# Guidelines for submitting Deep Learning exercise solutions

This document contains guidelines for submitting an exercise solution in the "Deep Learning" course. The goal of this document is for you as a student to get used to organizing your code projects and for the reviewer of the exercise solution to spend less time on figuring out how to run the code, who submitted it, what it depends on and to spend more time on the actual code and results.

## 1. Who to submit?

One and only one member of a group should submit the exercise solution on behalf of the group.

## 2. What to submit?

A submission is a single ZIP-file with the following naming convention: "Lecture_<lecture_number>_<group_id>.zip", where <lecture_number> is taken from moodle and <group_id> contains three digits randomly assigned by you and should be greater than 123 and smaller than 999. <group_id> should be kept the same through the course. <lecture_number> is the day number (1, 2 or 3) here.

## 3. What to include?

The following things should ALWAYS be included in the submitted ZIP-file:

- Lecture_<lecture_number>_<group_id>.<m/py> : The main script to run. As an example if a group with ID 888 submits a Matlab file for Lecture number 8 the main script should be called Lecture_8_888.m. A mandatory header should be provided in the top (see section 2 .1).
- Lecture_<lecture_number>_<group_id>.pdf: Pdf file containing results, comments, discussion etc., if any, provided by the group (see section 2 .2).
- data/: Folder containing any data used (e.g., images, data sets etc.). However, do not include any data that are given by the exercise.

If additionally custom functions are used which are in separate files, these files should be located in the root of the ZIP-folder.

### 3.1 Code Header

The following code should be used as a mandatory header and placed at the top in the main script, i.e., Lecture_<lecture_number>_<group_id>.<m/py> .

Matlab:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Group ID : <gr oup_id >
% Members : <name_of_each_person>
% Date : <date>
% Lecture: <lecture_number> < lecture _title> (see moodle)
% Dependencies: Toolbox, library, etc. needed to run, e.g., Libsvm, netlab, LDA tool.
% Matlab version:
% Functionality: Short Description. Example: This script trains a MLP for classifying
% handwritten digits. It also test the performance on a given data set for various
% settings.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Python:

```
# ####################################
# Group ID : <gr oup_id >
# Members : <name_of_each_person>
# Date : <date>
# Lecture: < lecture_number> < lecture_title> (see moodle)
# Dependencies: Toolbox, library, etc. needed to run, e.g., Libsvm, netlab, LDA tool.
# Python version:
# Functionality: Short Description. Example: This script trains a MLP for classifying
# handwritten digits. It also test the performance on a given data set for various
# settings.
# ####################################
```

### 3.2 Summary of results

A summary of the results should be included separately in a pdf-file. This file can contain (but is not limited to) accuracy/error-rate, plots, comments, and discussions. There is no strict requirement here, but as a general rule the summary should convey the results etc. without having to read or run the actual code.