Homework

Goals:

1. Introduction to Pytorch.
2. Introduction to neural network training.
3. How to work with the school's GPUs.

Autoencoder:

As explained in class, the Autoencoder contains two networks, E and D. E encodes the input into a latent space and D decodes the latent representation to an image. The loss function is || D(E(x)) – x || , which means the output of the networks should be similar to the input. For more details, check out the slides of the second class or the web.

Pytorch tutorials:

1. Read https://pytorch.org/tutorials/beginner/deep_learning_60min_blitz.html
2. Read https://pytorch.org/tutorials/beginner/dcgan_faces_tutorial.html

Task:

Implement an Autoencoder.

1. The Autoencoder should be implemented in Pytorch and run over the GPUs using CUDA. You must read the Pytorch tutorials.
2. Use a small version of the FFHQ dataset. The dataset is located in "/home/ML_courses/03683533_2025/dataset".
3. First 1K images should be reserved for validation. Others should be used for training.
4. Image size should be 256x256 (Input and output).
5. Number of layers should be at least 8.
6. Latent representation size should be no more than 256 (flatten).
7. Use whatever loss functions you like – simple choices are L1 and L2.
8. Use batch size larger than 2.
9. Try several hyperparameters and architectures to see what works best.

Submission:

1. Code + Results.
2. Code – all code written. See best practices for writing code in Python & Pytorch. Code should be readable and clean.
3. Results – a PDF presenting your work.
    a. Report the results of your best model along with 10 other models – different hyperparameters, architectures, optimizers, losses.
    b. Report both qualitative (visual) and quantitative (numeric) results for each model.
    c. Report the training loss curves for each model.
    d. Present an ablation study of the models.
4. The team with the best visual results will get 10 points bonus, second best will get 5 points.

Tips:

1. Output should be similar to input, but far from being identical, think about why we won't get identical output.
2. Recommendation: Use BatchNormalization, LeakyReLU & Adam.
3. Make sure you are using the GPU.
    a. Docs: https://pytorch.org/docs/stable/notes/cuda.html
    b. Example: https://github.com/Natsu6767/DCGAN-PyTorch/blob/master/train.py.

Work over school's GPUs – relevant also for final projects!

1. Instructions on how to use the servers are provided In a separate document.