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 from timbral information in a subtler way. 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 does not depend on instrumentation: this makes possible the transcription of polyphonic music under a single symbolic system [6].

Tuning auditory filters to a perceptual scale of pitches provides a time-frequency representation of music signals that satisfies the first two of these properties. It is thus a starting point for a wide range of MIR applications, which can be separated in two categories: pitch-*relative* (e.g. chord estimation [10]) and pitch-*invariant* (e.g. instrument recognition [7]). Both aim at disentangling pitch from timbral content as independent factors of variability, a goal that is made possible by the third aforementioned property. This is pursued by extracting mid-level features on top of the spectrogram, be them engineered or learned from training data. Both approaches have their limitations: a "bag-of-features" lacks flexibility to represent fine-grain class boundaries, whereas a purely learned pipeline often leads to uninterpretable overfitting, especially in MIR where the quantity of thoroughly annotated data is relatively small.

In this article, we strive to integrate domain-specific knowledge about musical pitch into a deep learning framework, in an effort towards bridging the gap between feature engineering and feature learning.

Section 2 reviews the related work on feature learning for signal-based music classification. Section 3 demonstrates that pitch is the major factor of variability among musical notes of a given instrument, if described by their mel-frequency cepstra. Section 4 describes a typical deep learning architecture for spectrogram-based classification, consisting of two convolutional layers and one densely connected layer. Section 5 improves the previous 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. Sections 6 discusses the effectiveness of the presented systems on a challenging dataset for music instrument recognition.

2. RELATED WORK

Spurred by the growth of annotated datasets and the democratization of high-performance computing, feature learning has enjoyed a renewed interest in recent years within the MIR community. Whereas unsupervised learning (e.g. k-means, Gaussian mixtures) is employed to fit the distribution of the data with few parameters of relatively low abstraction and high dimensionality, state-of-the-art supervised learning consists of a composition of multiple nonlinear transformations, jointly optimized to predict class labels, and whose behaviour gain in abstraction as depth increases [20].

The success of convolutional representations rely on a stationarity assumption, which claims that the input spectrograms are made of small time-frequency patches whose content is statistically independent from their location. As a consequence, linear transformations of the data can be learned efficiently by enforcing the transformation of every patch to be equal across the whole input. This method, known as weight sharing, decreases the number of parameters of each feature map while increasing the amount of available training data. For music instrument recognition: [15], [14].

Some other applications include onset detection [17], transcription [18], genre classification [4], chord recognition [10], boundary detection [19], and recommendation [20].

The most widely studied deep learning system for music information retrieval consists of two convolutional layers and two densely connected layers, with minor variations

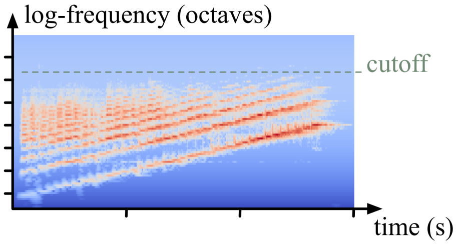

Figure 1: Constant-Q spectrogram of a chromatic scale played by a tuba. Although the harmonic partials shift progressively, the spectral envelope remains unchanged, as revealed by the presence of a fixed cutoff frequency. See text for details.

[10, 12, 14, 15, 17, 19].

3. 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. This section shows that engineering transposition-invariant features from the mel scale does not suffice to build pitch invariants for complex sounds, thus motivating further inquiry.

The time-frequency domain produced by a constant-Q filter bank tuned to the mel scale is covariant with respect to pitch transposition of pure tones. As a result, a chromatic scale played at constant speed would draw parallel, diagonal lines, each of them corresponding to a different partial wave. However, the physics of musical instruments constrain these partial waves to bear a negligible energy if their frequencies are beyond the range of acoustic resonance.

As shown on Figure 1, the constant-Q spectrogram of a tuba chromatic scale exhibits a fixed, cutoff frequency at about 2500 Hz, which delineates the support of its spectral envelope. This elementary observation implies that realistic pitch changes cannot be modeled by translating a rigid spectral template along the log-frequency axis. The same property is verified for a wide class of instruments, especially brass and woodwinds. As a consequence, the construction of powerful invariants to musical pitch is not amenable to delocalized operations on the mel-frequency spectrum, such as a discrete cosine transform (DCT) which leads to the mel-frequency cepstral coefficients (MFCC) classically used in music classification [7, 11].

To validate the above claim, we have extracted the MFCC of 2688 individual notes from the RWC dataset [8], as played by six instruments, with varying pitches, nuances, interprets, and manufacturers. Following a well-established rule [7, 11], the MFCC were defined the 13 lowest "quefrencies" among the DCT coefficients extracted from a filter bank of 40 mel-frequency bands. We then have computed the distribution of squared Euclidean distances between musical notes in the 13-dimensional space

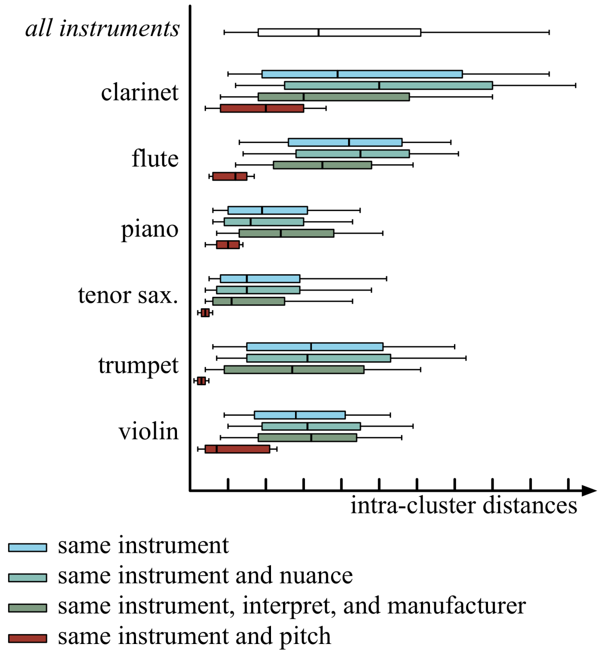

Figure 2: Distributions of squared Euclidean distances among various MFCC clusters in the RWC dataset. Whisker ends denote lower and upper deciles. See text for details.

of MFCC features.

Figure 2 summarizes our results. We found that restricting the cluster to one nuance, one interpret, or one manufacturer hardly reduces intra-class distances. This suggests that MFCC are fairly successful in building invariant representations to such factors of variability. In contrast, the cluster corresponding to each instrument is shrinked if decomposed into a mixture of same-pitch clusters, sometimes by an order of magnitude. In other words, most of the variance in an instrument cluster of mel-frequency cepstra is due to pitch transposition.

Keeping less than 13 coefficients certainly improves invariance, yet at the cost of inter-class discriminability, and vice versa. This experiment shows that the mel-frequency cepstrum is perfectible in terms of invariance-discriminability tradeoff, and that there remains a lot to be gained by feature learning in this area.

4. DEEP CONVOLUTIONAL NETWORKS

A deep learning system for classification is built by stacking multiple layers of weakly nonlinear transformations, whose parameters are 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 package [16] with $Q = 12$ filters per octave, center frequencies ranging from 55 Hz to 14 kHz (8 octaves from

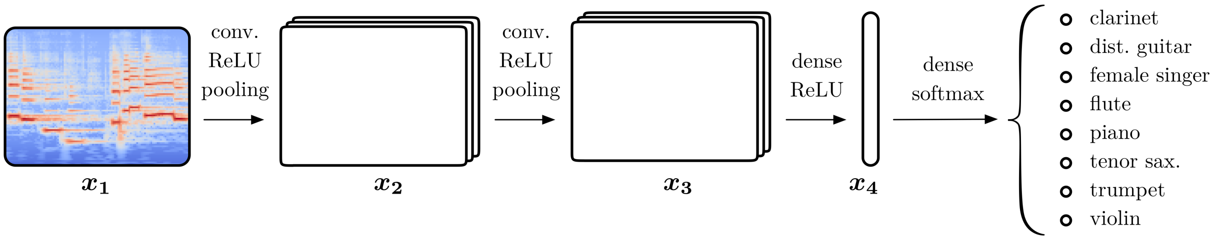

Figure 3: Architecture of a convolutional network with full weight sharing. See text for details.

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 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 is the rectified linear unit (ReLU), with a rectifying slope of $\alpha = 0.3$ for negative inputs.

$$\boldsymbol{y_2^+}[t, k_1, k_2] = \begin{cases} \alpha \boldsymbol{x_2}[t, k_1, k_2] & \text{if } \boldsymbol{x_2}[t, k_1, k_2] < 0 \\ \boldsymbol{x_2}[t, k_1, k_2] & \text{if } \boldsymbol{x_2}[t, k_1, k_2] > 0 \end{cases} \quad (2)$$

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 corresponding 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.

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

Tensors $\boldsymbol{y_3}^+$ and $\boldsymbol{x_3}$ are derived from $\boldsymbol{y_3}$ by ReLU and pooling, with formulae similar to Eqs. (2) and (3). The third layer consists of the linear projection of $\boldsymbol{x_3}$, viewed as a vector of the flattened index (t, k_1, k_3), over K_4 units:

$$\boldsymbol{y_4}[k_4] = \boldsymbol{b_4}[k_4] + \sum_{t, k_1, k_3} \boldsymbol{W_4}[t, k_1, k_3, k_4] \boldsymbol{x_3}[t, k_1, k_3] \quad (5)$$

We apply a ReLU to $\boldsymbol{y_4}$, yielding $\boldsymbol{x_4}[k_4] = \boldsymbol{y_4^+}[k_4]$. Finally, we project $\boldsymbol{x_4}$ onto a layer of output units $\boldsymbol{y_5}$ that should represent instrument activations: $\boldsymbol{y_5}[k_5] = \sum_{k_4} \boldsymbol{W_5}[k_4, k_5] \boldsymbol{x_4}[k_4]$. The final transformation is a softmax nonlinearity, that ensures that output coefficients are non-negative and sum to one, hence can be fit to a probability distribution.

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

The goal is to minimize the average loss $\mathscr{L}(\boldsymbol{x_5}, \mathcal{I})$ across all pairs $(\boldsymbol{x}, \mathcal{I})$ in the training set. This loss is defined as the categorical cross-entropy over shuffled mini-batches of size 64 with uniform class distribution, to which a weight decay term is added.

$$\mathscr{L}(\boldsymbol{x_5}, \mathcal{I}) = - \sum_{k_5 \in \mathcal{I}} \log \boldsymbol{x_5}[k_5] + \lambda_5 \|W_5\| \quad (7)$$

Each training example is a 3-second spectrogram whose boundaries are selected at random over non-silent regions of a song. Each spectrogram within a batch was globally normalized such that the whole batch had unit mean and unit variance. The learning rate policy for each scalar weight in the network is *Adam* [13], a state-of-the-art online optimizer for gradient-based learning. The architecture was built using the Keras library [5], and trained on a graphics processing unit within a few minutes

5. CONVOLUTIONS ON THE PITCH SPIRAL

However, although a dataset of music signals is unquestionably stationary over the time dimension – at least at the scale of a few seconds – it cannot be taken for granted that the frequency axis is

"Local neighborhoods in frequency do not share the same relationship", Humphrey *et al.* acknowledge in their review of deep learning techniques for MIR.

Indeed, since the neighboring partials of a harmonic sound are evenly spaced in frequency, they tend to get closer to each other on a mel scale. Due to the Heisenberg principle, the temporal resolution of auditory filters is lessened at lower frequencies, which results in a blurrier constant-Q spectrogram over the horizontal dimension.

Let $\boldsymbol{\phi}[k_1]$ be a window function of width $3Q$, that is three octaves. We have chosen a Tukey window ($\alpha = 0.33$), which has a flat top of width Q surrounded by cosine tapering lobes of width Q.

$$\begin{aligned} \boldsymbol{y_2}[t, k_1, k_2] = \boldsymbol{b_2}[k_2] \\ + \sum_{\tau, \kappa_1, j_1} \boldsymbol{W_2}[\tau, \kappa_1, j_1, k_2] \\ \times \boldsymbol{x_1}[t - \tau, k_1 - \kappa_1] \\ \times \boldsymbol{\phi}[k_1 - \kappa_1 - Q j_1]. \end{aligned} \tag{8}$$

Compare the above with Equation (1). Limited weight sharing has been introduced by Abdel-Hamid et al. [1].

6. APPLICATIONS

6.1 Experimental design

In order to train the proposed algorithms, we used MedleyDB v1.1. [3], a dataset of 122 multitracks annotated 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 Table 1). Stems with leaking instruments in the background were discarded.

The evaluation set consists of 126 recordings of solo music collected by Joder *et al.* [11], to which we add 23 stems of electric guitar and female voice from MedleyDB. In doing so, guitarists and vocalists were thoroughly put either in the training set or the test set, to prevent any artist bias. We discarded recordings with extended instrumental techniques, since they are very rare in MedleyDB.

Constant-Q spectrograms from the evaluation set were split into half-overlapping, 3-second excerpts. The predicted probability distributions were computed for every excerpt in a track, and then aggregated by geometric mean.

6.2 Results

Results are charted in Table 2.

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. We have presented a data-driven, supervised method to address pitch invariance while preserving good timbral discriminability. Future work will be devoted to integrating the proposed scheme with other advances in deep learning for music informatics, such as data augmentation [15], multiscale representations [2, 9], and adversarial training [12].

	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 forrest	–
ConvNet, full weight sharing	–
ConvNet, limited weight sharing	–

Table 2

8. REFERENCES

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

[2] Joakim Andén, Vincent Lostanlen, and Stéphane Mallat. Joint Time-Frequency Scattering for Audio Classification. In *Machine Learning for Signal Processing*, 2015.

[3] Rachel Bittner, Justin Salamon, Mike Tierney, Matthias Mauch, Chris Cannam, and Juan Bello. MedleyDB: a multitrack dataset for annotation-intensive MIR research. In *Proc. ISMIR*, 2014.

[4] Keunwoo Choi, George Fazekas, Mark Sandler, Jeonghee Kim, and Naver Labs. Auralisation of deep convolutional neural networks: listening to learned features. *ISMIR*, pages 4–5, 2015.

[5] François Chollet. Keras: a deep learning library for theano and tensorflow. https://github.com/fchollet/keras, 2015.

[6] Alain de Cheveigné. Pitch perception. In *Oxford Handbook of Auditory Science: Hearing*, chapter Pitch perc, pages 71–104. 2005.

[7] Antti Eronen and Anssi Klapuri. Musical instrument recognition using cepstral coefficients and temporal features. In *Acoustics, Speech, and Signal Processing, 2000. ICASSP'00. Proceedings. 2000 IEEE International Conference on*, volume 2, pages II753–II756. IEEE, 2000.

[8] Masataka Goto, Hiroki Hashiguchi, Takuichi Nishimura, and Ryuichi Oka. Rwc music database: Music genre database and musical instrument sound database. 2003.

[9] Philippe Hamel, Yoshua Bengio, and Douglas Eck. Building musically-relevant audio features through multiple timescale representations. In *ISMIR*, pages 553–558, 2012.

[10] Eric J Humphrey, Taemin Cho, and Juan P Bello. Learning a robust tonnetz-space transform for automatic chord recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pages 453–456. IEEE, 2012.

[11] 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.

[12] Corey Kereliuk, Bob L. Sturm, and Jan Larsen. Deep Learning and Music Adversaries. *IEEE Transactions on Multimedia*, 17(11):2059–2071, 2015.

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

[14] Peter Li, Jiyuan Qian, and Tian Wang. Automatic Instrument Recognition in Polyphonic Music Using Convolutional Neural Networks. *arXiv preprint*, 2015.

[15] Brian Mcfee, Eric J Humphrey, and Juan P Bello. A software framework for musical data augmentation. In *ISMIR*, pages 248–254, 2015.

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

[17] Jan Schlüter and Sebastian Bock. Improved musical onset detection with convolutional neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 6979–6983. IEEE, 2014.

[18] Siddharth Sigtia, Emmanouil Benetos, and Simon Dixon. An End-to-End Neural Network for Polyphonic Music Transcription. pages 1–13, 2015.

[19] Karen Ullrich, Jan Schlüter, and Thomas Grill. Boundary Detection in Music Structure Analysis Using Convolutional Neural Networks. *Proc. of the 15th International Society for Music Information Retrieval Conference*, pages 417–422, 2014.

[20] Aaron van den Oord, Sander Dieleman, and Benjamin Schrauwen. Deep content-based music recommendation. *NIPS*, pages 2643–2651, 2013.