Assignment 2

1 Segmentation task

You are supposed to implement a fully convolutional U-net like neural network for segmenting pixels into 3 categories. **Your are not allowed to copy code from online resources**. You should implement your solution in Tensorflow (if you want to use another framework, you should contact the teaching staff, but high level libraries like Keras will not be allowed).

Requested features:

- split the data into train and validation set (randomly),
- implement data augmentation: horizontal and vertical flips,
- when doing predictions on the validation set, average a few augmented versions of an image and rate the averaged predictions,
- you need to obtain cross entropy loss of at most 0.15 on your validation set.

2 Objective function

For each output pixel you should return a probability distribution over 3 classes in order to optimize the cross entropy loss.

3 Data

The data is located in directory `/data/spacenet2` at the ICM (tital-lab-gw). The ground truth for each pixel is obtained by reading the 3 channels RGB and dividing the values by 255 (note that for some pixel the ground truth might also be a distribution, so that you can have ground truth values (255,0,0), but also (100,155,0)).

The images are of size 650x650 and you should generate predictions of exactly this size. However, it is up to you to decide what is the exact size of input and output to your neural network due to scaling and cropping.

You can load the whole dataset into memory and provide the input to your neural network by `feed_dict` (using tensorflow queues and generate batches by reading and decoding images from hard disk on the fly is not required).

4 Benchmark

Your code should be able to reach the benchmark of 0.15 cross entropy loss in $2-3$ hours of training at ICM. However, if your solution is reasonable, but does not reach the benchmark in that time, you will still get a fair amount of points.

5 Logs

You should save the logs from your run, containing train and validation error for subsequent epochs as well as timestamps. Also, you should store a checkpoint to show the validation error during code inspection.

6 Scoring

Solving the problem without data augmentation gives a fraction of 2/3 of available points.

7 Additional solution features (nonobligatory)

If you want, you can additionally implement:

- tensorboard visualization of input, output and ground truth, images,
- more augmentation - scaling, rotations,
- do not store the whole dataset in memory, but use tensorflow queues to prepare batches of data on CPU (with several threads) which the training step on GPU is performed,
- use dilated convolutions.

8 Note

Implementing features from the nonobligatory list might help you getting full score even in case of minor drawbacks.

9 Deadline

You should submit your solution by email by 23:59 on 30.05.2017 to `cygan@mimuw.edu.pl` and `mucha@mimuw.edu.pl` with email title "[GSN] Assignment 2". Your code will be inspected either during lab session on 31.05 or 07.06.