Implementing Transfer Learning Along Data Augmentation and Fine Tuning CNN

Muhammad Junaid Akram – Dept of Computer Science SEECS, NUST

***Abstract*—Data labeling is one of the greatest challenges in the field of computer vision. Multiple techniques and models have been used to improve performance. The transfer learning technique has enabled the researchers to use the weights of pretrained models to train their models more efficiently. This paper shows the implementation of transfer learning in which the weights of VGG16 and VGG19 are being used which has been trained on IMAGENET dataset. These weights are being used on Intel Image Classification dataset. The other technique used is data augmentation which has been implemented for better data generalization, hence improving the performance of the model. The final model gave the 92.30% training accuracy and 90.67% validation accuracy with VGG19.**

Keywords—transfer learning, data augmentation, CNNs, image classification, VGG16

I. INTRODUCTION

Techniques for image classification and data labeling has changed a lot with the time. In machine learning, different filter kernels were used to learn features from the images manually but in the deep learning neural network learns features automatically. Over the period, different image classification changes came, out of which ImageNet was one the famous image classification challenge. It started back in 2010 and ended in 2016. A lot of convolutional neural network architecture showed tremendous performance and won the challenge. Few of them are AlexNet, VGG, GoogleNet, ResNet and SENet. A new concept of transfer learning, which allows the use the weights of pre-trained models for image classification. In this paper, intel image classification dataset is used which consist of six classes i.e. mountains, sea, forest, street, buildings and glaciers. For purpose of classification, transfer learning technique is used and the weights of VGG16 and VGG19 were used which were trained on ImageNet. The data set contains 14 thousand training images, 3 thousand test images and 7 thousand prediction images. Another technique used in this paper for better model training in data augmentation on the fly. This technique allows to generate more data for more better and generalized training of the model thus leading to more and more better results.

II. METHODOLOGY

In this paper, 2 different models were used for the image classification i.e. VGG16 and VGG19. Different models with different training setting were tested to get the best trained model. Whereas, the setting for data augmentation is kept same in all of the models trained. It includes rescaling, rotation range, width shift range and height shift range, zoom range, horizontal flip and fill mode. Remaining settings of different models are as follows.

A. Base Model

The base model is VGG16 and its all layers are non-trainable. At the end 2 Dense layers with dense layer 1 containing 64 neurons and final layer with 6 neurons for output. Categorial cross entropy loss is used with the adam optimizer. Batch size for the dataset is kept 32 and shuffling of data is kept on. Model is trained over 25 epochs.

B. 1 Layer Trainable – VGG16

Since top layers of the model are learning more specific features of the given image. This model is VGG16 and its all layers are non-trainable other than **block5_conv3** is kept trainable. At the end 2 Dense layers with dense layer 1 containing 64 neurons and final layer with 6 neurons for output. Categorial cross entropy loss is used with the adam optimizer. Batch size for the dataset is kept 32 and shuffling of data is kept on. Model is trained over 25 epochs.

C. 2 Layers Trainable – VGG16

Since top layers of the model are learning more specific features of the given image. So, enabling more layers will allow model to learn more current data specific features. This model is VGG16 and its all layers are non-trainable other than **block5_conv3** and **block5_conv2** are kept trainable. At the end 2 Dense layers with dense layer 1 containing 64 neurons and final layer with 6 neurons for output. Categorial cross entropy loss is used with the adam optimizer. Batch size for the dataset is kept 32 and shuffling of data is kept on. Model is trained over 25 epochs.

D. 3 Layers Trainable – VGG16

Since top layers of the model are learning more specific features of the given image. So, enabling more layers will allow model to learn more current data specific features. This model is VGG16 and its all layers are non-trainable other than **block5_conv3, block5_conv2** and **block5_conv1** are kept trainable. At the end 2 Dense layers with dense layer 1 containing 64 neurons and final layer with 6 neurons for output. Categorial cross entropy loss is used with the adam optimizer. Batch size for the dataset is kept 32 and shuffling of data is kept on. Model is trained over 25 epochs.

E. 2 Layers Trainable with SGD – VGG16

Since top layers of the model are learning more specific features of the given image. So, enabling more layers will allow model to learn more current data specific features. This model is VGG16 and its all layers are non-trainable other than **block5_conv3** and **block5_conv2** are kept trainable. At the end 2 Dense layers with dense layer 1 containing 64 neurons and final layer with 6 neurons for output. Categorial cross entropy loss is used with the sgd optimizer. Learning rate of 0.001, decay 1 e^{-7} and momentum is kept 0.9. Batch size for the dataset is kept 32 and shuffling of data is kept on. Model is trained over 25 epochs.

F. 3 Layers Trainable with SGD – VGG19

Since top layers of the model are learning more specific features of the given image. So, enabling more layers will

©2021 IEEE

allow model to learn more current data specific features. This model is VGG19 and its all layers are non-trainable other than **block5_conv4, block5_conv3** and **block5_conv2** are kept trainable. At the end 2 Dense layers with dense layer 1 containing 64 neurons and final layer with 6 neurons for output. Categorial cross entropy loss is used with the sgd optimizer. Learning rate of 0.001, decay 1 e^{-7} and momentum is kept 0.9. Batch size for the dataset is kept 64 and shuffling of data is kept on. Model is trained over 25 epochs. The fig 1.1 shows the architecture of the neural network

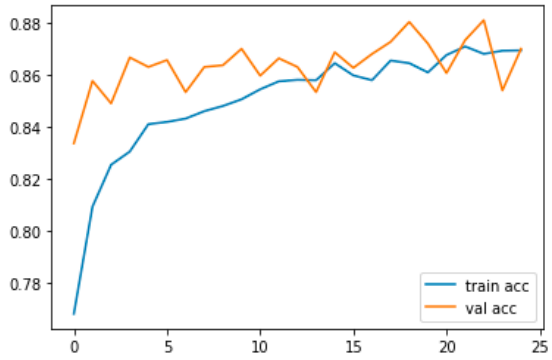
III. Results

Results of the models are as follows.

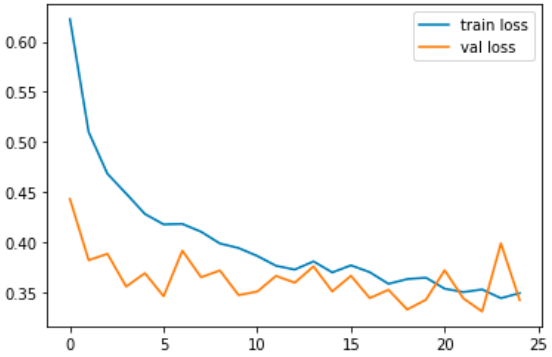
A. *Base Model*

This model is completely using all of the layers trained over ImageNet. No convolution layer of the model is being training again. We will be using base model to compare its results with upcoming models. Its training accuracy is 87.54% and validation accuracy is 87% and train and validation loss is 33.70% and 34.20% respectively.

Accuracy



Loss



Confusion Matrix

```
[[1064    6    4    4    4   62]
 [   2 1150    5    5    1    3]
 [   0    5 1133  164   26    2]
 [   5    6   59 1208   19    0]
 [   8    4   40   93  980    3]
 [  71   15    7    0   10 1133]]
```

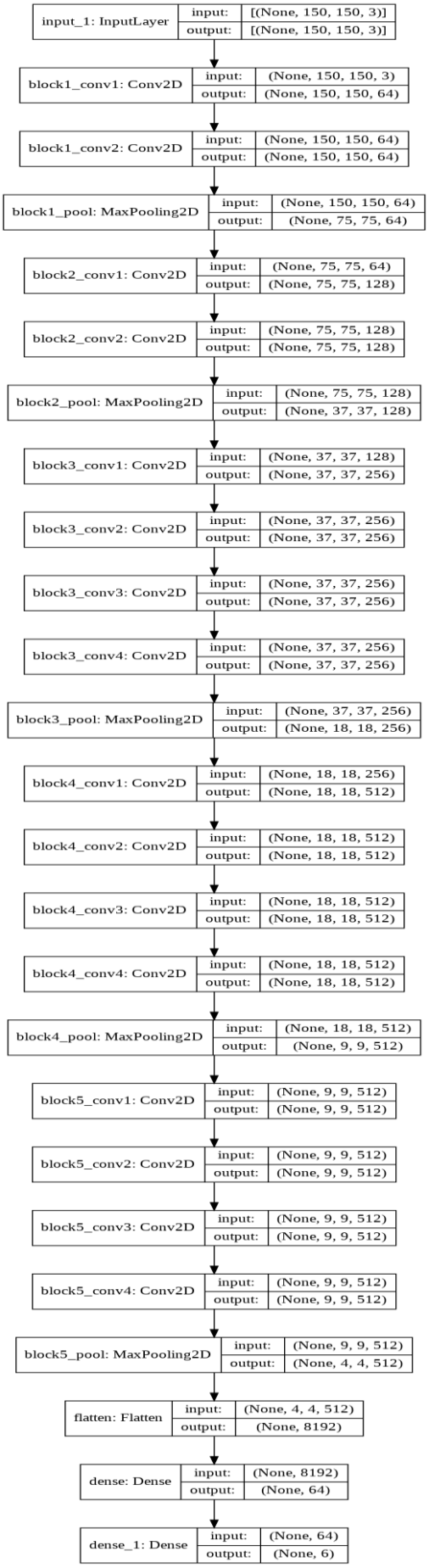
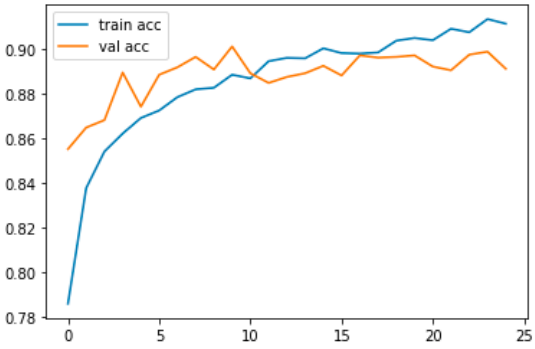


Fig 1.1 – VGG19 Model

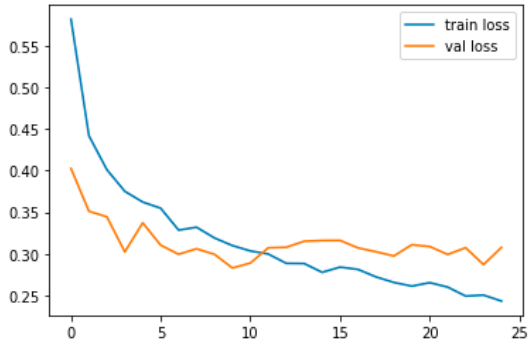
B. 1 Layer Trainable – VGG16

In this model only 1 convolutional layer was kept trainable from last model. This showed improvement as compared to our base model. This model gave training and validation accuracies of 91.06% and 89.10% respectively. Training and validation losses are 24.56% and 30.77% respectively.

Accuracy



Loss



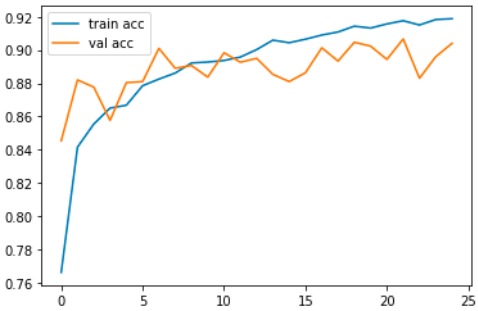
Confusion Matrix

```
[[1001    3    1    2    5  132]
 [   4 1150    0    1    3    8]
 [   6    7 1159  104   51    3]
 [   7    9  164 1025   86    6]
 [   9    5   18   40 1049    7]
 [  35    9    1    5    4 1182]]
```

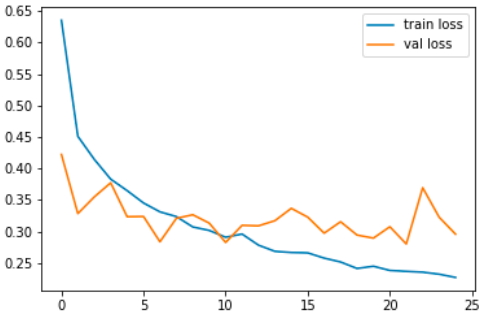
C. 2 Layers Trainable – VGG16

This model gave accuracy of 91.99% of training and 90.4% validation accuracy. Whereas model shows the training loss of 22.66% and 29.56% validation loss. This is almost 0.8% improvement in the performance of the last model.

Accuracy



Loss



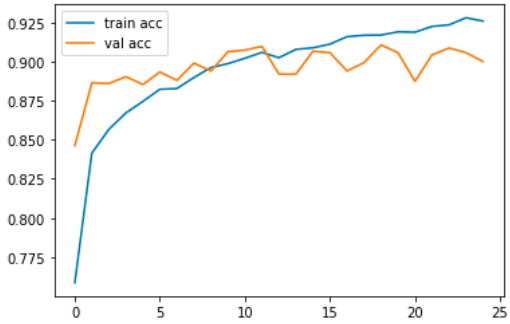
Confusion Matrix

```
[[1069    2    0    2    5   66]
 [   9 1143    3    3    3    5]
 [   5    6 1107  166   45    1]
 [  10    2   95 1111   77    2]
 [  12    3   27   56 1021    9]
 [  80    9    1    3    5 1138]]
```

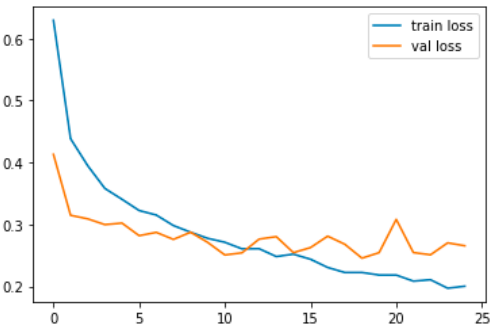
D. 2 Layers Trainable with SGD – VGG16

In this model the optimizer was changed from the last model. Its training accuracy showed little improvement of 0.66% but validation accuracy got minor degraded to 90%. Training loss got reduced to 19.26% and validation loss of 26.56%

Accuracy



Loss



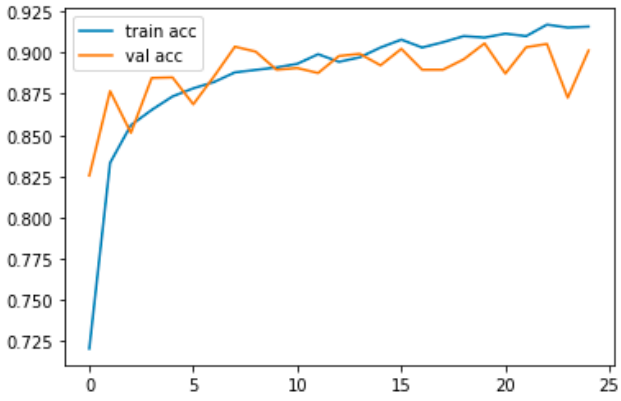
Confusion Matrix

```
[[1026    5    1    3    1  108]
 [   2 1150    0    4    1    9]
 [   3    5 1108  186   24    4]
 [   6    7   89 1145   46    4]
 [   8   10   41   71  983   15]
 [  38   11    1    3    2 1181]]
```

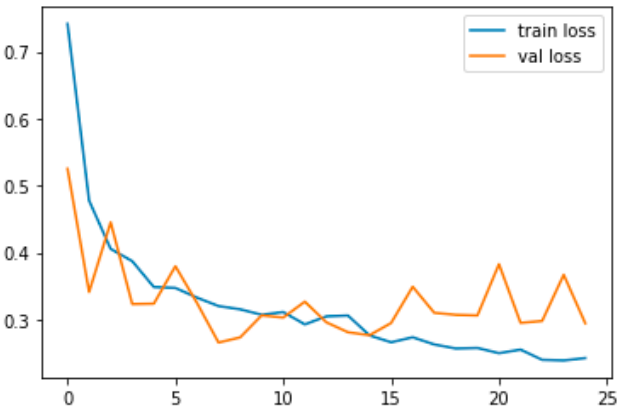
E. *3 Layers Trainable – VGG16*

In this model 3 layers were trained but this model showed bad results from the model where 2 layers were trained. Its training and validation accuracy dropped to 91.70% and 90.13% respectively. Its training loss increased to 24.74% and validation loss increased to 29.46%.

Accuracy



Loss



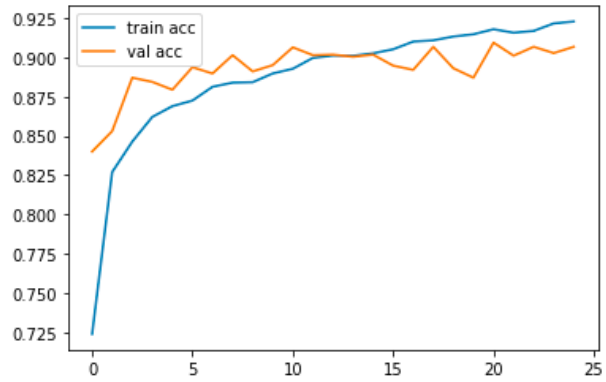
Confusion Matrix

```
[[1018    4    4    2    4  112]
 [   4 1142    4    4    2   10]
 [   2    3 1183  112   28    2]
 [   3    2  159 1065   62    6]
 [  12    2   40   75  990    9]
 [  50   11    4    3    5 1163]]
```

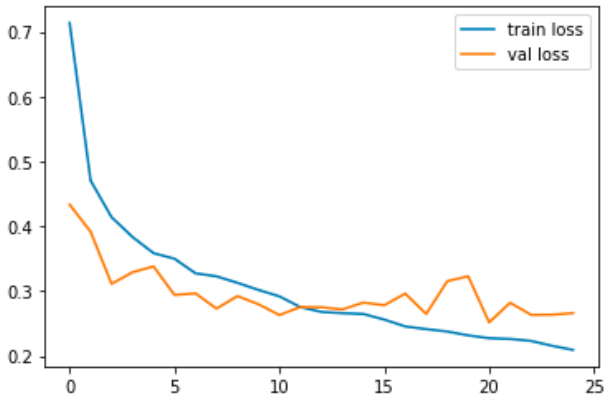
F. *3 Layers Trainable with SGD – VGG19*

In this model VGG19 is used along SGD optimizer. This model showed even better results from model where 2 layers were trained. Training and validation accuracy increased to 92.30% and 90.67% respectively. But its loss is higher than that of 2-layer trained model with SGD optimizer. Its training and validation losses are 21.04% and 26.61% respectively which is much higher than that of 2-layer trained model with SGD optimizer.

Accuracy



Loss



Confusion Matrix

```
[[1060    3    0    0    2   79]
 [   6 1143    2    1    3   11]
 [   6    6 1018  240   53    7]
 [   6    6   70 1152   60    3]
 [   8    4   18   61 1028    9]
 [  74    8    2    3    5 1144]]
```

IV. Discussion and Conclusion

According to the different models trained over different hyper parameters, the model which performed the best is the VGG19 model 3 Layers Trainable along SGD optimizer which achieved the highest accuracy of 92.30% of training and validation accuracy of 90.67%. The second-best model was the VGG16 model with 2 Layers Trainable along SGD optimizer. It showed the training accuracy of 92.65% and validation accuracy of 90%. Despite the fact the training accuracy of both models is almost the same but the validation accuracy of the 3-Layers with VGG19 model is 0.67% higher than that of 2-Layers VGG16 but the loss of VGG16 model was much lower than that of the VGG19. There is possibility that this model can be further improved with different hyperparameters.

V. Github

https://github.com/MuhammadJunaidAkram/Intel-Image-Classification---Assignment-3