**Deep Learning Project - Follow Me**

**Objectives**:

* Program a Quadcopter to identify a target and to follow it.
* Consolidate the knowledge on the lesson by also inviting to look external material and search.
* Expose to computation complexity and solutions like AWS servers (EC2 instances -> p2.xlarge) which show that GPUs are very key to deep learning application.
* Importance of utilizing proper library and tools likes TensorFlow and Keras.

##### **Primary Goal**

The primary goal is to locate a target (the Lady in red) and follow her. This is done by analyzing the images coming from the drone (a quadcopter) front camera. A Fully Convolutional Neural Network, analyzes and classify each pixel of the image stream. All happens in a simulation environment, QuadSim, which is run locally.

##### **Issues**

The main issue encountered in this project was the long training time. Although I followed the procedure to have access to an Amazon high performance server. I never managed to get one. My ticket is still open and I believe someday it will be processed. I have to use my local machine for this purpose therefore training the net and tuning parameters has taken a very long time (days). More trials surely can bring better settings but is not doable without a high performance machine. I have discussed this point also with my mentor who would see as a solution having the Amazon guys solving the ticket and extending the limits. Since this is going on for some days now and I am already in overdue time. I need to submit the project although with the min accepted accuracy > 40% and move on with the degree into the second semester.

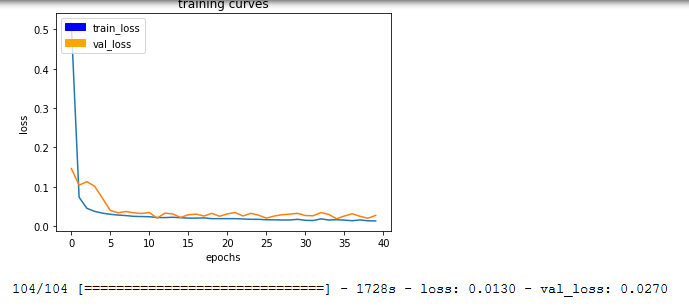
**Parameters chosen for this project**

* **Batch\_size = 40**: number of training samples/images that get propagated through the network in a single pass.
* **Num\_epochs = 40**: number of times the entire training dataset gets propagated through the network.
* **Steps\_per\_epoch = 104**: number of batches of training images that go through the network in 1 epoch. A recommended value to try would be based on the total number of images in training dataset divided by the batch\_size.
* **Validation\_steps = 30**: number of batches of validation images that go through the network in 1 epoch. This is similar to steps\_per\_epoch, except validation\_steps is for the validation dataset.
* **Workers = 10**: maximum number of processes to spin up. This can affect your training speed and is dependent on your hardware.

**Model**

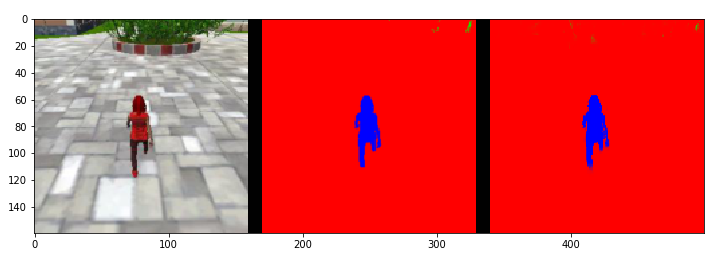
**Results**

*Final step of the training phase (epoch = 40)*

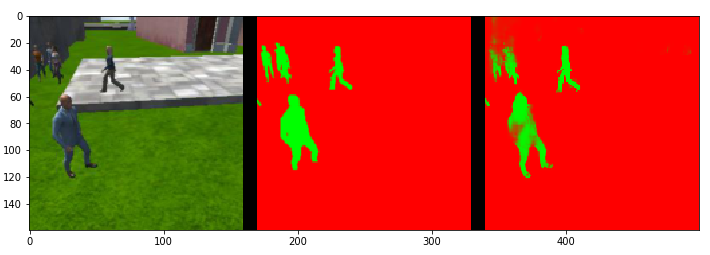


Note the very long time taken for one epoch (1728 s) which is a good representation of the average value per epoch.

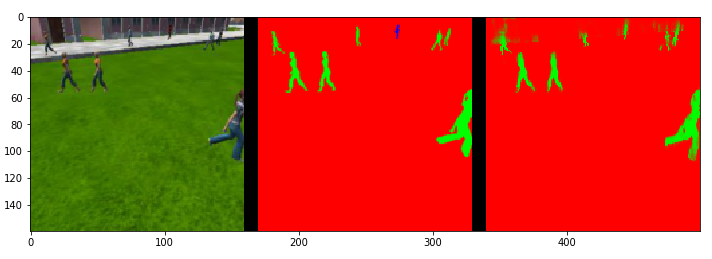
*Image of the target (while following)*



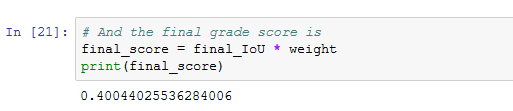
*Patrolling phase without a target*



*Patrolling phase with a target in it*



*Final score*



**Future Improvements**

It is clear to me that further work can be done to improve the model and the training together. As a started I would consider to improve the dataset and use eventually higher resolution in the images, although this means increasing the computation requirement.

Increasing the number of epochs and decreasing the learning rate might allow for better values of all parameters in the model, therefore to a higher accuracy in classification.

Additional layers and filters might also increase the chances of better identifying features leading then to higher accuracy in the predictions.