NUS-ISS
Real Time Audio-Visual Sensing and Sense Making

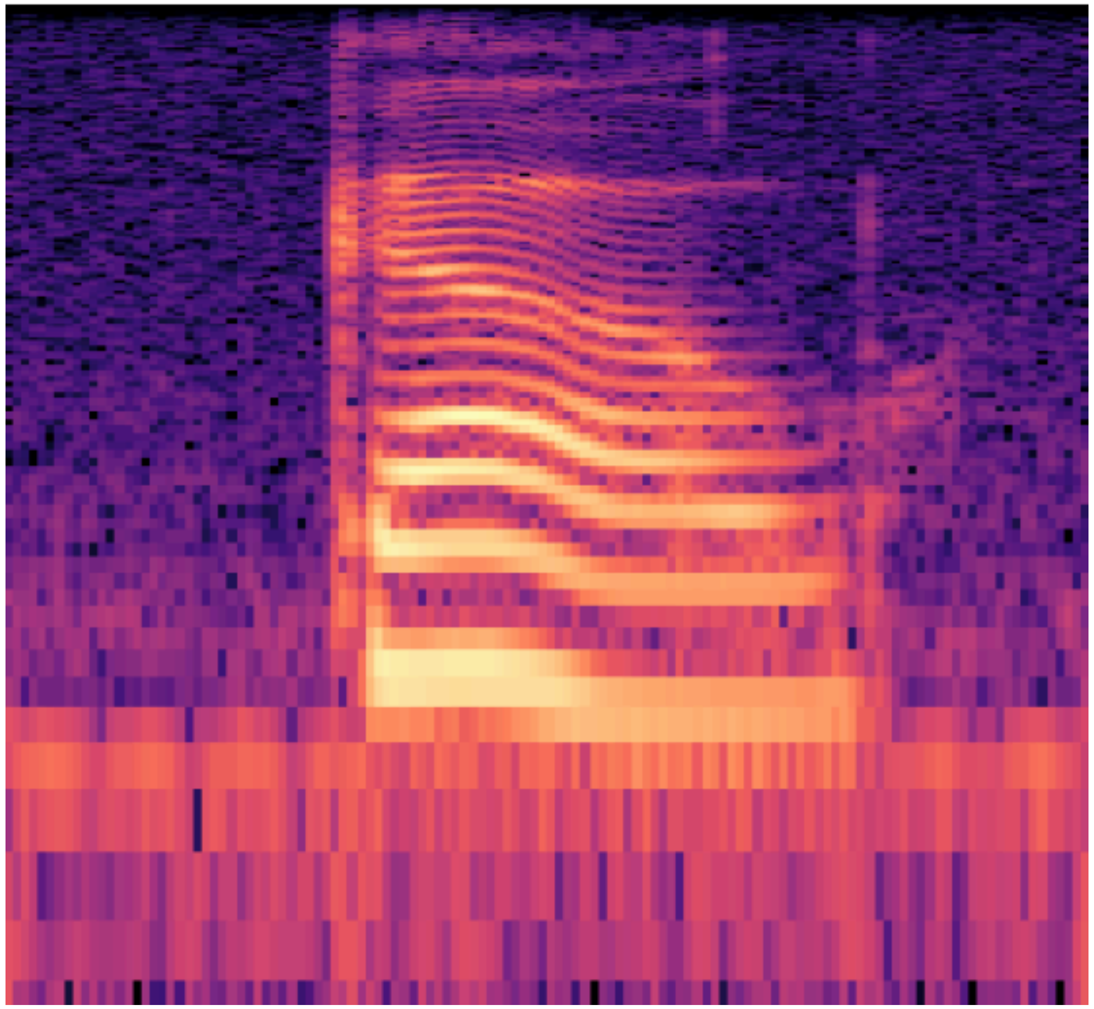
Module 7 - Workshop on real time audio recognition, part 2

Dr. Tan Jen Hong
Lecturer & Consultant
Institute of System Science
National University of Singapore
issjht@nus.edu.sg

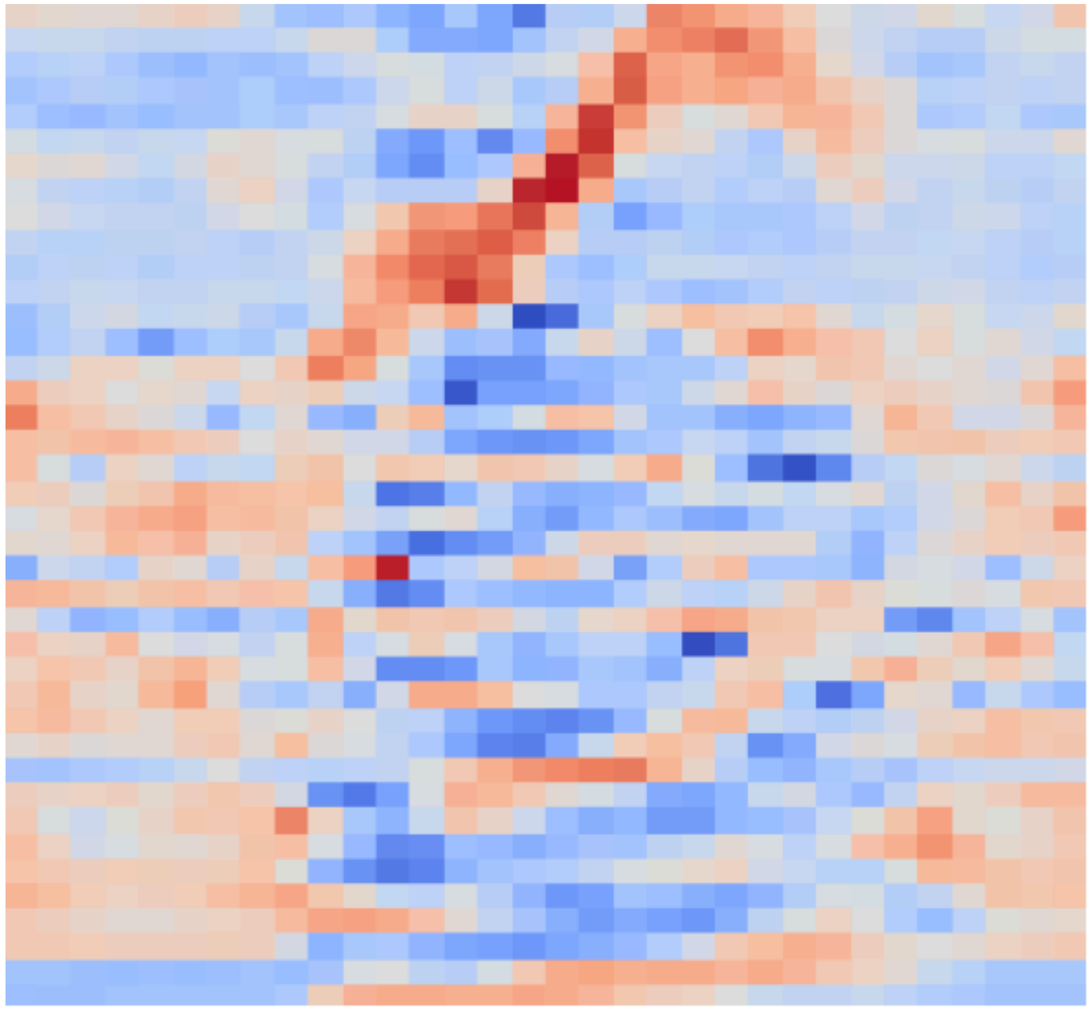
How to get them

in python and use them for machine learning

Spectogram



Mel-frequency cepstral coefficients



Before we start

import the necessary

- We use Librosa to create mel-scaled spectrogram, mel-frequency cepstral coeeficients

```
> import os
> import librosa
> import sklearn
```

for audio processing

```
> import matplotlib.pyplot as plt
> import numpy as np
> import librosa.display as libd
```

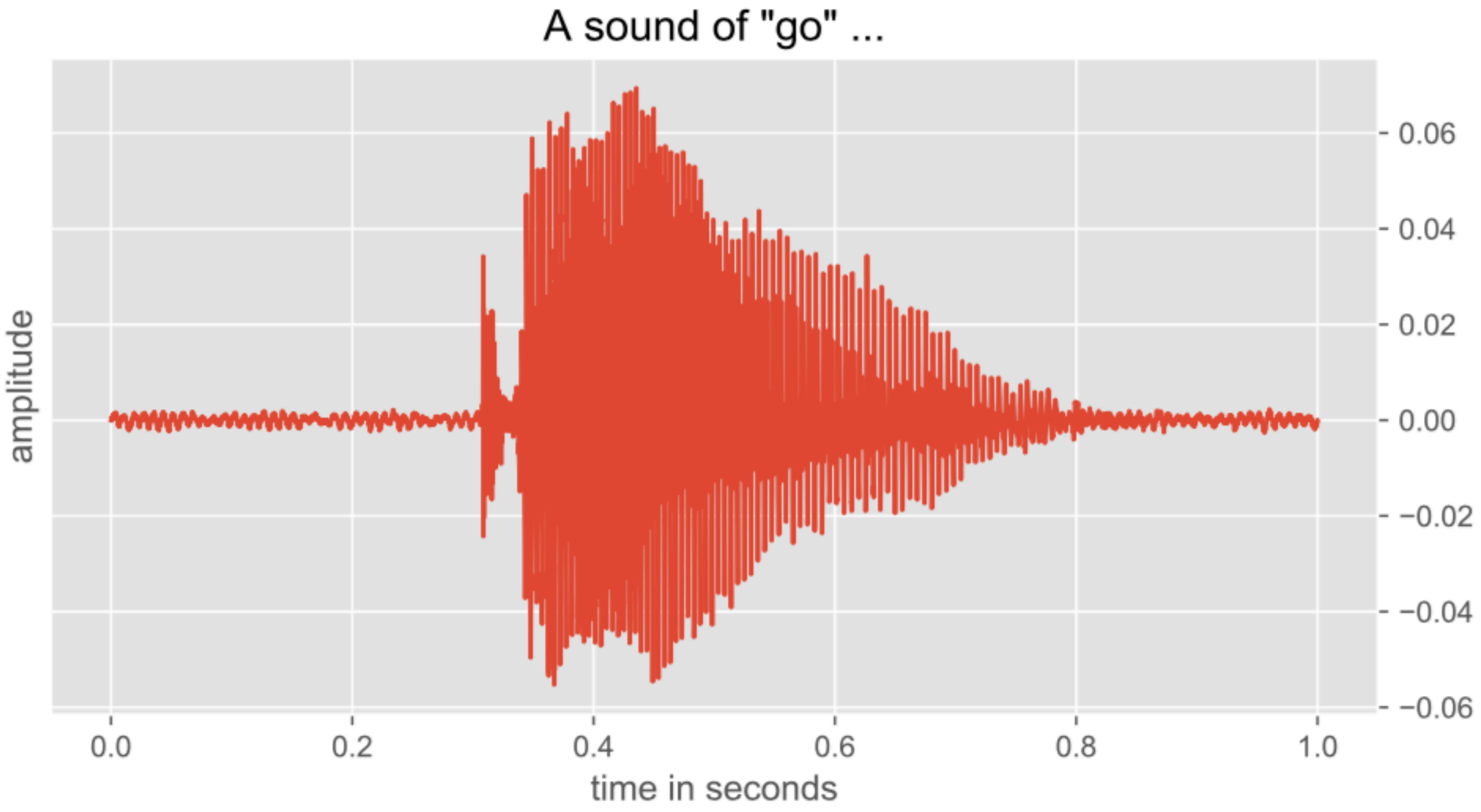
for the display of spectogram and mel-frequency cepstral coeeficients

```
> plt.style.use('ggplot')
```

The audio of interest

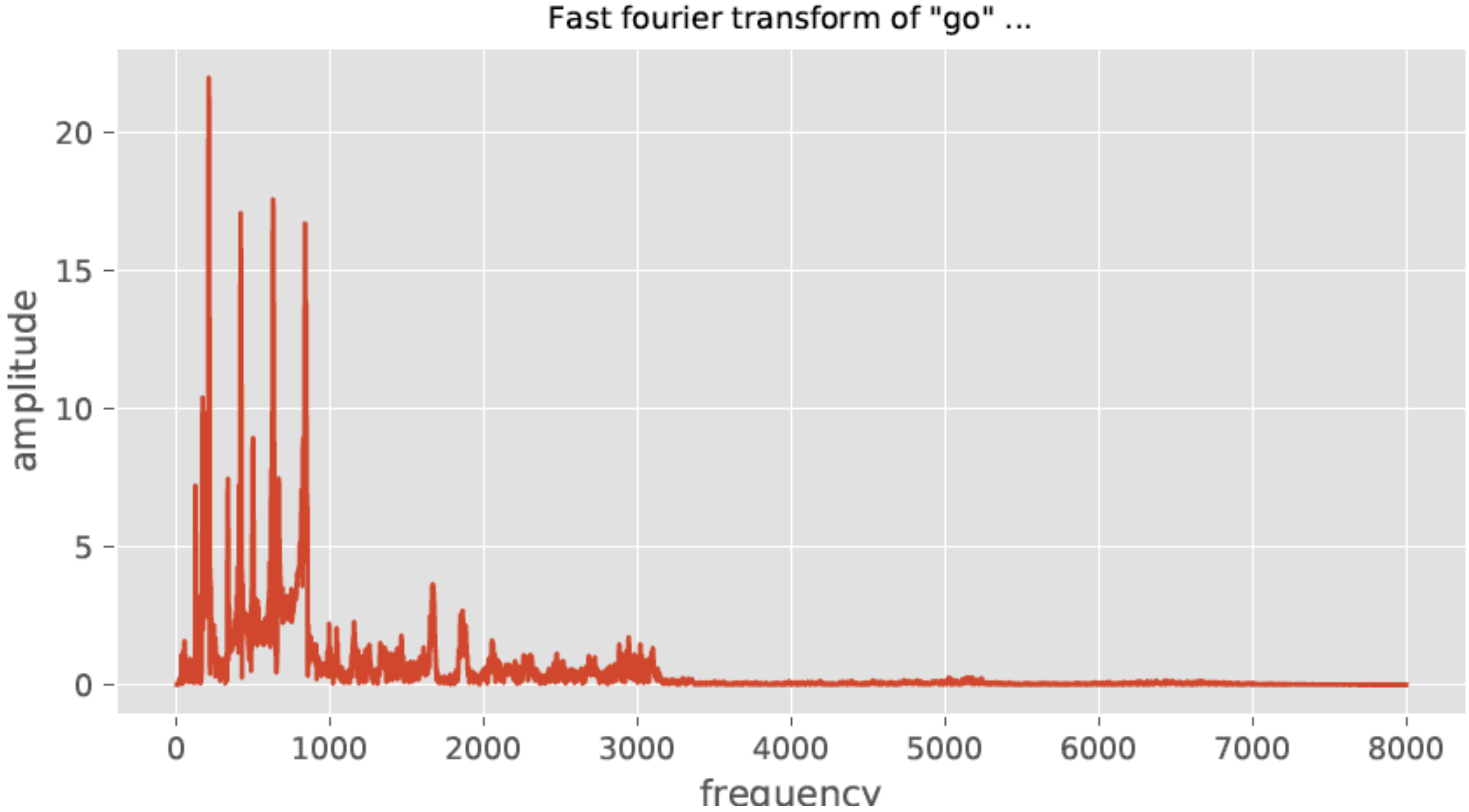
"go"

- We are looking at the same "go" recording



Let's look at the FFT

"go"



```
> smpFft      = np.abs(np.fft.fft(smp))
> smpFft      = smpFft[:8000]
```

We use the numpy in-built fft function

Only the first half is taken, the second half is a mirror of the first half

```
> plt.figure(figsize=(8,4))
> plt.plot(np.linspace(0,
                       smpR/2,
                       len(smp)/2),
          smpFft)
> plt.title('Fast fourier transform of "go" ...',fontsize=10)
> plt.xlabel('frequency')
> plt.ylabel('amplitude')
```

The maximum frequency, half of the sampling rate due to Nyquist-Shannon sampling theorem

Create the spectogram

"go"

- Using short-time Fourier transform

```
> fftSize     = 512
> smpStft     = np.abs(librosa.stft(y=smp,
                                    n_fft=fftSize))
```

The window size for FFT, required for librosa.stft

Note: We use the default value for hop_length in this function (not shown as argument). By default, the hop_length is 1/4 of FFT window size.

You can imagine hop length as the 'stride' for fft moving window

The size of smpStft is (257, 126). float32. The value 257 is dependent on the maximum frequency in the signal, the value 126 is dependent on the length of the signal, the window size and hop length

```
> spectogram  = librosa.amplitude_to_db(smpStft,
                                        ref=np.max)
```

convert spectogram from amplitude to decibel-scaled

This is required to ensure the values are scaled based on the maximum value in the input

```
> plt.figure()
> libd.specshow(spectogram,
                sr=16000,
                hop_length=fftSize/4,
                y_axis='log',
                x_axis='time')
> plt.title('Spectogram of the "go" ...',fontsize=10)
> plt.xlabel('time (s)')
> plt.colorbar(format='%+2.0f dB')
```

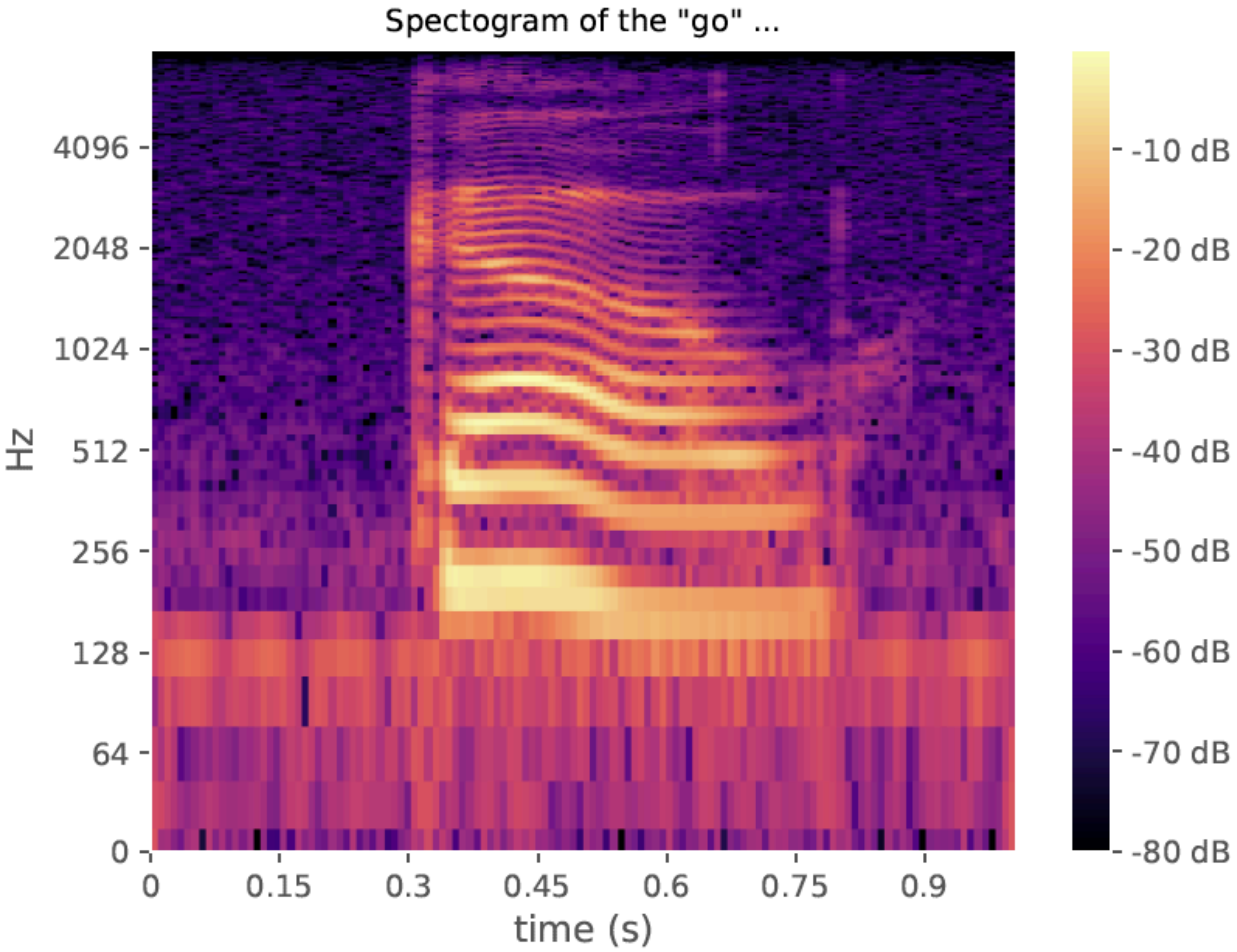
This must be put correctly to ensure the y axis value is correct

This must be put correctly to ensure the x axis value is correct

Create the spectogram

"go"

- The Y-axis is in log-scale (2^n)
- By Nyquist-Shannon theorem, the maximum frequency should be 8000Hz (the sampling rate is 16,000Hz)



Create the MFCC

"go"

```
> smpMfcc     = librosa.feature.mfcc(y=smp,
                                     sr=16000,
                                     n_mfcc=40)
```

Number of filter banks to be used. In the output, each row is the output of a filter bank. The size of smpMfcc is (40,32), float64.

The default hop length in this function is 512, and the fft window size is 2048, as specified in librosa.feature.melspectogram

```
> smpMfcc     = sklearn.preprocessing.scale(smpMfcc,
                                            axis=1)
```

Rescale each coefficient dimension (along the row), so that it has a mean of 0 and a variance of 1

```
> plt.figure()
> libd.specshow(smpMfcc,
                sr=16000,
                hop_length=512,
                x_axis='time')
> plt.title('Spectogram of the "go" ...',fontsize=10)
> plt.colorbar()
```

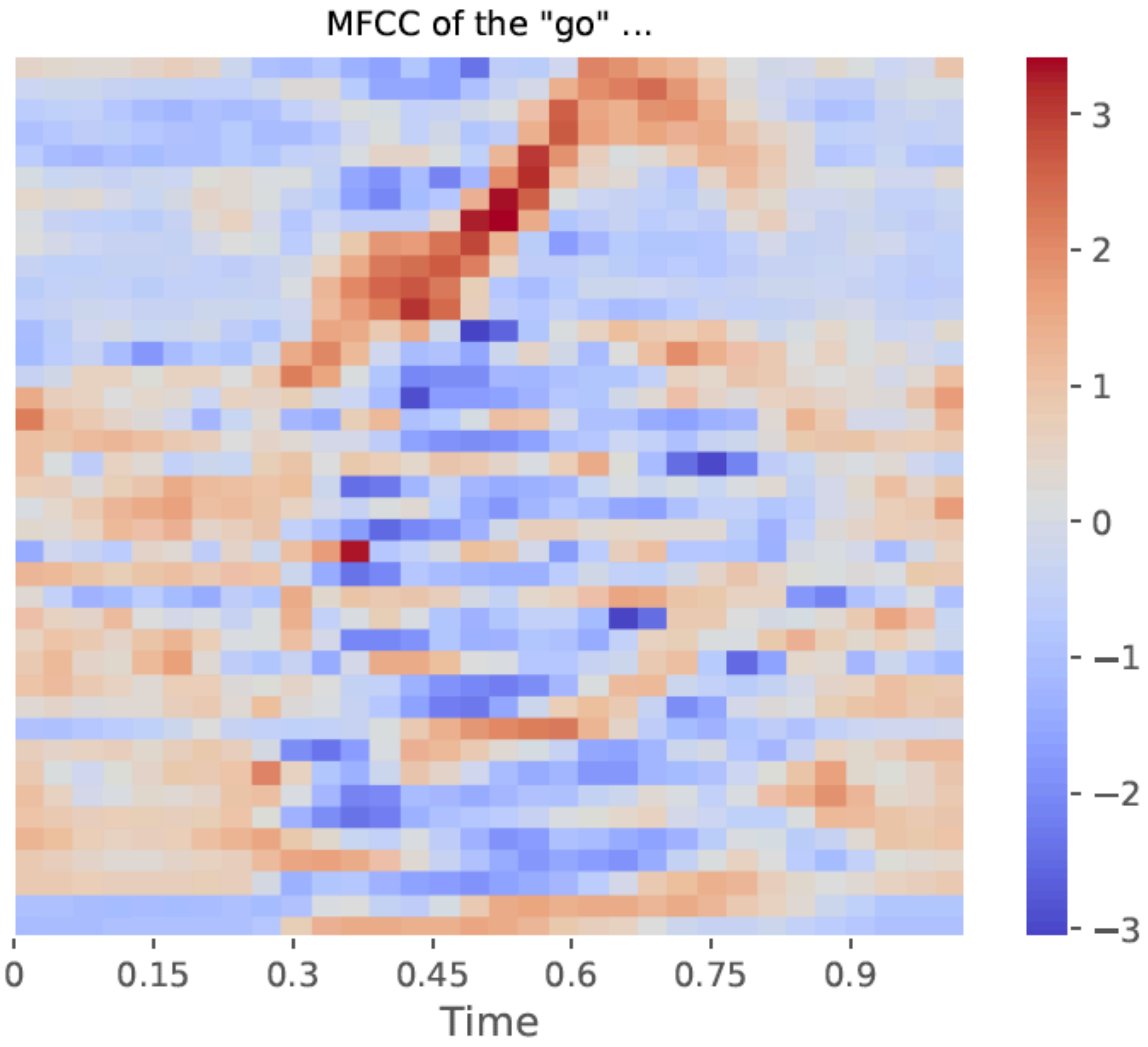
This must be put correctly to ensure the y axis value is correct

This must be put correctly to ensure the x axis value is correct

Create the MFCC

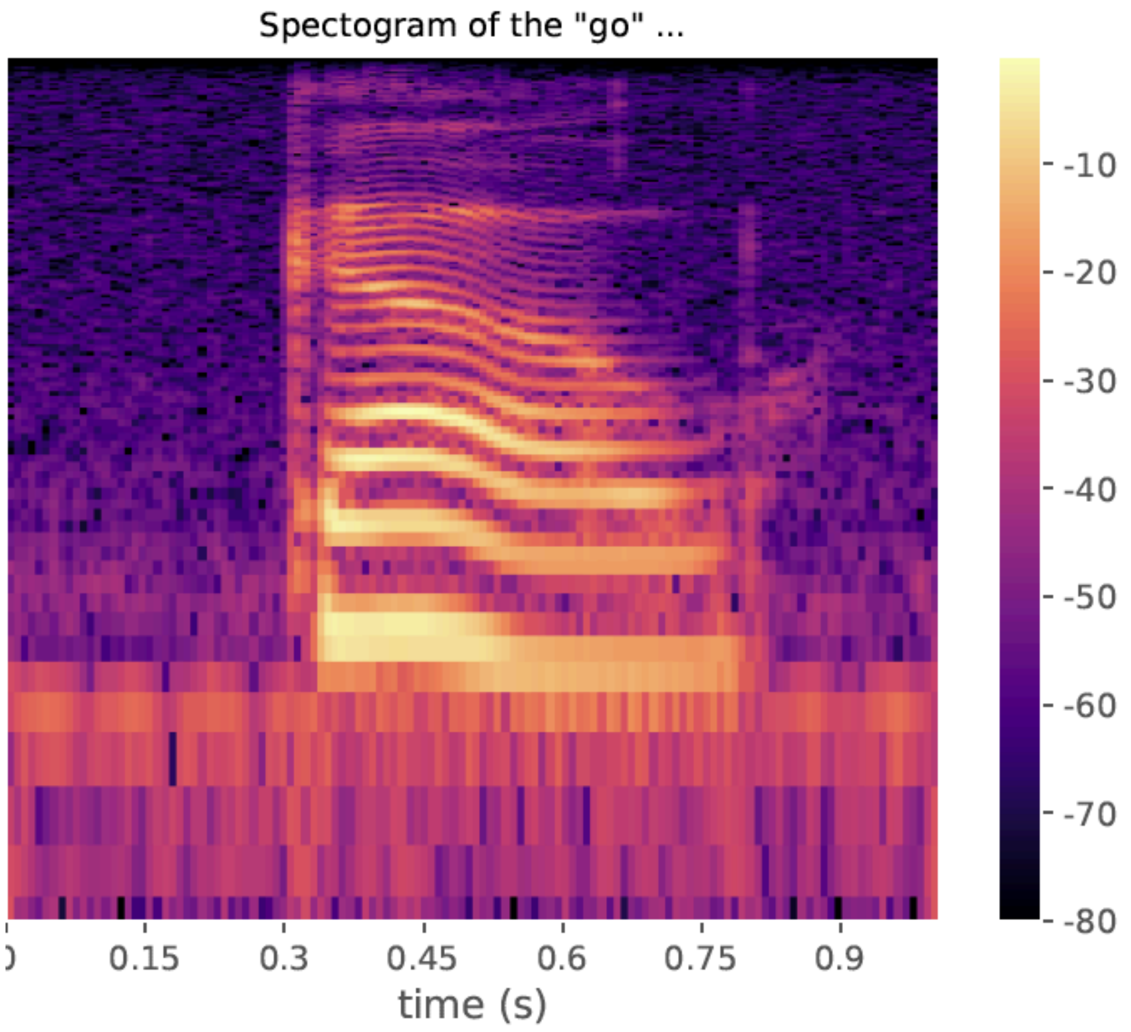
"go"

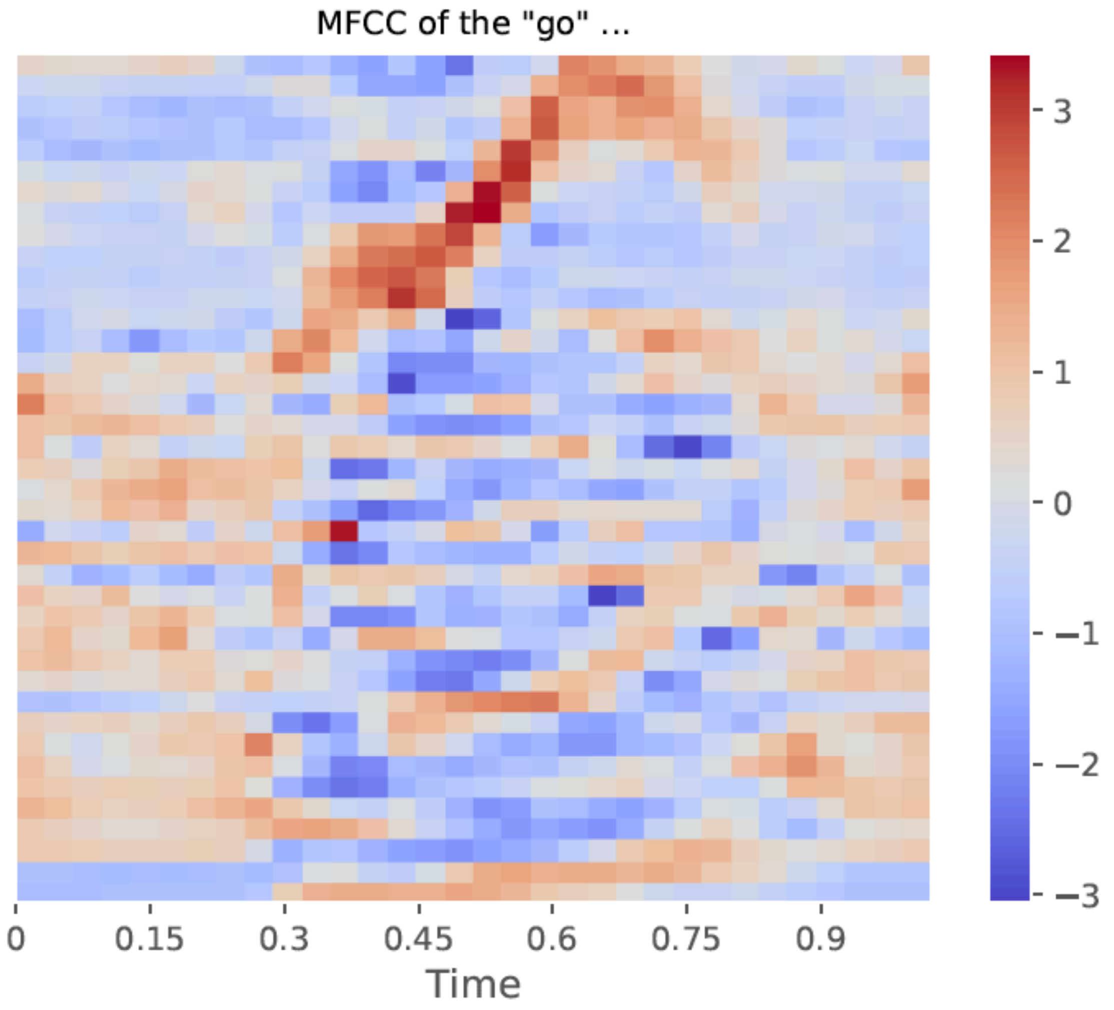
- Each row is the output of a filter bank with a mean of 0 and variance of 1



Input to deep learning model?

- If you were to train a deep learning model on audio, which 2D representation will you choose? Why?


Spectogram of the "go" ...


MFCC of the "go" ...