Sound Communication: Lab I

RealTime Machine Learning Modeling (I)

Sergio Giraldo
Universitat Pompeu Fabra

2018-2019

Introduction

The objective of this lab session is to introduce real time systems for Machine Learning Modeling and to illustrate some possible applications. Throughout the course we will use Wekinator, (an open-source, real-time machine learning library), as the main tool for the labs. In this session we will cover some fundamentals concepts of the Wekinator interface. We well briefly explain some basic audio signal processing concepts on how to extract features from audio to be later used to train models, aiming at different purposes. Finally we will implement a simple real-time machine learning system that takes as input some audio features, and outputs the predictions of 3 different models.

1 Background Theory Concepts

1.1 Audio signal processing

Audio signals can be represented in either digital or analog format, where signal processing may occur in either domain. Analog processors operate directly on the electrical signal, while digital processors operate mathematically on the binary representation of that signal. Digital audio signals are discrete signals that represents the pressure wave-form as a sequence of discrete values, as represented in Figure 1

Each value of amplitude from the digital audio representation is computationally stored computationally as an array of float values, whose amplitudes vary form 1 to -1 (equivalent to 0dBFs).

1.2 Frequency Domain Analysis

In nature we can often found pure and complex tones. A pure tone (less frequent) is a tone with a sinusoidal waveform; this is, a sine wave of any frequency, phase, and amplitude. In contrast, complex tones are the ones resulting from the addition of several pure tones at several frequencies. Thus, frequency analysis of audio signals aims at obtaining the frequency components (i.e. the pure tones) from which a particular or section of a sound is composed.

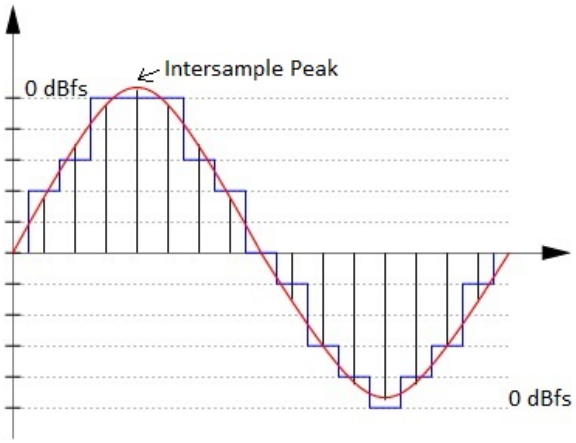

Figure 1: Discrete representation of an audio wave form

Frequency analysis is perform on a frame by frame basis. Because a sound might vary along time, only a small portion of an audio signal is analyzed at time. This small portion is called a frame (a.k.a. analysis window). The resulting frequency values discretized along frequency domain are called the spectrum. The most common visual representation of the frequencies composing an audio signal, as they vary with time, is called the spectrogram (see Figure 2).

1.3 Audio and Music Analysis

Audio and music analysis is a collection of methodologies and techniques for the automatic characterization of musical audio content. This content refers to the implicit information that is related to a piece of music/audio and that is represented in the piece itself.

1.3.1 Audio Feature Descriptors

Many different type of audio descriptors have been developed within the audio and speech recognition community. Audio descriptors help to characterize an excerpt of an audio signal (or an audio frame) in terms of its spectral, temporal, and spectro-temporal properties. Audio descriptors are divided into Global and Instantaneous descriptors. The foremost refer to those in which the whole signal is used for its computation (e.g. Attack duration), the later are computed for each frame at a time.

Because in this lab we will be dealing with real-time prediction, we will make use of some of the instantaneous spectral descriptors:

1. FFT bin magnitudes: Used for pitch, timbre and vocal recognition.
2. MFCCs: Used for timbre and vocal recognition
3. Constant-Q: This consists of a 12 band octave filter.
4. Peak Frequency: Outputs the predominant pitch of the sound.

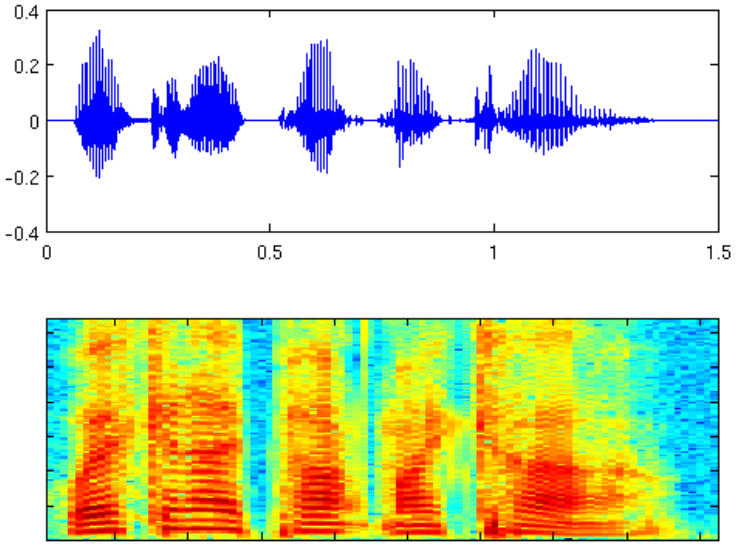

Figure 2: Spectrogram of an audio wave form

5. Spectral Centroid: Used for timbre recognition, outputs the center of gravity of the spectrum frequency distribution.

6. RMS: Is a measure of the overall loudness of a sound.

   For an overview of audio descriptors see: Peeters, G. (2004). A large set of audio features for sound description (similarity and classification) in the CUIDADO project.

2 Practical Work

In this section you will develop a system for automatic classification of audio signals in real time.

2.1 materials

- Computer with sound card and built-in microphone.
- Real-time audio feature extractor: executable file you can download from the Wekinator examples page: `http://www.wekinator.org/examples/#Audio`
- Wekinator. You can download wekinator from: `www.wekinator.org`
- Pure Data: A software for graphical computation. You can download from `https://puredata.info/`. You are encouraged to use *Processing or other programing language that facilitates the visualization of data.*

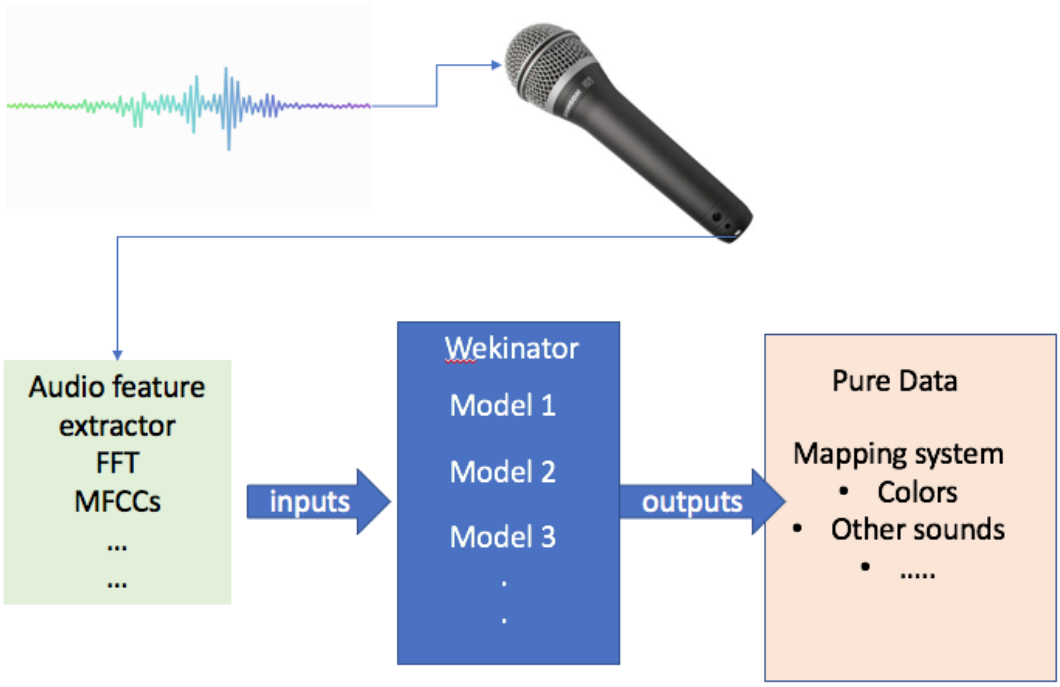

Figure 3: Real-time Machine Learning classification form audio

2.2 Framework

The framework of the expected system is depicted in Figure 3.

2.2.1 Input

The input of the system will be sounds, you can choose any sounds you like. Choose at least three different types of sounds. The sounds will be captured by the computer microphone, and processed by the provided audio feature extractor.

The feature extractor sends the feature data trough osc port 6448. Choose at the feature extractor as many features as you consider. Be aware of how many total inputs is the feature extractor sending.

2.3 Wekinator

Open Wekinator and start a new project. Choose the port number for the input as 6448.

Put the number of inputs wekinator is receiving from the audio feature extractor.

Choose the number of outputs (i.e. models) you want wekinator to train and output. Also set the output port of wekinator at 12000.

Train the models according to the expected output of your system until Wekinator is able to classify different sounds.

2.4 output

Wekinator sends the output of each model trough OSC. At pure data you will be provided with a patch which is able to listen to this port and output the received values to boxes. After, you can map these received classification probabilities values (i.e. the value of the slider) to an interesting feedback. For instance you can map it to another sound, or you can create a visualization based on colors. Choose something which is easy to understand, in order to be clearly aware of the class that has been predicted.

Submission

You have to deliver all the code and/or patches created for the practical work (i.e. wekinator files, and puredata files). You also have to submit as well a written report explaining the framework of your work, inputs, outputs models used. All the files should be submitted in a zip file though the Aula Global.

You will have to present and demonstrate your system at the next class.