SpeechPy Documentation

master

Amirsina Torfi

Nov 23, 2017

Preface

1 Introduction **1**
- 1.1 Foreword . . . 1
- 1.2 Motivation . . . 1
- 1.3 How to Install? . . . 2
- 1.4 Citation . . . 2

2 Preprocessing **3**
- 2.1 Stacking . . . 3

3 Features **5**
- 3.1 MFCC . . . 5
- 3.2 Mel Frequency Energy . . . 5
- 3.3 Log Mel Frequency Energy . . . 6
- 3.4 Extract Derivative Features . . . 6

4 postprocessing **7**
- 4.1 Global Cepstral Mean and Variance Normalization . . . 7
- 4.2 Local Cepstral Mean and Variance Normalization over Sliding Window . . . 7

5 Indices and tables **9**

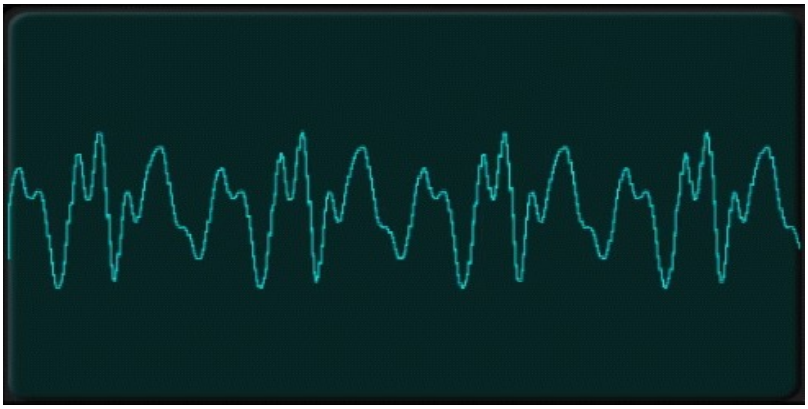
Python Module Index **11**

Chapter 1

Introduction

1.1 Foreword

The purpose of this project is to provide a package for speech processing and feature extraction. This library provides most frequent used speech features including MFCCs and filterbank energies alongside with the log-energy of filterbanks.



1.2 Motivation

There are different motivations for this open source project.

1.2.1 Deep Learning application

One of the main reasons for creating this package was to provide necessary features for deep learning applications such as ASR(Automatic Speech Recognition) or SR(Speaker Recognition). As a results, most of the features that are necessary are provided hear.

1.2.2 Pythonic Packaging

Another reason for creating this package was to have a Pythonic environment for speech recognition and feature extraction due to the fact that the Python language is becoming ubiquotous!

1.3 How to Install?

There are two possible ways for installation of this package: local installation and PyPi.

1.3.1 Local Installation

For local installation at first the repository must be cloned:

```
git clone https://github.com/astorfi/speech_feature_extraction.git
```

After cloning the reposity, root to the repository directory then execute:

```
python setup.py develop
```

1.3.2 Pypi

The package is available on PyPi. For direct installation simply execute the following:

```
pip install speechpy
```

1.4 Citation

If you used this package, please cite it as follows:

```
@misc{amirsina_torfi_2017_840395,
            author = {Amirsina Torfi},
            title = {{SpeechPy: Speech recognition and feature extraction}},
            month = aug,
            year = 2017,
            doi = {10.5281/zenodo.840395},
    url = {https://doi.org/10.5281/zenodo.840395}
}
```

Chapter 2

Preprocessing

2.1 Stacking

speechpy.processing.**stack_frames**(*sig*, *sampling_frequency*, *frame_length=0.02*, *frame_stride=0.02*, *Filter=<function <lambda>>*, *zero_padding=True*)

Frame a signal into overlapping frames.

Parameters

- **sig** (*array*) – The audio signal to frame of size (N,).
- **sampling_frequency** (*int*) – The sampling frequency of the signal.
- **frame_length** (*float*) – The length of the frame in second.
- **frame_stride** (*float*) – The stride between frames.
- **Filter** (*array*) – The time-domain filter for applying to each frame. By default it is one so nothing will be changed.
- **zero_padding** (*boolean*) – If the samples is not a multiple of frame_length(number of frames sample), zero padding will be done for generating last frame.

Returns Array of frames of size (number_of_frames x frame_len).

Return type array

Chapter 3

Features

3.1 MFCC

speechpy.main.**mfcc**(*signal*, *sampling_frequency*, *frame_length=0.02*, *frame_stride=0.01*, *num_cepstral=13*, *num_filters=40*, *fft_length=512*, *low_frequency=0*, *high_frequency=None*, *dc_elimination=True*)

Compute MFCC features from an audio signal.

Parameters

- **signal** – the audio signal from which to compute features. Should be an N x 1 array
- **sampling_frequency** – the sampling frequency of the signal we are working with.
- **frame_length** – the length of each frame in seconds. Default is 0.020s
- **frame_stride** – the step between successive frames in seconds. Default is 0.02s (means no overlap)
- **num_filters** – the number of filters in the filterbank, default 40.
- **fft_length** – number of FFT points. Default is 512.
- **low_frequency** – lowest band edge of mel filters. In Hz, default is 0.
- **high_frequency** – highest band edge of mel filters. In Hz, default is samplerate/2
- **num_cepstral** – Number of cepstral coefficients.
- **dc_elimination** – hIf the first dc component should be eliminated or not.

Returns A numpy array of size (num_frames x num_cepstral) containing mfcc features.

3.2 Mel Frequency Energy

speechpy.main.**mfe**(*signal*, *sampling_frequency*, *frame_length=0.02*, *frame_stride=0.01*, *num_filters=40*, *fft_length=512*, *low_frequency=0*, *high_frequency=None*)

Compute Mel-filterbank energy features from an audio signal.

Parameters

- **signal** – the audio signal from which to compute features. Should be an N x 1 array
- **sampling_frequency** – the sampling frequency of the signal we are working with.
- **frame_length** – the length of each frame in seconds. Default is 0.020s
- **frame_stride** – the step between successive frames in seconds. Default is 0.02s (means no overlap)

- **num_filters** – the number of filters in the filterbank, default 40.
- **fft_length** – number of FFT points. Default is 512.
- **low_frequency** – lowest band edge of mel filters. In Hz, default is 0.
- **high_frequency** – highest band edge of mel filters. In Hz, default is samplerate/2

Returns features: the energy of fiterbank: num_frames x num_filters frame_energies: the energy of each frame: num_frames x 1

3.3 Log Mel Frequency Energy

speechpy.main.**lmfe**(*signal*, *sampling_frequency*, *frame_length=0.02*, *frame_stride=0.01*, *num_filters=40*, *fft_length=512*, *low_frequency=0*, *high_frequency=None*)

Compute log Mel-filterbank energy features from an audio signal.

Parameters

- **signal** – the audio signal from which to compute features. Should be an N x 1 array
- **sampling_frequency** – the sampling frequency of the signal we are working with.
- **frame_length** – the length of each frame in seconds. Default is 0.020s
- **frame_stride** – the step between successive frames in seconds. Default is 0.02s (means no overlap)
- **num_filters** – the number of filters in the filterbank, default 40.
- **fft_length** – number of FFT points. Default is 512.
- **low_frequency** – lowest band edge of mel filters. In Hz, default is 0.
- **high_frequency** – highest band edge of mel filters. In Hz, default is samplerate/2

Returns features: the energy of fiterbank: num_frames x num_filters frame_log_energies: the log energy of each frame: num_frames x 1

3.4 Extract Derivative Features

speechpy.main.**extract_derivative_feature**(*feature*)

This function extracts temporal derivative features which are first and second derivatives. :param feature: The feature vector which its size is: N x M

Returns The feature cube vector which contains the static, first and second derivative features size: N x M x 3

Chapter 4

postprocessing

4.1 Global Cepstral Mean and Variance Normalization

speechpy.processing.**cmvn**(*vec*, *variance_normalization=False*)

This function is aimed to perform global cepstral mean and variance normalization (CMVN) on input feature vector "vec". The code assumes that there is one observation per row.

Parameters

- **vec** – input feature matrix (size:(num_observation,num_features))
- **variance_normalization** – If the variance normilization should be performed or not.

Returns The mean(or mean+variance) normalized feature vector.

4.2 Local Cepstral Mean and Variance Normalization over Sliding Window

speechpy.processing.**cmvnw**(*vec*, *win_size=301*, *variance_normalization=False*)

This function is aimed to perform local cepstral mean and variance normalization on a sliding window. (CMVN) on input feature vector "vec". The code assumes that there is one observation per row.

Parameters

- **vec** – input feature matrix (size:(num_observation,num_features))
- **win_size** – The size of sliding window for local normalization. Default=301 which is around 3s if 100 Hz rate is considered(== 10ms frame stide)
- **variance_normalization** – If the variance normilization should be performed or not.

Returns The mean(or mean+variance) normalized feature vector.

Chapter 5

Indices and tables

- genindex
- modindex
- search

Python Module Index

S

speechpy.main, 5
speechpy.processing, 3

Index

C

cmvn() (in module speechpy.processing), 7
cmvnw() (in module speechpy.processing), 7

E

extract_derivative_feature() (in module speechpy.main), 6

L

lmfe() (in module speechpy.main), 6

M

mfcc() (in module speechpy.main), 5
mfe() (in module speechpy.main), 5

S

speechpy.main (module), 5
speechpy.processing (module), 3, 7
stack_frames() (in module speechpy.processing), 3