**Capstone – Interim Report-Group 1**

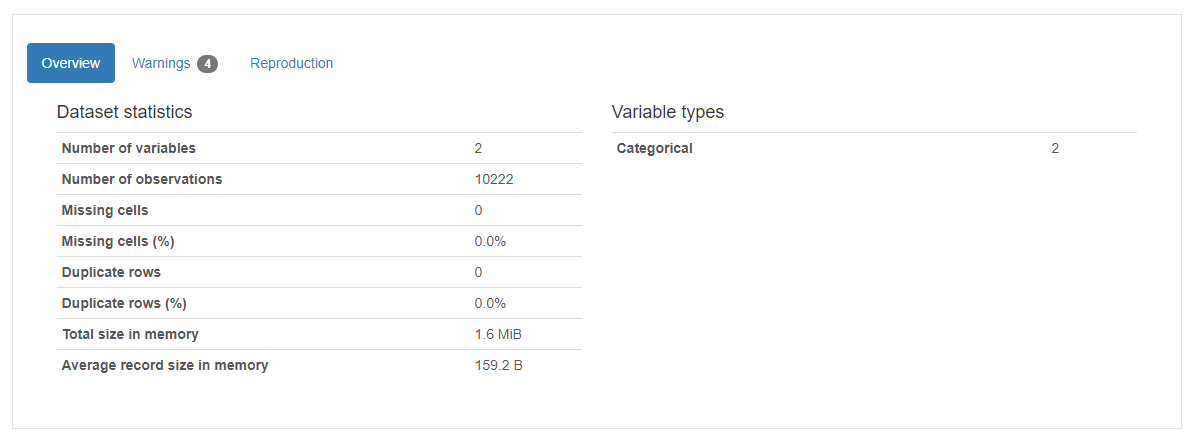
**Problem Statement**:

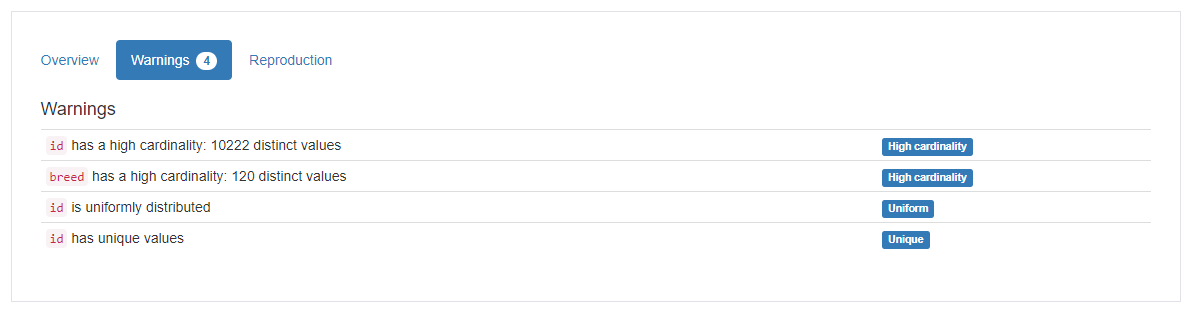
Determining the breed of a dog from its image using deep learning algorithms such as CNNs to extract useful features from the image to help in identifying the breed of the dog.

**Exploratory Data Analysis (EDA) and Early Results:**

Data Overview:

The data is obtained directly from the Kaggle competition for dog breed classification and can be found in the link [here](https://www.kaggle.com/c/dog-breed-identification/data?select=train). The dataset contains 10,222 images of dogs belonging to 120 different breeds. Pandas profiling was used to explore data statistics.

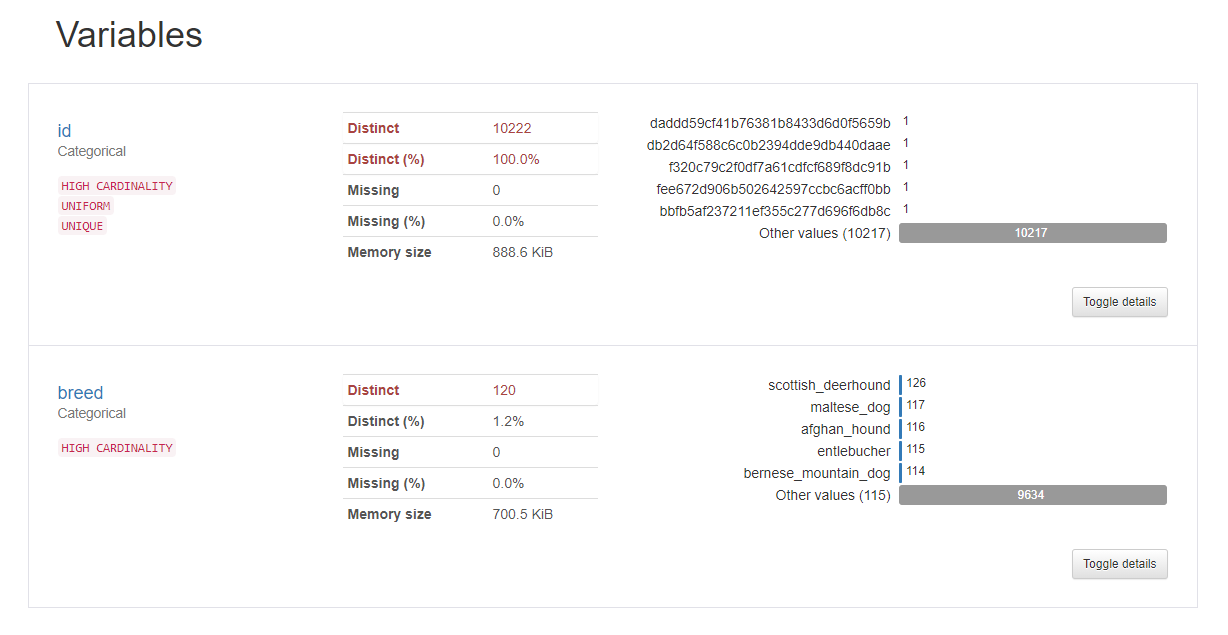




Sample image of an English Foxhound found in the training dataset is below:



The dataset contains two columns namely, the image id and the dog breed. An overview of these two columns can be found in the below report:



The dataset is clean and does not have missing values. All dog ids are unique and there are no duplicates found. Interestingly, the most common breed of dog in the dataset is the Scottish Deerhound which is followed by the Maltese Dog.

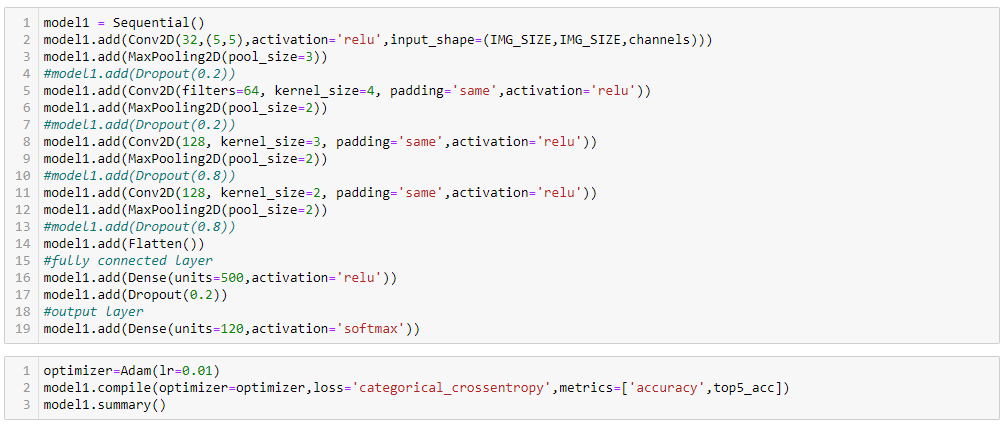
**Model Iterations:**

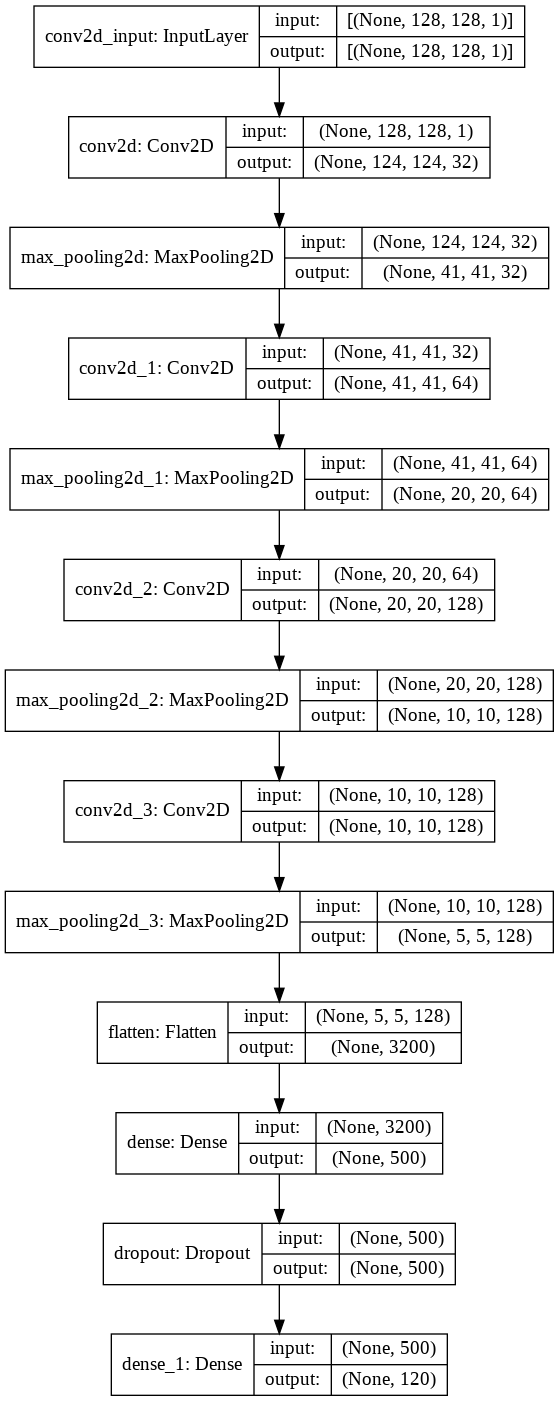
For all the model iterations the following parameters remain the same:  
Optimizer: Adam  
Loss Function: Categorical Crossentropy  
Metrics – Accuracy & Top5 Accuracy  
Batch Size = 32 and Epochs =20

Model iterations along with their details are shown in the table below:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Iteration** | **Model Name** | **Image Augmentation (Y/N)** | **Training  Accuracy** | **Validation  Accuracy** | **Val Top 5  Accuracy** | **ROC-AUC  Score** |
| 1 | Custom CNN Architecture | N | 1.31% | 1.76% | 5.72% | 0.50 |
| 2 | Custom CNN Architecture | Y | 1.24% | 0.99% | 5.80% | 0.49 |
| 3 | VGG-16 with pre-trained weights | N | 99.96% | 25.23% | 60.54% | 0.92 |
| 4 | VGG-16 with pre-trained weights | Y | 1.22% | 1.09% | 5.21% | 0.5 |
| 5 | Resnet-50 using pre-trained weights | N | 1.18% | 1.15% | 5.05% | 0.5 |
| 6 | Resnet-50 using pre-trained weights | Y | 1.20% | 1.16% | 5.09% | 0.5 |
| 7 | Resnet-50 with all layers trainable | N | 5.66% | 5.00% | 19.10% | 0.77 |
| 8 | Mobilenet V2 with pre-trained weights | N | 99.96% | 71.40% | 93.65% | 0.99 |
| 9 | Mobilenet V2 with pre-trained weights | Y | 72.97% | 65.27% | 91.83% | 0.99 |

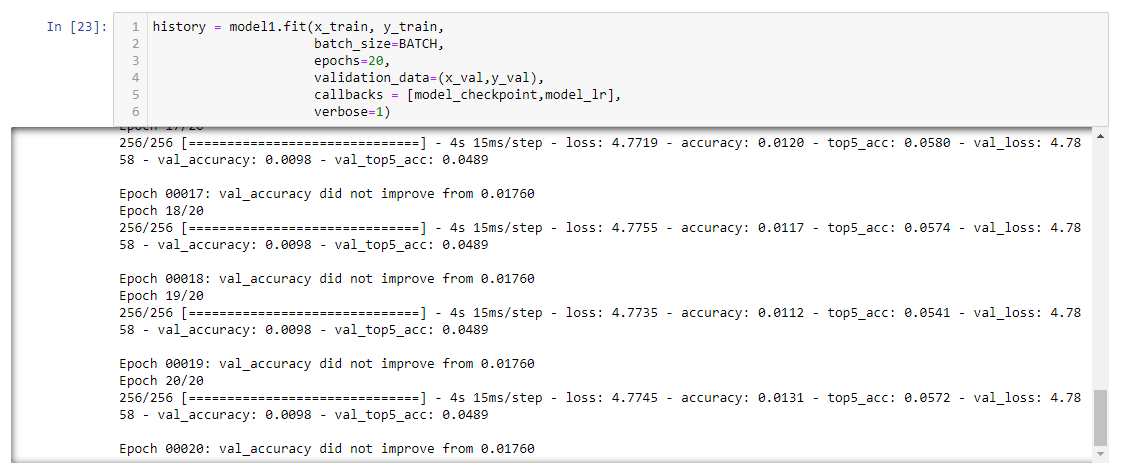
**Iteration1: Custom CNN architecture**  
Model Code:



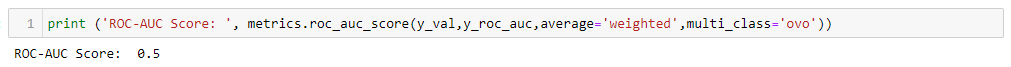
Model Diagram:  
  


A basic CNN architecture was used to get an initial idea of the accuracy one could expect to get with a plain model. With this model, we were able to achieve a train accuracy of 1%, a test accuracy of 0.9% and a top5 accuracy of 5.72%.

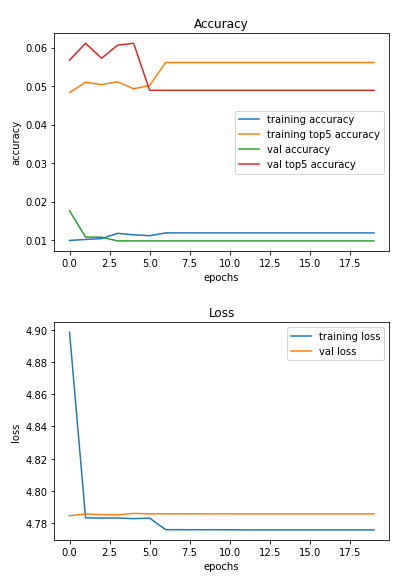
Model Performance:



ROC-AUC Score:

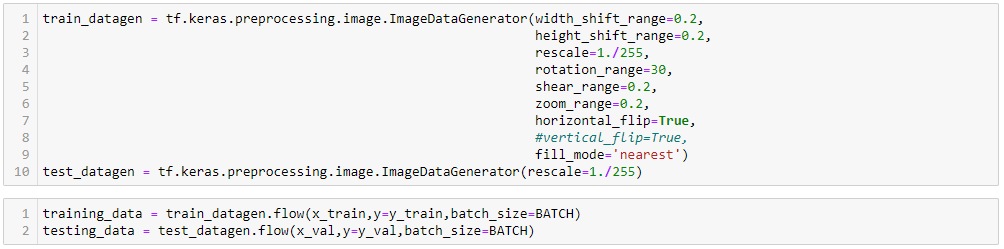


Plot of accuracy & loss across epochs:



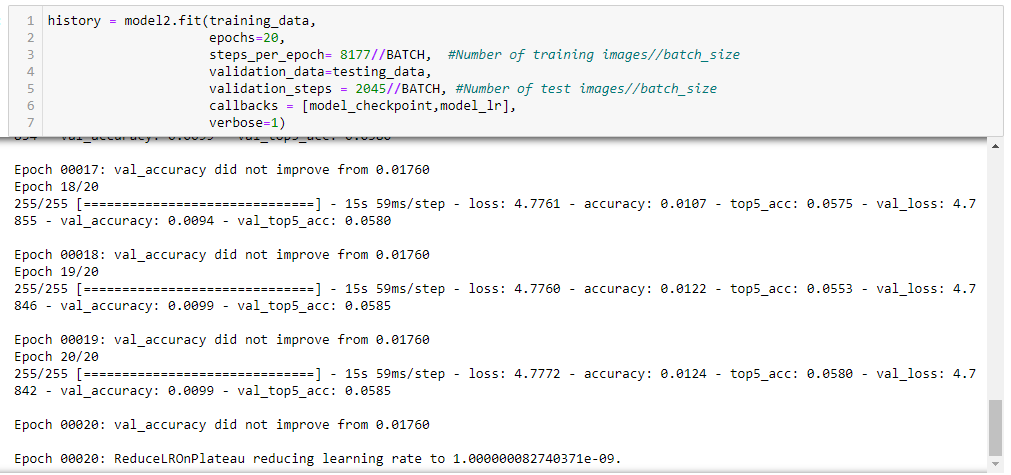
**Iteration2: The same CNN architecture as Iteration1 with Image Augmentation**

Image Augmentation Parameters:

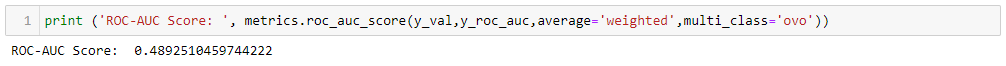


The image above shows the parameters we have used for augmenting the training images. We observed a very slight increase in val\_accuracy to 0.99%. The images below showcase the accuracy obtained at the end of the epochs along with the plot showing loss and accuracy change across epochs.

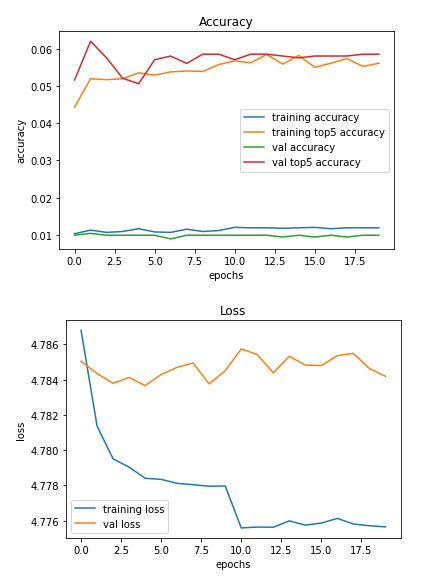
Model Performance:



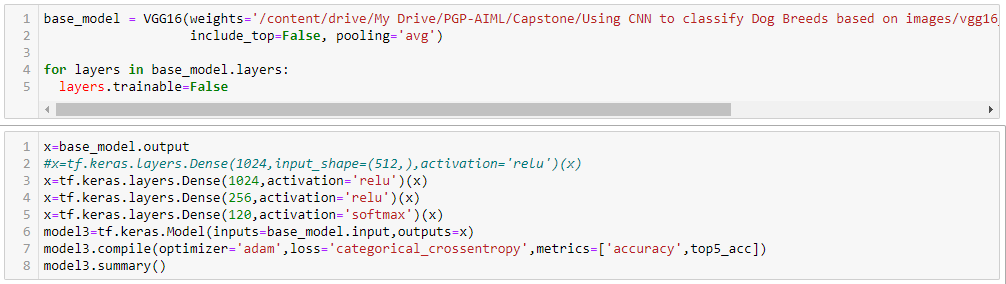
ROC-AUC score:



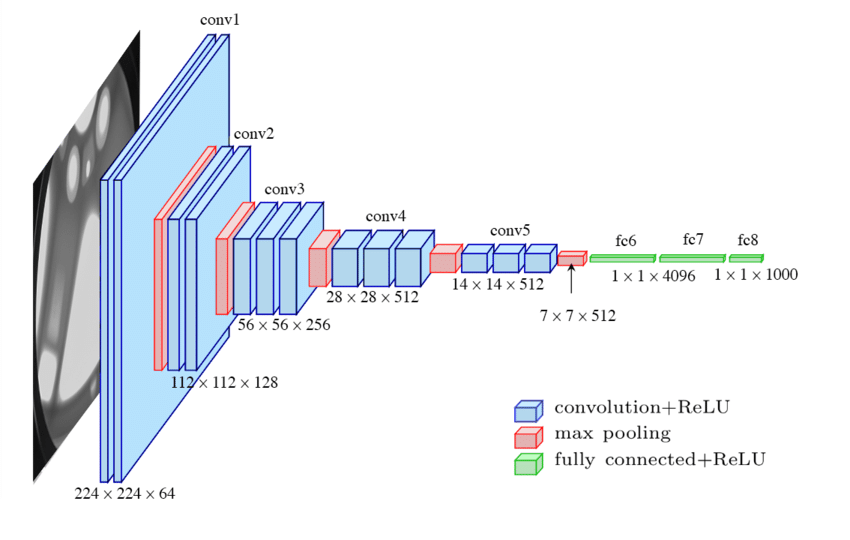
Plot of accuracy & loss across epochs:

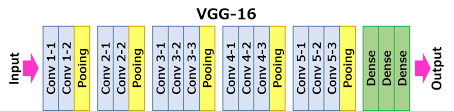


**Iteration3: VGG-16 with pre-trained weights based on ImageNet dataset**  
  
Model Code:



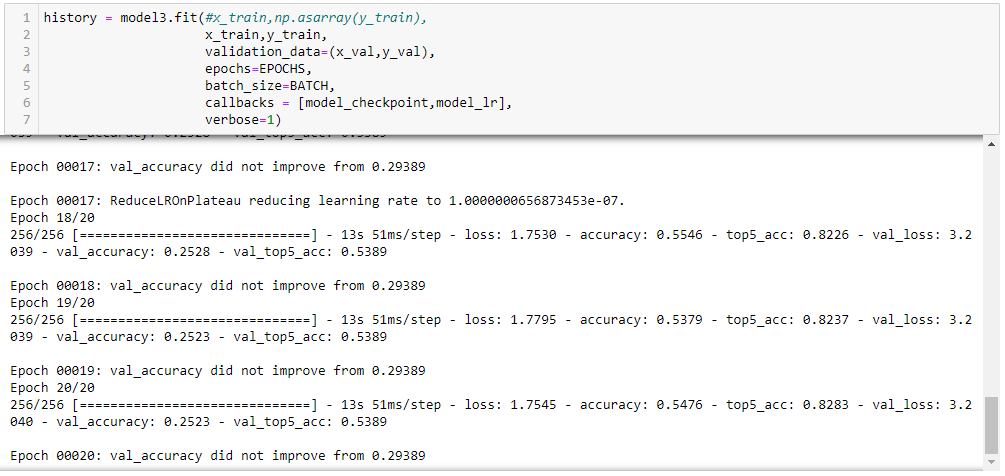
Model Diagram:

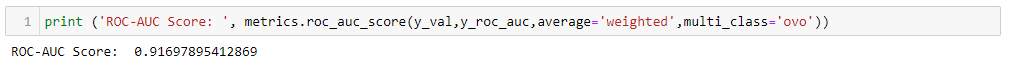


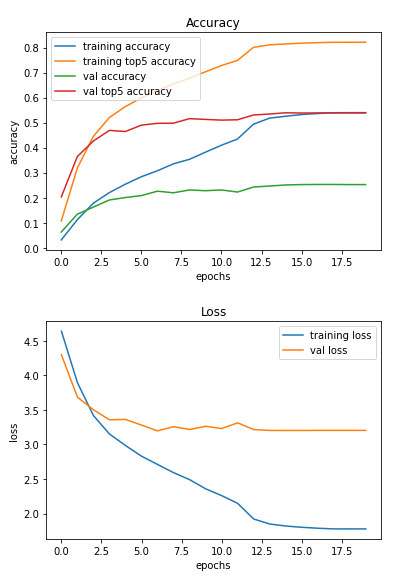


We have used ‘adam’ as our optimizer and ‘categorical-crossentropy’ as our loss function, the model was trained to improve accuracy.  
The VGG16 architecture yielded in significant improvement in accuracy as compared to our custom CNN architecture used in the previous iterations. The train accuracy was around 65% and the test accuracy was about 31% at then end of 20 epochs. Although the model is overfit, using image augmentation and other techniques can improve accuracy while eliminating overfitting.

Model Performance:

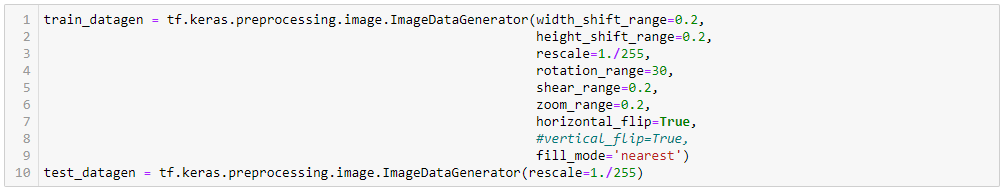


ROC-AUC Score:  
  


Plot of accuracy & loss across epochs:  


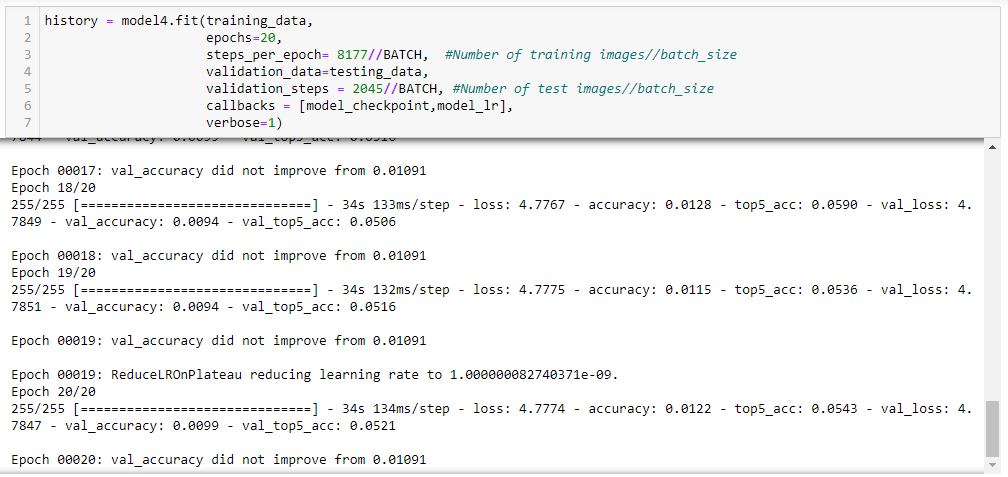
The above plot shows that the model was quite stable in learning across epochs but it can also be seen that the model is highly overfit as well.

**Iteration4: VGG-16 with pre-trained weights along with Image Augmentation**

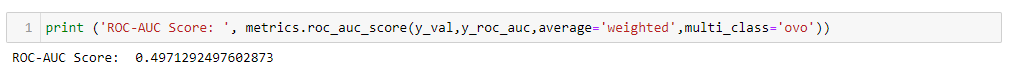


The same model as that in iteration 3 was used. And image augmentation was applied to the training data based on the parameters in the image above.

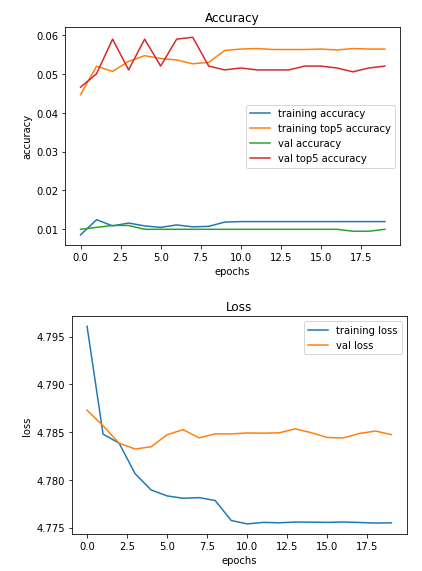
Model Performance



ROC-AUC Score:

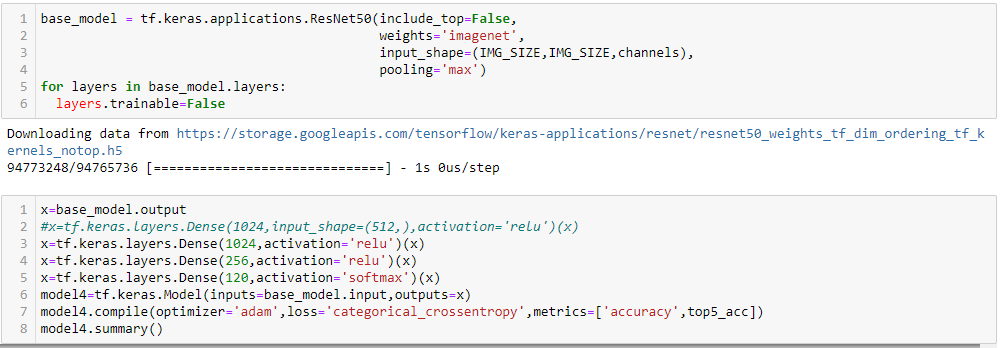


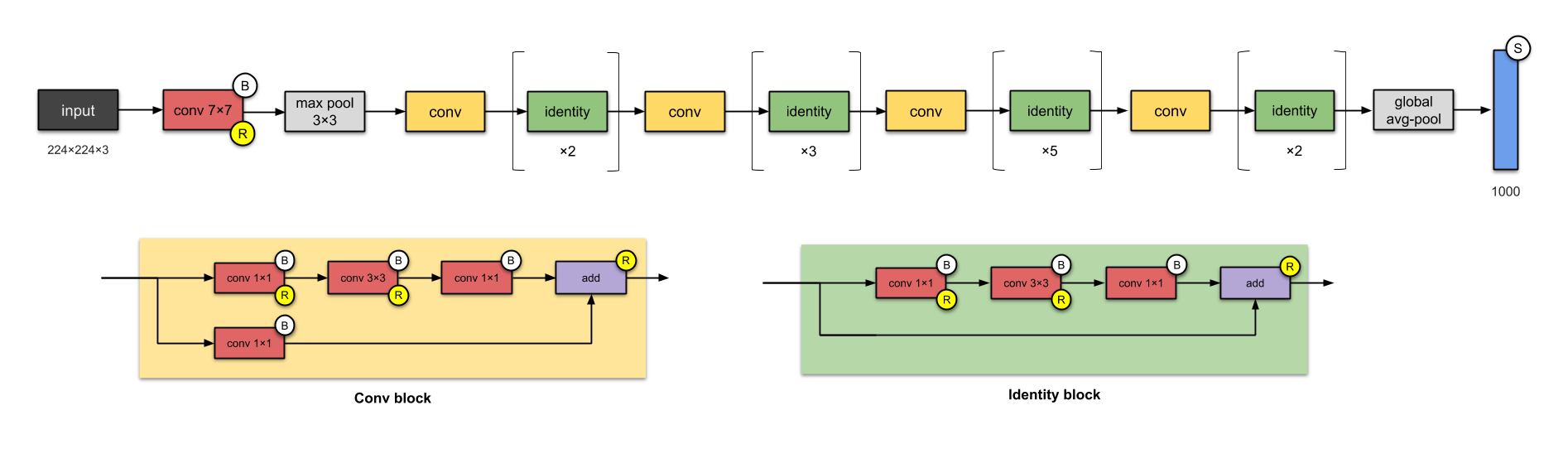
Plot of accuracy & loss across epochs:



**Iteration5: Resnet50 with pre-trained weights**

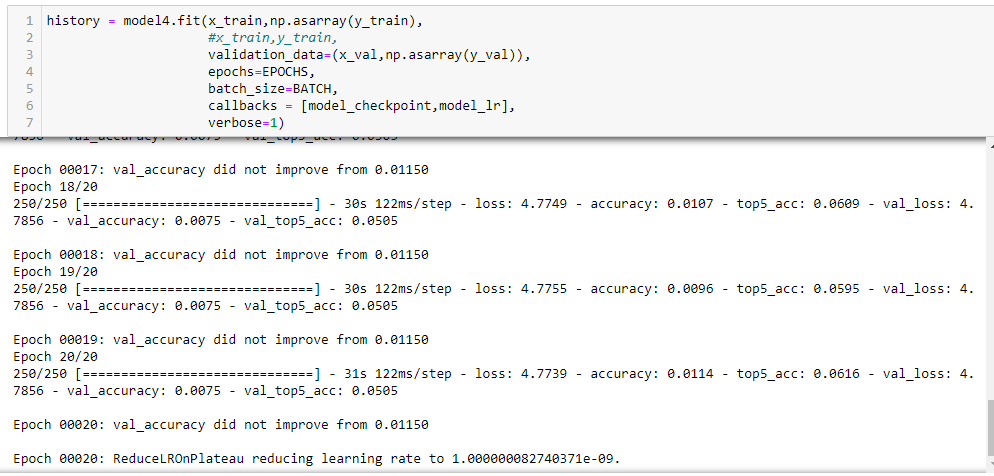
Model Code:



Model Diagram:

We have used ‘adam’ as our optimizer with ‘categorical crossentropy’ as our loss function. The model was trained to improve accuracy. The Resnet50 model performed similarly to our custom CNN architecture, with a training accuracy of 1.2% and a test accuracy of 1.15% with a validation top5 accuracy of 5%.

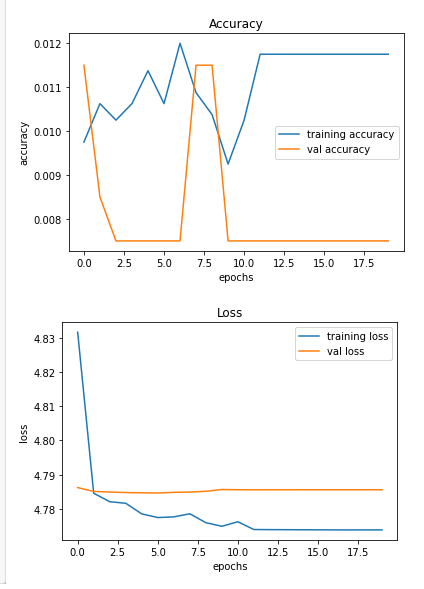
Model Performance:



ROC-AUC Score:



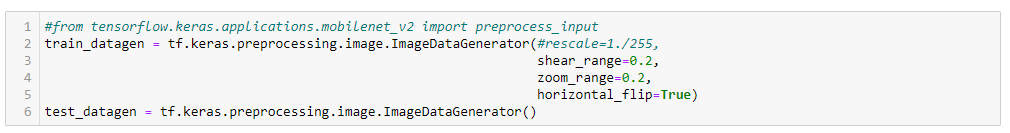
Plot of accuracy & loss across epochs:



The model was quite unstable across epochs as there are extreme fluctuations in accuracy.

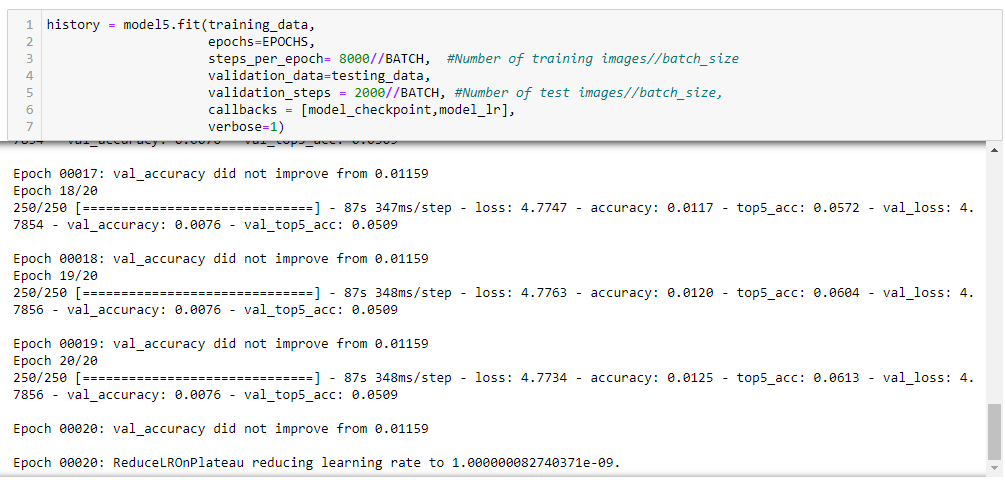
**Iteration 6: Resnet 50 with pre-trained weights along with image augmentation**

Image Augmentation Parameters:

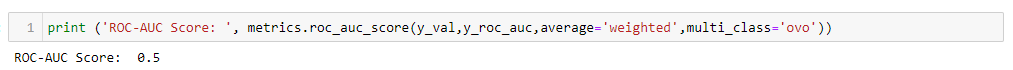


In this iteration we use the same Resnet50 model as in iteration 5 but the training images are augmented using ImageDataGenerator with the parameters shown in the image above. We are able to achieve a train accuracy of 1.20%, a test accuracy of 1.16% and a validation top-5 accuracy of 5.09%.

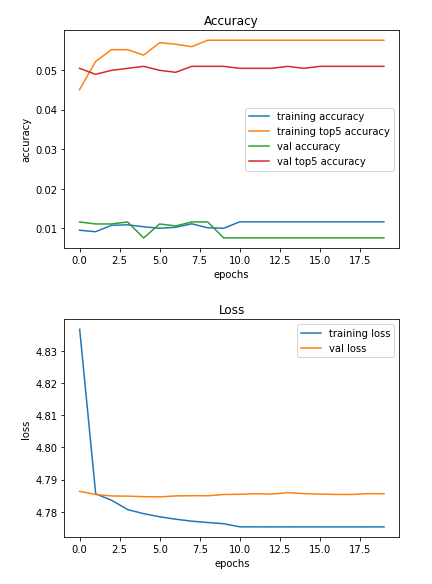
Model Performance:



ROC-AUC Score:



Plot of accuracy & loss across epochs:



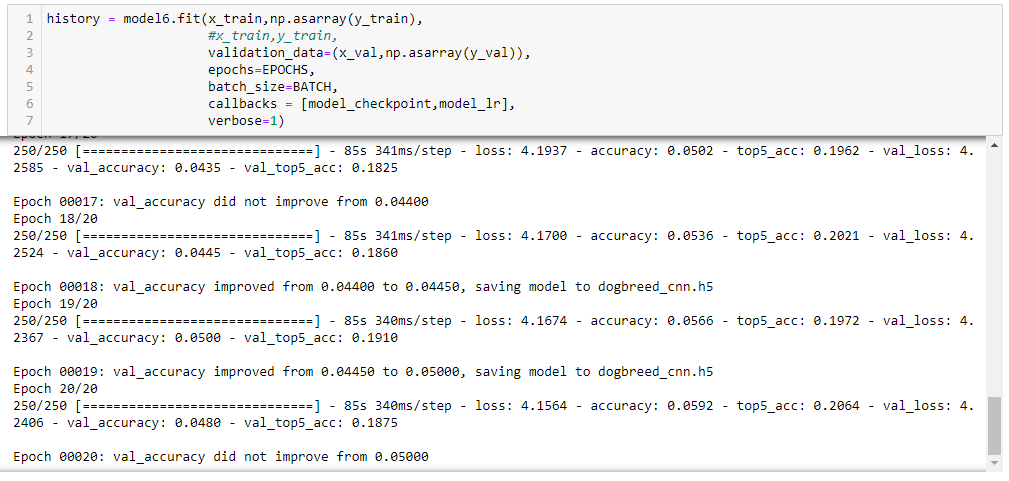
**Iteration 7: Resnet 50 model but keeping all layers as trainable**

Model Code:

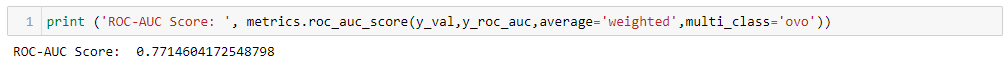
****

Keeping other parameters as the same as iteration 4 and making all layers trainable, the test accuracy in this scenario is 5% and the validation top-5 accuracy improves to 19%. On the other hand, the training accuracy maxed out at 6%.

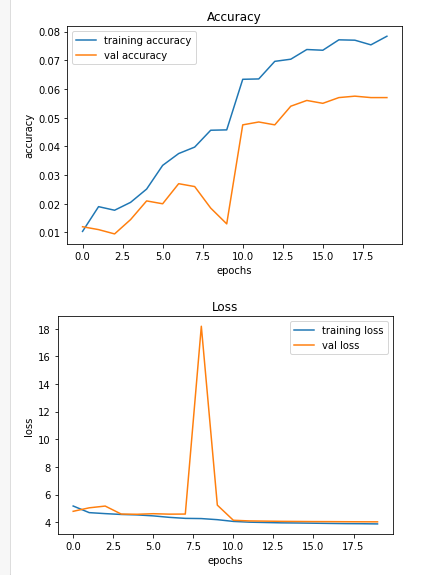
Model Performance:



ROC-AUC Score:

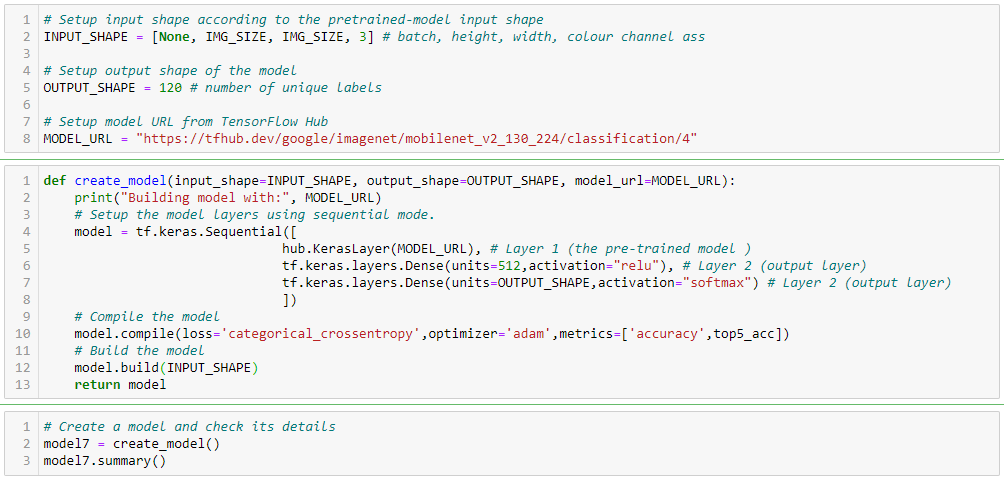


Plot of accuracy & loss across epochs:



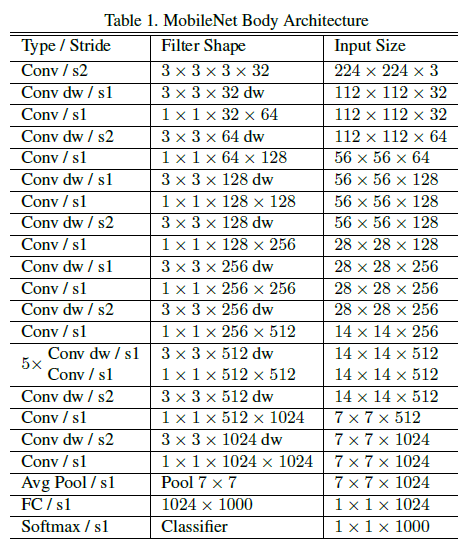
The learning of the model here is relatively more stable than the one in iteration 4, as can be seen by the absence of extreme fluctuations in accuracy across epochs.

**Iteration 8: Mobilenet V2 model with pre-trained weights**

Model Code: 

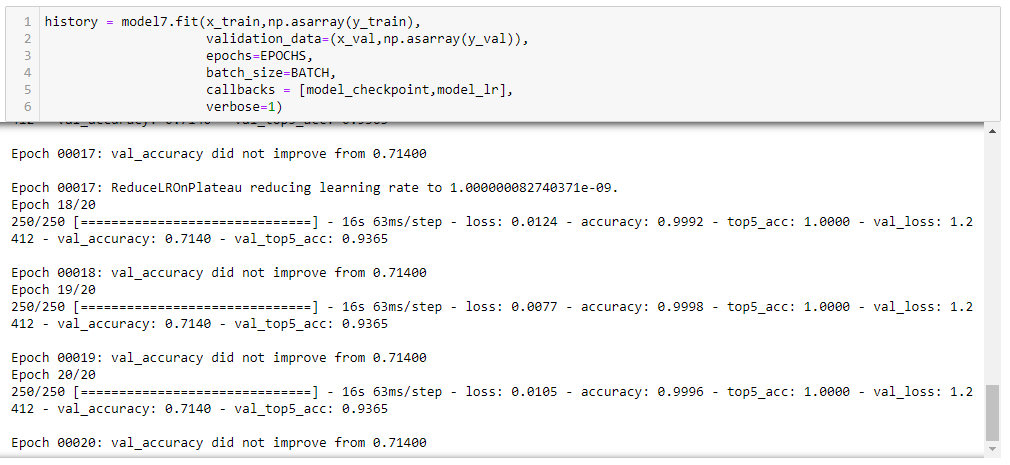
The model can be found in the link [here.](https://tfhub.dev/google/imagenet/mobilenet_v2_130_224/classification/4) The mobilenet model is famously known for its small model size and complexity. The architecture is based on depth wise convolution followed by a pointwise convolution. And in our case, it resulted in the best performance out of all the models we have used.

Model Architecture:

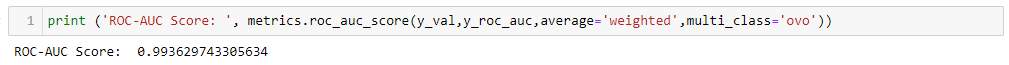


Using the mobilenet model we were able to achieve a test accuracy of 72% and a validation top-5 accuracy of 94% . But the model is overfit as the training accuracy is very high, 99.9%. In the next iteration we will try to reduce the overfit nature of this model by using image augmentation. The model performance can be found below.

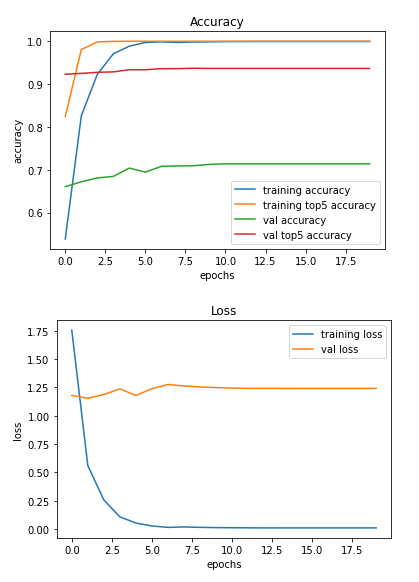
Model Performance:



ROC-AUC Score:

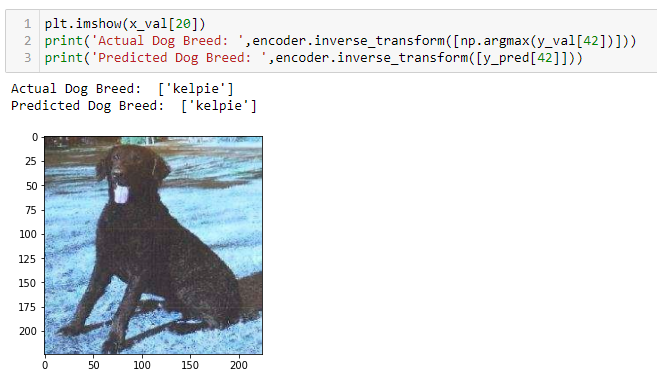


Plot of accuracy & loss across epochs:



The test accuracy plateaued around the 7th epoch, and the model became highly overfit around the 3rd epoch. On the other hand, the model is quite stable as the accuracy and loss curves ae smooth across epochs.

Sample Model Prediction:

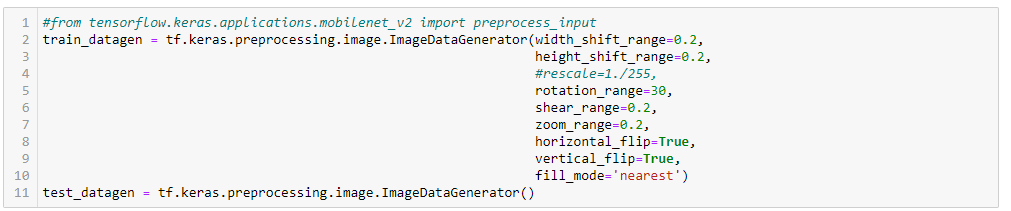


The model has correctly predicted the breed of the dog above as Kelpie.

**Iteration 9: MobileNet model with pre-trained weights and image augmentation**

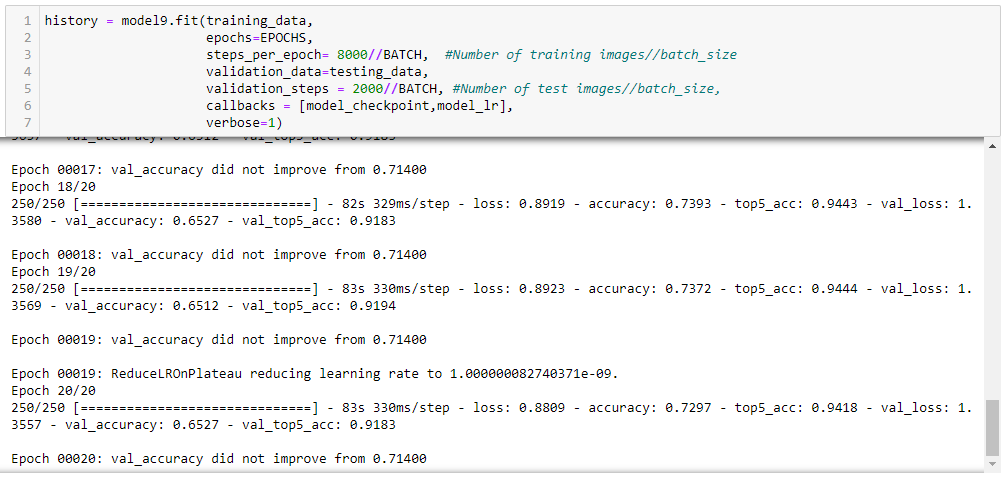
Model Code and Architecture is same as that in iteration 8. Only difference here is that we have added image augmentation to reduce overfitting.

Image Augmentation Code:

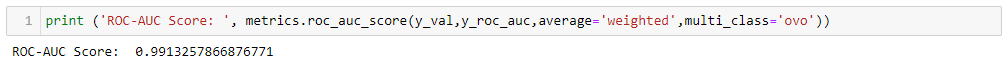


The parameters used in image augmentation can be found in the image above. The result is that, the test accuracy has slightly dipped to 66%, but the overfit nature of the model has been greatly reduced as the training accuracy at the end of 20 epochs is about 74%.  
Which is an improvement even though there was a slight decrease in accuracy.

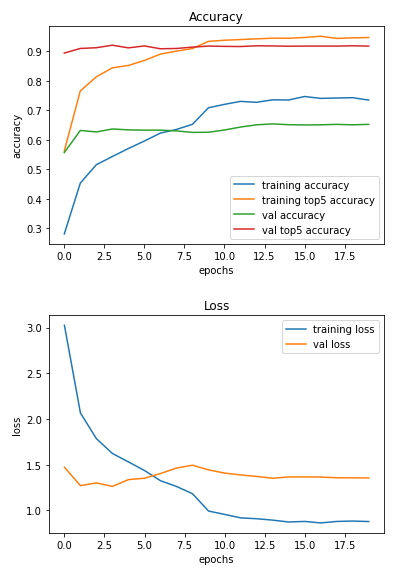
Model Performance:



ROC-AUC Score:



Plot of accuracy & loss across epochs:

****

The model was not as stable as that in iteration 8. This can be attributed to the image augmentation playing a role in diversifying the training data. An important observation is that, while in the previous iteration the training accuracy shot up exponentially around the 3rd epoch, in this iteration the growth is more controlled and linear.

**Inferences:**

The mobilenet V2 model outperformed the rest of the architectures significantly. It showed a promising top-5 accuracy of 92% in the last iteration, with a top-1 accuracy of 65%. In identifying dog-breeds due to instances of high diversity within a breed and low diversity between breeds, it is difficult for the model to predict the correct breed on the first go, which is why top-5 accuracy is a better metric for accuracy in this specific case.

**Tentative Algorithms to Explore:**

* Inception
* AlexNet
* Hyperparameter tuning of the mobilenet V2 model

**References:**

1. Spady, T.C., Ostrander, E.A.: Canine behavioral genetics: Pointing out the phenotypes and herding up the genes. AJHG 82(1), 10–18 (2008)
2. Branson, S., Wah, C., Schroff, F., Babenko, B., Welinder, P., Perona, P., Belongie, S.: Visual Recognition with Humans in the Loop. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part IV. LNCS, vol. 6314, pp. 438–451. Springer, Heidelberg (2010)
3. Nilsback, M.E., Zisserman, A.: Automated flower classification over a large number of classes. In: Proc. 6th Indian Conf. on Computer Vision, Graphics and Image Processing, pp. 722–729 (2008)
4. Farrell, R., Oza, O., Zhang, N., Morariu, V., Darrell, T., Davis, L.: Birdlets: Subordinate categorization using volumetric primitives and pose-normalized appearance. In: Proc. ICCV (2011)
5. Belhumeur, P.N., Chen, D., Feiner, S.K., Jacobs, D.W., Kress, W.J., Ling, H., Lopez, I., Ramamoorthi, R., Sheorey, S., White, S., Zhang, L.: Searching the World’s Herbaria: A System for Visual Identification of Plant Species. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part IV. LNCS, vol. 5305, pp. 116–129. Springer, Heidelberg (2008)
6. Belhumeur, P.N., Jacobs, D.W., Kriegman, D.J., Kumar, N.: Localizing parts of faces using a consensus of exemplars. In: Proc. CVPR (2011)
7. Csurka, G., Dance, C.R., Fan, L., Willamowski, J.: Visual categorization with bags of keypoints. In: Work. on Stat. Learning in Comp. Vis., ECCV, pp. 1–22 (2004)
8. Jurie, F., Triggs, B.: Creating efficient codebooks for visual recognition. In: Proc. ICCV (2005)
9. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: Proc. CVPR, pp. 2169–2178 (2006)
10. Gehler, P., Nowozin, S.: On feature combination for multiclass object classification. In: Proc. CVPR (2009)
11. Wang, Z., Hu, Y., Chia, L.-T.: Image-to-Class Distance Metric Learning for Image Classification. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part I. LNCS, vol. 6311, pp. 706–719. Springer, Heidelberg (2010)
12. Deselaers, T., Ferrari, V.: Visual and semantic similarity in imagenet. In: Proc. CVPR (2011)
13. Sadeghi, M.A., Farhadi, A.: Recognition using visual phrases. In: Proc. CVPR (2011)
14. Yao, B., Khosla, A., Fei-Fei, L.: Combining randomization and discrimination for fine-grained image categorization. In: Proc. CVPR (2011)
15. Bourdev, L., Maji, S., Brox, T., Malik, J.: Detecting People Using Mutually Consistent Poselet Activations. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part VI. LNCS, vol. 6316, pp. 168–181. Springer, Heidelberg (2010)
16. Parkhi, O., Vedaldi, A., Zisserman, A., Jawahar, C.: Cats and dogs. In: Proc. CVPR (2012)
17. Viola, P., Jones, M.: Robust real-time object detection. IJCV 57, 137–154 (2001)
18. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: Proc. CVPR, vol. 1, pp. 886–893 (2005)
19. Parkhi, O., Vedaldi, A., Jawahar, C.V., Zisserman, A.: The truth about cats and dogs. In: Proc. ICCV (2011)
20. Cristinacce, D., Cootes, T.: Feature detection and tracking with constrained local models. In: Proc. BMVC, pp. 929–938 (2006)
21. Milborrow, S., Nicolls, F.: Locating Facial Features with an Extended Active Shape Model. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part IV. LNCS, vol. 5305, pp. 504–513. Springer, Heidelberg (2008)
22. Saragih, J.M., Lucey, S., Cohn, J.F.: Face alignment through subspace constrained mean-shifts. In: Proc. ICCV (2009)
23. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. IJCV 20 (2004)
24. Kumar, N., Berg, A.C., Belhumeur, P.N., Nayar, S.K.: Attribute and simile classifiers for face verification. In: Proc. ICCV (2009)
25. Yin, Q., Tang, X., Sun, J.: An associate-predict model for face recognition. In: Proc. CVPR, pp. 497–504 (2011)
26. Arca, S., Campadelli, P., Lanzarotti, R.: A face recognition system based on automatically determined facial fiducial points. Pattern Recognition 39, 432–443 (2006)
27. Campadelli, P., Lanzarotti, R., Lipori, G.: Precise eye localization through a general-to-specific model definition. In: Proc. BMVC (2006)
28. Vedaldi, A., Gulshan, V., Varma, M., Zisserman, A.: Multiple kernels for object detection. In: Proc. ICCV, pp. 606–613 (2009)
29. Wang, J., Yang, J., Yu, K., Lv, F., Huang, T., Gong, Y.: Locality-constrained linear coding for image classification. In: Proc. CVPR, pp. 3360–3367 (2009)
30. Hu, T.; Qi, H. G.; Huang, Q. M.; Lu, Y. See better before looking closer: Weakly supervised data augmentation network for fine-grained visual classification. arXiv preprint arXiv:1901.09891, 2019.
31. Krause, J.; Stark, M.; Deng, J.; L. Fei-Fei. 3D object representations for fine-grained categorization. In: Proceedings of the IEEE International Conference on Computer Vision Workshops, 554–561, 2013.
32. 32.Maji, S.; Rahtu, E.; Kannala, J.; Blaschko, M.; Vedaldi, A. Fine-grained visual classification of aircraft. arXiv preprint arXiv:1306.5151, 2013.
33. Nilsback, M.; Zisserman, A. Automated flower classification over a large number of classes. In: Proceedings of the 6th Indian Conference on Computer Vision, Graphics & Image Processing, 722–729, 2008.
34. Deng, J.; Dong, W.; Socher, R.; Li, L.; Li, K.; Fei-Fei, L. ImageNet: A large-scale hierarchical image database. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 248–255, 2009.
35. Everingham, M.; van Gool, L.; Williams, C. K. I.; Winn, J.; Zisserman, A. The pascal visual object classes (VOC) challenge. International Journal of Computer Vision Vol. 88, No. 2, 303–338, 2010. [44] Lin, T.; Maire, M.; Belongie, S.; Bourdev, L.; Girshick, R.; Hays, J.; Perona, P.; Ramanan, D.; Zitnick, C. L.; Doll´ar, P. Microsoft COCO: Common objects in context. arXiv preprint arXiv:1405.0312, 2014.
36. Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classiﬁcation with deep convolutional neural net- works. In Advances in neural information processing sys- tems (pp. 1097–1105).