ELEC0135 Applied Machine Learning Systems II (23/24)

Assignment

General Overview

The AMLS assignment comprises individual code writing, training and testing on data, an individual report in the form of a conference paper, and supplementary material provided online via an online code repository of your choice. You are allowed to discuss ideas with peers, but your code, experiments and report must be done solely based on your own work.

Assignment summary

1. The assignment leverages elements covered in:
    a. The AMLS II lectures,
    b. The AMLS II lab sessions, and
    c. Relevant research literature, competitions and public domain datasets associated with machine learning systems.

    A number of public-domain research competitions ("challenges") are referenced in this assignment. Each student must pick one of them for their assignment and follow the challenge description in order to provide their own solution addressing the challenge. More specific details are presented in the assignment description section. The presented results for each assignment will be based on the knowledge acquired in the lectures and the labs. Each student should also search the relevant literature for additional information, e.g., papers on state-of-the-art methods in machine learning for the chosen competition and task.
2. Each solution should be implemented in a programming language of the student's choice, e.g., MATLAB, Python, C/C++, Java, etc. However, please note that the labs of the module are based on Python and Tensorflow. You are allowed to use third party components, e.g., components of neural network designs, ways of making performance more robust to noise, etc., but your report and code must clearly highlight what was used from third-parties and what was developed or customized by you.
3. Write a report summarising all steps taken to solve the tasks, explaining your model and design choices. In the report, you should also describe and analyse the results obtained via your experiments and provide accuracy prediction scores on unseen data. Please refer to Report and Code Format, and Marking Criteria section for more details about the report.

Goal of the assignment

- To further develop your programming skills.
- To further develop your skills and understanding of advanced machine learning systems.

- To acquire experience in dealing with real-world data and real-world research challenges.
- To develop good practice in model training, validation and testing.
- To read state-of-the-art research papers on machine learning systems and understand the current challenges and limitations.
- To develop your writing skills by presenting your solutions and findings in the form of a conference paper.

Assignment Description

Challenges

We provide the list of challenges and subtasks from which to select your assignment:

1. Image super-resolution using machine learning, i.e., both Track 1 and Track 2 of:

   http://www.vision.ee.ethz.ch/ntire17/

   and using training data from https://data.vision.ee.ethz.ch/cvl/DIV2K/ while validating and testing on either a subset of the DIV2K data not used for training or on a different image dataset of your choice. You can also focus on one of the tracks of a subsequent NTIRE competitions (2018-2022 by changing the above link to ntire18,…, ntire22).

2. An on-going or past machine learning competition from Kaggle from the past 5 years, https://www.kaggle.com/competitions

   You can select an on-going or past competition and follow the competition rules in order to derive and validate your machine-learning based solution. You can access the test dataset(s) of the competition, which is something that is normally available to registered contestants, or you can create your own training, validation and test splits and use them to benchmark your solution. You must frame your results in the context of solutions that may have been published during or after the competition, or discussed in the competition forum. To avoid selecting a competition where you may quickly hit complexity limitations when running machine learning algorithms locally, we recommend focusing on image classification competitions, e.g.,

   https://www.kaggle.com/search?q=image+classification+in%3Acompetitions

   with particular emphasis on: (i) competitions of the last 5 years; (ii) competitions having more than 10 teams. If needed, you can focus on a subset of the data in order to make training and inference feasible.

   Important notes:

   i) Since in-depth analysis of machine learning for natural language processing is examined in a different module, refrain from focusing on NLP-oriented competitions. Focus instead on a competition that involves multimodal inputs for a recognition or classification task and/or an image/computer vision task.

   ii) Not all competitions have the same level of difficulty (e.g., “research” vs. “getting started” on Kaggle). Therefore, if you use a simple task, such as a

binary classification problem or a 10-class MNIST type of competition, we expect to see very significant emphasis on the analysis of results and ablations in order to compensate for the lack of sophistication in the chosen task.

For your chosen challenge, you should report training errors, validation errors, hyper-parameter tuning, and comparisons with related methods from the competition in terms of accuracy and complexity. It is important to make an algorithmic design that balances complexity versus accuracy, as you will not have access to GPU hardware. You may however explore options of free cloud credit for CPU/GPU hardware available for students via public cloud providers, or serverless versions of such services like Google Colab notebooks:

https://colab.research.google.com/notebooks/intro.ipynb

Overall, given your limited access to compute resources, you are allowed to limit the amount of training data with proportional reduction in the complexity of the inference task (e.g., number of classes for an image classification problem, size of the input/output images, number of features and supplementary inputs to use, etc.).

Report and Code Format, and Marking Criteria

Report format and template

We provide both latex and MS word templates in ***AMLSII_assignment_kit*** on the ELEC0135 AMLS II Moodle. The criteria for each part are detailed in the template. For beginners in latex, we recommend overleaf.com, which is a free online latex editor.

Once you finish your report, please export it into a PDF document and name it with the following format:

Report_AMLSII_23-24_SN12345678.pdf

Code criteria

You should write your code in modules and organize them in the fashion included in the folder structure of the AMLS II assignment kit mentioned above. If we wish to check the results of your code, we will first run your code with a script. Therefore, please make sure your project is named properly as demonstrated in the figure above.

- The '**Datasets**' folder does not have to contain the raw datasets. You can use it to save the datasets you may have generated (e.g., if you did pre-processing, data augmentation, etc.).
- The '**A**', '**B**' … folders should contain the code files for each subtask (depending on your assignment, you may have two or more subtasks).
- The **README** file should contain:
    - a brief description of the organization of your project;
    - the role of each file;
    - the packages required to run your code (e.g. numpy, scipy, etc.).

    The recommend format for **README** file is markdown (**.md**). **.txt** is acceptable too.

- We should be able to run your project via '**main.py**'. The structure of '**main.py**' has been provided.
- You are NOT going to upload your code and dataset to Moodle. Please refer to Submission session for more details.

Marking scheme

The mark will be decided based on both the **report** and **corresponding code**. In particular, we will mark based on following scheme:

REPORT	80%	CORRESPONDING CODE	20%
Abstract	5%	-	-
Introduction	5%	-	-
Literature survey	10%	-	-
Description of models (Use flow charts, figures, equations etc. to explain your models and justify your choices)	20%	-	-
Implementation (the details of your implementation, explain key modules in your code.)	20%	Correct implementation	10%
Experimental Results, Analysis and Conclusion	20%	The code is expected (or confirmed) to be producing reasonable results if executed	10%

It should be noted that – whereas we expect students to develop machine learning models delivering reasonable performance on the chosen challenge – the assessment will not be based on the exact performance of the models. Instead, the assessment will predominantly concentrate on how you articulate about the choice of models, how you develop/train/validate these models, and how you report/discuss/analyse the results.

Submission

- **Deadline**: Please see the ELEC0135 AMLS II Moodle page
- **Report submission**: you should only submit your report on Moodle:
- **Code submission**: You must include a link to your code in an internet repository that is publicly accessible in your report, but the link is hidden (e.g., GitHub, public Dropbox or Google Drive link, or similar).

  You are also encouraged to use GitHub to save and track your project. Use your UCL github account (or create an account) to start a git repository named **AMLS_II_assignment23_24/**. Make sure to back-up your code on the git repository regularly and keep your repository private so it is not viewable by other students. Changes made after the assignment deadline may not be taken into account. The code should be well documented (i.e., each class and function should be commented) and an additional README.md file containing instructions on how to compile and use your code should be created in the repository. If necessary, we reserve the right to test the code and we may ask you to provide us with your GitHub commit history evidencing how you gradually built and tested your solution.