GTTS-EHU System for IberSPEECH-RTVE 2018 Speech to Text Transcription Challenge

Mikel Penagarikano, Amparo Varona, Luis J. Rodríguez-Fuentes, Germán Bordel

GTTS Group (http://gtts.ehu.es), Department of Electricity and Electronic
University of the Basque Country UPV/EHU, 48940, Leioa, Spain

mikel.penagarikano@ehu.eus

Abstract

This paper describes the steps to install and use the Google Cloud Platform [1] tools and the Google Cloud Speech-to-Text API [2] to run a state-of-the-art speech-to-text system. The submission of the Software Technology Working Group (http://gtts.ehu.es) of the University of the Basque Country (EHU) for the IberSPEECH-RTVE 2018 Speech to Text Transcription Challenge was just the output of this commercial system.

The simplicity of use, economic cost and performance of Google's commercial system makes it suitable to be used as a reference/baseline system by any research group working on Speech Technologies.

Index Terms: speech recognition, speech-to-text, google cloud platform, google cloud speech-to-text

1. Introduction

The effort needed to build a competitive Speech-to-Text system, has been drastically reduced thanks to open source projects that implement state-of-the-art technologies [3, 4, 5, 6, 7]. However, the skills, time and amount of data needed to train and fine tune such systems often goes beyond the capabilities and competencies of many research groups.

Commercial systems can play an important role if they are cheap, competitive and easy to handle (they offer a nice interface). This paper describes the steps to install and use the Google Cloud Platform [1] tools and the Google Cloud Speech-to-Text API [2] to run a state-of-the-art speech-to-text system. The submission of the Software Technology Working Group (http://gtts.ehu.es) of the University of the Basque Country (EHU) for the IberSPEECH-RTVE 2018 Speech to Text Transcription Challenge was just the output of this commercial system.

2. Sign in into Google Cloud and activate the Speech-to-Text API

The first step is to sign in Google Cloud (`https://cloud.google.com/`) using a Google account. Once logged, we will use the Google Cloud console (`https://console.cloud.google.com` to create a new project (by default, the system will suggest us to create a project entitled *Tutorial Project*). Any activity on the platform must be linked to a project.

In order to activate the Speech-to-Text API, we must first add a payment method. The accounts are initialized with a gift balance of $300, so we will not have to pay any amount as long as we do not spend that balance. Once the payment method has been configured, we can activate the Speech-to-Text API from the console.

3. Install the Google Cloud tools

There are three major ways to interact with Google cloud:

1. The Google Cloud console (web based).
2. Using the command line (cloud-sdk based).
3. Using a programmatic API (c++, c#, go, Java, node.js, PHP, Python and Ruby).

The command line option is simpler than using a specific language's API and allows for a minimum automatization of the tasks, so it was the one selected for this work.

The only software requirement for the rest of this paper will be Docker [8], a lightweight linux container software that runs on almost all OS (for more info on how to install Docker, see `www.docker.com`). Thanks to the Docker container technology, the complexity of running a lot of software packages is minimal and can be reproduced from almost any host OS.

In this paper, we will use the Google Cloud SDKbundle image (see `hub.docker.com/r/google/cloud-sdk`).

3.1. Create an authenticated docker container

Any access to Google Cloud services must be authenticated. In order to simplify the authentication method, we can create a docker volume that will ask for the credentials:

```
# docker run -ti --name gcloud-config \
    google/cloud-sdk gcloud auth login
```

Once the authentication proccess is successfully finished, the credentials are preserved in the volume of the *gcloud-config* container. This container can be used by subsequent commands or sessions:

```
# docker run --rm -ti --volumes-from \
    gcloud-config google/cloud-sdk gcloud help

NAME
 gcloud - manage Google Cloud Platform resources
 and developer workflow

SYNOPSIS
 gcloud GROUP | COMMAND [--account=ACCOUNT]
  [--configuration=CONFIGURATION]
  [--flatten=[KEY,...]] [--format=FORMAT]
  [--help] [--project=PROJECT_ID] [--quiet, -q]
  [--verbosity=VERBOSITY; default="warning"]
  [--version, -v] [-h] [--log-http]
  ...
```

3.2. Grant access to the host file system

If we run a docker container without any command, the container executes a shell:

```
$ docker run --rm -ti --volumes-from \
   gcloud-config google/cloud-sdk
root@docker:/#
```

Note that the container is an isolated environment. If we need to have access to the host filesystem from the container, we must map a local directory to a directory inside the container:

```
$ mkdir data && touch data/myfile
$ docker run --rm -ti --volumes-from \
   gcloud-config -v $(pwd)/data:/mnt \
   google/cloud-sdk
root@docker:/# ls -l /mnt
total 1
-rw-r--r-- 1 102 101 0 Oct 21 09:44 myfile
```

4. Database preparation

The IberSPEECH-RTVE 2018 Speech to Text Transcription Challenge test dataset contains 59 audio files with the *Advanced Audio Coding*. Google's Speech-to-Text API does not handle this coding, being *Free Lossless Audio Codec* (FLAC) the one that best fits. Note that *FLAC* is both a codec and a format, and therefore more concise. *ffmpeg* [9] is a powerful software for audio and video processing. It is posible to run it using Docker too. We can convert the `data/src.aac` file into a `data/dst.flac` file simply by:

```
$ docker run --rm -ti -v $(pwd)/data:/mnt \
   jrottenberg/ffmpeg -i /mnt/src.aac \
   /mnt/dst.flac
```

Note that the audio files are quite long (longer than an hour). Such long files could generate execution errors in the Google Cloud platform. We could simply split them into chunks of 1000 seconds to reduce the probability of error:

```
$ docker run --rm -ti -v $(pwd)/data:/mnt \
   jrottenberg/ffmpeg -i /mnt/src.aac \
   -f segment -segment_time 1000 \
   /mnt/dst%03d.flac
```

5. Upload the data to Google Cloud Storage

In order to upload the data to Google Cloud Storage, we need to create a storage bucket. Storage buckets are tied to Cloud's projects and they share a single namespace among all the gcloud users.

All the subsequent commands will be executed from the same Docker session. First we open a shell in a new container that uses the credentials stored on the previously created *gcloud-config* container:

```
$ docker run --rm -ti --volumes-from \
   gcloud-config -v $(pwd)/data:/mnt \
   google/cloud-sdk
root@docker:/#
```

Next, we set the current project (the project *id* can be obtained from the web console):

```
root@docker:/# gcloud config set project \
   my-project-id
```

And create a bucket (set `my-personal-name` to any specific name):

```
root@docker:/# gsutil mb gs://my-personal-name
```

Once we have created the bucket, we can upload all the audio files to the Google Cloud Storage:

```
root@docker:/# gsutil cp /mnt/*.flac \
  gs://my-personal-name/
```

6. Run the Speech-to-Text API

The S2T command needs just the URL of the input audio file and the code of the target language:

```
root@docker:/# gcloud ml speech \
  recognize-long-running \
  gs://my-personal-name/file.flac \
  --language-code='es-ES'
```

The output of the command (if succesful) is a JSON file containing the transcription of the audio file.

7. Procesing time

The test dataset was sequentially processed on 9 hours and 40 minutes.

8. Google Cloud expenses

The total cost of the processing was less than 60€ (0,02 €/minute + VAT).

9. Conclusions

Google Cloud based Speech-to-Text API is simple to use, has no hardware requirements and runs a service based billing (i.e. based on the processed audio lengths). Its performance makes it suitable to be used as a reference/baseline system by any research group working on Speech Technologies.

10. References

[1] "Google cloud platform," https://cloud.google.com/, (Accessed on 10/28/2018).

[2] "Google cloud speech-to-text api," https://cloud.google.com/speech-to-text/, (Accessed on 10/28/2018).

[3] S. Young, "The HTK hidden Markov model toolkit: Design and philosophy," *Entropic Cambridge Research Laboratory, Ltd*, vol. 2, pp. 2–44, 1994.

[4] P. Woodland, "An overview of HTK v3.5," Tech. Rep., phil Woodland's keynote talk on HTK v3.5 at the fourth meeting of the UK and Irish speech science and technology research community, University of East Anglia.

[5] K.-F. Lee, "Large-vocabulary speaker-independent continuous speech recognition: The Sphinx system," Ph.D. dissertation, Pittsburgh, PA, USA, 1988, aAI8826533.

[6] W. Walker, P. Lamere, P. Kwok, B. Raj, R. Singh, E. Gouvea, P. Wolf, and J. Woelfel, "Sphinx-4: A flexible open source framework for speech recognition," Sun Microsystems, Tech. Rep. TR-2004-139, 2004.

[7] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely, "The Kaldi speech recognition toolkit," in *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, Dec. 2011, iEEE Catalog No.: CFP11SRW-USB.

[8] D. Merkel, “Docker: Lightweight linux containers for consistent development and deployment,” *Linux J.*, vol. 2014, no. 239, Mar. 2014. [Online]. Available: http://dl.acm.org/citation.cfm?id=2600239.2600241

[9] Ffmpeg, “Ffmpeg,” 2010. [Online]. Available: http://www.ffmpeg.org