UNIVERSITY OF INFORMATION TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE

FINAL REPORT

MULIMEDIA INFORMATION RETRIEVAL

Image Retrieval

Ho Chi Minh City, 01/2025

Acknowledgement

Our group would like to express our heartfelt gratitude to **PhD. Ngo Duc Thanh**, a lecturer at the Department of Computer Science at the University of Information Technology, Vietnam National University - Ho Chi Minh City. As the instructor for the **CS336.P11.KHTN - Multimedia Information Retrieval**, his dedicated teaching and insightful feedback have been invaluable to the success of our project.

Throughout the process of completing this project, we strived to deliver the best possible results. However, despite our efforts, some unintentional errors may have occurred. We respectfully request candid feedback and constructive suggestions from the professor to help us further improve our work.

For any questions or feedback, feel free to contact us by email at the following addresses:

- 22521333@gm.uit.edu.vn (Nguyen Duy Thang)

We greatly appreciate your time and guidance!

No	**Fullname**	**Student ID**	**Percent**
1	Nguyen Duy Thang	22521333	100%

Table 1: Team members

Contents

1 Problem Statement

Problem Identification: Image retrieval is an important area of computer vision that focuses on finding relevant images from a database based on a query. This query can be an image, text, or other information. The goal of the problem is to find the images in the database that are most similar or relevant to the information need according to some predefined criteria.

Input:

- Query: An image or a text that represents information need.
- Database: A collection of images.

Output: A set of images that is relevant to information need.

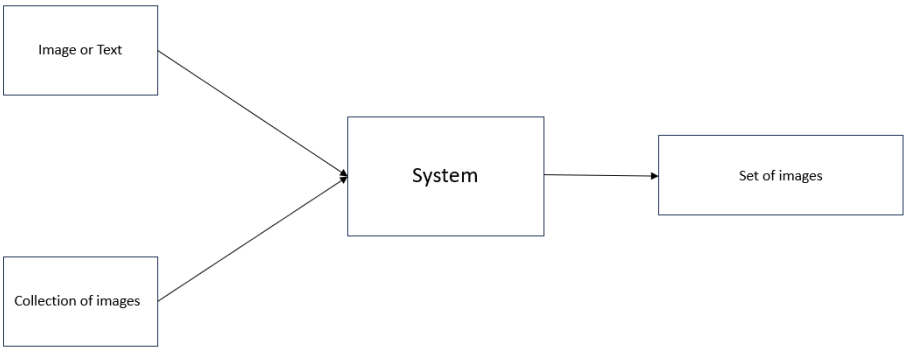

Figure 1: Problem Statement

2 Solution

2.1 Data Processing

Data Transformation: Since the images in the database may have different sizes and overall colors, this may cause the feature vectors of images with the same content but different elements to be different. So we will transform the images in the database as follows:

```
transform = Compose([Resize((224, 224)),
                     ToTensor(),
                     Normalize((0.48145466, 0.4578275, 0.40821073), (0.26862954, 0.26130258, 0.27577711))])
```

Feature Extraction: Since the Vector Space Model (VSM) represents both queries and data as feature vectors, we will convert the images in the database into vector representations. However, because the problem involves text-based queries, these feature vectors must establish a connection between text and images. To achieve this, we have selected the CLIP Radford et al. [2021] for feature extraction.

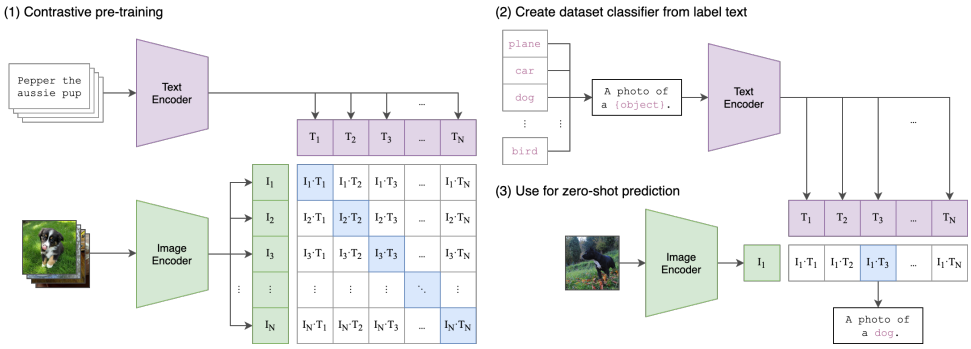

Figure 2: Contrastive Language-Image Pre-Training (CLIP)

- For images, the image is passed through an Image Encoder (ResNet or Vision Transformer) to extract the feature vector I.
- For text, the text is passed through a Text Encoder (CBOW or Text Transformer) to extract the feature vector T.
- Both sets of vectors are mapped into the same space through their respective matrices W, and then normalized.
- The dot product between the vectors I and T is calculated (this is the cosine similarity because the vectors have been normalized).
- The loss is computed and trained such that the values on the diagonal are large, and the off-diagonal values are small.

The reason we choose CLIP is that it is a powerful model which uses both language and images to understand and represent information and easy to implement. However, CLIP is pre-trained on its own dataset so it "may" not work well with other data, we will retrain the model to adapt it to specific dataset and use it alongside the original version for evaluation.

2.2 Index Construction

To achieve high performance querying with large databases, we will build different types of indexes. These types of indexes are implemented in FAISS Douze et al. [2024] - a library for efficient similarity search and clustering of dense vectors:

IndexFlatIP:

- Inner Product (IP) is a mathematical calculation that measures the similarity between two vectors:

$$similarity(a, b) = a \cdot b = \sum_{i=1}^{N} a_i \cdot b_i$$

- This method is suitable for vectors that have been normalized by the L2 norm, since in this case the inner product is equivalent to cosine similarity.
- Flat means that all data is stored in memory without any encryption or compression techniques applied. This index uses brute-force search, i.e. it calculates similarity to all vectors in the database, ensuring absolutely correct results.
- It is one of the most basic indexes in FAISS. It does not require complex configuration and is suitable for small to medium data sets.

IndexPQ:

- Product Quantization Jégou et al. [2011] divides each vector into multiple segments (sub-vectors) and then encodes each of these segments using a small dictionary (codebook).
- Instead of storing the entire vector, it only stores the quantization codes of each segment.
- It reduces storage space by storing only the numeric codes instead of the entire vector and speeds up searching by using lookup tables instead of direct calculations between the query vector and the vector in the database.

2.3 ReRanking

After having indexes and executing queries using FAISS, we will get rank-lists. However, to get better results, re-ranking is very important. Because query refomulation methods are quite difficult, in this project, we use Pseudo/Blind Feedback with the top 3 first returned results being considered relevant.

For each query, after the above step, there will be many different rank lists. To combine them into a single rank list, we use Fusion algorithms in Ranx Bassani and Romelli [2022] - a library of fast ranking evaluation metrics implemented in Python, leveraging Numba for high-speed vector operations and automatic parallelization. We will use 2 different types of methods, in which the criteria are easy to develop and effective.:

- ComMNZ Fox and Shaw [1993] - Score-based Methods: This method prioritizes models and results with non-zero scores, to avoid combining unreliable results (e.g., items with a score of 0 that may not be relevant to the query or are items for which the model did not find a match). ComMNZ will sum the non-zero scores from all rank lists and use the number of results with non-zero scores to calculate the overall score. This reduces the influence of models that produce unhelpful results.
- Reciprocal Rank Fusion Cormack et al. [2009] - Rank-based Methods: Instead of combining the rank lists directly, RRF reverses the rank lists of each model. This can help reduce the influence of models that have incorrect discrimination in their rank orders. After the rank orders are reversed, these orders are combined using a combination method.

3 Experiment

3.1 Metrics

Because it is a query problem and there are many queries, the metric used in this problem will be mAP (the dataset used will be evaluated in binary-relevant form).

AP is the average of Precision at different Recall levels, providing a measure of the overall performance of a query. MAP is the average of AP on many different queries. MAP helps to evaluate the performance of the system over the entire query set. The higher the MAP, the better the system performs.

$$AP = \frac{1}{N}.\sum_{k=1}^{N} P(k).rel(k)$$

$$mAP = \frac{1}{Q}.\sum_{i=1}^{Q} AP_i$$

With:

- N is the number of results returned with recall = 1.
- P(k) is the Precision at position k.
- rel(k) is a function that indicates whether the k-th item is a relevant item or not. 1 is relevant, 0 is irrelavant.
- Q is number of queries.

3.2 Dataset

The dataset used is the Paris Buildings Dataset (cite). This dataset consists of 6412 images collected from Flickr by searching for specific landmarks of Paris. However, the data is currently not available at the original link, so it will be searched from another source and use the ground-truth file at the original link.

- The dataset have "Note, 20 of the supplied images are corrupted but are still included for backwards compatibility ". So, we have 6392 available images.
- There are 50 queries in total stored in _query.txt files including file name and bounding box to represent the area to query.
- The relevant files corresponding to each query are saved in the form _ok.txt and _good.txt

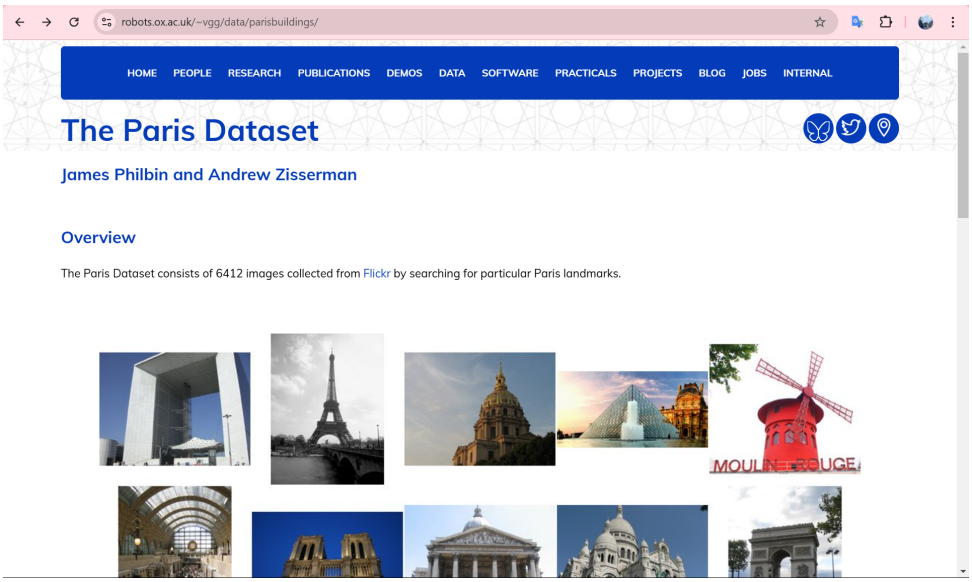

Figure 3: Paris Building Dataset

3.3 Results and Analysis

Compare CLIP and Retrain CLIP

- The CLIP retraining process is detailed in the paris_clip_training.ipynb file in the source code.
- Brute Force: We compare the query feature vector with each database feature vector based on cosine similarity.
- Index... : We use the index presented above to retrieval.

	CLIP	**Retrain CLIP**
Brute Force	0.6377	0.7721
IndexFlatIP	0.6375	0.7721
IndexPQ	0.6904	0.7869

Table 2: Compare CLIP and Retrain CLIP (mAP)

As the above results show, the system using retrained feature vectors from CLIP gives much better mAP results than feature vectors taken directly from CLIP. However, the above result is the result that has not been re-ranked. After being re-ranked 5-times with Retrain CLIP, the result is as follows:

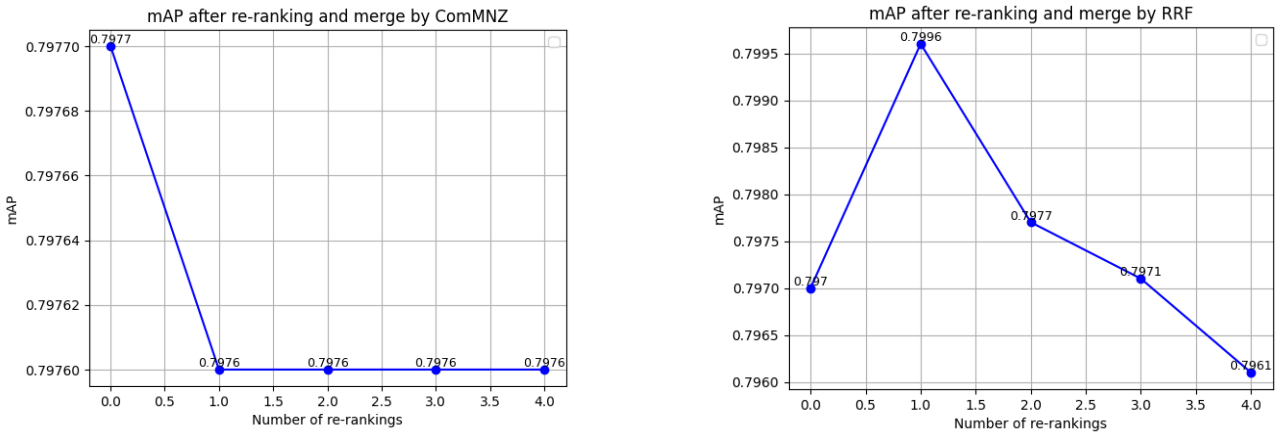

Figure 4: Brute Force after 5-times re-ranking

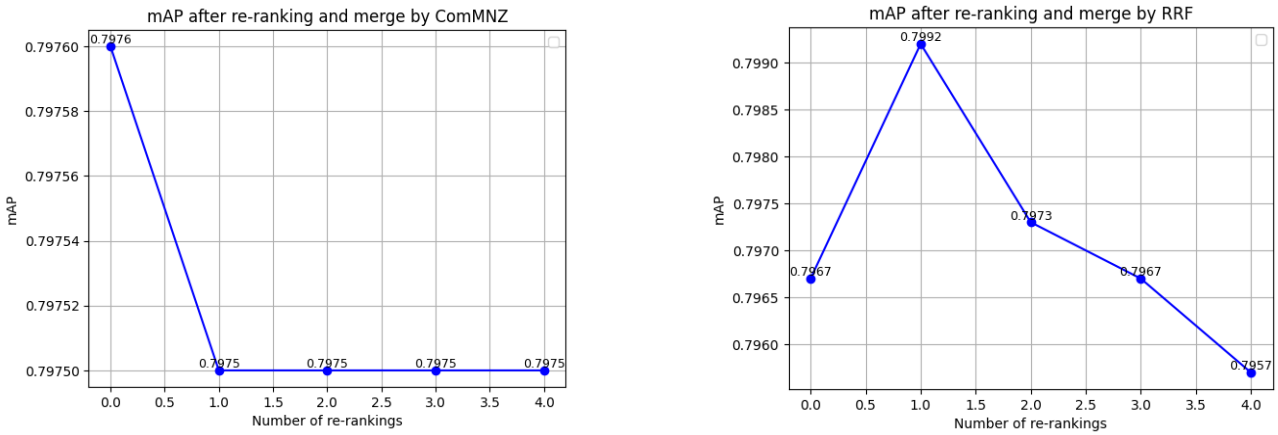

Figure 5: IndexFlatIP after 5-times re-ranking

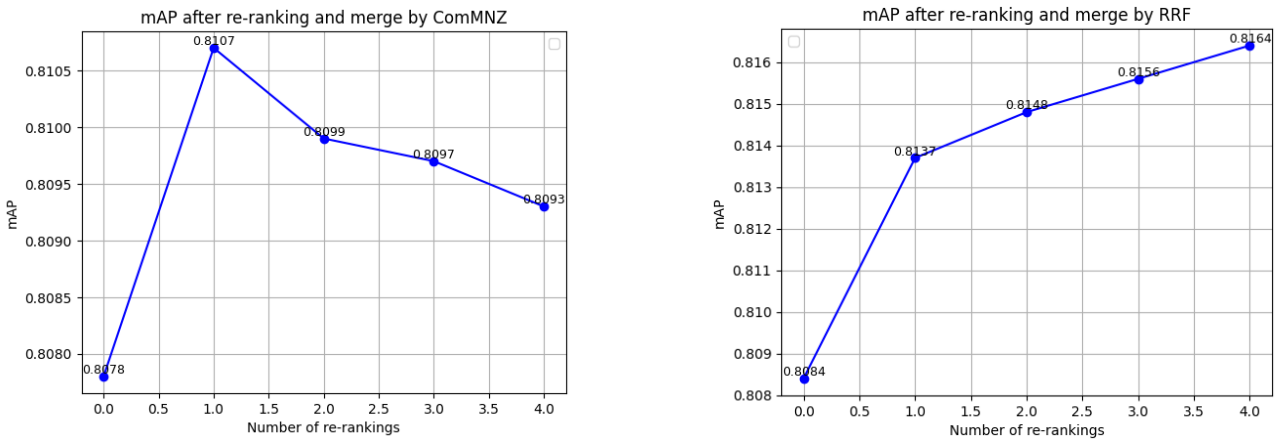

Figure 6: IndexPQ after 5-times re-ranking

In theory, IndexFlatIP should work like Brute Force so these two methods give almost the same result, but there is still a slight difference. As the result above, after re-ranking, mAP became larger, however, after many re-rankings, the result did not increase too much, even decreased a little. And the way to merge the results with these 2 different methods did not give too much difference.

Compared with current methods, this method is still not really effective and needs a lot of improvement.

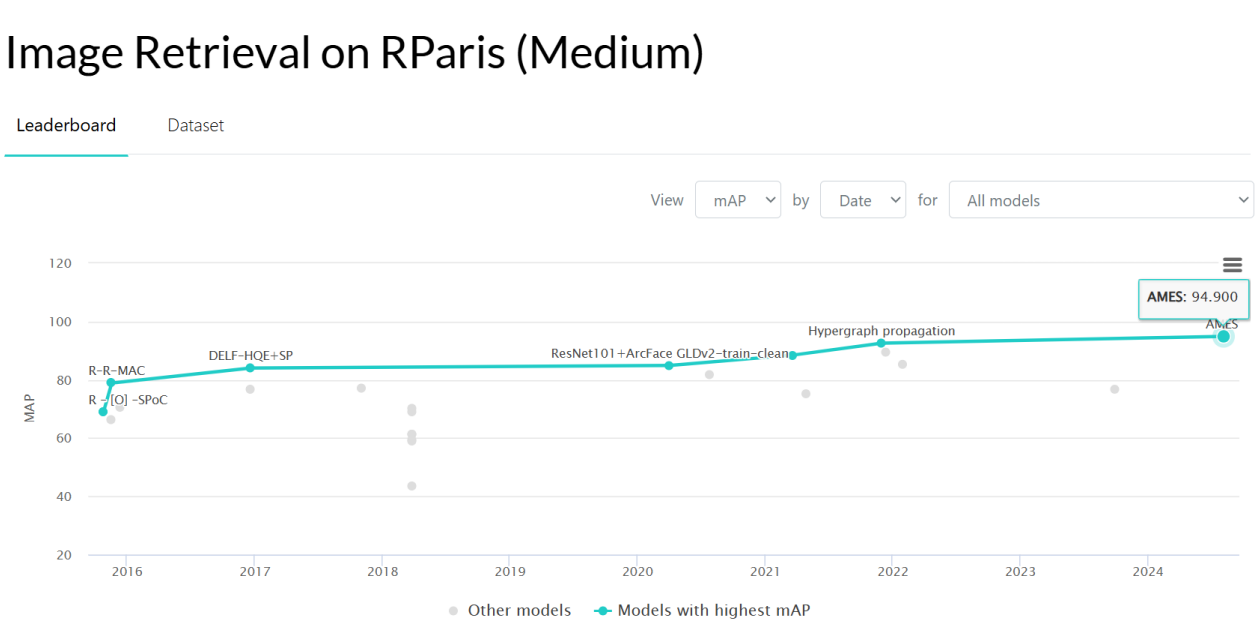

Figure 7: Current model

Summary: Advantages of this method:

- Easy to implement because there are supporting libraries.
- The related knowledge is quite easy to understand.

Disadvantages of this method:

- There are many other types of deep features that are newer and can perform better than CLIP.
- The system responds quickly but when re-ranking by multiple images it takes a lot of time.
- There are many other types of indexes that have not been tried yet.
- There are many methods to combine results but we have not been implemented yet.
- Re-ranking can still give good results if done longer than 5 times.

3.4 Demo

The demo system has 2 tabs for 2 separate queries: Image and Text.

- Image Screen allows to enter top k returned results and upload images to query.
- Text Screen allows entering top k returned results and descriptions for the images to be queried (Vietnamese is supported, however, there may be not right so it is recommended to use English).

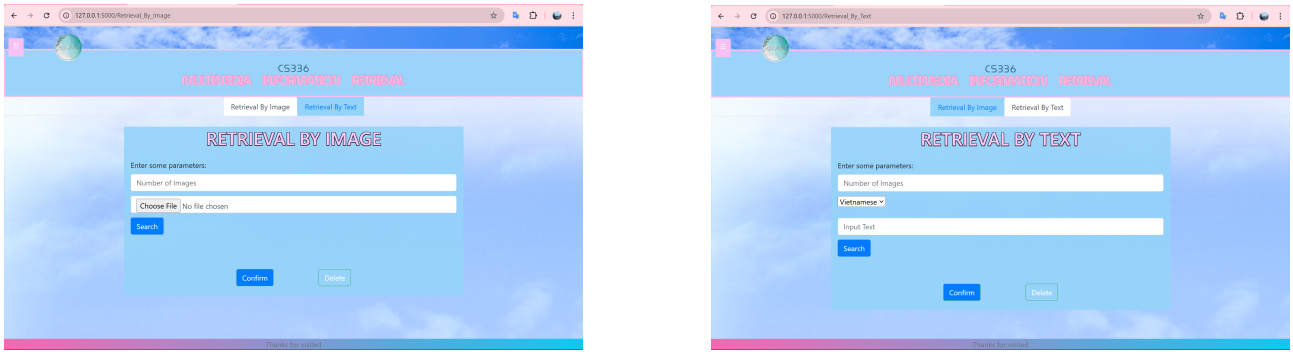

Figure 8: Retrieval Screen

After filling in the input and pressing Search, the system will operate and return a list of results as requested (if the query is an image, the image will be displayed for easy watching).

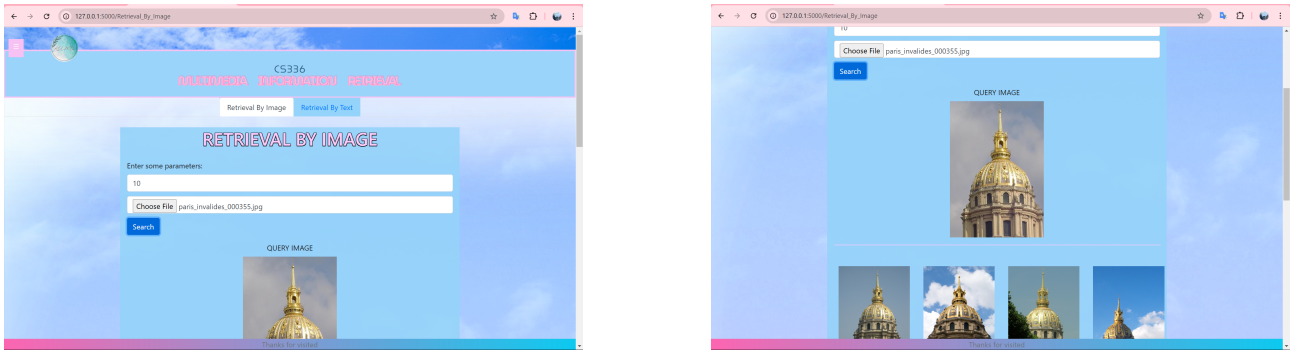

Figure 9: System Working

Below each returned image there will be a check box to select. When you click the Confirm button, the system will use the selected images to re-rank.

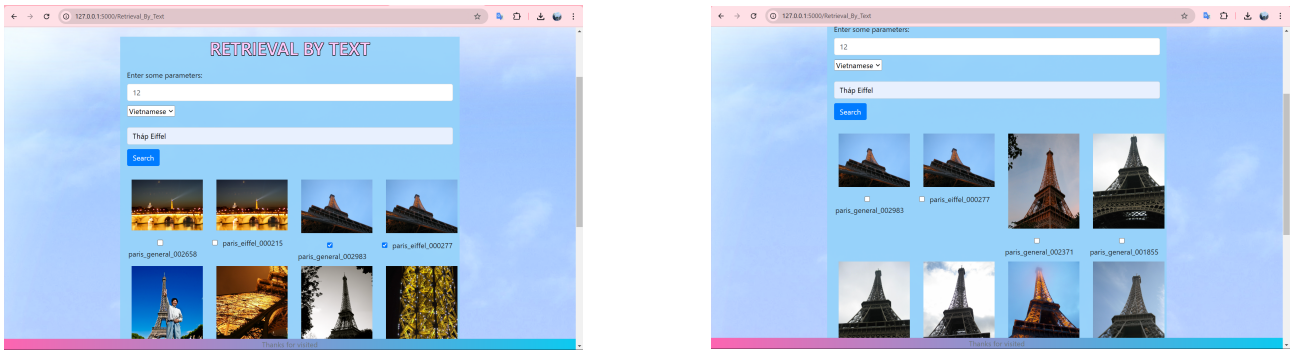

Figure 10: Before Re Rank (left) and After Re Rank (right)

After re-ranking the photos from bottom to top, the results have changed to mostly photos from the same angle. Source can be possible in: Github.

References

Elias Bassani and Luca Romelli. ranx.fuse: A python library for metasearch. In *CIKM*, pages 4808–4812. ACM, 2022. doi: 10.1145/3511808.3557207.

Gordon V. Cormack, Charles L. A. Clarke, and Stefan Büttcher. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *SIGIR*, pages 758–759. ACM, 2009.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. The faiss library. 2024.

Edward A. Fox and Joseph A. Shaw. Combination of multiple searches. In *TREC*, volume 500-215 of *NIST Special Publication*, pages 243–252. National Institute of Standards and Technology (NIST), 1993.

Herve Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128, 2011. doi: 10.1109/TPAMI.2010.57.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR, 2021. URL http://proceedings.mlr.press/v139/radford21a.html.