Exercise 01

Deadline: 25.04.2018, 1pm
Regulations:

- You need $\geq$ **50%** **of all points** in the weekly exercises and successfully work on a $\geq$ **2 weeks fulltime project** and submit a **written report** in order to pass this class.
- Hand in your solutions in **groups of two or three people**.
- Make sure your code works with python 3.6 or later.
- For exercises that include coding, hand in a **Juypter Notebook** for each exercise containing all your answers, results, plots and code. Separate the different sub-exercises clearly by using headings and describing text. Label each file with the number of the exercise. Before submission, make sure your Jupyter Notebook compiles without any errors when you do Run All. Additionally hand in a PDF version of the Notebook.
- For exercises that can be solved without coding, hand in a **PDF file** for each exercise containing all your answers and results. Separate the different sub-exercises clearly by using headings and describing text. Label each file with the number of the exercise.
- Submit all your results in a **single** .zip archive. The title of the submitted ZIP-file as well as the subject line of your email **must** start with **EX01** followed by the full names of **all group members**. **People not mentioned in the subject will not get credits.**
- Submit all your results to mlcvss18@gmail.com.
- Submissions that do not adhere to the deadline and regulations, or that are chaotic, will not be graded.

1 Unstructured semantic segmentaiton (20 Points)

This week's exercise will focus on unstructured prediction. You will compute different features and then use a Random Forest to generate a probability map for the different classes. This classifier only computes an unstructured, pixel-wise prediction. These prediction maps will be noisy but they can be used as the likelihood term. Next week's exercise will study the prior in more detail, and later in the semester we will use neural networks instead of Random Forest.

1.1 Download and load the data (4 Points)

The images for this exercise are taken from [1]. They are Optical Coherence Tomography (OCT) images of the retina and the goal of this exercise is to label each pixel as either background or as one of the 4 different layers in the retina.

- Download images_subject02.h5, images_subject10.h5, labels_subject02.h5 and labels_subject10.h5 from `https://hci.iwr.uni-heidelberg.de/ial/mlcv`.
- images_subject*.h5 contain 8 OCT images each, from two different subjects
- labels_subject*.h5 contain the corresponding labels. The classes are labeled with integers between 0 and 4. Unlabeled pixels have value -1.
- Load the images and the labels in python (Hint: use the package h5py. You can use HDFView to view *.h5 files.)
- Use images 0, 1 and 2 from subject02 as training set
- Use image 0 from subject10 as test set
- Visualize the training set and the labels on the training images.

1.2 Features (6 Points)

The Random Forest classifier generally only works well when informative features are used as inputs. Pixel values itself are not enough to get a useful prediction since they do not contain information about the pixel's surrounding area. Convolutional filters are commonly used instead.

- Use different filters (Gaussian, Gaussian-Laplace, Gaussian Gradient Magnitude) with different scales ($\sigma = \{0.7, 1, 1.6, 3.5, 5, 10\}$) from https://docs.scipy.org/doc/scipy/reference/ndimage.html and compute them on the training images.
- Compute different non-linear features (structure tensor and hessian eigenvalues) from http://scikit-image.org/docs/dev/api/skimage.feature.html on the training images using the same scales.
- Plot the filter images using e.g. matplotlib: http://matplotlib.org/users/image_tutorial.html (Hint: matplotlib.pyplot.imshow()).

1.3 Random Forest (6 Points)

- Extract the labeled pixels with their features. The result should be a numpy array with shape $(n_{\text{labeledpixels}}, n_{\text{features}})$ for each training image, where $n_{\text{labeledpixels}} = 259904$ and $n_{\text{features}} = 42$. Concatenate the results from the three training images such that the resulting numy array is of shape $(779712, 42)$.
- Train a Random Forest classifier on this data using http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html with default parameters (ten trees, at least two samples required to split, at least one sample in leaf nodes, no depth limit).
- Predict the class probabilities for the full test and training images and show the result for each class.
- Save the predicted class probabilities at each pixel for all training and test images in a .h5 file. (These results can be used as the likelihood in a later exercise.)
- Plot the predicted label images for training and test set by taking the argmax of the predicted class probabilities at each pixel.

The result will probably look reasonable but noisy. You should e.g. see single pixels with the wrong class in otherwise correctly labeled areas. In the following weeks you will learn how to fix those noise issues by enforcing structure in the prediction.

1.4 Accuracy score (4 Points)

In image segmentation the F_1-score is often used to compare the performance of different methods. Compute for each class the precision P, recall R and the F_1-score according to

$$P = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$
$$R = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$
$$F_1 = \frac{P \cdot R}{P + R}$$

References

[1] S. J. Chiu, M. J. Allingham, P. S. Mettu, S. W. Cousins, J. A. Izatt, and S. Farsiu. Kernel regression based segmentation of optical coherence tomography images with diabetic macular edema. *BIOMEDICAL OPTICS EXPRESS*, 6(4):1172–1194, 2015.