# Advanced Deep Learning 2024 Assignment 1

This is a **group assignment**, and its deadline is **Tuesday, May 7, 2023, 22:00**. You must submit your solution electronically via the Absalon home page.

Important rules and notes:

- All assignments in the course are either group or individual:
  - For individual assignments, you are not allowed to collaborate with anyone on the assignment and you are not allowed to communicate your solutions to other students. For group assignments, you are, of course allowed to collaborate internally in the group.
  - You must not ask for help from anyone except the teachers and TAs on the course. On the other hand, we encourage you to use the exercise classes and the Absalon forum to get help. The exercise sessions exist to help you with the assignments, and you are welcome to ask any questions related to the teaching material and the assignments on the forum.
  - If your solution contains material from other sources than the assignment text, you must cite the source of the material and any changes you have made. This also applies to material from textbooks, Absalon, etc.
  - If your solution uses methods or notation which are not used in the course material, you must specify where you have found the method or notation.
  - If you are in doubt about plagiarism or citation rules, please ask the teachers or TAs.

  Please be very observant of these rules. We do not want any plagiarism cases, both for your and our sake.
- A solution consists of:
  - A PDF file with detailed answers to the questions, which may include graphs and tables if needed. Do *not* include your complete source code in the PDF file except if you are asked to do so. However, you can show important parts (i.e., a few lines) of your source code.
  - A .zip file with all your solution source code with comments about the major steps involved in each question (see below). Source code must be submitted in the original file format, not as PDF.
- *Important: Do not zip the PDF file*, since zipped files cannot be opened in the speed grader. Zipped pdf submissions will not be graded.
- Your PDF report should be self-sufficient. That is, it should be possible to grade it without opening the .zip file. We do not guarantee opening the .zip file when grading.
- Your code should be structured such that there is one main file (or one main file per question) that we can run to reproduce all the results presented in your report. This main file can, if you like, call other files with functions, classes, etc.
- Your code should include a README text file describing how to compile and run your program, as well as a list of all relevant libraries needed for compiling or using your code.
- Handwritten solutions will not be accepted, please use the provided LaTeX template to write your report.
- Figures should be readable. They should have proper captions (same holds for tables) and labels, a legend when appropriate. In general, write proper sentences in the report.
- You are only asked to perform tasks that make sense from a machine learning point of view. If you do something that that does not make much sense, it is most likely wrong.

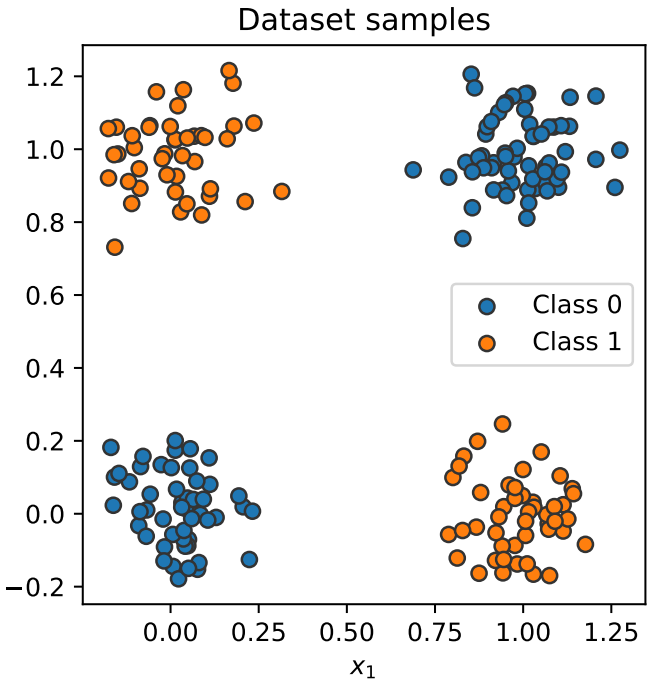


Figure 1: An example of the noisy-xor dataset with $\sigma = 0.1$.

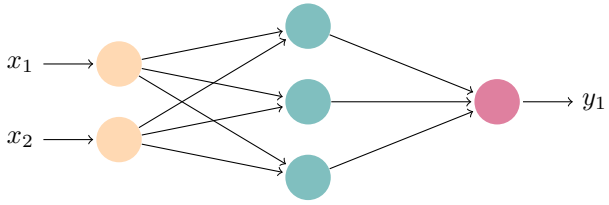


Figure 2: A simple feed-forward network shown as a graph.

# 1 Deep versus wide networks

In this assignment, you are going to work with feed-forward neural networks, and you are going to study the effect of widening and deepening networks on a non-linear classification task.

In this assignment, you will work with the Noisy-xor problem: $\{(x_1, x_2, y) : y = (x_1 \text{ xor } x_2) + \eta(\sigma)\}$, where $\eta$ is a normal distributed random variable with mean $\mu = 0$ and some standard deviation $\sigma$. An example of this dataset is shown in Figure 1. This is a non-linear classification problem since no single line can be drawn, which separates the orange from the blue points. However, some neural networks can classify such data well.

Consider the example of a simple feed-forward neural network shown in Figure 2. A PyTorch print of an implementation of this network is as follows:

```
Net(
  (model): Sequential(
    (0): Linear(in_features=2, out_features=3, bias=True)
    (1): Tanh()
    (2): Linear(in_features=3, out_features=1, bias=True)
    (3): Identity()
  )
)
```

where we see that the activation functions in the first and inner layers are tanh, and in the output layer, it is the identity function.

Figure 3: Examples of images in the MNIST dataset ["THE MNIST DATABASE of handwritten digits". Yann LeCun, Courant Institute, NYU Corinna Cortes, Google Labs, New York Christopher J.C. Burges, Microsoft Research, Redmond].

In the file, `feedforwardAssignment.ipynb`, accompanying this assignment, you will find a scaffold of a Jupyter notebook for generating noisy-xor data and for experimenting with a family of networks in Pytorch for the classification task. Your task is to:

1. Give a brief description of each part of `feedforwardAssignment.ipynb` in words.
2. Update `feedforwardAssignment.ipynb` such that it tests a range of networks for how well they can classify the noisy-xor data source. As a minimum try repeatedly all combinations of $0 \ldots 3$ hidden layers with widths $1 \ldots 3$. For each combination of depth and width, calculate the mean and standard deviation of the resulting loss function on a new data set.
3. Give a brief interpretation of the effect of changing the depth and width on the quality of classification in the above experiment. Calculate how many parameters each of the networks tested above has. Identify the training parameters used, and describe how changing them, may influence your conclusions on the quality of the classification.

## 2 Convolutional neural networks

In this assignment, you are going to work on convolutional neural networks, and you are going to compare the structure and efficiency of a feed-forward and a convolutional neural network on a classification task.

In this assignment, you will work with the MNIST dataset which consists of thousands of images of handwritten digits (0–9). Examples are shown in Figure 3.

In the file, `feedForwardMNIST.ipynb`, accompanying this assignment, you will find a complete program for classifying digits using a feed-forward network. Its Pytorch print is as follows:

```
Net(
  (main): Sequential(
    (0): Linear(in_features=784, out_features=128, bias=True)
    (1): ReLU()
    (2): Linear(in_features=128, out_features=64, bias=True)
    (3): ReLU()
    (4): Linear(in_features=64, out_features=10, bias=True)
    (5): LogSoftmax(dim=1)
  )
)
```

That is, it consists of 3 linear layers, 2 ReLu, and 1 LogSoftmax activation function. Your task is to:

1. Run `feedForwardMNIST.ipynb` and give a brief description of each part of the program. How many parameters does this model use, and how well is it able to correctly classify each digit class 0, 1, ..., 9?

Figure 4: An example of an annotate lung X-ray from [2].

2. Based on `feedForwardMNIST.ipynb`, make a new program `CNNMNIST.ipynb` where the feed-forward neural network has been replaced with a convolutional neural network. The new network must also use:

```
Net(
  (main): Sequential(
    (0): Conv2d(1, 16, kernel_size=(3, 3), stride=(1, 1))
    (1): ReLU()
    (2): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    (3): Conv2d(16, 32, kernel_size=(3, 3), stride=(1, 1))
    (4): ReLU()
    (5): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
    (6): Flatten(start_dim=1, end_dim=-1)
    (7): Linear(in_features=800, out_features=10, bias=True)
    (8): LogSoftmax(dim=1)
  )
)
```

   Why must the Linear layer be preceded by Flatten, and why does it have 800 input features? How well does it classify digits as compared to the feed-forward network, and what are its number parameters relative to the feed-forward network?

# 3 U-Nets and MLOps

The learning goals of this part of this assignment are to get hands-on experience in using fully convolutional neural networks for segmentation in practice, to improve understanding of PyTorch code, and to use an MLOps framework in practice.

We will consider the popular U-Net [1], which has become state-of-the-art in medical image segmentation. Consider the notebook `x_ray_segmentation_ERDA.ipynb`, which applies the U-Net to the lung segmentation task in x-ray images described in [2]. An example of an X-ray image with an expert's annotation of the lung pixels on top is shown in Figure 4

Below, you are asked to use an MLOps platform such as Weights and Biases. Tutorial and example notebook using W&B for tasks similar to the below are available on Absalon (week 1 module). You are free to use other similar frameworks as long as your choice supports the requested tasks.

Your tasks are to:

1. Briefly describe the structure of the notebook `x_ray_segmentation_ERDA.ipynb`.
2. Modify the notebook to log the training progress in the MLOps platform, and show the resulting plots in your report. Describe the modifications you made to use the MLOps platform and include code excerpts of the modifications in your report (should only be a few lines of code).
3. Implement a hyperparameter sweep using an MLOps platform, e.g., a W&B sweep. Sweep over the parameters you find most important (but keep the sweep of a size compatible with the compute resources you have - the size of the sweep will not affect the grading). Visualize the results using, e.g., a parallel coordinate plot in W&B.
4. Describe the experimental setup and the results in the report. You must describe what you did, present the results, discuss the results, and draw very careful preliminary conclusions.

# References


[1] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 9351 of *LNCS*, pages 234–241. Springer, 2015.

[2] B. van Ginneken, M. B. Stegmann, and M. Loog. Segmentation of anatomical structures in chest radiographs using supervised methods: a comparative study on a public database. *Medical Image Analysis*, 10(1):19–40, 2006.