PRACTICE: Faiss
A library for efficient similarity search and clustering of dense vectors

I. Goals

- Use the **FAISS** library, to experiment with efficient similarity searching and clustering of dense vectors.
- Use traditional image feature extraction and common deep learning/machine learning models.

II. Introduction

- **Faiss** is a library for efficient similarity search and clustering of dense vectors. It contains algorithms that search in sets of vectors of any size, up to ones that possibly do not fit in RAM. It also contains supporting code for evaluation and parameter tuning. Faiss is written in C++ with complete wrappers for Python/NumPy. Some of the most useful algorithms are implemented on the GPU. It is developed primarily at Meta's Fundamental AI Research group.

- **Faiss** contains several methods for similarity search. It assumes that the instances are represented as vectors and are identified by an integer and that the vectors can be compared with L2 (Euclidean) distances or dot products. Vectors that are similar to a query vector are those that have the lowest L2 distance or the highest dot product with the query vector. It also supports cosine similarity, since this is a dot product on normalized vectors.

III. Content

1. Prepare the necessary programming environment.
   - **Python** programming language, minimum recommend version from **3.10**
   - Recommend using a **virtual environment** for developing
   https://www.jetbrains.com/help/pycharm/creating-virtual-environment.html
   https://code.visualstudio.com/docs/python/environments
   - The recommended way to install **Faiss** is through **conda**. Stable releases are pushed regularly to the pytorch conda channel, as well as pre-release nightly builds.
   The CPU-only faiss-cpu conda package is currently available on Linux, OSX, and Windows. The faiss-gpu, containing both CPU and GPU indices, is available on Linux systems, for various versions of CUDA.
   To install the latest stable release:

```
  # CPU-only version
```

```
$ conda install -c pytorch faiss-cpu


# GPU(+CPU) version
$ conda install -c pytorch faiss-gpu

# or for a specific CUDA version
$ conda install -c pytorch faiss-gpu cudatoolkit=10.2 # for CUDA
10.2
```

2. Tasks

   - 1. Pick at least one image dataset (*more than one dataset: bonus point*), and provide the dataset link in the report with its description.
   - 2. Implement at least 2 different image feature extraction, one traditional method and one deep learning approach, i.e. **SIFT**/**ORB** for the traditional and **ResNet**/**DenseNet** for the deep learning (*more than 2: bonus points*) for the chosen dataset.
   - 3. Combine with the Faiss library, experiment with at least 3 different distance metrics, including both primary **L2** and **inner product** (*from 4 distance metrics: bonus point*).
   - 4. Report the detailed experimental results, with comments or explanations.

IV. Requirements

1. The directory structure of the compressed submission

   - *doc*: report files include MSSV_report.doc and MSSV_report.pdf.
   - *source*: contains entire source code directory / Jupyter Notebook, removed temporary files, intermediate compiled files if exist…

   Note: no report: consider 0 (zero) point.
2. Other requirements

   - The report should be presented clearly and intuitively: summarize what had finished (task 2.1-2.4 with percentage), list the features included in the program with proof images, summarize the usage and implementation (for example: through pseudo-code, description of methods, or how to do it, *do not copy the source code into the report*).
   - The source code needs to be commented on the corresponding lines.

   Note: all cases of copying or even of being copied could be considered 0 (zero) point.