PY0101EN-5.2_API_2

August 20, 2020

Application Programming Interface

In this notebook, you will learn to convert an audio file of an English speaker to text using a Speech to Text API. Then you will translate the English version to a Spanish version using a Language Translator API. Note: You must obtain the API keys and enpoints to complete the lab.

```
<a href="https://cocl.us/topNotebooksPython101Coursera">
     <img src="https://s3-api.us-geo.objectstorage.softlayer.net/cf-courses-data/CognitiveClass
</a>
```

Table of Contents

Speech To Text

Language Translator

Exercise

Estimated Time Needed: 25 min

[1]:
```
#you will need the following library
!pip install ibm_watson wget
```

```
Collecting ibm_watson
  Downloading https://files.pythonhosted.org/packages/dc/da/10f8774b319acd
da29885931c01fae862622519bff492957c73b0ba84743/ibm-watson-4.5.0.tar.gz (370kB)
     |                 | 378kB 9.0MB/s eta 0:00:01
Collecting wget
  Downloading https://files.pythonhosted.org/packages/47/6a/62e288da7bcda82b935f
f0c6cfe542970f04e29c756b0e147251b2fb251f/wget-3.2.zip
Requirement already satisfied: requests<3.0,>=2.0 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from ibm_watson)
(2.24.0)
Requirement already satisfied: python_dateutil>=2.5.3 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from ibm_watson)
(2.8.1)
Collecting websocket-client==0.48.0 (from ibm_watson)
  Downloading https://files.pythonhosted.org/packages/8a/a1/72ef9aa26cfe1a
75cee09fc1957e4723add9de098c15719416a1ee89386b/websocket_client-0.48.0-py2.py3-n
one-any.whl (198kB)
     |                 | 204kB 14.4MB/s eta 0:00:01
Collecting ibm_cloud_sdk_core==1.5.1 (from ibm_watson)
```

```
  Downloading https://files.pythonhosted.org/packages/b7/f6/10d5271c807d73d236e6
ae07b68035fed78b28b5ab836704d34097af3986/ibm-cloud-sdk-core-1.5.1.tar.gz
Requirement already satisfied: idna<3,>=2.5 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
requests<3.0,>=2.0->ibm_watson) (2.10)
Requirement already satisfied: certifi>=2017.4.17 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
requests<3.0,>=2.0->ibm_watson) (2020.6.20)
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
requests<3.0,>=2.0->ibm_watson) (1.25.10)
Requirement already satisfied: chardet<4,>=3.0.2 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
requests<3.0,>=2.0->ibm_watson) (3.0.4)
Requirement already satisfied: six>=1.5 in
/home/jupyterlab/conda/envs/python/lib/python3.6/site-packages (from
python_dateutil>=2.5.3->ibm_watson) (1.15.0)
Collecting PyJWT>=1.7.1 (from ibm_cloud_sdk_core==1.5.1->ibm_watson)
  Downloading https://files.pythonhosted.org/packages/87/8b/6a9f14b5f781697e5125
9d81657e6048fd31a113229cf346880bb7545565/PyJWT-1.7.1-py2.py3-none-any.whl
Building wheels for collected packages: ibm-watson, wget, ibm-cloud-sdk-core
  Building wheel for ibm-watson (setup.py) … done
  Stored in directory: /home/jupyterlab/.cache/pip/wheels/71/9a/0a/9b3ca8e
ca69bc5362eb04709a750b30055a9d27818fd0c9494
  Building wheel for wget (setup.py) … done
  Stored in directory: /home/jupyterlab/.cache/pip/wheels/40/15/30/7d8f7ce
a2902b4db79e3fea550d7d7b85ecb27ef992b618f3f
  Building wheel for ibm-cloud-sdk-core (setup.py) … done
  Stored in directory: /home/jupyterlab/.cache/pip/wheels/6a/42/50/f968881
16b329578304f9dda4693cef6f3e76e18272d22cb6c
Successfully built ibm-watson wget ibm-cloud-sdk-core
Installing collected packages: websocket-client, PyJWT, ibm-cloud-sdk-core, ibm-
watson, wget
Successfully installed PyJWT-1.7.1 ibm-cloud-sdk-core-1.5.1 ibm-watson-4.5.0
websocket-client-0.48.0 wget-3.2
```

Speech to Text

First we import SpeechToTextV1 from ibm_watson.For more information on the API, please click on this link

[2]:
```
from ibm_watson import SpeechToTextV1
import json
from ibm_cloud_sdk_core.authenticators import IAMAuthenticator
```

The service endpoint is based on the location of the service instance, we store the information in the variable URL. To find out which URL to use, view the service credentials.

[3]:
```
url_s2t = "https://stream.watsonplatform.net/speech-to-text/api"
```

You require an API key, and you can obtain the key on the Dashboard .

[4]:
```
iam_apikey_s2t = "6TPZe5tZTq862mC97U_Y1lNVAMh2htERDFG8V9AN4vtA"
```

You create a Speech To Text Adapter object the parameters are the endpoint and API key.

[5]:
```
authenticator = IAMAuthenticator(iam_apikey_s2t)
s2t = SpeechToTextV1(authenticator=authenticator)
s2t.set_service_url(url_s2t)
s2t
```

[5]:
```
<ibm_watson.speech_to_text_v1_adapter.SpeechToTextV1Adapter at 0x7fed20083b38>
```

Lets download the audio file that we will use to convert into text.

[6]:
```
!wget -O PolynomialRegressionandPipelines.mp3  https://s3-api.us-geo.
↪objectstorage.softlayer.net/cf-courses-data/CognitiveClass/PY0101EN/labs/
↪PolynomialRegressionandPipelines.mp3
```

```
--2020-08-20 14:18:52--  https://s3-api.us-geo.objectstorage.softlayer.net/cf-
courses-data/CognitiveClass/PY0101EN/labs/PolynomialRegressionandPipelines.mp3
Resolving s3-api.us-geo.objectstorage.softlayer.net (s3-api.us-
geo.objectstorage.softlayer.net)… 67.228.254.196
Connecting to s3-api.us-geo.objectstorage.softlayer.net (s3-api.us-
geo.objectstorage.softlayer.net)|67.228.254.196|:443… connected.
HTTP request sent, awaiting response… 200 OK
Length: 4234179 (4.0M) [audio/mpeg]
Saving to: ‘PolynomialRegressionandPipelines.mp3’

PolynomialRegressio 100%[===================>]   4.04M  6.31MB/s    in 0.6s

2020-08-20 14:18:53 (6.31 MB/s) - ‘PolynomialRegressionandPipelines.mp3’ saved
[4234179/4234179]
```

We have the path of the wav file we would like to convert to text

[7]:
```
filename='PolynomialRegressionandPipelines.mp3'
```

We create the file object wav with the wav file using open ; we set the mode to “rb” , this is similar to read mode, but it ensures the file is in binary mode.We use the method recognize to return the recognized text. The parameter audio is the file object wav, the parameter content_type is the format of the audio file.

[8]:
```
with open(filename, mode="rb")  as wav:
    response = s2t.recognize(audio=wav, content_type='audio/mp3')
```

The attribute result contains a dictionary that includes the translation:

[9]:
```
response.result
```

[9]:

```
{'result_index': 0,
 'results': [{'final': True,
   'alternatives': [{'transcript': 'in this video we will cover polynomial
regression and pipelines ',
     'confidence': 0.94}]},
  {'final': True,
   'alternatives': [{'transcript': "what do we do when a linear model is not the
best fit for our data let's look into another type of regression model the
polynomial regression we transform our data into a polynomial then use linear
regression to fit the parameters that we will discuss pipelines pipelines are
way to simplify your code ",
     'confidence': 0.9}]},
  {'final': True,
   'alternatives': [{'transcript': "polynomial regression is a special case of
the general linear regression this method is beneficial for describing
curvilinear relationships what is a curvilinear relationship it's what you get
by squaring or setting higher order terms of the predictor variables in the
model transforming the data the model can be quadratic which means the predictor
variable in the model is squared we use a bracket to indicated as an exponent
this is the second order polynomial regression with a figure representing the
function ",
     'confidence': 0.95}]},
  {'final': True,
   'alternatives': [{'transcript': 'the model can be cubic which means the
predictor variable is cute this is the third order polynomial regression we see
by examining the figure that the function has more variation ',
     'confidence': 0.95}]},
  {'final': True,
   'alternatives': [{'transcript': "there also exists higher order polynomial
regressions when a good fit hasn't been achieved by second or third order we can
see in figures how much the graphs change when we change the order of the
polynomial regression the degree of the regression makes a big difference and
can result in a better fit if you pick the right value in all cases the
relationship between the variable in the parameter is always linear ",
     'confidence': 0.91}]},
  {'final': True,
   'alternatives': [{'transcript': "let's look at an example from our data we
generate a polynomial regression model ",
     'confidence': 0.89}]},
  {'final': True,
   'alternatives': [{'transcript': 'in python we do this by using the poly fit
function in this example we develop a third order polynomial regression model
base we can print out the model symbolic form for the model is given by the
following expression ',
     'confidence': 0.92}]},
  {'final': True,
   'alternatives': [{'transcript': "negative one point five five seven X. one
```

```
cute plus two hundred four point eight X. one squared plus eight thousand nine
hundred sixty five X. one plus one point three seven times ten to the power of
five we can also have multi dimensional polynomial linear regression the
expression can get complicated here are just some of the terms for two
dimensional second order polynomial none pies poly fit function cannot perform
this type of regression we use the preprocessing librarian scikit learn to
create a polynomial feature object the constructor takes the degree of the
polynomial as a parameter then we transform the features into a polynomial
feature with the fit underscore transform method let's do a more intuitive
example ",
     'confidence': 0.9}]},
  {'final': True,
   'alternatives': [{'transcript': 'consider the feature shown here applying the
method we transform the data we now have a new set of features that are
transformed version of our original features as that I mention of the data gets
larger we may want to normalize multiple features as scikit learn instead we can
use the preprocessing module to simplify many tasks for example we can
standardize each feature simultaneously we import standard scaler we train the
object fit the scale object then transform the data into a new data frame on a
rate X. underscore scale there are more normalization methods available in the
pre processing library as well as other transformations we can simplify our code
by using a pipeline library there are many steps to getting a prediction for
example normalization polynomial transform and linear regression we simplify the
process using a pipeline ',
     'confidence': 0.9}]},
  {'final': True,
   'alternatives': [{'transcript': 'pipeline sequentially perform a series of
transformations the last step carries out a prediction first we import all the
modules we need then we import the library pipeline we create a list of topples
the first element in the topple contains the name of the estimator model the
second element contains model constructor we input the list in the pipeline
constructor we now have a pipeline object we can train the pipeline by applying
the train method to the pipeline object we can also produce a prediction as well
',
     'confidence': 0.89}]},
  {'final': True,
   'alternatives': [{'transcript': 'the method normalizes the data performs a
polynomial transform then outputs a prediction ',
     'confidence': 0.89}]}]}
```

[10]:
```
from pandas.io.json import json_normalize

json_normalize(response.result['results'],"alternatives")
```

```
/home/jupyterlab/conda/envs/python/lib/python3.6/site-
packages/ipykernel_launcher.py:3: FutureWarning: pandas.io.json.json_normalize
is deprecated, use pandas.json_normalize instead
```

```
  This is separate from the ipykernel package so we can avoid doing imports
until
```

[10]:

	transcript	confidence
0	in this video we will cover polynomial regress…	0.94
1	what do we do when a linear model is not the b…	0.90
2	polynomial regression is a special case of the…	0.95
3	the model can be cubic which means the predict…	0.95
4	there also exists higher order polynomial regr…	0.91
5	let's look at an example from our data we gene…	0.89
6	in python we do this by using the poly fit fun…	0.92
7	negative one point five five seven X. one cute…	0.90
8	consider the feature shown here applying the m…	0.90
9	pipeline sequentially perform a series of tran…	0.89
10	the method normalizes the data performs a poly…	0.89

[11]:
```
response
```

[11]:
```
<ibm_cloud_sdk_core.detailed_response.DetailedResponse at 0x7fed2003feb8>
```

We can obtain the recognized text and assign it to the variable recognized_text:

[12]:
```
recognized_text=response.result['results'][0]["alternatives"][0]["transcript"]
type(recognized_text)
```

[12]:
```
str
```

Language Translator

First we import LanguageTranslatorV3 from ibm_watson. For more information on the API click here

[13]:
```
from ibm_watson import LanguageTranslatorV3
```

The service endpoint is based on the location of the service instance, we store the information in the variable URL. To find out which URL to use, view the service credentials.

[14]:
```
url_lt='https://gateway.watsonplatform.net/language-translator/api'
```

You require an API key, and you can obtain the key on the Dashboard.

[15]:
```
apikey_lt='3OAEwzXyvgl2giuPVCV15NJsqPjj3OhvKL_pXbienqYr'
```

API requests require a version parameter that takes a date in the format version=YYYY-MM-DD. This lab describes the current version of Language Translator, 2018-05-01

[16]:
```
version_lt='2018-05-01'
```

we create a Language Translator object language_translator:

[17]:
```
authenticator = IAMAuthenticator(apikey_lt)
language_translator =␣
 ↪LanguageTranslatorV3(version=version_lt,authenticator=authenticator)
language_translator.set_service_url(url_lt)
language_translator
```

[17]:
```
<ibm_watson.language_translator_v3.LanguageTranslatorV3 at 0x7feccf84dc18>
```

We can get a Lists the languages that the service can identify. The method Returns the language code. For example English (en) to Spanis (es) and name of each language.

[18]:
```
from pandas.io.json import json_normalize

json_normalize(language_translator.list_identifiable_languages().get_result(),␣
 ↪"languages")
```

```
/home/jupyterlab/conda/envs/python/lib/python3.6/site-
packages/ipykernel_launcher.py:3: FutureWarning: pandas.io.json.json_normalize
is deprecated, use pandas.json_normalize instead
  This is separate from the ipykernel package so we can avoid doing imports
until
```

[18]:

	language	name
0	af	Afrikaans
1	ar	Arabic
2	az	Azerbaijani
3	ba	Bashkir
4	be	Belarusian
..	…	…
71	uk	Ukrainian
72	ur	Urdu
73	vi	Vietnamese
74	zh	Simplified Chinese
75	zh-TW	Traditional Chinese

[76 rows x 2 columns]

We can use the method translate this will translate the text. The parameter text is the text. Model_id is the type of model we would like to use use we use list the the langwich . In this case, we set it to 'en-es' or English to Spanish. We get a Detailed Response object translation_response

[19]:
```
translation_response = language_translator.translate(\
    text=recognized_text, model_id='en-es')
translation_response
```

[19]:
```
<ibm_cloud_sdk_core.detailed_response.DetailedResponse at 0x7feccf8586a0>
```

The result is a dictionary.

[20]:
```
translation=translation_response.get_result()
translation
```

[20]:
```
{'translations': [{'translation': 'en este vídeo cubriremos la regresión
polinómica y las tuberías '}],
 'word_count': 10,
 'character_count': 64}
```

We can obtain the actual translation as a string as follows:

[21]:
```
spanish_translation =translation['translations'][0]['translation']
spanish_translation
```

[21]:
```
'en este vídeo cubriremos la regresión polinómica y las tuberías '
```

We can translate back to English

[22]:
```
translation_new = language_translator.translate(text=spanish_translation␣
↪,model_id='es-en').get_result()
```

We can obtain the actual translation as a string as follows:

[23]:
```
translation_eng=translation_new['translations'][0]['translation']
translation_eng
```

[23]:
```
'in this video we will cover the polynomial regression and the pipes '
```

We can convert it to french as well:

[24]:
```
French_translation=language_translator.translate(
    text=translation_eng , model_id='en-fr').get_result()
```

[25]:
```
French_translation['translations'][0]['translation']
```

[25]:
```
'Dans cette vidéo nous couvrons la régression polynomiale et les tuyaux '
```

Language Translator

References

https://cloud.ibm.com/apidocs/speech-to-text?code=python

https://cloud.ibm.com/apidocs/language-translator?code=python

About the Author:

Joseph Santarcangelo has a PhD in Electrical Engineering, his research focused on using machine learning, signal processing, and computer vision to determine how videos impact human cognition. Joseph has been working for IBM since he completed his PhD.

Other contributor: Fan Jiang