AUTOMATIC INSTRUMENT RECOGNITION IN POLYPHONIC MUSIC USING CONVOLUTIONAL NEURAL NETWORKS

*Peter Li, Jiyuan Qian, and Tian Wang**

New York University
Center for Data Science
New York, NY 10003

ABSTRACT

Traditional methods to tackle many music information retrieval tasks typically follow a two-step architecture: feature engineering followed by a simple learning algorithm. In these "shallow" architectures, feature engineering and learning are typically disjoint and unrelated. Additionally, feature engineering is difficult, and typically depends on extensive domain expertise.

In this paper, we present an application of convolutional neural networks for the task of automatic musical instrument identification. In this model, feature extraction and learning algorithms are trained together in an end-to-end fashion. We show that a convolutional neural network trained on raw audio can achieve performance surpassing traditional methods that rely on hand-crafted features.

Index Terms— convolutional neural networks, deep learning, end-to-end learning, music information retrieval, source identification,

1. INTRODUCTION

Computer audition is the general study of the systems and methods necessary for audio understanding by a machine. In a sense, computer audition concerns itself with the study of designing computers that can "hear" as humans do. The goal is a machine that can "organize what they hear; learn names for recognizable objects, actions, events, places, musical styles, instruments, and speakers; and retrieve sounds by reference to those names." [1]

In this paper, we focus on the first two tasks. Given a musical recording, how can we train a system to identify the instruments that are present? We present an application of deep learning for the task of automatic musical instrument identification in polyphonic music. We show that an end-to-end system using convolutional neural networks trained on raw audio can surpasses traditional MIR models trained using hand-crafted features.

*All authors contributed equally to this work

1.1. Relation to Previous Work

In a paper calling for the adoption of deep architectures in MIR,[2] describe traditional MIR methods as follows:

> The traditional approaches to these problems are rather homogeneous, adopting a two-stage architecture of feature extraction and semantic interpretation, e.g. classification, regression, clustering, similarity ranking, etc. Feature representations are predominantly hand-crafted,drawing upon significant domain-knowledge from music theory or psychoacoustics.

Since good features are hard to craft, much of the recent research in the MIR community has concerned itself with the semantic interpretation part of the problem i.e. training better models given a set of standard audio features (e.g. Mel-Frequency Cepstral Coefficients or chroma)[2].

In this paper, we depart from the traditional MIR approach. Using convolutional neural networks, we train a model using raw audio as input. We utilize a deep architecture where feature extraction *and* semantic interpretation can both be learned from data directly.

This approach to audio signal processing has been explored before in speech processing e.g., [3] and musical audio tagging [4]. However, to the best of our knowledge, this is the first application of deep learning to source identification.

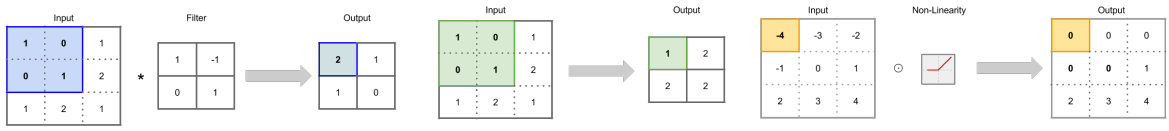
2. PROBLEM DEFINITION

Before going into the details of our work, we give a formal definition of our task:

Given a section of audio x, we would like to predict a vector $y \in \{0,1\}^l$ where l is the total number of instruments and

$$y_i = \begin{cases} 1, & \text{if instrument } i \text{ is in } x \\ 0, & \text{otherwise} \end{cases}$$

We treat this as a multi-label classification problem where



a) Convolution b) Pooling c) Non-Linearity

Fig. 1. Common ConvNet Layers

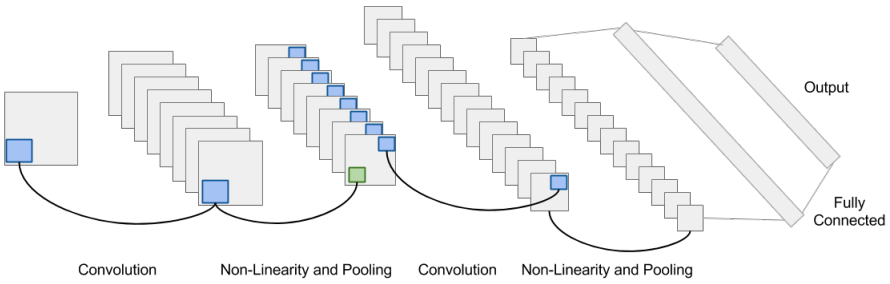


Fig. 2. Sample ConvNet Architecture

each label corresponds to an instrument. A model should output 1 if an instrument is present and 0 otherwise.

3. MODEL

Convolutional Neural Networks (CNN) [5] can be seen as a trainable feature extractor coupled with a learning model. Generally, they are know to be good at extracting high level features that represent abstract concepts from the original data. A typical model contains multiple layers of modules, where each module performs a simple data transformation. In Figure 1, we show a few common operations on a general matrix input. In a convolution layer, a filter, whose weights are learned, is convolved with its input by taking point-wise multiplication and then summing the results. Pooling is a down-sampling operation that combines nearby points. Non-linearities are applied point wise. In Figure 2, we show a sample model architecture. Each of the first two layers contains a convolution, pooling, and non-linearity operation. The final two layers are a fully connected neural network.

Hyper-parameters of the system include the number of layers and feature maps, convolution filter size, pooling size and which non-linear activation function to use. The choice of activation functions usually include the sigmoid function, hyperbolic tangent function and more recently Rectified Linear Unit (ReLU)[6].

Model Architecture

Our convolutional neural network contains three temporal convolutional layers (a convolution operation where the height of the filter is the height of the input) with ReLU and max pooling. These three layers are followed by two fully connected layers with ReLU and Dropout on the first layer and a sigmoid after the the second fully connected layer. This gives us a 11×1 vector $\hat{y}$ where each $\hat{y}_i \in [0, 1]$. These predicted activations are compared to training activations using a binary cross entropy loss function. The exact model Specifications are presented in Table 1.

Parameter	
Convolution Feature Maps 1	256
Convolution Filter Size 1	3101
Maxpooling Stride Size 1	20
Maxpooling Size 1	40
Convolution Feature Maps 2	384
Convolution Filter Size 2	300
Maxpooling Size 2	30
Maxpooling Stride Size 2	20
Convolution Feature Maps 3	384
Convolution Filter Size 3	20
Maxpooling Size 3	8
Maxpooling Stride Size 3	4
Layer 4 Output Size	400
Final Output Size	11

Table 1. ConvNet Specifications

4. EXPERIMENTS

4.1. Data and Setup

We trained and evaluated our model using data from MedleDB [7]. MedleyDB is a multitrack dataset of 122 annotated musical recordings. For each song we have three types of audio content: mix, stems, and raw audio. Mixed audio is comprised of a set of stems and stems are comprised of a mix of raw audio. Since MedleyDB was primarily created to support research on melody extraction, we had to create our own labels for model training. In the follow sections, we outline our procedure.

For each stem, we have annotations on instrument activation that represent the confidence of whether or not the instrument is active during that time frame. These annotations were generated using a standard envelope following technique on each stem, consisting of half-wave rectification, compression, smoothing, and down-sampling. For additional details see [7].

Dataset Split

To train our model, we sliced individual tracks into one second, non-overlapping clips. 80% of the clips were used for model training and the remaining 20% were reserved for testing. When splitting the data, there were two main considerations. First, we didn't want to have labels that were in the test set but not in the training set. Second, we didn't want to have clips belonging to the same track in both the test and training sets.

To address both issues at the same time, we use the algorithm in [8] to split the 122 mixed tracks into a training and test set based on the instruments that appear in each track. After the initial split, we cut each track into one second non-overlapping clips. This gives us our final training and test sets. Because tracks vary in length, after cutting into one second clips, the training and test set have 21177 and 4985 clips respectively.

Label Generation

To train our model, we need labels indicating whether or not an instrument appears in the entire audio clip. However, the activation confidence scores from MedleyDB provide local information i.e, activations at each point in time. To convert this to a global label for the entire clip, we take the maximum of the moving average of the activation confidences. An instrument is considered active in a clip as long as its average activation confidence exceeds a threshold over a particular window. For our experiment, we chose a windows length of 100ms and a threshold of 0.5. Figure 3 shows an example of this procedure.

The activation annotations cover 82 instrument. We grouped these instruments into 70 categories. To simplify model training and to have a more balanced dataset for learning, we combined all categories appearing in less than 20 songs into an 'OTHER' category. This gives us 11 classes for classification:

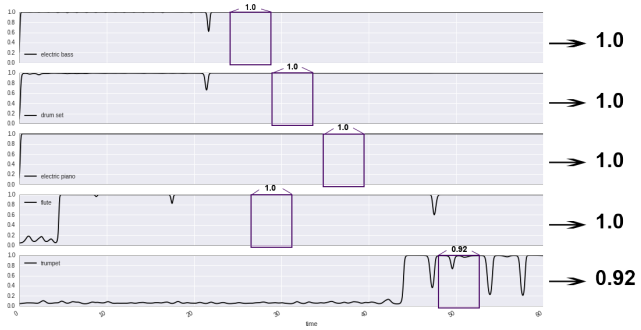


Fig. 3. Label Generation Procedure

- electric bass
- acoustic guitar
- synthesizer
- drum set
- fx/processed sound
- voice
- violin
- piano
- distorted electric guitar
- clean electric guitar
- OTHER

4.2. CNN Training

Our CNN was trained using stochastic gradient descent with a batch size of 16. To speed up training time, we transformed inputs using global contrast normalization. This is a standard preprocessing step and has been show to speed up training time. [9].

4.3. Benchmarks

For comparison, we also ran tests using more traditional MIR techniques. In the benchmarks, we use domain knowledge to construct music related features. For each audio clip, we first computed the Mel-frequency cepstral coefficients (MFCC) along with the first and second order differences, MFCC Δ and Δ^2. These three matricies where stacked and modelled using a Gaussian distribution [10].

$$\begin{pmatrix} MFCC \\ MFCC\Delta \\ MFCC\Delta^2 \end{pmatrix} = \begin{pmatrix} | & & | \\ m_1 & \dots & m_T \\ | & & | \end{pmatrix} ,$$

where $m_i \sim N(\mu, \Sigma)$

(μ, Σ) were used as features to train a random forest and logistic regression.

Models	Accuracy	Exact Match	Precision	Recall	F-micro	F -macro
Audio + CNN	**82.74%**	**25.78%**	0.7560	**0.6888**	**0.7208**	**0.6433**
MFCC + Random Forest	82.13%	17.53%	**0.7908**	0.5400	0.6418	0.4471
MFCC + Logistic Regression	81.80%	18.17%	0.7457	0.5857	0.6561	0.4840
Predict Majority Class	70.37%	9.95%	0.5001	0.4602	0.4793	0.1801

Table 2. Experiment Results

4.4. Experiment Results

The results of our experiments are shown in Table 2. In addition to the random forest and logistic regression, we also provide a naive benchmark that always predicts the three most common instruments in the training set. As seen, the Audio + CNN model generally outperforms the baseline models.

5. DISCUSSION

Convolutional neural networks have recently show remarkable results in a number of tasks. However, it is often times difficult to intuitively understand what they are doing and why they work. In this section, we make an attempt at examining the weights learned in the first convolutional layer. This sections follows the procedures of previous works in training CNN's on audio waveforms and confirms previous findings.

Examining the filter weights learned in the first layer, we see that the model does seem to learn a set of frequency selective filters. In Figure 4, we plot the magnitude spectra of each filter, sorted by dominant frequency. Similar to results in [4] and [3], the first convolutional layer seems to learn an auditory scale filter bank.

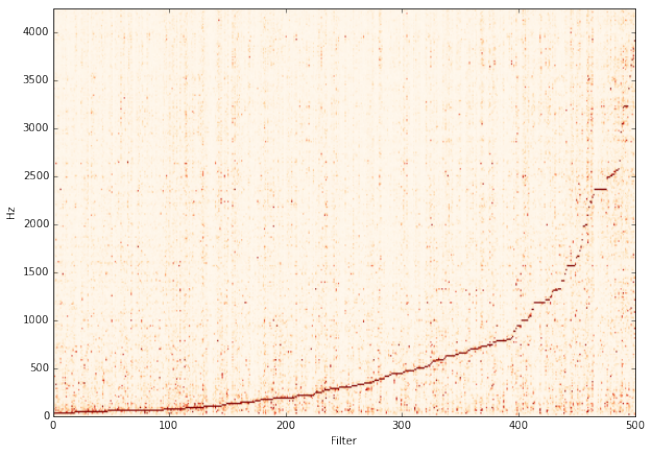


Fig. 4. Rescaled magnitude spectra sorted by dominant frequency. The spectra of each filter was rescaled to [0,1] by subtracting the minimum and dividing by the range.

In Figure 5, we show a sample of filters learned in the first layer. As in [4] many of the learned filters are translated versions of each other. This is not necessary surprising since phase invariance is most likely difficult learn given our architecture.

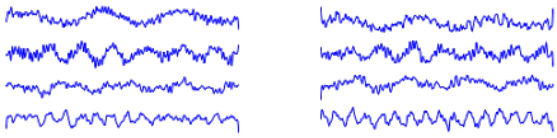


Fig. 5. Sample of filters learned in the first layer.Filters were low pass filtered to remove noise.

6. CONCLUSION AND FUTURE WORK

We present a convolutional neural network for instrument identification. We show that an end-to-end deep learning system can be trained to achieve performance in line with (and sometimes exceeding) traditional methods that rely on extensive domain knowledge.

In future work, we will investigate methods to further understand the transformations made by the CNN. Additionally, we would like to explore different architecture that may be able to learn phase and translation invariance.

Code

Code for this project can be found on
https://github.com/glennq/instrument-recognition.

7. REFERENCES

[1] Richard F Lyon, "Machine hearing: An emerging field [exploratory dsp]," *Signal Processing Magazine, IEEE*, vol. 27, no. 5, pp. 131–139, 2010.

[2] Eric J Humphrey, Juan Pablo Bello, and Yann LeCun, "Moving beyond feature design: Deep architectures and automatic feature learning in music informatics.," in *ISMIR*. Citeseer, 2012, pp. 403–408.

[3] Y. Hoshen, R. J. Weiss, and K. W. Wilson, "Speech Acoustic Modeling from Raw Multichannel Waveforms," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Brisbane, Australia, Apr. 2015.

[4] Sander Dieleman and Benjamin Schrauwen, "End-to-end learning for music audio," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 6964–6968.

[5] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[6] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[7] Rachel Bittner, Justin Salamon, Mike Tierney, Matthias Mauch, Chris Cannam, and Juan Bello, "Medleydb: a multitrack dataset for annotation-intensive mir research," in *15th International Society for Music Information Retrieval Conference (ISMIR 2014)*, 2014, pp. 155–160.

[8] Konstantinos Sechidis, Grigorios Tsoumakas, and Ioannis Vlahavas, "On the stratification of multi-label data," in *Machine Learning and Knowledge Discovery in Databases*, pp. 145–158. Springer, 2011.

[9] Yann A LeCun, Léon Bottou, Genevieve B Orr, and Klaus-Robert Müller, "Efficient backprop," in *Neural networks: Tricks of the trade*, pp. 9–48. Springer, 2012.

[10] Beth Logan et al., "Mel frequency cepstral coefficients for music modeling.," in *ISMIR*, 2000.