Optical Character Recognition (OCR) Tutorial

Learn how to perform optical character recognition (OCR) on Google Cloud Platform. This tutorial demonstrates how to upload image files to Google Cloud Storage (https://cloud.google.com/storage?hl=zh-tw), extract text from the images using the Google Cloud Vision API (https://cloud.google.com/vision?hl=zh-tw), translate the text using the Google Cloud Translation API (https://cloud.google.com/translate?hl=zh-tw), and save your translations back to Cloud Storage. Google Cloud Pub/Sub (https://cloud.google.com/pubsub?hl=zh-tw) is used to queue various tasks and trigger the right Cloud Functions to carry them out.

Objectives

- Write and deploy several Background Cloud Functions (https://cloud.google.com/functions/docs/writing/background?hl=zh-tw).
- Upload images to Cloud Storage.
- Extract, translate and save text contained in uploaded images.

Costs

This tutorial uses billable components of Cloud Platform, including:

- Google Cloud Functions
- Google Cloud Pub/Sub
- Google Cloud Storage
- Google Cloud Translation API
- Google Cloud Vision API

Use the Pricing Calculator (https://cloud.google.com/products/calculator?hl=zh-tw) to generate a cost estimate based on your projected usage.

Before you begin

1. 在 GCP Console 的專案選擇器頁面中，選取或建立 GCP 專案。

   注意：如果您不打算保留您在這個程序中建立的資源，請建立新的 GCP 專案，而不是選取現有的 GCP 專案。因此您在完成這些步驟之後，就可以刪除專案，並移除與該專案相關聯的所有資源。

   前往專案選取器頁面 (HTTPS://CONSOLE.CLOUD.GOOGLE.COM/PROJECTSELECTOR2/HOMI

2. 請確認您已啟用 Google Cloud Platform 專案的計費功能。瞭解如何確認您已啟用專案的計費功能 (https://cloud.google.com/billing/docs/how-to/modify-project?hl=zh-tw)。

3. 啟用Cloud Functions, Cloud Pub/Sub, Cloud Storage, Cloud Translation, and Cloud Vision必要的 API。

   啟用 **API** (HTTPS://CONSOLE.CLOUD.GOOGLE.COM/FLOWS/ENABLEAPI?APIID=CLOUDFUN(

4. Update `gcloud` components:

   ```
   gcloud components update
   ```

5. Prepare your development environment.

   NODE.JS   PYTHON   GO

   GO TO THE PYTHON SETUP GUIDE (HTTPS://CLOUD.GOOGLE.COM/PYTHON/SETUP?HL=

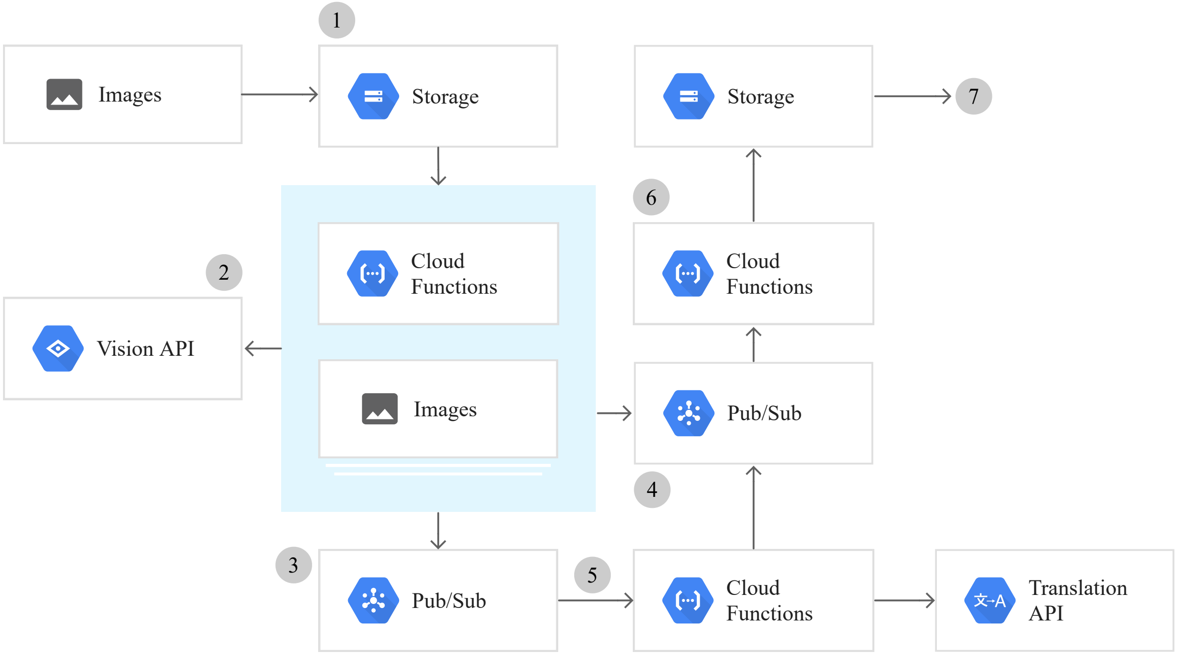
Visualizing the flow of data

The flow of data in the OCR tutorial application involves several steps:

1. An image that contains text in any language is uploaded to Cloud Storage.

2. A Cloud Function is triggered, which uses the Vision API to extract the text and detect the source language.
3. The text is queued for translation by publishing a message to a Pub/Sub topic (https://cloud.google.com/pubsub/docs/publisher?hl=zh-tw). A translation is queued for each target language different from the source language.
4. If a target language matches the source language, the translation queue is skipped, and text is sent to the result queue, another Pub/Sub topic.
5. A Cloud Function uses the Translation API to translate the text in the translation queue. The translated result is sent to the result queue.
6. Another Cloud Function saves the translated text from the result queue to Cloud Storage.
7. The results are found in Cloud Storage as `txt` files for each translation.

It may help to visualize the steps:



Preparing the application

1. Create a Cloud Storage bucket to upload your images, where ***YOUR_IMAGE_BUCKET_NAME*** is a globally unique bucket name:

```
gsutil mb gs://YOUR_IMAGE_BUCKET_NAME
```

2. Create a Cloud Storage bucket to save the translations, where ***YOUR_TEXT_BUCKET_NAME*** is a globally unique bucket name:

```
gsutil mb gs://YOUR_TEXT_BUCKET_NAME
```

3. Clone the sample app repository to your local machine:

NODE.JS PYTHON GO

```
git clone https://github.com/GoogleCloudPlatform/python-docs-sam
```

Alternatively, you can download the sample (https://github.com/GoogleCloudPlatform/python-docs-samples/archive/master.zip) as a zip file and extract it.

4. Change to the directory that contains the Cloud Functions sample code:

NODE.JS PYTHON GO

```
cd python-docs-samples/functions/ocr/app/
```

5. Configure the app:

NODE.JS PYTHON GO

Edit the `config.json` file in the `app` directory to have the following contents:

```
{
"RESULT_TOPIC": "YOUR_RESULT_TOPIC_NAME",
"RESULT_BUCKET": "YOUR_TEXT_BUCKET_NAME",
"TRANSLATE_TOPIC": "YOUR_TRANSLATE_TOPIC_NAME",
"TRANSLATE": true,
"TO_LANG": ["en", "fr", "es", "ja", "ru"]
}
```

- Replace ***YOUR_RESULT_TOPIC_NAME*** with a Pub/Sub topic name to be used for saving results once the translation step is complete.
- Replace ***YOUR_TEXT_BUCKET_NAME*** with a bucket name used for saving translation results.
- Replace ***YOUR_TRANSLATE_TOPIC_NAME*** with a Pub/Sub topic name to trigger the translation of the extracted text.

Understanding the code

Importing dependencies

The application must import several dependencies in order to communicate with Google Cloud Platform services:

NODE.JS PYTHON GO

functions/ocr/app/main.py (https://github.com/GoogleCloudPlatform/python-docs-samples/blob/master/functions/ocr/app/main.py)

ECLOUDPLATFORM/PYTHON-DOCS-SAMPLES/BLOB/MASTER/FUNCTIONS/OCR/APP/MAIN.PY)

```
import base64
import json
import os

from google.cloud import pubsub_v1
from google.cloud import storage
from google.cloud import translate
from google.cloud import vision

vision_client = vision.ImageAnnotatorClient()
translate_client = translate.Client()
publisher = pubsub_v1.PublisherClient()
storage_client = storage.Client()

project_id = os.environ['GCP_PROJECT']

with open('config.json') as f:
    data = f.read()
config = json.loads(data)
```

Processing images

The following function reads an uploaded image file from Cloud Storage and calls a function to detect whether the image contains text:

NODE.JS PYTHON GO

functions/ocr/app/main.py (https://github.com/GoogleCloudPlatform/python-docs-samples/blob/master/functions/ocr/app/main.py)

CLOUDPLATFORM/PYTHON-DOCS-SAMPLES/BLOB/MASTER/FUNCTIONS/OCR/APP/MAIN.PY)

```
def process_image(file, context):
    """Cloud Function triggered by Cloud Storage when a file is changed
    Args:
        file (dict): Metadata of the changed file, provided by the trig
                                Cloud Storage event.
        context (google.cloud.functions.Context): Metadata of triggerin
    Returns:
        None; the output is written to stdout and Stackdriver Logging
    """
    bucket = validate_message(file, 'bucket')
    name = validate_message(file, 'name')

    detect_text(bucket, name)

    print('File {} processed.'.format(file['name']))
```

The following function extracts text from the image using the Cloud Vision API and queues the text for translation:

NODE.JS PYTHON GO

functions/ocr/app/main.py (https://github.com/GoogleCloudPlatform/python-docs-samples/blob/master/functions/ocr/app/main.py)

CLOUDPLATFORM/PYTHON-DOCS-SAMPLES/BLOB/MASTER/FUNCTIONS/OCR/APP/MAIN.PY)

```
def detect_text(bucket, filename):
    print('Looking for text in image {}'.format(filename))

    futures = []

    text_detection_response = vision_client.text_detection({
        'source': {'image_uri': 'gs://{}/{}'.format(bucket, filename)}
    })
    annotations = text_detection_response.text_annotations
    if len(annotations) > 0:
        text = annotations[0].description
    else:
        text = ''
    print('Extracted text {} from image ({} chars).'.format(text, len(t

    detect_language_response = translate_client.detect_language(text)
    src_lang = detect_language_response['language']
    print('Detected language {} for text {}.'.format(src_lang, text))

    # Submit a message to the bus for each target language
    for target_lang in config.get('TO_LANG', []):
        topic_name = config['TRANSLATE_TOPIC']
        if src_lang == target_lang or src_lang == 'und':
            topic_name = config['RESULT_TOPIC']
        message = {
            'text': text,
            'filename': filename,
            'lang': target_lang,
            'src_lang': src_lang
        }
        message_data = json.dumps(message).encode('utf-8')
        topic_path = publisher.topic_path(project_id, topic_name)
        future = publisher.publish(topic_path, data=message_data)
        futures.append(future)
    for future in futures:
        future.result()
```

Translating text

The following function translates the extracted text and queues the translated text to be saved back to Cloud Storage:

NODE.JS PYTHON GO

functions/ocr/app/main.py (https://github.com/GoogleCloudPlatform/python-docs-samples/blob/master/functions/ocr/app/main.py)

ECLOUDPLATFORM/PYTHON-DOCS-SAMPLES/BLOB/MASTER/FUNCTIONS/OCR/APP/MAIN.PY)

```
def translate_text(event, context):
    if event.get('data'):
        message_data = base64.b64decode(event['data']).decode('utf-8')
        message = json.loads(message_data)
    else:
        raise ValueError('Data sector is missing in the Pub/Sub message

    text = validate_message(message, 'text')
    filename = validate_message(message, 'filename')
    target_lang = validate_message(message, 'lang')
    src_lang = validate_message(message, 'src_lang')

    print('Translating text into {}.'.format(target_lang))
    translated_text = translate_client.translate(text,
                                                 target_language=target
                                                 source_language=src_la
    topic_name = config['RESULT_TOPIC']
    message = {
        'text': translated_text['translatedText'],
        'filename': filename,
        'lang': target_lang,
    }
    message_data = json.dumps(message).encode('utf-8')
    topic_path = publisher.topic_path(project_id, topic_name)
    future = publisher.publish(topic_path, data=message_data)
    future.result()
```

Saving the translations

Finally, the following function receives the translated text and saves it back to Cloud Storage:

NODE.JS PYTHON GO

functions/ocr/app/main.py (https://github.com/GoogleCloudPlatform/python-docs-samples/blob/master/functions/ocr/app/main.py)

CLOUDPLATFORM/PYTHON-DOCS-SAMPLES/BLOB/MASTER/FUNCTIONS/OCR/APP/MAIN.PY)

```
def save_result(event, context):
    if event.get('data'):
        message_data = base64.b64decode(event['data']).decode('utf-8')
        message = json.loads(message_data)
    else:
        raise ValueError('Data sector is missing in the Pub/Sub message

    text = validate_message(message, 'text')
    filename = validate_message(message, 'filename')
    lang = validate_message(message, 'lang')

    print('Received request to save file {}.'.format(filename))

    bucket_name = config['RESULT_BUCKET']
    result_filename = '{}_{}.txt'.format(filename, lang)
    bucket = storage_client.get_bucket(bucket_name)
    blob = bucket.blob(result_filename)

    print('Saving result to {} in bucket {}.'.format(result_filename,
                                                     bucket_name))

    blob.upload_from_string(text)

    print('File saved.')
```

Deploying the functions

This section describes how to deploy your functions.

1. To deploy the image processing function with a Cloud Storage trigger, run the following command in the `app` directory:

   NODE.JS PYTHON GO

   ```
   gcloud functions deploy ocr-extract --runtime python37 --trigger
   ```

   where ***YOUR_IMAGE_BUCKET_NAME*** is the name of your Cloud Storage bucket where you will be uploading images.

2. To deploy the text translation function with a Cloud Pub/Sub trigger, run the following command in the `app` directory:

   NODE.JS PYTHON GO

   ```
   gcloud functions deploy ocr-translate --runtime python37 --trigg
   ```

   where ***YOUR_TRANSLATE_TOPIC_NAME*** is the name of your Cloud Pub/Sub topic with which translations will be triggered.

3. To deploy the function that saves results to Cloud Storage with a Cloud Pub/Sub trigger, run the following command in the `app` directory:

   NODE.JS PYTHON GO

   ```
   gcloud functions deploy ocr-save --runtime python37 --trigger-to
   ```

   where ***YOUR_RESULT_TOPIC_NAME*** is the name of your Cloud Pub/Sub topic with which saving of results will be triggered.

Uploading an image

1. Upload an image to your image Cloud Storage bucket:

   ```
   gsutil cp PATH_TO_IMAGE gs://YOUR_IMAGE_BUCKET_NAME
   ```

   where

   - ***PATH_TO_IMAGE*** is a path to an image file (that contains text) on your local system.
   - ***YOUR_IMAGE_BUCKET_NAME*** is the name of the bucket where you are uploading images.

   You can download one of the images from the sample project (https://github.com/GoogleCloudPlatform/nodejs-docs-samples/tree/master/functions/ocr/images)
   .

2. Watch the logs to be sure the executions have completed:

   ```
   gcloud functions logs read --limit 100
   ```

3. You can view the saved translations in the Cloud Storage bucket specified by the ***RESULT_BUCKET*** value in your configuration file.

Cleaning up

To avoid incurring charges to your Google Cloud Platform account for the resources used in this tutorial:

Deleting the project

The easiest way to eliminate billing is to delete the project that you created for the tutorial.

To delete the project:

> **注意**：刪除專案會造成以下結果：
>
> - **專案中的所有內容都會遭到刪除。** 如果您之前使用現有的專案來進行本教學課程，當您刪除該專案時，也會一併刪除您在該專案中完成的所有其他工作。
> - **自訂的專案 ID 會消失。**當您之前建立這個專案時，可能建立了想要在日後使用的自訂專案 ID。如要保留使用該專案 ID 的網址 (例如 `appspot.com` 網址)，請刪除在該專案中選取的資源，而不是刪除整個專案。
>
> 如果您打算進行多個教學課程及快速入門導覽課程，重複使用專案可協助您避免讓資源用量超出專案的配額限制。

1. 前往 GCP Console 中的「Manage resources」(管理資源) 頁面。

   前往「**MANAGE RESOURCES**」(管理資源) 頁面 (HTTPS://CONSOLE.CLOUD.GOOGLE.COM/I/

2. 在專案清單中選取您要刪除的專案，然後按一下「Delete」(刪除) 圖示 🗑 。
3. 在對話方塊中輸入專案 ID，然後按一下 [Shut down] (關閉) 以刪除專案。

Deleting the Cloud Functions

Deleting Cloud Functions does not remove any resources stored in Cloud Storage.

To delete the Cloud Functions you created in this tutorial, run the following commands:

```
gcloud functions delete ocr-extract
gcloud functions delete ocr-translate
gcloud functions delete ocr-save
```

You can also delete Cloud Functions from the Google Cloud Console (https://console.cloud.google.com/functions/list?hl=zh-tw).

上次更新：一月 14, 2020。