Huy Doan

In this assignment, I apply transfer learning to a pre-trained CNN for binary classification problem. The dataset include 4,000 images of each category for training and 1,000 images of each category for testing.

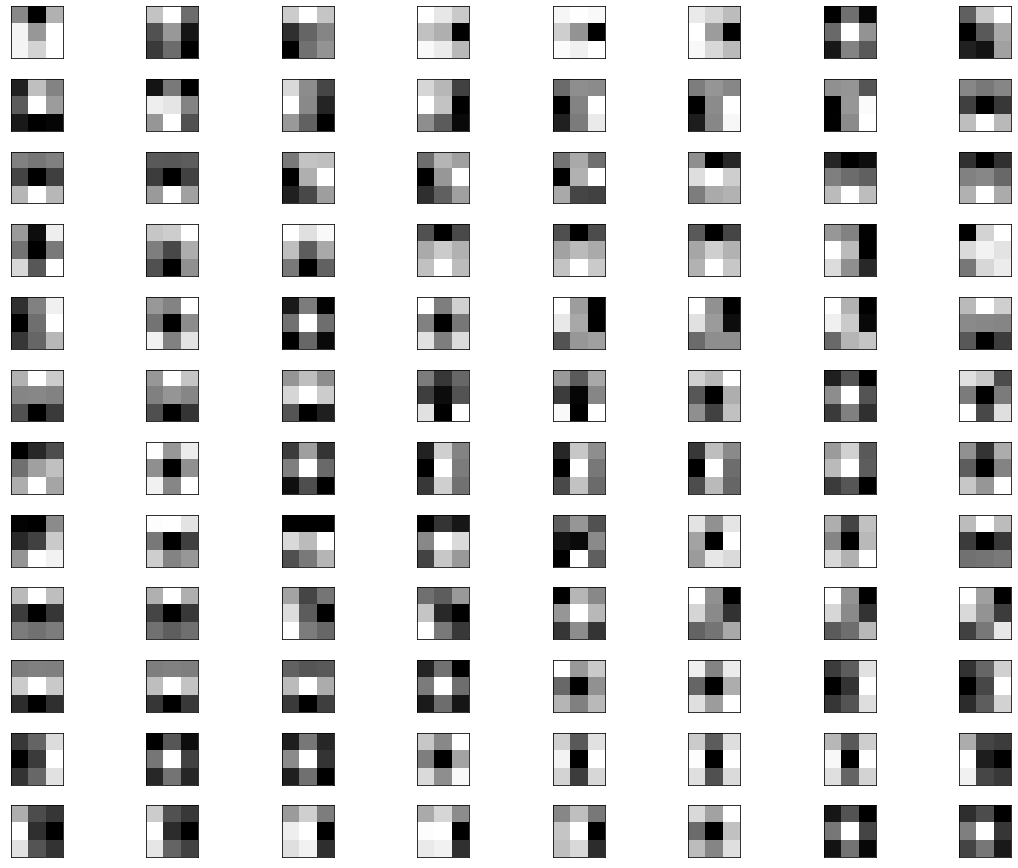
Firstly I loaded the pre-trained “InceptionResNetV2” model. The below image is pre-trained model structure.





There are 54,276,192 trainable parameters at total in this model.

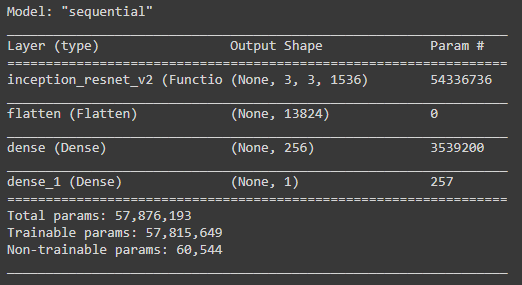
I visualized first layer filters from the model. To be more clarified, I visualized individual channel of each filter.



In the first layer, there are 32 3x3 filters at total which are used to identify different features present in an image like edges, vertical lines, horizontal lines, bends, etc.

To preprocess images from dataset, I divided their intensities by 255 to scale them down to be in range [0,1]

In the next step, I create transfer head for transfer learning. I freeze and reuse the weights of the pre-trained inception-resnetv2 model and add a new classification he. The below image is structure of new model.

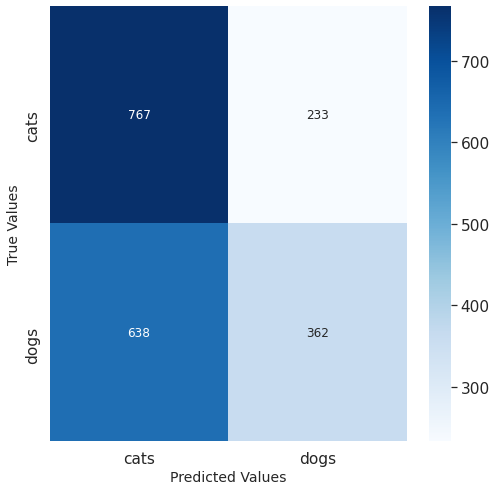


I used model.evaluate() to evaluate model on test set.

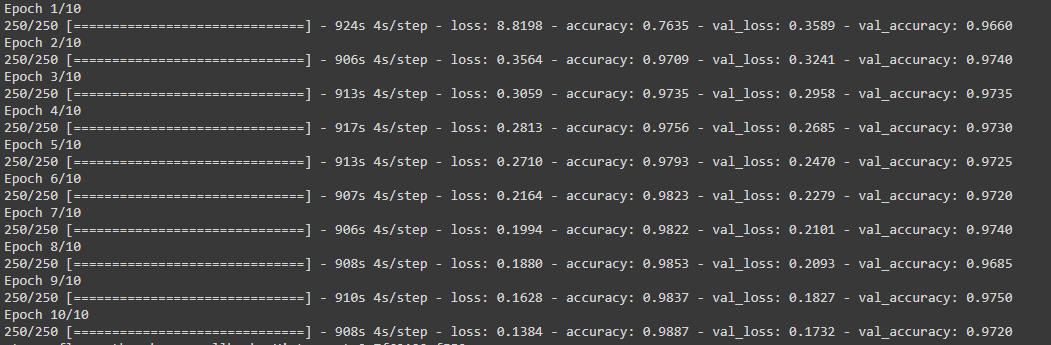


We can see that the new model only provide accuracy about 0.69 which is very low before training.

This is confusion matrix of test set when I evaluated by new model before training:



After that, I trained new model from available dataset with Adam optimizer at running rate 0.001. I ran this model for 10 epochs with early stopping if the model was converged.

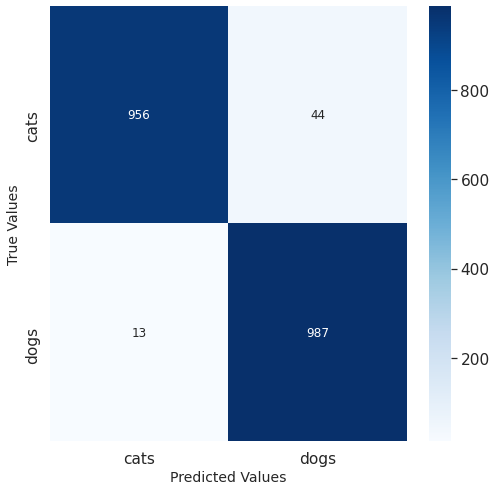


We can see the loss and validation loss from the model decreased after each epoch and accuracy increase contemporarily.

Then I evaluated again on test dataset and it yielded accuracy about 0.97.

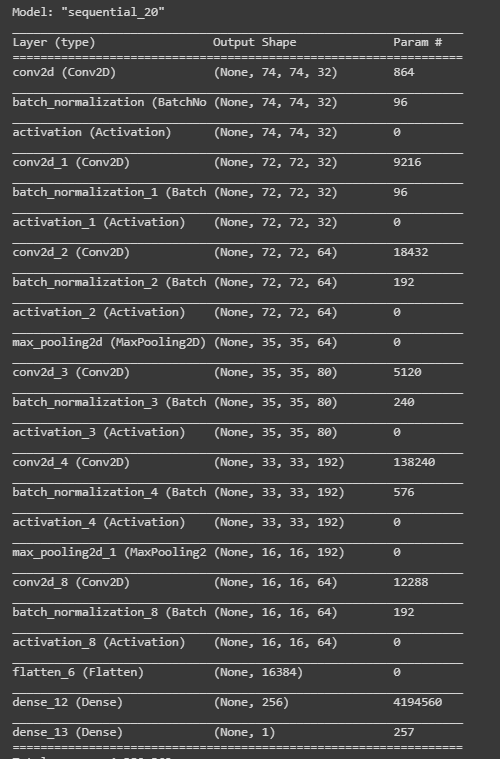


This is accuracy that I got from test dataset tested on the new model after training:



Sub-network:

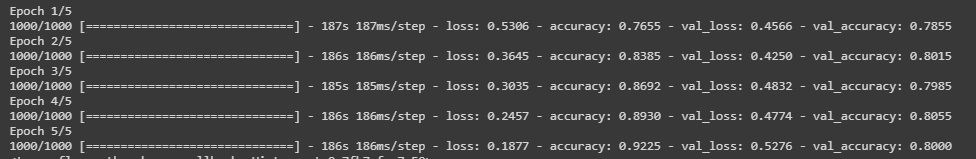
Then I created sub network with only 21 first layers from pre-trained network. I froze the weights of this sub-network and attach a “transfer head” as before then I got model as below:





There are 4,194,817 trainable parameters in this model which is very less than original previous model.

After that, I trained new model from available dataset with Adam optimizer at running rate 0.001. I ran this model for 10 epochs with early stopping if the model was converged.

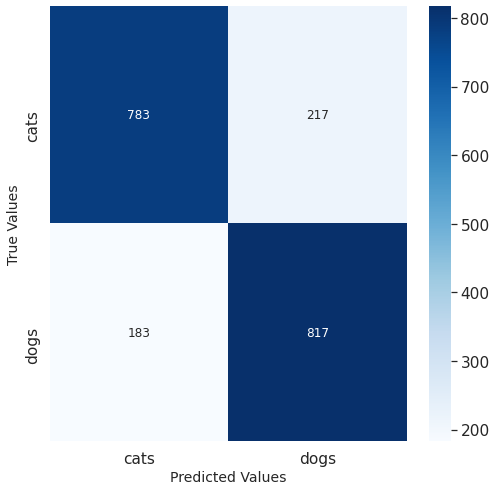


We can see that although the loss is decreasing and accuracy increasing on training set. However, the validation loss and accuracy were fluctuating.



I run test set on this model after training then I got about 0.8 accuracy.

This is confusion matrix from test dataset evaluated from model after training:



For comparison between 2 models, the first trained model is more efficient and stable to train data than this sub-network. The first trained model decreased both training loss and validation loss when increasing accuracy of both train and validation dataset as expected. While sub-network met difficulties to do that. I think the reason is the depth of the network and number of trainable parameters is really important in training data.