University of Rome, La Sapienza. Artificial Intelligence and Robotics.
Deep Learning Course
A.Y. 2023/24

In the following, we introduce four different projects *Pick one*!

Rules:

1. Each project must be completed in groups of up to three people. **Use Google Colab** (`https://colab.research.google.com/`) **to develop your solutions**. Code should be designed for GPU execution. **Important**: projects that do not use *pytorch lightning* or adhere to *PEP8 formatting rules* will not be accepted.

2. **A complete notebook must operate within Colab's runtime constraints**. Projects that do not comply with Colab's limitations will not be accepted.

3. Each project should be described in a report of **at most five pages** using this Overleaf's template. Reports longer than five pages will not be accepted. The report must explicitly state and discuss all the relevant design decisions.

4. Each notebook must automatically download all necessary datasets from a designated repository. *It is not permitted to pre-train a model on an external resource and download the parameters on the colab notebook.*

5. The code must execute and yield interpretable results. Merely reporting training validation or test errors is insufficient; you must employ relevant metrics aligned with the task objectives.

6. You are **not** permitted to "*copy*" code fragments from online sources. All solutions must be original. Copying entire models, training pipelines, or any elements central to the main project's code is strictly prohibited. If you decide not to abide by this rule and you incorporate a code fragment taken from an online source, you ***must*** cite the URL of the source in your code comments and justify why you had done so. Failure to do so will result in the project being considered invalid.

7. Copying project solutions from other students is, of course, considered cheating and is strictly forbidden.

8. Projects can be submitted anytime using this form, but the discussion date is fixed and will be scheduled after the predetermined exam date. Submissions after the last session of 2024 will not be accepted.

9. During the oral exam sessions, and before individuals are asked theory questions, each group presents the project in **at most ten minutes** using a deck of slides to be shown during the presentation.

10. **Do not send e-mails directly to the teacher or the teaching assistants** with questions about the project; use Google Classroom instead. Your doubt can be the doubt of your colleagues. The professor or the teaching assistants will answer to you there.

Evaluation (Max 31 pts.):

- **Minimum Project Requirements** (18 pts)
  - Projects will receive points if they successfully run on Colab, terminate correctly, and produce results. Results must be computed in a scientifically sound manner, ensuring training is not conducted on the test set. Specific paper implementations are not required; in fact original solutions are valued more. Code must be well-documented with appropriate comments.
  - Implement a non-trivial baseline method, which is an alternative approach for solving the task. This must be compared to facilitate discussion of your results. Note that a random baseline does not qualify as a valid baseline.
  - Hyperparameter tuning should be conducted systematically.
- **Code Quality** (+1 pt): Code should be modular and self-explanatory. It must include plots and tables for all results. Logging should cover all aspects of training and evaluation, including hyperparameter search details.
- **Project Paper and Presentation** (+3 pts): The presentation should be comprehensive and demonstrate an understanding of the developed techniques. All design choices must be clearly presented and discussed.
- **Oral Exam** (+9 pts): The oral exam will cover topics studied during the semester. Incorrect answers may result in a score lower than the total project score. In such cases, students may retain their project score and retake the oral exam, with a maximum of three attempts allowed.

Additional Rule on Project and Oral Exam Attempts. Students are permitted to use the same project for a maximum of two attempts at the oral exam. The grade awarded for the project will be retained for both attempts. However, if a student fails the oral exam (or refuses to accept the grade) twice while using the same project, they will be required to submit a new project, distinct from the first, for any subsequent attempts at the oral exam. This rule ensures that students have the opportunity to improve their performance while also upholding the standard of varied and original work for each exam attempt.

Neural inverted index for fast and effective information retrieval.

Information retrieval (IR) systems are designed to provide a ranked list of pertinent documents in response to user queries. Most contemporary information retrieval systems adopt the index-then-retrieve pipeline. Recently, an alternative approach called the Differentiable Search Index (DSI) has been proposed. Instead of segregating indexing and retrieval into two distinct components in an IR system, DSI aims to encompass all corpus information and execute retrieval within a single Transformer language model.

Task: The task is to build a model f that given a query q as input returns a ranked list of document ids. There should be a unified model trained to replicate the behavior of an index built on a corpus of documents and thereafter used to retrieve relevant documents. The proposed DSI should be different from the ones present in the literature.

Dataset: MS Marco. You can use the Pyserini library to work with it! Here you can find a colab notebook where you can see an example on how to use it.

Metrics:

- MAP: Mean Average Precision it's a crucial metric in the field of information retrieval. It's used to evaluate the effectiveness of search systems, like search engines or database queries. Here's a breakdown of what it means and how it's calculated:
  - Precision at K: Precision is a measure of relevancy. It's calculated as the number of relevant documents retrieved divided by the total number of documents retrieved. For example, if a search retrieves 10 documents and only 4 of them are relevant, the precision is 0.4.
  - Average Precision (AP): This is calculated for a single query. It's the average of the precision values calculated at the points in the ranking where each relevant document is retrieved. So, if there are several relevant documents, the precision is calculated each time one of these is encountered in the list of retrieved documents, and these values are averaged.
  - Mean Average Precision (MAP): This is the mean of the average precision scores for a set of queries. Essentially, you calculate the average precision for each query, and then find the mean of these values across all queries.
- Recall@1000: is the proportion of relevant items found in the top-1000 results.

References:

- https://paperswithcode.com/paper/transformer-memory-as-a-differentiable-search
- https://arxiv.org/pdf/2305.02073.pdf

Neural Network-Based Character-to-Symbol Sequence Translation for Text Compression

The objective of the project is to create a new neural network model to turn text into a format compatible with unzip software, making file extraction more efficient. By using deep learning, this model will connect human-readable text with machine-readable symbols, making it easy to integrate with existing unzip tools.

Task: Build a model that takes a text file as input and produces a zip file as output. For example, if you choose 7-Zip, the model should learn to generate output files similar to those produced by 7-Zip. For model evaluation, two metrics are required: the Levenshtein distance, which measures the similarity between the file generated by the model and the one produced by 7-Zip, and the 'unzip metric'. The 'unzip metric' is defined as follows: for each sample, assign a value of 1 if the chosen software successfully unzips the generated output file, and 0 otherwise.

Dataset: You can use any text dataset you like and create zip files for each text using your chosen software. The input sequence is the original text file the (expected) output sequence is the content of the zip file.

Metrics: Evaluate the model using two metrics: the Levenshtein distance, comparing the zip file produced by the software with the zip file generated by the model (you can open the zip file with a text editor) and the 'unzip metric.' The 'unzip metric' is defined as follows: for each sample, assign a value of 1 if the chosen software successfully unzips the generated output file, and 0 otherwise.

References:
While the task isn't identical, you can find inspiration from this paper: `https://paperswithcode.com/paper/llmzip-lossless-text-compression-using-large`

Graph Neural Networks for Next Point Of Interest Recommendation

A (Point of Interest) POI recommendation technique essentially exploits users' historical check-ins and other multi-modal information such as POI attributes and friendship network, to recommend the next set of POIs suitable for a user. A POI recommendation technique essentially exploits users' historical check-ins and other multimodal information to recommend the next set of POIs suitable for a user.

Task: To build a model that given a user's historical check-ins, along with POI attributes can predict and recommend the next set of POIs that would be suitable and interesting for the user. The output is a list or ranking of recommended POIs based on the user's preferences and historical behavior.
Here you can find a notebook on how to load the data.

Dataset: Foursquare NYC: Location-based social networks have attracted millions of users and massively contain their digital footprints. Some researchers crawled a part of these digital footprints from Foursquare in order to study the problems of personalized location recommendation and search. This dataset contains check-ins in NYC collected for about 10 months. For more information about the dataset look at Section 2 of the site.

Metrics:

- Accuracy@K: Acc@K $= \frac{1}{N_{\text{test}}} \# \text{hit} @ K$ where $\# \text{hit} @ K$ is the number of samples with the correct predictions made within the top K of the ranked set for $K \in \{1, 5, 10, 20 \ldots\}$, and N_{test} is the total;
- Mean Reciprocal Rank: $MRR = \frac{1}{N_{\text{test}}} \cdot \sum_{v=1}^{N_{\text{test}}} \frac{1}{\operatorname{rank}_v\left(l_{t_i}\right)}$, and $\operatorname{rank}_v\left(l_{t_i}\right)$ is the position of the ground truth next POI l_{t_i} in the predicted ranked set for each v-th test sample.

References:
https://dl.acm.org/doi/pdf/10.1145/3477495.3531989

Geometric Transformers *In recent years, two very important and popular research areas in Deep Learning have been:*

- *How can symmetries be encoded using equivariant deep learning?*
- *How can geometric objects be represented through algebraic structures?*

Task: This project aims to develop a Transformer-based architecture tailored for geometric data. The focus will be creating a model specifically designed to classify arteries exhibiting stenosis or bifurcation. To succeed, the model must identify essential features in the input data and represent the inputs, outputs, and hidden states within a projective geometric algebra framework. Additionally, to assess the model's ability to capture symmetries, the project must investigate symmetry group operations, mainly to test the model's equivariance.

Dataset: This dataset is designed for a binary classification task. It comprises two distinct classes:

- The first class includes idealized arteries with a single outlet and randomly located stenoses.
- The second class contains bifurcating arteries.

The dataset encompasses simulations of 2,000 single-outlet arteries and 2,000 bifurcating arteries. Additionally, it features a subset of 731 single arteries with pulsatile-flow simulations. The dataset is available for download at the following link.

Metrics: F1 Score

Refrences:
https://arxiv.org/abs/2305.18415