ĐẠI HỌC QUỐC GIA TP.HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

**Phòng Đào tạo Sau đại học & Khoa học công nghệ**

**ĐỒ ÁN MÔN HỌC**

**NHẬN DẠNG THÔNG TIN THỊ GIÁC**

GVHD:

Lê Đình Duy

Nguyễn Tấn Trần Minh Khang

Học viên:

Lê Hoàng Dũng - Mã HV: CH1501022

TP. Hồ Chí Minh – tháng 12/2017

**MỤC LỤC**

[TỔ CHỨC THƯ MỤC 2](#_Toc502006245)

[PHƯƠNG PHÁP XÂY DỰNG HỆ THỐNG 5](#_Toc502006246)

[TÀI LIỆU THAM KHẢO 12](#_Toc502006247)

[MỘT SỐ SOURCE CODE THAM KHẢO 12](#_Toc502006248)

# MÔ TẢ ĐỒ ÁN

**Xây dựng hệ thống truy vấn hình ảnh (Image Retrieval System)**

* Github:
* Youtube:

**Mục tiêu, nội dung đồ án**

Tìm hiểu và áp dụng các phương pháp dùng BoW để xây dựn một hệ thống truy vấn hình ảnh. Hệ thống cung cấp một giao diện người dùng website. Tích hợp web service cung cấp các API để phục vụ cho việc truy vấn

Hệ thống sử dụng bộ dữ liệu train chuẩn là oxford5k (tham khảo <http://www.robots.ox.ac.uk/~vgg/data/oxbuildings/oxbuild_images.tgz> ). Vì lí do hạn chế phần cứng nên sẽ giới hạn 1000 bức ảnh train (random).

Ngoài chức năng retrieval bằng ảnh, hệ thống còn có tính năng cho phép người dùng lựa chọn vùng truy vấn (cropped), hoặc chọn loại đặc trưng sẽ dùng.

Các phương pháp được sử dụng: rút trích các đặc trưng như SIFT, SURF, ROOTSIFT, phương pháp gom cụm Kmeans và tính độ tương đồng bằng TF-IDF

Ví dụ,  tham khảo matlab: https://www.mathworks.com/help/vision/ug/image-retrieval-with-bag-of-visual-words.html.

# TỔ CHỨC THƯ MỤC

**Tập tin đồ án gồm các thư mục sau:**



* *Bao cao*: File báo cáo đồ án, phương pháp, kĩ thuật xây dựng hệ thống
* *Sources*: Chứa source code gồm
  + *web-client*: xây dựng web bằng ngôn ngữ NodeJS
  + *python-backend*: xây dựng một server REST API truy vấn ảnh bằng ngôn ngữ Python
  + *README.md* : Thông tin cài đặt và chạy thử hệ thống

**Hướng dẫn cài đặt hệ thống**

Môi trường cài đặt:

* Web:
  + NodeJS version 8.9.1
  + Cách build hệ thống:
    - cd web-client

Cài đặt packages

* + - npm install

Khởi tạo server (localhost:3000)

* + - npm run watch
* Server
  + Python: version 2.7.14
  + Cách build hệ thống:
    - cd python-backend

Cài đặt packages

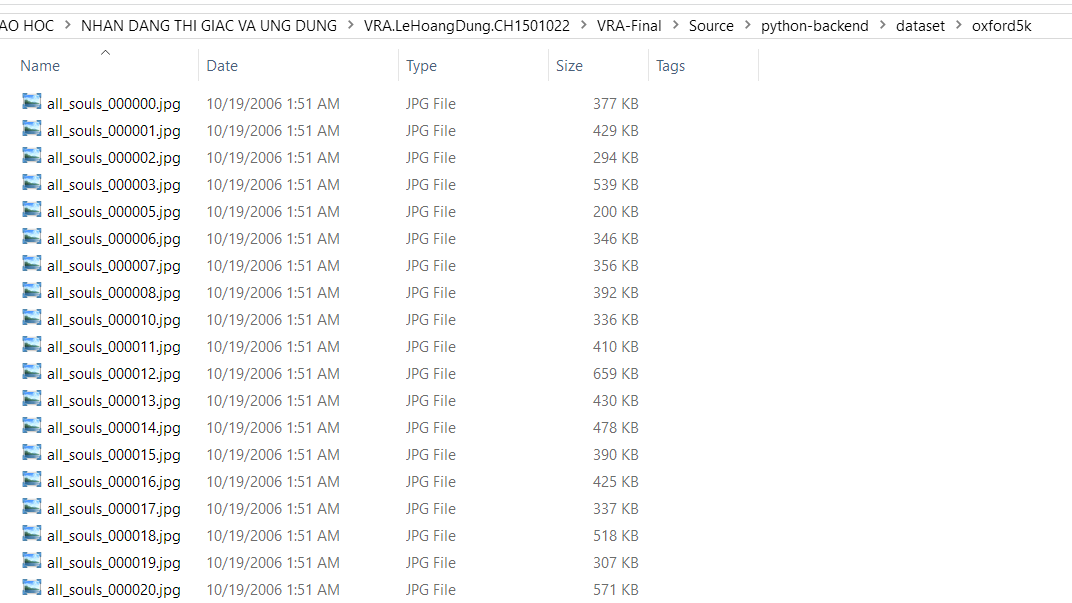
* + - python -m pip install -r requirements.txt

Khởi tạo data huấn luyện

* + - python index.py

Chạy server (localhost:5000)

* + - python server.py
* Thư mục dataset:
  + Copy ảnh bỏ vào thư mục: *python-backend/dataset/oxford5k/*



**Một số configuration trong hệ thống**

* Web:
  + File “config.json”:
    - **pyAPI.search**: link API đến server Python để truy vấn ảnh, giá trị default là <http://localhost:5000/search>
    - **pyAPI.upload**: link API đến server Python để upload ảnh, giá trị default là <http://localhost:5000/file-upload>
    - **externalUpload**: cho phép upload ảnh lên server Python, giá trị default là false
    - **searchByAjax**: cho phép gọi ajax thay vì submit ảnh lên server Python, giá trị default là true
* Server
  + File “config.py”:
    - **Settings.SEARCH\_THRESHOLD**: giá trị threshold để đánh giá ảnh trả về là phù hợp 🡪 trả về tất cả các ảnh có độ phù hợp (score) >= threshold
    - **Settings.MAX\_FILES**: số lượng file lớn nhất dùng để huấn luyện
    - **Settings.ROOT\_DATASET\_FOLDER**: thư mục lưu trữ ảnh huấn luyện, chứa các thư mục con là loại ảnh huấn luyện
    - **Settings.TRAIN\_DATASET**: tên thư mục loại ảnh được huấn luyện
    - **Settings.FEATURE**: loại đặc trưng dùng để rút trích
    - **Settings.FEATURE\_FILE**: tên file lưu trữ features
    - **KmeansTYPE**: phương pháp kmean sẽ dùng (type=1, 2, 3)
    - **Kmeans.NUM\_WORDS**: số lượng K clusters
    - **Kmeans.ITER**: số lần lặp Kmeans
    - **Features**: các loại đặc trưng sẽ dùng
    - **Resources**: chứa các cài đặt lên quan đến thư mục ảnh, phục vụ cho việc truy vấn đường dẫn file ảnh bằng các URL

# PHƯƠNG PHÁP XÂY DỰNG HỆ THỐNG

1. **Rút trích đặc trưng (file index.py)**

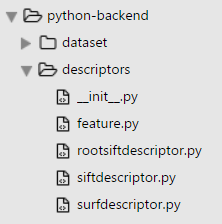
Ở đây nhóm chúng em sử dụng ngôn ngữ Python để cài đặt hệ thống truy vấn ảnh. Nhóm có sử dụng các thư viện của Python để hỗ trợ cho việc xử lý ảnh:

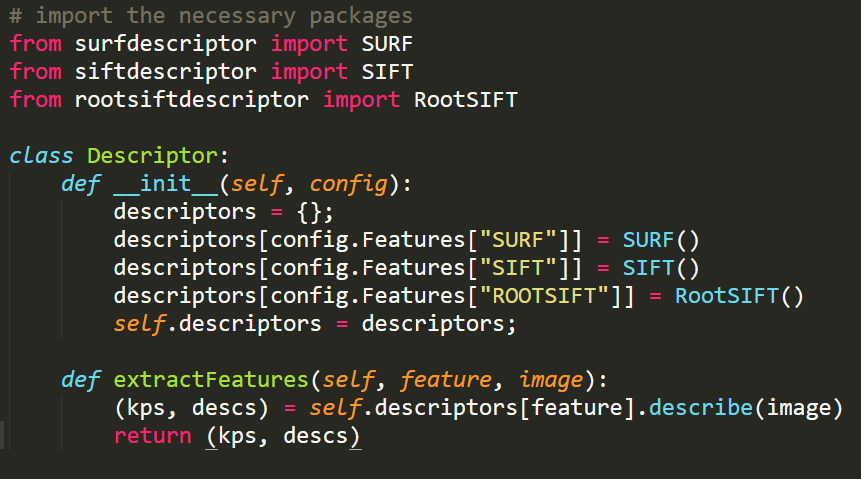
* + - * Numpy: chứa các hàm xử lý mảng, ma trận
      * Scipy: chứa các hàm xử lý ảnh, xử lý mảng, gom cụm
      * OpenCV: chứa các hàm xử lý ảnh, đọc ảnh, rút trích đặc trưng, hiển thị ảnh,..
      * Pillow: chứa các hàm để encode, lấy color,…
      * Sklearn: xử lý ảnh, gom cụm (clustering), đọc và lưu file

Quá trình huấn luyện dữ liệu train:

Nạp ảnh 🡪 Đọc và rút trích descriptors 🡪 clustering 🡪 tính TF-IDF 🡪 lưu kết quả tập tin

Nhóm sử dụng một số kĩ thuật rút trích đặc trưng như SIFT, SURF, ROOTSIFT. Các đặc trưng này được nhóm lựa chọn có sẵn trong thư viện của OpenCV và một số khác tham khảo từ trên internet. Các phương pháp được cài đặt trong thư mục *python-backend/descriptors*

**

**

Thực hiện train 500 ảnh với lần lượt các đặc trưng, nhóm nhận thấy ROOTSIFT có độ chính xác cao nhất trong các loại. SIFT và SURF cho kết quả gần giống nhau

Kết quả: ROOTSIFT > SIFT > SURF

Để thực hiện huấn luyện ảnh, chỉnh các cài đặt trong file config.py:

* **Settings.MAX\_FILES**: giới hạn kích thước ảnh được train
* **Settings.TRAIN\_DATASET**: thư mục con của thư mục datasetchứa ảnh train
* **Settings.FEATURE**: loại đặc trưng để train

Chay lệnh:

python index.py

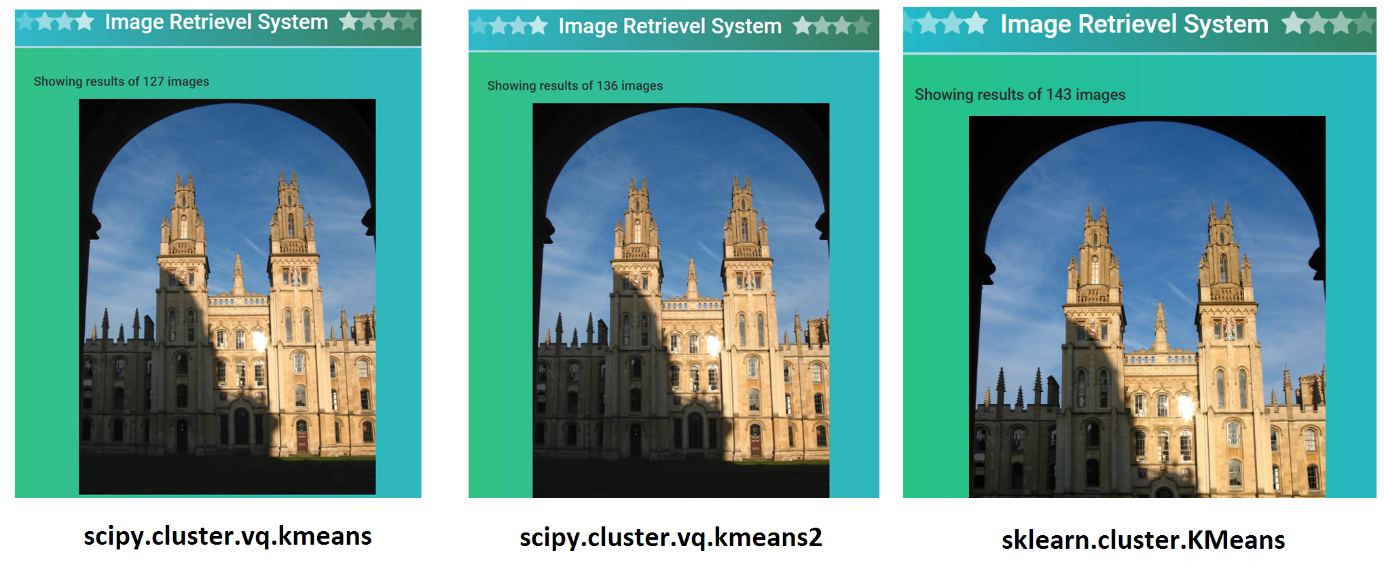
1. **Kĩ thuật gom cụm (Clustering) (file index.py)**

Nhóm sử dụng thư viện **sklearn** và **scipy** để phục vụ cho việc clustering. Bởi vì các thư viện này cung cấp phương pháp KMeans khác nhau. Thực hiện lần lượt các phương pháp và nhận thấy KMeans của thư viện **sklearn** cho thời gian khá lâu so với **scipy**, nhưng bù lại có độ chính xác cao hơn

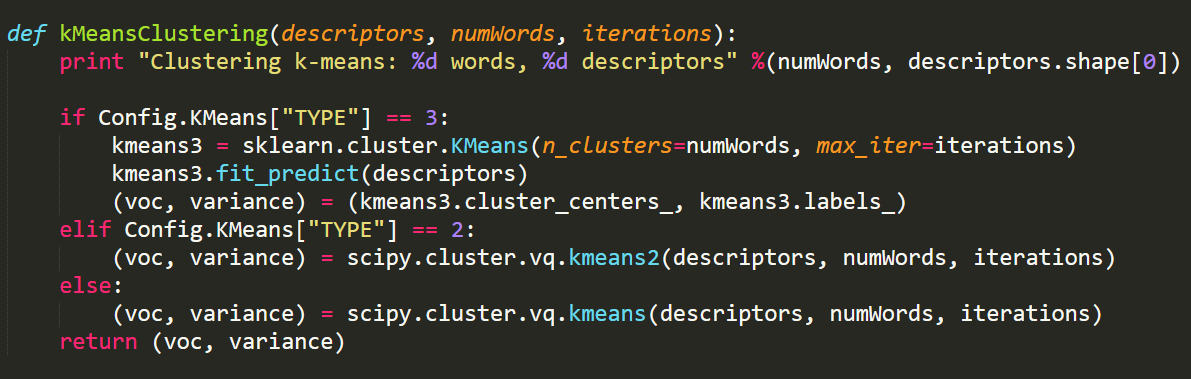
Ảnh query:



Kết quả của 3 phương pháp kmeans (K = 100, score threshold = 0.7)



Kết quả gom cụm ta sẽ thu được các center-clusters hay còn gọi là vocabularies



Một số cài đặt cho Kmeans

* **KMeans.TYPE**: phương pháp Kmeans được chọn(1, 2, 3)
* **KMeans.NUM\_WORDS**: số cụm (số K)
* **KMeans.ITER**: số lần lặp của thuật toán kmeans

1. **Tính khoảng cách TF-IDF (file index.py)**

Dựa vào kết quả bộ từ điển từ việc gom cụm, và danh sách các descriptors đã rút trích, nhóm tính toán khoảng cách vector dựa vào công thức TF-IDF và dạng chuẩn hóa L2



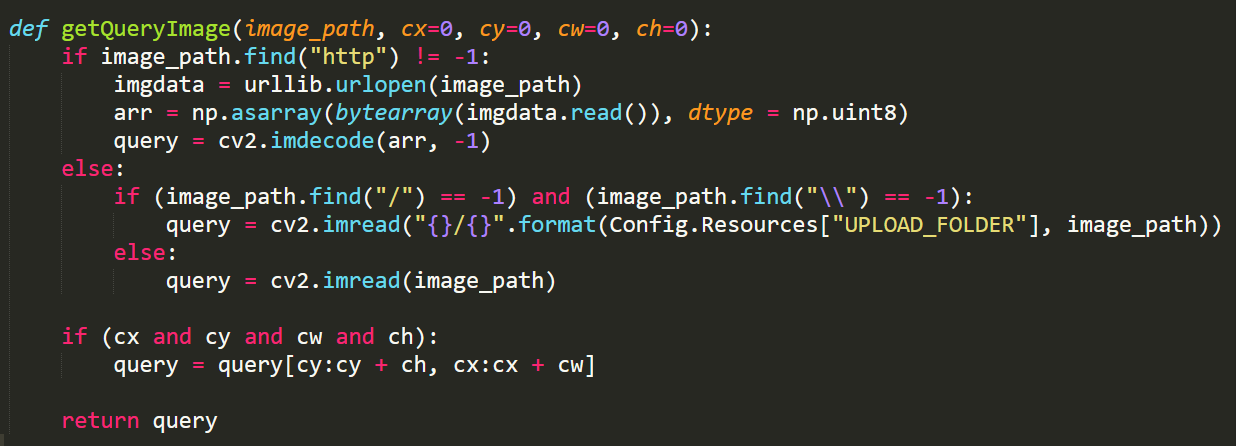
Kết quả được lưu trong tập tin:

*{tên thư mục train}\_{loại đặc trưng}\_ {loại kmean}\_features.bin*

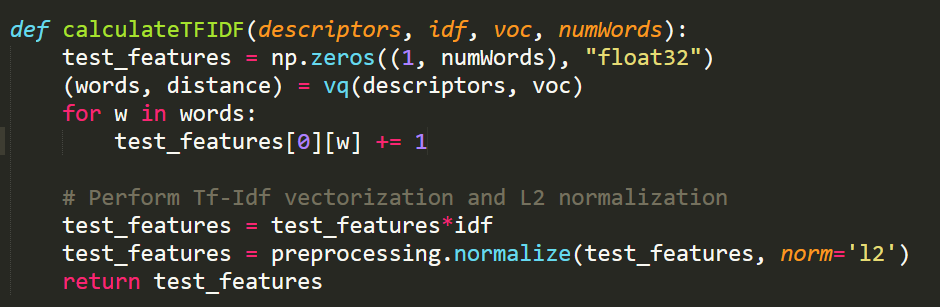
1. **Truy vấn ảnh (file search.py)**

Khi người dùng nhập ảnh query, ảnh sẽ được đọc và xử lý thông qua thư viện OpenCV.

Hàm **getQueryImage** có chức năng đọc ảnh và crop, nếu có đủ các tham số (x,y,w,h).

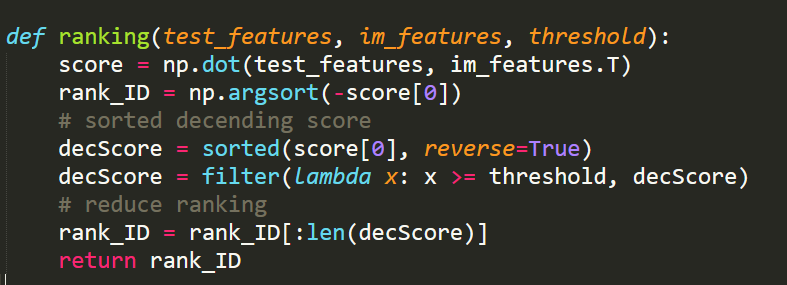


Tiếp đến ảnh query (hoặc crop) được rút trích đặc trưng và tính toán khoảng cách dựa vào giá trị TF-IDF và bộ từ điển lấy ra từ tập tin \*features.bin.

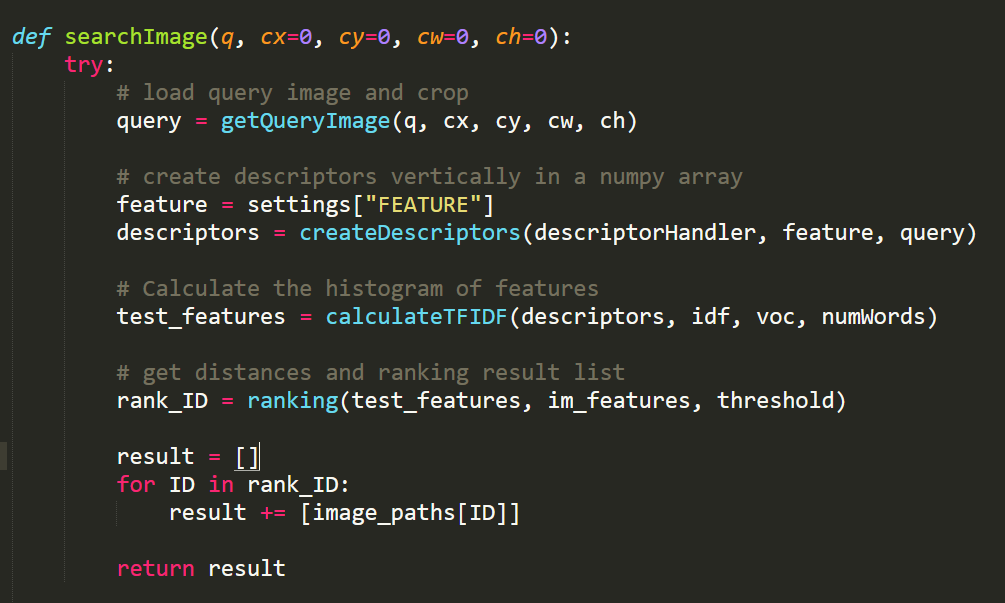


Sau đó ảnh trả về sẽ được lọc dựa trên giá trị THRESHOLD cho trước và sắp xếp (hàm **ranking**) theo thứ hạng từ cao đến thấp, sao cho ảnh gần giống query nhất sẽ xuất hiện ở vị trí đầu tiên

Để tính score, ta dựa vào biến **im\_features** chứa các đặc trưng rút ra từ tập train. Biến này có kích thước là (nRow, nCol) với nRow là số lượng ảnh train, nCol là số features rút trích. Nghĩa là mỗi dòng trong **im\_features** tương ứng với features của 1 ảnh. Tương tự **test\_features** là feature của ảnh query (chỉ có 1 dòng). Score được tính nhờ vào phép nhân ma trận **test\_features x invert(im\_features)**. Kết quả trả về là 1 danh sách có n ảnh result (n là tổng số ảnh train) mà mỗi phần tử có giá trị là độ tương đồng với ảnh query. Dựa vào danh sách này để trả về danh sách ảnh kết quả



Hàm **searchImage** gồm toàn bộ quy trình để tìm kiếm ảnh phù hợp câu query, tham số là query và các thông số crop ảnh (cx, cy, cw, ch)



1. **Xây dựng server (file server.py)**

Nhóm sử dụng thư viện Flask của Python và các thư viện con (Flask-Cors, Flask-Jsonpify, Flask-RESTful, Flask-Uploads) của nó để xây dựng một server cung cấp API để truy vấn ảnh

Việc cài đặt server được thực hiện trong tập tin server.py. Server được chạy trên localhost: 5000

Chạy lệnh:

python server.py

1. **Xây dựng website**

Website được cài đặt bằng NodeJS với các plugin jQuery (Jcrop) để crop ảnh.

Để kết nối đến server Python, cần cung cấp các cài đặt trong file config.json

* + **pyAPI.search**: link API đến server Python để truy vấn ảnh, giá trị default là <http://localhost:5000/search>
  + **pyAPI.upload**: link API đến server Python để upload ảnh, giá trị default là <http://localhost:5000/file-upload>
  + **externalUpload**: cho phép upload ảnh lên server Python, giá trị default là false
  + **searchByAjax**: cho phép gọi ajax thay vì submit ảnh lên server Python, giá trị default là true

Chạy lệnh (localhost: 3000):

npm run watch

1. **Demo chương trình**

# TÀI LIỆU THAM KHẢO

1. Slide bài giảng môn Tìm kiếm thông tin thị giác
2. Slide bài giảng môn Nhận dạng thông tin thi giác
3. TF-IDF and L2-norm.

Link: <http://blog.christianperone.com/2011/10/machine-learning-text-feature-extraction-tf-idf-part-ii/>

1. Implementing RootSIFT in Python and OpenCV.

Link: <https://www.pyimagesearch.com/2015/04/13/implementing-rootsift-in-python-and-opencv/>

1. K Means Algorithm. Link: <http://stanford.edu/~cpiech/cs221/handouts/kmeans.html>
2. Jcrop jQuery (crop ảnh). Link: <https://github.com/tapmodo/Jcrop>
3. B-lazy.js jQuery (tích hợp lazy load cho ảnh kết quả). Link: <http://dinbror.dk/blazy/>
4. Dropzone jQuery (cung cấp giao diện upload ảnh). Link: <http://www.dropzonejs.com/>
5. <http://zeus.robots.ox.ac.uk/oxfordbuildings/>

# MỘT SỐ SOURCE CODE THAM KHẢO

1. <https://www.pyimagesearch.com/2014/12/01/complete-guide-building-image-search-engine-python-opencv/>
2. <https://github.com/devashishp/Content-Based-Image-Retrieval>
3. <https://gist.github.com/anabranch/48c5c0124ba4e162b2e3>