Machine Learning Exercise

Task

Using the two images provided `rgb.png` and `gt.png` (ground truth), your task is to implement and train a Fully Convolutional Neural Network (FCNN) for the task of segmenting an image into two classes. You may choose whichever framework you are most comfortable with e.g. PyTorch or Tensorflow.

The network architecture is as follows:

`Input->Conv1->ReLU->Conv2->ReLU->MaxPool->Conv3->ReLU->UpSample->Conv4->Output`

The layer details are as follows:

- `Conv1` - 3x3 kernel, 16 channels, padding 1
- `Conv2` - 3x3 kernel, 32 channels, padding 1
- `MaxPool` - 2x2 pool size
- `Conv3` - 3x3 kernel, 16 channels, padding 1
- `Conv4` - 5x5 kernel, 2 channels, padding 2

The input patch is of size WxHx3 (RGB), and the corresponding output patch is of size WxHx2.

Requirements

- The code must be implemented in Python 3 and you are allowed to use image loading modules such as skimage, OpenCV or PIL.
- For training, write a script `train.py` that trains the network on the training image.
- Add the necessary code to plot the training loss as a function of the number of epochs and save this as `loss_vs_epochs.png`.
- Assume that at some point we will train this algorithm on a GPU and that the network cannot fit in the GPU memory for an input image larger than 256x256x3.
- Write a script `predict.py` that uses the trained model to make predictions on the training image. The restriction on the GPU memory from the previous item also applies here.
- You may also create additional python files for utility purposes if you see fit.
- Save the final prediction image with your best results to a file called `predictions.jpg`.
- Provide instructions on how we can execute your code as well as a description of your approach in the `README.md`. We will be running your code so ensure that it will run "out of the box".
- Provide a `requirements.txt` in the standard format.
- Assume we will run your `predict.py` script on an new image similar to `rgb.png` to assess the model performance. You are therefore encouraged not to overfit.
- Failure to provide all items in `bold` could result in immediate rejection.

Notes

- You may add additional **non-convolutional** layers if you feel it may help.
- There is no need to run this on a GPU, we are satisfied with the code running on the CPU. This way you will be able to try it on a laptop or your home computer.
- You are NOT to share this exercise on Github, GitLab etc or risk **immediate rejection**.
- You may need to make some assumptions in order to implement the training and prediction code. If you have any questions don't hesitate to contact us.