**Additional data**

We want to introduce as much data as we possibly can into the machine learning model because the more data a model has directly influences how it handles varying images.

So for example as we are creating a model which detects cars, exceptional variation may include: Different lighting, different angles, different models etc. Thus having more images of different vehicles allows the program to develop a more rounded view of exactly what a car is so that it may be identified.

**Justification**

In the following experiments, we will be using and iou metric for testing accuracy. The iou metric involves calculating how much 2 bounding boxes overlap one another.

Our model struggles significantly with detecting cars, As of the 8th of may our model has a detection rate of around ~72% on the testing data. and 74% on the training data. using 500 images with an 80:20 training:test ratio with a 40 epoch cycle. The model can correctly bound itself around a variety of cars however a constant failure state seems to appear. Cars that are on their side.

To attempt to improve this accuracy. I will be adding more images, due to memory complications I cannot add too many. The number of images I have found to be appropriate is around 1420.

This test with another 1000 images will be using the same method as the previous 2 experiments.

However another reason why it is important to have good accuracy is so that we can spot unusual fail cases. Such as a car facing sideways not being detected constantly. This therefore allows us to look for more data which would allow the model to detect those cases more efficiently.

**Data preparation**

No real additional preparation is required for these images. I will be using the same software that I previously used to create the bounding boxes. The script handles most of the data preparation for me. Such as image manipulation and assigning bounding boxes with images.

For the data transformations however I will be mirroring the image using ImageOps from pillow. After this I will have to do some maths with the bounding boxes to flip them. The maths for this is. Where X is the bounding boxes and where I is the input dimensions. (X – I) – X\*2. However this will quite literally flip the bounding box. Therefore the first 2 values in X will need to be swapped with the last 2 values in X.

**Impact report**

Using the same model and settings as used to test 500 images. We now use it with 1420 images. The result of the final epoch was ~77% on the training data and ~75% on the testing data. This is an increase but only a small increase. To test if this is due to the model restricting accuracy, I will be adding data transformations to artificially increase the size of my dataset.

The result of this after using the same method was a result of ~75% This was not at all what I was expecting. As it doubles the images available to be tested to 2840. Where there are 2272 sample images for training.

Due to the increase in images I did look over the pre-processing assuming that something was wrong there. However whenever I changed something model performance would go significantly down (>0 at a loss rate of over 5000). Therefore I can only assume that this is correct. Due to this, this must be a model issue. As performance rates still vary in the model, as I am looking for that sweet spot.

It may also simply be that there is not enough variation in the images Therefore. Now that I have somewhat fixed the issue regarding memory I will add even more data in order to attempt to improve accuracy. However I will therefore have to remove data augmentation. Otherwise I will not have enough memory to be able to load the images.

**Discussion**

So after adding another ~500 images and removing data augmentation I was able to achieve an accuracy of ~80% Seeing this I can now identify that the problem is my data. But after adding another ~600 images, I ran it again and only got slightly above 80% This was somewhat confusing to me. However I assume that this is because the data in this cluster of images was either not good quality. Or was not able to display the features that was being tracked. In other words again I am still missing data which would be useful in detecting other important features.

Throughout this we have seen a massive improvement in the program when more data is added. While it may have stopped at 80%. I believe that it may increase with better selected images. For example some of my images used were adverts, so there was a lot of graphics around the image which may have interfered with the accuracy of my model. This is important because our model wouldn’t realistically be looking at adverts of cars. But would be looking for cars on the road. Which means I should be looking for pictures where there are more cars at 90 degree angles. So that they can be detected more reliably as they are the most important ones to detect.

During the data processing where I establish bounding boxes around the cars I looked deeply at many of the images and determined their appropriateness. As I said just above many of the images were adverts, however they contained the whole car so I deemed them appropriate for use. However many images that didn’t make It to the final cut were removed from the program for either being too low quality. (One of them had a colour range of about 5) or only partially showing the car. These images I assumed would make the detection process harder. However I realise now that I could have used them as testing images to test how robust the model is. Going back to reality and how we planned for our model to be used in automated cars, the camera might not always have such clear vision. There may be heavy for or it may be extremely dark. And knowing how colours change in the darkness for cameras they have less colour options to work with, making it more pixelated, having that low colour range image may have been all the AI needed to increase model performance.

However I noticed after testing this new model on a test video that is severely struggles to see the car. It can get the shape correct where the top and bottom of the box are good at seeing the car. In comparison to giving it a list of images, where it’s performance is significantly better. However after looking at the results of those images I noticed that images that were significantly distorted in the pre-processing had extremely bad accuracy. This therefore makes me believe that in the case of the video this may be due to the pre-processing of the image. Perhaps by changing the input dimensions to be wider it may improve accuracy as many of my images are bigger horizontally.