CS7GV1 – MT2019 - Assignment (B) 40 marks (40%)

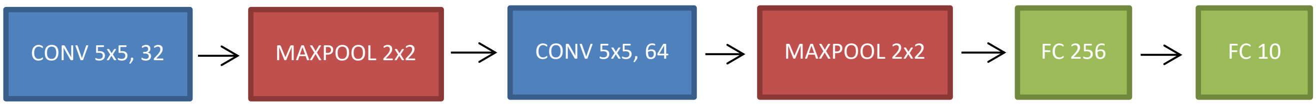
Deadline: Sunday 24th November 2019 @5pm on Blackboard.

Do **not** zip the files. Use prefix *YourEmail_* in the name of each file you submit. Your submission should include:

- The python codes with comments and a short report (*YourEmail.pdf*), or the jupyternotebook code with its output.
- Files with your results: *YourEmail_out.csv, YourEmail_params.csv, YourEmail_XXX_pretrained.png, YourEmail_XXX_clean.png, YourEmail_psnr.csv, YourEmail_XXX_noise.png, YourEmail_predicted.png, YourEmail_iou*.csv

Marking is based on the clarity and conciseness of codes and reports, correctness and speed of code execution, as well as the pertinence of the evaluations. Make sure to acknowledge all your sources of information for completing this assignment.

1) **MNIST classification (20 marks):** You are provided with code *mnist.py* that trains a Convolutional Neural Network on MNIST dataset. You are required to modify this file.
   a) In the file *mnist.py* identify where the Stochastic Gradient Descent optimizer is created. Train the default CNN architecture by choosing appropriate parameter values for:
      i) Learning rate – explain what happens when you use too large or too small value and explain why it is happening.
      ii) Weight decay – explain what happens when you use too large or too small regularization and explain why it is happening.
   b) Correct the mistakes in CNN2 and train it on MNIST train set. Desired architecture of CNN2 is displayed on following diagam:



   CONV KxK, N represents N features extracted by KxK filters, FC N represent fully-connected layer with N nodes. Set the padding so each convolution preserves input feature size.

   c) Change dataset to fashion MNIST (https://research.zalando.com/welcome/mission/research-projects/fashion-mnist/, hint: take a look at torchvision.datasets), estimate the dataset mean and standard deviation and use it to normalize the data in the data loader.

   d) Design CNN3 with additional regularization of your choosing. Explain benefits of such regularization and report its accurracy on fashion MNIST and its relative improvement over CNN2.

   e) Download file *YourEmail_in.csv* from the course drive. File contains 5x5 array of integers *X*. Manualy compute *valid* convolution

$$(f * g)[n,m] = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} f[i,j]g[n-i,m-j]$$

of *w* and *X* and submit output in file named *YourEmail_out.csv* (note the difference between convolution and cross-correlation, see https://en.wikipedia.org/wiki/Convolution).

$$w = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & -1 \end{bmatrix}$$

f) Download file *YourEmail_network.csv* from the course drive. File contains network definiton in the following format:

   i) Convolution, N, K – convolution layer with N feature maps and KxK filters

   ii) Maxpool, K – max pooling layer KxK pooling window

   iii) Fully-connected, N – fully connected layer with N neurons

   All convolutions use *same* padding that preserves input feature size. All convolutional and fully-connected layers use bias. The network is applied on 32x32 RGB images (3 input channels). Calculate the number of parameters and the number of FLOPS of this network and submit them in *YourEmail_params.csv* each of the 2 values in separate row (number of parameters in the first row).

2) **Image denoising (10 marks):** Read paper *Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising* (https://arxiv.org/abs/1608.03981). From the class drive download archive *DnCNN.zip* which contains the dataset, data preparation, training and testing scripts. You are provided with DnCNN-s model pretrained to denoise images with σ=25.
   a) Briefly explain how the authors have decided on the particular network depth. Which tasks is the DnCNN-3 designed to solve?
   b) Use *main_test.py* to test the pretrained model on denoising the whole Set68 with σ=25 and report the obtained average PSNR and SSIM metrics. Download *YourEmail_sigma.csv* that contains specific value of σ. Test the same model on the same set corrupted by this particular σ and report the results.
   c) Finetune the pre-trained model using script *main_train.py* for 1 epoch with learning rate 0.0001 with σ level given from *YourEmail_sigma.csv* and test its performance (PSRN/SSIM) on Set68 using this corruption level.
   d) Download *YourEmail_XXX.png* containing corrupted version of *data/Test/Set68/testXXX.png.* Denoise this image with the pre-trained model, submit it as Y*ourEmail_XXX_pretrained.png* and save both PSNR, SSIM values for this image in *2* columns preserving ordering, in a file named *YourEmail_psnr.csv.* Repeat the procedure with your finetuned model, save result to Y*ourEmail_XXX_clean.png* and report PSNR and SSIM of this image as a second row of *YourEmail_psnr.csv.*
   e) Modify the network to output the estimated noise (Residual image in Fig.1) instead of the clean image. Generate noise from Y*ourEmail_XXX.png* by your finetuned network and save it as a grayscale image centered around intensity 127 named Y*ourEmail_XXX_noise*.*png*. Report the standard deviation of the noise in this image.

3) **Semantic segmentation (10 marks):** Read paper *Fully Convolutional Networks for Semantic Segmentation* (https://people.eecs.berkeley.edu/~jonlong/long_shelhamer_fcn.pdf). From the class drive download script *segmentation.py*, containing example how to use pretrained models for semantic segmentation.
   a) Briefly explain:

i) How FCN upsamples predictions to match the original image size.
ii) How had the authors improved the coarse predictions produced by the deepest layer.

b) Download *YourEmail.png* and take a class index assigned to you from *classes.csv.* Modify *segmentation.py* so that you predict segmentation mask of this class (by FCN model) on the image given to you and highlight prediction via **red** mask blended with the original image. Submit this image as *YourEmail_predicted.png.* See example output for the class 4 (boat) below.

c) Decode the groundtruth image *YourEmail_mask.png* and calculate intersection over union (IOU) with the prediction for the class assigned to you. Report IOU in file *YourEmail_iou.csv.*