**Học viện Công nghệ Bưu chính Viễn thông**

**KHOA CÔNG NGHỆ KĨ THUẬT ĐIỆN – ĐIỆN TỬ**

**~~~~~**  **~~~~~**

****

**Báo cáo môn học**

Thị Giác Máy Tính

**Đề tài:**

**NHẬN DIỆN KHU DU TÍCH, ĐIỂM THAM QUAN**

**Giảng viên hướng dẫn** : **P.V. Sự**

**Sinh viên thực hiện** : B18DCDT034 : Nguyễn Hoàng Dương

B18DCDT011 : Nguyễn Việt Anh

B18DCDT042 : Hứa Sỹ Đạo

Hà Nội, 05/2022

**Mục Lục**

[**1.** **Chương 1 : Giới thiệu** 5](#_Toc102672053)

[**1.1.** **Lý do chọn đề tài** 5](#_Toc102672054)

[**1.2.** **Mục tiêu đề tài** 5](#_Toc102672055)

[**1.3.** **Các vấn đề cần giải quyết** 5](#_Toc102672056)

[**1.3.1.1.** **Tiền xử lý dữ liệu** 5](#_Toc102672057)

[**1.3.1.2.** **Xây dựng model ảnh** 5](#_Toc102672058)

[**1.3.1.3.** **Xây dựng chương trình nhận diện** 5](#_Toc102672059)

[**1.4.** **Phạm vi nghiên cứu** 5](#_Toc102672060)

[**2.** **Chương 2: Cơ sở lý thuyết** 6](#_Toc102672064)

[**2.1.** **Tiền xử lý dữ liệu** 6](#_Toc102672065)

[**2.2.** **Xây dựng model ảnh dựa trên mạng nơ ron tích chập (CNN)** 6](#_Toc102672066)

[**2.2.1.** **Lớp tích chập (Convolution Layer)** 6](#_Toc102672069)

[**2.2.2.** **ReLU activation** 9](#_Toc102672071)

[**2.2.3.** **Max Pooling** 10](#_Toc102672072)

[**2.2.4.** **Fully Connected Layer** 11](#_Toc102672073)

[**2.2.5.** **Tổng hợp** 12](#_Toc102672074)

[**3.** **Chương 3 : Thực hiện đề tài** 13](#_Toc102672075)

[**3.1.** **Tiền xử lý dữ liệu** 13](#_Toc102672076)

[**3.2.** **Xây dựng model ảnh** 16](#_Toc102672077)

[**3.3.** **Xây dựng chương trình nhận diện ảnh** 18](#_Toc102672078)

[**4.** **DEMO** 19](#_Toc102672079)

[**5.** **Tài liệu tham khảo** 20](#_Toc102672080)

1. **Chương 1 : Giới thiệu**
   1. **Lý do chọn đề tài**

Với sự phát triển không ngừng của công tác khảo cổ học, số lượng các di tích văn hóa ngày càng nhiều. Làm thế nào để quản lý một cách hiệu quả một số lượng lớn các di tích văn hóa đã trở thành trọng tâm của những suy nghĩ và nghiên cứu hiện nay. Để giải quyết vấn đề này, công nghệ thông tin hiện đại ra đời. Với sự tiến bộ của trí tuệ nhân tạo và công nghệ xử lý ảnh mang lại nguồn tài liệu tham khảo phong phú cho việc ghi nhận ảnh di tích văn hóa.

Một trong những phương pháp phổ biến nhất trong phát hiện và nhận dạng ảnh hiện nay chính là trích xuất các đặc trưng của ảnh sau đó sử dụng mạng nơ-ron tích chập và các thuật toán khác để phân loại các đối tượng này. Tuy nhiên, để nâng cao độ chính xác của việc phân loại đang là vấn đề của nghiên cứu hiện nay. Vì vậy, trên cơ sở những kiến ​​thức lý thuyết nói trên và thực hành ứng dụng phát hiện và nhận dạng ảnh, một phương pháp phát hiện và nhận biết di tích văn hóa bảo tàng dựa trên mạng nơ ron tích chập sẽ được nhóm trình bày trong bài viết dưới đây.

* 1. **Mục tiêu đề tài**

Dựa trên những bức ảnh được đưa đến đầu vào ( input ), chúng ta sẽ phân loại và điền tên cho bức ảnh đó thuộc khu di tích hay khu tham quan nào.

* 1. **Các vấn đề cần giải quyết**
     + 1. **Tiền xử lý dữ liệu**

Chuẩn bị data trên 5 địa điểm khác nhau ở Việt Nam

Mỗi địa điểm chuẩn bị khoảng 150 – 200 bức ảnh khác nhau

* + - 1. **Xây dựng model ảnh**

Sử dụng mạng Nơ ron tích chập (CNN) để trích xuất đặc trưng ảnh, sau đó xây dựng model ảnh để phân loại

* + - 1. **Xây dựng chương trình nhận diện**

Viết chương trình nhận diện các khu di tích, khu tham quan dựa vào ảnh được đưa vào

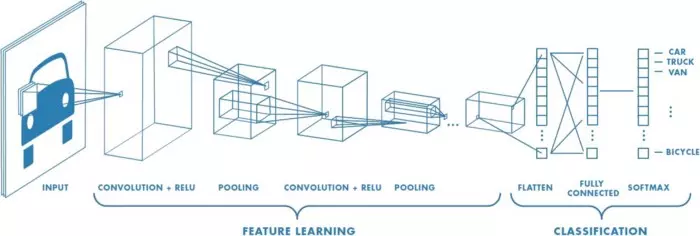
* 1. **Phạm vi nghiên cứu**
* Kiến thức : cần nắm được lý thuyết về mạng nơ ron tích chập (CNN)
* Ngôn ngữ sử dụng : Python
* Các modules hỗ trợ : ***opencv, tensorflow, keras, numpy***, os, selenium, beautifulSoup, requests, …

1. **Chương 2: Cơ sở lý thuyết**
   1. **Tiền xử lý dữ liệu**

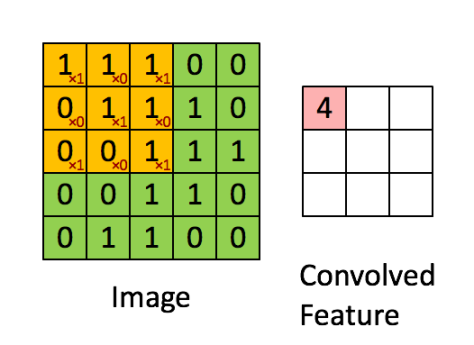
1. Sử dụng thư viện selenium và beautifulSoup để crawl ảnh từ google về

2. Quản lý data : đổi tên, đổi kích thước ảnh cho đồng bộ

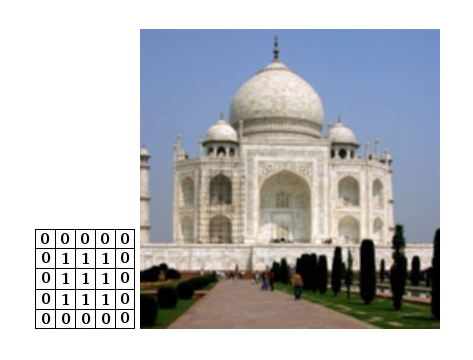
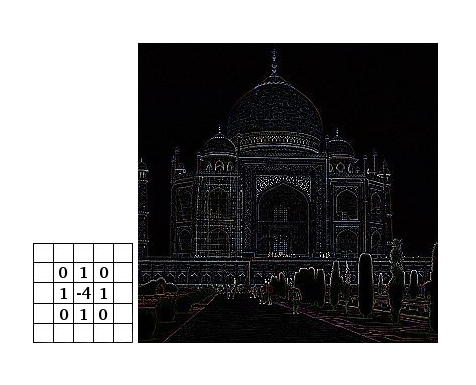
* 1. **Xây dựng model ảnh dựa trên mạng nơ ron tích chập (CNN)**
* Trong mạng neural, mô hình mạng neural tích chập (CNN) là 1 trong những mô hình để nhận dạng và phân loại hình ảnh
* Về kỹ thuật, mô hình CNN để training và kiểm tra, mỗi hình ảnh đầu vào sẽ chuyển nó qua 1 loạt các lớp tích chập với các bộ lọc (Kernals), tổng hợp lại các lớp được kết nối đầy đủ (Full Connected) và áp dụng hàm Softmax để phân loại đối tượng có giá trị xác suất giữa 0 và 1. Hình dưới đây là toàn bộ luồng CNN để xử lý hình ảnh đầu vào và phân loại các đối tượng dựa trên giá trị.



* + 1. **Lớp tích chập (Convolution Layer)**
* Các convolutional layer có các parameter(kernel) đã được học để tự điều chỉnh lấy ra những thông tin chính xác nhất mà không cần chọn các feature.
* Convolution hay tích chập là nhân từng phần tử trong ma trận Sliding Window hay còn gọi là kernel, filter hoặc feature detect là một ma trận vuông kích thước nhỏ (3x3, 5x5,…) như trong ví dụ trên là 3×3.
* Convolution hay tích chập là nhân từng phần tử bên trong kernel với ma trận ảnh đầu vào. Kết quả được một ma trận gọi là Convoled feature được sinh ra từ việc nhận ma trận Kernel với ma trận ảnh đầu vào.

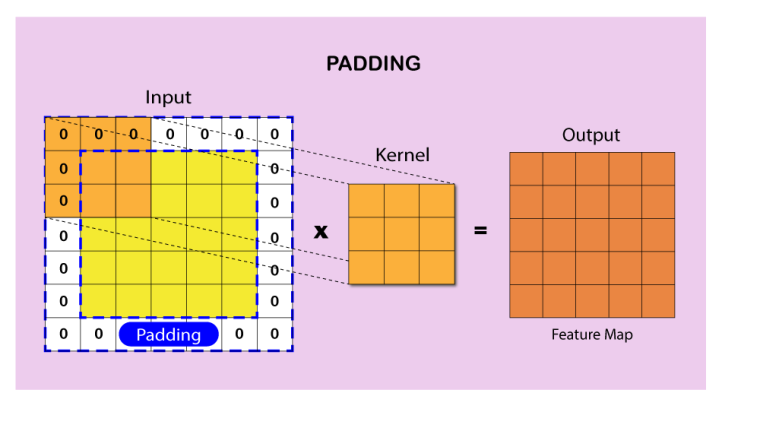


[*file:///E:/Python/Relics\_Detection/giphy.webp*](file:///E:/Python/Relics_Detection/giphy.webp)

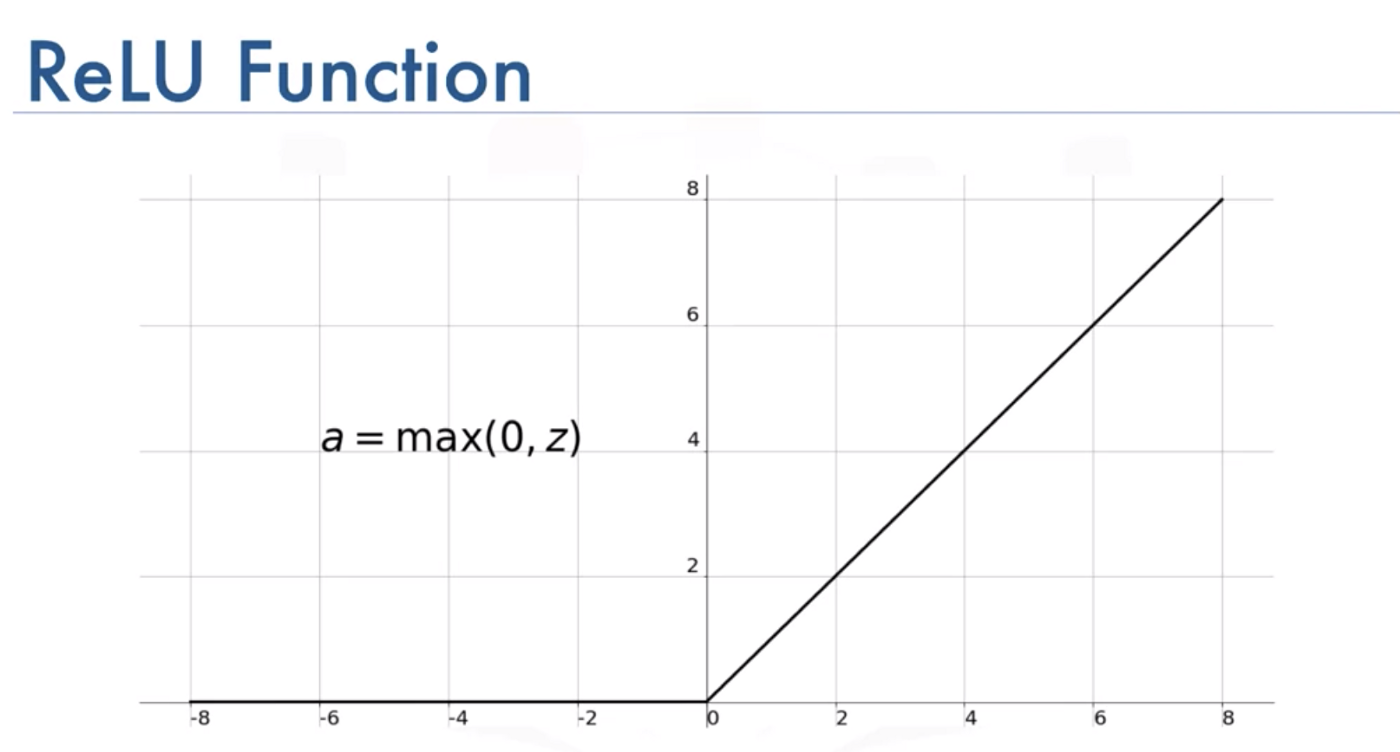


* Padding

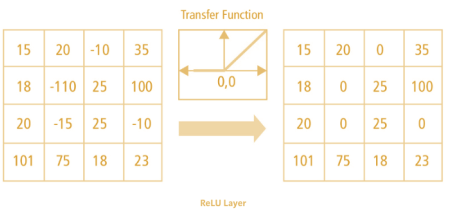
Vì khi bảng Kernel chập vào ảnh thường các điểm ảnh ở biên sẽ chỉ được chập 1 lần, nên Padding sẽ mở rộng khung ảnh ra sao cho các điểm ảnh đều được chập như nhau nhằm tránh mất mát thông tin ảnh



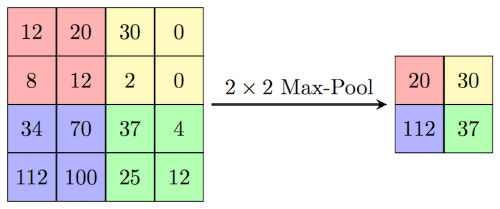
* + 1. **ReLU activation**



* ReLU viết tắt của Rectified Linear Unit, là 1 hàm phi tuyến. Với đầu ra là: ƒ (x) = max (0, x).
* Sau khi hình ảnh được đưa qua Convolution Layer, đôi khi sẽ có những điểm ảnh có giá trị âm bởi thực hiện các phép toán với bảng Kernel, đối với 1 tấm hình thì không nên có các giá trị âm, hoặc có giá trị âm là không đúng, cho nên activation ReLU sẽ cắt bỏ các giá trị này, chỉ giữ lại các giá trị dương

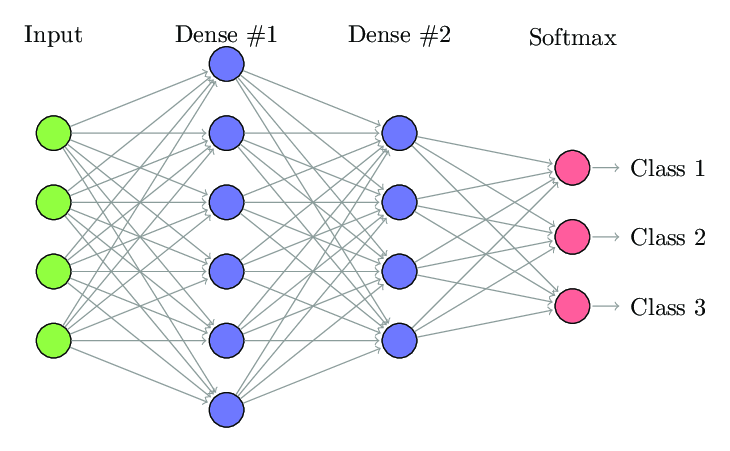


* + 1. **Max Pooling**
* Max Pooling Layer sẽ giảm bớt số lượng tham số khi hình ảnh quá lớn. Không gian pooling còn được gọi là lấy mẫu con hoặc lấy mẫu xuống làm giảm kích thước của mỗi map nhưng vẫn giữ lại thông tin quan trọng.
* Làm cho các ảnh đầu vào (feature dimension) nhỏ hơn từ đó dễ quản lý hơn
* Làm cho mạng luôn bất biến đối với các biến đổi nhỏ, biến dạng và dịch ảnh trong hình ảnh đầu vào (một biến dạng nhỏ trong đầu vào sẽ không thay đổi đầu ra của Pooling - vì lấy giá trị lớn nhất trong vùng lân cận cục bộ).

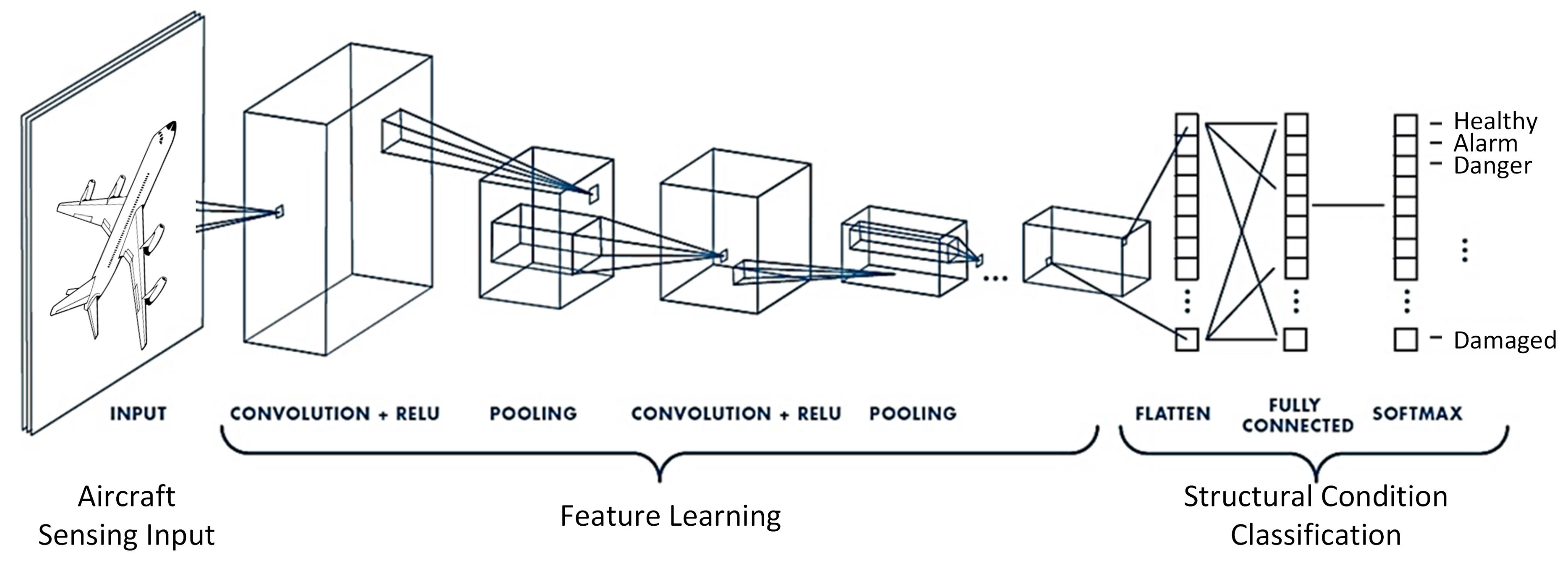


Max Pooling là lấy phần từ lớn nhất trong ma trận vuông (2x2, 3x3, 4x4, …) mà nó quét được từ đầu đến cuối bức ảnh

* + 1. **Fully Connected Layer**
* Fully Connected Layer là một Multi Layer Perceptron sử dụng chức năng kích hoạt softmax trong lớp đầu ra. Thuật ngữ “Fully Connected” ngụ ý rằng mọi nơ-ron ở lớp trước được kết nối với mọi nơ-ron ở lớp tiếp theo.
* Đầu ra từ các Convolution Layer và Max Pooling Layer trích ra các đặc trưng cao của hình ảnh đầu vào. Mục đích của lớp Fully Connected là sử dụng các đặc trưng này để phân loại hình ảnh đầu vào thành các lớp khác nhau dựa trên tập dữ liệu đào tạo.



* Từ các dữ liệu ảnh được đưa qua các lớp Convolution Layer, ReLU Layer, Max Pooling Layer, chúng ta đưa qua Flatten Function ( Làm phẳng ) trải rộng các dữ liệu trên ra 1 chiều duy nhất, mỗi một ma trận ảnh ta sẽ gán cho nó 1 label là một dãy bit chúng ta tự quy định để phân loại cho nó.
  + VD : Chó (100), Mèo (010), Chuột (001), …
* Dữ liệu Flatten sẽ kết nối với tất cả các Nodes mà chúng ta tạo ra, sau cùng sẽ thông qua hàm Softmax để phân loại ảnh.
* Hàm SoftMax có tác dụng là tạo ra kết quả là 1 xác suất để có thể dùng dự đoán.
  + 1. **Tổng hợp**

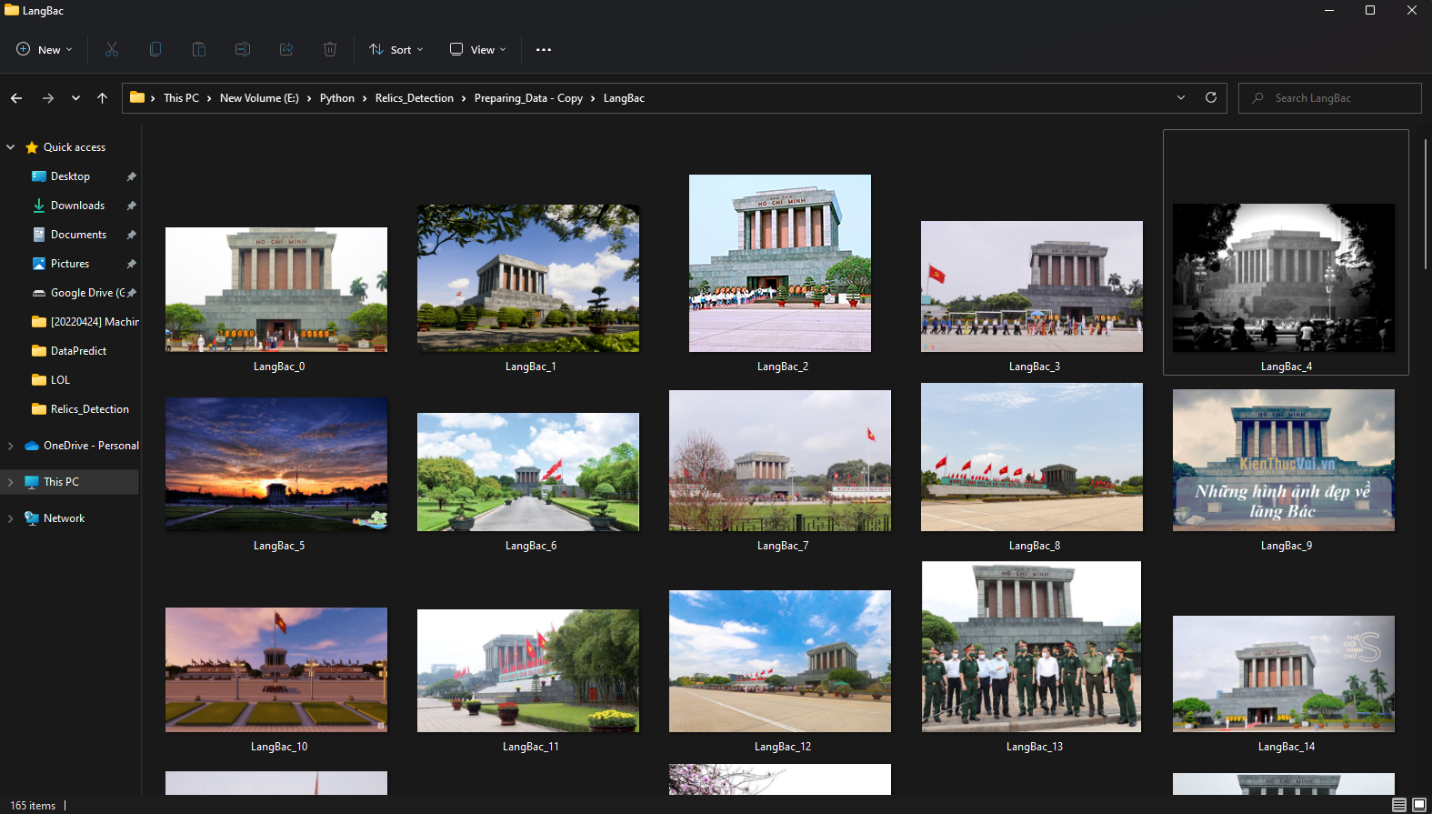


1. **Chương 3 : Thực hiện đề tài** 
   1. **Tiền xử lý dữ liệu**

* Mục đích : Thu thập các dữ liệu ảnh của các địa danh.
* Ở bài này, nhóm em sẽ lấy các dữ liệu từ trên google về máy tính.
* B1 : Chạy code sau:

import bs4  
import requests  
from selenium import webdriver  
import os  
import time  
from selenium.webdriver.common.by import By  
  
search\_URL = "https://www.google.com/search?q=nha+tho+lon+ha+noi&sxsrf=ALiCzsaxQOScrSi0y3HcnSbWE0lWGuEwkQ:165166355372" \  
 "9&source=lnms&tbm=isch&sa=X&ved=2ahUKEwjF-qGb3sX3AhX6yosBHTOADBYQ\_AUoAXoECAIQAw&biw=1920&bih=929&dpr=1"  
  
#creating a directory to save images  
folder\_name = 'E:\\Python\\Relics\_Detection\\Preparing\_Data\\NhaThoHaNoi'  
if not os.path.isdir(folder\_name):  
 os.makedirs(folder\_name)  
  
def download\_image(url, folder\_name, num):  
  
 reponse = requests.get(url)  
 if reponse.status\_code==200:  
 with open(os.path.join(folder\_name, str(num)+".jpg"), 'wb') as file:  
 file.write(reponse.content)  
  
chromePath=r'E:\\Download\\Chrome\\chromedriver\_win32\\chromedriver.exe'  
driver=webdriver.Chrome(chromePath)  
  
driver.get(search\_URL)  
  
a = input("Waiting...")  
  
driver.execute\_script("window.scrollTo(0, 0);")  
  
page\_html = driver.page\_source  
pageSoup = bs4.BeautifulSoup(page\_html, 'html.parser')  
containers = pageSoup.findAll('div', {'class':"isv-r PNCib MSM1fd BUooTd"} )  
  
print(len(containers))  
  
len\_containers = len(containers)  
  
for i in range(1, len\_containers+1):  
 if i % 25 == 0:  
 continue  
  
 xPath = """//\*[@id="islrg"]/div[1]/div[%s]"""%(i)  
 previewImageXPath = """//\*[@id="islrg"]/div[1]/div[%s]/a[1]/div[1]/img"""%(i)  
 previewImageElement = driver.find\_element(By.XPATH, previewImageXPath)  
 previewImageURL = previewImageElement.get\_attribute("src")  
  
 driver.find\_element(By.XPATH, xPath).click()  
  
 timeStarted = time.time()  
  
 while True:  
 #//\*[@id="Sva75c"]/div/div/div[3]/div[2]/c-wiz/div/div[1]/div[1]/div[3]/div/a/img  
 imageElement = driver.find\_element(By.XPATH, """//\*[@id="Sva75c"]/div/div/div[3]/div[2]/c-wiz/div/div[1]/div[1]/div[3]/div/a/img""")  
 imageURL= imageElement.get\_attribute('src')  
  
 if imageURL != previewImageURL:  
 #print("actual URL", imageURL)  
 break  
  
 else:  
 #making a timeout if the full res image can't be loaded  
 currentTime = time.time()  
  
 if currentTime - timeStarted > 10:  
 print("Timeout! Will download a lower resolution image and move onto the next one")  
 break  
 #Downloading image  
 try:  
 download\_image(imageURL, folder\_name, i)  
 print("Downloaded element %s out of %s total. URL: %s" % (i, len\_containers + 1, imageURL))  
 except:  
 print("Couldn't download an image %s, continuing downloading the next one"%(i))

* B2 : Sau khi chạy code trên chúng ta sẽ crawl được các bức ảnh trên Google về



* B3 : Đổi size các bức ảnh về cùng 1 size, đổi tên thư mục, tên files cho dễ quản lý
* B4 : Lặp lại các bước trên để lấy thêm dữ liệu từ các địa danh khác.
  1. **Xây dựng model ảnh**
* Mục đích : Xây dựng một model nhận diện ảnh được trainning từ các data đã chuẩn bị (tương tự như các models cifar10, cifar100,… sẵn có hiện nay).

import numpy as np  
import os  
from PIL import Image  
  
from keras import layers  
from keras import models  
  
train\_path = 'E:\\Python\\Relics\_Detection\\Preparing\_Data\\TrainData'  
test\_path = 'E:\\Python\\Relics\_Detection\\Preparing\_Data\\TestData'  
  
xtrain = []  
ytrain = []  
  
xtest = []  
ytest = []  
  
dict = {'ChuaMotCot': [1, 0, 0, 0, 0], 'LangBac': [0, 1, 0, 0, 0],  
 'MuCangChai': [0, 0, 1, 0, 0], 'NhaThoHaNoi': [0, 0, 0, 1, 0],  
 'VinhHaLong': [0, 0, 0, 0, 1]}  
  
  
def getData(i\_path, lstData):  
 for folder in os.listdir(i\_path):  
 file = i\_path + '\\' + folder  
 lst\_filename = []  
 for data in os.listdir(file):  
 filename = i\_path + '\\' + folder + '\\' + data  
 labels = filename.split('\\')[5]  
 # print(labels)  
 img = np.array(Image.open(filename))  
 lst\_filename.append((img, dict[labels]))  
  
 lstData.extend(lst\_filename)  
 return lstData

xtrain = getData(train\_path, xtrain) # Ma trận ảnh vào  
xtest = getData(test\_path, xtest)# Các Labels gán cho bức ảnh đó  
  
# print(xtrain)  
np.random.shuffle(xtrain)  
np.random.shuffle(xtrain)  
# # print(xtrain[100])  
#  
models\_training\_first = models.Sequential([  
 layers.Conv2D(32, (3, 3), input\_shape=(512, 512, 3), activation='relu'),  
 layers.MaxPool2D((2, 2)),  
 layers.Dropout(0.15),  
  
 layers.Conv2D(64, (3, 3), activation='relu'),  
 layers.MaxPool2D((2, 2)),  
 layers.Dropout(0.15),  
  
 layers.Conv2D(128, (3, 3), activation='relu'),  
 layers.MaxPool2D((2, 2)),  
 layers.Dropout(0.15),  
  
 layers.Flatten(),  
 layers.Dense(1000, activation='relu'),  
 layers.Dense(256, activation='relu'),  
 layers.Dense(5, activation='softmax')  
])  
# models\_training\_first.summary()  
  
models\_training\_first.compile(optimizer='adam',  
 loss='categorical\_crossentropy',  
 metrics=['accuracy']  
 )  
#  
models\_training\_first.fit(np.array([x[0] for \_, x in enumerate(xtrain)]), np.array([y[1] for \_, y in enumerate(xtrain)]), epochs=10)  
models\_training\_first.save('model\_5relics\_10epochs.h5')

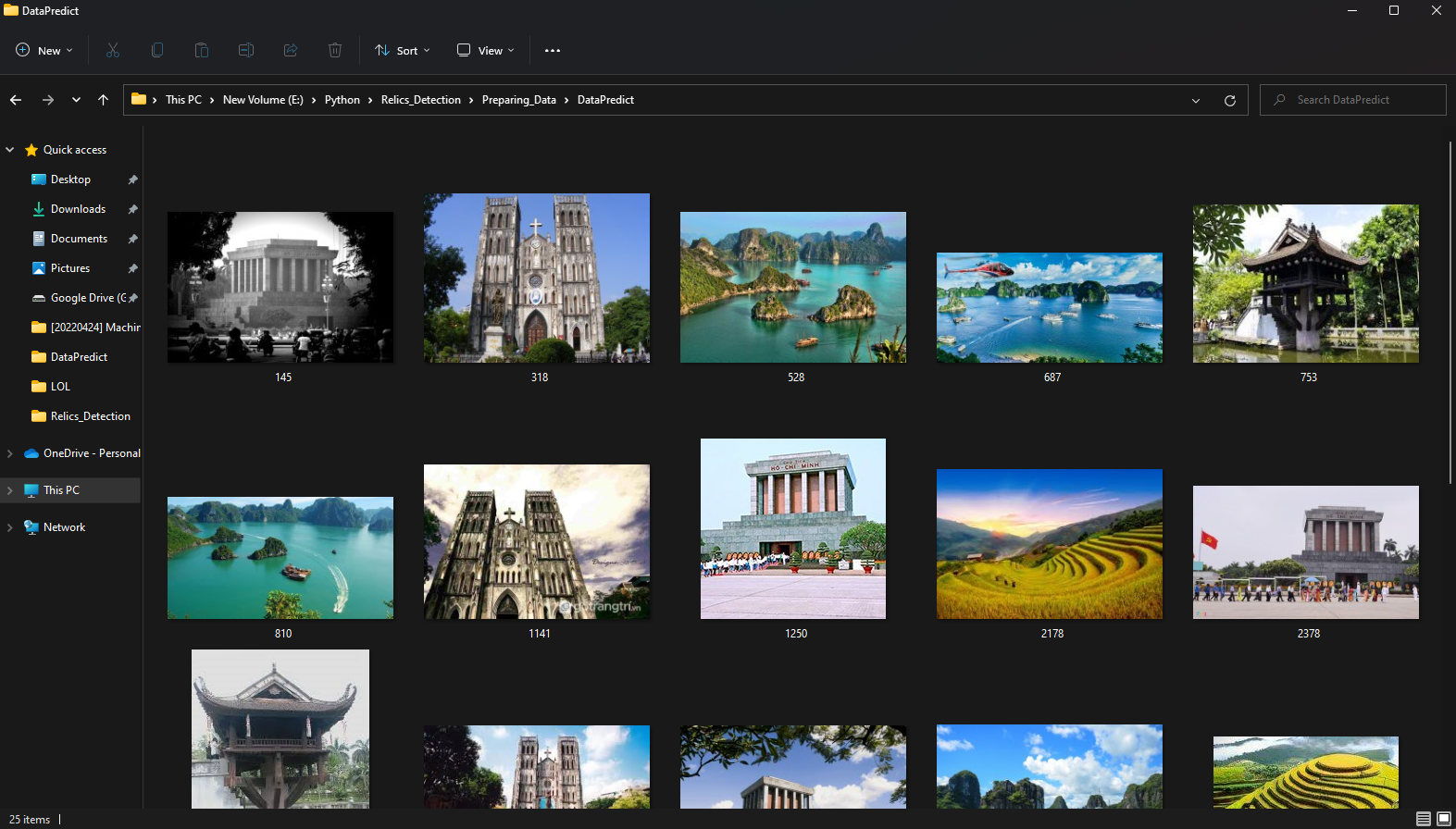
* Ứng với mỗi địa danh, chúng ta gán một label theo kỹ thuật one hot encoding khác nhau
* Các ảnh đầu vào sẽ qua các lớp Convolution, ReLU và MaxPooling sau đó được làm phẳng Flatten rồi kết nối với các nodes ở Hidden Layer
* Cuối cùng được kết nối với 5 nodes (tương ứng với 5 địa danh) với activation softmax để có được kết quả là các xác suất dự đoán.
* Lưu models training định dạng : “Tên.h5”
* Models có thể có dung lượng rất lớn
  1. **Xây dựng chương trình nhận diện ảnh**
* Mục đích : dự đoán các địa danh từ ảnh được đưa vào

import cv2  
import numpy as np  
import os  
import matplotlib.pyplot as plt  
from keras import models  
  
lstResult = ['Chua mot Cot', 'Lang bac', 'Mu cang chai', 'Nha tho ha noi', 'Vinh ha long']  
models = models.load\_model('model\_5relics\_10epochs.h5')  
  
i\_path = "E:\\Python\\Relics\_Detection\\Preparing\_Data\\DataPredict"  
x = 1  
plt.figure(figsize=(10, 10))  
for i in os.listdir(i\_path):  
 img = cv2.imread(i\_path + '\\' + i)  
 img1 = plt.imread(i\_path + '\\' + i)  
 print(i\_path + '\\' + i)  
 roi = cv2.resize(img, (512, 512))  
 result = np.argmax(models.predict(roi.reshape((-1, 512, 512, 3))))  
 plt.subplot(5, 5, x)  
 plt.imshow(img1)  
 plt.title(lstResult[result])  
 plt.axis('off')  
 x+=1  
plt.savefig("Result.jpg")  
plt.show()

* i\_path : Thư mục chứa các ảnh cần dự đoán
* lstResult : Tên các địa danh theo thứ tự one hot coding từ trước

1. **DEMO**

* Chuẩn bị ảnh vào 1 folder



* Kết quả sau khi chạy chương trình nhận diện ảnh

1. **Tài liệu tham khảo**

*1.* [*https://www.mathworks.com/discovery/convolutional-neural-network-matlab.html*](https://www.mathworks.com/discovery/convolutional-neural-network-matlab.html)

*2.* [*https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/*](https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/)

*3.* [*https://datawow.io/blogs/interns-explain-cnn-8a669d053f8b*](https://datawow.io/blogs/interns-explain-cnn-8a669d053f8b)

*4.* [*https://www.superdatascience.com/blogs/convolutional-neural-networks-cnn-softmax-crossentropy*](https://www.superdatascience.com/blogs/convolutional-neural-networks-cnn-softmax-crossentropy)