Translate using Google Cloud Platform (GCP) Translation API

Purushottam Mohanty [email] [github]

2021-08-30

This is a short write-up for beginners who want to use the Google Cloud Platform (GCP) Translation Client API to translate one or multiple variables from a dataset.

The GCP Translate API can be used for quick and effective translation. The Basic API is free to use upto 500,000 characters per month and thereafter carries a cost of 20 USD for each 1 million character. There are also advanced, media and auto ML translate APIs which is beyond the scope of the article. Note that if you translate a string or text without specifying the source language, the characters are only counted once towards the quota. There's no additional cost of detecting the language. However, explicitly detecting source language count towards the quota.

Google provides an official translation API for Python and that is what the following blog uses. First of all we are going to setup a GCP account and create our credentials.

1. Go to Google Cloud and sign in using your Google Account.

2. Once you are in your dashboard, go to the top bar and create a new project.

3. Then click billing and then setup billing for the project. You would ideally need your credit card to set it up. (Don't worry you will not be charged for any request beyond your quota unless to move to a paid account. The APIs will provide an error if you exceed your quota and will not proceed further without converting to a paid account.) Additionally, at the time of writing this post, Google provided a joining credit of 300 USD which can be used.

4. Once billing has been setup, select the "APIs and Services" option in the right side menu and search for "Translate API" and click "Enable".

5. Select "IAM and Admin" and then "Service Account".

6. Create a Service Account and then create keys. (The keys automatically get downloaded as a json file. You cannot download the keys again so keep it somewhere safe in your local machine.)

Then we shall download the required packages directly from terminal in case you haven't downloaded these previously. Simply run these code in terminal.

```
$ python3 -m pip install pandas
$ python3 -m pip install numpy
$ python3 -m pip install google-cloud-translate
```

Import Necessary Packages

Now run the following chunk directly in Python to import the libraries.

```python
import pandas as pd
import numpy as np
import json
from google.cloud import translate
```

Setup GCP Client Credentials

```python
# set service account credentials
client_credentials = json.load(open("FULL_PATH_TO_KEY.json"))
# set project id and build client
project_id = client_credentials['project_id']
assert project_id
parent = f"projects/{project_id}"
client = translate.TranslationServiceClient()
```

Getting all Language Codes

```python
# Get all languages
response = client.get_supported_languages(parent = parent, display_language_code = "en")
languages = response.languages

print(f" Languages: {len(languages)} ".center(60, "-"))
for language in languages:
    print(f"{language.language_code}\t{language.display_name}")
```

Translate Example

```python
# GCP translate
sample_text = ["Bonjour", "Oui"]
target_language_code = "en"

response = client.translate_text(
    contents = sample_text,
    target_language_code = target_language_code,
    parent = parent,
)

for translation in response.translations:
    print(translation.translated_text)
```

Import Dataset

```python
# load dataset as pandas dataframe
input_df = pd.read_stata("PATH_TO_DATASET.dta")

# tidy dataset (remove line separators from data)
input_df = input_df.replace(r'\r', '', regex=True)
input_df = input_df.replace(r'\n', '', regex=True)

# GCP translate API doesn't support empty strings (to avoid errors)
# identify empty rows in variable
input_df['french_var_empty'] = np.where(input_df['french_var'] == "", 1, 0)
```

```
# set a fake string to avoid errors
input_df['french_var'] = np.where(input_df['french_var'] == "", "1", input_df['french_var'])
```

Translate from Dataset

The GCP translate API doesn't accept large chunks of text hence we divide the data into batches of 200 rows and translate within a loop until we exhaust the entire length of the dataframe.

```
# set empty translated list
translated_output_list = []

# translate in chunks
x1 = 0
x2 = 200

while x1 < len(input_df):
    input_text = input_df[x1:x2]['french_var']
    target_language_code = "en"
    # translate
    response = client.translate_text(
        contents = input_text,
        target_language_code = target_language_code,
        parent = parent,
    )
    # append to list
    for n in range(0,len(response.translations)):
        translated_output = response.translations[n].translated_text
        translated_output_list.append(translated_output)
    # update counter
    x1 = x1 + 200
    x2 = x1 + 200
```