
Botium Speech Processing

license MIT PR welcome Awesome for sure!

Botium Speech Processing is a unified, developer-friendly API to the best available free and Open-Source Speech-To-Text and Text-To-Speech services.

What is it ?

Botium Speech Processing is a *get-shit-done*-style Open-Source software stack, the configuration options are rudimentary: it is highly opinionated about the included tools, just get the shit done.

- With Kaldi a reasonable speech recognition performance is available with freely available data sources.
- MaryTTS is currently the best freely available speech synthesis software
- SoX is the *swiss army-knife* for audio file processing

While the included tools in most cases cannot compete with the big cloud-based products, for lots of applications the trade-off between price and quality is at least reasonable.

Read about the project history here

Possible Applications

Some examples what you can do with this:

- Synthesize audio tracks for Youtube tutorials
- Build voice-enabled chatbot services (for example, IVR systems)
 - see the Rasa Custom Voice Channel
- Classification of audio file transcriptions
- Automated Testing of Voice services with Botium

Installation

Software and Hardware Requirements

- 8GB of RAM (accessible for Docker) and 40GB free HD space (for full installation)
- Internet connectivity
- docker

-
- docker-compose

Note: memory usage can be reduced if only one language for Kaldi is required - default configuration comes with two languages.

Full Installation (Prebuilt Docker Images)

Clone or download this repository and start with docker-compose:

```
1 > docker-compose up -d
```

This will download the latest released prebuilt images from Dockerhub. To download the latest developer images from Dockerhub:

```
1 > docker-compose --env-file .env.develop up
```

Point your browser to <http://127.0.0.1> to open the Swagger UI and browse/use the API definition.

Slim Cloud-Specific Installation (Prebuilt Docker Images)

For the major cloud providers there are additional docker-compose files. If using those, the installation is more slim, as there is only the *frontend*-service required. For instance, add your Azure subscription key and Azure region key to the file *docker-compose-azure.yml* and start the services:

```
1 > docker-compose -f docker-compose-azure.yml up -d
```

Optional: Build Docker Images

You can optionally build your own docker images (if you made any changes in this repository, for instance to download the latest version of a model). Clone or download this repository and run docker-compose:

```
1 > docker-compose -f docker-compose-dev.yml up -d
```

This will take some time to build.

Configuration

This repository includes a reasonable default configuration:

- Use MaryTTS for TTS

-
- Use Kaldi for STT
 - Use SoX for audio file conversion
 - Languages included:
 - German
 - English

Configuration changes with environment variables. See comments in this file.

Recommendation: Do not change the `.env` file but create a `.env.local` file to overwrite the default settings. This will prevent troubles on future *git pull*

Request-Specific Configuration

If there is a JSON-formatted request body, or a multipart request body, certain sections are considered:

- **credentials** to override the server default credentials for cloud services
- **config** to override the server default settings for the cloud API calls

See samples below

Securing the API

The environment variable `BOTIUM_API_TOKENS` contains a list of valid API Tokens accepted by the server (separated by whitespace or comma). The HTTP Header `BOTIUM_API_TOKEN` is validated on each call to the API.

Caching

For performance improvements, the result of the speech-to-text and text-to-speech calls are cached (by MD5-hash of audio or input text). To enforce reprocessing empty the cache directories:

- `frontend/resources/.cache/stt`
- `frontend/resources/.cache/tts`

Testing

Point your browser to `http://127.0.0.1/` to open Swagger UI to try out the API.

Point your browser to `http://127.0.0.1/dictate/` to open a rudimentary dictate.js-interface for testing speech recognition (*for Kaldi only*)

Attention: in Google Chrome this only works with services published as HTTPS, you will have to take of this yourself. For example, you could publish it via ngrok tunnel.

Point your browser to `http://127.0.0.1/tts` to open a MaryTTS interface for testing speech synthesis.

Real Time API

It is possible to stream audio from real-time audio decoding: Call the `/api/sttstream/{language}` endpoint to open a websocket stream, it will return three urls:

- `wsUri` - the Websocket uri to stream your audio to. By default, it accepts wav-formatted audio-chunks
- `statusUri` - check if the stream is still open
- `endUri` - end audio streaming and close websocket

File System Watcher

Place audio files in these folders to receive the transcript in the folder `watcher/stt_output`:

- `watcher/stt_input_de`
- `watcher/stt_input_en`

Place text files in these folders to receive the synthesized speech in the folder `watcher/tts_output`:

- `watcher/tts_input_de`
- `watcher/tts_input_en`

API Definition

See `swagger.json`:

- HTTP POST to `/api/stt/{language}` for Speech-To-Text

```
curl -X POST "http://127.0.0.1/api/stt/en" -H "Content-Type: audio/wav" -T sample.wav
```
- HTTP POST to `/api/stt/{language}` for Speech-To-Text with Google, including credentials

```
curl -X POST "http://127.0.0.1/api/stt/en-US?stt=google" -F "google={\"credentials\": {\"private_key\": \"xxx\", \"client_email\": \"xxx\"}}\" -F content=@sample.wav
```

- HTTP POST to **/api/stt/{language}** for Speech-To-Text with Google, including switch to MP3 encoding

```
curl -X POST "http://127.0.0.1/api/stt/en-US?stt=google" -F "google={\"config\": {\"encoding\": \"MP3\"}}\" -F content=@sample.mp3
```

- HTTP POST to **/api/stt/{language}** for Speech-To-Text with IBM, including credentials

```
curl -X POST "http://127.0.0.1/api/stt/en-US?stt=ibm" -F "google={\"credentials\": {\"apikey\": \"xxx\", \"serviceUrl\": \"xxx\"}}\" -F content=@sample.wav
```

- HTTP GET to **/api/tts/{language}?text=...** for Text-To-Speech

```
curl -X GET "http://127.0.0.1/api/tts/en?text=hello%20world" -o tts.wav
```

- HTTP POST to **/api/convert/{profile}** for audio file conversion

```
curl -X POST "http://127.0.0.1/api/convert/mp3tomonowav" -H "Content-Type: audio/mp3" -T sample.mp3 -o sample.wav
```

Contributing

To be done: contribution guidelines.

We are open to any kind of contributions and are happy to discuss, review and merge pull requests.

Big Thanks

This project is standing on the shoulders of giants.

- **Kaldi GStreamer server** and **Docker images**
- **MaryTTS**
- **SVOX Pico Text-to-Speech**
- **Kaldi**
- **Kaldi Tuda Recipe**
- **Zamia Speech**
- **Deepspeech** and **Deepspeech German**
- **SoX**
- **dictate.js**

Changelog

2022-03-06

- Voice effects to consider audio file length

2022-02-28

- Applied Security Best Practices (not run as root user)

2022-01-12

- Added support for Azure Speech Services

2021-12-07

- Added endpoints for streaming audio and responses

2021-12-01

- Added option to hand over cloud credentials in request body

2021-01-26

- Added several profiles for adding noise or other audio artifacts to your files
- Added custom channel for usage with Rasa

2020-12-18

- Adding support for Google Text-To-Speech
- Adding support for listing and using available TTS voices
- Added sample docker-compose configurations for PicoTTS and Google

2020-03-05

- Optional *start/end* parameters for audio file conversion to trim an audio file by time codes formatted as mm:ss (01:32)

2020-02-22

- Additional endpoint to calculate the Word Error Rate (Levenshtein Distance) between two texts
- When giving the *hint*-parameter with the expected text to the STT-endpoint, the Word Error Rate will be calculated and returned
- When multiple STT- or TTS-engines are configured, select the one to use with the *stt* or *tts* parameter (in combination with the Word Error Rate calculation useful for comparing performance of two engines)

2020-01-31

- Using pre-trained models from Zamia Speech for speech recognition
- Using latest Kaldi build
- Added *file system watcher* to transcribe and synthesize audio files