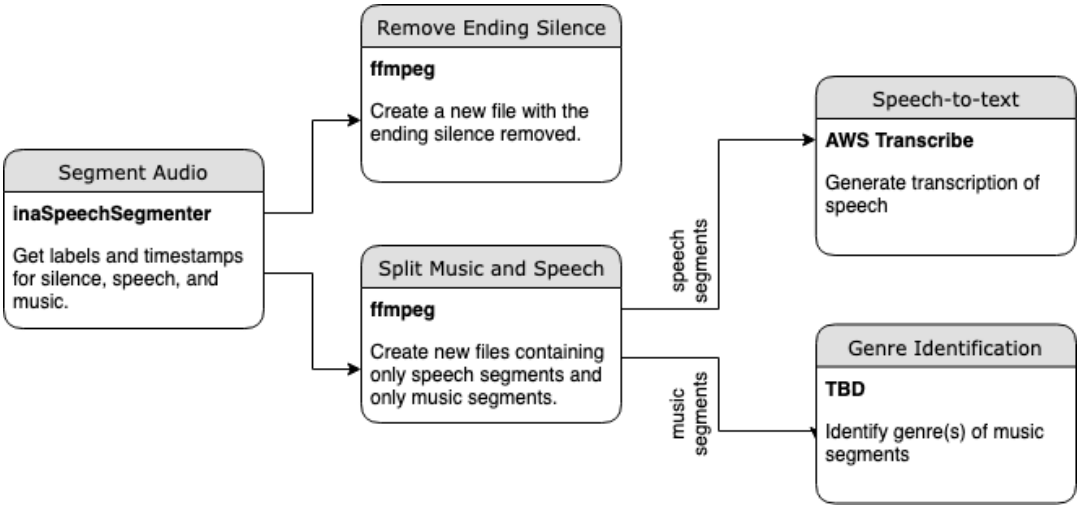
MGM - Segmentation

Category description and use cases

Segmentation MGMs detect when silence, speech, and/or music occur in an audio file. This information may be interesting in its own right for determining how much of an object in an archive has content (e.g. is half the tape silence?). Segment data could also be used to route files (or parts of files) to different MGMs based on the content (for example, sending the speech portions into a workflow that includes STT and the music portions into a music workflow).

Note: these tools do not split the audio files themselves, only output timestamped labels for the contents of a segment. Splitting would need to be handled by another tool, such as ffmpeg.

Workflow example:



Output standard

Summary: An array of segments, each with a label, start, and end. Start and end are timestamps in seconds. The label may be one of: "speech", "music", "silence." If the label is "speech," a gender" may be specified as either "male" or "female."

Element	Datatype	Obligation	Definition
media	object	required	Wrapper for metadata about the source media file.
media.filename	string	required	Filename of the source file.
media.duration	string	required	The duration of the source file audio.
numSpeakers	integer	optional	Number of speakers (if used for diarization).
segments	array	required	Wrapper for segments of silence, speech, or audio.
segments[*]	object	optional	A segment of silence, speech, or audio.
segments[*].label	string	required	The type of segment: silence, speech, or audio.
segments[*].start	string	required	Start time in seconds.
segments[*].end	string	required	End time in seconds.
segments[*].gender	string	optional	The classified gender of the speaker.
segments[*].speakerLabel	string	optional	Speaker label from speaker diarization.

JSON schema


```
{
    "$schema": "http://json-schema.org/schema#",
    "type": "object",
    "title": "Audio Segment Schema",
    "required": [
        "media",
        "segments"
    ],
    "properties": {
        "media": {
            "type": "object",
            "title": "Media",
            "description": "Wrapper for metadata about the source media file.",
            "required": [
                "filename",
                "duration"
            ],
            "properties": {
                "filename": {
                    "type": "string",
                    "title": "Filename",
                    "description": "Filename of the source file.",
                    "default": "",
                    "examples": [
                        "myfile.wav"
                    ]
                },
                "duration": {
                    "type": "string",
                    "title": "Duration",
                    "description": "Duration of the source file audio.",
                    "default": "",
                    "examples": [
                        "25.888"
                    ]
                }
            }
        },
        "numSpeakers": {
            "type": "integer",
            "title": "Number of speakers",
            "description": "Number of speakers (if used for diarization).",
            "default": 0
        },
        "segments": {
            "type": "array",
            "title": "Segments",
            "description": "Segments of silence, speech, or audio.",
            "items": {
                "type": "object",
                "required": [
                    "label",
                    "start",
                    "end"
                ],
                "oneOf": [
                    {
                        "additionalProperties": false,
                        "properties": {
                            "label": {
                                "type": "string",
                                "title": "Label",
                                "description": "The type of segment.",
                                "enum": [
                                    "speech"
                                ]
                            },
                            "start": {
```

```
                            "type": "string",
                            "title": "Start",
                            "description": "Start time in seconds",
                            "default": 0.0,
                            "examples": [
                                "123.45"
                            ]
                        },
                        "end": {
                            "type": "string",
                            "title": "End",
                            "description": "End time in seconds",
                            "default": 0.0,
                            "examples": [
                                "123.45"
                            ]
                        },
                        "gender": {
                            "type": "string",
                            "title": "Gender",
                            "description": "The classified gender of the speaker.",
                            "enum": [
                                "male",
                                "female"
                            ],
                            "default": "unknown"
                        },
                        "speakerLabel": {
                            "type": "string",
                            "title": "Speaker label",
                            "description": "Speaker label from speaker diarization",
                            "default": "unknown"
                        }
                    }
                },
                {
                    "additionalProperties": false,
                    "properties": {
                        "label": {
                            "type": "string",
                            "title": "Label",
                            "description": "The type of segment.",
                            "enum": [
                                "music",
                                "silence"
                            ]
                        },
                        "start": {
                            "type": "string",
                            "title": "Start.",
                            "description": "Start time in seconds",
                            "default": 0.0,
                            "examples": [
                                "123.45"
                            ]
                        },
                        "end": {
                            "type": "string",
                            "title": "End",
                            "description": "End time in seconds",
                            "default": 0.0,
                            "examples": [
                                "123.45"
                            ]
                        }
                    }
                }
            ]
        }
    }
}
```

```
}
```

Sample output

Sample segmentation output

```
{
        "media": {
            "filename": "mysong.wav",
             "duration": "124.3"
          },
        "segments": [
        {
            "label": "speech",
            "start": "0.0",
            "end": "12.35",
            "gender": "male",
                  "speakerLabel": "speaker1"
        },
        {
            "label": "music",
            "start": "10",
            "end": "20"
        }
    ]
}
```

Recommended tool(s)

inaSpeechSegmenter

Official documentation: GitHub

Language: Python

Description: inaSpeechSegmenter detects music, speech, noise, silence ("no energy") and the apparent gender of the speaker. Zones of speech over music are usually tagged as speech but sometimes as music.

Cost: Free (open source)

Social impact: Trained on French-language samples, so its idea of what male and female voices sound like are based on an unknown sample of French speakers. From initial testing, the results have been more or less accurate for our samples in English, but this is an important note.

Notes:

Installation & requirements

Requires ffmpeg and TensorFlow

Install via pip:

```
pip install inaSpeechSegmenter
```

Parameters

None

Because inaSpeechSegmenter does not have any parameters for the minimum length of a segment or maximum length of silence allowed within a speech/music segment, it may be beneficial to add another step in the workflow (or built in to the ina adapter) that allows the output from ina to be filtered/altered based on such parameters.

Input formats

All media formats accepted by ffmpeg (wav, mp3, mp4, etc.)

AMP Implementation notes

In order to create "smoother" segments for our use cases, segments of noise and silence should be a minimum of 10 seconds. Short segments should be folded into the previous segment. This removes very brief segments of silence in between words/sentences/speakers to produce higher-level regions of speech, silence, noise, and music.

For use in sending just regions of speech through speech-to-text MGMs, remove segments of noise and silence that are longer than 1 minute (timestamps will be readjusted later) to cut down on transcription processing time.

Example Usage

inaSpeechSegmenter Example

```
from inaSpeechSegmenter import Segmenter

seg = Segmenter()
segmentation = seg("path/to/file.wav")

for s in segmentation:
        label = s[0]
        start = s[1]
        end = s[2]
        print("Detected {} from {} seconds to {} seconds".format(label, start, end))
```

Example Output

inaSpeechSegmentation Output

```
# Output has been printed in the order start, end, label

0.0        23.76        Music
23.78        28.080000000000002        NOACTIVITY
28.080000000000002        36.6        Music
36.62        37.2        NOACTIVITY
37.2        38.04        Music
38.06        38.9        NOACTIVITY
38.9        44.72        Music
44.74        46.04        NOACTIVITY
46.04        46.58        Music
46.6        47.56        NOACTIVITY
47.56        254.24        Music
254.24        255.26000000000002        Female
255.28        274.82        Music
274.84000000000003        275.32        NOACTIVITY
275.32        277.90000000000003        Music
277.92        278.74        NOACTIVITY
278.74        279.88        Female
279.90000000000003        345.0        Music
345.02        347.5        NOACTIVITY
347.5        355.42        Music
355.44        356.34000000000003        NOACTIVITY
356.34000000000003        372.66        Music
372.68        378.12        NOACTIVITY
378.12        395.2        Music
```

Other evaluated tools

pyannote-audio

Official documentation: GitHub

Language: Python

Description: "Neural building blocks for speaker diarization: speech activity detection, speaker change detection, overlapped speech detection, speaker embedding"

Cost: Free (open source)

Notes: Seems to be somewhat accurate in detecting when speech occurs. However, it only identifies speech segments, not music or silence like we would prefer. Primarily used for detecting different speakers, which is not something we have a particular use case for.

Installation & requirements

Install via pip (pyannote.audio)

Requires Python 3.7 on Linux/MacOS

Example Usage

pyannote-audio Example

```
import sys, os
from datetime import datetime
from pyannote.audio.labeling.extraction import SequenceLabeling
from pyannote.audio.signal import Binarize

def main():
        if len(sys.argv) < 2:
                print("Arguments: input-file [output-file]")

        # Get input/output files
        audio_file = sys.argv[1]
        if len(sys.argv) > 2:
                out = sys.argv[2]
        else:
                out = "pyannote_{}_.txt".format(os.path.basename(audio_file))

        #init model
        media = {'uri': 'filename', 'audio': audio_file}
        SAD_MODEL = ('pyannote-audio/tutorials/models/speech_activity_detection/train/'
             'AMI.SpeakerDiarization.MixHeadset.train/weights/0280.pt')
        sad = SequenceLabeling(model=SAD_MODEL)
        sad_scores = sad(media)

        # Run segmentation
        print("\n\nSegmenting {}".format(media))
        startTime = datetime.now()
        binarize = Binarize(offset = 0.94, onset = 0.70, log_scale = True)
        speech = binarize.apply(sad_scores, dimension = 1)


        # Write output
        print("\n\nWriting to {}".format(out))
        with open(out, 'w') as o:
                for s in speech:
                        result = "{}\t{}\t Speech \n".format(s.start, s.end)  # start  end  label
                        o.write(result)
                        print(result)


        # Print run time
        endTime = datetime.now()
        print("Finished!\n Runtime: {}".format(endTime-startTime))


if __name__ == "__main__":
        main()
```

Example Output

```
4.65        4.66         Speech
4.67        4.68         Speech
4.69        4.7         Speech
4.71        4.72         Speech
4.73        4.74         Speech
4.75        4.76         Speech
4.7700000000000005        4.78         Speech
4.79        4.8         Speech
4.8100000000000005        4.82         Speech
4.83        4.84         Speech
4.8500000000000005        4.86         Speech
4.87        4.88         Speech
4.89        4.9         Speech
4.91        4.92         Speech
4.93        4.94         Speech
4.95        4.96         Speech
4.97        4.98         Speech
4.99        5.0         Speech
6.57        6.58         Speech
7.04        7.05         Speech
7.0600000000000005        7.07         Speech
```

LIUM SpkDiarization

Official documentation: LIUM SpkDiarization

Language: Java, run via command line.

Description: "LIUM_SpkDiarization comprises a full set of tools to create a complete system for speaker diarization, going from the audio signal to speaker clustering based on the CLR/NCLR metrics. These tools include MFCC computation, speech/non-speech detection, and speaker diarization methods."

Cost: Free (open source)

Notes: Like pyannotate-audio, this is primarily for speaker diarization rather than the kind of segmentation we want. We don't like that it assumes a gender of the speaker, though we would just never use that information if we were to use the tool.

We saw so-so results with Astin Patten lecture sample. The tool was not good at detecting a lack of speech in “silent” segments with noise, and (more troublingly) occasionally detected silence in places where there is speech. It detected 6 different speakers, and was more often than not correct in assigning a segment to the correct speaker, though we did not precisely evaluate the tool's accuracy at this task.

For the Women and AIDs sample, the results were nearly useless. Large chunks of audio came out unlabelled despite there being near constant speech in the sample. The tool only identified one speaker, and guessed the speaker was male, when the event has many, mostly female, speakers.

Installation & requirements

Download JAR (see documentation)

Input formats

All media formats accepted by ffmpeg (wav, mp3, mp4, etc.)

Example Usage

inaSpeechSegmenter Example

```
$ /usr/bin/java -Xmx2024m -jar ./LIUM_SpkDiarization.jar --fInputMask=./fileName.wav --sOutputMask=.
/fileName.seg --doCEClustering  fileName
```

Example Output

In addition to start and length of speech segment, this outputs the assumed gender of the speaker (M/F) and a speaker number (S0, S1...)

```
astinPatten 1 0 1004 M S U S0
astinPatten 1 1004 292 M S U S0
astinPatten 1 1296 539 M S U S0
astinPatten 1 1835 1282 M S U S0
astinPatten 1 3117 722 M S U S0
astinPatten 1 3839 953 M S U S0
astinPatten 1 4792 623 M S U S0
astinPatten 1 5415 1324 M S U S0
astinPatten 1 6739 840 M S U S0
astinPatten 1 7579 1403 M S U S0
astinPatten 1 8982 421 M S U S1
astinPatten 1 9403 1504 M S U S0
astinPatten 1 10907 1099 M S U S0
astinPatten 1 12006 372 M S U S0
```

Evaluation summary

See Segmentation Analysis Google Doc