Name: Badr AlKhamissi

ID: 900141572

CSCE 4930

Practical Machine Deep Learning

Assignment #3

TinyImageNet Classification Challenge

Description

During the past three weeks of this assignment I used transfer learning with many different architectures with different hyper parameters, I won't be able to mention all of them in this report, so I will only mention the ones that are significantly different. In the following attempts I resized the input image to 256x256.

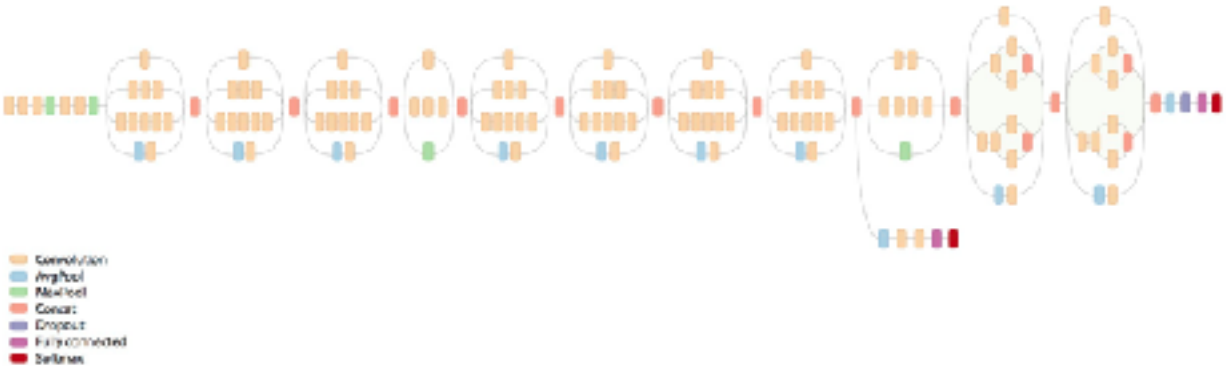
Friday, April 21, 2017

Attempt 1: GoogLeNet

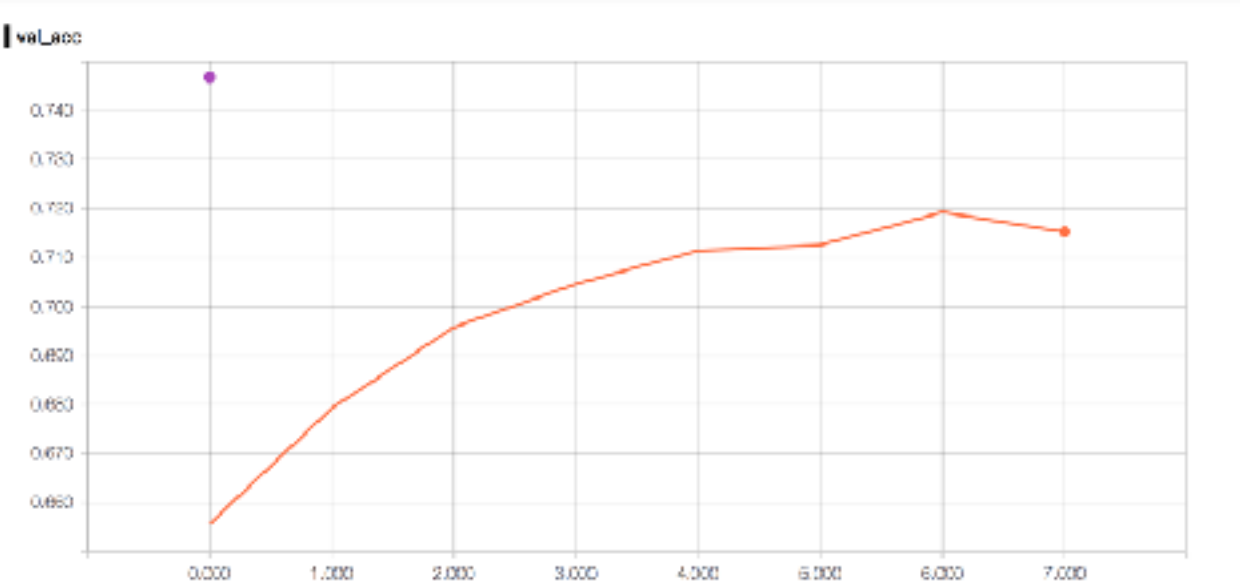
I first tried the GoogLeNet architecture using Nvidia Digits. At first I froze the convolution layers and trained only the fully connected one, but that didn't prove to be effective, so I then trained the first couple of convolution layers after the fully connected, namely the last inception module with different combinations of base learning rates, optimization methods, decay policies and data augmentations (rotation, translation, flipping, adding noise, zooming) However the maximum accuracy I was able to achieve was about 69%, using Adam with a low base learning rate of 1e-3, and a step decay function. I then was able to see that the best way to achieve accuracy was increasing the learning rate multiplier gradually starting from the first layers to the last. However, I then switched to to another models in my next attempt.

Attempt 2: InceptionV3

I then switched to Keras and used the InceptionV3 model, I tried first freezing the whole convolution layers except the last two, layers 172 and 173, then adding global average pooling and a fully connected layer of size 1024, followed by the soft-max linear classifier in the end. I used SGD with a learning rate of 1e-3 and momentum of 0.9. However I wasn't able to achieve good results that way, so I decided to not freeze any layers and train the whole network. I lowered the momentum to 0.7, removed the fully connected layer, and tried once using Adam and another time using SGD, and lowered the learning rate on plateau of validation loss. I was able to achieve about 73% with the stochastic gradient descent one.



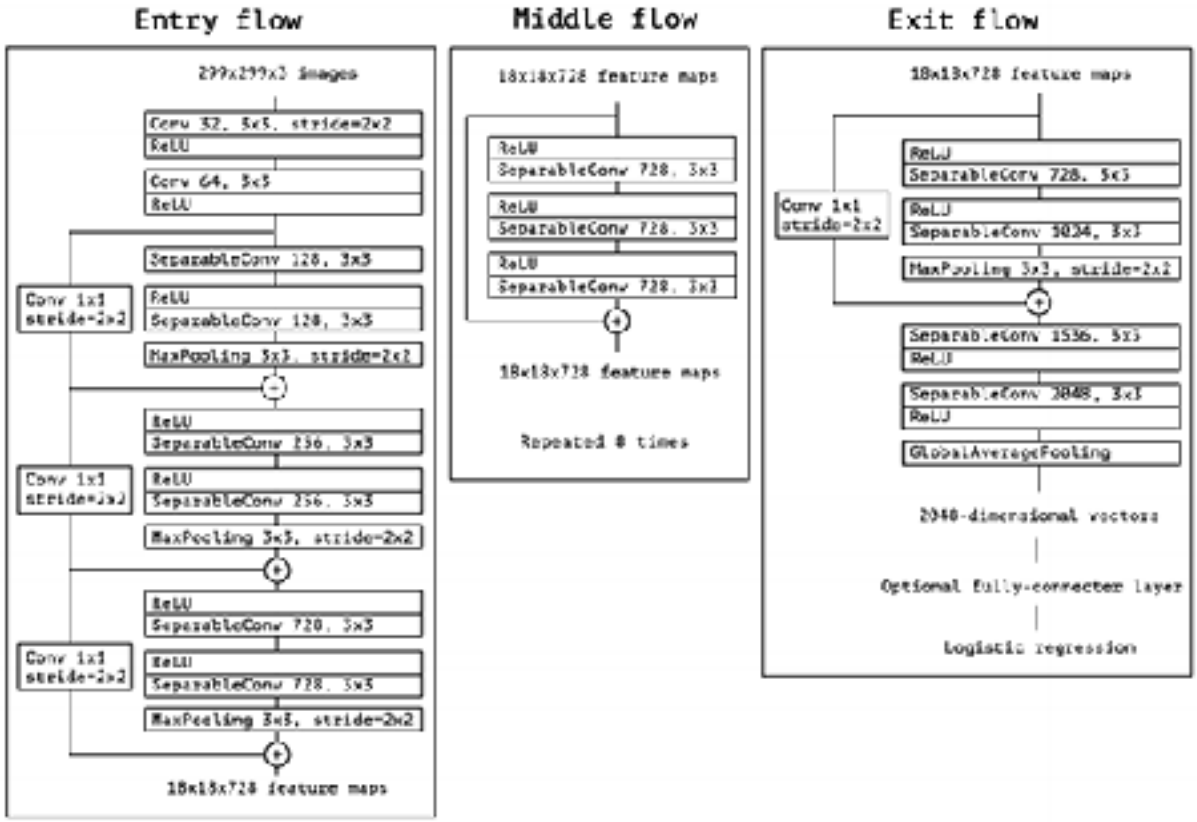
GRAPH OF HIGHEST ACCURACY USING INCEPTIONV3



Attempt 3: Xception

Finally, I used the Xception model, which is similar to InceptionV3 in its use of inception modules but smaller and with residual blocks similar to ResNet, diagram shown in the next page. I trained the whole network using the same parameters that got me the highest accuracy when using InceptionV3 and was able to achieve about 80% testing accuracy. I then tried the following:

- I attempted to use Adam optimization method, but it didn't get me good results.
- Then all the following attempts using SGD with nestrov momentum of 0.7 and decay of 1e-4, first I tried freezing the "entry flow" block shown below and training the other layers, but it only got me about 78%.
- I then tried to be creative and reduced the size of the network by repeating the middle flow 4 times instead of 8, and decreasing the size of image but it only achieved an accuracy about 68% after a few epochs and began overfitting.
- I added a fully connected dense layer of size 1024 followed by a dropout of 0.5, and got me 81.5% testing accuracy
- I then tried to lower the momentum and increase it gradually [0.5, 0.7, 0.9] but I wasn't able to get good results
- I then tried to train the fully connected layer (1024) with a dropout of 0.5 and the linear classifier only for 2 epochs, then training the whole network afterwards.
- I tried adding 3 fully connected layers following a bottleneck architecture [1024,512,1024] in the end with dropout of 0.5, following the same procedure as the last one, training them only for 2 epochs then training the whole network.
- I finally removed all the fully connected layers, trained the soft-max linear classifier in the end for a few epochs, then training the whole network again but wasn't able to leave it for enough epochs to achieve a considerably good accuracy



GRAPH OF HIGHEST ACCURACY USING XCEPTION