UNIVERSITÀ
DEGLI STUDI
FIRENZE

Project Guidelines

Computational Learning

Daniele Castellana – DISIA – Università di Firenze

daniele.castellana@unifi.it

Recall: Exams

- **Final Project**
  - Develop and test a ML application
  - Write a report about the project
  - Focus on critical thinking!
    - Not only performances!

TWO DISTINCT GRADES

THE AVERAGE IS THE FINAL GRADE

- **Oral Exam**
  - Question on the course contents
  - No proof will be asked (but you should know the intuition)

What you should delivery

The project is composed of:

- The code which implement your solution to the assigned project. The code should be runnable

- A report which describe (almost) everything you have done to write the software. Also, it should include a guide on how to execute the software

When you should delivery it

- You submit your code and your report via Moodle
- I will create an assignment where you must submit the project (code + report)
- Once the project is evaluated, we can schedule the oral exam

Signing up in the Exam Session

REMEMBER TO SIGN UP FOR THE EXAM SESSION YOU CHOOSE

- https://sol-portal.unifi.it/

The project

- You can do the project in group of at most 2 students
- You implement one advanced model that we briefly discussed during the course. You should train and test the model on a single task.
- There will be different topics:
  1. CNN on Images
  2. LSTM on Text
  3. VAE for Anomaly Detection
  4. RL for control

The Code

What you should implement

The code should implement the whole ML pipline:

1. Manage the data
   - (Download), pre-processing, split, …
2. Model Selection
3. Model Assessment

Suggestion to save time

EACH POINT SHOULD BE RUNNED SEPARATELY (if desired)!

You should store intermediate results (e.g. the best config, the split, ecc..)

Organise the Code

- You must organise the code properly
  - **NO ALL THE CODE IN A JUPYTER NOTEBOOK**
    - You can use it to show results
- All the gold rules to write good software apply to ML
  - No redundant code
  - Organise your code in class (or packages)
  - All the other practice from Software Engineering
- This requirement is for your future!
  - Everyone in industry or academy wants organised code!

Reproducibility

- In ML code there are a lot random choice
  - Weight initialisation, split, permutation of the training data, …
- Every time you stare an execution, the first step is to **fix a seed and store it**
- This will help you **a lot** to debug your code
- Ideally, I should be able to reproduce the results you write in the report

Things to bear in mind

- Implement ML code is not easy, but **nothing is magic**!
  - You should always reason about the property of ML models
  - You **must** know the theory!

- If your implementation does not match the theory, there is a bug!
  - It might be the case that the bug is really hard to find
    - The important thing is that you discuss the mismatch in the report
    - Of course, a code without a bug will gain an higher score

- Remember that numerical errors can happen!
  - As we have seen to invert matrices

Danger Zone ⚠

- Everything that highlights you **do not understand the theory**
  - **If your implementation does not match the theory** and you have not discussed it in the report
  - **If you do not use data split correctly**
    - E.g. you use test set to do something else that is not generalisation error estimation

- Everything that **breaks the project rule** (next slide)
  - **You use library that are not allowed**

Rules for the projet

- You **must** use pytorch
- You **cannot use** any library to split data (e.g. scikit learn)

- You **cannot use** any already implemented Pytorch module
  - You **must** implement your own module!

- You can use pytorch autograd
  - You do not need to derive the derivative by hand
- You can use Pytorch optimisation algorithms
  - ADAM, AdaGradm, SGD, ecc..

The Report

Report Content

The report should contain almost everything you did to write the code and to execute it

1. Discussion on the ML models implemented
2. Implementation choice and details
   - Which library do you use, how to configure the environment, ecc..
   - How do you organise the code (and why you take this decision)
   - Any other implementation details that you found interesting (or painful) and why
3. Design choice about the ML experiments
   - The validation schema you choose and why
   - Which hyper-parameters you will validate and why
   - If you perform some trials, you can write all of them explaining your reasoning
4. The results of the experiments and their discussion
   - **Remember to make some plots**, not only table!

Big Advice

Write the report while you write the code

- The report should help you to:
  - reason about the problem you face in the implementation
  - remember what you have done and why