NHẬN DẠNG HÌNH ẢNH BẰNG GOOGLE VISION API 1

[**I. Phát hiện nhãn 1**](#_gjdgxs)

[1. Giới thiệu phát hiện nhãn 1](#_9lg6necxvltz)

[2. Phương pháp phát hiện nhãn 1](#_973vg9q4eop)

[3. Hướng dẫn chi tiết mã nguồn 3](#_9mdz7073j8fu)

[3.1. Import các thư viện cần thiết và cài đặt thông tin xác thực cho Google Cloud 3](#_h66flj8deg3t)

[3.2. Xây dựng hàm nhận diện 3](#_8yqz0jyg07sq)

[3.3. Sử dụng hàm 4](#_x6i272rj6wxd)

[4. Chi Tiết mã nguồn 4](#_3m2hb522ju9k)

[**II. Phát hiện văn bản 5**](#_30j0zll)

[1. Giới thiệu văn bản 5](#_gnxjy9y5vw89)

[2. Phương pháp phát hiện văn bản 5](#_bslp1s4xmoyq)

[3. Hướng dẫn chi tiết mã nguồn 7](#_g0e5bybyt13l)

[3.1. Import các thư viện cần thiết và cài đặt thông tin xác thực cho Google Cloud 7](#_c2m1bw9251dx)

[3.2. Xây dựng hàm nhận diện 7](#_90y7tlamvux5)

[3.3. Sử dụng hàm 8](#_ak4ff3be40nr)

[4. Chi Tiết mã nguồn 8](#_bel0xu9fu5oh)

[**III. Kết hợp thành chương trình chạy bằng câu lệnh cmd 9**](#_7gvs48r2ow3l)

# Phát hiện nhãn

## Giới thiệu phát hiện nhãn

Phát hiện nhãn là một tính năng phát hiện rộng các danh mục từ hình ảnh đến phương tiện vận chuyển đến động vật. Khi phát hiện danh mục được thực hiện, hình ảnh được phân tích và thông tin được nhận ra trong hình ảnh được dán nhãn và xuất ra. Ví dụ, trong trường hợp hình ảnh của một tòa nhà, nhiều thông tin khác nhau được hiển thị, chẳng hạn như tòa nhà nào được đặt và những gì xung quanh tòa nhà.

## Phương pháp phát hiện nhãn

Tải xuống mẫu sử dụng Vision để yêu cầu thông tin như phát hiện nhãn từ hình ảnh..

| git clone https://github.com/GoogleCloudPlatform/python-docs-samples/ |
| --- |

Thực hiện phát hiện danh mục trên ảnh bằng Vision. Trong thiết bị đầu cuối, điều hướng đến thư mục chứa mã có liên quan.

| cd python-docs-samples/vision/snippets/detect |
| --- |

Cài đặt các thư viện khác nhau là cần thiết để chạy mã nguồn chương trình phát hiện nhãn bằng lệnh sau:

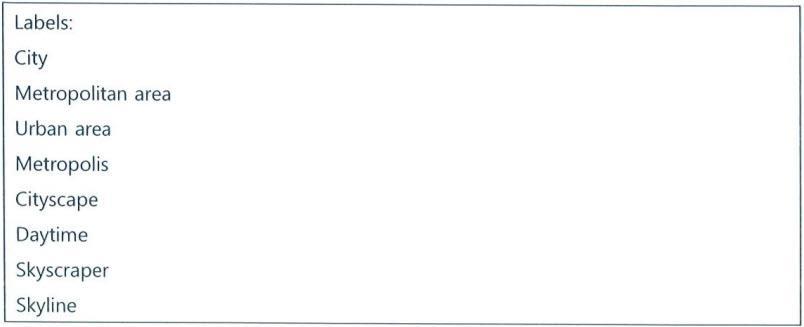
| sudo pip install -r requirements.txt |
| --- |

Tận dụng các hình ảnh nằm trong thư mục tài nguyên gốc để chạy mã code và thực hiện yêu cầu phát hiện nhãn hình ảnh đầu tiên.

| python detect.py labels .resources/label.jpg |
| --- |



Khi lệnh được thực hiện, có thể thấy kết quả sau. Hiển thị nhãn của các nội dung phát hiện được trong hình ảnh:





## Hướng dẫn chi tiết mã nguồn

### 3.1. Import các thư viện cần thiết và cài đặt thông tin xác thực cho Google Cloud

* Cần import os để cho phép tương tác với hệ điều hành
* Cài đặt biến môi trường **'GOOGLE\_APPLICATION\_CREDENTIALS'** thành đường dẫn tới tệp JSON chứa thông tin xác thực dịch vụ (Service account credentials) cho Google Cloud. Tệp này được sử dụng để xác thực ứng dụng khi truy cập API Google Cloud Vision.

| import os  os.environ['GOOGLE\_APPLICATION\_CREDENTIALS'] = 'ServiceAccToken.JSON' |
| --- |

### 3.2. Xây dựng hàm nhận diện

* Hàm này nhận một đối số duy nhất, đó là đường dẫn đến tệp hình ảnh cần phân tích.

| def detect\_labels(path): |
| --- |

* Bước 1: khởi tạo client Google cloud vision: Cần import module vision từ gói google.cloud và khởi tạo một client cho API Google Cloud Vision.

| from google.cloud import vision  client = vision.ImageAnnotatorClient() |
| --- |

* Bước 2: Đọc tệp hình ảnh:

| with open(path, "rb") as image\_file:  content = image\_file.read() |
| --- |

* Bước 3: Tạo đối tượng vision.Image (Biến image sẽ được gán bởi đối tượng vision.Image với nội dung hình ảnh được lấy từ tệp)

| image = vision.Image(content=content) |
| --- |

* Bước 4: Gửi hình ảnh để phát hiện nhãn (Hình ảnh sẽ được gửi đến API Vision để phát hiện nhãn và lưu trữ phản hồi trong biến response)

| response = client.label\_detection(image=image) |
| --- |

* Bước 5: Trích xuất các nhãn từ phản hồi (Các nhãn được phát hiện từ phản hồi API sẽ được lưu trong biến labels)

| labels = response.label\_annotations |
| --- |

* Bước 6: In các nhãn đã phát hiện

| print("Labels:")  for label in labels:  print(' - ', label.description) |
| --- |

* Bước 7: Xử lý lỗi

Ở bước này, chương trình sẽ kiểm tra xem phản hồi từ API Vision có chứa lỗi hay không. Nếu có thì nghĩa là yêu cầu không thành công hoặc có lỗi xảy ra trong quá trình xử lý hình ảnh. Cần tạo ra một ngoại lệ và ném ngoại lệ đó với thông báo lỗi từ phản hồi.

| if response.error.message:  raise Exception(  "{}\nFor more info on error messages, check: "  "https://cloud.google.com/apis/design/errors".format(response.error.message)  ) |
| --- |

### 3.3. Sử dụng hàm

Biến img\_name sẽ là tên tệp hình ảnh và xây dựng đường dẫn đường dẫn đầy đủ đường dẫn đến tệp bằng biến file\_path. Sau đó sẽ gọi hàm detect\_labels với đường truyền dẫn tệp hình ảnh.

| img\_name = 'logos.jpg'  file\_path = f'./resources/{img\_name}'  detect\_labels(file\_path) |
| --- |

## Chi Tiết mã nguồn

| import os  os.environ['GOOGLE\_APPLICATION\_CREDENTIALS'] = 'ServiceAccToken.JSON'  def detect\_labels(path):  """Detects labels in the file."""  from google.cloud import vision  client = vision.ImageAnnotatorClient()  # [START vision\_python\_migration\_label\_detection]  with open(path, "rb") as image\_file:  content = image\_file.read()  image = vision.Image(content=content)  response = client.label\_detection(image=image)  labels = response.label\_annotations    print("Labels:")  for label in labels:  print(' - ',label.description)  if response.error.message:  raise Exception(  "{}\nFor more info on error messages, check: "  "https://cloud.google.com/apis/design/errors".format(response.error.message)  )  img\_name = 'logos.jpg'  file\_path = f'./resources/{img\_name}'  detect\_labels(file\_path) |
| --- |

Mã nguồn chương trình để phát hiện nhãn từ hình ảnh trong thư mục là như sau. Khởi tạo ứng dụng khách bằng cách nhập Thư viện Google Cloud Client. Tải tệp hình ảnh vào bộ nhớ và thực hiện phát hiện nhãn trên tệp hình ảnh.

# II. Phát hiện văn bản

## Giới thiệu văn bản

Phát hiện văn bản là một chức năng để thực hiện nhận dạng ký tự quang học. Nó phát hiện và trích xuất văn bản trong hình ảnh và hỗ trợ nhiều ngôn ngữ. Nhận dạng ngôn ngữ tự động cũng được cung cấp.

## Phương pháp phát hiện văn bản

Tải xuống mẫu sử dụng Vision để yêu cầu thông tin như phát hiện nhãn từ hình ảnh..

| git clone https://github.com/GoogleCloudPlatform/python-docs-samples/ |
| --- |

Thực hiện phát hiện danh mục trên ảnh bằng Vision. Trong thiết bị đầu cuối, điều hướng đến thư mục chứa mã có liên quan.

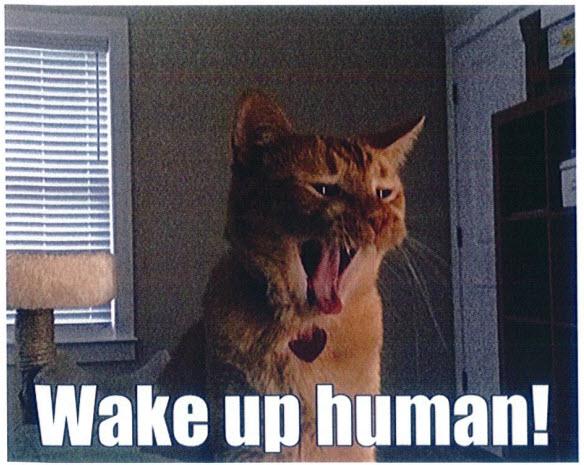
| cd python-docs-samples/vision/snippets/detect |
| --- |

Cài đặt các thư viện khác nhau là cần thiết để chạy mã nguồn chương trình phát hiện nhãn bằng lệnh sau:

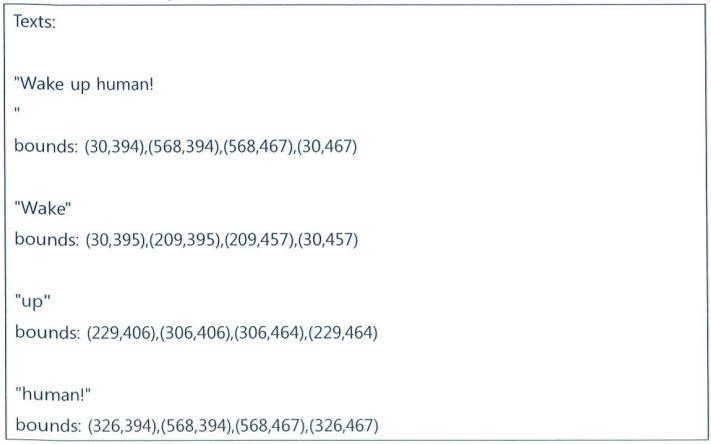
| sudo pip install -r requirements.txt |
| --- |

Tận dụng các hình ảnh nằm trong thư mục tài nguyên gốc để chạy mã code và thực hiện yêu cầu phát hiện nhãn hình ảnh đầu tiên.

| python detect.py texts .resources/label.jpg |
| --- |



Khi lệnh được thực hiện, có thể thấy kết quả sau. Hiển thị văn bản được phát hiện và phạm vi kích thước của văn bản trong hình ảnh.



Mã nguồn chương trình để phát hiện văn bản từ hình ảnh trong thư mục là như sau. Khởi tạo ứng dụng khách bằng cách nhập Thư viện Google Cloud Client. Tải tệp hình ảnh vào bộ nhớ và thực hiện phát hiện văn bản trên tệp hình ảnh.

## Hướng dẫn chi tiết mã nguồn

### 3.1. Import các thư viện cần thiết và cài đặt thông tin xác thực cho Google Cloud

* Cần import os để cho phép tương tác với hệ điều hành
* Cài đặt biến môi trường **'GOOGLE\_APPLICATION\_CREDENTIALS'** thành đường dẫn tới tệp JSON chứa thông tin xác thực dịch vụ (Service account credentials) cho Google Cloud. Tệp này được sử dụng để xác thực ứng dụng khi truy cập API Google Cloud Vision.

| import os  os.environ['GOOGLE\_APPLICATION\_CREDENTIALS'] = 'ServiceAccToken.JSON' |
| --- |

### 3.2. Xây dựng hàm nhận diện

* Hàm này nhận một đối số duy nhất, đó là đường dẫn đến tệp hình ảnh cần phân tích.

| def detect\_labels(path): |
| --- |

* Bước 1: khởi tạo client Google cloud vision: Cần import module vision từ gói google.cloud và khởi tạo một client cho API Google Cloud Vision.

| from google.cloud import vision  client = vision.ImageAnnotatorClient() |
| --- |

* Bước 2: Đọc tệp hình ảnh:

| with open(path, "rb") as image\_file:  content = image\_file.read() |
| --- |

* Bước 3: Tạo đối tượng vision.Image (Biến image sẽ được gán bởi đối tượng vision.Image với nội dung hình ảnh được lấy từ tệp)

| image = vision.Image(content=content) |
| --- |

* Bước 4: Gửi hình ảnh để phát hiện nhãn (Hình ảnh sẽ được gửi đến API Vision để phát hiện nhãn và lưu trữ phản hồi trong biến response)

| response = client.label\_detection(image=image) |
| --- |

* Bước 5: Trích xuất văn bản từ phản hồi (Các văn bản được phát hiện từ phản hồi API sẽ được lưu trong biến texts)

| texts = response.text\_annotations |
| --- |

* Bước 6: In các nhãn đã phát hiện

Trong đó:

* print(f'\n"{text.description}"'): In ra nội dung của đoạn văn bản đã nhận dạng.
* vertices = [...]: Lấy thông tin về vị trí của đoạn văn bản trên hình ảnh và lưu vào biến vertices.
* print("bounds: {}".format(",".join(vertices)): In ra thông tin về vị trí của văn bản trên hình ảnh.

| for text in texts:  print(f'\n"{text.description}"')  vertices = [  f"({vertex.x},{vertex.y})" for vertex in text.bounding\_poly.vertices  ]  print("bounds: {}".format(",".join(vertices))) |
| --- |

* Bước 7: Xử lý lỗi

Ở bước này, chương trình sẽ kiểm tra xem phản hồi từ API Vision có chứa lỗi hay không. Nếu có thì nghĩa là yêu cầu không thành công hoặc có lỗi xảy ra trong quá trình xử lý hình ảnh. Cần tạo ra một ngoại lệ và ném ngoại lệ đó với thông báo lỗi từ phản hồi.

| if response.error.message:  raise Exception(  "{}\nFor more info on error messages, check: "  "https://cloud.google.com/apis/design/errors".format(response.error.message)  ) |
| --- |

### 3.3. Sử dụng hàm

Biến img\_name sẽ là tên tệp hình ảnh và xây dựng đường dẫn đường dẫn đầy đủ đường dẫn đến tệp bằng biến file\_path. Sau đó sẽ gọi hàm detect\_labels với đường truyền dẫn tệp hình ảnh.

| img\_name = 'logos.jpg'  file\_path = f'./resources/{img\_name}'  detect\_labels(file\_path) |
| --- |

## 

## Chi Tiết mã nguồn

| import os  os.environ['GOOGLE\_APPLICATION\_CREDENTIALS'] = 'ServiceAccToken.JSON'  def detect\_texts(path):  from google.cloud import vision  client = vision.ImageAnnotatorClient()  with open(path, "rb") as image\_file:  content = image\_file.read()  image = vision.Image(content=content)  response = client.text\_detection(image=image)  texts = response.text\_annotations  print("Texts:")  for text in texts:  print(f'\n"{text.description}"')  vertices = [  f"({vertex.x},{vertex.y})" for vertex in text.bounding\_poly.vertices  ]  print("bounds: {}".format(",".join(vertices)))  if response.error.message:  raise Exception(  "{}\nFor more info on error messages, check: "  "https://cloud.google.com/apis/design/errors".format(response.error.message)  )  img\_name = 'logos.jpg'  file\_path = f'./resources/{img\_name}'  detect\_texts(file\_path) |
| --- |

Mã nguồn chương trình để phát hiện nhãn từ hình ảnh trong thư mục là như sau. Khởi tạo ứng dụng khách bằng cách nhập Thư viện Google Cloud Client. Tải tệp hình ảnh vào bộ nhớ và thực hiện phát hiện nhãn trên tệp hình ảnh.

# III. Kết hợp thành chương trình chạy bằng câu lệnh cmd

| import sys  import os  os.environ['GOOGLE\_APPLICATION\_CREDENTIALS'] = 'ServiceAccToken.JSON'  def detect\_labels(path):  """Detects labels in the file."""  from google.cloud import vision  client = vision.ImageAnnotatorClient()  # [START vision\_python\_migration\_label\_detection]  with open(path, "rb") as image\_file:  content = image\_file.read()  image = vision.Image(content=content)  response = client.label\_detection(image=image)  labels = response.label\_annotations  print("Labels:")  for label in labels:  print(' - ', label.description)  if response.error.message:  raise Exception(  "{}\nFor more info on error messages, check: "  "https://cloud.google.com/apis/design/errors".format(  response.error.message)  )  def detect\_texts(path):  from google.cloud import vision  client = vision.ImageAnnotatorClient()  with open(path, "rb") as image\_file:  content = image\_file.read()  image = vision.Image(content=content)  response = client.text\_detection(image=image)  texts = response.text\_annotations  print("Texts:")  for text in texts:  print(f'\n"{text.description}"')  vertices = [  f"({vertex.x},{vertex.y})" for vertex in text.bounding\_poly.vertices  ]  print("bounds: {}".format(",".join(vertices)))  if response.error.message:  raise Exception(  "{}\nFor more info on error messages, check: "  "https://cloud.google.com/apis/design/errors".format(  response.error.message)  )  if len(sys.argv) != 3:  print("Sử dụng: python Detect.py [Text|Label] [img\_path]")  else:  option = sys.argv[1]  img\_path = sys.argv[2]  if option == "Text":  detect\_texts(img\_path)  elif option == "Label":  detect\_labels(img\_path)  else:  print("Không rõ lựa chọn:", option) |
| --- |

Giải thích Đoạn mã nhận câu lệnh từ cmd:

* Bước 1: Kiểm tra đối số dòng lệnh (Command Line Arguments)

Đoạn mã bắt đầu kiểm tra số lượng đối số được truyền vào từ dòng lệnh command line. Python cung cấp biến sys.argv để truy cập các đối số này. Điều này kiểm tra xem dòng lệnh truyền vào đã đúng hay chưa.

| if len(sys.argv) != 3: |
| --- |

* Bước 2: Thông báo sử dụng nếu đối số không hợp lệ

Nếu số lượng đối số không đúng (nghĩa là khác 3), chương trình sẽ in ra một thông báo hướng dẫn về cách sử dụng chương trình. Nó yêu cầu cung cấp 2 đối số: loại (**Text** hay **Label**) và đường dẫn của hình ảnh **img\_path**

| else:  option = sys.argv[1]  img\_path = sys.argv[2] |
| --- |

* Bước 3: Lấy giá trị của đối số từ dòng lệnh

Nếu số lượng đối số là 3, chương trình sẽ lấy giá trị của đối số thứ nhất (**sys.argv[1]**) và gán cho biến **option**, và lấy giá trị của đối số thứ hai (**sys.argv[2]**) và gán cho biến **img\_path**.

| else:  option = sys.argv[1]  img\_path = sys.argv[2] |
| --- |

* Bước 4: Xử lý dựa trên lựa chọn

Sau khi lấy giá trị của option, chương trình kiểm tra giá trị của option để quyết định loại công việc nào cần thực hiện. Nếu option là "**Text**," nó gọi hàm **detect\_texts(img\_path)** để phát hiện văn bản trong hình ảnh. Nếu option là "**Label**," nó gọi hàm **detect\_labels(img\_path)** để phát hiện nhãn (**label**) trong hình ảnh.

| if option == "Text":  detect\_texts(img\_path)  elif option == "Label":  detect\_labels(img\_path) |
| --- |

* Bước 5: Thông báo nếu lựa chọn không rõ ràng

Nếu option không phải "Text" hoặc "Label," chương trình sẽ in ra thông báo cho biết lựa chọn không rõ ràng.

| else:  print("Không rõ lựa chọn:", option) |
| --- |

Cách chạy câu lệnh trên cmd: **python Detect.py [Text|Label] [img\_path]**

Ví dụ: python Detect.py Text ./resources/text.jpg

