Audio classification App

Background

In collaboration with the Seal Rehabilitation and Research Centre, we need an **application** where veterinarians can easily upload a recorded audio file and automatically analyze it. This application will help them study the survival prospect of the seal under evaluation.

Goal and functionalities

The goal of this project is to create a **Windows application**. This software will be based on only one user profile who will have access to different functionalities:

- Functionalities:
  - **Upload** an audio file.
  - **Pre-process** the audio.
  - **Apply** our trained **Deep Learning model (CNN)** model to classify the audio.
  - **Display** classification result with the values that characterize the audio.

Besides these simple user functionalities, the application backend should pre-process the uploaded audio signal in order to match the input of the CNN model. The following steps are implemented and need to be embedded in the backend application in order to process the data:

- **Pre-process** signal (usually there are libraries that do this for you)
- Resample the uploaded signal using a sampling rate of 8000Hz
- Get the **Mel Sceptral Coefficients** of the audio file
  - These are just a set of features (mfcc = 20 for our project) which concisely describe the overall shape of a spectral envelope. We humans can't hear many high freq, so these coefficients help us to transform a signal into something more related to what humans can perceive as sounds.
- Reformat the **Mel Sceptral Coefficients** array such that it follows the Convolutional Neural Networks input format.
- Input the **Mel Sceptral Coefficients** array into our model's evaluate() function.

Useful tools

Python libraries

- **Librosa:**
  - **librosa.load(audio, sampling_rate)**: pre-processes the audio signal given as the first parameter. It also resamples the audio if you give a value to the second parameter (i.e sampling_rate
  - **librosa.mfcc(y=audio, sr=sampling_rate, n_mfcc=20)**: Returns an array with the first 20 mel spectral coefficients

Frameworks and Languages

Because the weights of the model are decoupled from the python implementation, you can build the application using any framework or language that allows the weights of the model to be imported. However, make sure that data is processed in the same way as explained above.

The number of coefficients should 20 and the audio signal should be resampled at 8000Hz. These numbers should be kept no matter what framework or language you use.

Customer Contact

Estefanía Talavera Martínez (e.talavera.martinez@rug.nl)