CS542 Class Challenge Report

Team: Janvee Patel, Yanzheng Wu

05/01/2022

Task 1 Description

In task one, we used **VGG16** that has been trained on ImageNet as our base model [1]. The VGG16 architecture follows an arrangement of convolutional layers with a 3x3 filter and stride of 1 followed by a maxpool layer with a 2x2 filter and stride of 2, resulting in 13 convolutional layers and 3 fully connected layers at the end [1]. This architecture employs smaller filters with a larger depth in the convolutional layers. In our model, the classification layer was removed, making the output dimension from the VGG16 base model layer as [None, 7, 7, 512]. There are four additional layers, other than the base model as our head layers per Table 1. These contain the following layers respectively: “flatten” layer (output dimension as [None, 25088]), “dense1” dense layer with 256 units and ReLU activation function (output dimension as [None, 256]), “dropout1” dropout layer with dropout rate of 0.3 (output dimension as [None, 256]), and “pred_dense” dense layer with 1 unit and sigmoid activation function (output dimension as [None ,1]). A dropout layer was added as a method for overcoming overfitting, and the dropout rate of 0.3 was determined for best performance after fine-tuning this parameter.

Table 1. Summary of Model Architecture for Task 1

Layer	Output Shape	Parameters
VGG16	(None, 7, 7, 512)	14,714,688
flatten	(None, 25088)	0
dense1	(None, 256)	6,422,784
dropout1	(None, 256)	0
pred_dense	(None, 1)	257

Our combined model has total parameters of 21,137,729 with 6,423,041 trainable and 14,714,688 non-trainable. The model weights within the VGG16 base model were set to not be trained. The model uses the Adam optimizer with a learning rate equal to 0.001. The loss function we adopted is binary cross-entropy since the output matrix is binary. We used the mini-batch method with a batch size of 10. To perform dimensionality reduction for feature visualization, features were extracted from the dense1 layer and t-SNE was used to reduce the dimensionality of the extracted features to 2 dimensions.

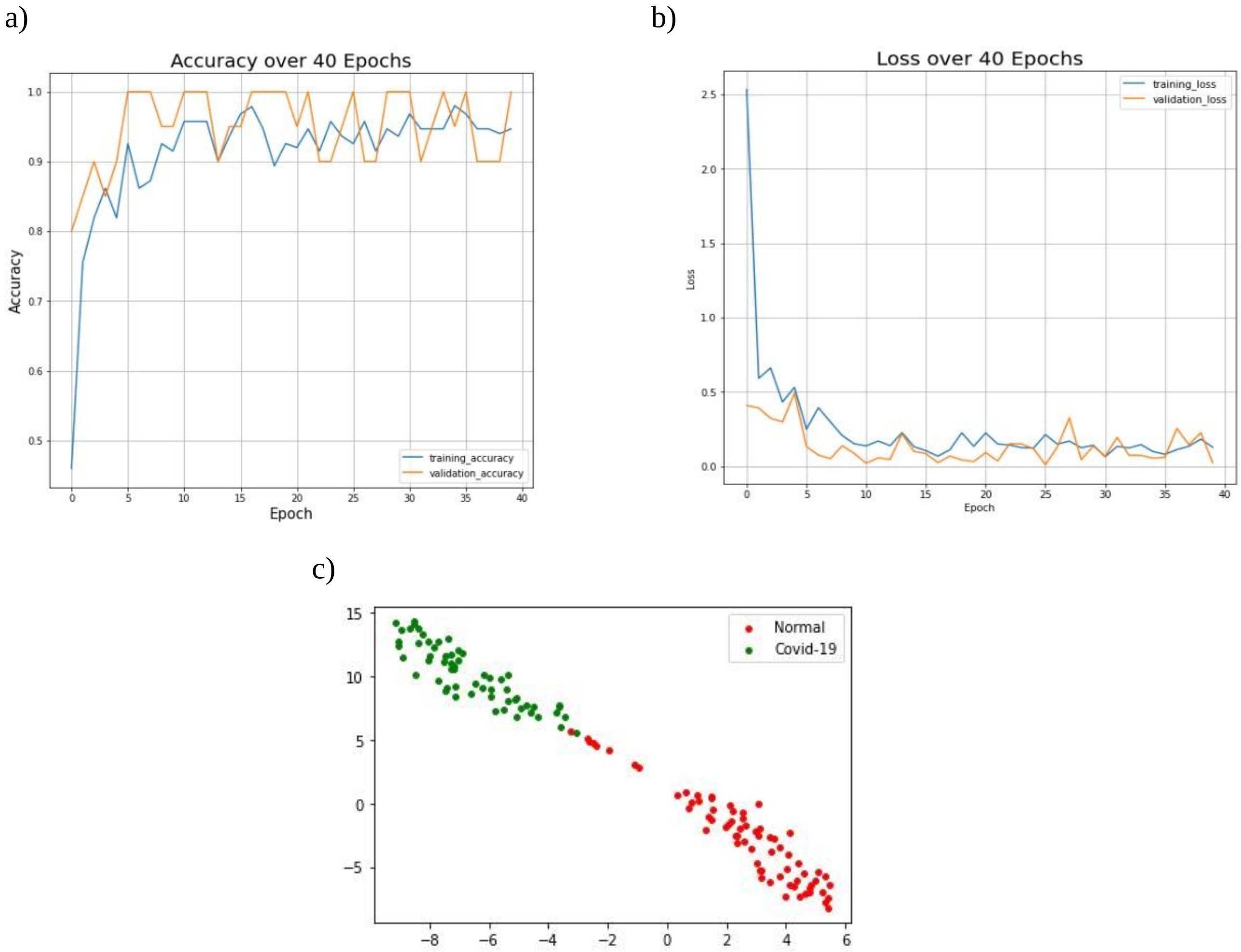


Figure 1. Performance Metric Plots for Task 1 Model
Fig 1a. shows the training and validation accuracy over 40 epochs, while Fig 1b. shows the training and validation loss over 40 epochs. Fig 1c. is a t-SNE plot of 2D features colored by Covid-19 and Normal.

The above Figure 1a. shows the task 1 model's training and validation accuracy over the 40 epochs. We can observe that both accuracies converge roughly at the fifth epoch. Then both accuracies are oscillating within the range of 0.9 to 1.0. The above Figure 1b. shows the training and validation loss over 40 epochs. The elbow point again is roughly at the fifth epoch. After its convergence, both losses are fluctuating between 0.245 to 0.05. The t-SNE plot in Figure 1c. shows the result of task 1 model classifying 130 X-ray images belonging to two classes. We can see that the model has a clear discrimination over the "Normal" group and the "Covid-19". There are a few points classified as "Normal" that are not clustered distinctly with the "Normal" class, and this could be due to a low learning rate in t-SNE. Upon evaluating on the test dataset, our task 1 model had 0.045 loss and 100% test accuracy given 18 different test X-ray images.

Task II Description

In task two, we used **Xception** trained on ImageNet as our first base model [2]. The Xception architecture, divided by entry, middle, and exit flow, has 36 convolutional layers, and implements depth-wise separable convolutions in which these convolutions have a depth-wise spatial convolution performed on each channel followed by pointwise convolution on the output together [2]. In our model, the classification layer was removed, so that output dimension of the Xception base model was [None, 7, 7, 2048]. There are four additional layers as shown in Table 2. These contain the following layers respectively: “global_average_pooling2d” average pooling layer (output dimension as [None, 2048]), “dense1” dense layer with 256 units and a ReLU activation function (output dimension as [None, 256]), “dropout1” dropout layer with a dropout rate of 0.2 (output dimension as [None, 256]), and “pred_dense” dense layer with 4 units and a softmax activation function (output dimension as [None, 4]). Instead of flattening the output of the Xception base model, a global average 2D pooling was applied which reduces each feature map to a single value for better representation of the output vector. This step improved the performance accuracy and helped minimize overfitting behavior because the output shape was reduced to [None, 2048], reducing the number of trainable parameters in the dense1 layer. Dropout was implemented as a technique for reducing overfitting.

Table 2. Summary of Model 1 Architecture for Task 2 (Xception)

Layer	Output Shape	Parameters
Xception	(None, 7, 7, 2048)	20,861,480
global_average_pooling2D	(None, 2048)	0
dense1	(None, 256)	524,544
dropout1	(None, 256)	0
pred_dense	(None, 4)	1028

Our combined model 1 has total parameters of 21,387,052 with 525,572 trainable and 20,861,480 non-trainable. The weights within the layers of the Xception base model were set to not be trained. The model uses the Adam optimizer with a learning rate equal to 0.0001. This learning rate was determined for best performance after fine-tuning the hyperparameter. The loss function we adopted is Categorical cross-entropy due to having multiple classes. Again, we used the mini-batch method with a batch size of 10. The model was trained for 100 epochs and used training and validation batches. For feature visualization, features were extracted from the dense1 layer and passed into t-SNE for dimensionality reduction.

We used **MobileNet V2** trained on ImageNet as our second base model for comparison [3]. The MobileNet V2 architecture implements an inverted residual structure which comprises of one block with a stride of 1 and a second block with a stride of 2, and consists of 1x1 convolution with ReLU6 activation, followed by 3x3 depth-wise convolution with ReLU6 activation, and a third convolutional layer without a non-linearity function [3]. The classification layer was removed, making the output dimension of the base model [None, 7, 7, 1280]. There are four layers in addition to the base model as our head layers as shown in the Table 3 summary of model 2 architecture. These contain the following layers respectively: “flatten” layer which flattens the output from the base model layer (output dimension as [None, 62720]), “dense1” dense layer with 128 units and a ReLU activation function (output dimension as [None, 128]), “dropout1” dropout layer with a dropout rate of 0.35 (output dimension as [None, 128]), and “pred_dense” dense layer with 4 units and a softmax activation function (output dimension as [None, 4]). In this model, due to the higher number of trainable parameters in the following layer, we chose to a smaller number of units set at 128. This number of units in the dense1 layer showed a better performance compared to using higher unit values of 256 and 512. In addition, dropout was implemented as a technique for overcoming overfitting, and a value of 0.35 was determined for best performance upon fine-tuning the parameter.

Table 3. Summary of Model 2 Architecture for Task 2 (MobileNet V2)

Layer	Output Shape	Parameters
Mobilenet	(None, 7, 7, 1280)	2,257,984
flatten	(None, 62720)	0
dense1	(None, 128)	8,028,288
dropout1	(None, 128)	0
pred_dense	(None, 4)	516

Our combined model 2 has total parameters of 10,286,788 with 8,028,804 trainable and 2,257,984 non-trainable. The model uses the Adam optimizer with a learning rate equal to 0.0001 which was fine-tuned for best performance. The loss function we adopted is Categorical cross-entropy due to having multiple classes. A batch size of 10 was used, and the model was trained for 85 epochs using training and validation batches. For feature visualization, t-SNE was used to reduce the dimensionality of the features extracted from the dense1 layer.

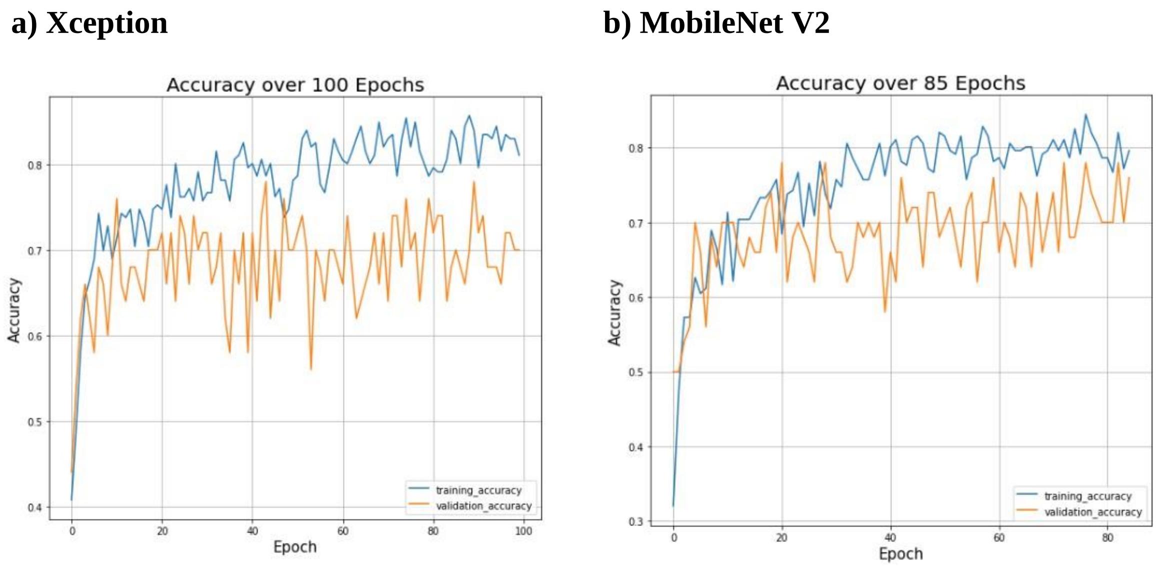
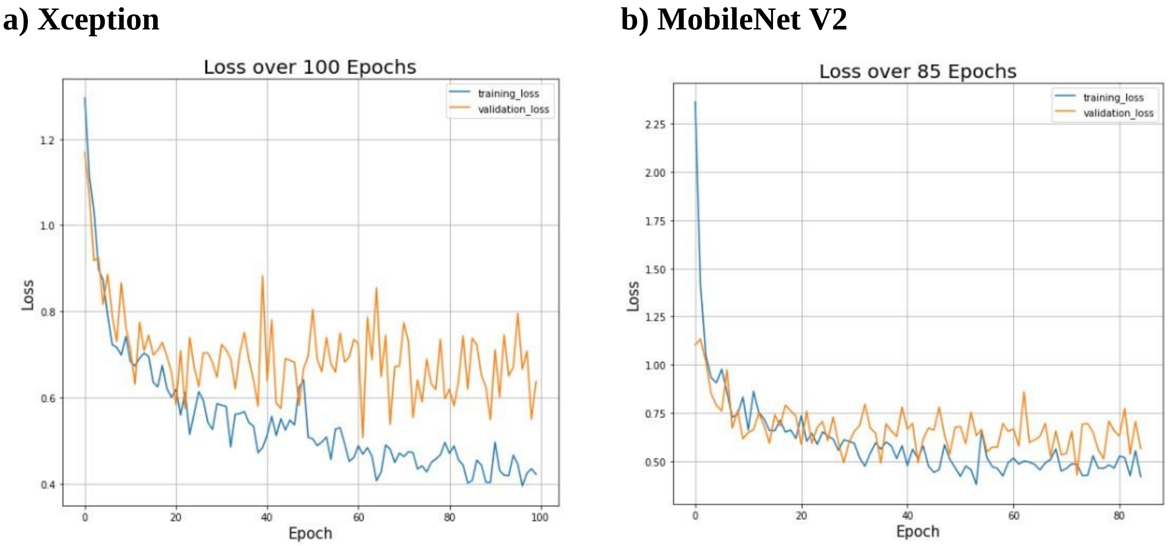


Figure 2. Accuracy Plots for a) Pre-trained Xception Model 1 b) Pre-trained MobileNet V2 Model 2

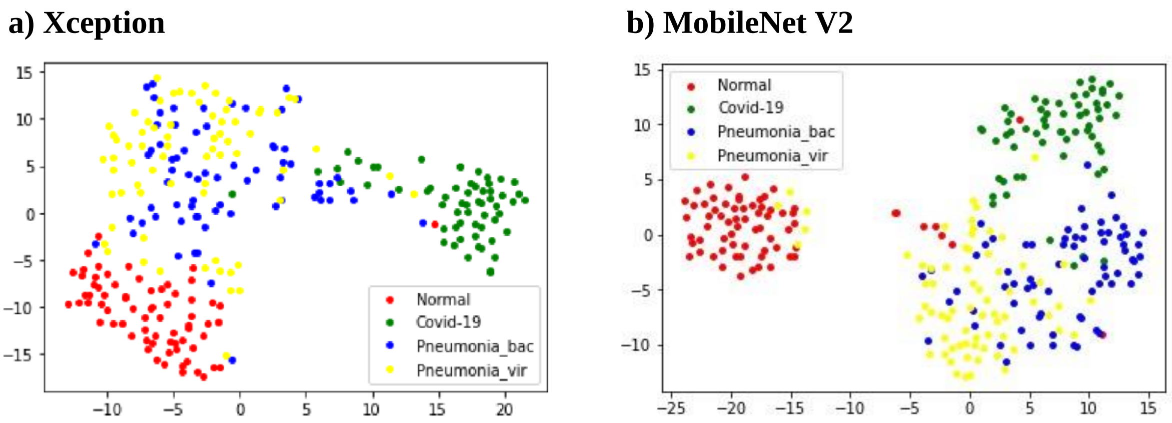
The above plots are the accuracy plots for the two models in task 2. Figure 2a shows that both the first model (Xception)'s training and validation accuracies converge at around 20 epochs with the training accuracy varying at roughly 0.8 and validation accuracy at approximately 0.7. Figure 2b shows that both the first model(MobileNet V2)’s training and validation accuracies converge at around 15 epochs with the training accuracy varying at roughly 0.74 and validation accuracy at approximately 0.7. In addition, both the training and validation accuracies appear to be gradually increasing as the number of epochs increases. The training accuracy appears to be higher in the Xception model maintaining above 82-84% as it approaches 85 epochs, while the training accuracy in MobileNet V2 model maintains around 80% as it approaches 85 epochs. The validation accuracy showed a greater fluctuation in the Xception model from 35-60 epochs but maintained a similar validation accuracy as the MobileNet V2 model as it approached 70 epochs.



a) Xception **b) MobileNet V2**

Figure 3. Loss Plots for a) Pre-trained Xception Model 1 b) Pre-trained MobileNet V2 Model 2

The above plots are the loss plots for the two models in task 2. Figure 3a shows that both the first model (Xception)'s training and validation losses converge at around 23 epochs with the training loss continually decreasing to 0.4 and validation loss fluctuating at approximately 0.7. Whereas Figure 3b shows that both the second model (MobileNet V2)'s training and validation losses converge at around 10 epochs with the training loss continually decreasing below 0.5 and validation loss fluctuating at approximately 0.60. In both models, the training loss appears to be gradually decreasing as the number of epochs increases. Likewise to the accuracy plot of Xception’s model, there is a greater fluctuation in the training loss from 40-60 epochs, but maintains a validation loss below 0.75 as it approaches 85 epochs similar to the MobileNet V2 model.


a) Xception

b) MobileNet V2

Figure 4. t-SNE Plots for a) Pre-trained Xception Model 1 b) Pre-trained MobileNet V2 Model 2

Above are the t-SNE graphs for Xception based model and MobileNet V2 based model, respectively. From the graphs, we see that both models are able to distinguish the “Covid-19” group and “Normal” group pretty well, as the red and green clusters are distinctly separated. MobileNet V2 shows a distinct separated cluster for the “Normal” class with a few outliers, while Xception’s model shows a distinct cluster for the “Normal” class with one outlier. However, neither model could discriminate between the “Pneumonia_bac” group and the “Pneumonia_vir” group well, as the yellow and blue clusters are overlapping with each other. These results could be due to the usage of a learning rate that was too low for t-SNE. Since both models were not able to distinctly discriminate between the two Pneumonia classes, it is likely that the characteristics of these two diseases are quite similar in an X-ray, so the model would need to learn more complex features to be able to discriminate correctly. By using a larger dataset, this behavior might be able to improve.

Overall, both the Xception model 1 and Mobilenet V2 model 2 were evaluated on the test dataset. The loss for model 1 was 0.5150 and had a test accuracy of 78% given 36 different test X-ray images. The loss for model 2 was 0.8866 and had a test accuracy of 64% given 36 different test X-ray images. Based on these results, the first model has a better overall performance than our second model. Since our dataset is small, performance metrics would be best evaluated on a large dataset of X-ray images to assess which of the two models performs better overall. Nevertheless, a potential reason for our results could be that MobileNet V2 has 53 layers whereas the Xception has 71 layers. With the addition of our classification layers, our first model has almost twice as many total parameters than our second model, which the first might be better in capturing some subtle traits from the X-ray images as it has a large number of parameters within the base model alone. In addition, a global average pooling layer was added in the Xception model which could have led to a better representation of the output features from the base model. Whereas the MobileNet V2 model used a flattening method and therefore had a greater number of trainable parameters which addressed overfitting by using a smaller number of units in the dense layer and dropout. Another possible explanation could be the major difference in the architecture of base models. The Xception base model is a depth-wise separable convolutional network, which proved to give better performance results under a discrete spectrum situation. As for the MobileNet V2, it is a much lighter architecture that is mobile-oriented compared to Xception hence MobileNet V2 might be in less advantage when more complexities are introduced by utilizing X-ray images.

References

[1] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition", *arXiv preprint arXiv:1409.1556*, 2014.

[2] Chollet F. “Xception: Deep learning with depthwise separable convolutions”, *arXiv preprint arXiv*: 1610.02357, 2017

[3] Sandler, M., Howard, A.G., Zhu, M., Zhmoginov, A., & Chen, L. (2018). MobileNetV2: Inverted Residuals and Linear Bottlenecks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4510-4520.