museval Documentation

Release 0.1.0

Fabian-Robert Stöter

Aug 15, 2019

Contents

A python package to evaluate **sigsep musdb18** source separation results as part of the MUS task of the Signal Separation Evaluation Campaign (SISEC)

Contents:

CHAPTER 1

Usage

The purpose of this package is to evaluate source separation results and write out standardized `json` files that can easiliy be parsed by the SiSEC submission system. Furthermore we want to encourage users to use this evaluation output format as the standardized way to share source separation results for processed tracks. We provide two different ways to use `museval` in conjunction with your source separation results.

1.1 Run and Evaluate

- If you want to perform evaluation while processing your source separation results, you can hook `museval` into your `musdb` user_function:

Here is an example for such a function separating the mixture into a **vocals** and **accompaniment** track:

```
import musdb
import museval

output_dir = ...
estimates_dir = ...

def estimate_and_evaluate(track):
    # generate your estimates
    estimates = {
        'vocals': track.audio,
        'accompaniment': track.audio
    }

    # Evaluate using museval
    scores = museval.eval_mus_track(
        track, estimates, output_dir=output_dir
    )

    # print nicely formatted mean scores
    print(scores)
```

(continues on next page)

(continued from previous page)

```
    # return estimates as usual
    return estimates

# your usual way to run musdb
musdb.DB().run()
```

- Make sure `output_dir` is set. `museval` will recreate the `musdb` file structure in that folder and write the evaluation results to this folder. **This whole folder should be submitted for your SiSEC contribution**.

1.2 Evaluate later

If you have already computed your estimates (maybe through the use of MATLAB), we provide you with an easy-to-use function to process evaluation results afterwards.

Simply use the `museval.eval_mus_dir` to evaluate your `estimates_dir` and write the results into the `output_dir`. For convenience, the `eval_mus_dir` function accepts all parameters of the `musdb.run()`. That way e.g. multiprocessing can easily be enabled by setting `parallel=True`:

```
import musdb
import museval

# initiate musdb
mus = musdb.DB()

# evaluate an existing estimate folder with wav files
museval.eval_mus_dir(
    dataset=mus,  # instance of musdb
    estimates_dir=...,  # path to estimate folder
    output_dir=...,  # set a folder to write eval json files
    subsets="Test",
    parallel=True
)
```

1.2.1 Commandline tool

We provide a commandline wrapper of `eval_mus_dir` by calling the `museval` commandline tool. The following example is equivalent to the code example above:

```
museval -p --mus path/to/musdb -o path/to/output_dir path/to/estimate_dir
```

1.3 Using Docker for Evaluation

If you don't want to set up a Python environment to run the evaluation, we would recommend to use Docker.

1.3.1 1. Pull Docker Container

Pull our precompiled `sigsep-mus-eval` image from dockerhub:

```
docker pull faroit/sigsep-mus-eval
```

1.3.2 2. Run evaluation

Lets assume you have stored the estimates directory in `path/to/estimate_dir`, the MUSDB18 is stored in `path/to/musdb`, and you want to write the output.json files to `path/to/output_dir`. You then just mount these directories into the docker container using the `-v` flags and run the container:

```
docker run --rm -v path/to/estimate_dir:/est -v path/to/musdb:/mus -v path/to/output_
↪dir:/out faroit/sigsep-mus-eval --mus /mus -o /out /est
```

Please note hat docker requires absolute paths so you have to rely on your command line environment to convert relative paths to absolute paths (e.g. by using `$HOME/` on Unix).

warning `museval` requires a significant amount of memory for the

evaluation. Evaluating all five targets for musdb18 may require more than 4GB of RAM. If you use multiprocessing by using the `-p` switch in `museval`, this results in 16GB of RAM. It is recommended to adjust your Docker preferences, because the docker container might just quit if its out of memory.

CHAPTER 2

museval modules

CHAPTER 3

BSSEval v4

CHAPTER 4

References

If you use this package, please reference the following paper