핵심은 feature를 잘 뽑아내는 게 중요

task 1은 당일 수업 끝날때까지 내야함
task 2는 수요일(14)일까지 제출

Image Matching Challenge

Longbin Jin

Artificial Inteligence
& Computer Vision
Laboratory

포털상에서는 아직 이미지 검색이 안되고 있음, 영상은 영상 자체의 컨텐츠를 키워드해서 검색 되고 있음

Recap: Image Matching System

우리가 할 일 :

indexing 작업

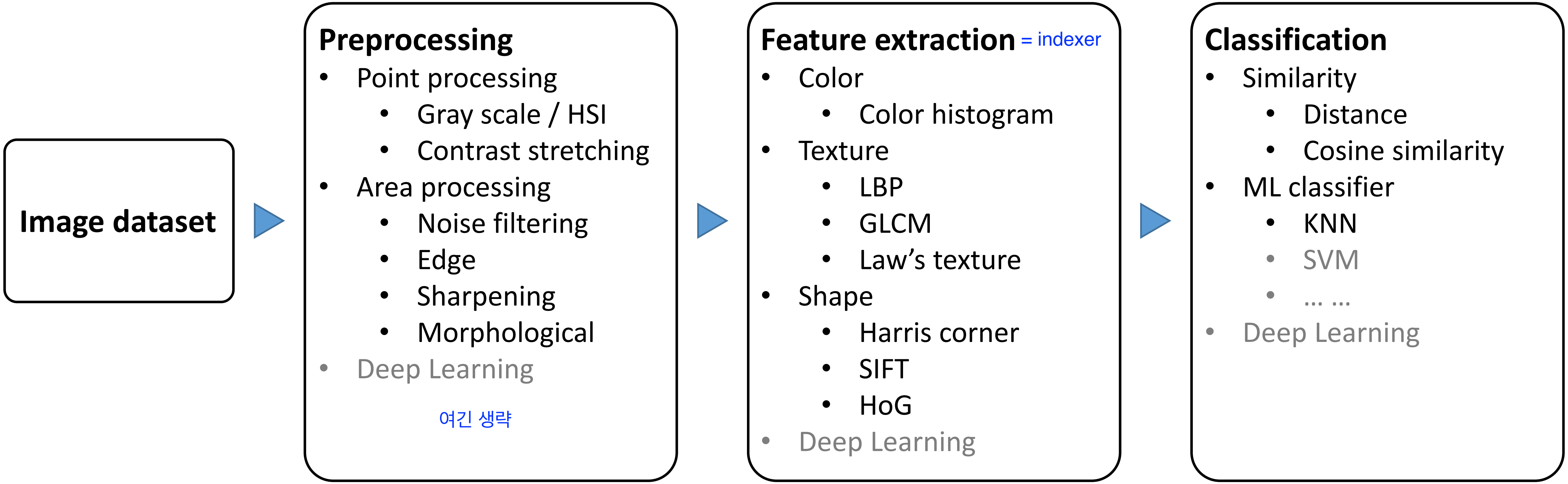


Image dataset

Preprocessing
- Point processing
  - Gray scale / HSI
  - Contrast stretching
- Area processing
  - Noise filtering
  - Edge
  - Sharpening
  - Morphological
- Deep Learning

여긴 생략

Feature extraction = indexer
- Color
  - Color histogram
- Texture
  - LBP
  - GLCM
  - Law’s texture
- Shape
  - Harris corner
  - SIFT
  - HoG
- Deep Learning

Classification
- Similarity
  - Distance
  - Cosine similarity
- ML classifier
  - KNN
  - SVM
  - … …
- Deep Learning

early vision 으로 뽑아낼 수 있음

Image Matching System Challenge 1 & 2

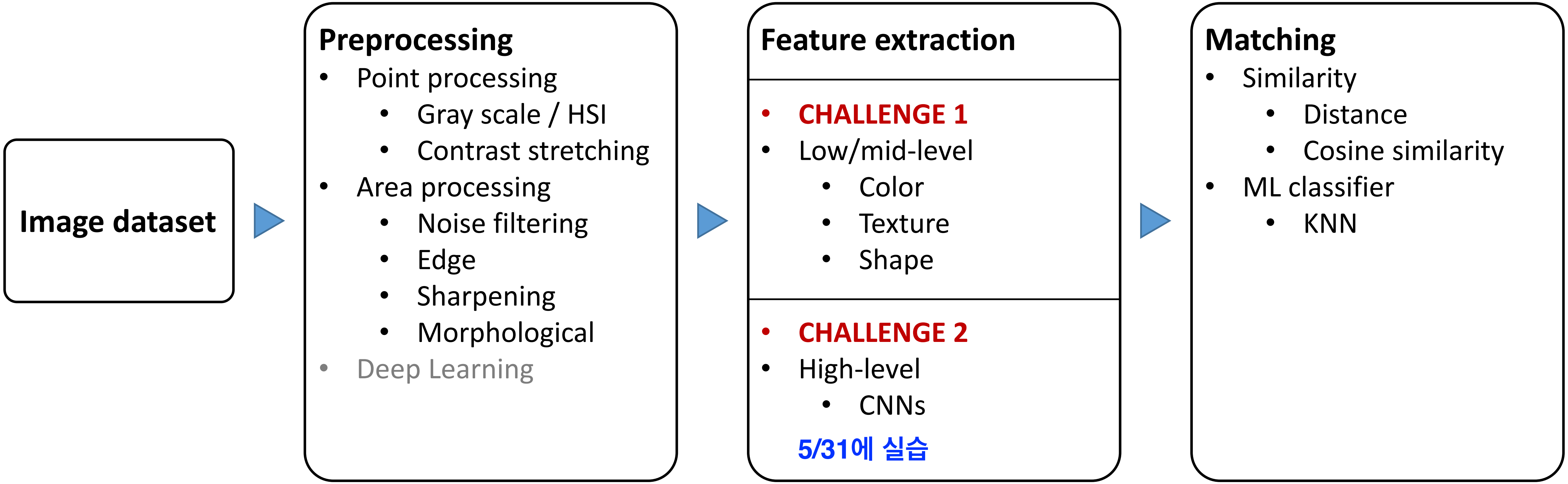


Image dataset

Preprocessing

- Point processing
  - Gray scale / HSI
  - Contrast stretching
- Area processing
  - Noise filtering
  - Edge
  - Sharpening
  - Morphological
- Deep Learning

Feature extraction

- **CHALLENGE 1**
- Low/mid-level
  - Color
  - Texture
  - Shape

- **CHALLENGE 2**
- High-level
  - CNNs

5/31에 실습

Matching

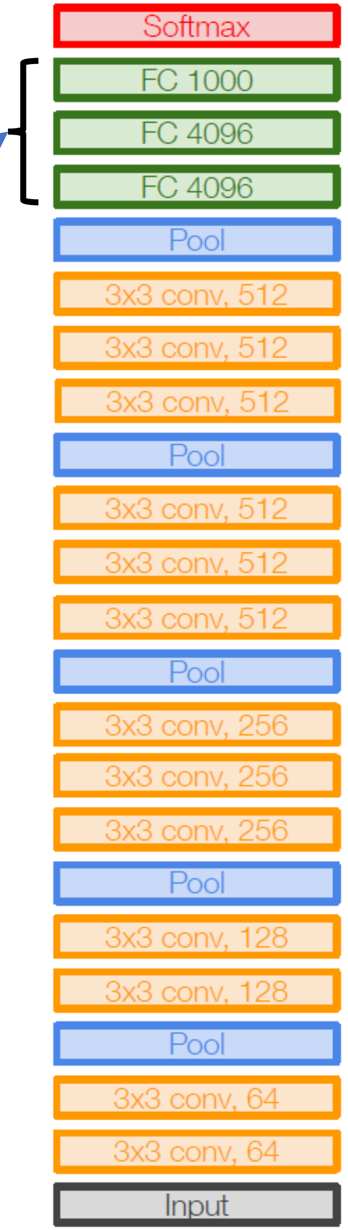
- Similarity
  - Distance
  - Cosine similarity
- ML classifier
  - KNN

CHALLENGE 2

- Convolutional Neural Networks (CNNs)
  - will be covered in next class

- Extract CNN output feature vectors

- Pytorch-based CNN practice in 5.31 (online)



Softmax

FC 1000

FC 4096

FC 4096

Pool

3x3 conv, 512

3x3 conv, 512

3x3 conv, 512

Pool

3x3 conv, 512

3x3 conv, 512

3x3 conv, 512

Pool

3x3 conv, 256

3x3 conv, 256

3x3 conv, 256

Pool

3x3 conv, 128

3x3 conv, 128

Pool

3x3 conv, 64

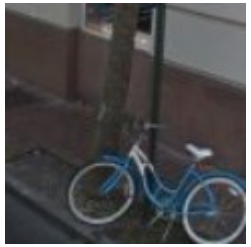
3x3 conv, 64

Input

VGG16

DB Images (train image)

- link
  - git clone https://github.com/folfcoder/recaptcha-dataset.git
- Classes (without mountain)



Bicycle (800)

Bridge (553)

Bus (1229)

Car (3578)

Chimney (56)

Crosswalk (1260)

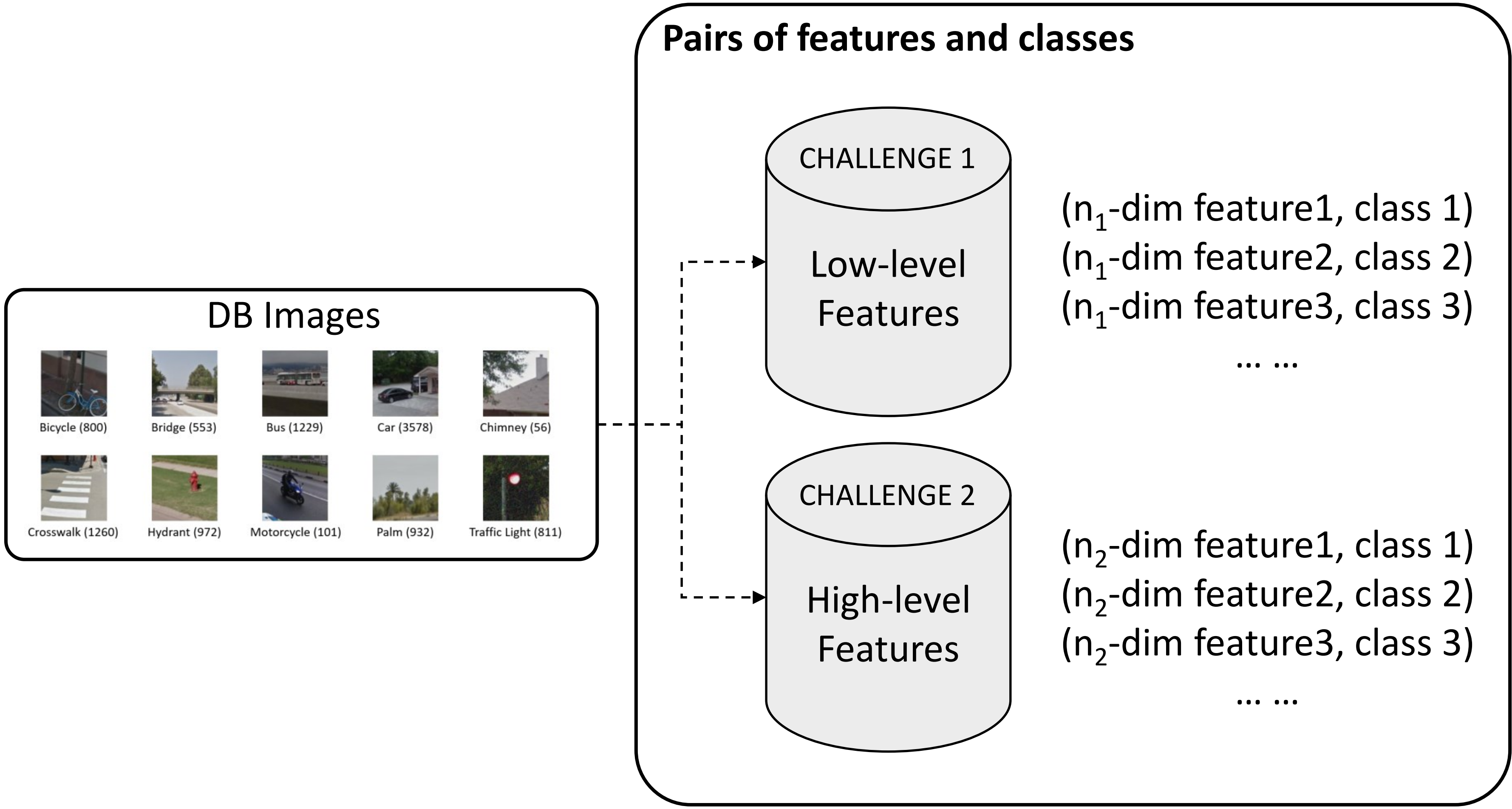
Hydrant (972)

Motorcycle (101)

Palm (932)

Traffic Light (811)

Representing DB images by feature vectors



Pairs of features and classes

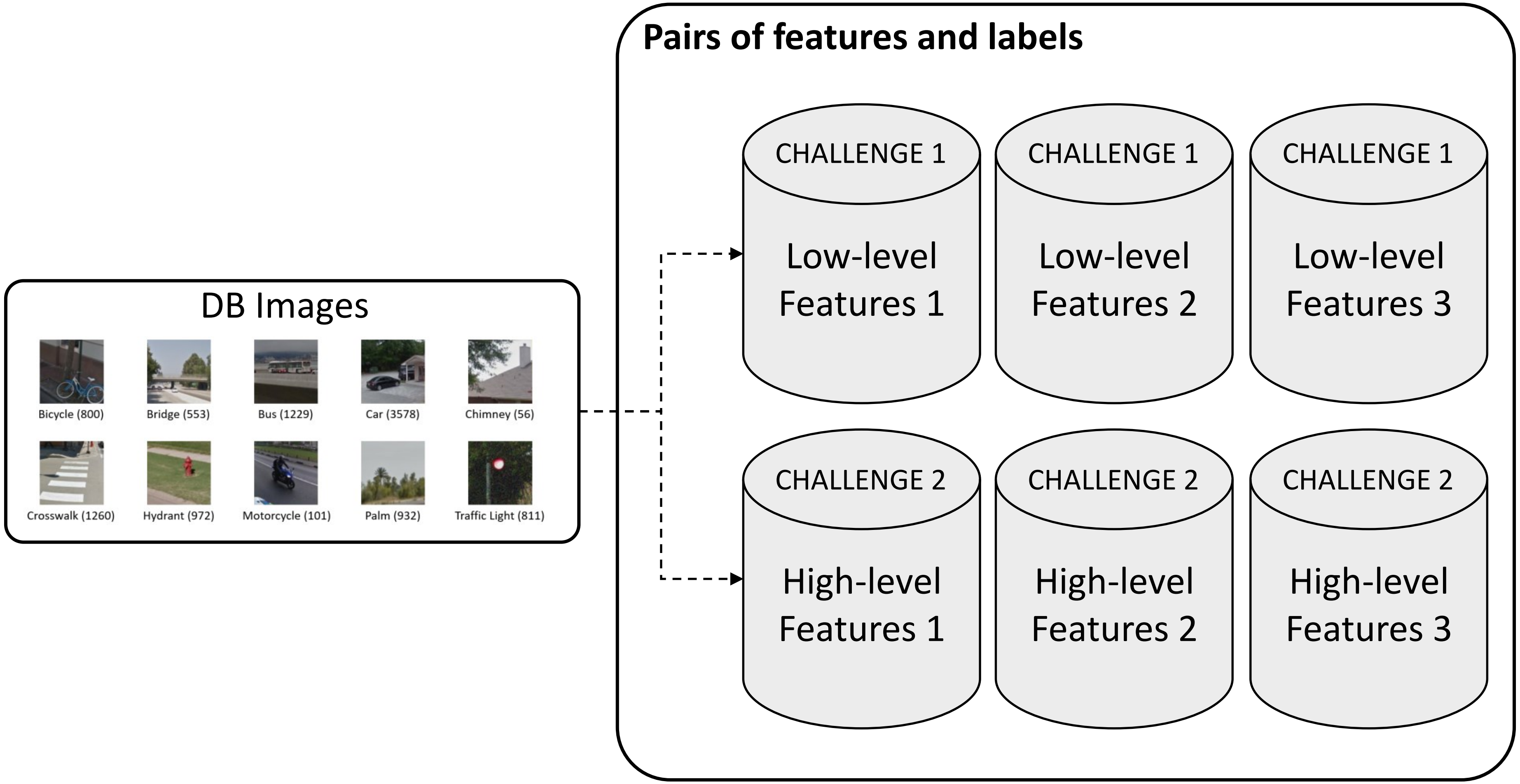
(n_1-dim feature1, class 1)
(n_1-dim feature2, class 2)
(n_1-dim feature3, class 3)
… …

(n_2-dim feature1, class 1)
(n_2-dim feature2, class 2)
(n_2-dim feature3, class 3)
… …

feature 잘 정의되었는지 검증하고 와야함

Explore and save the best feature sets using cross-validation in DB image

Representing DB images by feature vectors



DB Images

Pairs of features and labels

CHALLENGE 1 Low-level Features 1

CHALLENGE 1 Low-level Features 2

CHALLENGE 1 Low-level Features 3

CHALLENGE 2 High-level Features 1

CHALLENGE 2 High-level Features 2

CHALLENGE 2 High-level Features 3

한 챌린지 당 3개까지 저장할 수 있음

You can save up to three feature sets for each CHALLENGE

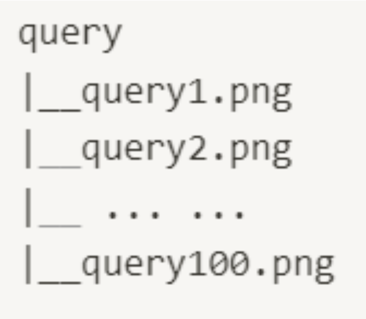
Query Images (test image)

- Total 100 images
  - 10 images for each class

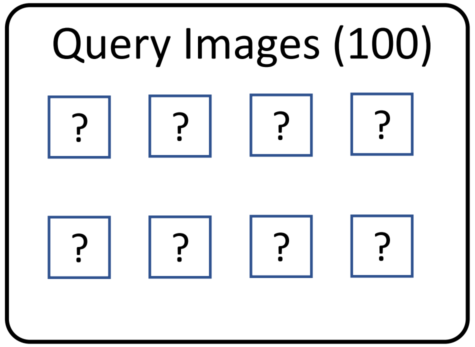
query image = test image 총 100개

- Query Images will be available on **12 June (Mon)**
  - We provide query images in class, without label

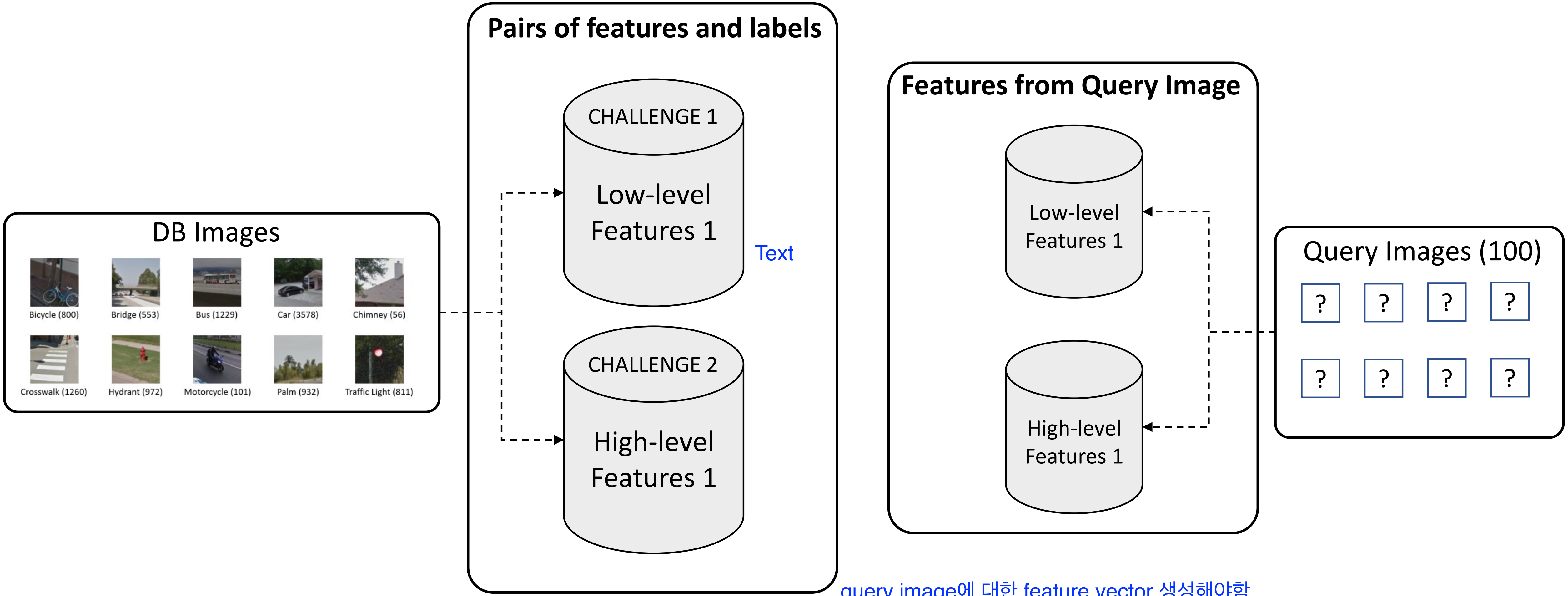
- Query folder structure



```
query
|__query1.png
|__query2.png
|__ ... ...
|__query100.png
```



During the Challenge



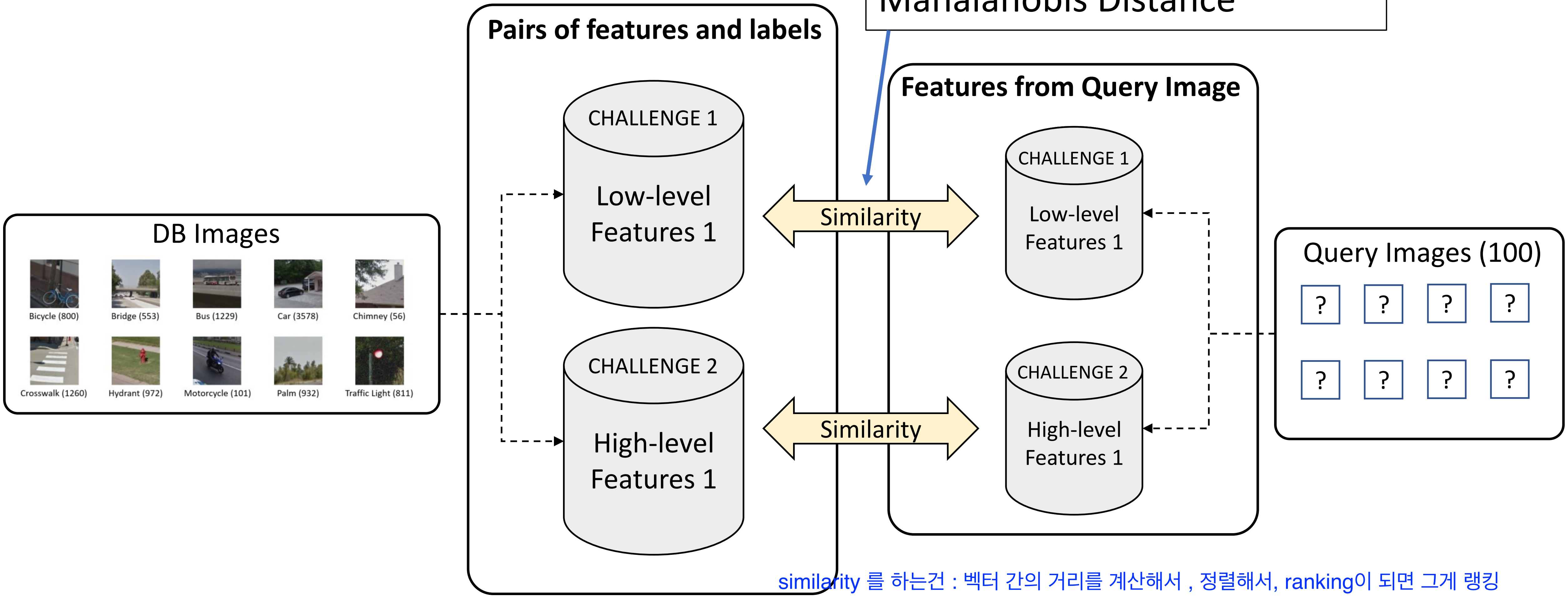
Text

query image에 대한 feature vector 생성해야함

Step 1. representing query images by your own feature sets

During the Challenge

Cosine Similarity
Manhattan distance (L1 norm)
Euclidean distance (L2 norm)
Mahalanobis Distance



DB Images

Bicycle (800), Bridge (553), Bus (1229), Car (3578), Chimney (56)

Crosswalk (1260), Hydrant (972), Motorcycle (101), Palm (932), Traffic Light (811)

Pairs of features and labels

CHALLENGE 1
Low-level Features 1

CHALLENGE 2
High-level Features 1

Similarity

Similarity

Features from Query Image

CHALLENGE 1
Low-level Features 1

CHALLENGE 2
High-level Features 1

Query Images (100)

? ? ? ?
? ? ? ?

similarity 를 하는건 : 벡터 간의 거리를 계산해서 , 정렬해서, ranking이 되면 그게 랭킹

Step2. Calculate the similarity between features of all DB Images and given query Image

Tasks for the CHALLENGE

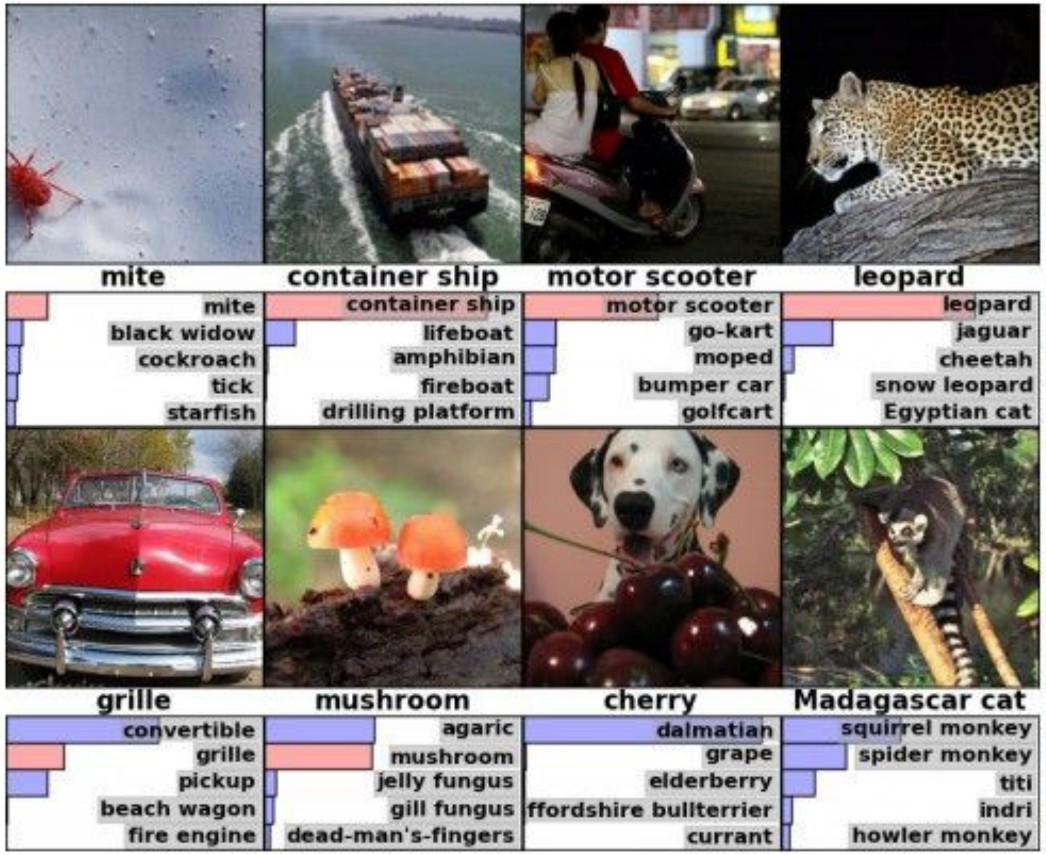
10개 안에 몇개를 맞추냐

그래서 knn을 하는거고, k값은 우리가 조절하면 됨

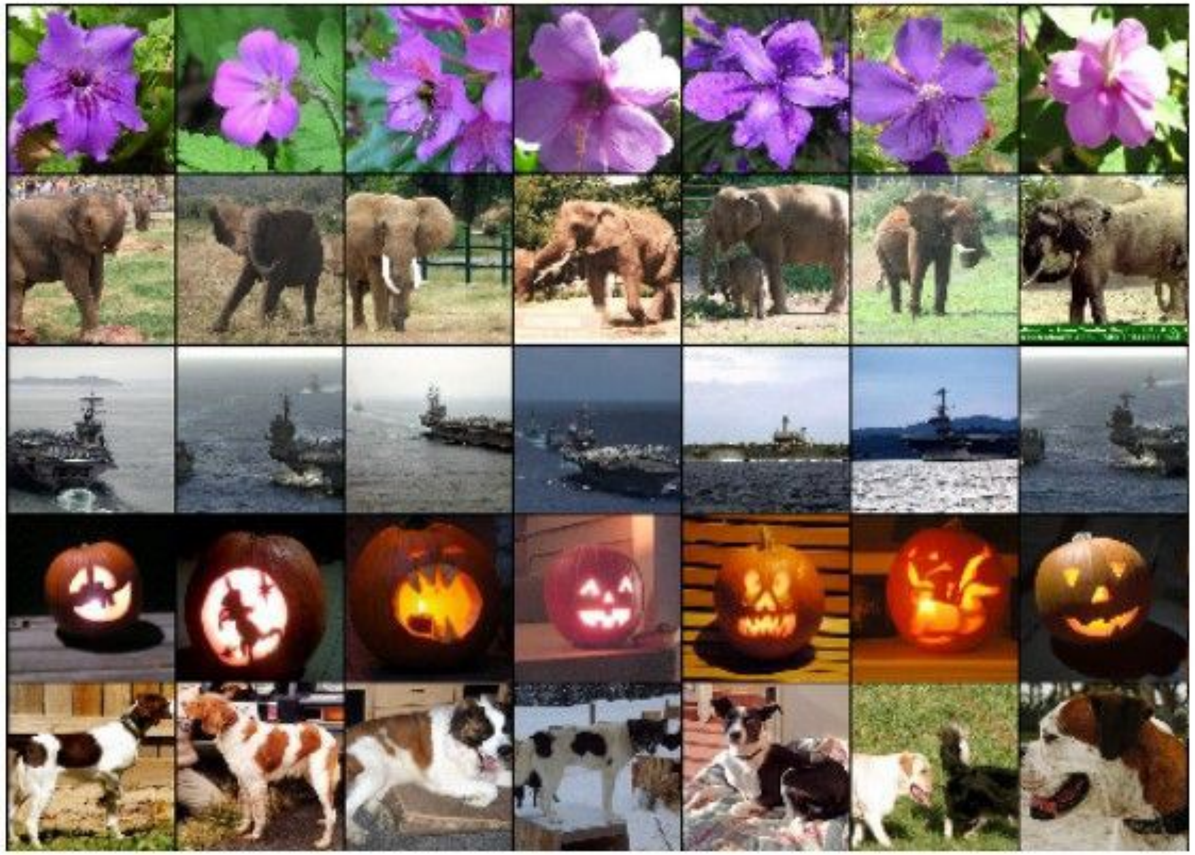
task 1 : class를 정확히 맞추는 것

task 2 : 검색 시스템으로서의 검증

Task1: Classification

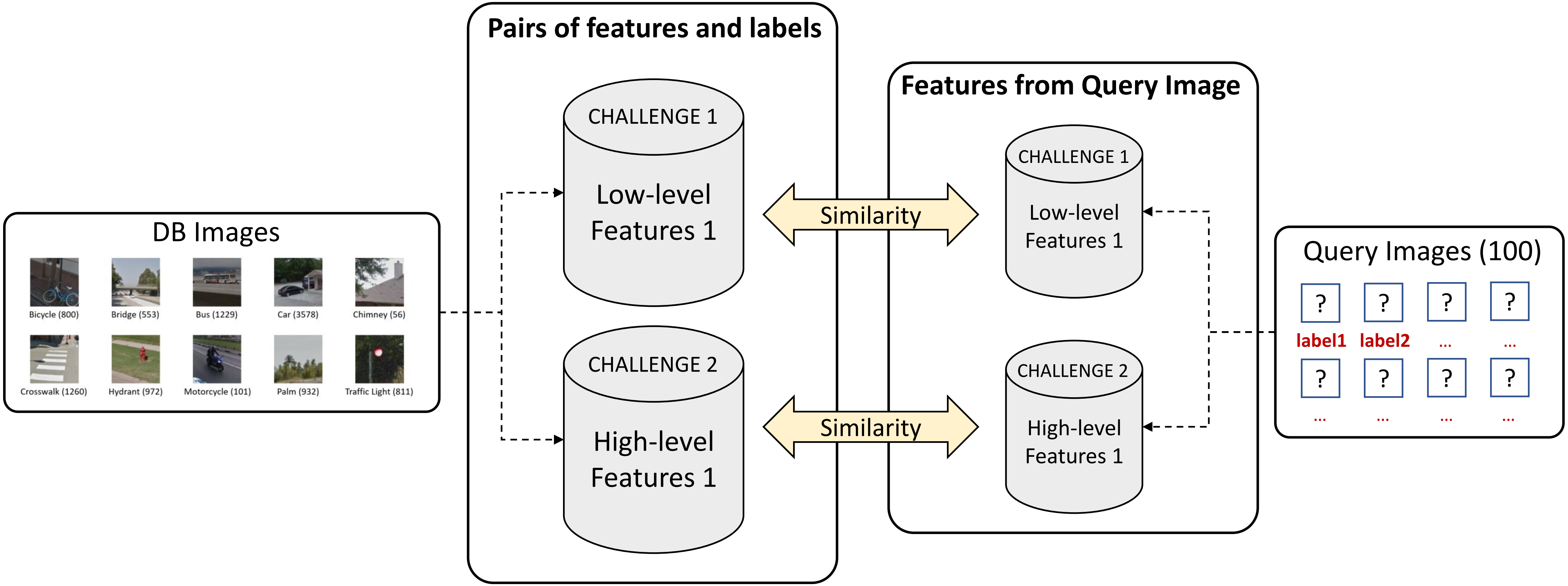


Task2: Retrieval



Figures copyright Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton, 2012.

During the Challenge

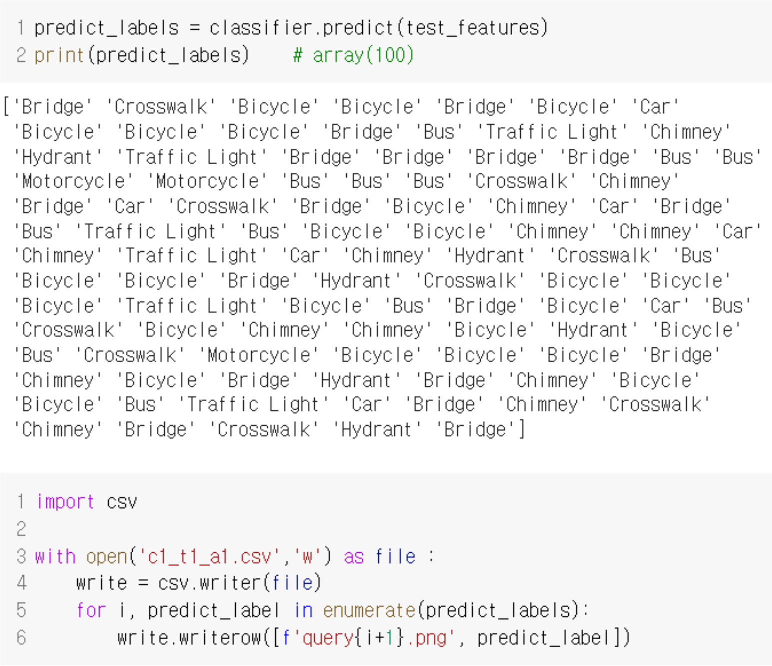


Step3. Classification Result (Task 1): Classify and tag the label for every query image

Step4. Retrieval Result (Task 2): Retrieve top ranked 10 images for every query image

Classification Result (Task 1)

- Outputs
  - 100 dimension array
  - Save the results as csv file for each attempt



```
predict_labels = classifier.predict(test_features)
print(predict_labels)     # array(100)
```

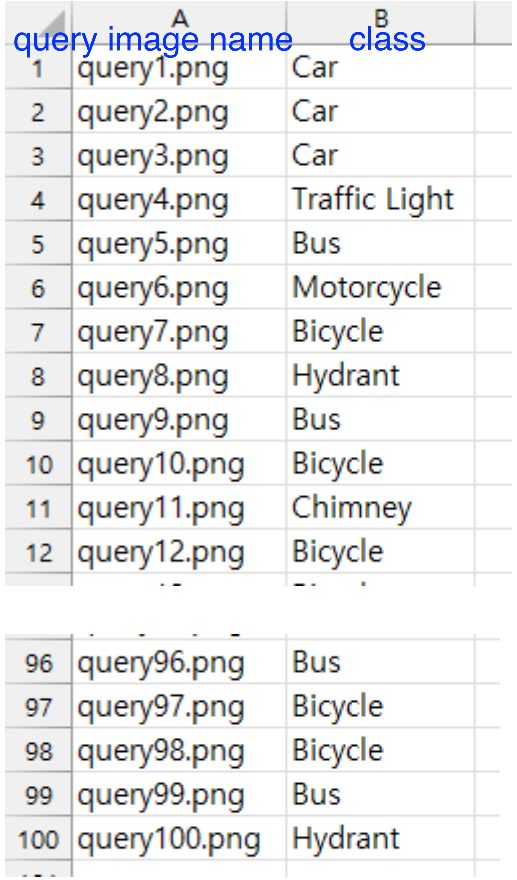
```
['Bridge' 'Crosswalk' 'Bicycle' 'Bicycle' 'Bridge' 'Bicycle' 'Car'
 'Bicycle' 'Bicycle' 'Bicycle' 'Bridge' 'Bus' 'Traffic Light' 'Chimney'
 'Hydrant' 'Traffic Light' 'Bridge' 'Bridge' 'Bridge' 'Bridge' 'Bus' 'Bus'
 'Motorcycle' 'Motorcycle' 'Bus' 'Bus' 'Bus' 'Crosswalk' 'Chimney'
 'Bridge' 'Car' 'Crosswalk' 'Bridge' 'Bicycle' 'Chimney' 'Car' 'Bridge'
 'Bus' 'Traffic Light' 'Bus' 'Bicycle' 'Bicycle' 'Chimney' 'Chimney' 'Car'
 'Chimney' 'Traffic Light' 'Car' 'Chimney' 'Hydrant' 'Crosswalk' 'Bus'
 'Bicycle' 'Bicycle' 'Bridge' 'Hydrant' 'Crosswalk' 'Bicycle' 'Bicycle'
 'Bicycle' 'Traffic Light' 'Bicycle' 'Bus' 'Bridge' 'Bicycle' 'Car' 'Bus'
 'Crosswalk' 'Bicycle' 'Chimney' 'Chimney' 'Bicycle' 'Hydrant' 'Bicycle'
 'Bus' 'Crosswalk' 'Motorcycle' 'Bicycle' 'Bicycle' 'Bicycle' 'Bridge'
 'Chimney' 'Bicycle' 'Bridge' 'Hydrant' 'Bridge' 'Chimney' 'Bicycle'
 'Bicycle' 'Bus' 'Traffic Light' 'Car' 'Bridge' 'Chimney' 'Crosswalk'
 'Chimney' 'Bridge' 'Crosswalk' 'Hydrant' 'Bridge']
```

```
import csv

with open('c1_t1_a1.csv','w') as file :
    write = csv.writer(file)
    for i, predict_label in enumerate(predict_labels):
        write.writerow([f'query{i+1}.png', predict_label])
```

결과물 이렇게 제출해야함

c1_t1_a1.csv

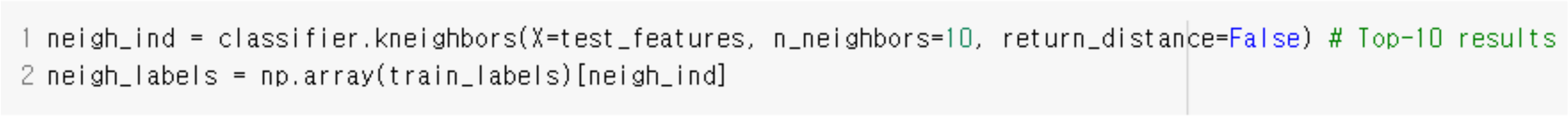


	A	B
	query image name	class
1	query1.png	Car
2	query2.png	Car
3	query3.png	Car
4	query4.png	Traffic Light
5	query5.png	Bus
6	query6.png	Motorcycle
7	query7.png	Bicycle
8	query8.png	Hydrant
9	query9.png	Bus
10	query10.png	Bicycle
11	query11.png	Chimney
12	query12.png	Bicycle
96	query96.png	Bus
97	query97.png	Bicycle
98	query98.png	Bicycle
99	query99.png	Bus
100	query100.png	Hydrant

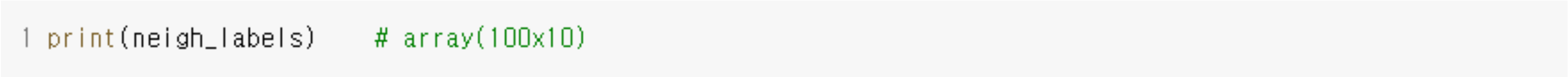
https://colab.research.google.com/drive/1nC_YMtHQn1GYMzBUGRS3esftKS8YKAZn?hl=ko#scrollTo=Ad4y-3QSh5mA

Retrieval Result (Task 2)

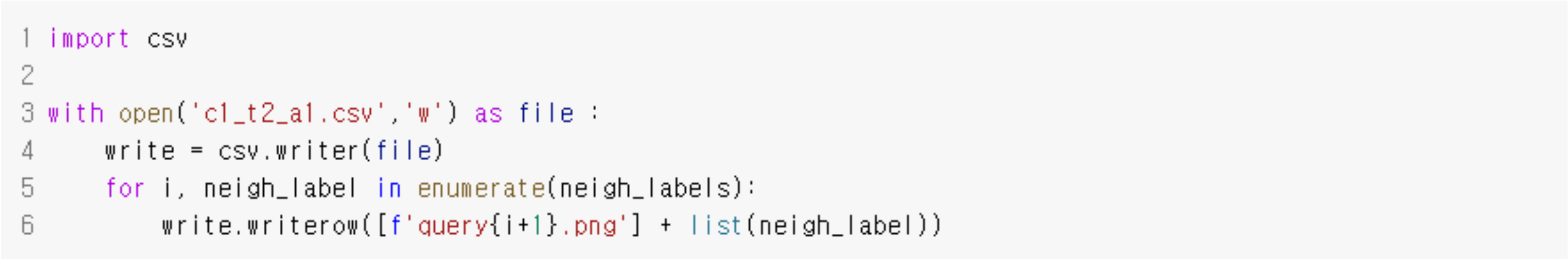
- Outputs
  - 100x10 dimension array
  - Save the results as csv file for each attempt



```
neigh_ind = classifier.kneighbors(X=test_features, n_neighbors=10, return_distance=False) # Top-10 results
neigh_labels = np.array(train_labels)[neigh_ind]
```



```
print(neigh_labels)     # array(100x10)
```



```
import csv

with open('c1_t2_a1.csv','w') as file :
    write = csv.writer(file)
    for i, neigh_label in enumerate(neigh_labels):
        write.writerow([f'query{i+1}.png'] + list(neigh_label))
```

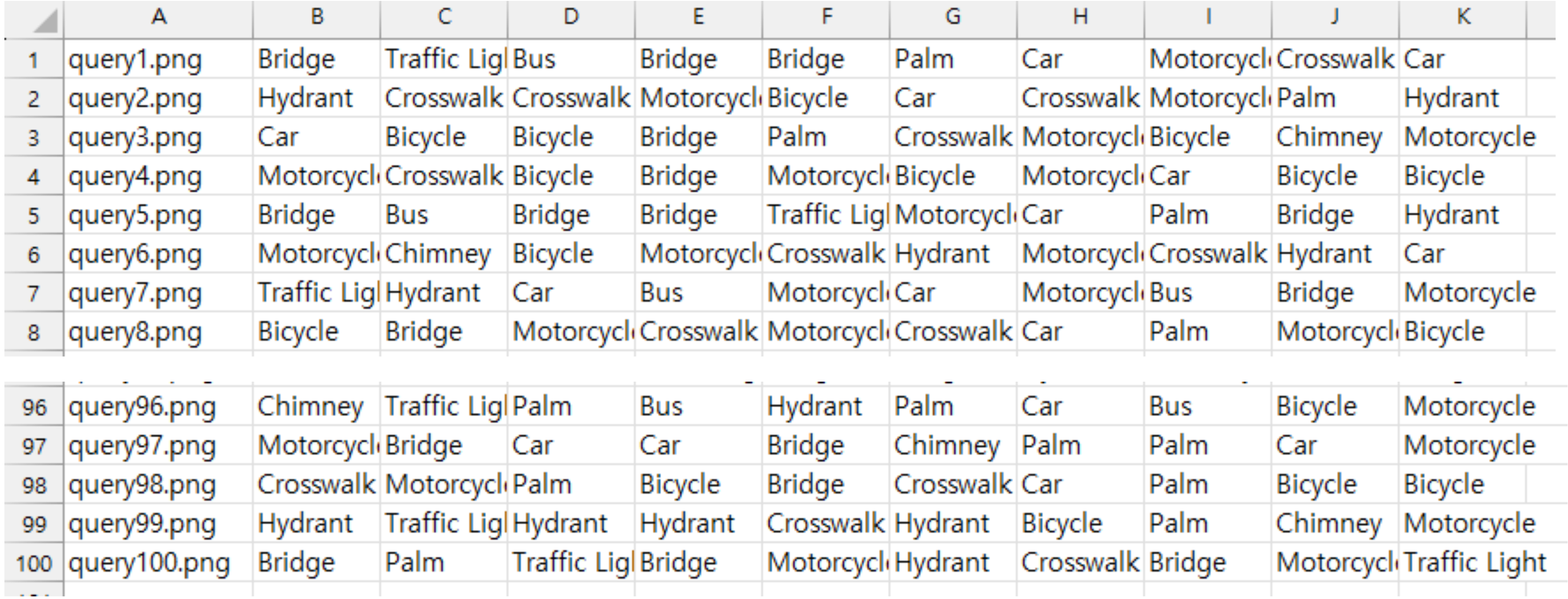
https://colab.research.google.com/drive/1nC_YMtHQn1GYMzBUGRS3esftKS8YKAZn?hl=ko#scrollTo=Ad4y-3QSh5mA

Retrieval Result (Task 2)

- Outputs
  - 100x10 dimension array
  - Save the results as csv file for each attempt

엑셀 파일을 제출
각 챌린지의 피쳐들 마다 (최대 3개) 파일 제출

c1_t2_a1.csv



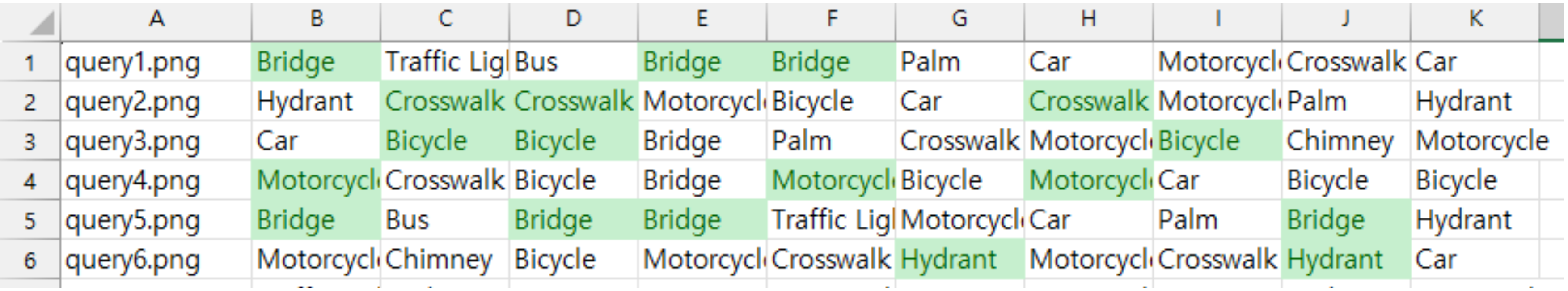
	A	B	C	D	E	F	G	H	I	J	K
1	query1.png	Bridge	Traffic Lig	Bus	Bridge	Bridge	Palm	Car	Motorcycl	Crosswalk	Car
2	query2.png	Hydrant	Crosswalk	Crosswalk	Motorcycl	Bicycle	Car	Crosswalk	Motorcycl	Palm	Hydrant
3	query3.png	Car	Bicycle	Bicycle	Bridge	Palm	Crosswalk	Motorcycl	Bicycle	Chimney	Motorcycle
4	query4.png	Motorcycl	Crosswalk	Bicycle	Bridge	Motorcycl	Bicycle	Motorcycl	Car	Bicycle	Bicycle
5	query5.png	Bridge	Bus	Bridge	Bridge	Traffic Lig	Motorcycl	Car	Palm	Bridge	Hydrant
6	query6.png	Motorcycl	Chimney	Bicycle	Motorcycl	Crosswalk	Hydrant	Motorcycl	Crosswalk	Hydrant	Car
7	query7.png	Traffic Lig	Hydrant	Car	Bus	Motorcycl	Car	Motorcycl	Bus	Bridge	Motorcycle
8	query8.png	Bicycle	Bridge	Motorcycl	Crosswalk	Motorcycl	Crosswalk	Car	Palm	Motorcycl	Bicycle
96	query96.png	Chimney	Traffic Lig	Palm	Bus	Hydrant	Palm	Car	Bus	Bicycle	Motorcycle
97	query97.png	Motorcycl	Bridge	Car	Car	Bridge	Chimney	Palm	Palm	Car	Motorcycle
98	query98.png	Crosswalk	Motorcycl	Palm	Bicycle	Bridge	Crosswalk	Car	Palm	Bicycle	Bicycle
99	query99.png	Hydrant	Traffic Lig	Hydrant	Hydrant	Crosswalk	Hydrant	Bicycle	Palm	Chimney	Motorcycle
100	query100.png	Bridge	Palm	Traffic Lig	Bridge	Motorcycl	Hydrant	Crosswalk	Bridge	Motorcycl	Traffic Light

https://colab.research.google.com/drive/1nC_YMtHQn1GYMzBUGRS3esftKS8YKAZn?hl=ko#scrollTo=Ad4y-3QSh5mA

Ranking

- Task1: Classification
  1. Highest accuracy of all attempts
  2. Ties will be broken by F1 score 동점시 F1 점수를 봄

- Task2: Retrieval
  1. Highest (average) Top-10 accuracy of all attempts
  2. Ties will be broken by number of attempts



	A	B	C	D	E	F	G	H	I	J	K
1	query1.png	Bridge	Traffic Lig	Bus	Bridge	Bridge	Palm	Car	Motorcycl	Crosswalk	Car
2	query2.png	Hydrant	Crosswalk	Crosswalk	Motorcycl	Bicycle	Car	Crosswalk	Motorcycl	Palm	Hydrant
3	query3.png	Car	Bicycle	Bicycle	Bridge	Palm	Crosswalk	Motorcycl	Bicycle	Chimney	Motorcycle
4	query4.png	Motorcycl	Crosswalk	Bicycle	Bridge	Motorcycl	Bicycle	Motorcycl	Car	Bicycle	Bicycle
5	query5.png	Bridge	Bus	Bridge	Bridge	Traffic Lig	Motorcycl	Car	Palm	Bridge	Hydrant
6	query6.png	Motorcycl	Chimney	Bicycle	Motorcycl	Crosswalk	Hydrant	Motorcycl	Crosswalk	Hydrant	Car

Matching Results

- You should upload
  - Output files
    - c1_t1_a1.csv, c1_t1_a2.csv, c1_t1_a3.csv
    - c1_t2_a1.csv, c1_t2_a2.csv, c1_t2_a3.csv
    - c2_t1_a1.csv, c2_t1_a2.csv, c2_t1_a3.csv
    - c2_t2_a1.csv, c2_t2_a2.csv, c2_t2_a3.csv
  - Python code (.py or .ipynb)
    - c1_t1_a1.ipynb, c1_t1_a2.ipynb, c1_t1_a3.ipynb
    - c1_t2_a1.ipynb, c1_t2_a2.ipynb, c1_t2_a3.ipynb
    - c2_t1_a1.ipynb, c2_t1_a2.ipynb, c2_t1_a3.ipynb
    - c2_t2_a1.ipynb, c2_t2_a2.ipynb, c2_t2_a3.ipynb
  - DB features file (.npy or .pkl or others) if necessary

Tips & Tricks

중요) k 값 조정
논문 찾아보면 좋을거야
이거 쓰면 좋을까요? 궁금하면 물어봐라, 대답해줌

- Normalize the features

  feature 를 많이 쓴다고 좋은거 아님, 핵심적인 feature를 잘 찾아보기

  - Normalize the features to have zero mean and unit variance
  - e.g. Concatenation(Norm(GLCM), Norm(histogram))

- Consider using a dimensionality reduction technique
  - It works well on low-dimension data (compact features)

- Cross-validation
  - try different hyperparameter values and keep the values that lead to the best performance on the validation set