INF265 Project 2

1.

Object Localization:

We start of by loading the data and normalizing it. Then we explore the data by printing a random image and its shape. Then we define a three variants of models of LeNet5. To train the models correctly we have to write our own loss function where we use BinaryCrossEntropy as a loss function for the prediction of there being a number in the imagine or not. Then we use MeanSquareError for the bounding box. And CrossEntropy for the classifier. The loss function is vectorized and we use a boolean mask to filter out in training the samples where there is no number in the picture for the training of the boundary box and classifier. We have written our own functions to calculate the IoU and accuracy of the models, of which we chose the best one. For the best model we plot some predictions and find a test data accuracy.

Object Detection:

All steps are the same other than the loss function being vectorized. It rather takes in each cell of the 2x3 grid for training with its corresponding label.

2.

Object Localization:

We have chosen to examine three models. The first is a standard LeNet5 architecture. We then chose to examine how a shallower and deeper LetNet5 architecture would compare to the original. To achieve this comparison we established a set of common hyperparameters. The drawback here is obviously that the deep network should have a lower learning rate and train for longer than the standard LeNet5. And vice versa for the shallower model. We therefore took this into account when choosing the hyperparameters. Using ADAM optimizer with betas (0.9, 0.999) and epsilon (1e-8) are standard practice according to Andrew. We also added L2 regularization of a modest degree to counteract overfitting for the shallower model and extend epochs, lambda (weight decay) of (1e-4). For batch size and learning rate we have to be careful since if we chose either to large or low the model won’t learning. In our case we chose Learning Rate (1e-3) and Batch size (64).

Object Detection:

We did the exact same for object detection with the same reasoning.

3.

Obs: We measure accuracy as (IoU + Accuracy) / 2

Object Localization:

10 epochs:

The standard got an accuracy of 22%, shallow 66% and deep 23%. So it's clear that the shallow LeNet5 is superior for a lower epoch. And that the other models would have needed more time training to preform better.

Test data accuracy: 72% which is very good for 10 epochs. No overfitting as well

Object Localization:

10 epochs:

The standard got an accuracy of 29%, shallow 29% and deep 30%. So it's clear that the shallow LeNet5 is superior for a lower epoch. And that the other models would have needed more time training to preform better.

Test data accuracy: 31% which is very ok for 7 epochs. Here also no overfitting as well

Conclusion:

From the results we see that we should have ran it for more epochs with out chosen hyper paramters.

4.

Look at the jupyter notebook provided.

5.

Had problems with just printing the bounding box for object detection, though the IoU is correctly calculated.

In object detection you would normally have to use non-max suppression to filter out the most probable boundary box for each object. In our case each cell can only detect one object, since it builds on object localization just for a 2x3 grid. And since an object's center can only lie in one grid cell we get that if there are two objects in one grid one of the objects will only be predicted. And if an object crosses between grids, only one of the grids should able to detect it with a high enough of a confidence.