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. This article investigates the construction of multistage architectures for instrument recognition, given a limited amount of annotated training data. We show that the main drawback of mel-frequency cepstral coefficients (MFCC) resides in their lack of invariance with respect to realistic pitch shifts, despite being designed to be invariant to the frequency transposition of pure tones. In turn, a deep convolutional network (ConvNet) in the time-frequency domain is able to disentangle pitch from timbral information, hence yielding better classification accuracy. By recombining convolutional feature maps over the Shepard pitch spiral, we further improve the learned representation by introducing weight sharing strategies dedicated to quasi-harmonic sounds with fixed spectral envelope, which are archetypal of musical notes.

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 [5].

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 [11]) 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, both in supervised and unsupervised settings. Whereas unsupervised learning (e.g. k-means [23], Gaussian mixtures [12]) 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 [25].

As compared to other deep learning techniques for audio processing, convolutional networks happen to strike the balance between learning capacity and robustness. The convolutional structure of learned transformations is derived from the assumption that input data, be it a one-dimensional waveform or a two-dimensional spectrogram, is stationary ; which means that content is independent from location. Moreover, the most informative dependencies between signal coefficients are assumed to be concentrated to neighborhoods in time and/or frequency. Under such hypotheses, linear transformations of the data can be

learned efficiently by limiting their support to a small kernel which is convolved over 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.

Newton and Smith [19] have built a network to represent the temporal dynamics of isolated musical notes. They created a neurally inspired tone descriptor using a model of the auditory system's response to sound onset, based on dynamic synapses and leaky integrate-and-fire neurons. Final classification is then performed using an echo-state network in the time domain.

McFee et al. [17] have trained a deep convolutional network on constant-Q representations of music instruments in order to illustrate the benefits of artificial data augmentation. Using two convolutional layers followed by two densely connected layers, the authors successfully proved that the overfitting of the network is greatly reduced by randomly manipulating the data with digital audio effects, such as frequency transposition, time stretching, and the addition of realistic background noise.

Li et al. [15] have also used a deep convolutional network but trained directly on raw audio waveforms for polyphonic instrument recognition. Their aim was to prove that and end-to-end approach (from the learning of features to the classification task) outperform more traditional approaches. The authors also remark that the filters learned in the first layer seem to be frequency selective and similar to an auditory scale filter bank.

Some other applications of deep convolutional networks include onset detection [21], transcription [22], genre classification [3], chord recognition [11], boundary detection [24], and recommendation [25].

As a matter of facts, the most widely studied deep learning system for music information retrieval consists of two convolutional layers and two densely connected layers, with minor variations [6, 11, 13, 15, 17, 21, 24].

There is little to none theoretical examination on this kind of networks and the reason for which the discussed architecture is the most accepted for audio analysis is not totally clear. Convolutional layers are aimed at learning factors of variability with different degree of abstraction. The joint action of convolution filters and non linear operators seems to implement the so-called *contraction* and *separation* properties of deep convolutional networks [16]: the former creates a non-linear reduction of the feature space to decrease the data variability along local symmetries. The latter, on the other hand, avoids that elements belonging to different classes clash together as consequence of the contraction.

The deepness of the network is a mean of controlling the ratio between contraction and separation and, indirectly, the abstraction level of the representation: we advance the hypothesis, in this context, that two layers of convolutions (with related non linearities) are the optimal trade-off for the complexity of the signals treated and for the size of the dataset used.

Many challenges, however, remain to be addressed. The process of gaining abstraction is somehow related, in deep networks, to the change of scale by means of pooling operators. In musical signals, therefore, it would be interesting to find specific time-frequency patterns over longer time scales. Moreover, in the context of music instrument classification, it would be interesting to build invariants to pitch and melody that preserve the separation among the instruments.

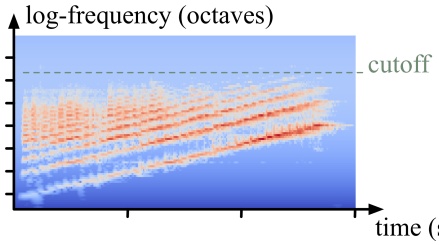

Figure 1: Constant-Q spectrogram of a chromatic scale played by a tuba. Although the harmonic partials shift progressively, the spectral envelope remains unchanged, as revealed by the presence of a fixed cutoff frequency. See text for details.

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, 12].

To validate the above claim, we have extracted the MFCC of 1116 individual notes from the RWC dataset [8],

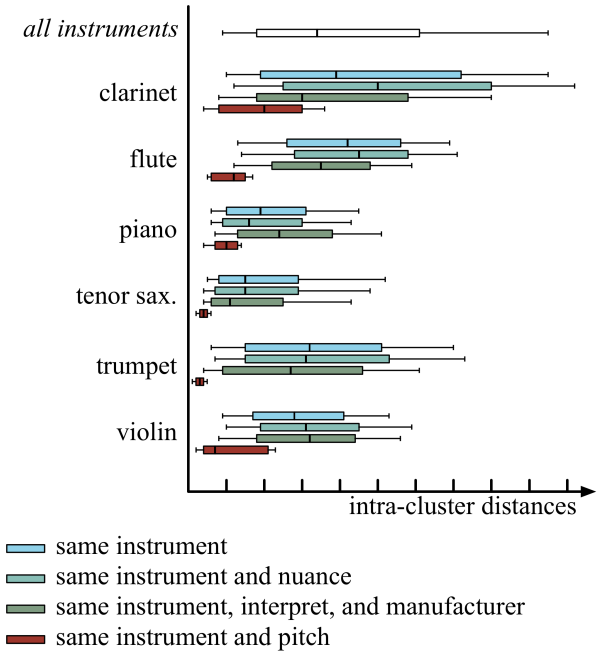

Figure 2: Distributions of squared Euclidean distances among various MFCC clusters in the RWC dataset. Whisker ends denote lower and upper deciles. See text for details.

as played by 6 instruments, with 32 pitches, 3 nuances, and 2 interprets and manufacturers. When more than 32 pitches were available (e.g. piano), we selected a contiguous subset of 32 pitches in the middle register. Following a well-established rule [7,12], the MFCC were defined the 12 lowest nonzero "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 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 [18] 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 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:

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

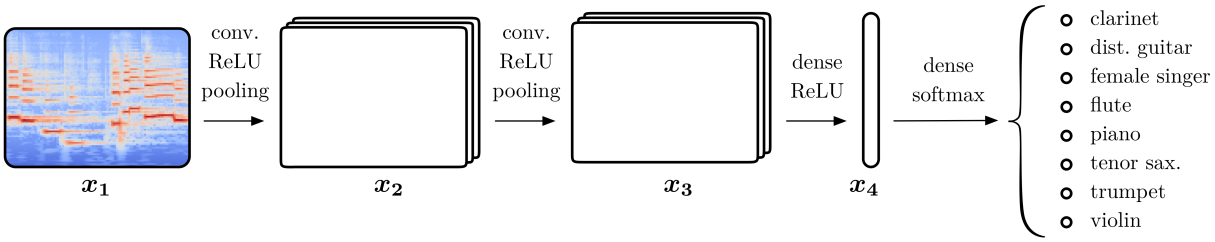

Figure 3: Architecture of a convolutional network with full weight sharing. See text for details.

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, which ensures that output coefficients are non-negative and sum to one, hence can be fit to a probability distribution.

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

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 32 with uniform class distribution, to which is added a weight decay term upon the last layer.

$$\mathscr{L}(\boldsymbol{x_5}, \mathcal{I}) = -\sum_{k_5 \in \mathcal{I}} \log \boldsymbol{x_5}[k_5] + \lambda_5 \|W_5\|_2 \tag{7}$$

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 zero mean and unit variance. The learning rate policy for each scalar weight in the network is Adam [14], a state-of-the-art online optimizer for gradient-based learning. Mini-batch training was stopped after the average training loss stopped decreasing over one full epoch of size 8192. The architecture was built using the Keras library [4], and trained on a graphics processing unit within a few minutes.

5. IMPROVED WEIGHT SHARING STRATEGIES

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 all frequency bands of a mel-frequency spectrogram would have the same local statistics [10]. In this section, we introduce two alternative architectures to address nonstationarity of music on the mel-frequency axis, while still leveraging the efficiency of convolutional representations in the time-frequency domain.

Some objections the stationarity assumption among local neighborhoods in frequency are the following: the spectral envelope of musical instruments remains fixed in spite of pitch changes (see Section 2) ; partials of a harmonic comb get closer to each other in high frequencies on a mel scale ; due to the Heisenberg principle, the temporal resolution of auditory filters is lessened at lower frequencies.

5.1 One-dimensional convolutions at high frequencies

Facing nonstationary constant-Q spectra, the most conservative workaround is to increase the height $\Delta\kappa_1$ of each convolutional kernel up to the total number of bins K_1 in the spectrogram. As a result, $\boldsymbol{W_1}$ and $\boldsymbol{W_2}$ are no longer transposed over adjacent frequency bands, since convolutions are merely performed over the time variable. The definition of $\boldsymbol{y_2}[t, k_1, k_2]$ rewrite as

$$\begin{aligned}
&\boldsymbol{y_2}[t, k_1, k_2] \\
&= \boldsymbol{b_2}[k_2] + \boldsymbol{W_2}[t, k_1, k_2] \overset{t}{*} \boldsymbol{x_1}[t, k_1] \\
&= \boldsymbol{b_2}[k_2] + \sum_{0 \leq \tau < \Delta t} \boldsymbol{W_2}[\tau, k_1, k_2]\boldsymbol{x_1}[t - \tau, k_1],
\end{aligned} \tag{8}$$

and similarly for $\boldsymbol{y_3}[t, k_1, k_3]$. While this approach is theoretically capable of encoding pitch invariants, it is prone to early specialization of low-level features, thus not fully taking advantage of the network depth.

However, the situation is improved if the one-dimensional kernels are restricted to the highest frequencies of the constant-Q spectrum. It should be observed that, around the n^{th} partial of a quasi-harmonic sound, the distance in log-frequency between neighboring partials decays like $1/n$, and the unevenness between those distances decays like $1/n^2$. Consequently, at the topmost octaves of the constant-Q spectrum, where n is equal or greater than Q, the partials appear close to each other and almost evenly spaced. Furthermore, due to the logarithmic compression of loudness, the polynomial decay of the spectral envelope is linearized: thus, at high frequencies, transposed pitches have similar spectra up to some additive bias. The combination of these two phenomena implies that the correlation between constant-Q spectra of different pitches is greater towards high frequencies, and that the learning of polyvalent feature maps becomes tractable.

5.2 Convolutions on the pitch spiral at low frequencies

The weight sharing strategy presented above exploits the facts that, at high frequencies, quasi-harmonic partials are numerous, and that the amount of energy within a frequency band is independent of pitch. At low frequencies, we make the exact opposite assumptions: we claim that the harmonic comb is sparse and covariant with respect to pitch shift. Observe that, for any two distinct partials taken at random between 1 and n, the probability that they are in octave relation is slightly above $1/n$.

Leveraging the fact that power-of-two harmonics are one octave apart, we roll up the log-frequency into a Shepard pitch spiral, such that octave intervals correspond to full turns.

$$\begin{aligned}\boldsymbol{y_2}[t, k_1, k_2] = &\boldsymbol{b_2}[k_2] \\ &+ \sum_{\tau, \kappa_1, j_1} \boldsymbol{W_2}[\tau, \kappa_1, j_1, k_2] \\ &\times \boldsymbol{x_1}[t - \tau, k_1 - \kappa_1 - Qj_1] \quad (9)\end{aligned}$$

The linear combinations of frequency bands that are one octave apart, as proposed here, bears a resemblance with engineered features for music instrument recognition [20], such as tristimulus, empirical inharmonicity, harmonic spectral deviation, odd-to-even harmonic energy ratio, as well as octave band signal intensities (OBSI) [12].

In summary, the classical two-dimensional convolutions make a stationarity assumption among frequency neighborhoods. This approach gives a coarse approximation of the spectral envelope. Resorting to one-dimensional convolutions allows to disregard nonstationarity, but does not yield a pitch-invariant representation per se: thus, we only apply them at the topmost frequencies, i.e. where the invariance-to-stationarity ratio in the data is already favorable. Conversely, two-dimensional convolutions on the pitch spiral addresses the invariant representation of sparse, transposition-covariant spectra: they are best suited to the lowest frequencies, i.e. where partials are further apart and pitch changes can be approximated by log-frequency translations.

The outputs of each transformed subband are finally aggregated into an unstructured vector.

6. APPLICATIONS

In order to train the proposed algorithms, we used MedleyDB v1.1. [2], 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. [12], supplemented with 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 extremely rare in MedleyDB.

	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

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 to provide a decision at the scale of the entire audio file.

In order to compare the results against shallow classifiers, we also extracted a typical "bag-of-features" over half-overlapping, 3-second excerpts in the training set. These features consist of the temporal averages and standard deviations of spectral shape descriptors, i.e. centroid, bandwidth, skewness, and rolloff, as well a zero-crossing rate ; supplemented with the temporal averages of MFCC and its first and second derivatives. We trained a random forest of 100 decision trees on the resulting feature vector of dimension

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 [17], multiscale representations [1, 9], and adversarial training [13].

8. REFERENCES

[1] Joakim Andén, Vincent Lostanlen, and Stéphane Mallat. Joint time-frequency scattering for audio classification. In *Proc. MLSP*, 2015.

[2] 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.

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

[4] François Chollet. Keras: a deep learning library for Theano and TensorFlow, 2015.

[5] Alain de Cheveigné. Pitch perception. In *Oxford Handbook of Auditory Science: Hearing*, chapter 4, pages 71–104. Oxford University Press, 2005.

[6] Sander Dieleman and Benjamin Schrauwen. End-to-end learning for music audio. In *Proc. ICASSP*, 2014.

[7] Antti Eronen and Anssi Klapuri. Musical instrument recognition using cepstral coefficients and temporal features. In *Proc. ICASSP*, 2000.

[8] Masataka Goto, Hiroki Hashiguchi, Takuichi Nishimura, and Ryuichi Oka. RWC music database: music genre database and musical instrument sound database. In *Proc. ISMIR*, 2003.

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

[10] Eric J. Humphrey, Juan P. Bello, and Yann Le Cun. Feature learning and deep architectures: New directions for music informatics. *JIIS*, 41(3):461–481, 2013.

[11] Eric J. Humphrey, Taemin Cho, and Juan P. Bello. Learning a robust tonnetz-space transform for automatic chord recognition. In *Proc. ICASSP*, 2012.

[12] Cyril Joder, Slim Essid, and Gaël Richard. Temporal integration for audio classification with application to musical instrument classification. *IEEE TASLP*, 17(1):174–186, 2009.

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

[14] Diederik P. Kingma and Jimmy Lei Ba. Adam: a method for stochastic optimization. In *Proc. ICML*, 2015.

[15] Peter Li, Jiyuan Qian, and Tian Wang. Automatic instrument recognition in polyphonic music using convolutional neural networks. *arXiv preprint*, (1511.05520), 2015.

[16] Stéthane Mallat. Understanding deep convolution networks. *arXiv preprint*, (1601.04920v1), 2016.

[17] Brian McFee, Eric J. Humphrey, and Juan P. Bello. A software framework for musical data augmentation. In *Proc. ISMIR*, 2015.

[18] 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.

[19] Michael J. Newton and Leslie S. Smith. A neurally inspired musical instrument classification system based upon the sound onset. *JASA*, 131(6):4785, 2012.

[20] Geoffroy Peeters. A large set of audio features for sound description (similarity and classification) in the CUIDADO project. Technical report, Ircam, 2004.

[21] Jan Schlüter and Sebastian Bock. Improved musical onset detection with convolutional neural networks. In *Proc. ICASSP*, 2014.

[22] Siddharth Sigtia, Emmanouil Benetos, and Simon Dixon. An end-to-end neural network for polyphonic music transcription. *arXiv preprint*, page 1508.01774, 2015.

[23] Dan Stowell and Mark D. Plumbley. Automatic large-scale classification of bird sounds is strongly improved by unsupervised feature learning. *PeerJ*, 2:e488, 2014.

[24] Karen Ullrich, Jan Schlüter, and Thomas Grill. Boundary detection in music structure analysis using convolutional neural networks. In *Proc. ISMIR*, 2014.

[25] Aaron van den Oord, Sander Dieleman, and Benjamin Schrauwen. Deep content-based music recommendation. In *Proc. NIPS*, 2013.