**WALnet**

**Implementing MFCC**

Python packages required :

1. librosa
2. pandas
3. numpy
4. os
5. Csv

The aim to convert the raw audio snippets (extracted from the DCASE 2019 in .wav format) to Mel Coefficients. See *mel\_features.py.* Header String is created which puts a label for every coefficient to be used. For this task, 20 mfcc features, 18 delta-mfcc features and 18 double delta features have been extracted. Make sure the path defined contains no white spaces as well doesn’t start with a numeric or lowercase alphabet like n, t ,r etc.he frequency range was confined to range of human hearing i.e. 20Hz (f\_min or minimum frequency) to 20kHz (f\_max or maximum frequency). The dataset created is stored in a csv file, named mfcc\_dataset.

Sampling Rate = sr = 44100;

Number of mel-bands =  n\_mel = 128;

**Using VGGish\_audioset\_extraction pipeline**

Python packages used :

1. Numpy
2. Scipy
3. Resampy
4. Tensorflow
5. Six

Additional downloads required :

1. VGGish model checkpoint (in Tensorflow checkpoint format)

<https://storage.googleapis.com/audioset/vggish_model.ckpt>

1. Embedding PCA parameters (in numpy compressed archive format)

Files downloaded must be placed in the same directory where ‘this’ instruction file is placed.

Here we use VGGish as a feature extractor which converts audioinput features into semantically meaningful, high level 128-embedding which can be fed as a downstream for any model. VGGish is a variant of the [VGG] (*https://arxiv.org/abs/1409.1556*) model, in particular Configuration A with 11 weight layers. Specifically, here are the changes we made:

\* The input size was changed to 96x64 for log mel spectrogram audio inputs.

\* We drop the last group of convolutional and maxpool layers, so we now have

  only four groups of convolution/maxpool layers instead of five.

\* Instead of a 1000-wide fully connected layer at the end, we use a 128-wide

  fully connected layer. This acts as a compact embedding layer.

The model definition provided here defines layers up to and including the 128-wide embedding layer.

1. *vggish\_params.py*  
   Contains global parameters for the VGGish model, and are implemented in *vggish\_slim.py*

**Classifier**

Python packages required :

1. sys
2. os
3. time
4. TORCH
5. random
6. sklearn