

CNN Applications

Visual Understanding AI

using vision cloud service API

AI - 영상 이해 서비스

- Google Vision API : [Cloud Vision API Demo](#)
 - MicroSoft Vision API : [MS Azure Vision API Demo](#)
 - Amazon Rekognition API : [AWS Rekognition API Demo](#) ⇐ (requires to log-in)
-
- API (응용 프로그램 인터페이스: Application Program Interface) : 소프트웨어 응용 프로그램 구축을 위한 일련의 루틴, 프로토콜 및 도구 집합들을 지칭합니다. 기본적으로 API는 소프트웨어 **구성 요소의 상호 작용 방식**을 지정합니다. 좋은 API를 사용하면 프로그램을 쉽게 개발할 수 있습니다.

Google Vision AI (Link : <https://cloud.google.com/vision/>)

- 사이트를 방문하여 이미지 파일을 상자로 드래그하거나, 컴퓨터 파일을 업로드하십시오.

AI & Machine Learning Products

영업팀에 문의

VISION AI

Try the API



Drag image file here or
Browse from your computer



Faces

Objects

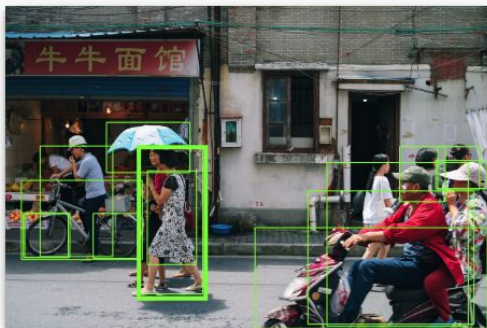
Labels

Web

Text

Properties

Safe Search



test.jpeg

Person	88%
Person	87%
Bicycle wheel	86%
Person	85%
Bicycle	84%
Bicycle wheel	81%
Tire	80%
Person	80%

- 선행 학습된 Google 모델을 사용하여 이미지에 라벨을 할당하고 수백만 개의 사전 정의된 카테고리 분류합니다. 객체 및 얼굴 감지, 인쇄 및 필기 텍스트 읽기 등이 가능합니다.

- sample image

Google Vision AI (Link : <https://cloud.google.com/vision/>)

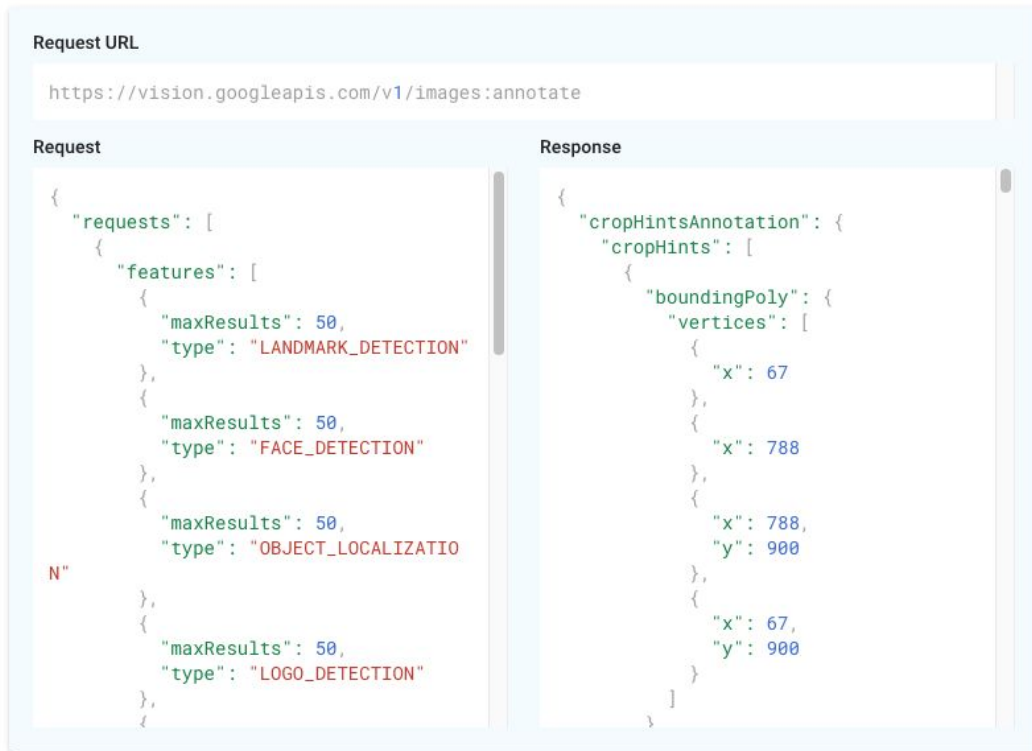
요청(request)과 응답(response)

형식: JSON

(JavaScript Object Notation)

Sample JSON Parser

- <https://jsonformatter.org/json-parser>
- <https://jsonparser.org/>



The screenshot shows a web-based JSON parser interface. At the top, there is a 'Request URL' field containing the URL `https://vision.googleapis.com/v1/images:annotate`. Below this, the interface is split into two panels: 'Request' and 'Response'. The 'Request' panel displays a JSON object with three requests: 'LANDMARK_DETECTION', 'FACE_DETECTION', and 'OBJECT_LOCALIZATION'. The 'Response' panel displays a JSON object with 'cropHintsAnnotation' containing a 'boundingPoly' with vertices at (67, 67), (788, 67), (788, 900), and (67, 900).

```
Request URL
https://vision.googleapis.com/v1/images:annotate

Request
{
  "requests": [
    {
      "features": [
        {
          "maxResults": 50,
          "type": "LANDMARK_DETECTION"
        },
        {
          "maxResults": 50,
          "type": "FACE_DETECTION"
        },
        {
          "maxResults": 50,
          "type": "OBJECT_LOCALIZATION"
        }
      ]
    }
  ]
}

Response
{
  "cropHintsAnnotation": {
    "cropHints": [
      {
        "boundingPoly": {
          "vertices": [
            {
              "x": 67,
              "y": 67
            },
            {
              "x": 788,
              "y": 67
            },
            {
              "x": 788,
              "y": 900
            },
            {
              "x": 67,
              "y": 900
            }
          ]
        }
      }
    ]
  }
}
```

지원 인식 기능

- 객체 감지 :
- 인쇄 및 필기 입력 텍스트 감지 :
- 얼굴 감지 :
- 명소 및 제품 로고 식별 :
- 일반 이미지 속성 할당 :
- 웹 항목 및 페이지 감지 :
- 콘텐츠 검토 :

The image displays a collage of screenshots from a computer vision analysis interface, showing various results for an image of a street scene in Shanghai (labeled 'shanghai.jpg').

Top Left Screenshot (Objects Tab): Shows the main image with green bounding boxes around detected objects. A sidebar on the right lists detected objects and their confidence scores:

- Person: 88%
- Person: 87%
- Bicycle wheel: 86%
- Person: 85%
- Bicycle: 84%
- Bicycle wheel: 81%
- Tire: 80%
- Person: 80%

Top Middle Screenshot (Text Tab): Shows the main image with green bounding boxes around detected text. A sidebar on the right lists detected text blocks and their confidence scores:

- +Page 1: 100%
- +Block 1: 100%
- +Block 2: 100%
- +Block 3: 100%

Bottom Left Screenshot (Safe Search Tab): Shows the main image with green bounding boxes around detected faces. A sidebar on the right lists detected faces and their confidence scores:

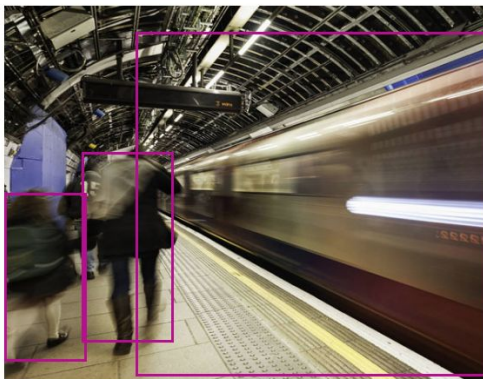
- Adult: Very Unlikely
- Spoof: Very Unlikely
- Medical: Very Unlikely
- Violence: Unlikely
- Racy: Unlikely

Bottom Right Screenshot (Labels Tab): Shows the main image with green bounding boxes around detected labels. A sidebar on the right lists detected labels and their confidence scores:

- People: 94%
- Street: 89%
- Mode Of Transport: 87%
- Vehicle: 84%
- Transport: 84%
- Snapshot: 83%
- Urban Area: 80%
- Infrastructure: 73%

MicroSoft Vision AI (Link : [MS Vision API Demo](#))

- 사이트를 방문하여 이미지 URL 입력하거나, 찾아보기로 업로드합니다.



기능 이름:	값
개체	[{ "rectangle": { "x": 93, "y": 178, "w": 115, "h": 237 }, "object": "person", "confidence": 0.764 }, { "rectangle": { "x": 0, "y": 229, "w": 101, "h": 206 }, "object": "person", "confidence": 0.624 }, { "rectangle": { "x": 161, "y": 31, "w": 439, "h": 423 }, "object": "subway train", "parent": { "object": "train", "parent": { "object": "Land vehicle", "parent": { "object": "Vehicle", "confidence": 0.926, "confidence": 0.923 }, "confidence": 0.917 }, "confidence": 0.801 } }
태그	[{ "name": "train", "confidence": 0.9975446 }, { "name": "platform", "confidence": 0.9955431 }, { "name": "station", "confidence": 0.979800761 }, { "name": "indoor", "confidence": 0.9277198 }, { "name": "subway", "confidence": 0.8389395 }, { "name": "clothing", "confidence": 0.5043765 }, { "name": "pulling", "confidence": 0.4317162 }]

이미지 URL

재출

찾아보기



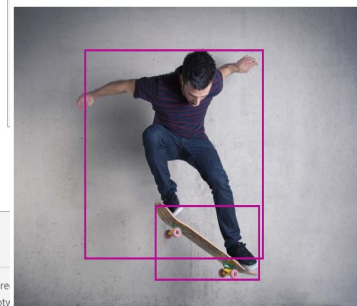
- 시각적 데이터를 사용하여 콘텐츠(개체에서 개념까지)에 레이블을 지정하고, 인쇄된 텍스트와 필기 텍스트를 추출하고, 브랜드와 랜드마크 같은 친숙한 주제를 인식하고, 콘텐츠를 조정하는게 가능합니다.

지원 인식 기능

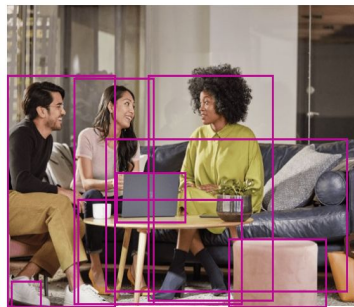
- 콘텐츠 태그
- 개체 감지
- 이미지 분류
- 이미지 설명
- 얼굴 감지
- 이미지 형식 감지
- 도메인 특성 콘텐츠
- 색 구성표 감지
- 스마트 썸네일
- 인쇄 및 필기 텍스트 인식
- 성인 콘텐츠 검색



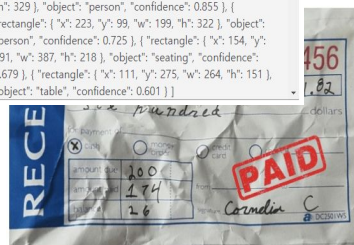
기능 이름:	값
개체	[{"rectangle": {"x": 34, "y": 283, "w": 73, "h": 62}, "object": "Tableware", "confidence": 0.521}, {"rectangle": {"x": 106, "y": 10, "w": 210, "h": 219}, "object": "plant", "confidence": 0.642}]
태그	[{"name": "table", "confidence": 0.994518042}, {"name": "laptop", "confidence": 0.999999999}]



기능 이름:	값
개체	[{"rectangle": {"x": 238, "y": 299, "w": 177, "h": 117}, "object": "Skateboard", "confidence": 0.903}, {"rectangle": {"x": 118, "y": 63, "w": 305, "h": 321}, "object": "person", "confidence": 0.955}]
태그	[{"name": "skating", "confidence": 0.999951541}, {"name": "snowboarding", "confidence": 0.990067363}, {"name": "sports equipment", "confidence": 0.9774853}, {"name": "person", "confidence": 0.9605776}, {"name": "roller skating", "confidence": 0.945730746}, {"name": "boardsport", "confidence": 0.9242261}, {"name": "man", "confidence": 0.9188208}, {"name": "outdoor", "confidence": 0.9107821}, {"name": "riding", "confidence": 0.900007248}, {"name": "skiing", "confidence": 0.894337356}, {"name": "footwear", "confidence": 0.894337356}]



기능 이름:	값
개체	[{"rectangle": {"x": 104, "y": 147, "w": 147, "h": 147}, "object": "person", "confidence": 0.763}, {"rectangle": {"x": 174, "y": 236, "w": 113, "h": 74}, "object": "Laptop", "parent": {"object": "computer", "confidence": 0.56}, {"confidence": 0.553}, {"rectangle": {"x": 351, "y": 331, "w": 154, "h": 99}, "object": "seating", "confidence": 0.525}, {"rectangle": {"x": 0, "y": 101, "w": 174, "h": 329}, "object": "person", "confidence": 0.855}, {"rectangle": {"x": 223, "y": 99, "w": 199, "h": 322}, "object": "person", "confidence": 0.725}, {"rectangle": {"x": 154, "y": 191, "w": 387, "h": 218}, "object": "seating", "confidence": 0.679}, {"rectangle": {"x": 111, "y": 275, "w": 264, "h": 151}, "object": "table", "confidence": 0.601}]



기능 이름:	값
개체	[{"rectangle": {"x": 86, "y": 56, "w": 467, "h": 343}, "object": "Office supplies", "confidence": 0.577}]
태그	[{"name": "text", "confidence": 0.9999335}, {"name": "handwriting", "confidence": 0.9937743}]
설명	{ "tags": ["text"], "captions": [{ "text": "a close up of text on a white background", "confidence": 0.8143099 }] }
이미지 형식	"jpeg"
이미지 크기	430 x 558

Amazon Rekognition (Link : [AWS Rekognition Demo](#))

- 사이트를 방문하여 이미지 URL 입력하거나, “업로드” 버튼으로 또는 끌어서 놓기로 업로드합니다.
- Rekognition은 이미지에 있는 피사체, 개념, 장면에 자동으로 레이블을 지정하고 신뢰도 점수를 제공합니다.

The screenshot shows the AWS Rekognition Demo interface. The left sidebar contains a navigation menu with options like 'Amazon Rekognition', '데모', '객체 및 장면 감지', '이미지 조절', '얼굴 분석', '유명 인사 인식', '얼굴 비교', '이미지 내 텍스트', '비디오 데모', '비디오 분석', '지표', '지표', '추가 리소스', '시작 안내서', 'SDK 다운로드', and '개발자 리소스'. The main content area is titled '객체 및 장면 감지' (Object and Scene Detection) and includes a description: 'Rekognition은 이미지에 있는 피사체, 개념, 장면에 자동으로 레이블을 지정하고 신뢰도 점수를 제공합니다.' (Rekognition automatically labels objects, concepts, and scenes in images and provides confidence scores). Below this is a large image of a street scene with bounding boxes around various objects. A red arrow points to the '업로드' (Upload) button. Below the image are two sample images and a section for '자신만의 고유 이미지 사용' (Use your own images), which includes a text input for '이미지 URL 사용' (Use image URL) and an '이동' (Go) button. On the right, a table shows the detection results:

결과	
Vehicle	98.8 %
Car	98.8 %
Transportation	98.8 %
Automobile	98.8 %
Person	98.3 %
Human	98.3 %

Below the table, there are links for '자세히 알아보기' (Learn more), '요청' (Request), and '응답' (Response).

요청(request)과 응답(response)

Format : JSON

▼ 요청

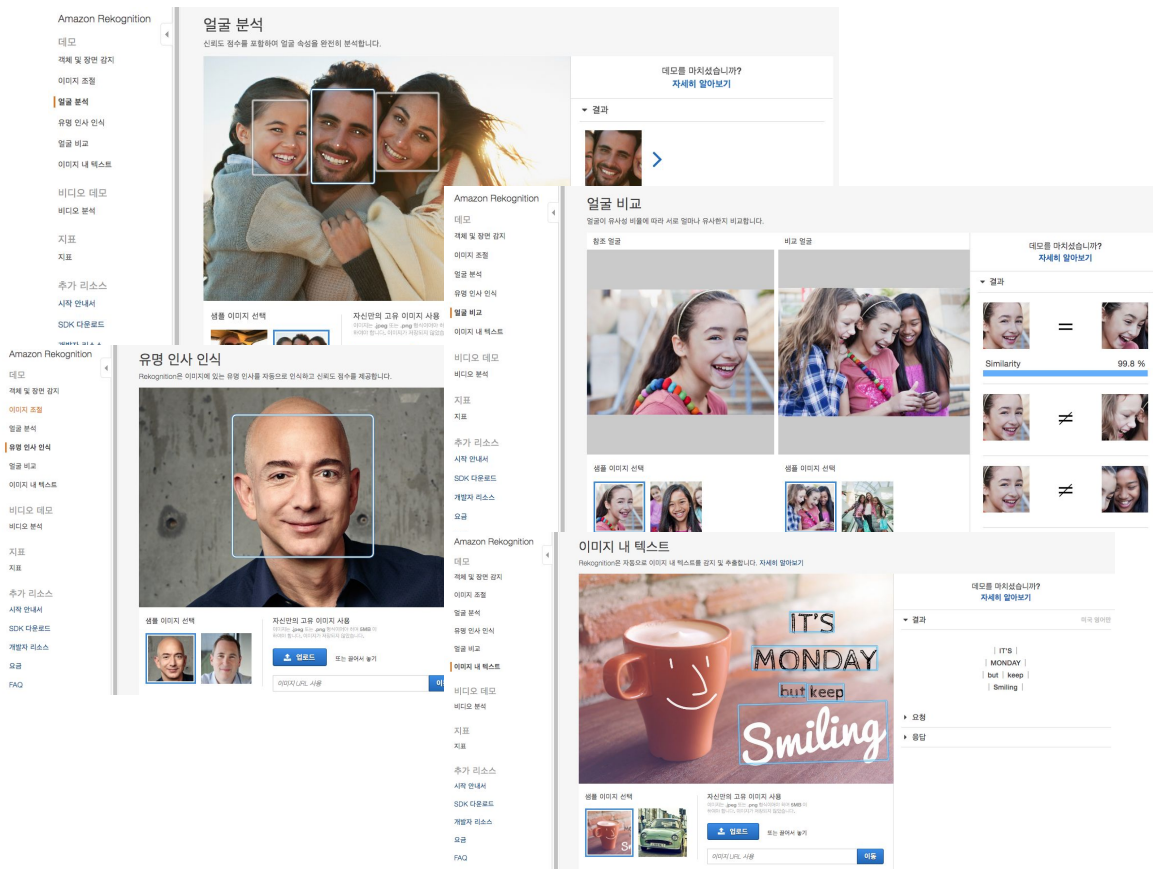
```
{
  "Image": {
    "S3Object": {
      "Bucket": "console-sample-images-icn",
      "Name": "skateboard.jpg"
    }
  }
}
```

▼ 응답

```
{
  "Labels": [
    {
      "Name": "Transportation",
      "Confidence": 98.87621307373047,
      "Instances": [],
      "Parents": []
    },
    {
      "Name": "Automobile",
      "Confidence": 98.87621307373047,
      "Instances": [],
      "Parents": [
        {
          "Name": "Vehicle"
        },
        {
          "Name": "Transportation"
        }
      ]
    }
  ],
  {
    "Name": "Car",
    "Confidence": 98.87621307373047,
    "Instances": [
      {
        "BoundingBox": {
          "Width": 0.10527367144823074,
          "Height": 0.18472492694854736,
          "Left": 0.0042892382480204105,
          "Top": 0.5051581859588623
        }
      },
      "Confidence": 98.87621307373047
    ]
  }
}
```

지원 인식 기능

- 객체 및 장면 감지
- 이미지 조절
- 얼굴분석
- 유명 인사 인식
- 얼굴 비교
- 이미지 내 텍스트



JSON

JSON (JavaScript Object Notation)은 간단한 데이터 교환 형식입니다.

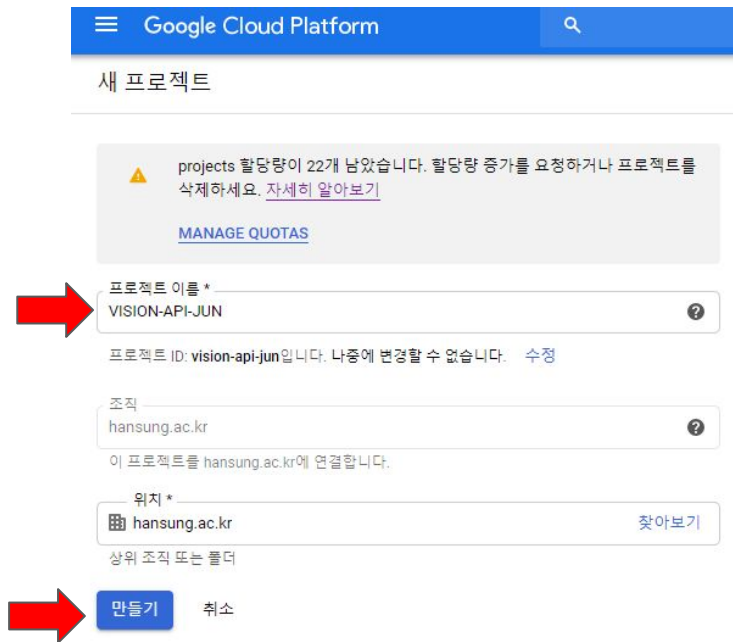
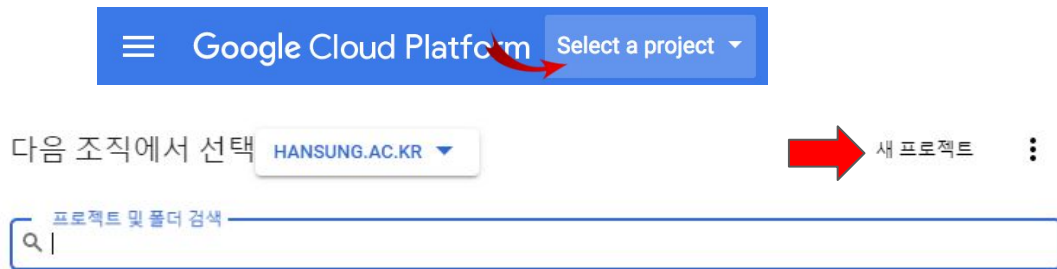
- 읽고 쓰기가 쉽습니다.
- 기계가 구문 분석하고 생성하기 쉽습니다.
- JSON은 완전히 언어에 독립적인 텍스트 형식입니다.
- C++, C#, Java, **JavaScript**, Perl, **Python** 및 기타 프로그램 언어들에서 이상적인 데이터 교환 언어로 사용됩니다.

JSON Syntax

- 데이터는 **이름:값** 쌍으로 넣습니다. ex) { "name": "Jun" }
- 데이터는 **쉼표**로 구분됩니다. ex) { "name": "Jun", "age": 55 }
- 객체(object)는 **중괄호 { }** 에 넣습니다.
ex) { "employee": { "name": "John", "age": 30, "city": "New York" } }
- 배열(array)은 **대괄호 []** 에 넣습니다.
ex) { "employees": ["John", "Anna", "Peter"] }
- 값으로는 문자열, 숫자, **JSON 객체**, 배열, 불리안(boolean), null 이 올 수 있다. ex) { "item": "shirt", "sale": true, "brand": null }
- [JSON Parser example](#)

Using the Google Vision API with Python ([Link](#))

1. Google 계정이 필요 (e.g. ****@gmail.com**)
2. Sign-in to Google Cloud Platform console (console.cloud.google.com) and **create a new project** (or 기존의 것 사용):



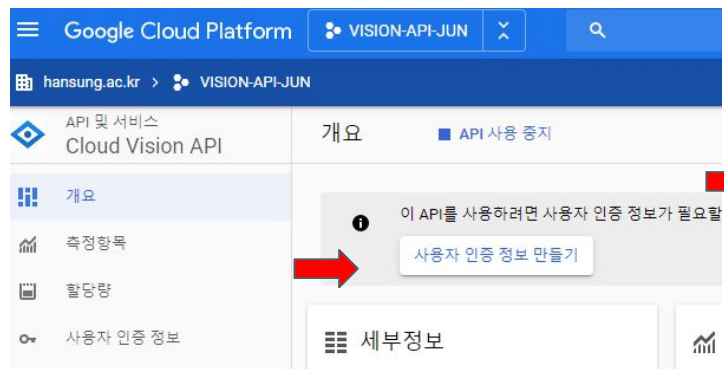
Using the Google Vision API with Python

3. Enable Vision API : **API** 탐색 및 설정 => 라이브러리 => “vision” 입력 => 클릭

The image illustrates the steps to enable the Google Vision API in the Google Cloud Platform console:

- API 탐색 및 설정:** The first screenshot shows the Google Cloud Platform console sidebar. A red arrow points to the **API** link under the '시작하기' (Get started) section.
- 라이브러리:** The second screenshot shows the 'API 및 서비스' (APIs & Services) page. A red arrow points to the **라이브러리** (Library) option in the left sidebar.
- Cloud Vision API:** The third screenshot shows the search results for 'vision' in the library. A red arrow points to the **Cloud Vision API** result, which is highlighted.

4. 서비스계정 Key 만들기 : 사용자 인증정보 만들기 => 서비스계정 이름입력 => 권한 “프로젝트 소유자” => 클릭 “키 만들기” => “JSON” => 다운로드 파일 저장



서비스 계정 만들기

- 1 서비스 계정 세부정보
- 2 이 서비스 계정에 프로젝트에 대한 액세스 권한
- 3 사용자에게 이 서비스 계정에 대한 액세스 권한 부여 (선택사항)

서비스 계정 세부정보

서비스 계정 이름
VISION-API-JUN

이 서비스 계정의 표시 이름입니다.

서비스 계정 ID
vision-api-jun

@vision-api-jun.iam.gserviceaccount.com

서비스 계정 설명

이 서비스 계정에서 수행할 작업을 설명하세요.

서비스 계정 권한(선택사항)

이 서비스 계정에 VISION-API-JUN에 대한 액세스 권한을 부여하여 프로젝트의 리소서 특정 작업을 완료할 권한을 줍니다. [자세히 알아보기](#)

역할
소유자

조건
조건 추가

모든 리소스에 대한 전체 권한입니다.

다른 역할 추가

키 만들기(선택사항)

비공개 키가 포함된 파일을 다운로드합니다. 이 키를 안전하게 저장하세요. 키가 왜 필요한지 확실하지

키 유형

☒ JSON

권장

☐ P12

P12 형식을 사용하는 코드와의 하위 호환용

키 만들기(선택사항)

비공개 키가 포함된 파일을 다운로드합니다. 이 키가 손실될 경우 복구할 수 없으므로 파일을 안전하게 저장하세요. 하지만 키가 왜 필요한지 확실하지 않다면 이 단계를 일단 건너뛰세요.

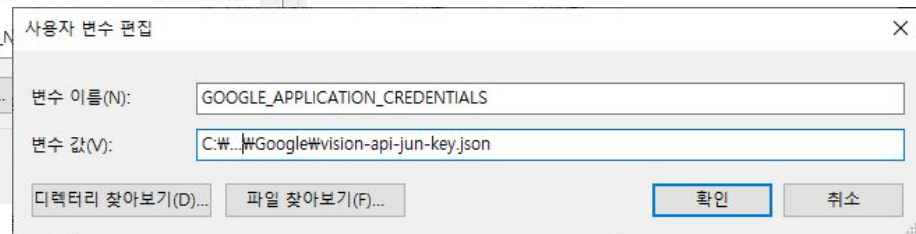
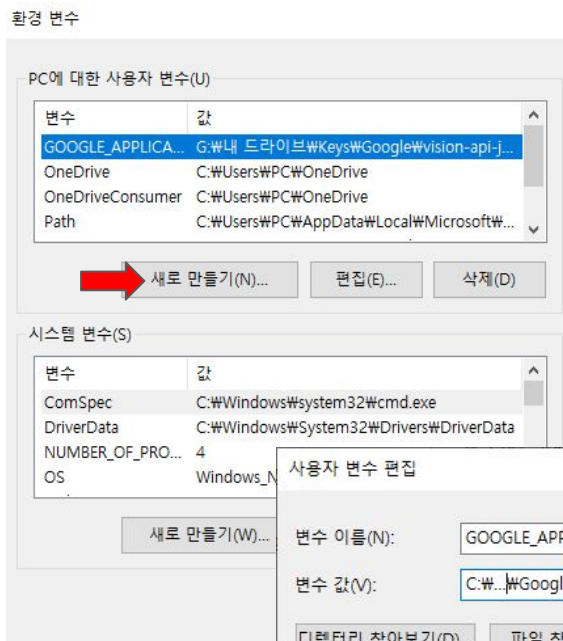
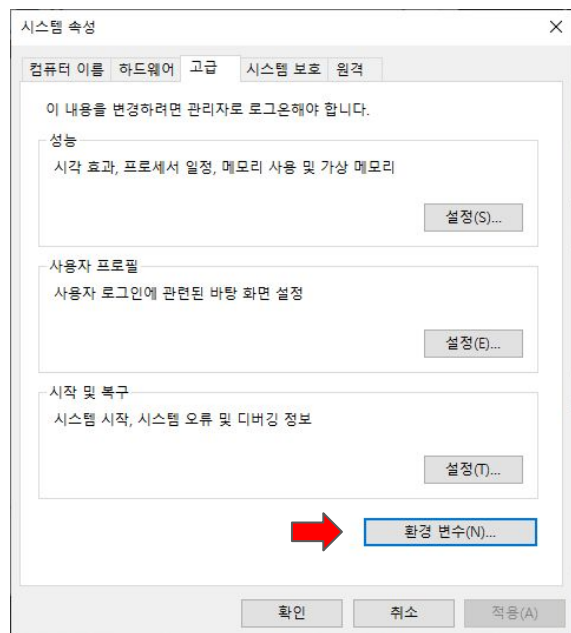
+ 키 만들기

완료 취소

만들기 취소

vision-api-jun-e3...json

5. 환경변수에 다운 받은 key 설정 : 제어판 => 시스템 및 보안 => 시스템 (또는 내 컴퓨터 오른쪽 마우스 클릭 후 설정) => 고급 시스템 설정 => 환경변수 => 사용자 변수 : 새로 만들기 => **변수이름 : GOOGLE_APPLICATION_CREDENTIALS**
변수 값 : C:\...\key-파일명.json



6. Cloud Vision client python library 설치 : 명령창(command)에 다음처럼 실행

```
>> pip install google-cloud-vision
```

7. label_detect.py 를 다음과 같이 작성

```
from google.cloud import vision
import io, os
# os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = "my_key_name.json"    # 직접 사용도 가능
image_uri = 'gs://cloud-samples-data/vision/using_curl/shanghai.jpeg' #URL로 이미지 지정

client = vision.ImageAnnotatorClient()
image = vision.Image()          # the new vision object doesn't have types
image.source.image_uri = image_uri

response = client.label_detection(image=image)

print('Labels (and confidence score):')
print('=' * 79)
for label in response.label_annotations:
    print(f'{label.description} ({label.score*100:.2f}%)')
```

8. label_detect.py 실행하면, 그 아래와 같은 파일 출력.

```
>> python label_detect.py
```

Labels (and confidence score):

=====

People (95.05%)
Street (89.12%)
Mode of transport (89.09%)
Transport (85.13%)
Vehicle (84.69%)
Snapshot (84.11%)
Urban area (80.29%)
Infrastructure (73.14%)
Road (72.74%)
Pedestrian (68.90%)



8. (local image 활용) label_detect2.py 를 다음과 같이 작성

```
import io, os
# os.environ['GOOGLE_APPLICATION_CREDENTIALS'] = "my_key_name.json"    # 직접 사용
from google.cloud import vision
from google.cloud.vision import types

client = vision.ImageAnnotatorClient()    # Instantiates a client
file_name = os.path.abspath('cat.jpg')    # The name of the image file to annotate

with io.open(file_name, 'rb') as image_file:    # Loads the image into memory
    content = image_file.read()

image = types.Image(content=content)
response = client.label_detection(image=image)    # Performs label detection on the image file
labels = response.label_annotations

print('Labels:')
for label in labels:
    print(f'{label.description} ({label.score*100:.2f}%)')
```

9. (local image 활용) label_detect2.py 실행하면, 그 아래와 같은 파일 출력.

>> python label_detect2.py

Labels:

Mammal (98.90%)

Vertebrate (98.51%)

Canidae (94.74%)

Cat (94.34%)

Dog breed (94.22%)

Dog (94.09%)

Carnivore (93.42%)

Whiskers (89.37%)

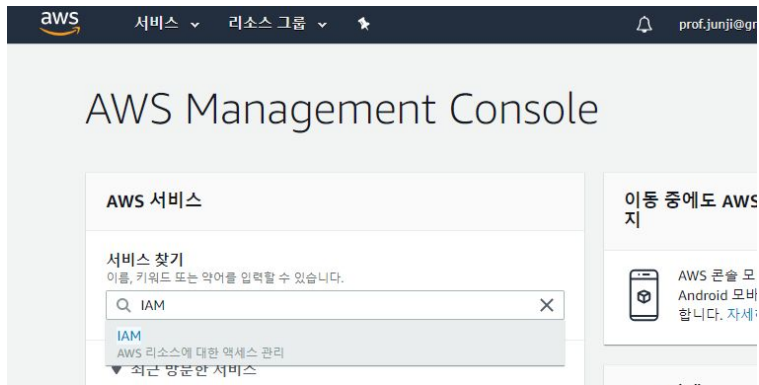
Companion dog (86.17%)

Felidae (83.34%)



Using AWS Rekognition API

1. Amazon AWS 계정 생성 (e-mail & password) : 1 year free
2. 로그인 AWS Management Console
3. IAM(Identity and Access Management) 에서 credential.csv 다운로드
 - a. Click “user(사용자:0)”



Identity and Access Management 소개

IAM 사용자 로그인 링크:

<https://836096688231.signin.aws.amazon.com/console>

사용자 지정

IAM 리소스

사용자: 2

역할: 2

그룹: 0

자격 증명 공급자: 0

고객 관리형 정책: 0

Using AWS Rekognition API

- Click “add user(사용자 추가)” => Type in “user_name” => Check “프로그래밍 방식 액세스”=> Click Next:Tags (다음:권한)

사용자 추가 사용자 삭제

<input type="checkbox"/>	사용자 이름	그룹	액세스 키 수명
<input type="checkbox"/>	junji	없음	✓ 20 일
<input type="checkbox"/>	junji2	없음	✓ 오늘

사용자 추가

1 2 3 4 5

사용자 세부 정보 설정

동일한 액세스 유형 및 권한을 사용하여 한 번에 여러 사용자를 추가할 수 있습니다. [자세히 알아보기](#)

사용자 이름

junji5

[다른 사용자 추가](#)

AWS 액세스 유형 선택

해당 사용자가 AWS에 액세스하는 방법을 선택합니다. 마지막 단계에서는 액세스 키와 자동 생성된 비밀번호가 제공됩니다. [자세히 알아보기](#)

액세스 유형

☒

프로그래밍 방식 액세스

AWS API, CLI, SDK 및 기타 개발 도구에 대해 액세스 키 ID 및 비밀 액세스 키 을(를) 활성화합니다.

☐

AWS Management Console 액세스

사용자가 AWS Management Console에 로그인할 수 있도록 허용하는 비밀번호 을(를)

필수


[취소](#)


[다음: 권한](#)


Using AWS Rekognition API

- Click “기존 정책 직접 연결”
- 검색: Rekognition => Check: ☐ AmazonRekognitionFullAccess
- 검색: S3 => Click: ☐ AmazonS3FullAccess
- 다음:태그 => Click:
- 다음:검토 => Click: 사용자만들기

▼ 권한 설정

 그룹에 사용자 추가

 기존 사용자에서 권한 복사

 기존 정책 직접 연결

정책 생성

정책 필터 2 결과 표시[취소](#)[이전](#)[다음: 검토](#)

게 연결됩니다.

RekognitionFullAccess

S3FullAccess

[취소](#)[이전](#)[다음: 태그](#)[취소](#)[이전](#)[사용자 만들기](#)

Using AWS Rekognition API

- Click: csv 다운로드 => credential.csv
- 관리자 권한으로 명령 프롬프트 실행
- >> pip install boto3

install boto3 module



다음 처럼 **test.py** 생성 후, 이미지(e.g. hot.jpg)와 함께 실행 >> **python test.py**
(주의: 실행 전에 **boto3** 설치 필요: >> **pip install boto3**)

```
import csv, boto3

with open('credentials.csv', 'r') as input:
    next(input)                                # skip the first line
    reader = csv.reader(input)
    for line in reader:
        access_key_id = line[2]
        secret_access_key = line[3]

client = boto3.client('rekognition', aws_access_key_id = access_key_id,
                        aws_secret_access_key = secret_access_key, region_name = 'us-west-1')

photo = 'hot.jpg'
with open(photo, 'rb') as source_image:
    source_bytes = source_image.read()

response = client.detect_labels(Image={'Bytes': source_bytes}, MaxLabels=10)

print(response)
```

```
>> python test.py
```

```
{'Labels': [{'Name': 'Balloon', 'Confidence': 99.81890869140625, 'Instances': [{'BoundingBox': {'Width': 0.506658673286438, 'Height': 0.8700017333030701, 'Left': 0.11146698147058487, 'Top': 0.06764169782400131}, 'Confidence': 99.81890869140625}], 'Parents': [{'Name': 'Ball'}], {'Name': 'Ball', 'Confidence': 99.81890869140625, 'Instances': [], 'Parents': []}, {'Name': 'Aircraft', 'Confidence': 94.19926452636719, 'Instances': [], 'Parents': [{'Name': 'Vehicle'}, {'Name': 'Transportation'}]}, {'Name': 'Vehicle', 'Confidence': 94.19926452636719, 'Instances': [], 'Parents': [{'Name': 'Transportation'}]}, {'Name': 'Hot Air Balloon', 'Confidence': 94.19926452636719, 'Instances': [], 'Parents': [{'Name': 'Aircraft'}, {'Name': 'Vehicle'}, {'Name': 'Transportation'}]}, {'Name': 'Transportation', 'Confidence': 94.19926452636719, 'Instances': [], 'Parents': []}], 'LabelModelVersion': '2.0', 'RequestId': 'd8fc766e-52f9-47c6-8f6c-9a1b150a2173', 'HTTPStatusCode': 200, 'application/x-amz-json-1.1', 'date': 'Wed, 22 Apr 2020 08:00:00 GMT', 'content-length': 1000, 'RetryAttempts': 0}}
```

