WEEK 5 Progress

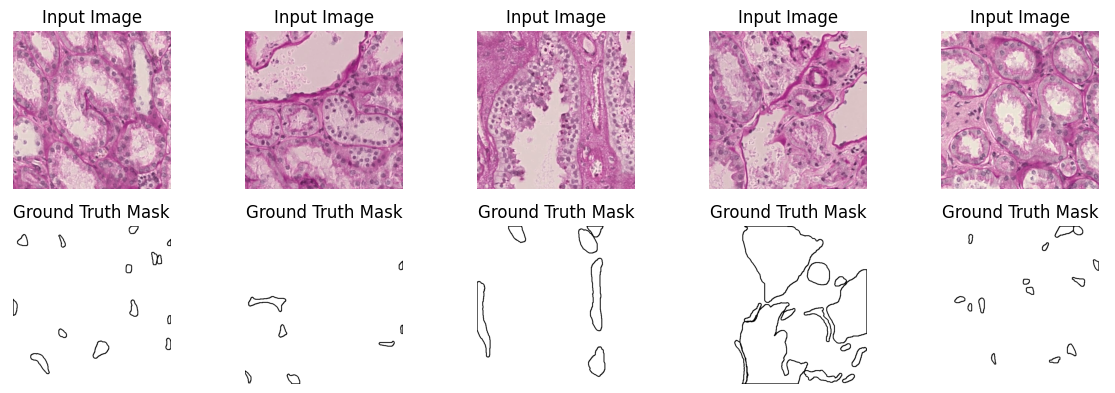
Jaison Lewis

------------------------------------------------------------------------------------------------------------------------

Programs made this week:

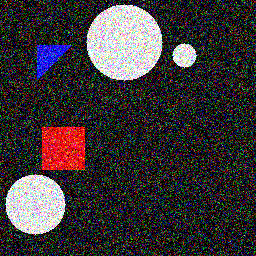
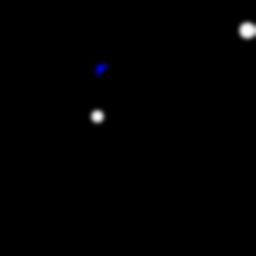
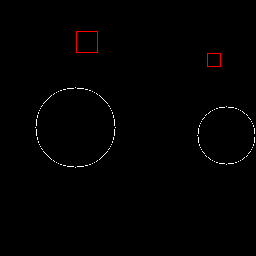
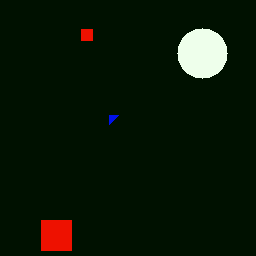
1: Unet program fixed to use the correct mask with the correct image while training.   
Earlier, the program was taking the wrong mask, which I determined could be causing the problem with building our kidney model. Added a few lines of code that check the correct image and mask are being picked up.

Github link: <https://github.com/jaisonlewis/unet-adpated-kidney>



2: Reworked program to create synthetic data to create data with blemishes.

Added code to the original synthetic data maker to add blemishes to the data. The adjusted images are 80% of the total images available for training. This is adjustable.  
Github link: <https://github.com/jaisonlewis/Syn-data-blemished>



3: Program converts CSV to the COCO JSON format.

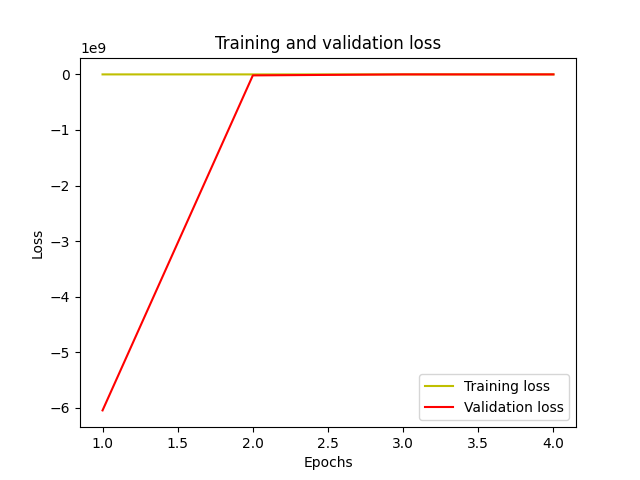
The VGG Image Annotator COCO JSON was not compatible with Detectron2 as it was missing some key data points. Made a program that enters missing data by generating it and getting it from the images in the directory.

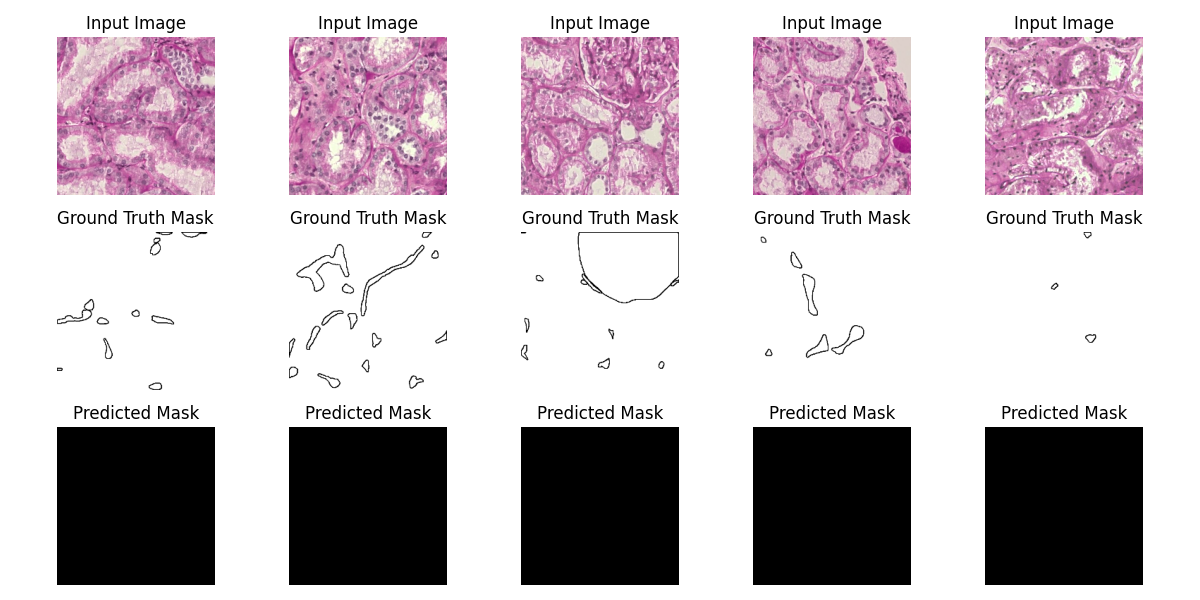
Github link: <https://github.com/jaisonlewis/csv-coco/>

4: Detectron 2  
Adapted Detectron 2 code to work with kidney files and annotations. Found making the conda environment a bit challenging. Will have to rework the conda to allow this file to run.

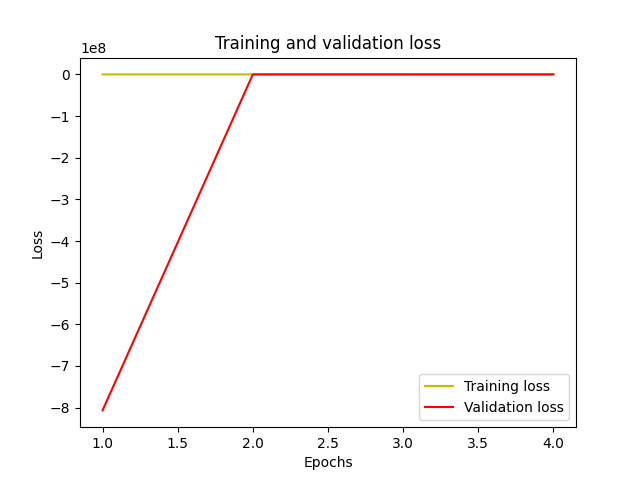
Github link: <https://github.com/jaisonlewis/kidney-detectron2/>

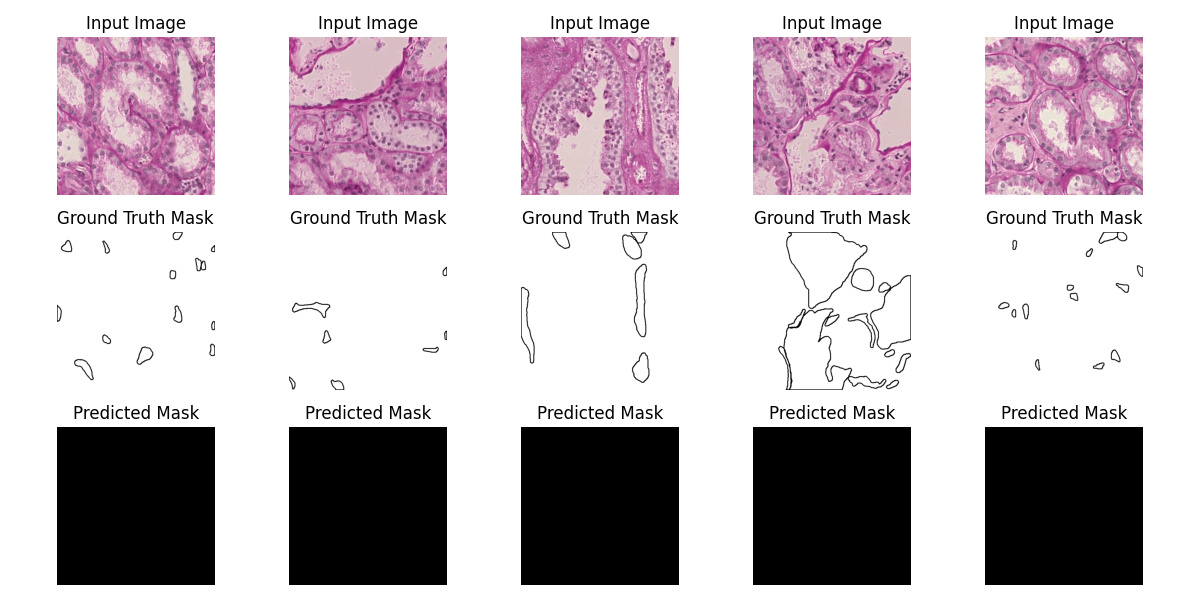
Because last week's UNet trainings were a bust. I ran some more tests this week with promising results. It was still stopping early, sometimes as early as two epochs, depending on the backbone used, and I was getting better results with normalising, so I did that with later training exercises. For the kidney data 10 epochs was not nearly enough so I ran more epochs for some tests. Here are the results of almost all the tests. Will add Seresnext50 results after the 100 epoch test is done later today. Will also add results of the code on the new set of blemished images.

Resnet50 Colour 2Epochs 256X256 pixels  


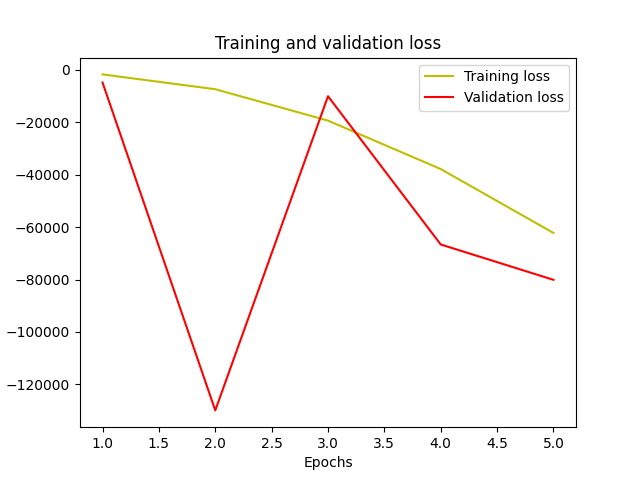


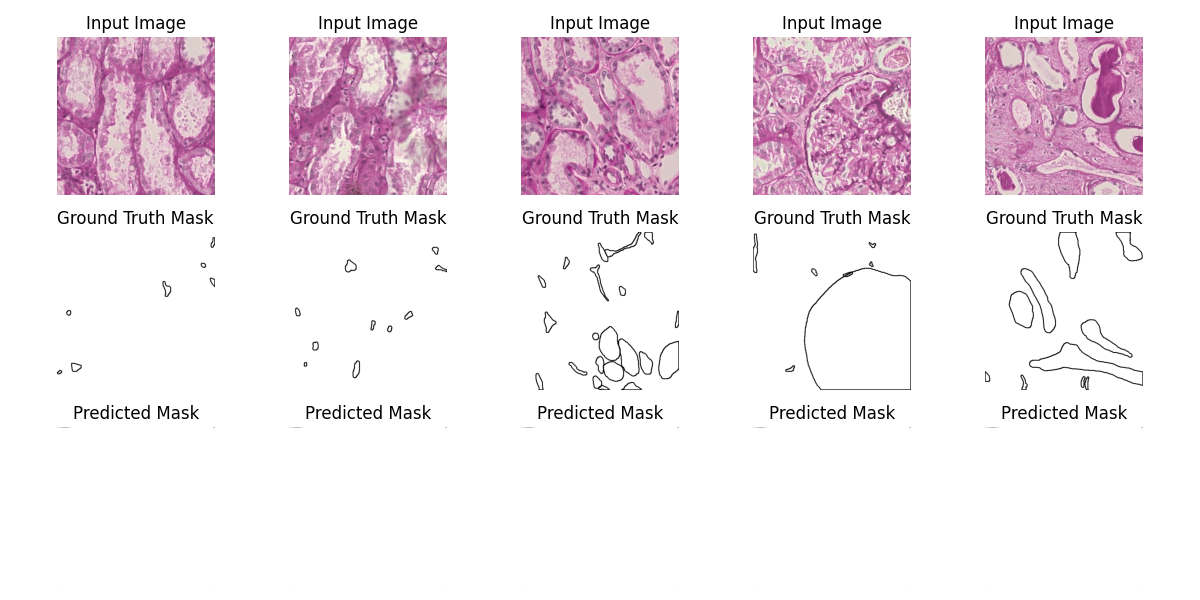
Resnet50 512x512 colour, 4 Epochs



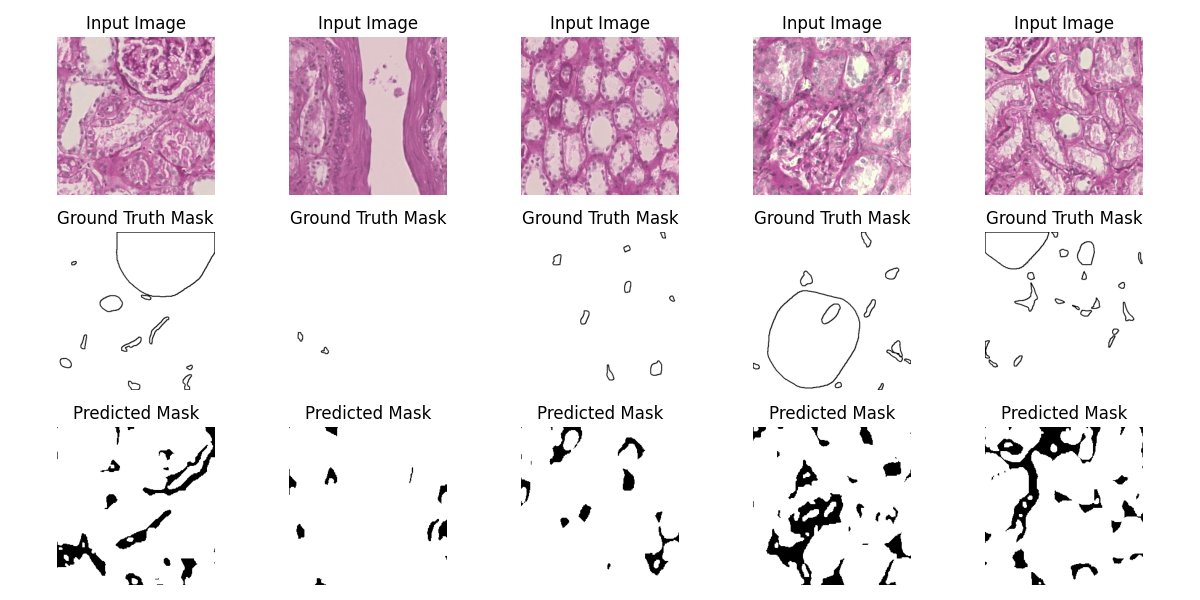


Resnet50 512x512 normalised, 5 Epochs

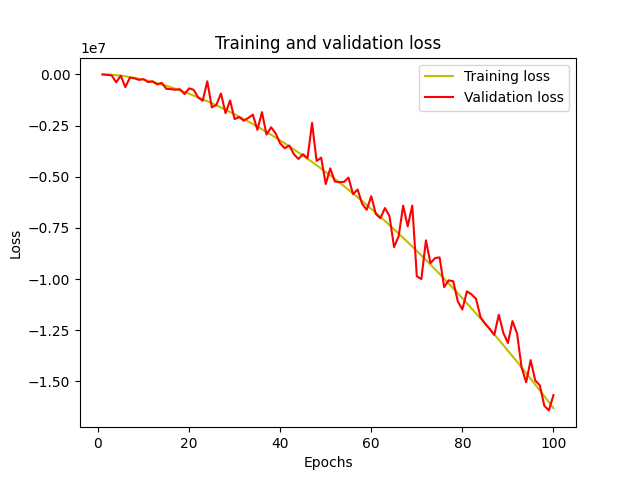


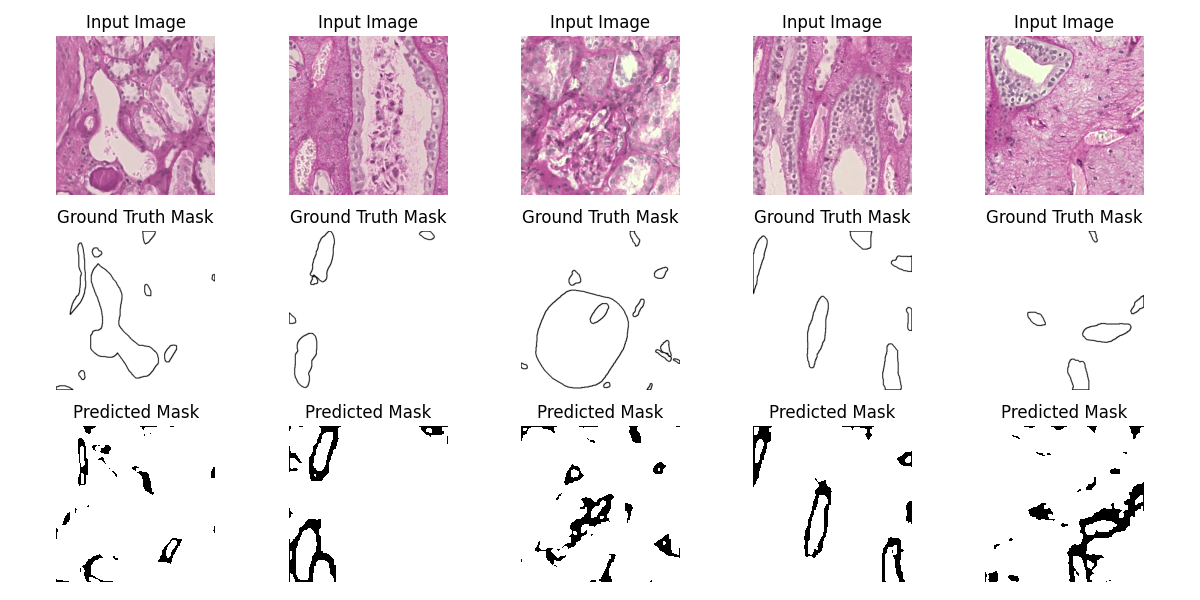


Resnext50 normalised 20 Epochs no earlystopping  

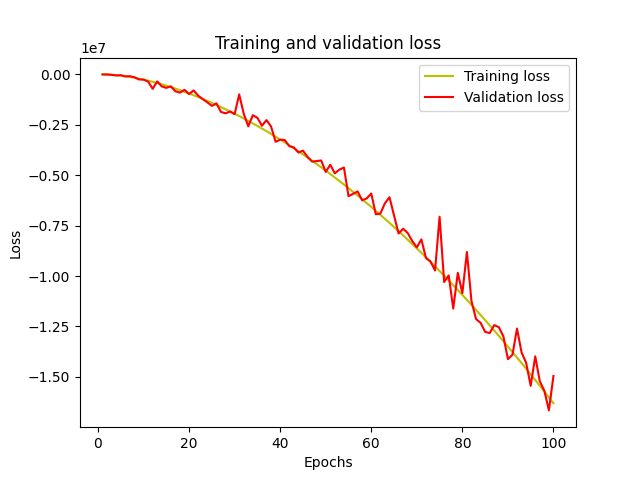



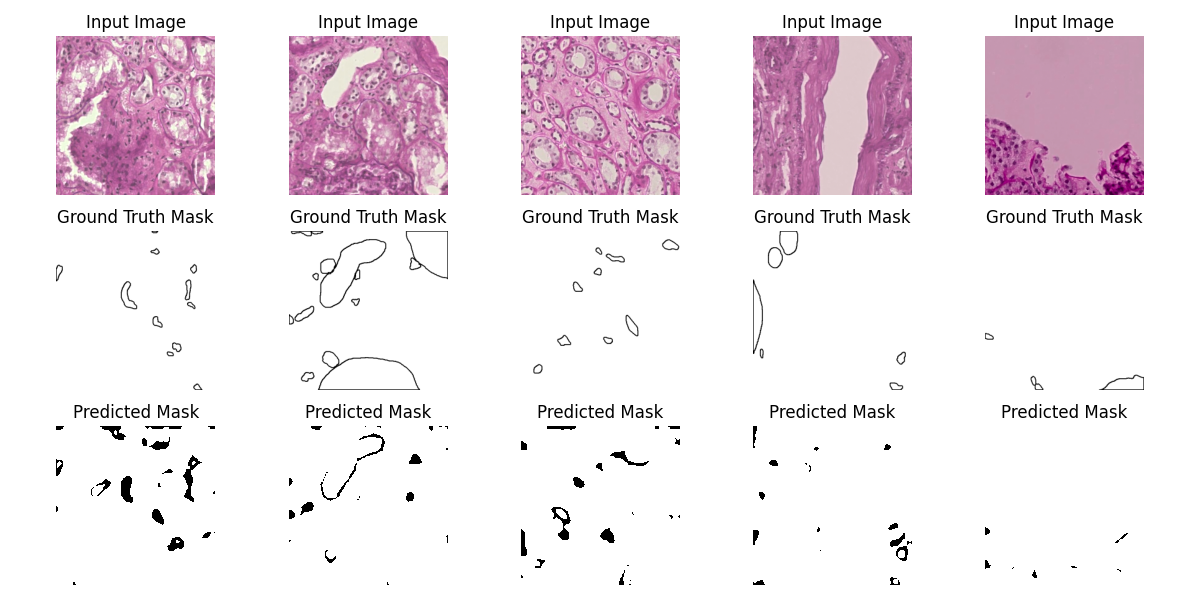
EfficientnetB1 normalised 100 epochs 256X256 No early stopping





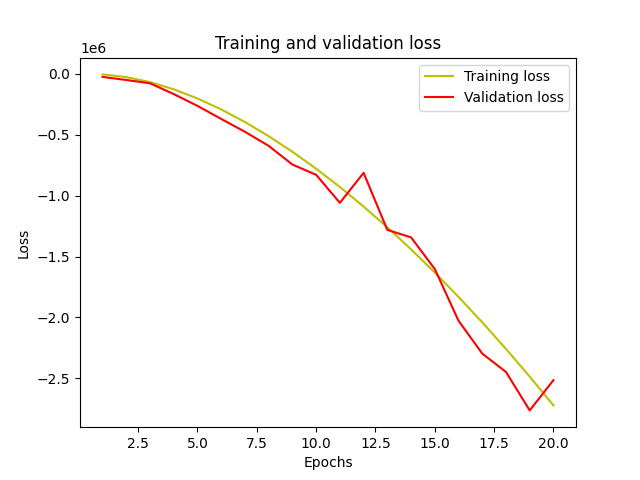
Seresnext50 normalised 100 epochs 256x256 no early stopping

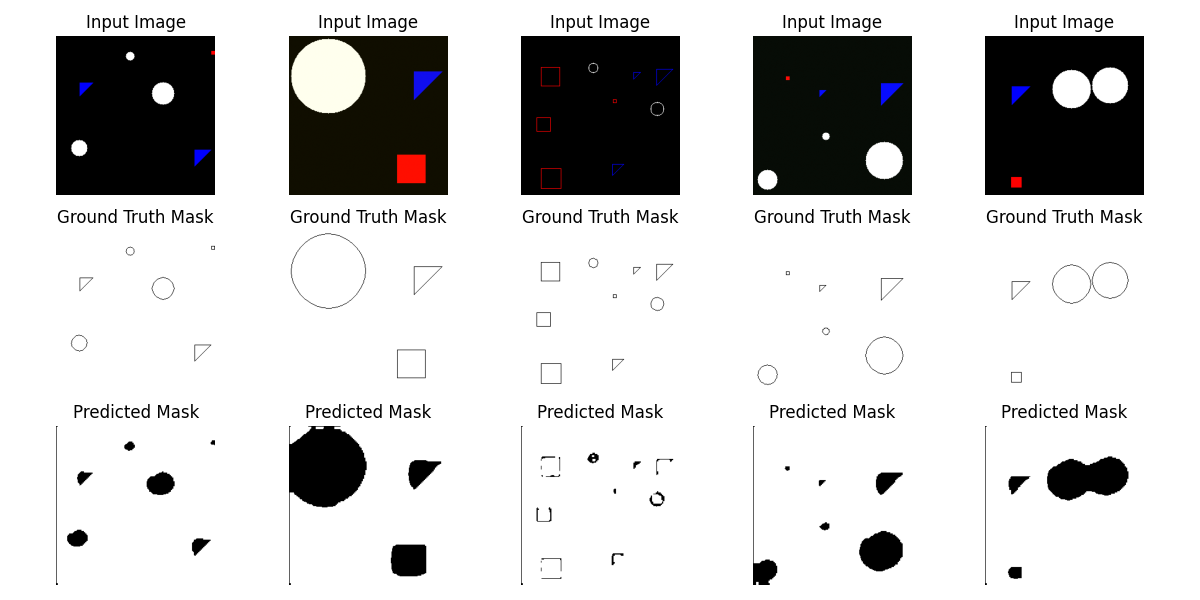




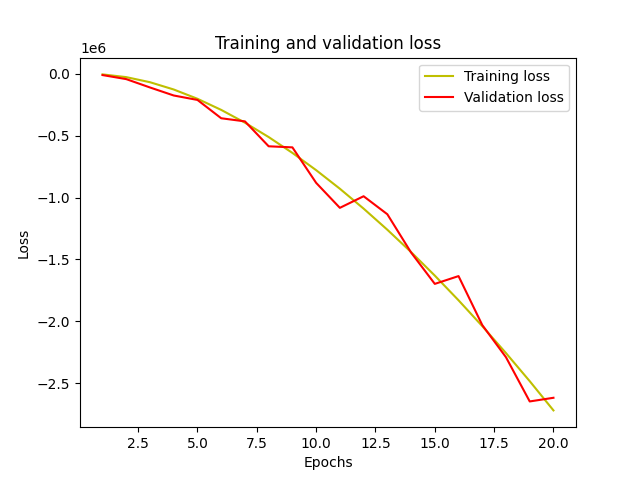
Conclusion: With the masks and images fixed early stopping removed and bad predictions were still a problem. Removing the early stopping code helped. Decent results from normalization also. I think maybe 200 epochs will get more useable results. However, which backbone will still be a matter of more short tests around 20 epochs each.

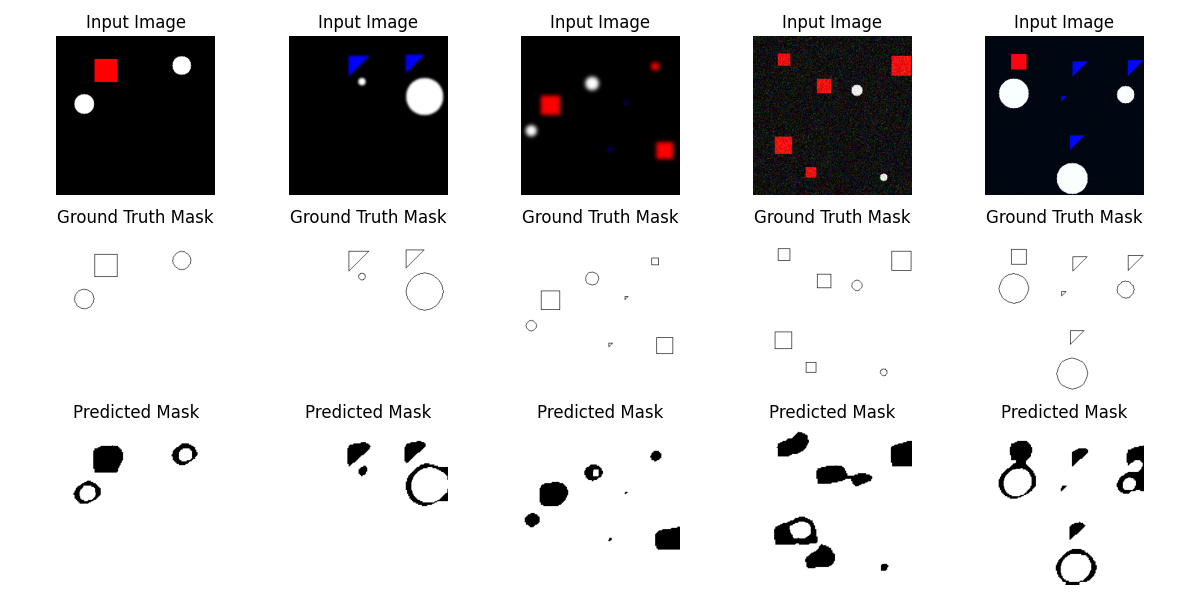
Synthetic blemished Normalised 256x256 without earlystopping seresnext50 20epochs



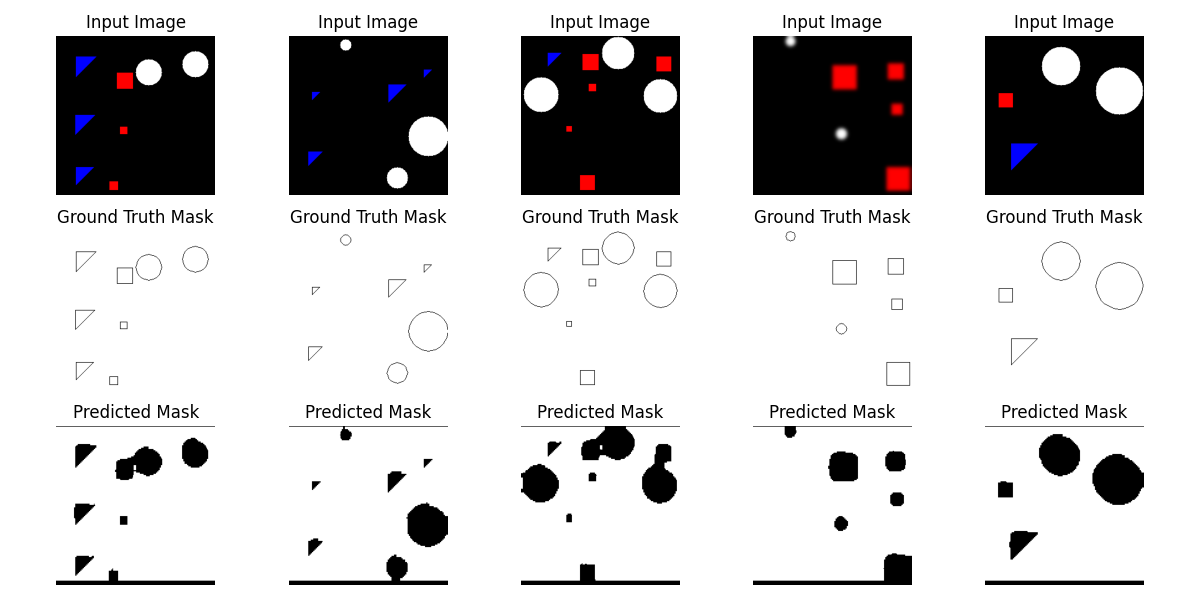
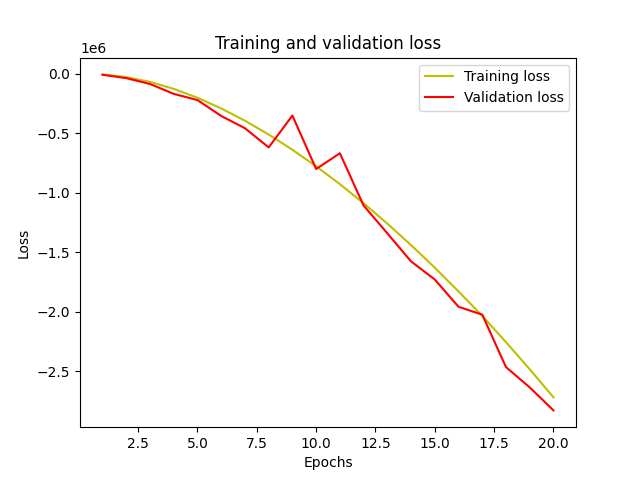


Synthetic blemished Normalised 256x256 without earlystopping efficientnetb1 20epochs





Synthetic blemished not normalised 256x256 without earlystopping efficientnetb1 20epochs



Conclusion: For the synthetic data both normalized and colour images work well. With the blemishes more training is required to get ground truth quality mask which was not the case with the unblemished synthetic data. The fact remains that the code still works and can be deployed with more epochs for a better result. It might also be prudent to rework the code so it gives us a progress chart and comparison every 10 or 20 epochs.