DAISY course "Big Data Engineering"

Project "Image Recommender"

Setting: small teams (2, max. 3 members)

Requirements to pass the module

- Working `ImageRecommender` Python software, provided as GitHub repository (private or public).
- Project documentation (as pdf)
- Project presentation (all team members, 15 minutes including short demo + 15 minutes discussion)
- You must have made clearly visible, extensive code contributions to the GitHub repository (make sure to use your own account for commits!).

Software:

Functional requirements

The final Python software should be able to perform the following actions:

- For a single input image: Find the 5 best matching images from the provided dataset (about 400,000 images) based on at least three different similarity measures. The search should take no longer than a few seconds.
- (Required for 85-100 point ranges)
  For up to 5 input images: Find the 5 best matches just as before, but now images that are ideally related to all 5 inputs.

Required building blocks

- **Python generator** to load images.
- Internal handling of images via unique image-IDs
- Use of **one or multiple databases**. Your project should contain at least one relational database to link image-ID, file name and location, as well as relevant metadata (relevant here means: relevant for your software).
- Optional: Create and store lower resolution version of each image.
- Own implementation of **3 similarity metrics** to quantify the similarity between images. Each one should be a Python function. Optimize each function as good as you can, for instance using faster libraries, numba compilation, or parallelization (dask).
  You can be creative and invent your own similarities measures, but you should include:
  - one measure that is based on **color profile**/similarity
  - one measure that used compressed image **embeddings** (using deep learning model)

    - one measure that you can choose freely (for instance: content-based via classifier, or classical dimensionality reduction)
- Include unit tests, at least for:
    - all image similarity functions
    - the data retrieval steps that use the database(s)
- Stick to best practices/clean code (PEP8, at least mostly + documentation).

Analysis + Documentation

Part 1 - Image recommender software

1. Motivation/Aim of the project
2. Program design (sketch + text)
   What are the main elements of your software? How do they interact? What key techniques are used (such as databases, libraries ...)
3. Image similarity
   How do you measure image similarity? Describe, explain and compare your methods. Discuss their advantages and disadvantages.
4. Performance analysis ("profiling") + runtime optimization
   What are the bottlenecks of your program? How long does it take to select similar images from the full dataset and which steps take the most time?
   Which bottlenecks did you detect and address, for instance through using runtime optimization or optimized code? Which optimization techniques did you use?
5. Feasibility analysis/Discussion
   Discuss, based on your performance analysis, how "scalable" your software is. Do the different image retrieval techniques work fast enough to use it as search engine? For which number (and size) of images is your software suitable? What are main limitations and why. And how could those be addressed?

Part 2 - Big data image analysis

1. Plot the position of all images in a 2D (or if you prefer: 3D) plot. The position should be computed such that similar images are close together and dissimilar images tend to be far away from each other.
2. For this we can use common dimensionality reduction algorithms. Try: UMAP, t-SNE and/or T-MAP
3. Since we have a huge number of images (= many data points + each with many feature dimensions), those algorithms will only work if they are used with highly reduced images embeddings **or** with pre-computed similarity values (see 2A point 1.).
4. Discussion. What can we learn from the plot(s)? Does the image position in 2D (or 3D) make sense?What worked, and what didn't (and why)?

Bringing structure to chaos...

Suggestion for project milestones:

1. Part - Data Exploration & Basic handling

- Data Exploration
- Dataset assembly
- Data Generator to read images
- Code to collect the main metadata.

2. Part - Similarity Measures (1)

- Create a color-based similarity measure.
  (Function + tests)

3. Part - Similarity Measures (2)

- Build a image embedding based similarity measure.
  (Function + tests)

4. Part - Data Pipeline and Software Design (1)

- Sketch the overall architecture of the software
- What database(s) to use?
- Implement >= 1 database (1 for metadata...)

5. Part - Data Pipeline and Software Design (2)

- Design an efficient image retrieval pipeline
- Sketch the full software design based on this pipeline.
- Tasks to consider:
    1. input image --> 5 most similar images
    2. Select five images from data --> recommend a good match from those