

**VIETNAM NATIONAL UNIVERSITY HO CHI MINH CITY  
UNIVERSITY OF INFORMATION TECHNOLOGY**



**UIT**

**UNIVERSITY OF INFORMATION TECHNOLOGY**

# **PROJECT REPORT**

**SUBJECT: MULTIMEDIA INFORMATION RETRIEVAL**

**TOPIC**

**IMAGE RETRIEVAL**

**Lecturer:** Ngo Duc Thanh

**Team's member:** To Thanh Hien - 19521490

Tran Vi Hao - 19521482

Le Đang Đang Huy - 19521612

Nguyen Duong Hai - 19521464

**Class:** CS336.N11.KHCL

*Ho Chi Minh City, February 10<sup>th</sup>, 2023*

## TABLE OF CONTENTS

I. OVERVIEW .....	3
1. Introduction .....	3
2. Description of problem .....	3
3. Dataset .....	3
a. The Oxford Buildings Dataset (Oxford5k) .....	3
b. The Paris Dataset (Paris6k) .....	4
II. BUILD SYSTEM .....	5
1. Feature extraction .....	5
2. Create index table .....	6
3. Query .....	6
III. EVALUATION AND RESULT .....	7
1. Evaluation .....	7
a. Precision .....	7
b. Precision at k .....	8
c. Average precision .....	8
d. Mean average precision (mAP) .....	9
2. Result .....	9
IV. CONCLUSION .....	11
V. DEMO .....	12
VI. REFERENCES .....	13

# I. OVERVIEW

## 1. Introduction

Image Retrieval or Image Search is not a recent concept. Back in 2017, Google Lens and Pinterest Lens were launched together. At the same time, the ASOS Style Match application was also born. Despite their different functions, the apps all have one big feature in common, which is that they allow users to take or upload an image of an object and get information about that object or suggest it to them similar objects.

And this has brought great value, in just one year, Pinterest Lens has received more than 600 million searches per month. In 2019, statistics have shown that more than 1 billion queries are made through Google Lens. ASOS makes a profit from providing products that are similar to the user queries they have.

All of these applications are based on the image search engine. Bing, Amazon, eBay and dozens of smaller companies have also begun taking the customer experience to the next level. This is why it's time to start focusing more on image search.

## 2. Description of problem

In this project, we will learn about the Image Retrieval problem with:

- + Input:
  - A large database of photos.
  - An image or an image region used for querying related images in the database.
- + Output: A list of images most relevant to the query image.

To build this query system, we propose the **EfficientNetV2L** model and will experiment in turn on two separate datasets.

## 3. Dataset

### a. The Oxford Buildings Dataset (Oxford5k)

The Oxford Buildings Dataset consists of 5062 images collected from Flickr by searching for particular Oxford landmarks. The collection has been manually annotated to generate a comprehensive ground truth for 11 different landmarks, each represented by 5 possible queries. This gives a set of 55 queries over which an object retrieval system can be evaluated.



For each image and landmark in our dataset, one of four possible labels was generated:

- Good - A nice, clear picture of the object/building.
- OK - More than 25% of the object is clearly visible.
- Bad - The object is not present.
- Junk - Less than 25% of the object is visible, or there are very high levels of occlusion or distortion.

#### **b. The Paris Dataset (Paris6k)**

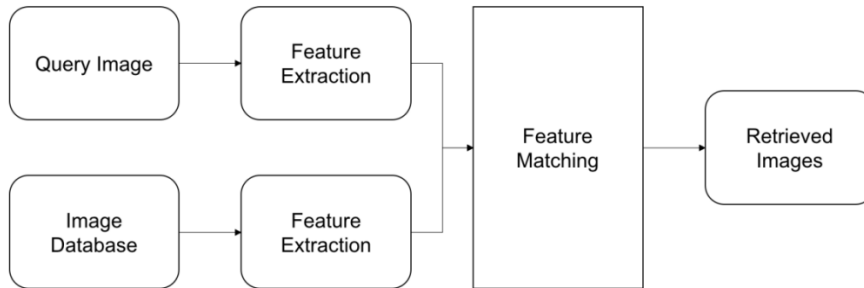
The Paris Dataset consists of 6412 images collected from Flickr by searching for particular Paris landmarks.



## II. BUILD SYSTEM

To build the Image Query system in this project, our work will follow the following steps sequentially:

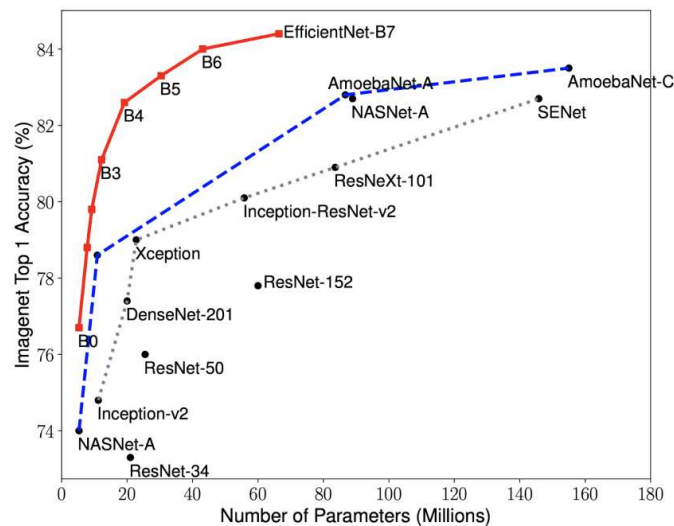
1. Download data.
2. Vectorizing (feature extraction) image data and storing, we have a set of vectors  $V$  with  $N$  vectors.
3. Vectorize (feature extraction) the image to be queried.
4. Calculate the distance from vector  $v$  to all vectors in set  $V$ .
5. Display  $K$  images with the closest distance.



### 1. Feature extraction

In the image feature extraction step, we use EfficientNetB7 model. As for why we chose this model, we must mention the article “EfficientNet: Improving Accuracy and Efficiency through AutoML and Model Scaling” by Mingxing Tan and Quoc V. Le. There, they clearly showed the performance superiority of the model compared to the rest of the methods on ImageNet. Among the variants from B0 to B7, we decided to choose B7, simply because the performance of the B7 is the highest of the variants and also the most modern. Specifically, in the blog post they wrote the following:

“We have compared our EfficientNets with other existing CNNs on ImageNet. In general, the EfficientNet models achieve both higher accuracy and better efficiency over existing CNNs, reducing parameter size and FLOPS by an order of magnitude. For example, in the high-accuracy regime, our EfficientNet-B7 reaches state-of-the-art 84.4% top-1 / 97.1% top-5 accuracy on ImageNet, while being 8.4x smaller and 6.1x faster on CPU inference than the previous Gpipe. Compared with the widely used ResNet-50, our EfficientNet-B4 uses similar FLOPS, while improving the top-1 accuracy from 76.3% of ResNet-50 to 82.6% (+6.3%).



Though EfficientNets perform well on ImageNet, to be most useful, they should also transfer to other datasets. To evaluate this, we tested EfficientNets on eight widely used transfer learning datasets. EfficientNets achieved state-of-the-art accuracy in 5 out of the 8 datasets, such as CIFAR-100 (91.7%) and Flowers (98.8%), with an order of magnitude fewer parameters (up to 21x parameter reduction), suggestive that our EfficientNets also transfer well. "

In this step, the input image is transformed from (224, 224, 3) to (1, 224, 224, 3) to fit the input of the model. The vector then continues through the pattern to form (1, 7, 7, 2560). Finally go through flatten to (125440, ) and normalize to get the final output.

## 2. Create index table

With the features extracted above, we start to create the index table as follows:

Index	Image Path	Image Feature
0	a/b/c/p0.jpg	[0.001, 0.002, ..., 0.123]
first	a/b/c/p1.jpg	[0.001, 0.002, ..., 0.123]
2	a/b/c/p2.jpg	[0.001, 0.002, ..., 0.123]
3	a/b/c/p3.jpg	[0.001, 0.002, ..., 0.123]
4	a/b/c/p4.jpg	[0.001, 0.002, ..., 0.123]
5	a/b/c/p5.jpg	[0.001, 0.002, ..., 0.123]

These values are then stored into two \*.npy files:

- + feature\_list.npy (contains image feature)
- + feature\_id.npy (contains image path)

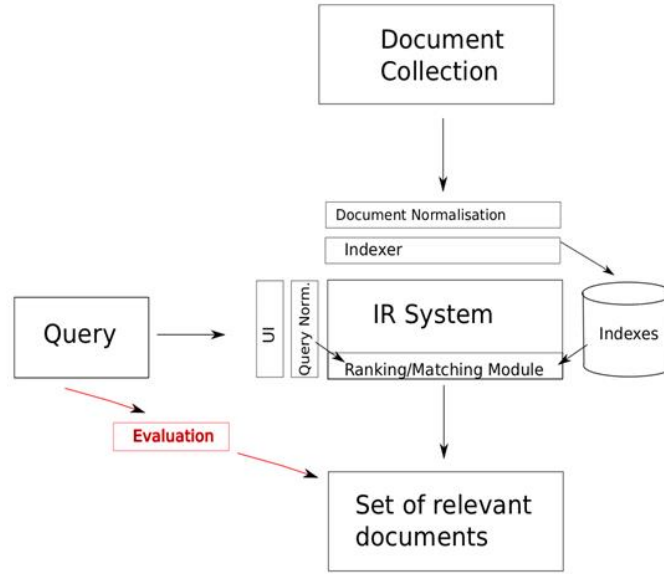
Creating the index table as above is done only once and offline, in case we want to update the database in the future, we have to do this step again.

## 3. Query

Finally, the query step, in this step we will read 2 \*.npy files received from the above step. Then we create a KDTree (**Nearest neighbor search**) with feature\_list. When querying, we use the feature of the current image to calculate dist and indx (indx is a list of indexes corresponding to query input in order of *top2bot* ). Finally, we display the image according to the paths in the feature\_id corresponding to the indx.

### III. EVALUATION AND RESULT

#### 1. Evaluation



We use the **mean average precision** (mAP) to measure the accuracy of information retrieval models.

The mAP is a value between 0-1, with higher scores demonstrating a more accurate model.

We describe it by the following formula:

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

In the above formula,  $N$  is the total number of queries, and  $AP_i$  is the average precision of  $query_i$ . In simple terms, mAP is the average of average precisions across all queries.

To understand the calculation of mAP, we must first explore: *precision*, *precision at k* and *average precision*.

##### a. Precision

Precision is the ratio of correctly-identified positive values to the total number of predicted positive values.

We must slightly modify the formula for precision to use it with an information retrieval model, as follows:

$$Precision = \frac{|\{relevant\ documents\} \cap \{retrieved\ documents\}|}{|\{retrieved\ documents\}|}$$

The *relevant documents* set refer to the ground truth values. In other words, these are the expected documents that should be received for the given query.

The *retrieved documents* set includes the documents selected by the information model based on similarity scores. This set includes all relevant and non-relevant documents that the model picks. Hence, it is equal to  $TP + FP$  from the usual precision formula.


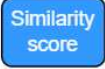

The intersection size of *relevant documents* and *retrieved documents* can be considered the **true positive count** ( $TP$ ).



## b. Precision at k

Usually, precision takes all the retrieved documents into account. However, another way to precision calculate involves a cut-off rank, k. Through this method, we calculate precision only for the top k documents. This metric is called precision at k, or **P(k)**.

To understand P(k), let's consider an information retrieval model that accepts a query and returns documents that are similar to that query.

 Database	D3	D4	D1	D5	D2
 Similarity score	0.8	0.7	0.5	0.3	0.1
 Is relevant?	Y	Y	N	Y	N
P(1)	P(2)	P(3)	P(4)	P(5)	
1/1 = 1	2/2 = 1	2/3 = 0.67	3/4 = 0.75	3/5 = 0.6	

Calculation of P(k) for k = 1 to 5

## c. Average precision

Average precision is the area under the precision-recall curve. For information retrieval models, where recall is a less critical metric, we can calculate AP using the following formula:

$$AP = \frac{1}{RD} \sum_{k=1}^n P(k)r(k)$$

Where RD is the number of relevant documents for the query, n is the total number of documents, P(k) is the precision at k, and r(k) is the relevance of the k<sup>th</sup> retrieved document (0 if not relevant, and 1 if relevant).

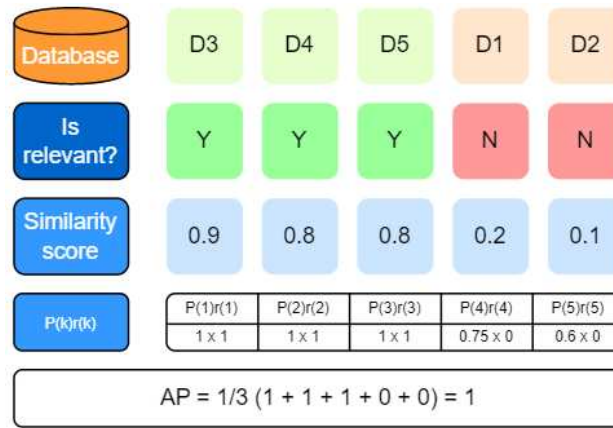
Let's continue with the previous example. We can calculate the AP of the information retrieval model as follows:

Database	D3	D4	D1	D5	D2
Is relevant?	Y	Y	N	Y	N
Precision at k	P(1) 1/1 = 1	P(2) 2/2 = 1	P(3) 2/3 = 0.67	P(4) 3/4 = 0.75	P(5) 3/5 = 0.6
Relevance of k	r(1) 1	r(2) 1	r(3) 0	r(4) 1	r(5) 0
P(k)r(k)	P(1)r(1) 1	P(2)r(2) 1	P(3)r(3) 0	P(4)r(4) 0.75	P(5)r(5) 0
AP = 1/3 (1 + 1 + 0 + 0.75 + 0) = 0.92					

We calculate AP, where the number of relevant documents, RD, is equal to 3.



Let's consider another model that assigns the highest scores to the relevant documents.

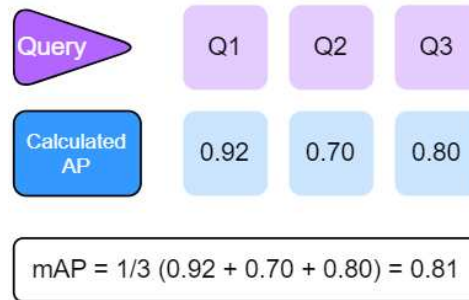


AP calculation for a more accurate model

As we can see, more accurate models - those that assign higher scores to the relevant documents - have greater average precision.

#### d. Mean average precision (mAP)

We repeat the calculation for AP for each query. The mAP across this set of queries is the average of the calculated APs.



Calculation of mAP across three queries

## 2. Result

We test, evaluate and compare the performance of the model we developed with other models that have been published in previous scientific papers. As follows:

Oxford5k	
Method	mAP
VGG GEM[1]	0,43
VGG MAC[2]	0,56
AlexNet MAC[3]	0,41
Resnet MAC[4]	0,69
<b>EfficientNetB7 (ours)</b>	<b>0,76</b>

In this dataset, our method is superior to the other methods, not only that, in terms of query time, we reach an approximate threshold of **84,53 seconds** system-wide, an acceptable number for the trade-off of query time for a high performance return.

In addition, our method also achieves high performance on the Paris6k dataset, the specific indexes are as follows:

Paris6k	
mAP	Time
<b>0,71</b>	<b>80,27sec</b>

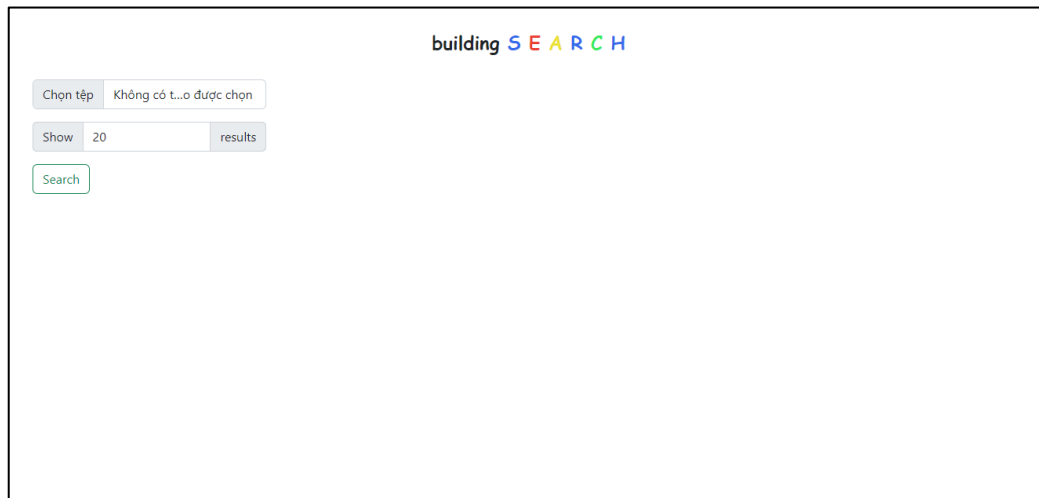
Although the system-wide query time is still quite high, with a certain number of results returned, the method we propose is really impressive. Specifically, we can clearly see this in [the demo video](#).

## IV. CONCLUSION

With this project, we clearly show the outstanding performance of EfficientNetB7 - one of the most modern and efficient models today in image feature extraction, compared to other methods. From there, we understand the step-by-step process to build an information query system. Also, our system isn't perfect yet, it still has problems with large amounts of data, it might be fine with datasets with a few thousand images but nothing can guarantee that it works well on datasets with hundreds of thousands of images and that's something we need to improve on in the future.

## V. DEMO

This is the main interface of the demo. In this project, we allow users to choose the number of output images, and users can also choose to crop a part or take the entire input image for querying. The source code is published by us [here](#).



Home page

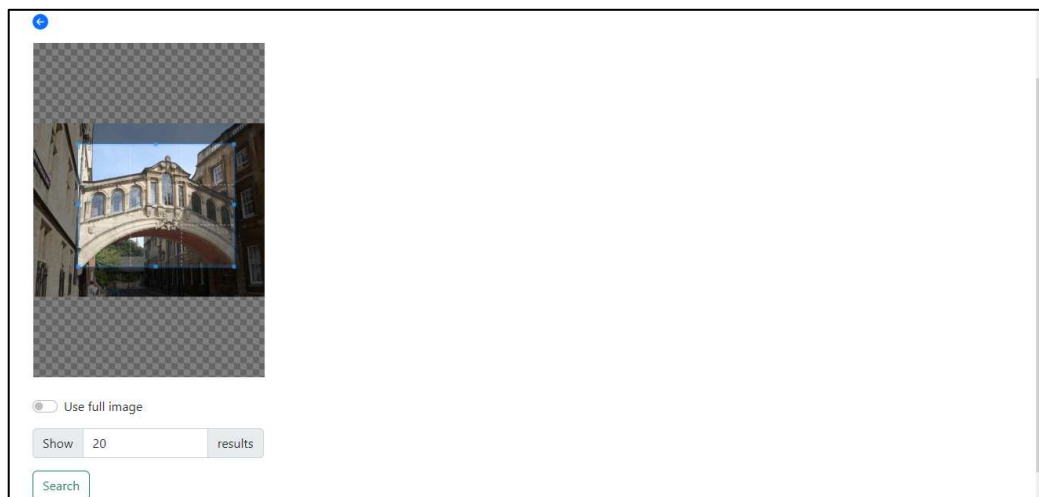
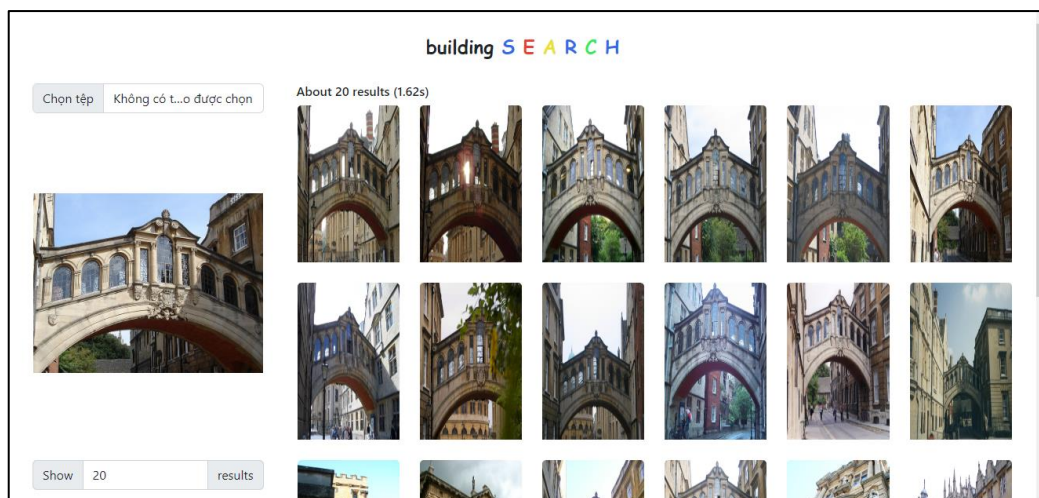


Image crop function



Output with 20 images and query time 1,62 seconds

## VI. REFERENCES

- [1] G. Tolias and O. Chum F. Radenovic, "Fine-tuning CNN Image Retrieval with No Human Annotation," *arXiv* , 2018.
- [2] J. Sullivan, S. Carlsson and A. Maki AS Razavian, "Visual instance retrieval with deep convolutional networks," *ITE Trans. MTA* , 2016.
- [3] G. Tolias, and O. Chum F. Radenovic, "CNN image retrieval learns from BoW: Unsupervised fine-tuning with hard examples," *ECCV* , 2016.
- [4] R. Sirc, and H. Jegou G. Tolias, "Particular object retrieval with integral max-pooling of CNN activations," *ICLR* , 2016.
- [5] Rana Abdul Muneem. (2023) educative.io. [Online]. <https://www.educative.io/answers/what-is-the-mean-average-precision-in-information-retrieval>