Deep Learning

Assignment 1

Report

Name: Milind Shaileshkumar Parvatia

SID: s3806853

Table of Contents

*) *Cover Page* 1

1) *Your final selected approach* 3

2) *Why you selected this approach* 3

3) *Parameter settings and other approaches you have tried* 3

4) *Limitation* 4

5) *Improvements that are required for real-world implantation* 4

1) Your final selected approach

I have selected ResNet-34 Architecture for both target predictions. I have combined both targets (tilt and pan) as target feature and use ResNet34 to predict angles from Images.

2) Why you selected this approach

For this project I have decide to use ResNet model because it is extremely efficient architecture, ResNet are used as basic starter point for computer visualization problems. with ResNet we can construct very deep Neuron network with 1000 layers without breaking network, it uses residual networks which skips layer, so model's performance won't decrease and provide efficient output.

I used ResNet-34 specifically because It is faster than other ResNet (50, 102, ETC) for testing purpose since it has smaller architecture it is very easy to run and compile on colab and kaggle notebooks. Other than that, my Image size is already smaller than original size required for original ResNet, so using smallest ResNet was best option.

I have Combined two targets to make modelling easier, this way I can manage both targets from single Tensor and I don't have to run them separately. It makes [input=x, output=y1, y2].

I have Created tensors with Data Pipeline (custom functions) to get more control over function, I have also applied image augmentation in this function.

3) Parameter settings and other approaches you have tried

My final parameter is kernel's = [64,128,256,512], number of layers = [3,4,6,3], lambda values for regulation = 2e-07. Getting into deeper models were stated to overfit so I used less complex model.

- To reach here, my main methodology is to use base model of kernel's = [64,128,256,512], number of layers = [3,4,6,3] as starting point.
- Since number of images were extremely low, I try to apply augmentation on images, by changing saturation, hue and brightness.
- I saw huge difference in validation curve so I applied momentum = 0.7 to Batch normalization, I learned that for smaller set of batches like in my case momentum should be between 0.7 to 0.9 and larger batch 0.5 to 0.7.
- I have applied Regularization after that because of high variance in model curve, I have tried different parameter's and 2e-07 came out as most efficient on graph. So that's why I have used it.

- I also tried different iterations of models layer, from kernel's = [64,128,256,512], number of layers = [3,4,6,3], lambda values for regulation = 2e-07, If I decrease model's layer, model started underfitting and if I increase model's layers model become highly overfit. To make it as much as generalized I ends up with my base model's parameter which are ResNet34's architecture.
- Other than that, I have used MirrorStrategy which utilise gpu performance and make cache a lot faster.

4) Limitation

Main limitation was of very small dataset (~3k Images). Due to that model prediction is not perfect.

5) Improvements that are required for real-world implantation

Model has very hi accuracy but still predicts a lot of images wrong. I think getting more sample can easily improve this.

Difference in label frequency which generates class imbalance which makes harder to epidictic less frequency samples. In this project I have Ignored this problem since ADAM optimizer has very small effects on class imbalanced.

Handling InverseTimeDecay in optimiser was extremely hard, I have tried following [5,10,20,50,100,500] but ends up with 1000 into STEPS_PER_EPOCH.