cs577 Assignment 2: Report

Yuanxing Cheng, A20453410, CS577-f22
Department of Mathematics
Illinois Institute of Technology

November 11, 2022

1 Binary Classification

Problem statement

Binary classification for cat and dog.

Proposed solution

The data is retrieved inside google colab notebook. The data splitting is also done there. Once done, we continue training a custom CNN. To improve, we adjust the epoch number and batch number as a simple hyper tuning.

To visualize the performance, I plot all the activation of some input and also plot the filters learned in training using gradient ascent.

Finally I use VGG16 as base convolution layer and train only top full connected layers with certain data augmentation to reduce overfitting. Then I unfreeze the last convolution base and continue training.

Implementation details

Custome CNN with batch size 20 and 30 epoch. The optimizer is rmsprop with learning rate 2e-5.

Layer	output shape	params used
conv 2d	148,148,32	896
max pooling 2d	74,74,32	0
conv 2d	72,72,64	18496
max pooling	36,36,64	0
conv 2d	34,34,128	73856
max pooling	17,17,128	0
conv 2d	15,15,128	147584
batch normalizing	15,15,128	512
flatten	28800	0
dense	512	14746112
dense	1	512

Then we see the result as follows:
So I increaes the batch size to 40 and use only 20 epoch and run again. Here's the result.

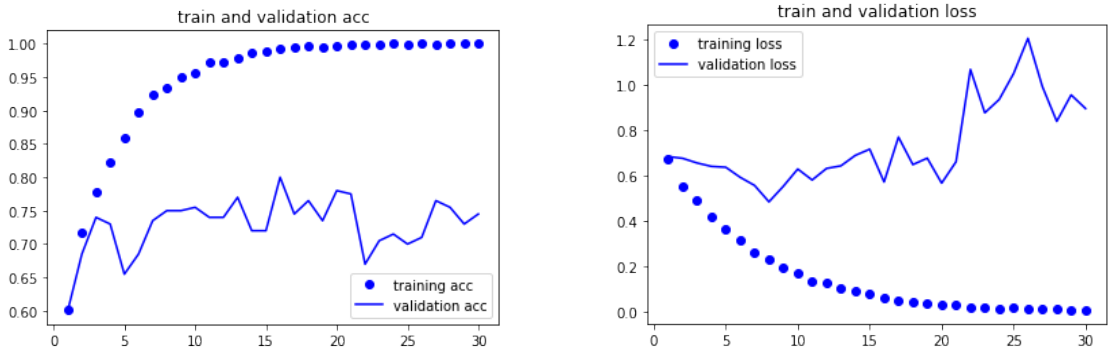


Figure 1: *first try*

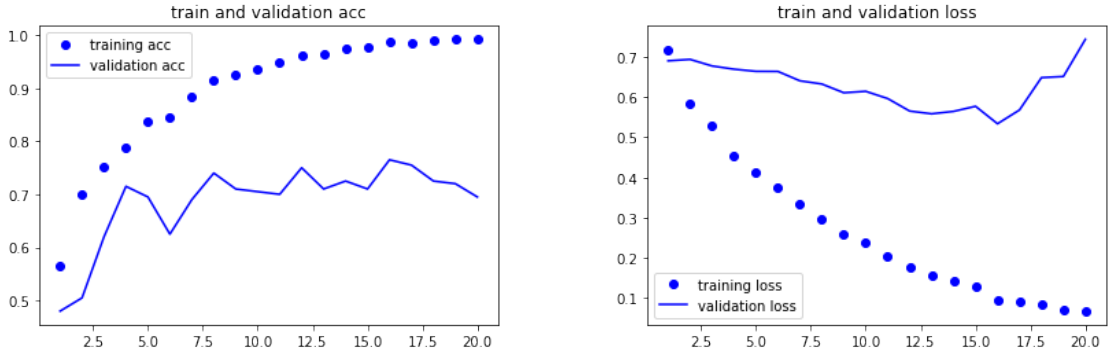


Figure 2: *second try*

Next we use VGG16 as freezed base. We get the following result.

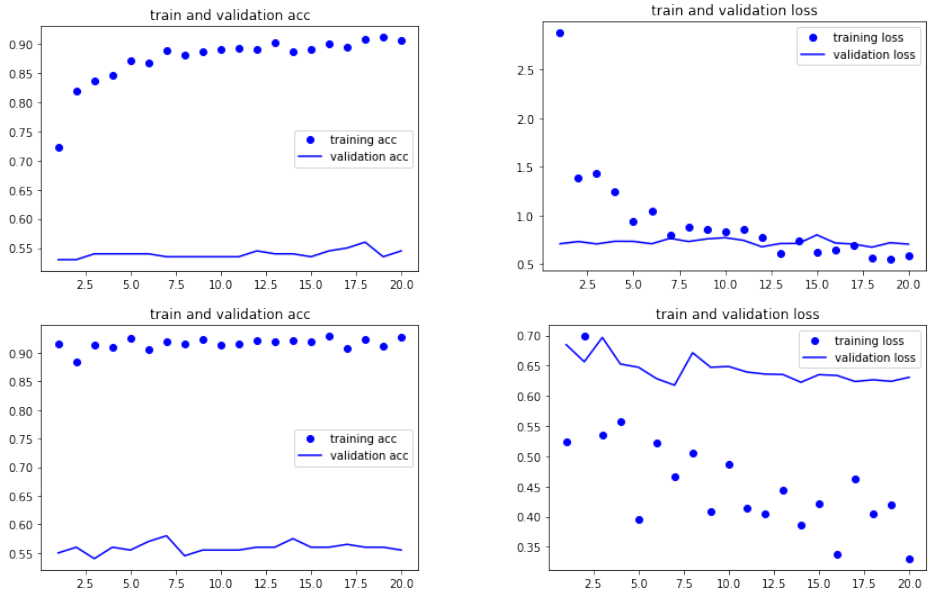


Figure 3: *First row are results from freezed base training, second row are results after I free the last layer of conv base*

Then we use some data augmentation and do above process again.

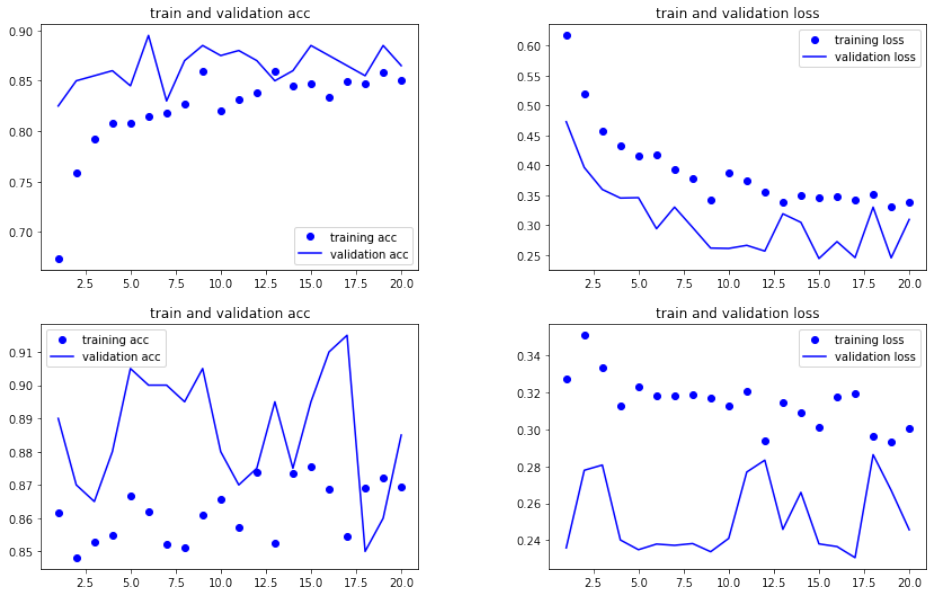
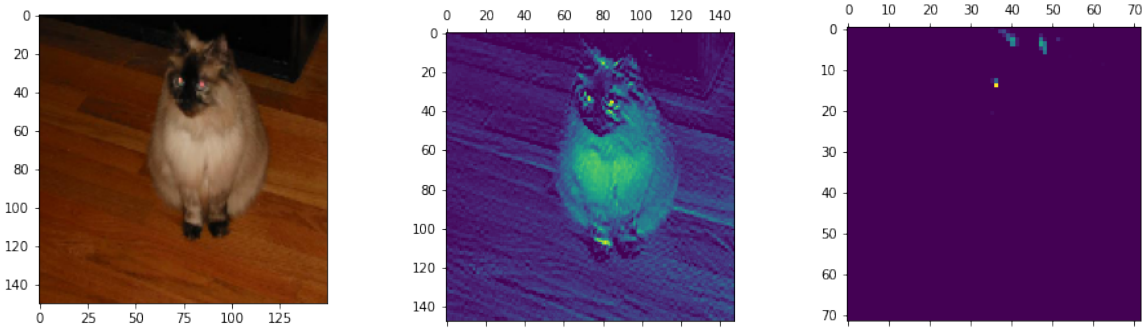


Figure 4: *First row are results from freezed base training, second row are results after I free the last layer of conv base*

Results and discussion

Now this custom CNN model has accuracy 0.70.

Below are some visualizing.



(a) *original cat image* ***(b)*** *activation at layer 0, channel 4* ***(c)*** *activation at layer 2, channel 2*

As for VGG based model accuracy. Without augmentation, we get accuracy 0.52 after training only the fc layers. And it is increased to 0.545 after freeing the last conv layer. With augmentation, we get accuracy 0.915 after training only the fc layers. And it is increased to 0.935 after freeing the last conv layer.

2 Multiclass classification

Problem Statement

A multiclass classification on CIFAR-10 dataset.

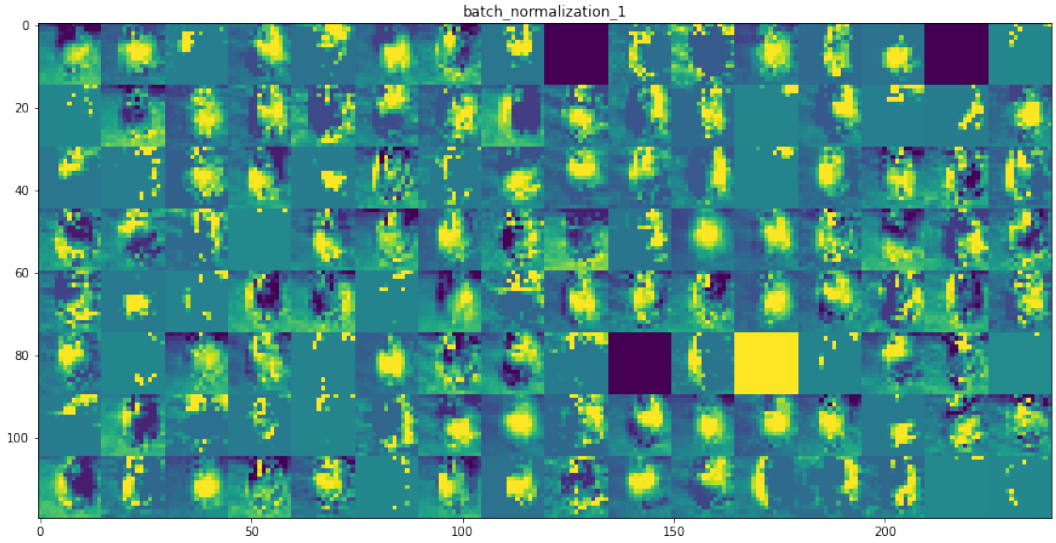


Figure 6: *all activations*

Proposed Solution

The dataset are load directly from keras package. The data splitting is also done there. Once done, we continue training a custom CNN. To improve, we adjust the epoch number and batch number as a simple hyper tuning.

Next I add one more inception blocks and do again.

Similarly I add instead a residual block and go over everything again.

Implementation details

Custome CNN with batch size 512 and 30 epoch. The optimizer is rmsprop with learning rate 1e-3.

Layer	output shape	params used
conv 2d	30,30,32	896
max pooling 2d	15,15,32	0
conv 2d	13,13,64	18496
max pooling	6,6,64	0
conv 2d	4,4,128	73856
batch normalizing	4,4,128	512
flatten	2048	0
dense	512	1049088
dense	10	5130

The neural network with inception block looks like below. Notice to shorten the training time, all the dataset are only kept as one tenth of the original number.

It's first trained with batch size 512 and 25 epochs. Later I tuned it to 64 batch size and 20 epochs.

Here's the training results.

As for the CNN with a residual block.

Here's the training result.

Results and discussion

The original CNN model only achieved 0.4843 accuracy. After using the inception block, it becomes 0.461. And is 0.47 if using residual block.

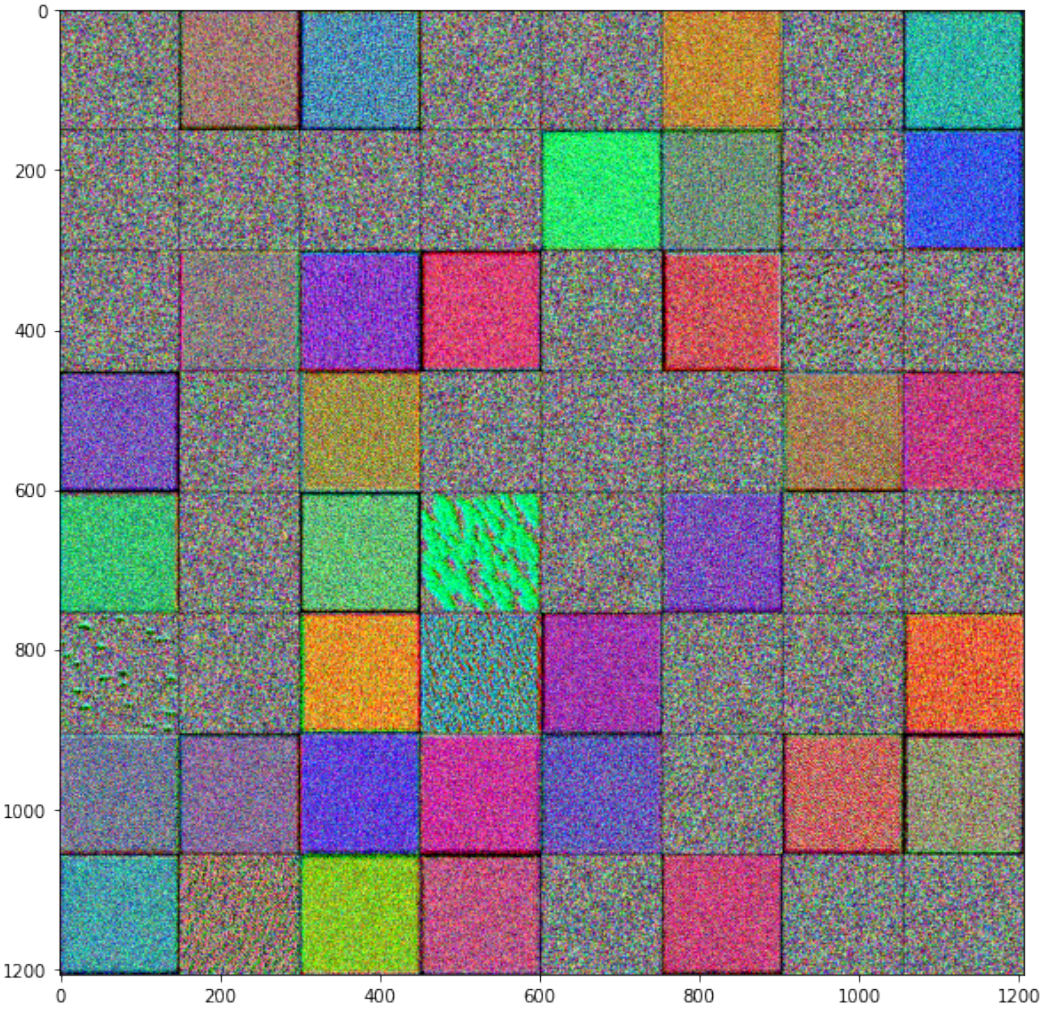


Figure 7: *all filters*

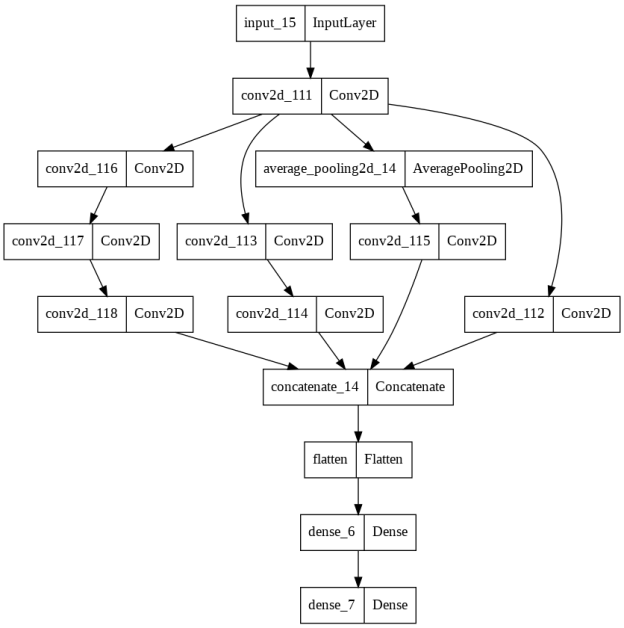


Figure 8: *CNN with inception block*

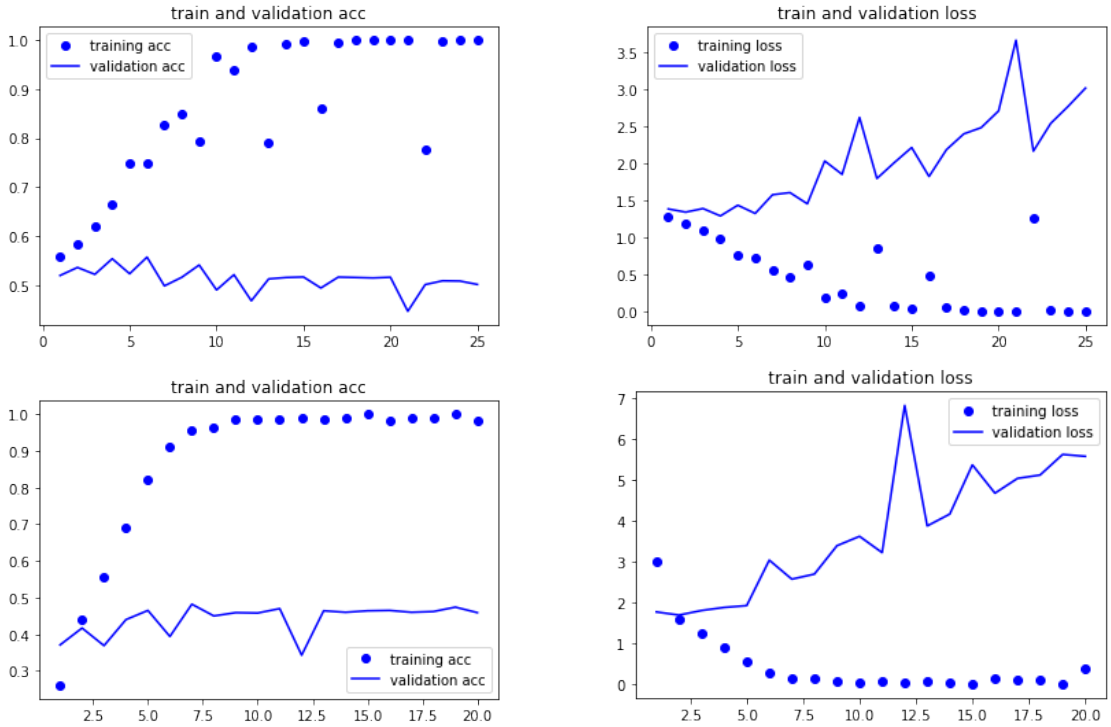


Figure 9: *training the CNN with inception block. First row is initial try and second row is done after hyper tuning.*

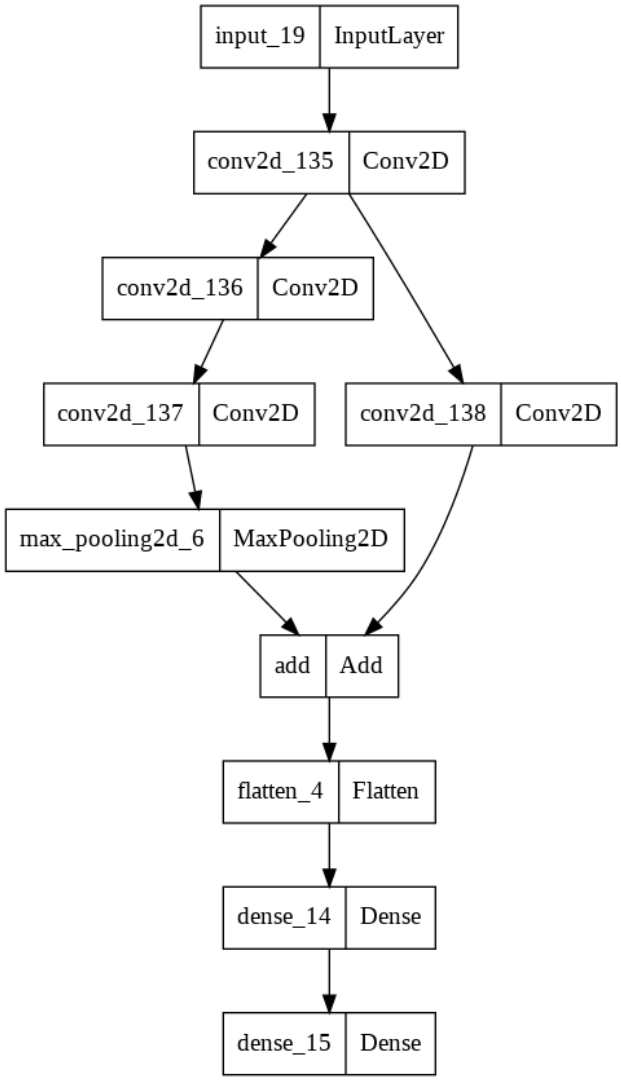


Figure 10: *CNN with residual block*

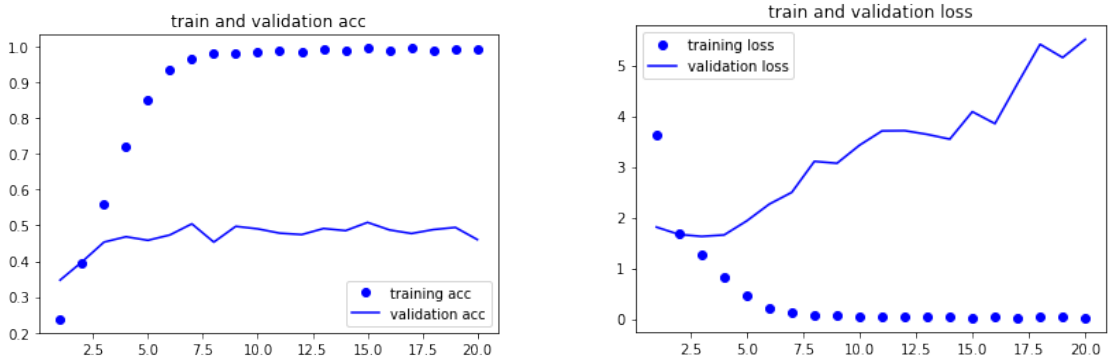


Figure 11: *training the CNN with residual block*