Instrument Classification using Spark MLlib and Elephas

Felipe Schreiber Fernandes

Motivation

- You are creating your own music app
- How would you organise your playlist?
  - By genre, maybe.
- What if you don't know the genre?
  - Then train a model to classify
- Which features can be extracted from audio?
  - MIR (Music Information Retrieval)
    - Instruments, per example.

Dataset Description

- IRMAS (Instrument Recognition in Musical Audio Signals) [2];
- WAV files with most prominent instrument being played annotated
- Instruments available in the dataset:
    - Violoncelo;
    - Clarinet;
    - Flute;
    - Guitar;
    - Among others.

Feature Extraction

- Spectral Centroid
- Spectral Bandwidth
- Spectral Rolloff
- Zero-Crossing Rate
- RMS Energy (Root Mean Square Energy)
- MFCC (Mel-Frequency Cepstral Coefficients)
- Delta Features for MFCC

Spectral Centroid

- Intuition: Obtain the most representative frequency in the window

Spectral Centroid: The spectral centroid represents the "center of mass" of a spectral power distribution. It is calculated as the weighted mean of the frequencies present in the signal, determined using a fourier transform, with their magnitudes as the weights:

$$Centroid, \mu = \frac{\sum_{i=1}^{N} f_i \cdot m_i}{\sum_{i=1}^{N} m_i} \quad (9)$$

where m_i represents the magnitude of bin number i, and f_i represents the center frequency of that bin.

Spectral Bandwidth

- Intuition: Get the frequency range of the signal

$$\left(\sum_k S(k)(f(k) - f_c)^p\right)^{\frac{1}{p}}$$

where S(k) is the spectral magnitude of frequency at “bin” k,
f(k) is the frequency at “bin” k and
f_c is the spectral centroid.

- The default value of “p” is 2

Spectral Kurtosis

- Intuition: Measures how flatten the energy distribution is around the centroid

$$\text{kurtosis} = \frac{\sum_{k=b_1}^{b_2} (f_k - \mu_1)^4 s_k}{(\mu_2)^4 \sum_{k=b_1}^{b_2} s_k}$$

where

- f_k is the frequency in Hz corresponding to bin k.
- s_k is the spectral value at bin k.
- b_1 and b_2 are the band edges, in bins, over which to calculate the spectral skewness.
- μ_1 is the spectral centroid, calculated as described by the `spectralCentroid` function.
- μ_2 is the spectral spread, calculated as described by the `spectralSpread` function.

Taken from
https://www.mathworks.com/help/audio/ref/spectralkurtosis.html#mw_1172e6ff-e502-4fc2-b763-2ca0629a4f7c

Spectral Skewness

- Intuition: Measures the asymmetry of the energy distribution around the centroid

$$\text{skewness} = \frac{\sum_{k=b_1}^{b_2} (f_k - \mu_1)^3 s_k}{(\mu_2)^3 \sum_{k=b_1}^{b_2} s_k}$$

where

- f_k is the frequency in Hz corresponding to bin k.
- s_k is the spectral value at bin k.
- b_1 and b_2 are the band edges, in bins, over which to calculate the spectral skewness.
- μ_1 is the spectral centroid, calculated as described by the `spectralCentroid` function.
- μ_2 is the spectral spread, calculated as described by the `spectralSpread` function.

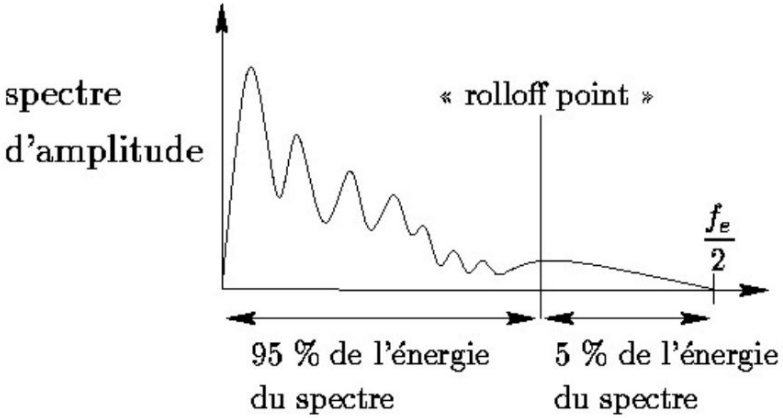
Taken from
https://www.mathworks.com/help/audio/ref/spectralskewness.html#mw_0866778f-4e7f-451d-b26f-fa0517b1deaf

Spectral Rolloff

- Frequency below which a specified percentage of spectrum energy is contained
- Frequencies above rolloff frequency decay quickly.

$$\underset{f_c \in \{1,\ldots,N\}}{argmin} \sum_{i=1}^{f_c} m_i \geq c \sum_{i=1}^{N} m_i$$

where m_i is the magnitude of i-th spectrum frequency, f_c is the "rolloff" frequency and c is a constant at [0,1] interval (indicates how much of energy of spectre will be considered)



Zero Crossing Rate

- Indicates how many times the signal crosses the X axis

$$zcr = \frac{1}{T-1}\sum_{t=1}^{T-1}\mathbb{I}\{s_t s_{t-1} < 0\}$$

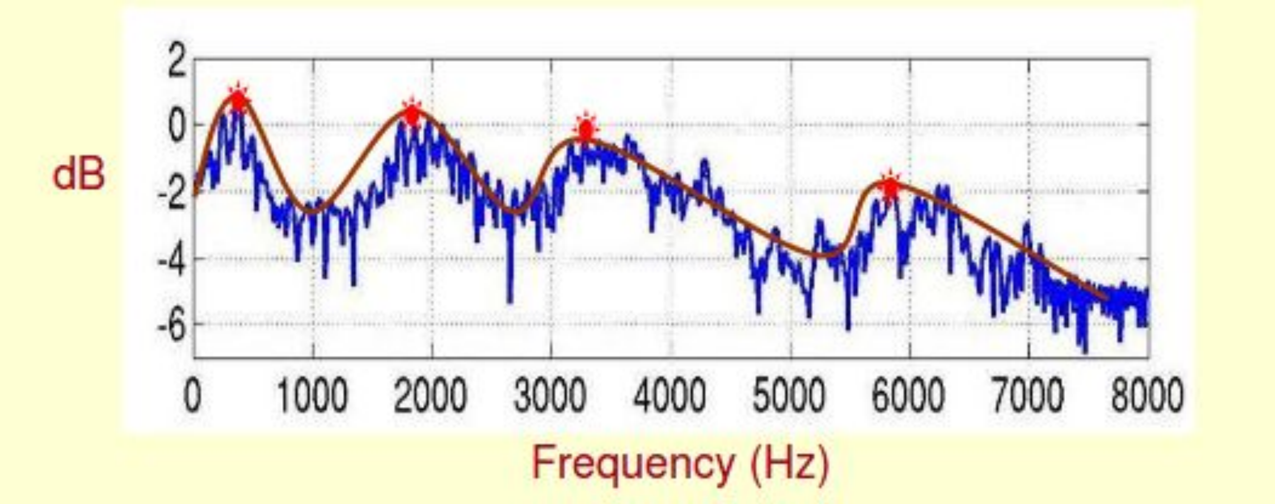
II is the indicative function (1 if X = True, 0 otherwise)

RMS Energy (Root Mean Square Energy)

- At each frame is calculated the root of the quadratic mean of the signal amplitude over time. This represents the average signal strength in that frame.

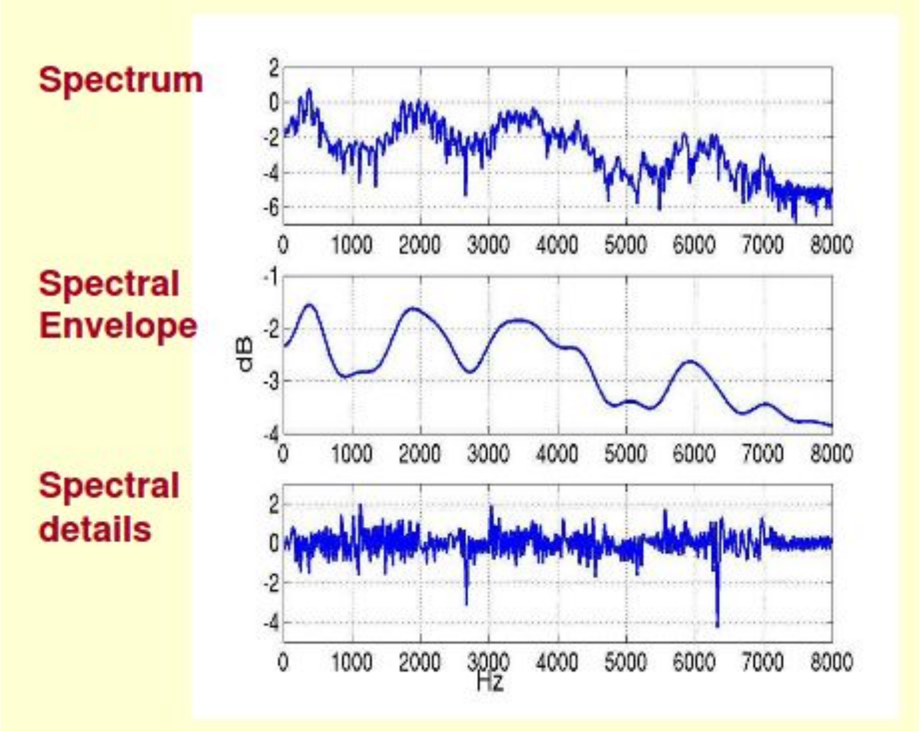
$$RMSE = \sqrt{\frac{1}{N} \sum_{n} |x(n)|^2}$$

MFCC (Mel-Frequency Cepstral Coefficients)



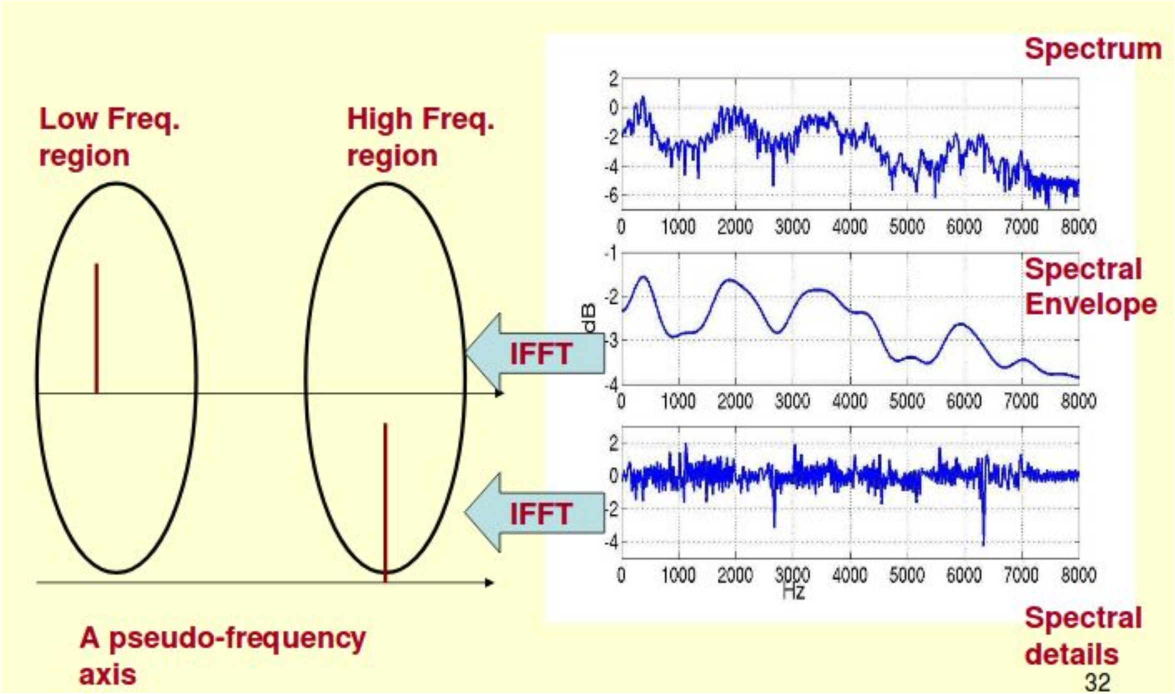
Spectral representation of the signal
Goal: Capture the envelope - in brown - which best characterizes the signal

MFCC (Mel-Frequency Cepstral Coefficients)



Be H[k] the spectral envelope and E[k] the spectral details. Hence log(X[k]) = log(H[k]) + log(E[k]). In practice we don't have neither H[k] nor E[k]. **How can we make the separation? Inverse transform!!!**

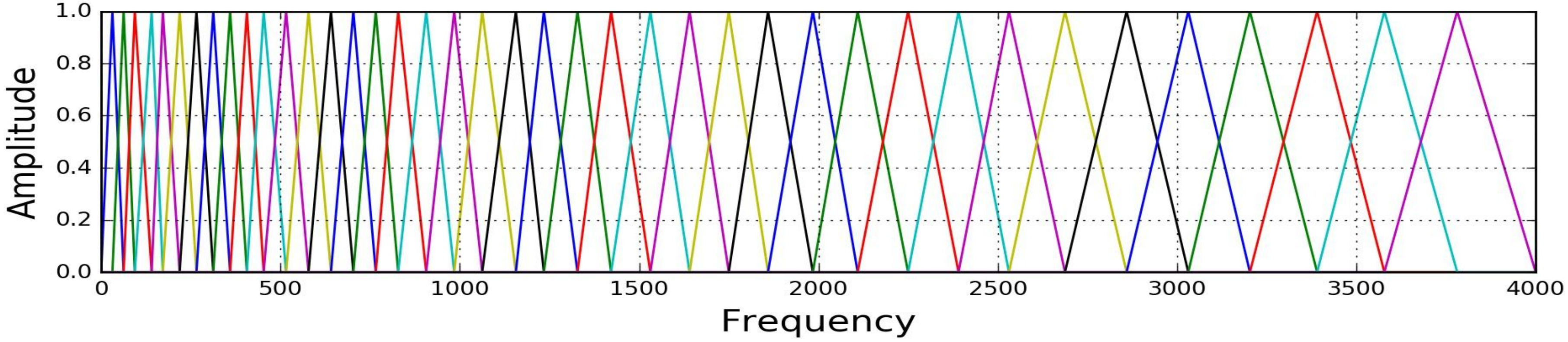
MFCC (Mel-Frequency Cepstral Coefficients)



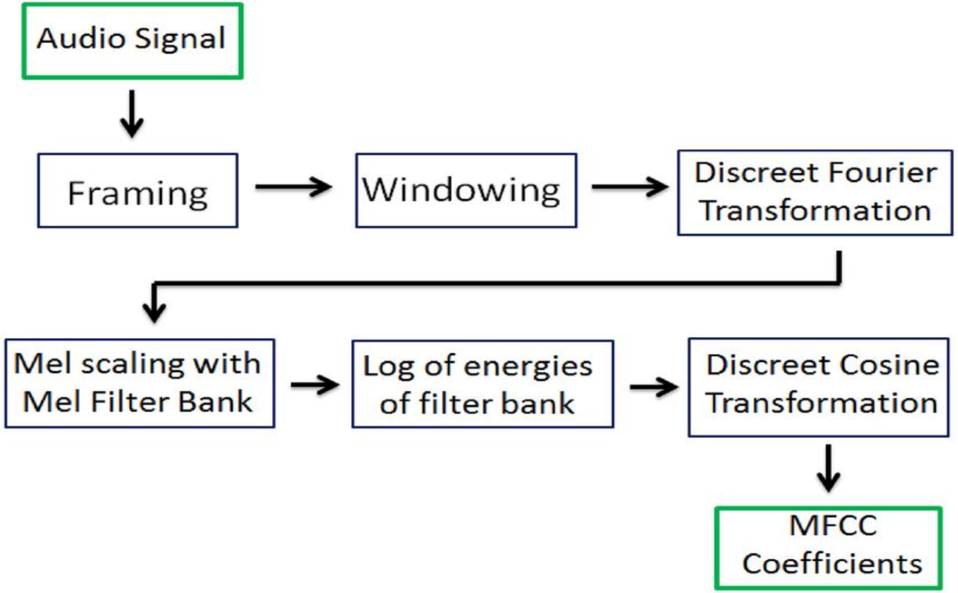
Applying low band pass filter we obtain the desired part. Now we go back to the previous domain. **Now, which filter we use? One that represents human perception!**

The Mel Filter Bank

- The final step to computing filter banks is applying triangular filters, typically 40 filters, nfilt = 40 on a Mel-scale to the power spectrum to extract frequency bands. The Mel-scale aims to mimic the non-linear human ear perception of sound, by being more discriminative at lower frequencies and less discriminative at higher frequencies.



MFCC (Mel-Frequency Cepstral Coefficients)



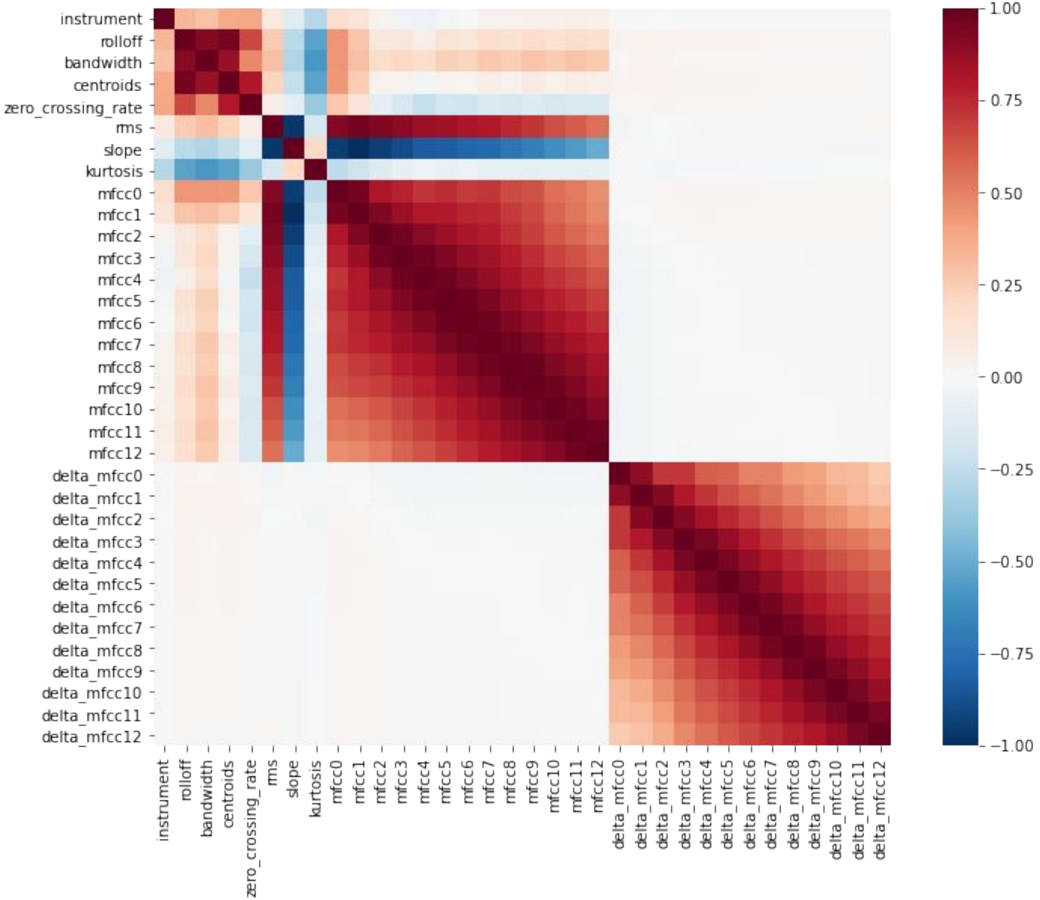
Cosine Transform

$$y(n) = \sqrt{\frac{2}{N}} C(n) \sum_{k=0}^{N-1} x(k) \cos \frac{(2k+1)n\pi}{2N}$$

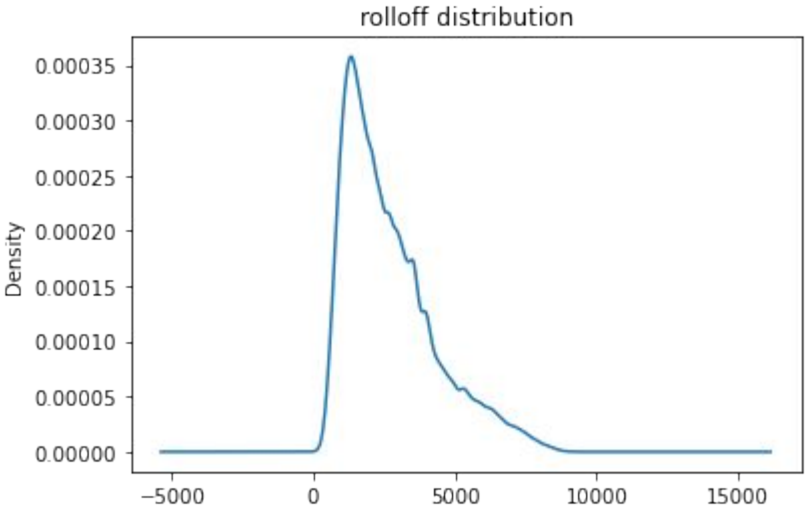
$$n = 0, 1, \ldots, N-1$$

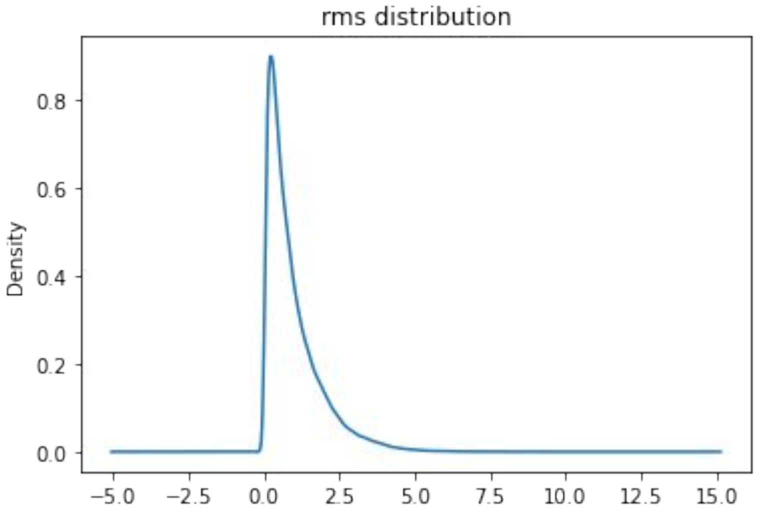
$$C(n) = \begin{cases} \frac{1}{\sqrt{2}} & , n = 0 \\ 1 & , n \neq 0 \end{cases}$$

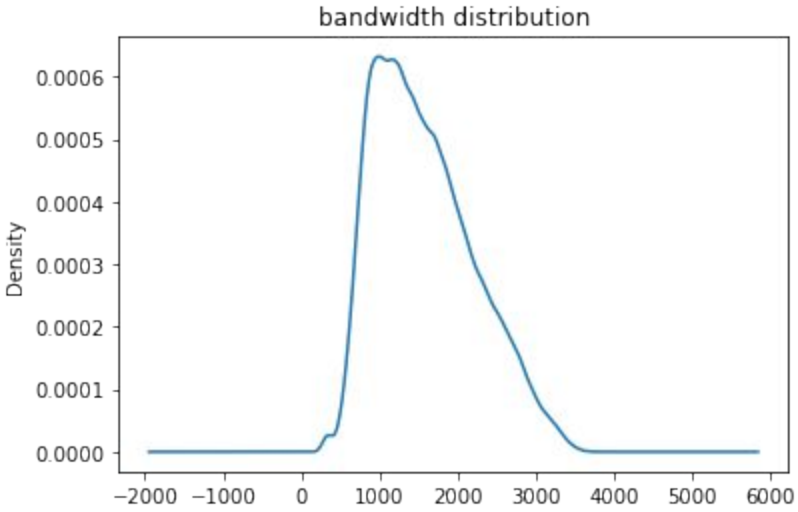
Exploratory Data Analysis



Data Distribution







Models

With Spark:

RandomForest- Area Under the Curve: 0.845

(best params: 32 trees and max depth 8)

With Elephas:

Convolution Neural Network- Area Under the Curve: 0.444

(20 epochs)

Presentation Link

https://www.youtube.com/watch?v=jU-PzGNywcw

References

[1] Understanding LSTM Networks. <https://colah.github.io/posts/2015-08-Understanding-LSTMs/>. Access: October 09, 2020.

[2] IRMAS: a dataset for instrument recognition in musical audio signals. <https://www.upf.edu/web/mtg/irmas>. Access: October 09, 2020

[3] Bosch, J. J., Janer, J., Fuhrmann, F., & Herrera, P. “A Comparison of Sound Segregation Techniques for Predominant Instrument Recognition in Musical Audio Signals”, in Proc. ISMIR (pp. 559-564), 2012

[4] MCKINNEY, Martin; BREEBAART, Jeroen. Features for audio and music classification. 2003.

[5] Audio Data Analysis Using Deep Learning with Python (Part 1). <https://www.kdnuggets.com/2020/02/audio-data-analysis-deep-learning-python-part-1.html> Access: October 13, 2020.