Google Cloud Natural Language API

Overview:

Google Cloud Natural Language API is Google's pre-trained natural language processing API with various features. The Natural Language API has several methods for performing analysis and annotation on your text. Each level of analysis provides valuable information for language understanding. These methods are listed below:

- Sentiment analysis: inspects the given text and identifies the prevailing emotional opinion within the text, especially to determine a writer's attitude as positive, negative, or neutral. Sentiment analysis is performed through the analyzeSentiment method.
- Entity analysis: inspects the given text for known entities (Proper nouns such as public figures, landmarks, and so on. Common nouns such as restaurant, stadium, and so on.) and returns information about those entities. Entity analysis is performed with the analyzeEntities method.
- Entity sentiment analysis: inspects the given text for known entities (proper nouns and common nouns), returns information about those entities, and identifies the prevailing emotional opinion of the entity within the text, especially to determine a writer's attitude toward the entity as positive, negative, or neutral. Entity analysis is performed with the analyzeEntitySentiment method.
- Syntactic analysis: extracts linguistic information, breaking up the given text into a series of sentences and tokens (generally, word boundaries), providing further analysis on those tokens. Syntactic Analysis is performed with the analyzeSyntax method.
- Content classification: analyzes text content and returns a content category for the content. Content classification is performed by using the classifyText method.

This API does not offer any feature that can extract labels from given audio or video. Therefore, Google's Natural Language API will not be considered.

Open Source

[1] Fairseq (PyTorch, python)
Facebook AI Research Sequence to Sequence Toolkit. Supports parallel training.
https://github.com/pytorch/fairseq
https://fairseq.readthedocs.io/en/latest/

[2] Wav2Letter++ (C++)
Wav2Letter++ was released by Facebook’s AI Research Team. It needs you first to build a training model for the language you desire by yourself to train the algorithms. No pre-built support of any language (including English) is available. It’s just a machine-learning-driven tool to convert speech to text.
https://github.com/flashlight/wav2letter

[3] ESPnet (torch, python)

An end-to-end speech processing toolkit. It gives very good performance in many benchmarks and supports for other tasks such as machine translation and speech translation.
https://github.com/espnet/espnet

[4] Kaldi (C++, Bash)
Kaldi is a toolkit for speech recognition, intended for use by speech recognition researchers and professionals. It's extendable and modular, has a Python pre-built engine with trained models ready to use and many 3rd-party modules.
https://github.com/kaldi-asr/kaldi
http://kaldi-asr.org/doc/

[5] Pytorch-kaldi
PyTorch-Kaldi is an open-source repository for developing state-of-the-art DNN/HMM speech recognition systems. The DNN part is managed by PyTorch, while feature extraction, label computation, and decoding are performed with the Kaldi toolkit.
https://github.com/mravanelli/pytorch-kaldi

[6] Athena (tensorflow, python)
Athena is an open-source implementation of end-to-end speech processing engine. Supports unsupervised pre-training and multi-GPUs processing.
https://github.com/athena-team/athena

[7] eesen (C++)
Eesen is to simplify the existing complicated, expertise-intensive ASR pipeline into a straightforward sequence learning problem.
https://github.com/srvk/eesen

[8] PIKA
PIKA is a lightweight speech processing toolkit based on Pytorch and (Py)Kaldi. Pytorch as deep learning engine, Kaldi for data formatting and feature extraction.
https://github.com/tencent-ailab/pika
https://mp.weixin.qq.com/s/BJMLbzdjTeBEKSKAG9r-Mg

[9] DeepSpeech (tensorflow, C++, python)
Made by Mozilla, the organization behind the Firefox browser. It can be used to build training models by yourself to get better results.
https://github.com/mozilla/DeepSpeech

[10] DeepSpeech2 (python)
Researchers at Baidu built it. It uses the "PaddlePaddle" deep learning framework for converting both English & Mandarin Chinese languages speeches into text. The engine can be trained on any model and for any language you desire. The models are not released with the code. You'll have to build them yourself.
https://github.com/PaddlePaddle/DeepSpeech

[11] Julius (C)
Able to perform real-time STT processes, low memory usage (Less than 64MB for 20000 words). Mainly for academic and research usage. The software is probably available to install easily using your Linux distribution's repository.
https://github.com/julius-speech/julius

[12] Vosk
Started in 2020. It is ready to use after installation with portable 50MB-sized models to up to 1.4GB models already available to use. It works on iOS and Android devices with a streaming API to connect. Vosk has bindings for Java, Python, JS, C# and NodeJS.
https://alphacephei.com/vosk/

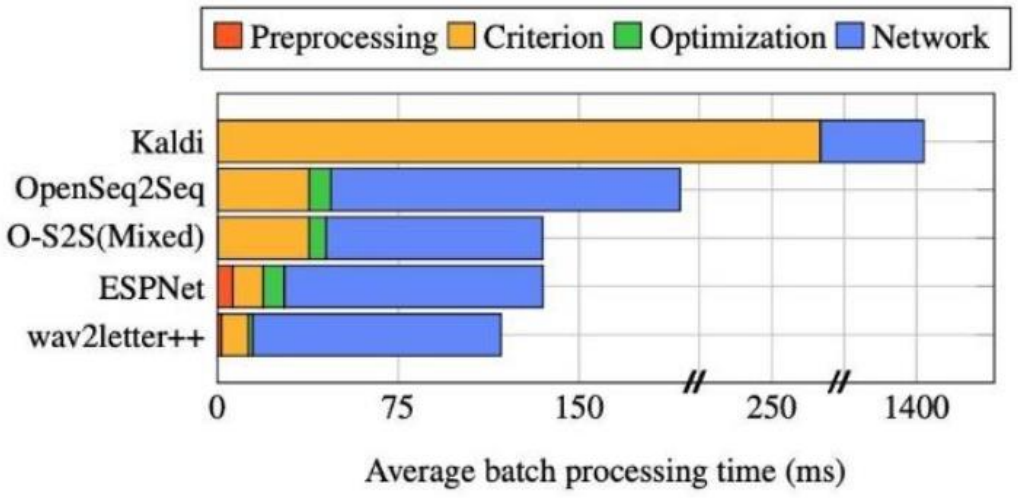
Vosk is is written in Python and works on iOS, android and Raspberry pi too, and supports up to 10 languages. It also provides a huge training dataset if you shall need it, and a smaller one for portable applications.

Any of Fairseq, OpenSeq2Seq, Athena and ESPnet should be more than enough to train and build own models, and they are the most modern state-of-the-art toolkits.

As for Mozilla's DeepSpeech, it lacks a lot of features behind its other competitors in this list. And its future is concerning as Mozilla restructure.
Traditionally, Julius and Kaldi are also very much cited in the academic literature.

The difference between proprietary speech recognition and open source speech recognition, is that the library used to process the voices should be licensed under one of the known open source licenses, such as GPL, MIT and others.

Benefits: Mainly, you get few or no restrictions at all on the commercial usage for your application, as the open source speech recognition libraries will allow you to use them for whatever use case you may need.
Also, most – if not all – open source speech recognition toolkits in the market are also free of charge, saving you tons of money instead of using the proprietary ones.

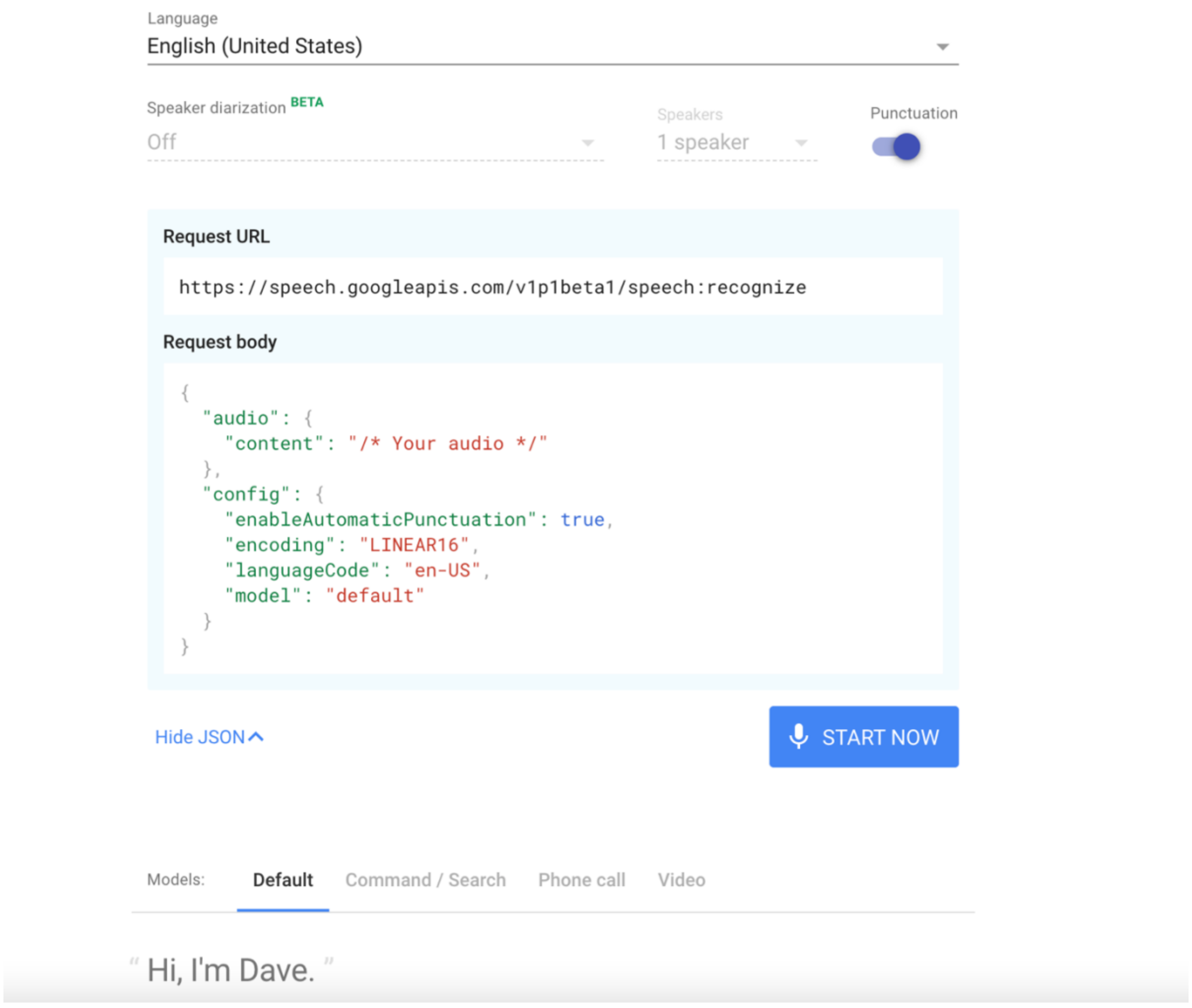


Google Speech to Text Service

Overview:

This service can accurately convert speech into text using an API powered by Google’s AI technologies.

For example, we have uploaded a short audio, and the AI can automatically convert it into an JSON file.



Request URL

```
https://speech.googleapis.com/v1p1beta1/speech:recognize
```

Request body

```
{
  "audio": {
    "content": "/* Your audio */"
  },
  "config": {
    "enableAutomaticPunctuation": true,
    "encoding": "LINEAR16",
    "languageCode": "en-US",
    "model": "default"
  }
}
```

“ Hi, I'm Dave. ”

As the figure show above, the audio "Hi, I'm Dave." has been successfully converted into text, and the JSON file produced contains configurations of the data such as the language, the model used and the encoding method.

We could use this service to pre process the video by convert it into text files for further analysis.

Amazon Comprehend

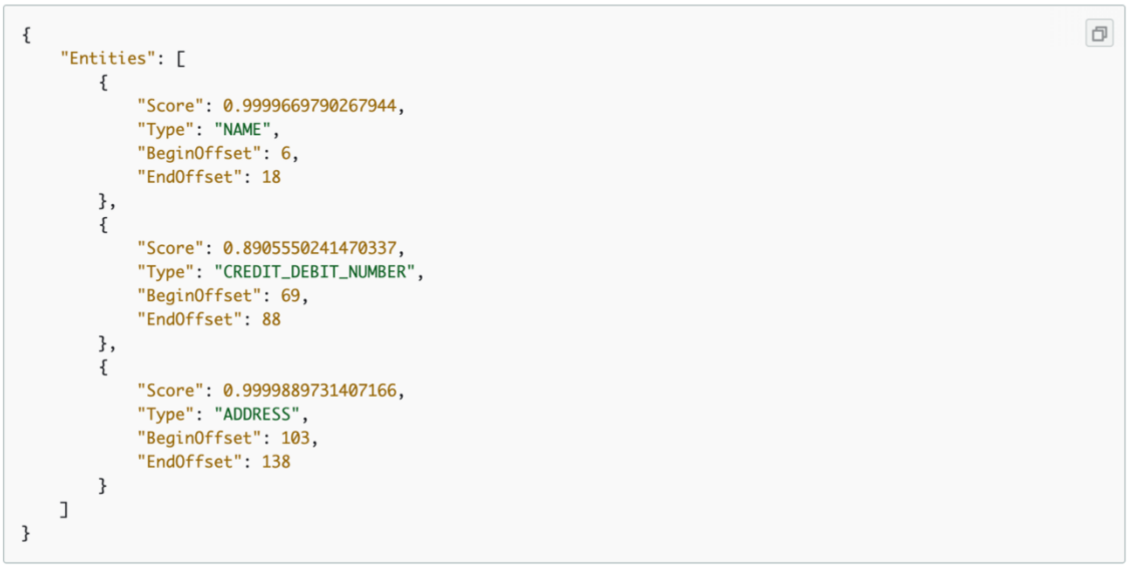
Overview:

Amazon Comprehend uses natural language processing (NLP) to extract insights about the content of documents. Amazon Comprehend processes any text file in UTF-8 format. It develops insights by recognizing the entities, key phrases, language, sentiments, and other common elements in a document. Use Amazon Comprehend to create new products based on understanding the structure of documents. For example, using Amazon Comprehend you can search social networking feeds for mentions of products or scan an entire document repository for key phrases.

Some of the features that can be used in project:

- Detect Events: Analyse text documents to detect specific type of events or related details.
- Detect Key Phrases: Amazon comprehend will give each key phrase a score which you can use to determine if the detection has high enough confidence for you to be used.
- Detect PII: Amazon Comprehend can identify entities in the text that contain personal identifiable information.

The API will produce a JSON file that contains all extracted labels and information regarding the confidence level.



```
{
    "Entities": [
        {
            "Score": 0.9999669790267944,
            "Type": "NAME",
            "BeginOffset": 6,
            "EndOffset": 18
        },
        {
            "Score": 0.8905550241470337,
            "Type": "CREDIT_DEBIT_NUMBER",
            "BeginOffset": 69,
            "EndOffset": 88
        },
        {
            "Score": 0.9999889731407166,
            "Type": "ADDRESS",
            "BeginOffset": 103,
            "EndOffset": 138
        }
    ]
}
```