Statistical Learning and Neural Networks

Image Segmentation and Neural Network Quantization

Lab duration: 3h

Download the starting notebook from Portale della didattica (lab3.ipynb) and upload it to Google Colab.

Exercise – Cultivated land detection

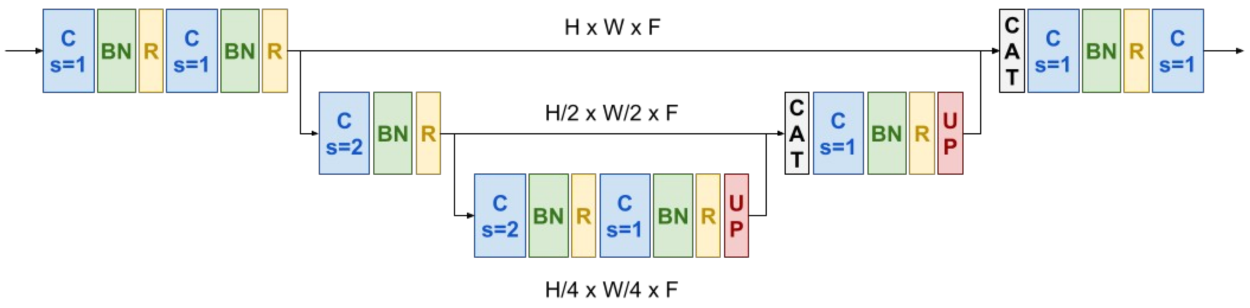
Detecting cultivated parcels of land is very important to monitor agricultural activities, optimize productivity and tackle climate change. In this exercise, we will use a modified version of the AI4EO Sentinel 2 dataset released by the European Space Agency (ESA).

Sentinel 2 is a satellite launched by ESA with the capability to capture Earth observation multispectral images at a resolution of 10 meters per pixel at 12 different light wavelengths (bands) from the infrared to the visible. Capturing the Earth reflectance at multiple wavelengths allows to detect materials because their properties make them reflect light in a different manner at the various wavelengths.

The modified dataset for this exercise is composed of 50 images with resolution 256 x 256 and 12 spectral bands. Note that bands 1,2,3 (counting from 0) are the usual Blue, Green, Red channels (mind the BGR ordering!) if you want to use them for visualization.
Each image is associated to a binary segmentation map whose value is 1 if the pixel corresponds to cultivated land and 0 otherwise. The dataset is provided as an npz file with two arrays: 'X' for the images and 'Y' for the segmentation maps. Code to load the training and testing data is already provided.

In this exercise, you will use the Sentinel 2 dataset to perform detection of cultivated land. This is an image segmentation problem where you want to classify each pixel as belonging to the "cultivated" or "not cultivated" classes.

The model to be implemented is a simple U-net. U-net is a popular segmentation architecture that is based on convolutional layers. The first part of the network uses strided convolution to reduce the spatial resolution and capture more and more global features. The second part of the network uses upsampling and convolutional layers to return to the target image resolution. This second part also receives features at the same resolution from the first part via skip connections. The skip connection concatenates the two sets of features along the channel dimension. The following picture shows the architecture to be implemented for this exercise:



Legend:	
C s=1	: convolutional layer with stride=1
C s=2	: convolutional layer with stride=2
BN	: batch normalization
R	: ReLU
UP	: upsampling by factor of 2 (hint: tf.keras.layers.UpSampling2D)
CAT	: concatenation along channel dimension

Train the neural network and test it on the test set . Measure the accuracy on the test set as your perfromance metric.

After that, apply post-training quantization by converting the model to TFLite and using the optimization option that only quantizes weights but keeps calculations as floats. Check the loss in accuracy due to quantization. Note you will need to code your own function to perform evaluation and measure accuracy on the TFLite model.

[Optional] Finetune the model using quantization-aware training.