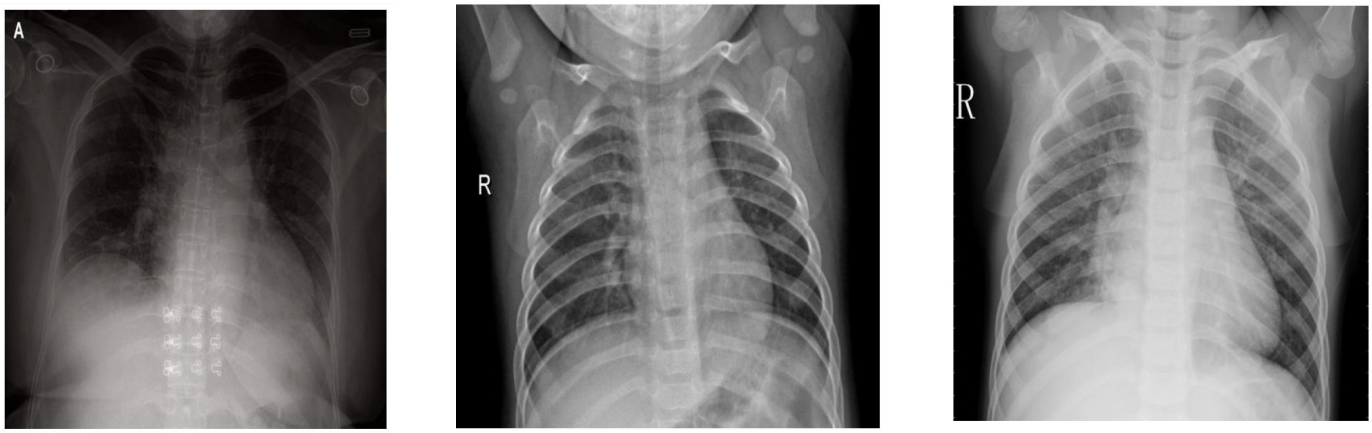
CO468-Computer Vision **Samyak Jain-16CO254**
Assignment-2 **Mishal Shah-16CO125**
X-Ray Disease Classification **Namrata Ladda-16CO121**

Aim: To classify X-ray images of patients as COVID-19, Normal and Pneumonia

Data Analysis:
The dataset comprises high resolution X-ray images each of size 512x512x3. There are 3 types of x-ray images COVID-19, normal and pneumonia. The intention here is to build a deep learning model that can correctly perform the multi-class(3) classification task. The major challenge here is that the dataset contains a minority of COVID-19 images. Following images showcase the difference between the 3 classes:



A B C

The first image A is the X-Ray of the COVID-19 patient, as evident there is some amount of congestion in the lungs due to the infection. The second image B is of a normal patient and one can clearly see a very clean X-Ray without any congestion. The last image C is of a person suffering from pneumonia. It is difficult to make out the differences between A & C by a non-domain person and a domain expert(doctor) is needed to tell the difference.

Data Preparation:
We had access to 2318 samples of images and we first did a train-test split keeping 80% data for training and 20% for testing. This made sure our test data consisting of 465 sample images becomes an unbiased estimator of the accuracy of the classifier. For the rest of the training data, since the data was highly imbalanced we applied the following techniques:

- *Weighted Loss Function:* we used the normal training data itself but used a weighted loss function so that the minority COVID-19 images get more weight and training is done properly without any biases.
- *Down Sampling & Augmentation:* we down sampled the number of normal and pneumonia images to the number of COVID-19 images and then applied similar types of Augmentation on all 3 classes of images. Augmentation included rotation, translation, zooming etc. Finally, we had 6,838 samples of augmented training data with nearly the same proportion of each class.

Methodology:

We have tried several different deep learning approaches to classify the X-Ray Images. The approaches are discussed as follows:

- **State-of-the-art image classification models**

  We used several state-of-the-art image classification models to solve the problem. These models were trained on both the types of training data as discussed above. These models included ResNet-50, DenseNet-121, EfficientNet-B0, InceptionNet-V3.

- **Transfer Learning Techniques**

  We used transfer learning techniques by loading the imagenet classification weights for each of the state-of-the-art image classification models. We adopted a strategy via which we froze all the weights of the model and only the last fully connected weights were trained. So it was essentially using a pre-trained model and then doing fine tuning for a deeper layer. Once again models were trained on both the type of data and we applied the transfer learning techniques using ResNet-50, DenseNet-121, XceptionNet.

- **Custom Model/Novelty**

  In order to further improve the classification accuracy we used a novel ensemble approach wherein we combined two state-of-the-art image classification models. The architecture design is depicted in Figure 1.

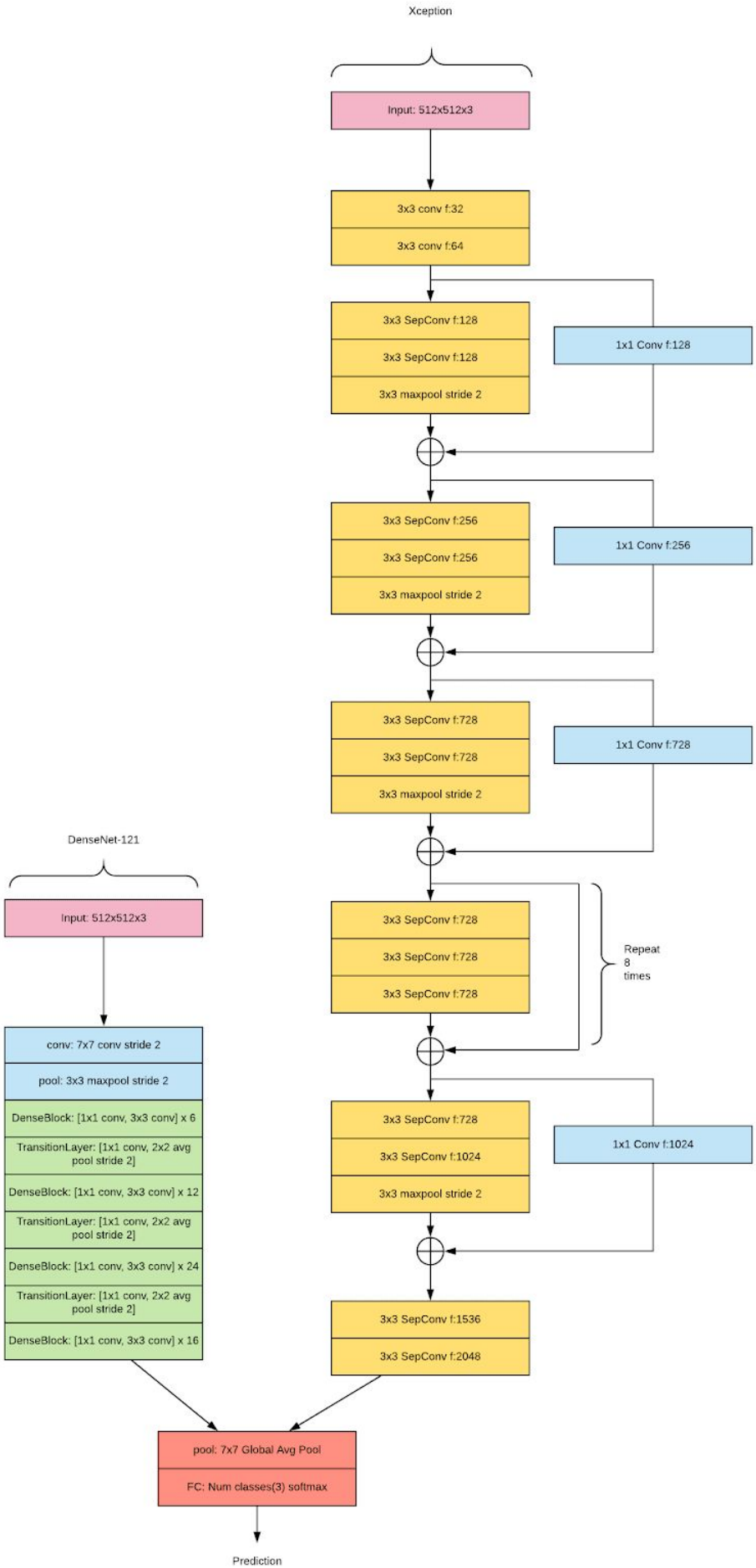

Xception
Input: 512x512x3
3x3 conv f:32
3x3 conv f:64
3x3 SepConv f:128
3x3 SepConv f:128
3x3 maxpool stride 2
1x1 Conv f:128
3x3 SepConv f:256
3x3 SepConv f:256
3x3 maxpool stride 2
1x1 Conv f:256
3x3 SepConv f:728
3x3 SepConv f:728
3x3 maxpool stride 2
1x1 Conv f:728
3x3 SepConv f:728
3x3 SepConv f:728
3x3 SepConv f:728
Repeat 8 times
3x3 SepConv f:728
3x3 SepConv f:1024
3x3 maxpool stride 2
1x1 Conv f:1024
3x3 SepConv f:1536
3x3 SepConv f:2048
DenseNet-121
Input: 512x512x3
conv: 7x7 conv stride 2
pool: 3x3 maxpool stride 2
DenseBlock: [1x1 conv, 3x3 conv] x 6
TransitionLayer: [1x1 conv, 2x2 avg pool stride 2]
DenseBlock: [1x1 conv, 3x3 conv] x 12
TransitionLayer: [1x1 conv, 2x2 avg pool stride 2]
DenseBlock: [1x1 conv, 3x3 conv] x 24
TransitionLayer: [1x1 conv, 2x2 avg pool stride 2]
DenseBlock: [1x1 conv, 3x3 conv] x 16
pool: 7x7 Global Avg Pool
FC: Num classes(3) softmax
Prediction

Figure 1: Architecture for Ensemble Approach

Here, DenseNet-121 and Xception models are combined to give a boost to accuracy. The internal structure of the models remains the same, the final activation maps of both the models are concatenated followed by a global average pooling and a fully connected layer with softmax activation.In this approach also we combined several models- DenseNet-121+ResNet-50, DenseNet-121+MobileNet-V2, DenseNet-121+XceptionNet. We also utilised the concepts of transfer learning, by freezing both the models weights and only the last FC layer depicted in Red color was fine tuned.

Results & Analysis

The following table shows the detailed results and the performance analysis. The experiments were performed on Google Colab with 26GB available RAM and GPU acceleration with tensorflow version of `2.2.0`.

Table 1 depicts the performance of various methods as described above for the Test Data prepared by us. [Note: Here, precision, recall and F-Score are the weighted average reported by the classification report]

SNo	Data	Model	Accuracy	Precision	Recall	F-Score
1.	Augmented Train Data	ResNet-50	47%	75%	47%	31%
2.	Augmented Train Data	DenseNet-121	79%	85%	79%	78%
3.	Augmented Train Data	EfficientNet-B0	91%	92%	91%	91%
4.	Train Data with weighted loss function	ResNet-50	40%	73%	40%	36%
5.	Train Data with weighted loss function	DenseNet-121	88%	90%	88%	88%
6.	Train Data with weighted loss function	EfficientNet-B0	78%	92%	78%	80%
7	Augmented Train Data	InceptionNet-V3	91%	92%	91%	91%

8.	Train Data with weighted loss function	Transfer Learning DenseNet-121	95%	95%	95%	95%
9.	Train Data with weighted loss function	Transfer Learning ResNet-50	81%	87%	81%	81%
10.	Train Data with weighted loss function	Transfer Learning Xception Net	95%	96%	95%	95%
11.	Augmented Train Data	Transfer Learning DenseNet-121	91%	92%	91%	91%
12.	Augmented Train Data	Transfer Learning ResNet-50	94%	94%	94%	94%
13.	Augmented Train Data	Transfer Learning Xception Net	90.11%	92%	90%	90%
14.	Train data with weighted loss function	Transfer Learning + Ensemble of DenseNet-121 and MobileNet V2	97%	97%	97%	97%
15.	**Train data with weighted loss function**	**Transfer Learning + Ensemble of DenseNet-121 + Xception Net**	**98%**	**98%**	**98%**	**98%**
16.	Train data with weighted loss function	Transfer Learning + Ensemble of DenseNet-121 and ResNet-50	97%	97%	97%	97%
17.	Augmented Train data	Transfer Learning +	91%	91%	91%	91%

		Ensemble of DenseNet-121 and MobileNet V2				
18.	Augmented Train data	Transfer Learning + Ensemble of DenseNet-121 + Xception Net	92%	92%	92%	92%
19.	Augmented Train data	Transfer Learning + Ensemble of DenseNet-121 and ResNet-50	94%	94%	94%	94%
20.	Train data with weighted loss function	Ensemble of DenseNet-121 and MobileNet V2	90%	91%	90%	90%
21.	Train data with weighted loss function	Ensemble of DenseNet-121 + Xception Net	97%	97%	97%	97%
22.	Train data with weighted loss function	Ensemble of DenseNet-121 + ResNet-50	97%	97%	97%	97%
23.	Augmented train data	Ensemble of DenseNet-121 + MobileNetV2	85%	87%	85%	85%
24.	Augmented train data	Ensemble of DenseNet-121 + Xception Net	95%	95%	95%	95%
25.	Augmented train data	Ensemble of DenseNet-121 + Res Net-50	93%	93%	93%	93%

Table I - Performance Analysis of Various Techniques*

**Top performing model has been highlighted in Green Color*

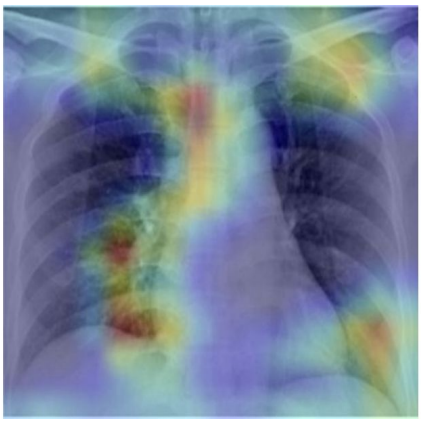
Observations

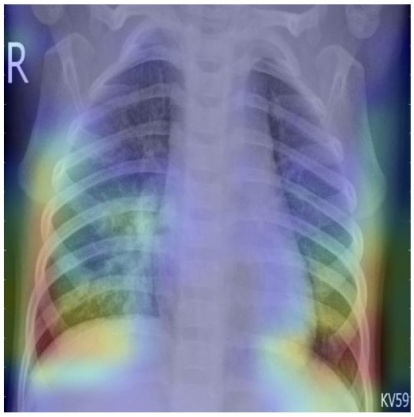
Most of the state-of-the-art image classification models performed well on both types of Data. Some of the models like EfficientNet-B0 and InceptionNet-V3 gave 90+ accuracy. Transfer learning also gave a tremendous boost to the accuracy giving us nearly 95% accuracy. This could be due to the fact that the pre-trained weights were used that were trained on 1 million images of the ImageNet data and hence earlier layers were able to capture simple features like edges, corners accurately. One more thing to notice is that train data with weighted loss function gave better results as compared to the augmented data. The reason could be while down sampling we might be losing essential information and the other thing is that we could only perform data normalization on the train data and not on augmented train data due to computational constraints. Therefore due to normalization convergence would have been faster resulting in better performance. Ensemble approach gave another 2% boost to the accuracy leading to 97% accuracy. ***Almost all the ensemble approach models (as depicted above in Green Shade) gave us 97-98% accuracy using the train data with weighted loss function.*** Here, both transfer learning and training from scratch gave similar results.

Note: Out of the 5 comparable models, we have chosen the "DenseNet-121 + Xception" ensemble that was trained using transfer learning on train data with weighted loss function (Row No 15 in Performance Analysis table above) as our best performing model. Following GradCam visualizations and ROC curves are w.r.t to this model.

GradCam Visualization

GradCam is a visualization technique which uses gradient information flowing into the last convolutional layer to understand the decision interest of each neuron. Basically it gives the discriminative regions in the image. The following are gradcam visualizations of different classes of test images.



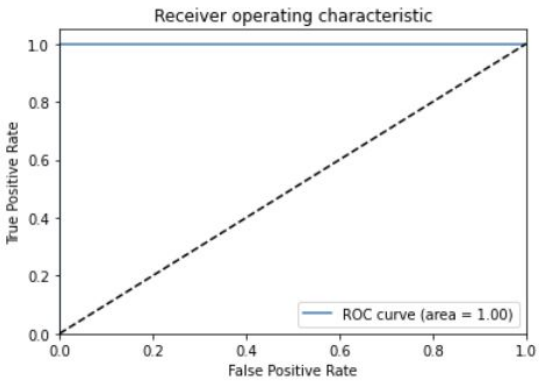


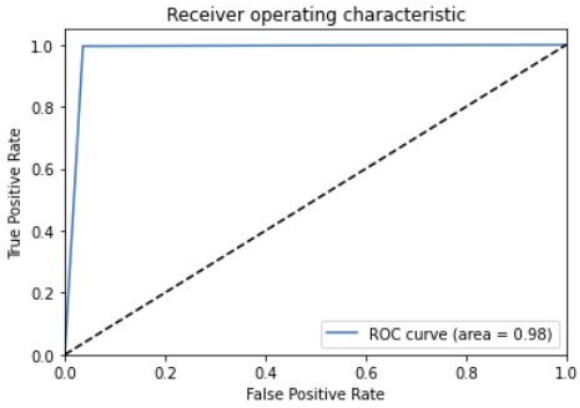
A B C

Image A is the GradCam for COVID-19 patients, as evident there are a number of red patches in the lungs and hence the model is able to learn and figure out the patient has COVID-19 as the infection is prominent in lungs. In case of normal patient (B), the X-ray is very clean and there are no red patches in lungs, indicating the model learns that there is no infection in the person. Last, image C shows an image for pneumonia patients and as evident the model learns from the red patches at the sides of lungs indicating pneumonia infection in lungs.

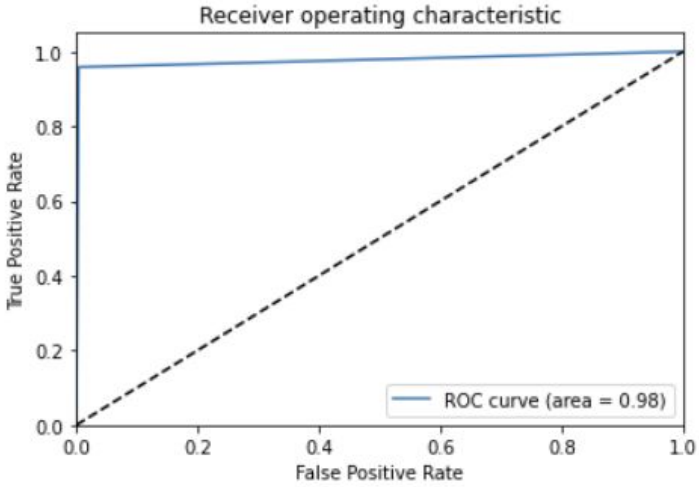
ROC Curve Visualization

ROC curves give a relationship between precision and recall and help us to find how the model is performing for multiclass classification using one vs all approach. The following are the ROC curves for the best model.





ROC Curve - COVID-19 **ROC Curve - Normal**



ROC Curve - Pneumonia

As evident from the ROC curves the model is not biased towards a particular class and has high AUC (1,0.98,0.98) for all the classes.

Note: Please normalize the test data before performance evaluation. Use the following train_mean and train_std for that:

- *train_mean = 124.72302388415788*
- *train_std = 62.71547651610164*

Conclusion, Future Work & Recommendations

Out of all the experiments performed, we got the best results using the ensemble approach of combining two state-of-the-art image classification models and using train data with weighted loss function. Due to limited computational resources at hand we could only train the ensemble model on a very small batch size of 4. So, it was more like performing stochastic gradient descent, therefore we recommend to try out the solution using larger batch size so it becomes much closer to mini-batch gradient descent with better performance. Also, we were not able to normalize the augmented train data so we recommend trying out the solution by normalizing the augmented train data as it's a common notion that deep learning models perform well on more data. Lastly, we tried to integrate the concept of progressive resizing that we analysed in Assignment-1, but then the model became too complex and we were not able to train it given the limited resources. We recommend trying out the integration of ensemble and progressive resizing as a novel solution to solve the problem.