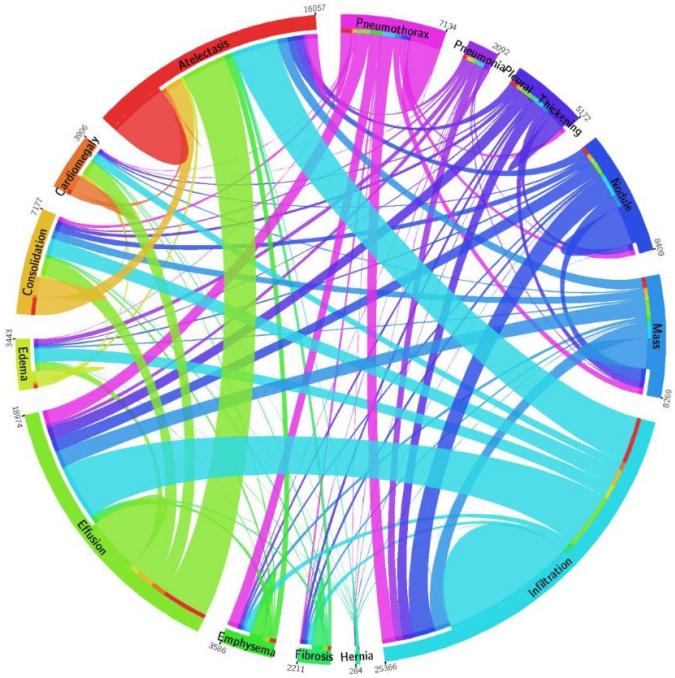


Variance



This image lets us conclude that all the thoraxes are somewhat aligned to the center and are of comparable size. Subtracting the image from the original dataset we get the normalized dataset. Normalized dataset contains images with darker thoracic cavity to create slightly darker contrast.

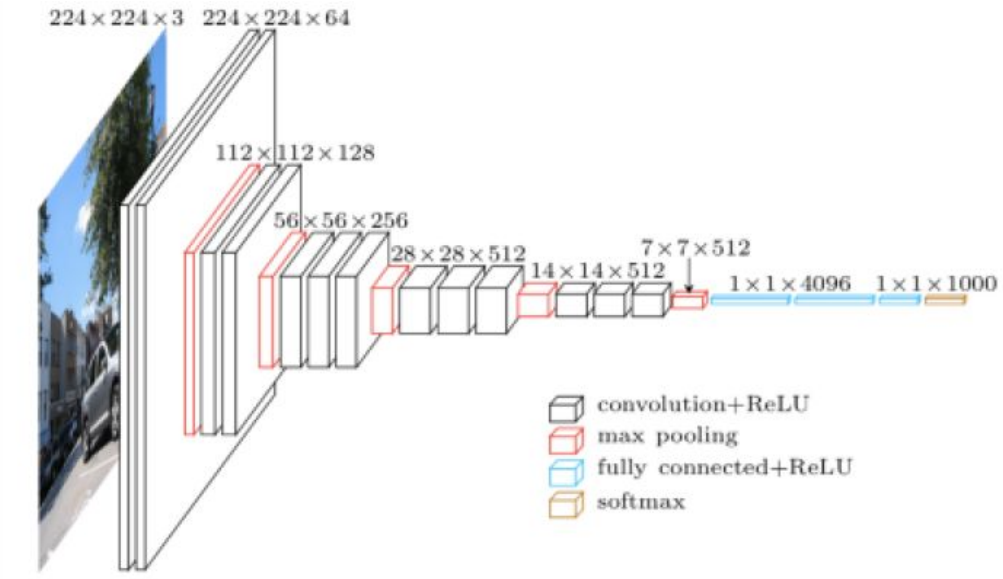
Dataset class dependency graph:



Design of the model

- Transfer learning is employed to train the model. As VVG16 is one of the best classifier net we are using it to train the dataset for disease classification. VGG16 consists of 16 layer deep neural network weights. the model expects images as input with the size 224 x 224 pixels with 3 channels (e.g. color).
- Using numpy arrays we convert the images into an list of array of dimensions (N,224,224,3).Where N is the size of the dataset. On top of the VVG16 net, added(appended) one dense layer and one output softmax layer with 14 tensors(Which is the number of classes).
- Totally there are 23,107,407(23M parameters) trainable tensors.

- **Architecture of the model**



- **Code of the model**

```
base_model = applications.VGG16(weights='imagenet',
include_top=False, input_shape=(IMG_SIZE, IMG_SIZE, 3))
add_model = Sequential()
add_model.add(Flatten(input_shape=base_model.output_shape[1:]))
add_model.add(Dense(256, activation='relu'))
add_model.add(Dense(y_train.shape[1], activation='softmax'))
model = Model(inputs=base_model.input,
outputs=add_model(base_model.output))
```

```
model.compile(loss='categorical_crossentropy',
optimizer=optimizers.SGD(lr=1e-4, momentum=0.9),
            metrics=['accuracy'])
model.summary()
batch_size = datasetSize / 150
epochs = 150
train_datagen = ImageDataGenerator(
      rotation_range=30,
      width_shift_range=0.1,
      height_shift_range=0.1,
      horizontal_flip=True)
train_datagen.fit(x_train)
history = model.fit_generator(
      train_datagen.flow(x_train, y_train, batch_size=batch_size),
      steps_per_epoch=x_train.shape[0] // batch_size,
      epochs=epochs
)
```

VGG16 net overview :

input_1 (InputLayer)	(None, 224, 224, 3)	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
flatten (Flatten)	(None, 25088)	0
fc1 (Dense)	(None, 4096)	102764544
fc2 (Dense)	(None, 4096)	16781312
predictions (Dense)	(None, 1000)	4097000

Total params: 138,357,544
Trainable params: 138,357,544
Non-trainable params: 0

Results

The model training constraints :

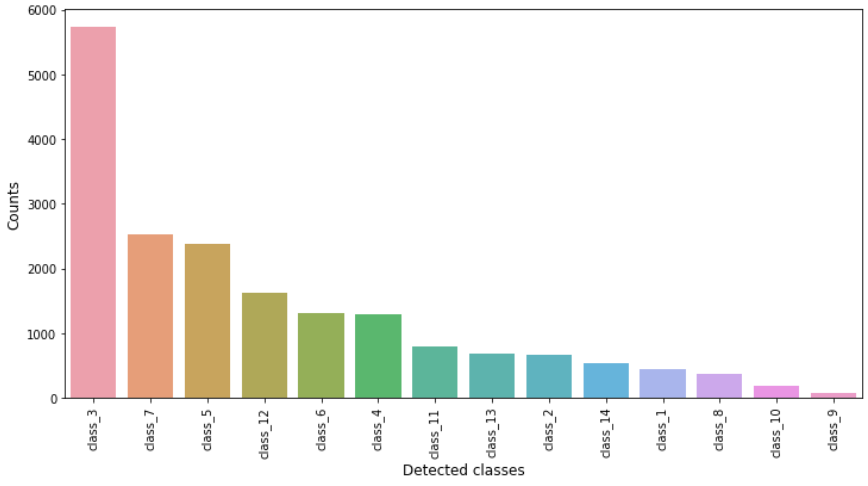
CPU specifications	:	16GB i7 processor
GPU specifications	:	Nvidia GTX GeForce 1050 model 4GB
Trainable parameters	:	23 Million
Image dataset	:	18,577 (18K)
Test Dataset	:	1000 (Images of known classes)
Epochs of training	:	150
Batch size	:	120 images per epoch
Deep learning technique	:	Transfer Learning with added layers
Neural Network	:	VGG16 from Keras
Durations of training	:	avg 89 seconds per epoch. Totally 3Hr 45 minutes

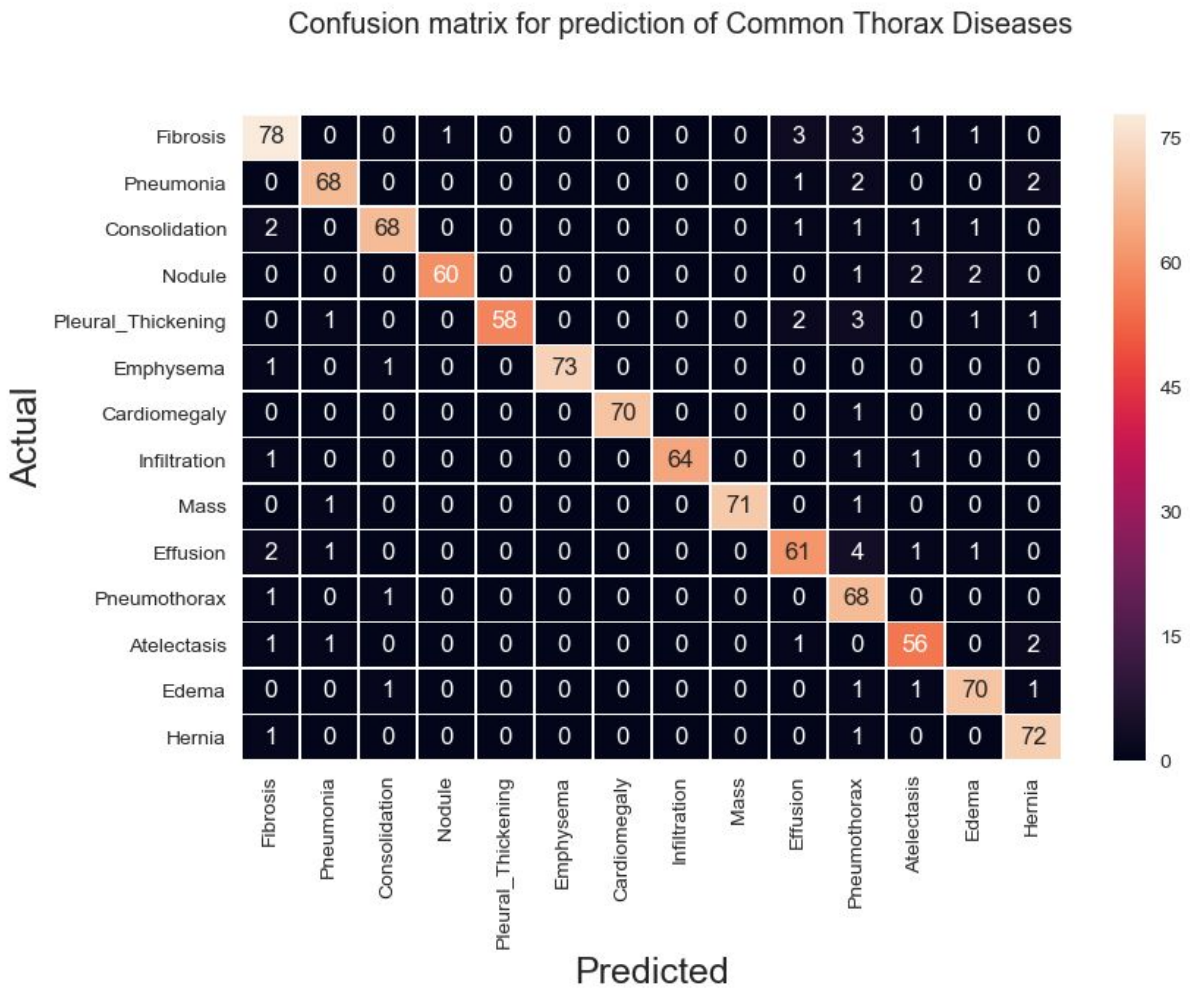
The accuracy increased from 34.45% to **96.82%**. But the accuracy for the test dataset with known classes is as follows:

Classification accuracy = Correct predictions / Total predictions * 100
Error rate = (1 - (Correct predictions / Total predictions)) * 100
Total test dataset count = 1000
Correct predictions = 937

Accuracy and Error percentage

```
Accuracy score :  93.7 %
Error rate :  6.2999999999999945 %
```





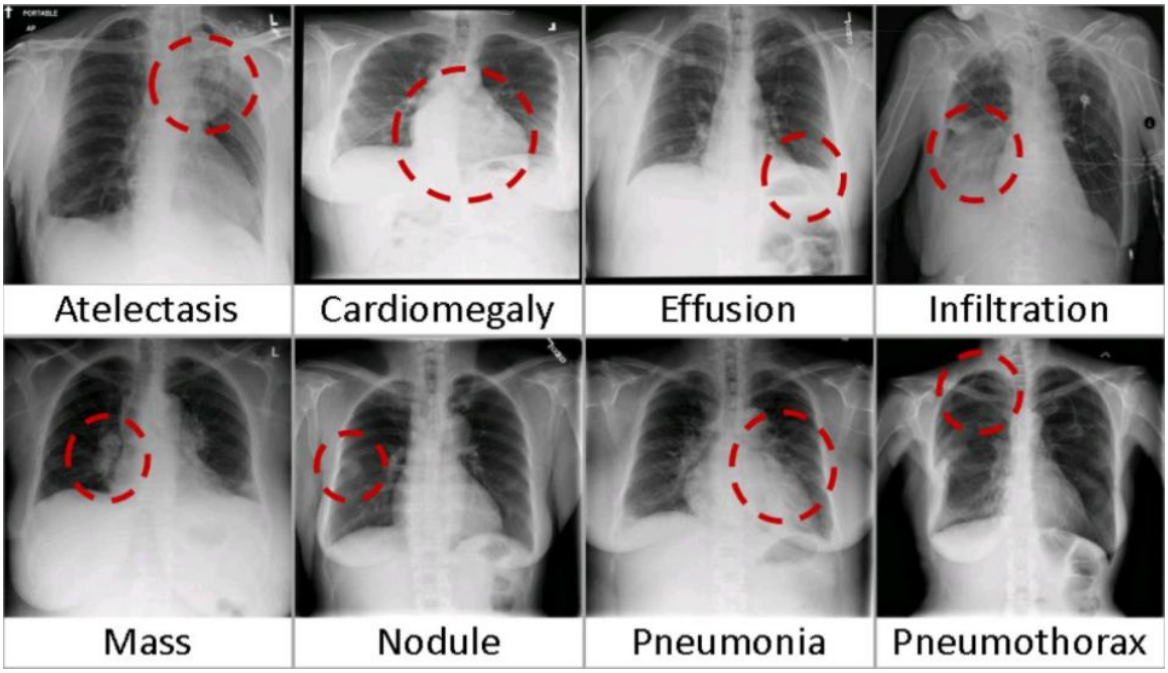
Confusion matrix : The above matrix show quality of models predictions. Approximately 65(-varies depending on the dataset) images from each class was chosen for the confusion matrix.

Concluding remarks :

We were able to predict the class of disease with almost 93.7% accuracy from a dataset of 18K+ count. The images were slightly abrupt, but made it 90% clean by preprocessing.
Major part of our knowledge gain is on Transfer Learning and its in-depth mechanisms. Also we were able to understand the importance of preprocessing of data.

To some extent we able to picturize how the machines interpret the data. On deconvolution of images, it gave weird images which the machines has learnt.

Our future enhancement is to accurately predict even which region of the cavity has the symptoms from their new dataset:



However this could lead to resource exhaustion on GPU ideally as the increase count of parameters.

References

https://machinelearningmastery.com/use-pre-trained-vgg-model-classify-objects-photographs/

https://nihcc.app.box.com/v/ChestXray-NIHCC/file/220660789610

https://keras.io/preprocessing/image/

https://machinelearningmastery.com/transfer-learning-for-deep-learning/

https://machinelearningmastery.com/best-practices-document-classification-deep-learning/

https://www.hackerearth.com/challenge/competitive/deep-learning-challenge-2/