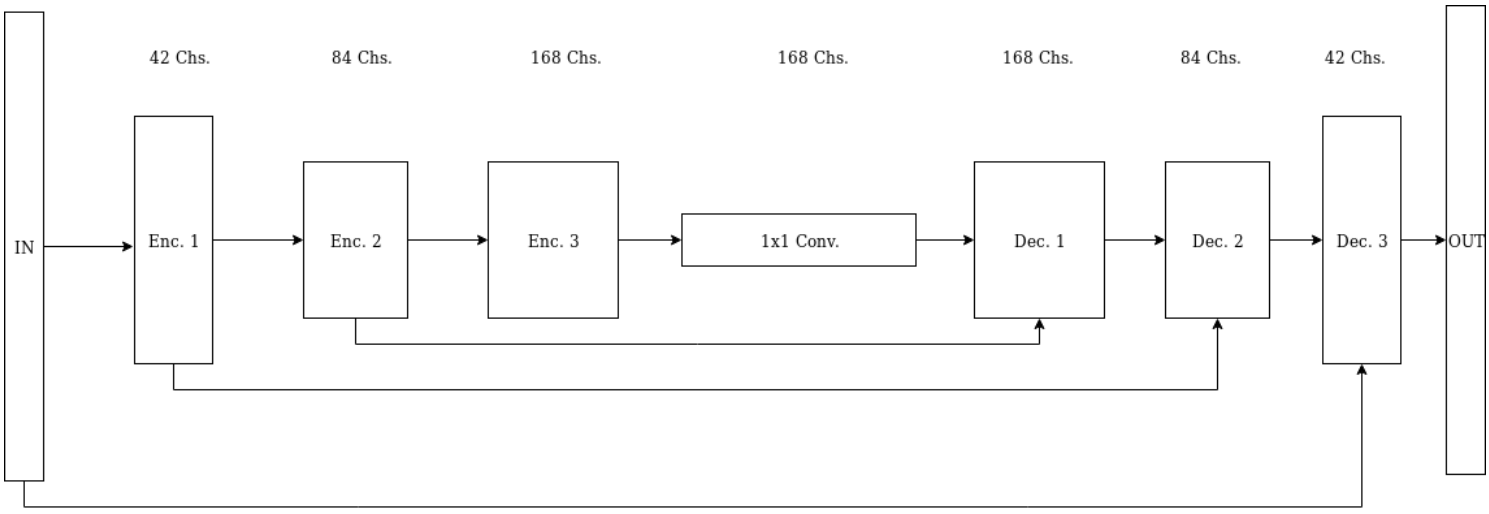
Project 3: Follow Me

The purpose of the project was to to train a Fully Constitutional neural network to identify a certain person to be used on a drone to follow them around.

Network Architecture:

The network architecture was defined in the fcn_model function in the notebook. It consisted of a 3 encoder blocks to identify features in the image, which are fed into a 1x1 convolution to do the classification of the target. Finally, this data is fed into the decoder to scale up the image to the original size to identify not only the target but where it was in the original image.
The images start wide then get narrower as we get into the encoder part, while increasing the depth of the image to get multiple kinds of information from the image.

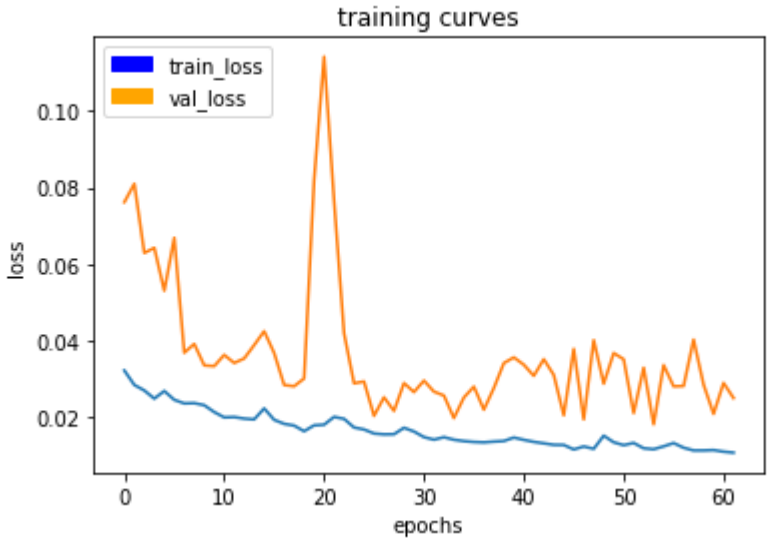
This diagram shows the final network architecture that was chosen:



Before finally choosing this network, several architectures were tested. Mainly these 2 architectures:

- 2 encoders, 2 decoders, filters parameter set to 5: this one turned to be pretty small for the task and stopped improving after around 25 epochs with a final score of 24%
- 3 encoders, 3 decoders, filters parameter set to 10: this one was clearly better but also stopped improving after around 30 epochs with a final score of 34.8%

For context, the final one reached a final score of 43.56% after training for 60 epochs and it still had room to improve which is shown in the graph.



Choices of hyperparamters:

- `learning_rate = 0.004`
  This was chosen mainly based on experimentation. The general analogy is a higher learning rate may result in faster training but with a noisier progress (it may not settle to a minimum without annealing). But a lower learning rate would take gradual less noisy steps towards a minimum (but it may settle in a local minimum).

- `batch_size = 128`
  This was also chosen based on experimentation. A higher batch size would make a for a better random sample to use for gradients update, which means less noisy progress and potentially less epochs to train. But a smaller batch size would result in lower computation time and potentially less training time overall.

- `num_epochs = 100`
  This number was stretched to see how the network would perform if the training was stopped and to be able to stop it midway if wanted.

- `Workers = 8`
  This one was chosen as a higher value than the default (2) because the network training was a little bit slow on the default setting.

The use of encoders and decoders:

Encoders are used to extract features from the image. At first, it catches smaller features and as the layers progress it extracts more apparent features by combining the previous features to finally use a 1x1 convolution to classify the pixels.
What we did until now is that we identified if the desired target exists in the image or not. But the drone needs know where to head, so here comes the decoder into play. It scales up the data coming from the 1x1 convolution while combining it with higher resolution data from multiple positions form the encoder part to produce the segmented output in the same size as the input.

One drawback of using FCNs is that the data entering the decoder has lost a lot of detail from the original image so the decoded result won't be as accurate. A special technique is used to retain that detail back from the early stages. It is called **skip connections**, where an early layer from the encoding phase gets concatenated to a layer in the decoding phase to use the information from the early layers that have not lost as many features.

Another technique that helped is batch normalization. It is based on the idea of normalizing the data before entering it in each layer instead of normalizing the input of the whole network only. It helps to cut down on the training time, allows the use of higher learning rates and also provides a bit of regularization.

1x1 convolutions:

It substitutes the role of fully connected layers in doing classification. Also, while benefiting from the spatial information and also preserving it. On the contrary, the input of the fully connected layer needs to be flattened and the output would be a flat vector with no spatial information to reconstruct the image.

Working on classifying another object:

This combination of data and network would perform really poorly in classifying another object. The dataset needs to be recollected and the new object needs to be masked in each image.
Then the encoder part needs to be retrained on the new object, while freezing the first layer to cut on training time and benefit from the knowledge of the lower level features which are probably the same.

Future improvements and real-world deployment:

To deployed in the real world, the dataset needs to have more variety especially images with the hero far away. Because that is an area where the classifier is not performing well.
It can also benefit from using better training techniques, like learning rate annealing with restarts. The annealing part is used to make the network make more steady progress as ti reaches a minimum. The restarts would also make the network get out of a local minimum if gets stuck into one.