PROJECT AND EXAM FOR THE MASTER COURSE

SEMINAR 2
Academic year: 2024-2025
Teacher: Dr. Phuong T. Nguyen
University of L'Aquila, Italy
Email: phuong.nguyen@univaq.it
Website: https://www.disim.univaq.it/ThanhPhuong

1. Introduction

The project consists of an application realized in Python, running on the Colab platform, and a report to explain all the technical details. The project can be implemented by a group consisting of a minimum of 1 to a maximum of 3 students.

We remind you that the project has a weight of 80% on the final evaluation. The remaining 20% of the assessment will consist of a single oral interview aimed at ascertaining the knowledge acquired during the course. In addition, there will also be a discussion of the project in the interview.

The project must be delivered 7 days before the exam by sending an email to the teacher, expressing the willingness to take the exam. Therefore, the teacher and collaborators will consider the project developed by this deadline. This means that any subsequent changes will not be taken into account.

2. Common requirements

Each individual project chosen must be implemented with a good degree of flexibility and modularity and must take into account the following non-functional requirements/constraints:

- The data must be managed both in memory and through files present in the file system.
- The application must be designed in such a way that the use of one of the two persistence layers (memory or files) involves few changes to the application itself and, in particular, the representation (in terms of class / interfaces and relative methods) and the interaction with the so-called business logic must remain unchanged;
- Projects need to be regularly committed to GitHub. Before the exam, the lecturer will check each repository to see the history, and understand who has contributed what in the project.
- Write a document to explain the work.

IMPORTANT NOTES

The evaluation of the application will be conducted considering various aspects such as:

- Completeness of implementation
- Use of conventions in writing Python code
- Organization of the application in functions.
- Documentation through comments of the most significant portions of code.

Each project must be accompanied by a short report whose template will be found within the project associated with the GitHub repository.

It is highly advisable to use LaTeX to compose the report. This is a preparation step towards the master thesis, where students are expected to be capable of using LaTeX as well as the editing environment.

There are the following projects:

<u>Project 1:</u> Issue Report Classification

- Datasets: NLBSE'24 tool competition on issue report classification, 3,000 samples. https://nlbse2024.github.io/tools/.
- Type: Multi-label classification.
- Implementing Machine Learning or Deep Learning classifiers.
- Evaluating the classifiers using suitable metrics.
- Splitting data for k-fold cross validation technique.
- Selecting a suitable machine learning technique.
- Running the evaluation.
- Comparing with the dedicated baselines (based on Sentence Transformers).
- Analyzing the obtained results.

<u>Project 2:</u> Code Comment Classification

- Datasets: NLBSE'23 tool competition on issue report classification, 6,738 samples. https://nlbse2023.github.io/tools/.
- Type: Multi-label classification.
- Implementing Machine Learning or Deep Learning classifiers.
- Evaluating the classifiers using suitable metrics.
- Splitting data for k-fold cross validation technique.
- Selecting a suitable machine learning technique.
- Running the evaluation.
- Comparing with the dedicated baselines (Transformer-based RoBERTa).
- Analyzing the obtained results.

<u>Project 3:</u> Detection of rating based on TripAdvisor reviews.

- Datasets: TripAvisor hotel review https://www.kaggle.com/datasets/andrewmvd/trip-advisor-hotel-reviews.
- Implementing Machine Learning, Deep Learning classifiers.
- Evaluating the classifiers using suitable metrics.
- Splitting data for ten-fold cross validation technique.
- Selecting a suitable machine learning technique.
- Running the evaluation.
- Comparing the prediction performance.

- Analyzing the obtained results.

Project 4: Summarization of GitHub issues

- Dataset: https://www.kaggle.com/datasets/davidshinn/github-issues
- Running the evaluation.
- Analyzing the obtained results.
- Generating textual descriptions for commits, issues.
- Requirement: Summary a long text to yield a short/informative one.
- Techniques: Transformer, BERT.
- Text summarization: HuggingTransformer, T5.

Project 5: Summarization of Stack Overflow Posts

- Dataset: https://github.com/BonanKou/SOSum-A-Dataset-of-Extractive-Summaries-of-Stack-Overflow-Posts-and-labeling-tools
- Paper: SOSum: A Dataset of Stack Overflow Post Summaries, Automated Summarization of Stack Overflow Posts
- Running the evaluation.
- Analyzing the obtained results.
- Requirement: Summary a long text to yield a short/informative one.
- Techniques: Transformers, BERT, LLMs (Llama 2).

Project 6: Summarization of GitHub README.MD

- Dataset: https://github.com/MDEGroup/GitSum
- Paper: Too long; didn't read: Automatic summarization of GitHub README.MD with Transformers
- Running the evaluation.
- Analyzing the obtained results.
- Requirement: Summary a long text to yield a short/informative one.
- Technique: Transformer, BERT, LLMs.

Project 7: Code Summarization

- Dataset: https://github.com/wssun/LLM4CodeSummarization
- Paper: Source Code Summarization in the Era of Large Language Models
- Running the evaluation.
- Analyzing the obtained results.
- Requirement: Summary a snippet of code to yield a short sentence that can summarize the main intent of the code.