VietNam National University Ho Chi Minh City

University of Information Technology - VNUHCM

-----------📖-----------

INFORMATION RETRIEVAL CLASS PROJECT

TOPIC: CONTENT-BASED IMAGE RETRIEVAL

Lecture Instructor:	PhD. Ngo Duc Thanh	
Team members:	1. Phan Nhat Minh	19520166
	2. Pham Viet Tai	19522155
	3. Nguyen Minh Phu	19520218
	4. Nguyen Van Chinh	19521287

Ho Chi Minh, January, 2022

1. Introduction **4**

2. Content-based image retrieval system **4**

2.1. Frontend 4

2.2. Backend 5

2.2.1. Dataset 5

2.2.2. Algorithms & Methods Involved 5

2.2.3. CBIR search Pipelines 6

2.2.4. Description of each step: 6

3. Results **9**

4. Comparison **12**

5. Conclusion **13**

Abstract

In this project, we will represent the technologies applied in our project. The procedure of building a user-friendly interface with Streamlit to get and respond with the search results to requests from web pages. The operation of the content-based image retrieval method for searching images in the dataset - the Paris6K dataset. First, image features are extracted from the dataset with Image Descriptors using BRISK and RootSIFT, then they are stored and implemented redis for inverted indexing. The model accuracy is evaluated by TF-IDF.

1. Introduction

In modern days, with the excessively fast development of technologies, humans crave for applications and tools that are simple and convenient. And what people tend to do everyday is apparently searching, they search for news, addresses or documents. All they have to do is go to a web search, type some words and get all the information about what they want to know, this is the classic text search. However this is not enough, what if they have an image and want to understand what is in it: whose name is this celebrity? Where is this scenery? Thanks to modern technology, this task, which is known as visual search, has become within easy reach. Visual search is a new way of searching for information through the Internet. Users upload images to a web search and wait for the search engine to return the results, usually in forms of metadata.

Computer Vision represents a field of study that derives information from visual data such as images, videos or even metadata. Computer Vision plays a major role in the development of visual search as to optimize information acquired from the solely image that is put into the query (rather than metadata with textual information tagged along with the image). Thus, applied with extraordinary Computer Vision techniques, Content-based Image Retrieval method is created with the concept relying solely on the contents of the image.

2. Content-based image retrieval system

2.1. Frontend

The webpage is built by using Streamlit - a python library. By using Streamlit, we can easily deploy our model, showing results. We are using Collaboratory as a server and using localtunnel to have a link to access the website. Additionally, we use a plugin called Streamlit-cropper to crop pictures for querying part of the image.

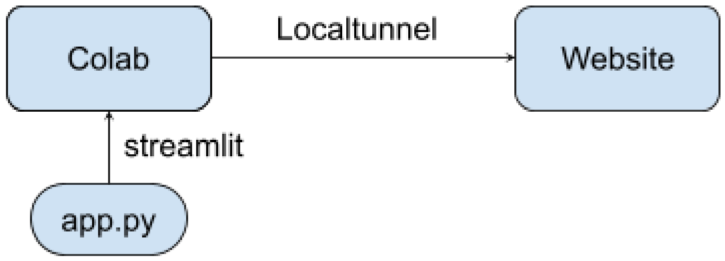
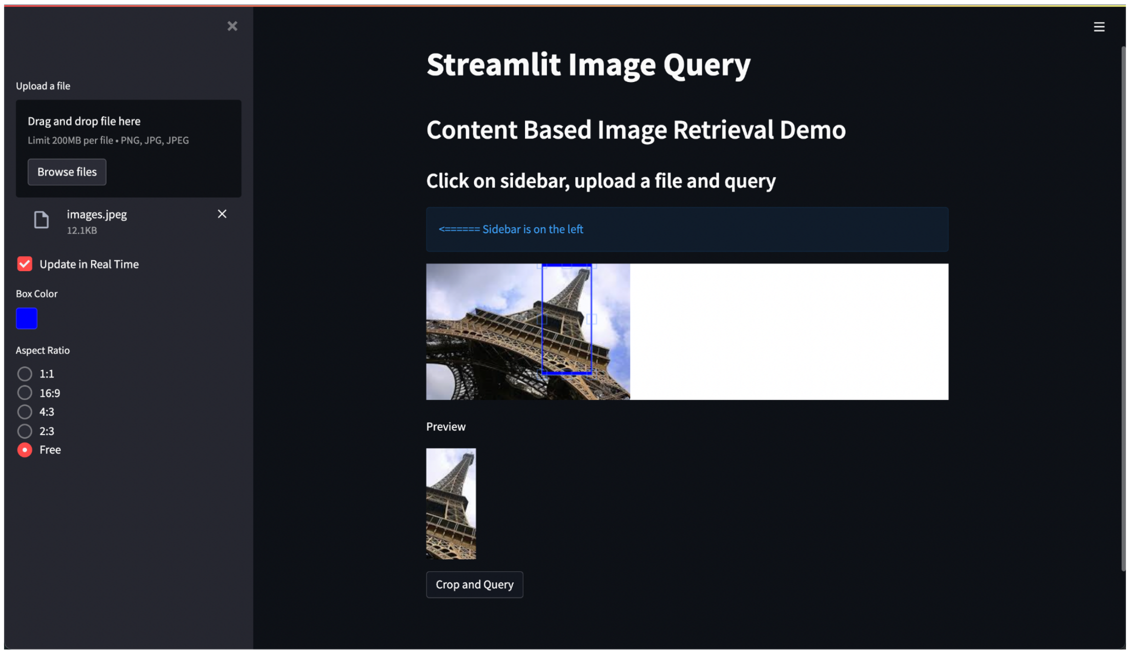


Figure 1. Flowchart of the website



Streamlit Image Query

Content Based Image Retrieval Demo

Click on sidebar, upload a file and query

<======= Sidebar is on the left

[✓ Update in Real Time]

Aspect Ratio: 1:1 16:9 4:3 2:3 [✓ Free]

Figure 2. Website interface

For re-using the project, we have built an end-to-end notebook file (ipynb) and also include the code for running it.

2.2. Backend

2.2.1. Dataset

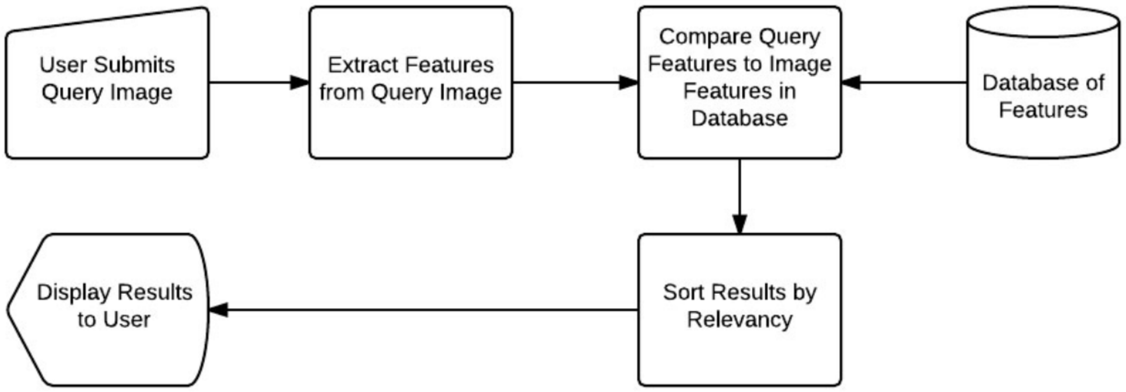
Dealing with this problem includes: to implement the tf-idf, RANSAC algorithm to evaluate accuracy of models, the dataset must be large enough. Thus we chose a well-known dataset which is ordinary in systems created to deal with the CBIR task. The Paris6k Dataset consists of 6412 images collected from Flickr by searching for particular Paris landmarks.

2.2.2. Algorithms & Methods Involved

- Keypoints and descriptors extraction
    - Binary Robust Invariant Scalable Keypoints detector algorithms (BRISK)
    - Local scale-invariant feature descriptors (RootSIFT)
- Feature storage and indexing
    - Structure HDF5 dataset
- Clustering features to generate a codebook
    - K-means algorithms
- Visualizing codeword entries (centroids of clustered features)
- Vector quantization
    - BOVW extraction

    - BOVW storage and indexing
- Inverted indexing
    - Implement redis for inverted index
- Search performing
    - Term frequency-inverse document frequency (tf-idf)
    - Cosine distance
- System accuracy evaluation
    - "Points-based" metric

2.2.3. CBIR search Pipelines


User Submits Query Image
Extract Features from Query Image
Compare Query Features to Image Features in Database
Database of Features
Sort Results by Relevancy
Display Results to User

2.2.4. Description of each step:

- Step 1: Extract keypoints and descriptors.

Every image in the dataset is put through a BRISK Feature Detector which detects corner-point-like keypoints in the image (Figure 2). After that, the keypoints are extracted through a RootSIFT Feature Descriptor which is a widely used technique for feature extraction. The features are stored in h5py files for ease of access and calculation.

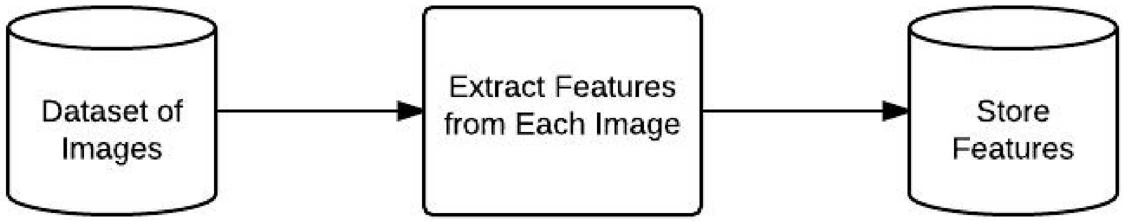


Figure 3. Extracting features Description

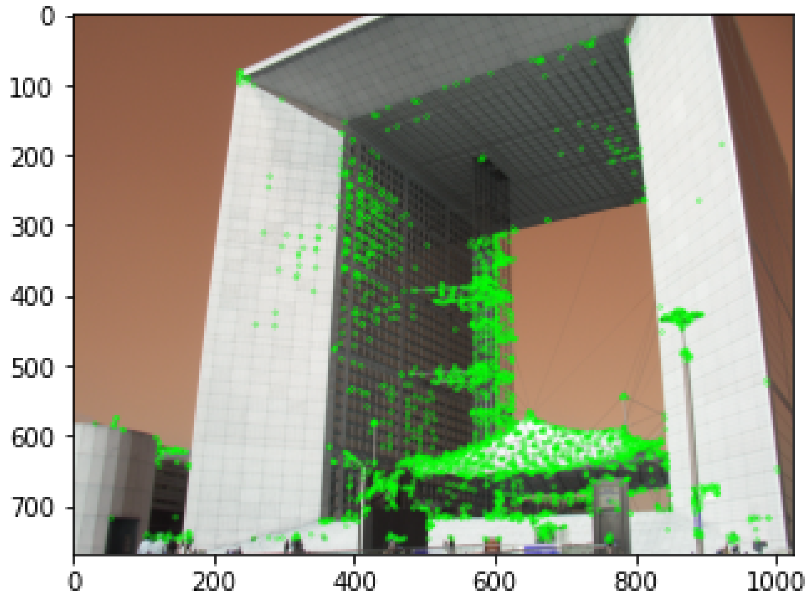


Figure 4. an example of BRISK keypoint detecting

- Step 2: Cluster features

  After the phase of feature extraction, the next step is to cluster extracted feature vectors to form "vocabulary" to provide results for the succeeding step, the MiniBatchK-means clustering is applied, with batch size taken of 25% percent dataset size. However, the elbow method is run before that to search for the most likely appropriate number of clusters (Figure 3). As the figure indicates, the number of clusters chosen is 3000. And also, the reason MiniBatchK-means is used instead of standard K-means is because MiniBatchK-means essentially works by sorting and breaking the dataset into smaller segments, then merging them together to form the final solution, this method results in tremendously fast performance in comparison with the standard K-means. Some visual words may encode for corner regions. Other visual words may represent edges. Even other visual words symbolize areas of low texture.

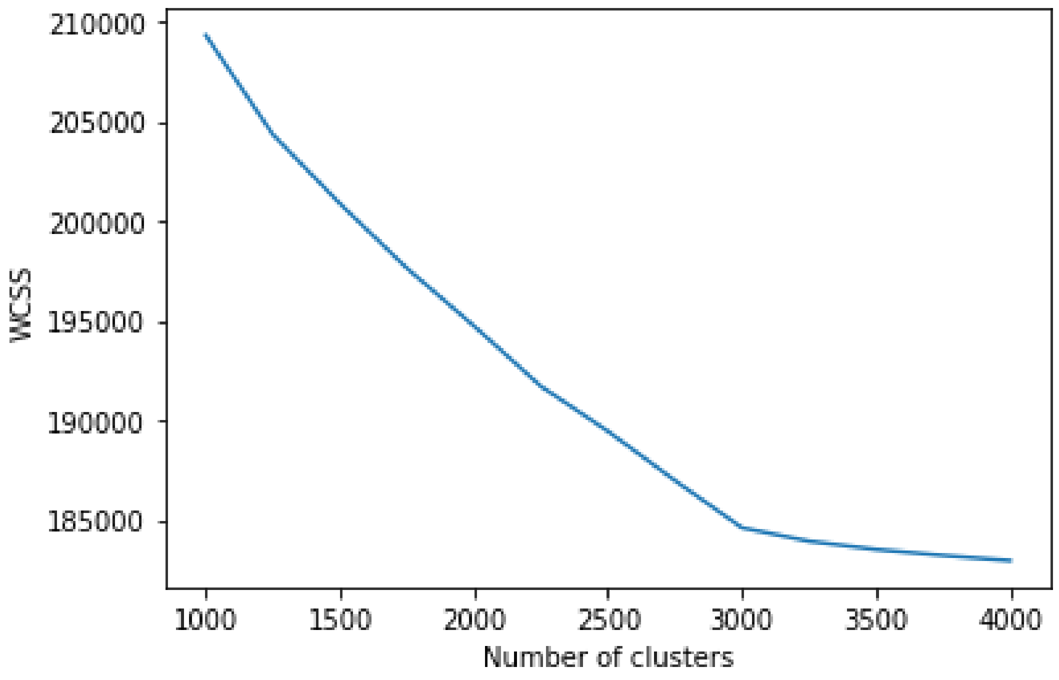


Figure 5. Elbow method with WCSS for features data

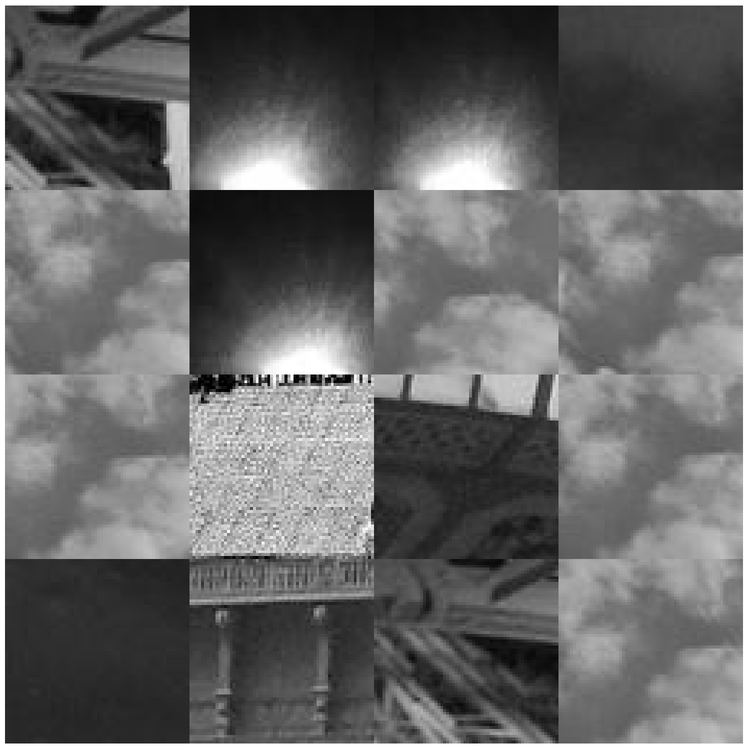


Figure 6. A visualization of a cluster feature

- Step 3: Vector quantization

  The vector quantization is used to utilize visual vocabulary (named codebook) to reduce this multi-feature representation down to a single feature vector each representing a whole image that is more suitable for CBIR and image classification. The clusters from the previous step is used to compute

- Step 4: Inverted indexing

  Inverted indexing is a technique in which the inverted index stores mappings of unique word IDs to the document IDs that the words occur in. Query image will be extracted into visual words in the form of a vector of BOVW histogram. Redis is also implemented to boost search speed.

- Step 5: Search performance

  Term frequency-inverse document frequency or TF-IDF, is used to reflect how important a word or document is in a collection/corpus. The tf-idf is applied to boost accuracy of the search. We use cosine distance to evaluate the similarity between the query image and the images in the dataset.

3. Results

Queries used for evaluation are the Groundtruth files of Paris6k itself provided in https://www.robots.ox.ac.uk/~vgg/data/parisbuildings/paris_120310.tgz by the dataset's owner, there are 55 queries in total. The ground-truth data is split into 3 types:

- Good - A nice, clear picture of the object/building.
- OK - More than 25% of the object is clearly visible.
- Junk - Less than 25% of the object is visible, or there are very high levels of occlusion or distortion.

For each query, the system will output 100 result images and those results will be calculated using the provided ground-truth to get the average precision.

Query name	Image Name	Average Precision
defense_1_query	paris_defense_000605	0.06
defense_2_query	paris_defense_000331	0.14

defense_3_query	paris_defense_000216	0.32
defense_4_query	paris_defense_000056	0.09
defense_5_query	paris_defense_000254	0.15
eiffel_1_query	paris_general_002985	0.71
eiffel_2_query	paris_general_001729	0.86
eiffel_3_query	paris_eiffel_000266	0.6
eiffel_4_query	paris_general_002645	0.75
eiffel_5_query	paris_general_002391	0.8
invalides_1_query	paris_invalides_000355	0.26
invalides_2_query	paris_invalides_000072	0.37
invalides_3_query	paris_invalides_000490	0.36
invalides_4_query	paris_invalides_000229	0.36
invalides_5_query	paris_invalides_000360	0.3
louvre_1_query	paris_louvre_000081	0.24
louvre_2_query	paris_louvre_000135	0.14
louvre_3_query	paris_louvre_000050	0.11
louvre_4_query	paris_louvre_000035	0.02
louvre_5_query	paris_louvre_000139	0.0
moulinrouge_1_query	paris_moulinrouge_000667	0.55
moulinrouge_2_query	paris_moulinrouge_000868	0.77
moulinrouge_3_query	paris_moulinrouge_000657	0.44
moulinrouge_4_query	paris_moulinrouge_000794	0.74
moulinrouge_5_query	paris_moulinrouge_000004	0.41
museedorsay_1_query	paris_museedorsay_000527	0.25

museedorsay_2_query	paris_museedorsay_000012	0.19
museedorsay_3_query	paris_museedorsay_000897	0.32
museedorsay_4_query	paris_museedorsay_000564	0.19
museedorsay_5_query	paris_museedorsay_000878	0.33
notredame_1_query	paris_notredame_000256	0.74
notredame_2_query	paris_notredame_000965	0.65
notredame_3_query	paris_notredame_000390	0.44
notredame_4_query	paris_general_003117	0.81
notredame_5_query	paris_notredame_000581	0.79
pantheon_1_query	paris_pantheon_000466	0.66
pantheon_2_query	paris_pantheon_000520	0.48
pantheon_3_query	paris_pantheon_000232	0.42
pantheon_4_query	paris_pantheon_000547	0.53
pantheon_5_query	paris_pantheon_000339	0.75
pompidou_1_query	paris_pompidou_000432	0.28
pompidou_2_query	paris_pompidou_000444	0.19
pompidou_3_query	paris_pompidou_000252	0.42
pompidou_4_query	paris_pompidou_000471	0.56
pompidou_5_query	paris_pompidou_000636	0.51
sacrecoeur_1_query	paris_sacrecoeur_000162	0.18
sacrecoeur_2_query	paris_sacrecoeur_000417	0.41
sacrecoeur_3_query	paris_sacrecoeur_000237	0.33
sacrecoeur_4_query	paris_sacrecoeur_000586	0.36
sacrecoeur_5_query	paris_sacrecoeur_000437	0.37

triomphe_1_query	paris_triomphe_000369	0.75
triomphe_2_query	paris_triomphe_000016	0.37
triomphe_3_query	paris_triomphe_000135	0.84
triomphe_4_query	paris_triomphe_000149	0.38
triomphe_5_query	paris_defense_000038	0.7

Based on the results of average precision of 55 queries, we get the value of mAP is 43%

4. Comparison

In recent years, with the groundbreaking birth of CNN in Computer Vision applying in image retrieval and gradually achieving state-of-the-art.

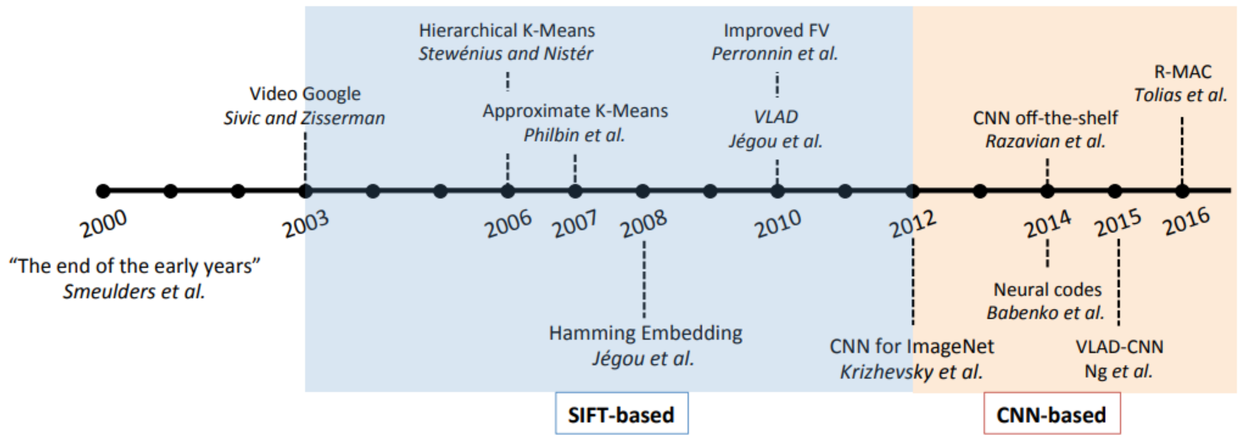


Figure 7. Milestones in detect-and-extract features techniques

In this report, we suggest a few SIFT-based methods to compare with. In other word, we choose some model that close with ours model to see the superiority as well as fairness with our model:

- SIFT-FV and SIFT-VLAD: The Scale Invariant Feature Transform (SIFT) [2] features are extracted and then aggregate by Fisher Vector (FV) [3] and Vector of Locally Aggregated Descriptors (VLAD) [4].

Method	mAP
SIFT-FV [7]	36.91
SIFT-VLAD [7]	41.49
Ours	43

All results are evaluated on the Paris 6k dataset.

In comparison with improved SIFT-based techniques, our model shows a significantly better of 2% in performance. We believe the better results are derived from RootSIFT descriptor and BRISK keypoint detector as proved in the previous article [5], [6]

5. Conclusion

In this project, we build an information retrieval system using the Paris6k dataset with bag-of-visual-word method, in which we use BRISK detector and RootSift descriptor for feature extraction. Besides, we use Redis server to speed up the query process, making it suitable for querying in a large dataset like Paris6k. In general, our method gives quite good results compared to other bag-of-visual-word methods. However, this is a fairly old method, so the accuracy may not be equal to modern methods using CNN or some other network architecture. In future, we intend to improve our model by implementing modern models as they are potentially effective in enhancing performance of the information retrieval system.

Reference

[1] [The complete guide to building an image search engine with Python and OpenCV] (https://www.pyimagesearch.com/2014/12/01/complete-guide-building-image-search-engine-python-opencv/)

[2] "Distinctive image features from scale-invariant keypoints." Lowe, D. G. *IJCV 2004.*

[3] "Large-scale image retrieval with compressed fisher vectors." Perronnin, F., Liu, Y., Sánchez, J., & Poirier, H. In *CVPR 2010.*

[4] "Aggregating local descriptors into a compact image representation." Jégou, H., Douze, M., Schmid, C., & Pérez, P. In *CVPR 2010.*

[5] Relja Arandjelovic and Andrew Zisserman. Three things everyone should know to improve object retrieval. In CVPR, 2012

[6] Tareen, Shaharyar Ahmed Khan & Saleem, Zahra. (2018). A comparative analysis of SIFT, SURF, KAZE, AKAZE, ORB, and BRISK. 10.1109/ICOMET.2018.8346440.

[7] Towards Visual Feature Translation." Hu, J., Ji, R., Liu, H., Zhang, S., Deng, C., & Tian, Q. In CVPR 2019

Assignment Table

ID	Name	Work	Progress
19520166	Phan Nhat Minh	- Evaluation - Comparison - Write report	100%
19522155	Pham Viet Tai	- Building back-end - Tuning parameter for clustering - Write report	100%
19520218	Nguyen Minh Phu	- Building front-end - Visualization - Write report	100%
19521287	Nguyen Van Chinh	- Building back-end - Statistics results of queries - Write report	100%