Problem set 6: Deep-learning-based image processing

Theoretical image science
Mats Persson,

February 2022

Discussing solutions is encouraged but you must **individually** write and hand in your solutions. For numerical computations and plots, you can use the programming language of your choice, e.g. Matlab or Python.

These homework problems build on material covered in all lectures, in particular the two last ones.

1 Deep-learning-based image denoising and the channelized hotelling observer (10p)

In this exercise we will use a demonstration script from Mayo Clinic where neural networks are trained to denoise low-dose x-ray computed tomography images and thereby generate images that look like normal-dose (i.e., less noisy) images: `https://github.com/RSNA/MagiciansCorner/blob/master/CNNDenoisingTutorial_MagiciansCorner.ipynb` An introduction to how to run it can be found in this article: `https://pubs.rsna.org/doi/epdf/10.1148/ryai.2019190072`

To run this code you need python with the deep-learning packages tensorflow and Keras installed. To avoid having to set this up, you can create a Google Colab account and run it directly in the browser.

When running locally, I had to comment out the first line `%tensorflow_version 1.x`. Also, the second cell will only work on unix-type systems. On Windows, download the dataset manually `https://docs.google.com/uc?export=download&id=1-ZqL_1cqWeG6LsRABOTwiddW8TgQ-q70` and extract to create a subfolder `Denoising_Data` in the folder where your notebook is located.

a) Run the notebook in order to train deep convolutional neural networks for denoising with two different loss functions: MSE loss and perceptual (feature) loss. (Even if you do not have access to a GPU, you should be able to train this in 5-10 minutes.)

Which of these networks is most successful in preserving the spatial resolution of the image? (2p)

b) To investigate the effect of the number of channels in each layer, try changing to four channels instead of the default 64 and retrain the network. How does this change the resulting image? (1p)

c) After each layer of the network, there are four image channels. Display these four images for each of the six `conv2d` layers in a 6×4 plot matrix and describe what is happening as the images is processed through the network. (1p)

e) Now change back to using the original 64 channels. Retrain the network with l_1 (mean absolute error, "MAE") loss instead of MSE loss. Include this network in the line profile plot to compare the resolution to the MSE and feature loss. (1p)

The next step will be to evaluate the resulting image quality using the Channelized Hotelling observer (CHO). Unfortunately I am not aware of a publicly available python implementation (if you know of one, please let me know!) If you want to avoid having to implement the CHO in python, you can export images from python to matlab with `sio.savemat`. and use the Matlab implementation of the CHO in the image quality toolbox:
`https://wp.optics.arizona.edu/cgri/objectives/image-quality-toolbox/`
Note that there is a bug in the code. Line 36 of `RunExperiment.m` should look like this:
`subplot(2,2,i),imagesc(reshape(s(:,i),128,128));colormap(gray);` In this problem you can use the same Laguerre-Gauss channels that are used by default in this script.

f) Generate 200 random images, 128 pixels $\times$ 128 pixels, filled with white noise of $\sigma = 70$ Hounsfield Units (HU), and add a circular disc with a radius of 10 pixels to 100 of these. Run these images through the neural network and measure the resulting channelized Hotelling SNR. Adjust the image value inside the disc so that the CHO has an AUC of approximately 0.8. Then compare the AUC obtained with MSE, MAE and feature loss to the AUC of the CHO applied to the original, noisy image. If the results are too heavily affected by statistical uncertainty, you may want to increase the number of random images. (5p)

Note: If you have done HW2.5 so that you have code to generate ramp filter noise, feel free to use this kind of noise instead of white noise. Ramp filter noise is more realistic in this case since it is more similar to the noise in the training dataset.