Google Cloud Speech

Google Cloud Platform

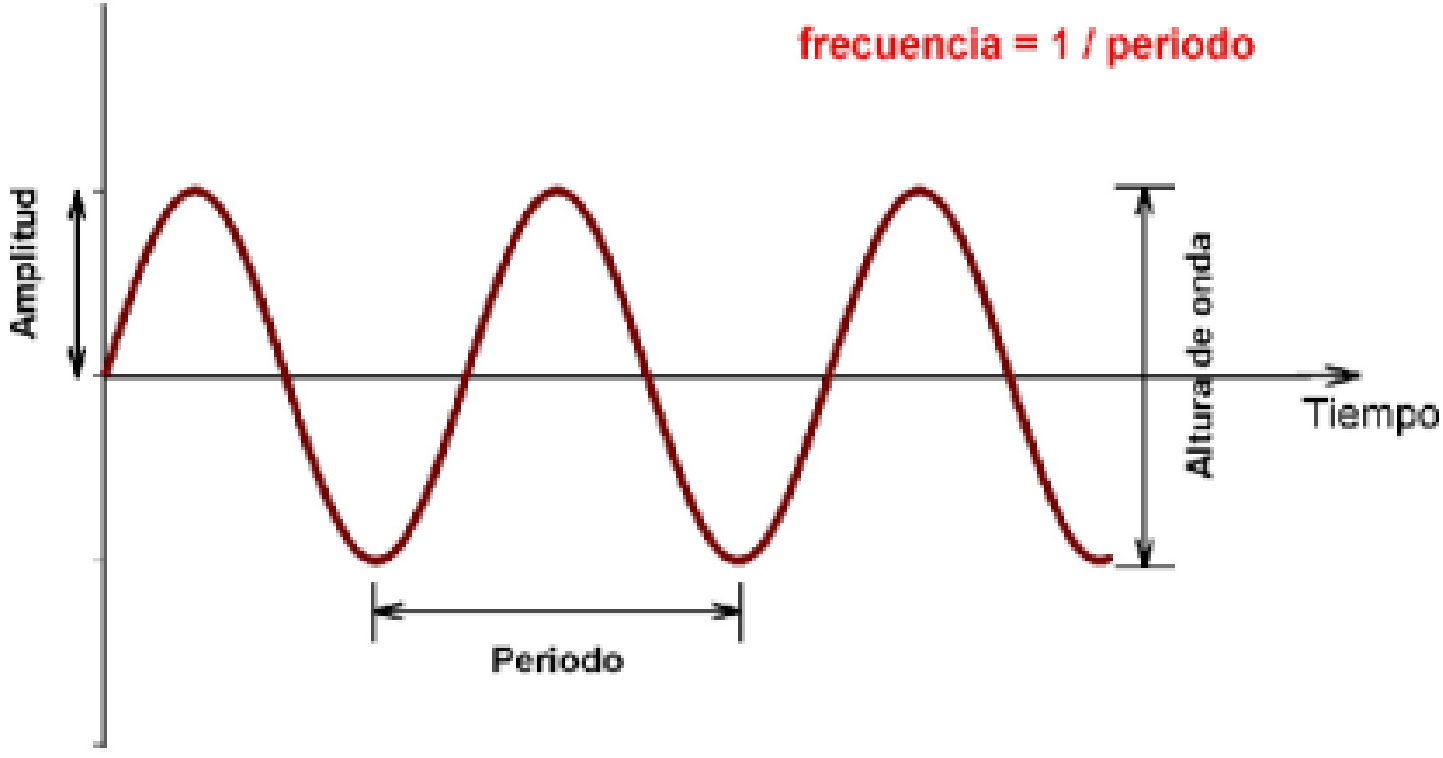
API to convert audio to test

Created by Sergio Rodríguez (@serrodcal)

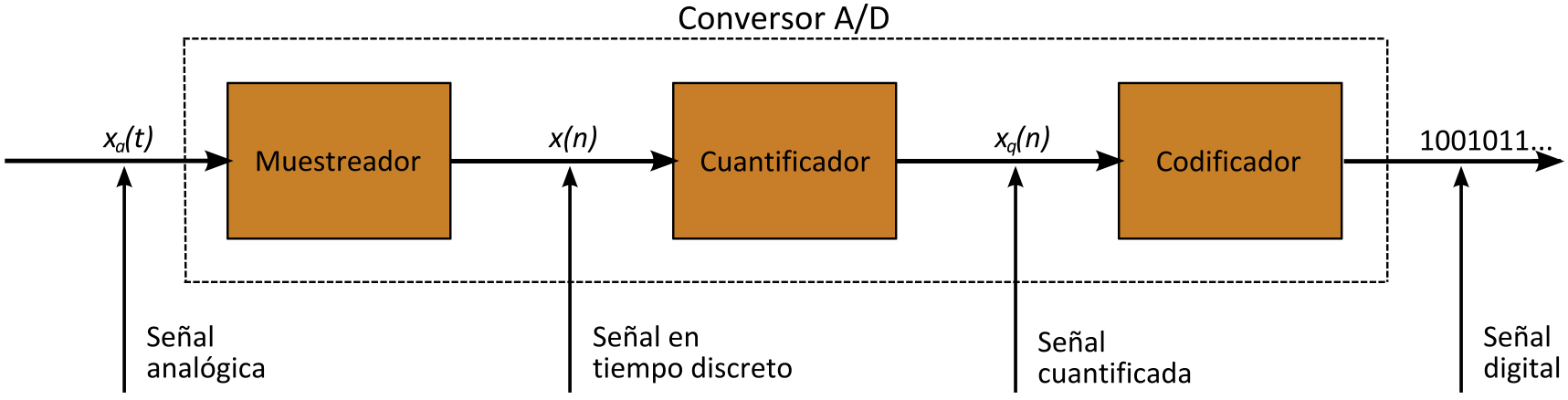
Introduction

- API to convert audio to test.
- Powerful neural network models in an easy to use API.
- Recognizes over 80 languages and variants.
- Accuracy improves over time.
- Return text result in real-time.
- Accurate in noisy environments.

Sound


frecuencia = 1 / periodo
Amplitud
Altura de onda
Tiempo
Periodo

A/D converter



Audio encoding

It has the following parameters:

- Channel.
- Sampling.
- Digital resolution.
- Bit rate.
- Lossy compression.

Audio formats

- Note that audio format is not equivalent to audio encoding.
- Defines the format of the header of an audio file.
- Don't assume a kind of file has any particular encoding until you inspect its header.
- However, there is a 'formart' like `FLAC` which is both a file format and an encoding.

Ways to use the API

- REST.
- Client libraries.
- RPC.

API Rest

- All URIs below are relative to https://speech.googleapis.com
- This service provides the following discovery document: https://speech.googleapis.com/$discovery/rest?version=v1

Speech Recognition

- Performs asynchronous speech recognition.
- `POST /v1/speech:longrunningrecognize`

```
{
  "config": { #information to the recognizer
    object(RecognitionConfig)
  },
  "audio": { # audio data
    object(RecognitionAudio)
  },
}
```

Speech Recognition

- Performs synchronous speech recognition.
- `POST /v1/speech:recognize`

```
{
  "config": { #information to the recognizer
    object(RecognitionConfig)
  },
  "audio": { # audio data
    object(RecognitionAudio)
  },
}
```

Speech Recognition

- In longrunningrecognize the response contains an instance of `Operation`.
- In recognize the response is:

```
{
  "results": [ # Array for alternatives
    {
      object(SpeechRecognitionResult)
    }
  ],
}
```

Speech Recognition

- In asynchronous, we can get the latest status of a long-running operation.
- `GET /v1/operations/{name}`

Speech Recognition

- Response for operation resource:

```
{
  "name": string,
  "metadata": {
    "@type": string,
    field1: ...,
    ...
  },
  "done": boolean,
  "error": { # error case, done equals to false
    object(Status)
  },
  "response": { # success case, done equals to true
    "@type": string,
    field1: ...,
    ...
  }, # Could be several responses
}
```

Client libraries

- There are several client libraries: C#, GO, Java, Node.js, PHP, Python and Ruby.
- In python we must use PIP to install the client library:

```
pip install --upgrade google-cloud-speech
```

- In Java we must use Maven o Gradle to provide the dependency for our project:

```
<dependency>
    <groupId>com.google.cloud</groupId>
    <artifactId>google-cloud-speech</artifactId>
    <version>0.17.1-alpha</version>
</dependency>
```

Client libraries

- At now, just import it.

```
# Imports the Google Cloud client library
from google.cloud import speech

# Instantiates a client
speech_client = speech.Client()

# Rest of code
```

Code

```
import io
import os
from google.cloud import speech

class Audio_2_Text(object):

    def __init__(self, file_name, audio_lang):
        self.speech_client = speech.Client()
        self.file_name = file_name
        self.audio = None
        self.audio_lang = audio_lang
```

Code

```
def find_file(self):
    file_name = os.path.join(
        os.path.dirname(__file__),
        '.',
        self.file_name
    )
```

Code

```
def load_audio(self):
    with io.open(self.file_name, 'rb') as audio_file:
        content = audio_file.read()
        self.audio = speech_client.sample(
            content,
            source_uri=None,
            encoding='LINEAR16',
            sample_rate_hertz=16000
        )
```

Code

```
def recognize(self):
    alternatives = audio_sample.recognize(
                                            self.audio_lang)
    return '{}'.format(alternatives[0].transcript)
```

Code

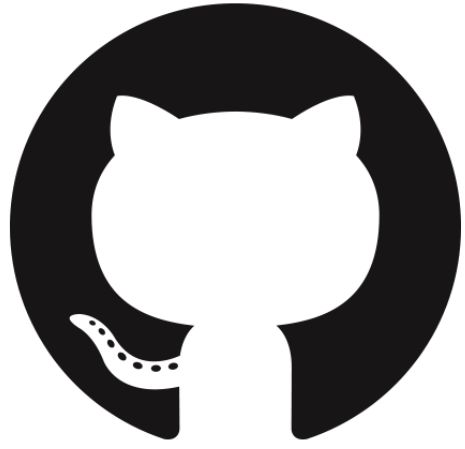
```
if __name__ == "__main__":

    ...

    audio_2_text = Audio_2_Text(args.audio_file,
                                            audio_lang)
    audio_2_text.find_file()
    audio_2_text.load_audio()
    text_to_predict = audio_2_text.recognize()

    ...
```

More examples



More examples aviables in Github

Behind Google Cloud Speech

- Tensor flow models on Clouds TPUs Technology.

Python integration demo

Any questions?

email: sergiorodriguezcalvo@gmail.com