HW6

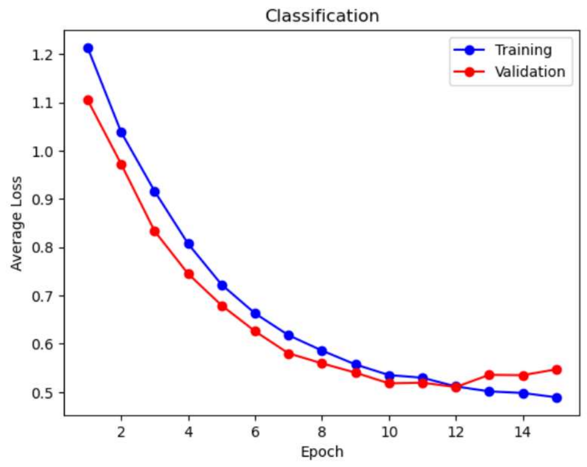
Testing and Validation Phases:

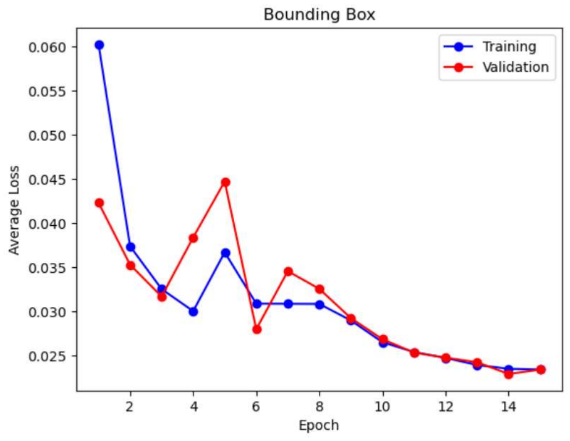
The structure of my RCNN network used the Resnet18 convolution layers and global average pooling layers as the backbone, with a 512 input to 128 output Linear layer, ReLU layer, 128 input to 64 output Linear Layer, ReLu layer, and a final 64 input to 5 output Linear layer to represent the 4 detection classes and "nothing" class as part of the Classification block or tail.

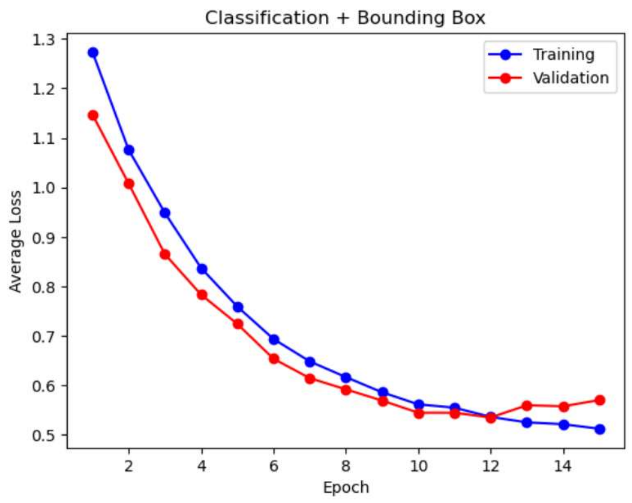
For the Bounding box determination, I used a 512 input to 64 output Linear layer, TanH activation layer, and then a 64 input to 16 output Linear layer to estimate the 16 bounding box coordinates for the 4 detection classes. The reason for using TanH was because I found that I was ending up with a lot of NaN values on the output which would lead to NaN batch averages. My suspicion was that there was numerical instability being produced from ReLU. Since the bounding box coordinates are fractional, I decided to use TanH to keep the output in the range of -1 to 1.

For data loading and batching, I used batches of 32 and 6 workers for loading which leads to 700 batches. I used this batch size since it worked well for me in the previous homework and 64 data points per batch might smoothen the graphs to much and give misleading results. Throughout the training process, I stuck with lambda = 1 since the result was pretty good for it. Lambda > 1 seemed to have increased the starting loss for the classification and bounding box.

Model 1:



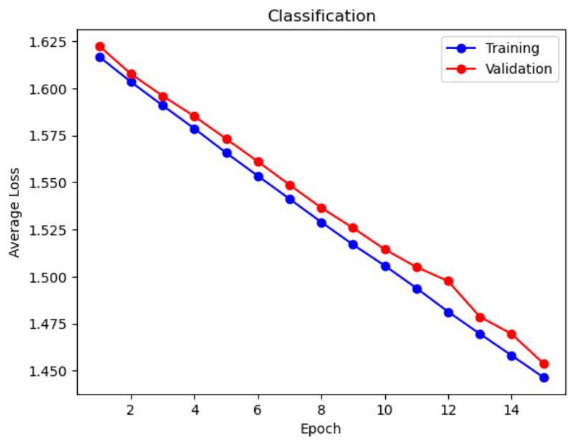


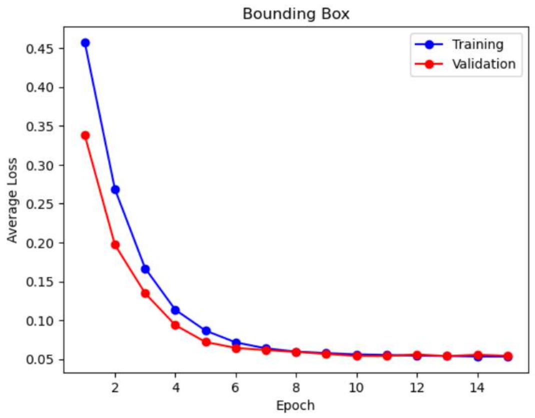


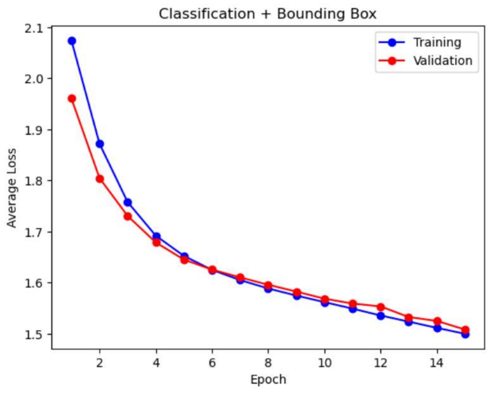
```
Avg Training Error:
        Classification: 0.48895182007470644, Bounding Box: 0.023444186188784, Combined: 0.5123960056619065
Avg Validation Error:
        Classification: 0.5469511624987565, Bounding Box: 0.023431750823478294, Combined: 0.5703829121489764
Avg Positive Detection Accuracy: 0.9100397614314115
Avg IOU: {:}
 0.6785301467254744
Training Confusion Matrix
Validation Confusion Matrix
[[8631  100  208   87 1318]
 [ 212  358   16   16  151]
 [ 357    0 1451   10  246]
 [ 229   38   31  302  124]
 [1166   39   91   30 8438]]
[[2483   20   74   21   90]
 [  84   95    0    1    5]
 [ 117    0  403    1   16]
 [  71    1    7   86    8]
 [ 641   11   47    4 1428]]
```

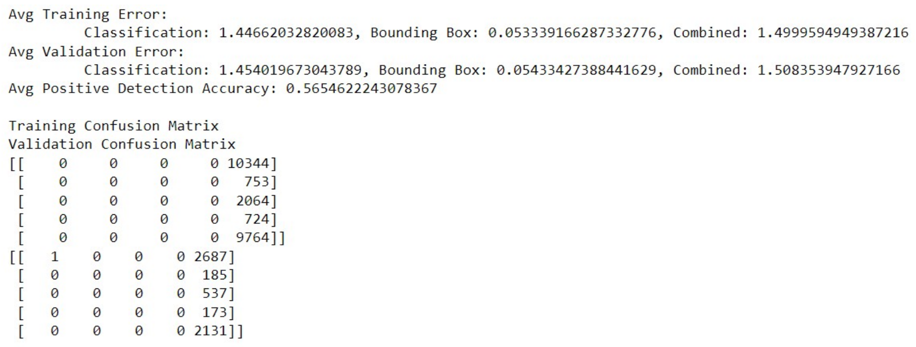
The first RCNN model I trained used SGD, was trained for 15 epochs, and used a learning rate of 1e-3. As we can see from the graphs, the network is doing quite a decent job of bringing the validation error down and following the training loss closely. The bounding box regression seems to have the biggest reduction in error over time, but I personally think this could be because deviation of bounding box coordinate values is more meaningful in error reduction rather than passing back the error of classification. The classification for each of the models I trained seemed to take longer to train, which is why decided to use more epochs in the third model to see if the tail of the classification error could be reduced. From the summary, the combined error was brought down to 0.57, the positive detection accuracy was 91 percent and the average IOU was 0.67 on the last epoch for validation. The confusion matrix also shows that the model did a decent job of separating class 1 and class 4, but the other classes didn't do nearly as well. In validation, the model is predicting class 0 instead of class 1 84 times as compared to 95 times for the ground truth.

Model 2:





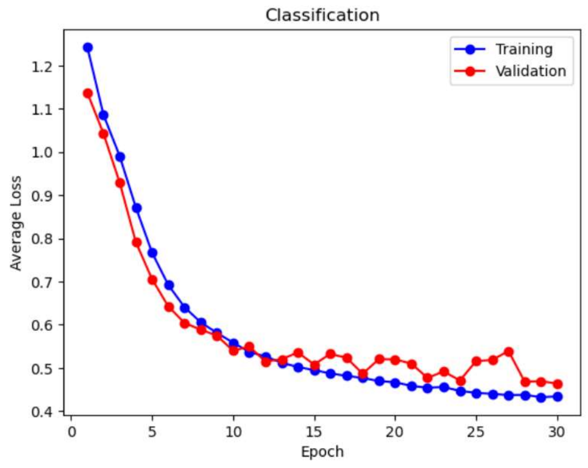


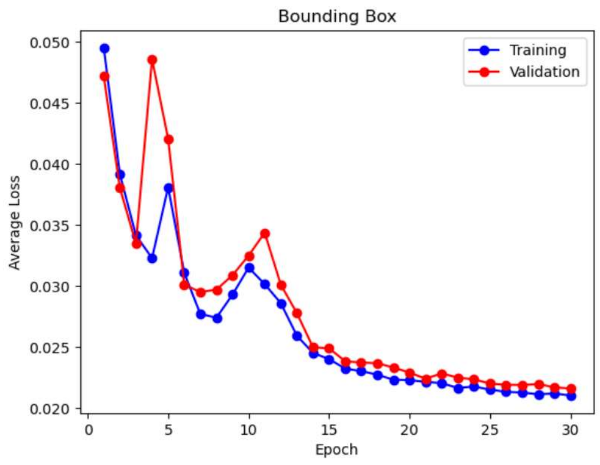


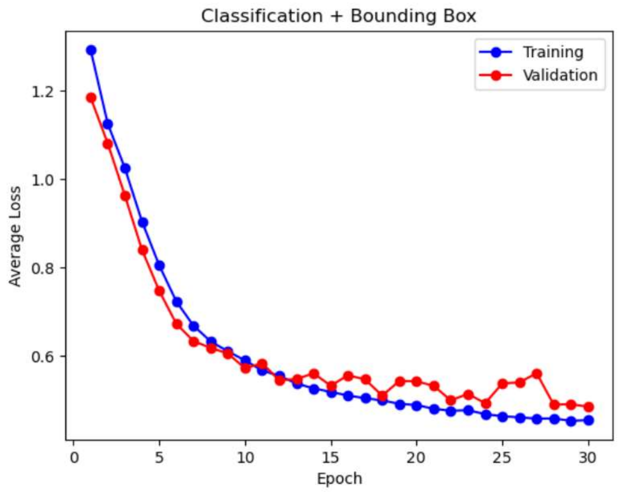
```
Avg Training Error:
         Classification: 1.44662032820083, Bounding Box: 0.053339166287332776, Combined: 1.4999594949387216
Avg Validation Error:
         Classification: 1.454019673043789, Bounding Box: 0.05433427388441629, Combined: 1.508353947927166
Avg Positive Detection Accuracy: 0.5654622243078367

Training Confusion Matrix
Validation Confusion Matrix
[[    0     0     0     0 10344]
 [    0     0     0     0   753]
 [    0     0     0     0  2064]
 [    0     0     0     0   724]
 [    0     0     0     0  9764]]
[[   1    0    0    0 2687]
 [   0    0    0    0  185]
 [   0    0    0    0  537]
 [   0    0    0    0  173]
 [   0    0    0    0 2131]]
```

The second RCNN model I trained used SGD, was trained for 15 epochs, and used a learning rate of 1e-5. Although this model had the worst result wise due to lack of epochs, I trained to see if the reduction in classification and bounding box errors was purely due to random chance, or if the model is succeeding. Based off the results, clearly more training is required to attain results even close to model 1.

Model 3:



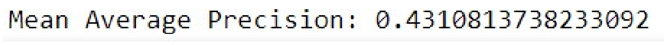




```
Avg Training Error:
         Classification: 0.4229633129730418, Bounding Box: 0.021013172288626633, Combined: 0.4439764855479872
Avg Validation Error:
         Classification: 0.5278251882038969, Bounding Box: 0.020507062577210657, Combined: 0.548332249652074
Avg Positive Detection Accuracy: 0.9381800197823936
Avg IOU: {:}
 0.6992714232194799
Training Confusion Matrix
Validation Confusion Matrix
[[8813   95  179  107 1150]
 [ 163  446   20   15  109]
 [ 290    1 1568    9  196]
 [ 145   27   24  407  121]
 [1049   31   94   46 8544]]
[[2509   22   74   18   65]
 [  77  106    0    0    2]
 [ 115    0  395    4   23]
 [  62    4    4   90   13]
 [ 654   15   27    4 1431]]
```

The third model I trained used SGD, was trained for 30 epochs, and used a learning rate of 1e-3 since the results from the 1e-5 seemed to require many more epochs of training to get to the same loss levels that the first model achieved and 1e-3 had the best losses for classification and bounding box. As a result, I decided to train for 30 epochs to see if the accuracy might increase and losses might go down. The model's validation error follows the reduction in training error extremely closely, but that could also mean that potentially the data from training is very similar to the validation set's data (just a theory) since the randomization of the dataloader still produced very similar results to model 1. But the important thing to note is that the validation error didn't start to increase continually and deviate from the training error at high epochs meaning that 30 epochs is a positive increase from 15. From the summary, we can see the detection accuracy has gone up to 93 percent, the final epoch's average validation combined loss went down to 0.54 from 0.57 from model 1, and average IOU has increased to 69 percent. In addition, Class 1's prediction is now slightly further away from misclassifying Class 0 from model 1 (almost by 30 predictions).

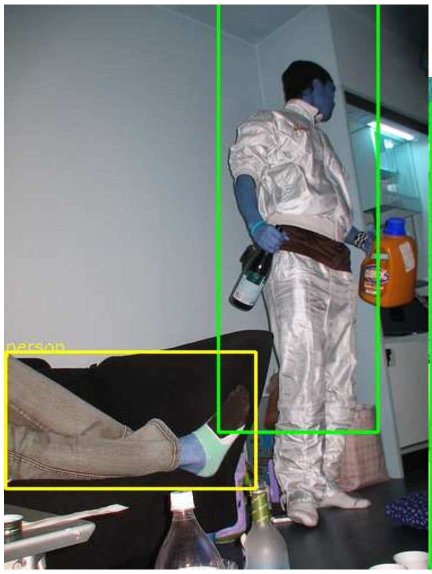
Testing:

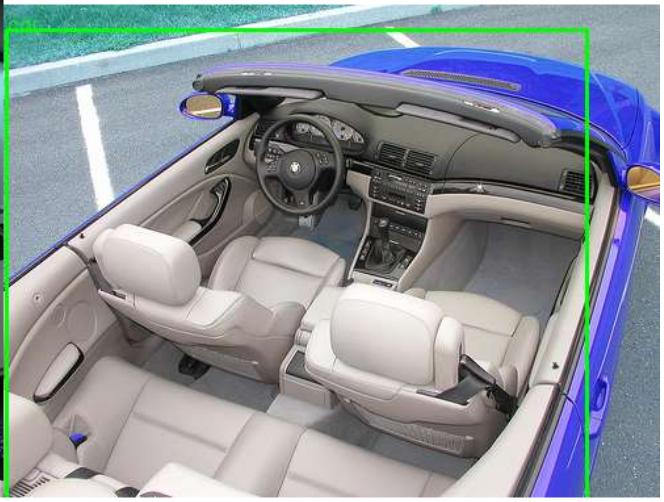


```
Mean Average Precision: 0.4310813738233092
```

The final Mean Average Precision my model retains on the testing data is 0.43. The reason for the lack of precision might be due to not enough training or a need for refactoring the RCNN classification design. I say this because the precision measurement is dependent on how well the classification problem can be solved since it deals with retrieval. From the training I saw that I couldn't bring the classification error down quickly even though I used 3 layers. Maybe because this is a classification problem, more features need to extracted and examined to tell with more confidence whether a candidate region contains a specific object. Maybe with more hidden layers or experimenting with the activation functions might retrieve a better result. One thing I noticed it that the model was extremely good at predicting Class 0 and Class 4, so maybe there's a possibility that the model was trained heavily on those 2 classes/image regions.

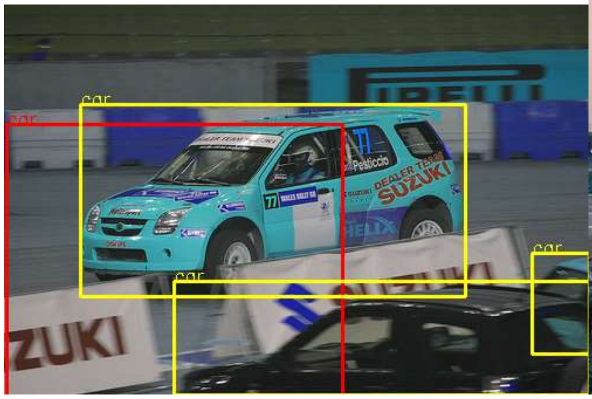
Good result:

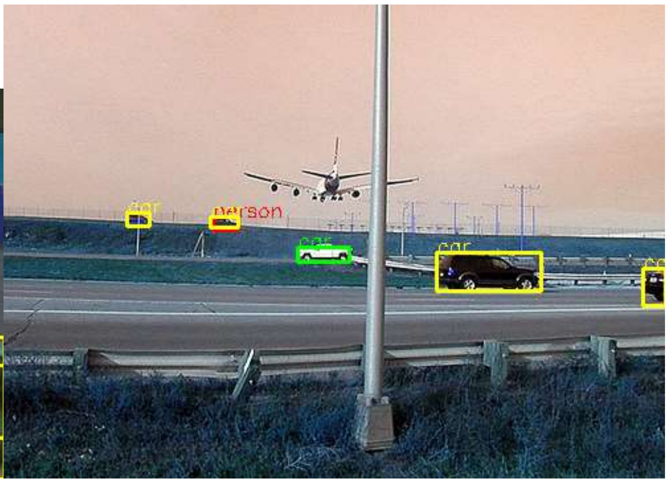




Model was able to classify person even though their face wasn't showing. But the other person's leg was not identified probably due to lack of features captured (just the leg and shoe). The car on the other hand was surprising since the model was able to greater take into account the interior of the car for determination of class since the car's shape and exterior are hidden.

Bad Result:





Model predicted too large of a bounding box which then didn't overlap nearly enough with the other bounding boxes to pick out the cars, since it was either too far way or the Union of the bounding boxes with the prediction was too high. Second one couldn't classify the obvious black car and misclassified one of the cars as a person probably due to lack of features from distance.