LEARNING PITCH INVARIANTS FOR INSTRUMENT RECOGNITION

Names should be omitted for double-blind reviewing
Affiliations should be omitted for double-blind reviewing

ABSTRACT

Musical performance combines a wide range of pitches, nuances, and expressive techniques. Audio-based classification of musical instruments thus requires to build signal representations that are invariant to such transformations. Focusing on pitch invariance, this article investigates the construction of multi-stage architectures for instrument recognition. We show that Mel-frequency cepstral coefficients (MFCC) lack invariance with respect to realistic pitch shifts. In turn, a convolutional neural network (ConvNet) in the time-frequency domain is able to disentangle pitch variability from timbral information in a subtler way. We further improve the ConvNet architecture by limiting weight sharing to octave-wide frequency bands at the first layer, while allowing full weight sharing at deeper layers. We extend our method to the recognition of multiple instruments playing simultaneously.

1. INTRODUCTION

Among the cognitive attributes of musical tones, pitch is distinguished by a combination of three properties. First, it is relative: ordering pitches from low to high gives rise to intervals and melodic patterns. Secondly, it is intensive: multiple pitches heard simultaneously produce a chord, not a single unified tone – contrary to loudness, which adds up with the number of sources. Thirdly, it is invariant to instrumentation: this makes possible the transcription of polyphonic music under a single symbolic system. Section 2 demonstrates that pitch is the major factor of variability among musical notes of a given instrument, if described by their Mel-frequency cepstra. Section 3 describes a typical deep learning architecture for spectrogram-based classification, consisting of two convolutional layers and one densely connected layer. Section 4 improves the aforementioned architecture by splitting spectrograms into octave-wide frequency bands, training specific convolutional layers over each band in parallel, and gathering feature maps at a later stage. Section 5 discusses the effectiveness of the presented systems on a challenging dataset for music instrument recognition.

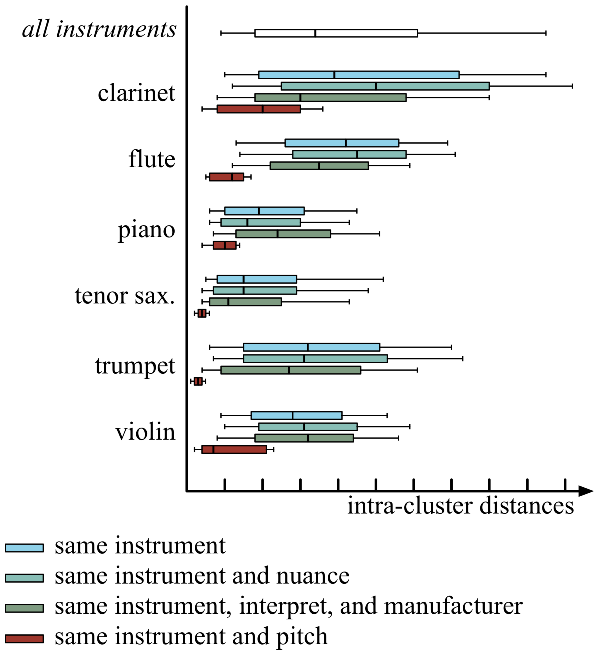

Figure 1: Distributions of squared Euclidean distances among various clusters in the RWC dataset. Whisker ends denote lower and upper deciles. See text for details.

2. HOW INVARIANT IS THE MEL-FREQUENCY CEPSTRUM ?

The mel scale is a quasi-logarithmic function of acoustic frequency designed such that perceptually similar pitch intervals appear equal in width over the full hearing range. Tuning band-pass filters to the mel scale

The MFCCs were extracted from a filterbank of 40 Mel-frequency bands and 13 discrete cosine transform coefficients.

3. DEEP CONVOLUTIONAL NETWORKS

A deep learning system for classification is built by stacking multiple layers of weakly nonlinear transformations, whose parameters are jointly optimized such that the top-level layer fits a training set of labeled examples. This section introduces a typical deep learning architecture for audio classification and describes the functioning of each layer.

The input of our system is a constant-Q wavelet scalogram, which is very comparable to a mel-frequency spectrogram. We used the implementation from the librosa

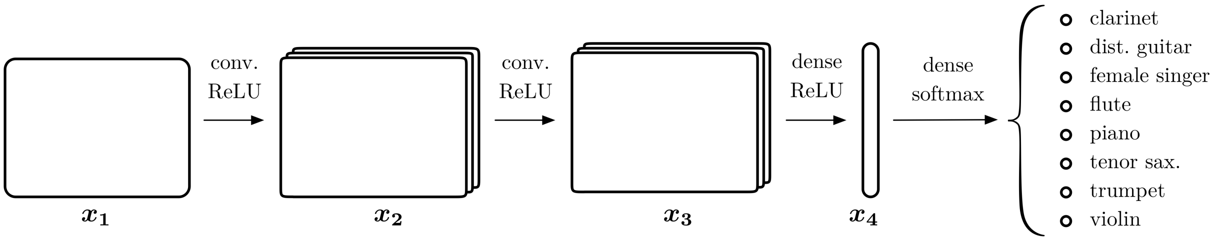

Figure 2: Architecture of a convolutional network with full weight sharing. See text for details.

package [5] with $Q = 12$ filters per octave, center frequencies ranging from 55 Hz to 14 kHz (8 octaves from A1 to A9), and a hop size of 23 ms. Furthermore, we applied nonlinear perceptual weighting of loudness in order to reduce the dynamic range between the fundamental partial and its upper harmonics. A 3-second sound excerpt $\boldsymbol{x}[t]$ is represented by a time-frequency matrix $\boldsymbol{x_1}[t, k_1]$ of width $T = 128$ samples and height $K_1 = 96$ frequency bands.

Each layer in a convolutional network typically consists in the composition of three operations: two-dimensional convolutions, application of a pointwise nonlinearity, and local pooling. A convolutional operator is defined as a family $\boldsymbol{W_2}[\tau, \kappa_1, k_2]$ of $K_2 = 32$ two-dimensional filters, whose impulse repsonses are all constrained to have width Δt and height Δk_1. Element-wise biases $\boldsymbol{b_2}[k_2]$ are added to the convolutions, resulting in the three-way tensor

$$\begin{aligned}
&\boldsymbol{y_2}[t, k_1, k_2] \\
&= \boldsymbol{b_2}[k_2] + \boldsymbol{W_2}[t, k_1, k_2] \overset{t,k_1}{*} \boldsymbol{x_1}[t, k_1] \\
&= \boldsymbol{b_2}[k_2] + \sum_{\substack{0 \leq \tau < \Delta t \\ 0 \leq \kappa_1 < \Delta k_1}} \boldsymbol{W_2}[\tau, \kappa_1, k_2] \boldsymbol{x_1}[t - \tau, k_1 - \kappa_1]. \quad (1)
\end{aligned}$$

The pointwise nonlinearity we have chosen the *rectified linear unit* (ReLU) because of its popularity in computer vision and its computational efficiency.

$$\boldsymbol{y_2^+}[t, k_1, k_2] = \max\left(\boldsymbol{y_2}[t, k_1, k_2], 0\right) \quad (2)$$

The pooling step consists in retaining the maximal activation among neighboring units in the time-frequency domain (t, k_1) over non-overlapping rectangles of width Δt and height Δk_1.

$$\boldsymbol{x_2}[t, k_1, k_2] = \max_{\substack{0 \leq \tau < \Delta t \\ 0 \leq \kappa_1 < \Delta k_1}} \left\{ \boldsymbol{y_2^+}[t - \tau, k_1 - \kappa_1, k_2] \right\} \quad (3)$$

The hidden units in $\boldsymbol{x_2}$ are in turn fed to a second layer of convolutions, ReLU, and pooling. Observe that the convolutional operator $\boldsymbol{W_3}[\tau, \kappa_1, k_2, k_3]$ performs a linear combination of time-frequency feature maps in $\boldsymbol{x_2}$ along the channel variable k_2.

$$\begin{aligned}
&\boldsymbol{y_3}[t, k_1, k_3] \\
&= \sum_{k_2} \boldsymbol{b_3}[k_2, k_3] + \boldsymbol{W_3}[t, k_1, k_2, k_3] \overset{t,k_1}{*} \boldsymbol{x_2}[t, k_1, k_2]. \quad (4)
\end{aligned}$$

$$\boldsymbol{y_4}[k_4] = \sum_{t, k_1, k_3} \boldsymbol{W_4}[t, k_1, k_3, k_4] \boldsymbol{x_3}[t, k_1, k_3] \quad (5)$$

We apply a ReLU to $\boldsymbol{y_4}$, yielding $\boldsymbol{x_4}[k_4] = \boldsymbol{y_4^+}[k_4]$. $\boldsymbol{y_5}[k_5] = \sum_{k_4} \boldsymbol{W_5}[k_4, k_5] \boldsymbol{x_4}[k_4]$.

$$\boldsymbol{x_5}[k_5] = \frac{\exp \boldsymbol{y_5}[k_5]}{\sum_{\kappa_5} \exp \boldsymbol{y_5}[\kappa_5]} \quad (6)$$

The above ensures that the coefficients of $\boldsymbol{x_5}$ are non-negative and sum to one, hence can be fit to a probability distribution.

$$\mathscr{L}(\boldsymbol{x_5}, \mathcal{I}) = -\sum_{k_5 \in \mathcal{I}} \log \boldsymbol{x_5}[k_5] + \sum_{m=1}^{4} \lambda_m \|\boldsymbol{W_m}\|_2. \quad (7)$$

The goal is to minimize the average loss $\mathscr{L}(\boldsymbol{x_5}, \mathcal{I})$ for across all pairs $(\boldsymbol{x}, \mathcal{I})$ in the training set.

The network is trained on categorical cross-entropy over shuffled mini-batches of size 512 with uniform class distribution. Each training example is a 3-second spectrogram whose boundaries are taken at random within the training set. The learning rate policy for each scalar weight in the network is *Adam* [4], a state-of-the-art online optimizer for gradient-based learning. The architecture was built using the Keras library, and trained on a Titan X graphics processing unit within a few minutes.

4. LIMITED WEIGHT SHARING

An Euclidean division of k_1 by Q yields $k_1 = j_1 \times Q + \chi_1$.

$$\begin{aligned}
\boldsymbol{y_2}[t, k_1, k_2] = \quad & \boldsymbol{b_2}[j_1, k_2] \\
& + \boldsymbol{W_2}[t, \chi_1, j_1, k_2] \overset{t,\chi_1}{*} \boldsymbol{x_1}[t, \chi_1, j_1]. \quad (8)
\end{aligned}$$

Limited weight sharing has been introduced by Abdel-Hamid et al. [1].

5. SINGLE-INSTRUMENT CLASSIFICATION

5.1 Experimental design

In order to train the proposed algorithms, we used MedleyDB v1.1. [2], a dataset of 122 multitracks annotated

	minutes	tracks	minutes	tracks
clarinet	10	7	13	18
dist. guitar	15	14	17	11
female singer	10	11	19	12
flute	7	5	53	29
piano	58	28	44	15
tenor sax.	3	3	6	5
trumpet	4	6	7	27
violin	51	14	49	22
total	158	88	208	139

Table 1

Representation	Error rate (%)
MFCC & random forest	–
ConvNet, full weight sharing	–
ConvNet, limited weight sharing	–

Table 2

with instrument activations as well as melodic f_0 curves when present. We extracted the monophonic stems corresponding to a selection of eight pitched instruments [see Figure 2]. Stems with leaking instruments in the background were discarded. The evaluation set consists of 120 recordings of solo music collected by Joder et al. [3]. We discarded recordings with extended instrumental techniques, since they are under-represented in MedleyDB.

5.2 Results

Results are charted in Table 2.

6. POLYPHONIC CLASSIFICATION

6.1 Experimental design

6.2 Results

Results are charted in Table 3.

7. CONCLUSIONS

Understanding the influence of pitch in audio streams is paramount to the design of an efficient system for automated classification, tagging, and similarity retrieval in music.

Representation	Error rate (%)
MFCC & random forest	–
ConvNet, full weight sharing	–
ConvNet, limited weight sharing	–

Table 3

8. REFERENCES

[1] Ossama Abdel-Hamid, Abdel-rahman Mohamed, Hui Jiang, Li Deng, Gerald Penn, and Dong Yu. Convolutional neural networks for speech recognition. *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, 22(10):1533–1545, 2014.

[2] Rachel Bittner, Justin Salamon, Mike Tierney, Matthias Mauch, Chris Cannam, and Juan Bello. Medleydb: a multitrack dataset for annotation-intensive mir research. *International Society for Music Information Retrieval Conference*, 2014.

[3] Cyril Joder, Slim Essid, and Gaël Richard. Temporal integration for audio classification with application to musical instrument classification. *IEEE Transactions on Audio, Speech and Language Processing*, 17(1):174–186, 2009.

[4] Diederik P. Kingma and Jimmy Lei Ba. Adam: a Method for Stochastic Optimization. *International Conference on Learning Representations*, pages 1–13, 2015.

[5] Brian McFee, Matt McVicar, Colin Raffel, Dawen Liang, Oriol Nieto, Eric Battenberg, Josh Moore, Dan Ellis, Ryuichi Yamamoto, Rachel Bittner, Douglas Repetto, Petr Viktorin, Joo Felipe Santos, and Adrian Holovaty. librosa: 0.4.1. zenodo. 10.5281/zenodo.18369, October 2015.