END-TO-END LEARNING FOR MUSIC AUDIO

Sander Dieleman, Benjamin Schrauwen

Ghent University
Electronics and information systems department
Sint-Pietersnieuwstraat 41, Gent, Belgium

ABSTRACT

Content-based music information retrieval tasks have traditionally been solved using engineered features and shallow processing architectures. In recent years, there has been increasing interest in using feature learning and deep architectures instead, thus reducing the required engineering effort and the need for prior knowledge. However, this new approach typically still relies on mid-level representations of music audio, e.g. spectrograms, instead of raw audio signals. In this paper, we investigate whether it is possible to apply feature learning directly to raw audio signals. We train convolutional neural networks using both approaches and compare their performance on an automatic tagging task. Although they do not outperform a spectrogram-based approach, the networks are able to autonomously discover frequency decompositions from raw audio, as well as phase- and translation-invariant feature representations.

Index Terms— feature learning, end-to-end learning, convolutional neural networks, music information retrieval, automatic tagging

1. INTRODUCTION

In music information retrieval (MIR), researchers have traditionally relied on a two-stage approach to solve content-based MIR tasks: features are extracted from music audio signals, and are then used as input to a regressor or classifier, such as support vector machines or logistic regression. The features are designed to uncover information in the input that is salient for the task at hand. This requires considerable expertise about the problem and constitutes a significant engineering effort.

In recent years, feature learning has become increasingly popular. In many domains, it has allowed researchers to build models of data requiring only a minimum of prior knowledge. In computer vision and speech recognition, modern feature learning techniques have become the state of the art [1, 2]. Lately, feature learning has also been receiving more attention from the MIR community [3]. Most research on the application of feature learning to MIR problems relies on using mid-level representations of audio, such as spectrograms. This is also a form of prior knowledge. In computer vision, on the other hand, modern feature learning techniques are able to operate directly on raw pixel representations of images, without requiring any form of preprocessing [4].

In this paper, we investigate whether it is possible to apply feature learning directly to raw audio signals, thus further reducing the amount of prior knowledge required. We use convolutional neural networks to solve an automatic tagging task and compare different input representations and network architectures. In Section 2, we describe the mid-level representations of audio that are frequently used in MIR. In Section 3, we discuss the end-to-end learning paradigm. Our experiments and results are described in Section 4, and we conclude in Section 5.

2. MID-LEVEL REPRESENTATIONS

Many higher-level characteristics of sound relate to the energies in different frequency bands. This explains the utility of time-frequency representations of audio such as spectrograms, which are frequently used in literature [5, 6, 7, 8]. Another advantage in the context of feature learning is that they convert the audio data into a representation that is very similar to an image. This makes it easier to translate feature learning approaches, which are often designed with image data in mind, to an audio context.

Most researchers do not use raw spectrograms, which have a linear frequency scale. The mel scale or a logarithmic frequency scale are preferred instead: they reduce the resolution for higher frequencies, which matches the human perception of frequency, and reduce the size of the representation. Another common preprocessing step is to apply some form of dynamic range compression, for example by taking the logarithm of the spectrograms.

Although there has been some work about learning features directly from speech signal fragments [9, 10, 11], to our knowledge feature learning has not been applied directly to raw music audio signals in literature.

978-1-4799-2893-4/14/$31.00 ©2014 IEEE

3. END-TO-END LEARNING

The term *end-to-end learning* is used to refer to processing architectures where the entire stack, connecting the input to the desired output, is learned from data [12]. An end-to-end learning approach greatly reduces the need for prior knowledge about the problem, and minimises the required engineering effort; only the tuning of the model hyperparameters requires some expertise, but even that process can be automated [13]. Learning features can result in better performance than engineering them, because they are automatically tailored to the task at hand. Furthermore, training the entire processing architecture can lead to new insights about what kind of information is salient for a given task [3, 6].

Convolutional neural networks (CNNs) [14] in particular lend themselves well to this setting, because they consist of many layers of processing that are all learned using the same objective function, which is propagated through the network. CNNs have been used for image classification [2, 15], speech recognition [5], epileptic seizure detection [16], and many other applications. In music information retrieval, they have been used for onset detection [17], genre classification [5, 18, 19], artist recognition [5, 19], instrument classification [20] and content-based recommendation [21].

4. EXPERIMENTS AND RESULTS

To compare end-to-end learning with the traditional MIR approach of using a mid-level representation of the audio signals, we trained deep CNNs to perform automatic tagging on the Magnatagatune dataset [22]. The dataset contains 25863 29-second audio clips with a sample rate of 16 kHz, taken from songs by 230 artists, annotated with 188 tags. It comes in 16 parts, of which we used the first 12 for training, the 13th for validation and the remaining 3 for testing. We only used the 50 most frequent tags.

4.1. Experimental setup

The CNN architecture that we used as a basis for all our experiments is visualised in Figure 1. It consists of 6 layers in total: two convolutional layers with 32 filters of length 8, alternating with max-pooling layers with pooling size 4, and two dense layers with 100 and 50 units respectively. We used rectified linear units [23] in all layers except for the top layer, where we used sigmoidal units. We extended this architecture for each the experiments described in the following subsections. Note that all convolutions and pooling operations are one-dimensional, i.e. only along the axis representing time. Although we could convolve and pool in the frequency direction in the case of spectrogram input [24], we did not investigate this approach here to ensure a fair comparison, as it is not possible with raw audio input.

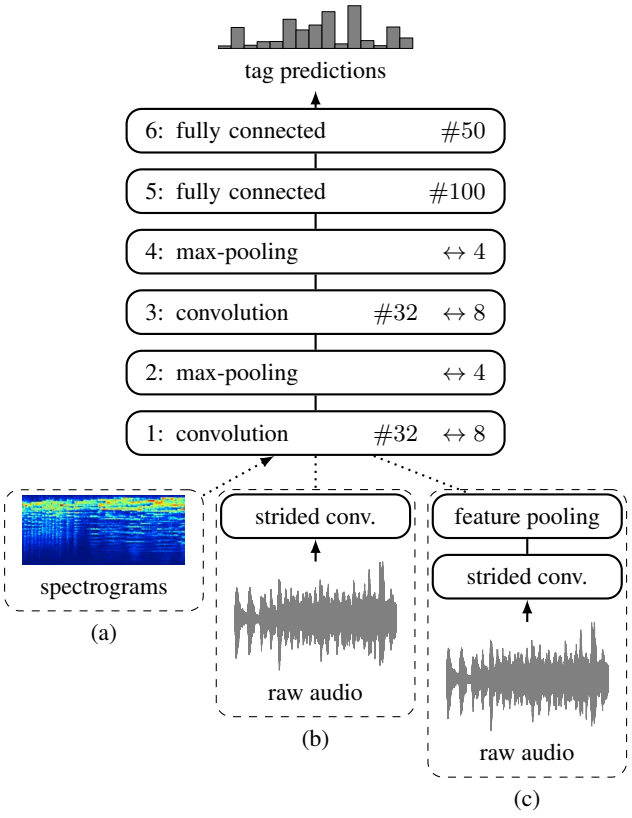

Fig. 1. The convolutional neural network architecture we used for our experiments. The filter sizes and pooling sizes (↔) and numbers of units (#) are indicated. We consider three possible approaches: (a) spectrograms as input, (b) raw audio as input by adding an additional strided convolutional layer, and (c) raw audio with feature pooling.

We trained the network using minibatch gradient descent, with minibatches of 10 examples. We used windows of about 3 seconds of audio as input. To compute tag predictions for a clip, we averaged the predictions over consecutive windows. We evaluated the use of dropout regularization [25] in the fully connected layers, but this did not affect performance significantly. To evaluate the predictions, we computed the area under the ROC curve (AUC) for each tag and computed the average across all 50 tags. For each experiment, we performed roughly 5 million parameter updates and validated the model at regular intervals. We report results on the test set for the parameters that achieved the best validation score. We used the Theano library to enable GPU acceleration [26].

4.2. Spectrograms versus raw audio

To assess whether the task of tag prediction can be solved with a CNN using only raw audio, we compared two approaches:

- **spectrograms**: we extracted mel-spectrograms with 128 components, and performed dynamic range compression

length	stride	**AUC** (spectrograms)	**AUC** (raw audio)
1024	1024	0.8690	0.8366
1024	512	0.8726	0.8365
512	512	0.8793	0.8386
512	256	0.8793	0.8408
256	256	0.8815	0.8487

Table 1. Results for the tag prediction task on Magnatagatune, with convolutional neural networks using spectrograms and raw audio, for different combinations of filter lengths and strides. We report AUCs on the test set. 1024 samples correspond to 64 ms at a sample rate of 16 kHz.

by applying the elementwise function $f(x) = \log(1 + C \cdot x)$, where C is a constant controlling the amount of compression [27], which we set to $10,000$. This representation was used as input to the network, as shown in Figure 1a. We tried a number of different window lengths and strides (hop sizes), which are listed in Table 1.

- **raw audio**: we extended the basic architecture described in the previous subsection by adding an additional convolutional layer at the input side, and used raw audio as input, as shown in Figure 1b. The additional layer performs a *strided convolution*, i.e. a convolution with a stride larger than one, because it would be computationally infeasible to compute full convolutions on raw audio signals. For this layer, we tried the same lengths and strides as for the spectrogram extraction. The raw audio signals were not preprocessed in any way.

The results for these experiments are listed in Table 1. The spectrogram approach consistently outperforms using raw audio, and a smaller window length consistently improves results. Presumably the increased time granularity is useful for this task. Reducing the stride to half the window length does not result in any significant improvement.

To see whether the network using raw audio as input is able to learn frequency-selective filters in the lowest layer, we computed their squared magnitude spectra. We have visualised the spectra of the filters ordered according to the dominant frequency (low to high) in Figure 2. From this visualisation, it is clear that most features are indeed frequency-selective, and they cover the lower half of the spectrum. This is to be expected, as the harmonic content of music tends to be mostly in the lower end of the spectrum. It is especially worth noting that the filters seem to linearly span the frequency range up to about 1000 Hz, whereas filters that are selective for higher frequencies are more spread out. This is reminiscent of the mel scale. Some of the learned filters are shown in Figure 3, ordered by dominant frequency. They are quite noisy, but the dominant frequency is visible for most.

Despite the fact that the training procedure is able to discover a frequency decomposition from raw audio examples, there is a significant performance gap between both approaches. This may be because the network architecture for raw audio input is not sufficiently expressive to perform an operation similar to spectrogram extraction. In the following experiments, we have attempted to incorporate some aspects of spectrograms into the network architecture, to assess whether they are important to achieve good performance.

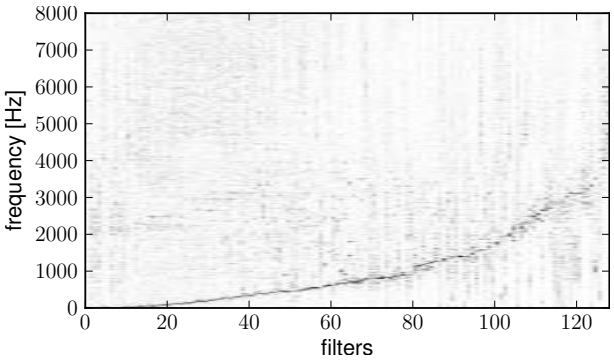

Fig. 2. Normalised magnitude spectra of the filters learned in the lowest layer of a convolutional neural network that processes raw audio signals, ordered according to the dominant frequency (from low to high). Each vertical line in the graph represents the frequency spectrum of a different filter. The filters have a length and stride of 512 samples.

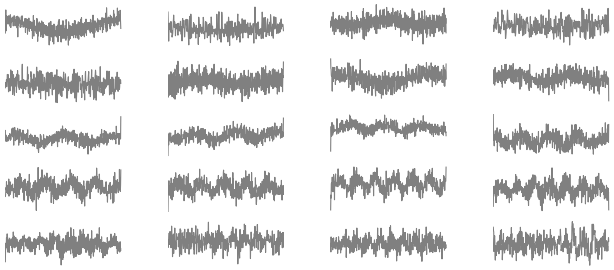

Fig. 3. A subset of filters learned in the lowest layer of a convolutional neural network that processes raw audio signals, ordered by dominant frequency. The filters have a length and stride of 512 samples.

4.3. Dynamic range compression

The mel-spectrograms were compressed using the nonlinear function described in Section 4.2. This type of nonlinearity is probably difficult to learn using a network consisting of rectified linear units. We replaced the rectification nonlinearity in the strided convolutional layer with a compression function. The results are shown in Table 2. Unfortunately, this does not have the desired effect, and in fact it severely degrades performance. The logarithmic nonlinearity seems to impede the optimization process.

nonlinearity	$f(x)$	AUC
rectified linear	$\max(0, x)$	0.8366
logarithmic	$\log(1 + C \cdot x^2)$	0.7508
logarithmic	$\log(1 + C \cdot \|x\|)$	0.7487

Table 2. Results for the tag prediction task on Magnatagatune, with convolutional neural networks using raw audio as input, for different types of nonlinearities used in the strided convolutional layer. We report AUCs on the test set. A filter length and stride of 1024 were used for all experiments.

pooling	pool size	AUC
no pooling	1	0.8366
L2 pooling	2	0.8387
L2 pooling	4	0.8387
max-pooling	2	0.8183
max-pooling	4	0.8280

Table 3. Results for the tag prediction task on Magnatagatune, with convolutional neural networks using raw audio as input, for different types of feature pooling after the strided convolutional layer. We report AUCs on the test set. A filter length and stride of 1024 were used for all experiments.

4.4. Invariance

Spectrograms exhibit various types of invariance, which are likely to be useful for the task of tag prediction: they are phase-invariant, and translation-invariant to a limited extent as well. Automatically discovering these invariances from data may be quite challenging. To facilitate this process, we can further modify the network architecture to pool across groups of filters, as shown in Figure 1c. Hyvärinen and Hoyer [28] showed that summing the squares of the activations of a set of linear filters (L2 pooling) allows for learning phase- and translation-invariant features. More recently, units computing the maximal activation across a set of linear filters (*maxout* units) have been used to achieve state of the art image classification performance on several benchmark datasets [29]. We have evaluated both approaches.

The results are shown in Table 3. Although L2 pooling does not seem to perform significantly better than no pooling, and max-pooling performs worse, it should be noted that combining linear filters by pooling reduces the effective number of features computed in the strided convolutional layer. This in turn reduces the number of parameters in the next convolutional layer, leading to a network with fewer trainable parameters. Some filters learned by the network with L2 pooling and a pool size of 4 are shown in Figure 4. As expected, most of the pools consist of filters that are translated or phase-shifted versions of each other.

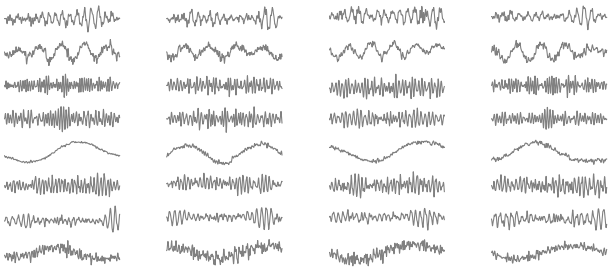

Fig. 4. A subset of filters learned in a convolutional neural network with a feature pooling layer (L2 pooling with pools of 4 filters). The filters have a length and stride of 1024 samples. Each row represents a filter group. The filters were low-pass filtered to remove noise and make the dominant frequencies stand out.

5. CONCLUSION AND FUTURE WORK

In this paper, we have investigated whether end-to-end learning for music audio is feasible using convolutional neural networks to solve an automatic tagging task. Although the performance level of a spectrogram-based approach was not reached, we have shown that the networks are able to learn useful features from raw audio: they are able to autonomously discover frequency decompositions, and when a feature pooling layer is incorporated, they discover phase- and translation-invariant features as well.

In future work, we will investigate the use of larger networks, in an effort to attain the performance of the spectrogram approach using only raw audio signals. We will also investigate the effect of different initializations of the strided convolutional layer. For example, we can initialize the weights such that it performs an operation that mimics spectrogram extraction, and then finetune these weights with gradient descent. Finally, we would like to investigate the feasibility of end-to-end learning for music audio using unsupervised feature learning techniques, instead of purely supervised learning.

6. REFERENCES

[1] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al., "Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups," *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 82–97, 2012.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, 2012.

[3] Eric J. Humphrey, Juan P. Bello, and Yann LeCun, "Moving beyond feature design: Deep architectures and automatic feature learning in music informatics," in *Proceedings of the 13th*

International Conference on Music Information Retrieval (ISMIR), 2012.

[4] Quoc V. Le, Marc'Aurelio Ranzato, Rajat Monga, Matthieu Devin, Kai Chen, Greg S. Corrado, Jeff Dean, and Andrew Y. Ng, "Building high-level features using large scale unsupervised learning," in *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, 2012.

[5] Honglak Lee, Peter Pham, Yan Largman, and Andrew Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks," in *Advances in Neural Information Processing Systems 22*. 2009.

[6] Mikael Henaff, Kevin Jarrett, Koray Kavukcuoglu, and Yann LeCun, "Unsupervised learning of sparse features for scalable audio classification," in *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR)*, 2011.

[7] J. Wülfing and M. Riedmiller, "Unsupervised learning of local features for music classification," in *Proceedings of the 13th International Society for Music Information Retrieval Conference (ISMIR)*, 2012.

[8] Sander Dieleman and Benjamin Schrauwen, "Multiscale approaches to music audio feature learning," in *Proceedings of the 14th International Conference on Music Information Retrieval (ISMIR)*, 2013.

[9] Jong-Hwan Lee, Ho-Young Jung, Te-Won Lee, and Soo-Young Lee, "Speech feature extraction using independent component analysis," in *Acoustics, Speech, and Signal Processing, 2000. ICASSP'00. Proceedings. 2000 IEEE International Conference on*. IEEE, 2000, vol. 3, pp. 1631–1634.

[10] Michael S Lewicki, "Efficient coding of natural sounds," *Nature neuroscience*, vol. 5, no. 4, pp. 356–363, 2002.

[11] Navdeep Jaitly and Geoffrey E. Hinton, "Learning a better representation of speech soundwaves using restricted boltzmann machines," in *Proceedings of ICASSP 2011*, 2011, pp. 5884–5887.

[12] Urs Muller, Jan Ben, Eric Cosatto, Beat Flepp, and Yann L Cun, "Off-road obstacle avoidance through end-to-end learning," in *Advances in neural information processing systems*, 2005, pp. 739–746.

[13] J Bergstra, D Yamins, and DD Cox, "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures," in *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, 2013, vol. 28, pp. 115–123.

[14] 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.

[15] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proceedings of the 26th Annual International Conference on Machine Learning*, New York, NY, USA, 2009, ICML '09, pp. 609–616, ACM.

[16] P. W. Mirowski, Y. LeCun, D. Madhavan, and R. Kuzniecky, "Comparing svm and convolutional networks for epileptic seizure prediction from intracranial eeg," *Machine Learning for Signal Processing, 2008. MLSP 2008. IEEE Workshop on*, pp. 244–249, 2008.

[17] Alexandre Lacoste and Douglas Eck, "A supervised classification algorithm for note onset detection," in *EURASIP Journal on Applied Signal Processing*, 2007.

[18] TL Li, Antoni B Chan, and AH Chun, "Automatic musical pattern feature extraction using convolutional neural network," in *Proc. Int. Conf. Data Mining and Applications*, 2010.

[19] Sander Dieleman, Philémon Brakel, and Benjamin Schrauwen, "Audio-based music classification with a pretrained convolutional network," in *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR)*, 2011.

[20] Eric J. Humphrey, Aron P. Glennon, and Juan Pablo Bello, "Non-linear semantic embedding for organizing large instrument sample libraries.," in *ICMLA (2)*, Xue wen Chen, Tharam S. Dillon, Hisao Ishbuchi, Jian Pei, Haixun Wang, and M. Arif Wani, Eds. 2011, pp. 142–147, IEEE Computer Society.

[21] Aäron van den Oord, Sander Dieleman, and Benjamin Schrauwen, "Deep content-based music recommendation," in *Advances in Neural Information Processing Systems 26*, 2013.

[22] Edith Law and Luis von Ahn, "Input-agreement: a new mechanism for collecting data using human computation games," in *Proceedings of the 27th international conference on Human factors in computing systems*, 2009.

[23] Vinod Nair and Geoffrey E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, 2010.

[24] Ossama Abdel-Hamid, Abdel-rahman Mohamed, Hui Jiang, and Gerald Penn, "Applying convolutional neural networks concepts to hybrid nn-hmm model for speech recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, 2012, pp. 4277–4280.

[25] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," Tech. Rep., University of Toronto, 2012.

[26] James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio, "Theano: a CPU and GPU math expression compiler," in *Proceedings of the Python for Scientific Computing Conference (SciPy)*, June 2010.

[27] M. Muller, D.P.W. Ellis, A. Klapuri, and G. Richard, "Signal processing for music analysis," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 5, no. 6, pp. 1088–1110, 2011.

[28] Aapo Hyvärinen and Patrik Hoyer, "Emergence of phase- and shift-invariant features by decomposition of natural images into independent feature subspaces," *Neural Comput.*, vol. 12, no. 7, pp. 1705–1720, July 2000.

[29] Ian J Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio, "Maxout networks," *arXiv preprint arXiv:1302.4389*, 2013.