Timbre spaces with sparse autoencoders

Pablo E. Riera[1] **, Manuel C. Eguía**[1] **, Matías Zabaljáuregui**[2]

[1]Laboratorio de acústica y percepción sonora
Escuela Universitaria de Artes, Universidad Nacional de Quilmes
Roque Sáenz Peña 352, Bernal Buenos Aires, Argentina

[2]Laboratorio de Inteligencia Artificial Aplicada
Departamento de Computación, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires
Pabellón I, Ciudad Universitaria, Ciudad de Buenos Aires, Argentina.

pablo.riera@gmail.com

Abstract

Timbre perception studies emphasize the multidimensional nature of timbre and many rely on dimensionality reduction techniques to visualize perceptual similarity evaluations or sound descriptors that encompass timbre perception. In this work, we explore the uses of sparse autoencoders to perform unsupervised learning and nonlinear dimensionality reduction to extract a spectral code representation that is used for timbre analysis and visualization. Using only one music fragment in the autoencoder learning process generates an overfitted reconstruction but gives a low dimensional neuronal activity pattern which encodes all the sound spectrum information and could be used for synthesis as a neuronal music score.

1. Introduction

Timbre is widely recognized as a highly complex and multidimensional percept that cannot be (or hardly can be) accounted in terms of a few quantitative descriptors [1][2]. The process of timbre perception is closely related to the tasks of sound identification and classification and is concomitant to the hierarchical organization of sensory systems [3]. A common approach to tackle this issue in computational studies consists of calculating some sound descriptors relevant to timbre perception, such as Mel-frequency Cepstral Coefficients (MFCC) [4] or Spectral Contrast [5] and then applying dimensionality reduction techniques for visualization, as multidimensional scaling [6], isomaps [7] and self-organized maps [8], among others.

On the other hand, recent advances in deep neural networks, in which several layers of nodes are used to build up progressively more abstract representations of the inputs, have contributed to develop a new approach to this problem, yielded promising results in music related tasks such as instrument classification [9][10], timbre analysis [11], genre classification [12][13], among others, [14][15] and sample based sound synthesis [16]. Several architectures and learning procedures have proven successful at processing audio data, in particular, we can mention recent uses of autoencoders in audio synthesis [17][18], statistical parametric speech synthesis [19], adaptive reduced-dimensionality equalization [20], and denoising and dereverberation [21]. As an alternative to autoencoders, linear methods like sparse coding and nonnegative matrix factorization have also been used for sound classification [22] and source separation [23].

Autoencoders are a type of neural network having a coding stage and a decoding stage, in such a way that a bottleneck is generated in the middle layer (see Figure 1). Usually, the learning strategy involves minimizing the difference between the input and the coded-decode output. This learning procedure could be used fo nonlinear dimensionality reduction [24] or nonlinear PCA [25][26].

In this work, we explore the use of sparse autoencoders with spectral inputs for efficiently learning timbre representation and synthesis, in a similar fashion to [18]. In contrast to this work, we perform a complete unsupervised learning with one musical or audio fragment at a time. This generates a network that overfits the audio

fragment used, and focus on nonlinear dimensionality reduction. In addition, a sparse penalty is included in the cost function in order to favor a sparser representation in the code layer. When using the system for sound synthesis, the spectral representation of the input needs to be invertible and bear an appropriate method for phase reconstruction [27].

In the next section, we describe the autoencoder structure and the learning methods. In the results section, we compare timbre spaces generated with the autoencoder code layer to those generated with MFCC and Spectral Contrast as descriptors.

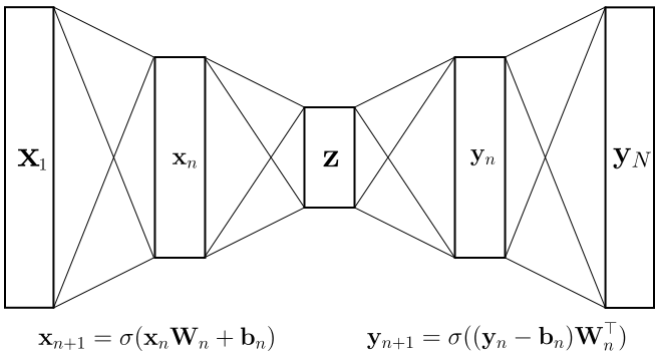

Figure 1: Autoencoder layout. X_n represents the activities on the coding stage Z represents the code layer activity, and Y_n the activities on the decoding stage. A clear bottleneck is visualized.

2. Methods

The log-magnitude power spectrogram of a monaural audio fragment was used as input, after resampling it to a lower sampling rate (in order to make the computations lighter) and computing the short-time Fourier transform (STFT) with fixed window and step length. Every STFT window was considered as a data sample (number of observations), thus the number of frequency bins was equal to the number of input neurons (number of features).

The autoencoder was built with tied weights in the coding and decoding stages. The numbers of layers and neurons per layer were explored manually before adopting a simplified structure in which the number of neurons of each layer was reduced by a factor of two until a minimum number is obtained in the code layer, and for the decoding stage, the numbers are increased by the same factor. This architecture allowed a reasonable learning performance. In concrete, the number of neurons per layer finally adopted was $2^k \;\; k = 10, 9, 7, 5, 3, 5, 7, 9, 10$, hence the code layer has only 8 neurons.

Several activation functions were tested. The results shown here were obtained using a softplus function and min max normalization. Similar results were obtained with tanh function and z-score normalization.

The cost function consisted of three terms:

$$E = \frac{1}{2N_0}\sum_{n=1}^{N_0}(X_n - Y_n)^2$$

$$-\frac{\alpha}{N_z}\sum_{n=1}^{N_z}\hat{Z}_n \log(\hat{Z}_n{}^2)$$

$$+\lambda\frac{1}{2K}\sum_{k=1}^{K}\left(\|W_k\|_2^2 + \|b_k\|_2^2\right) \qquad (1)$$

The first term corresponds to the averaged squared difference between the input X and the output Y of the autoencoder. N_0 denotes the size of input and output layers. The second term corresponds to a sparse regularization component, provided by the entropy of the normalized activity of the code layer $\sum(\hat{Z}) = 1$. N_z stands for the code layer size. Finally, the third term includes a $L2$ regularization on the weights and bias. K corresponds to the number of weight matrices.

Before training the complete autoencoder, a pretraining [24] stage was performed on single hidden layer autoencoders, where the input of a single layer of the autoencoder was the code layer activity of the previous one. The network was optimized with Adam method [28].

After the training, the analysis consisted on inspecting the activity of the code layer. This activity plus the weights and bias contain all the information for reconstructing the original audio signal. In a general sense, this neuronal activity can be put into correspondence to a *timbre score*

Parameters	Values
Sampling frequency	22050
Window size	1024 (46 ms)
Step size	256 (11 ms)
Number of spectrum samples	5168
Layers dimensions	2^k $k = 10, 9, 7, 5$ $3, 5, 7, 9, 10$
Activation function	softplus
α	0.8
λ	0.005
Learning rate	0.005
Batch size	200

Table 1: Hyper-parameters of the sparse autoencoder

of the musical fragment used as input, or a trajectory on the timbre space of the audio signal.

For visualization of the timbre space, reduction of the dimensionality of the activations of the code layer was done by principal components analysis (PCA). Finally, for the sonification of the reconstructed spectrogram, we used the original phase information. This was done with the aim of preserving the audio quality, but phase reconstruction algorithms could be used as well.

3. Results

The results displayed here are obtained using the first 7 cycles of Grisey's *Partiels* [29] as input. We have chosen this example because, in one hand, this piece has an outstanding historic relevance in the development of timbre in 20th-century music, and on the other hand, it behaves as a good data set, mainly due to the fact that is basically composed by repetitions of similar sounds with variations.

The hyper-parameters of the autoencoder used for this results are shown in table 1.

In figure 2, we show the spectrogram of the original audio signal along with the reconstruction obtained as output. Despite being a highly detailed reconstruction, the output displays fewer variations than the original, due to the fact that the autoencoder has some denoising properties.

In figure 3 we display the neural activity pattern of the code layer. The sparse regularization cost forces a sparse activity pattern, and some neurons ended with null activation. Nonetheless, different runs of the optimization could end with different activation patterns (and same overall cost), due to random initial weights and bias.

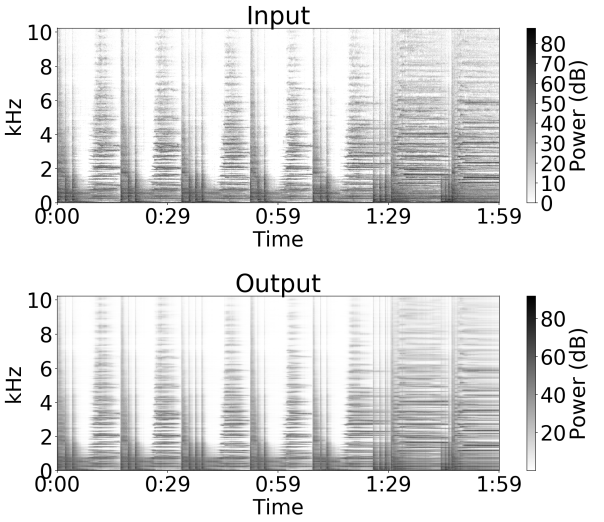

Figure 2: Top: Input data spectrogram of a fragment of Grisey's *Partiels*. Bottom: the reconstructed spectrogram by the autoencoder.

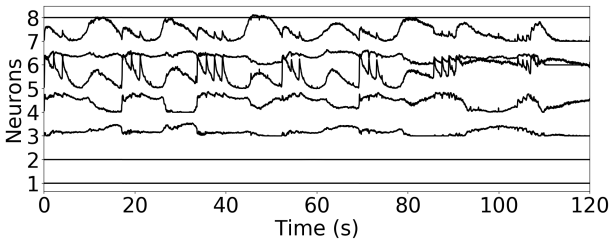

Figure 3: Code layer activities. As the reconstruction is almost identical to the original sound, we could interpret this activities as the music neural score.

Figure 4, shows on the top row, the timbre space generated by three principal components of standard timbre descriptors (in this case a combination of 20 MFCC and 7 Spectral Contrast coefficients). The color shades were obtained by k-means clustering on those descriptors. Seven clusters were used to guide the visual inspection of both timbre spaces and to match the different regions. In the middle row are the

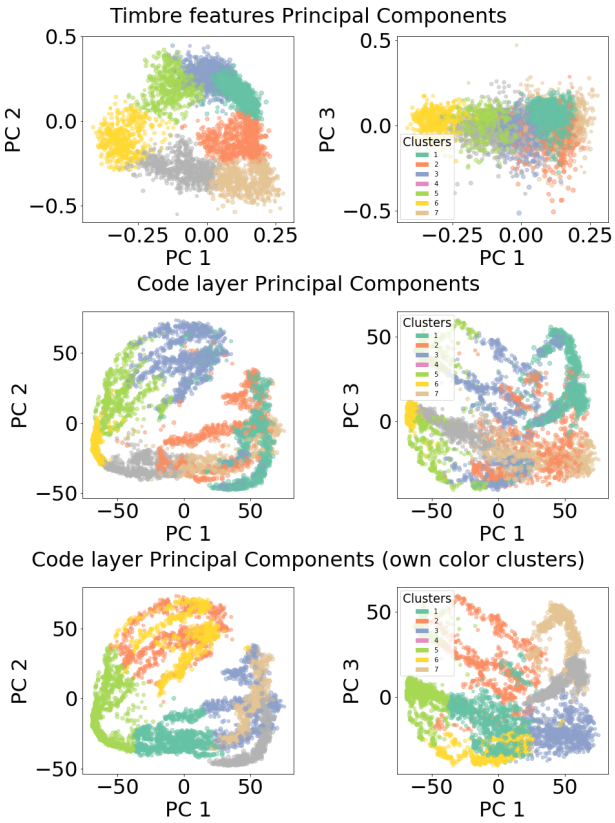

Figure 4: Top and middle: Timbre space generated by the first three principal components of a set of standard timbre descriptors (MFCC, Spectral Contrast) and by the code layer activities. Bottom: Same data as in the middle plot, but with different clustering colors (see text for a full description).

PCA components for the code layer activity pattern with the same color clustering. The bottom row uses its own clusters instead. It could be observed that some clusters remain close in both top and middle row timbre spaces, but there are differences with those clusters generated by the code layer in the bottom plot, more structure arise. This is reflected also in the difference between the third PCA component from the timbre features and the third component of the code layer. The standard timbre descriptors elicit a more compacted space than from the autoencoder which has a torn structure.

In the first 7 cycles of *Partiels*, an alternating pattern is produced between the trombone and contrabass low notes and the strings and woods high notes that generate a cyclic trajectory in the timbre space which is present in both timbre spaces (timbre features and code layer).

4. Conclusions

We presented a computational timbre analysis method involving a sparse autoencoder. When using this type of neural networks, the code layer is able to learn a meaningful representation that is capable of reconstructing the original input data, the sound spectrum in the present work. The activity of the code layer is used for timbre analysis and it behaves like an audio specific, a small set of sound descriptors.

The results presented here focused on learning from a single sound or music fragment, but the same approach could be expanded to learning from a corpus of music fragments. Different network architectures may be necessary and also a different learning paradigm like classification [18].

Regarding the synthesis capabilities of the system, when trained with a music fragment, we consider the code layer activities as a neural score that interprets the music. This could be useful for timbre-oriented audio processing and source separation.

Generated timbre spaces by the autoencoder shown more structure than those generated by standard timbre descriptors. For a complete analysis of the timbre space, a perceptual measurement or an exhaustive listening task is needed by comparing the clusters and their relative locations.

5. References

References

[1] Stephen McAdams. Musical timbre perception. *The psychology of music*, pages 35–67, 2013.

[2] Carol L Krumhansl. Why is musical timbre so hard to understand ? In *Structure and perception of electroacoustic sound and music*, volume 9, pages 43–53. 1989.

[3] Kailash Patil, Daniel Pressnitzer, Shihab Shamma, and Mounya Elhilali. Music in our ears: the biological bases of musical timbre perception. *PLoS Comput Biol*, 8(11):e1002759, 2012.

[4] Beth Logan. Mel Frequency Cepstral Coefficients for Music Modeling. *International Symposium on Music Information Retrieval*, 28:11p., 2000.

[5] Dan-Ning Jiang, Lie Lu, Hong-Jiang Zhang, Jian-Hua Tao, and Lian-Hong Cai. Music type classification by spectral contrast feature. In *Multimedia and Expo, 2002. ICME'02. Proceedings. 2002 IEEE International Conference on*, volume 1, pages 113–116. IEEE, 2002.

[6] John M. Grey. Multidimensional perceptual scaling of musical timbres. *The Journal of the Acoustical Society of America*, 61(5):1270–1277, 1977.

[7] John Ashley Burgoyne and Stephen McAdams. Non-linear scaling techniques for uncovering the perceptual dimensions of timbre. In *ICMC*, 2007.

[8] MA Loureiro, HB de Paula, and HC Yehia. Timbre Classification Of A Single Musical Instrument. *ISMIR*, 2004.

[9] Yoonchang Han, Jaehun Kim, Kyogu Lee, Yoonchang Han, Jaehun Kim, and Kyogu Lee. Deep convolutional neural networks for predominant instrument recognition in polyphonic music. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 25(1):208–221, 2017.

[10] Philippe Hamel, Sean Wood, and Douglas Eck. Automatic Identification of Instrument Classes in Polyphonic and Poly-Instrument Audio. *Ismir*, pages 399–404, 2009.

[11] Jordi Pons, Olga Slizovskaia, Rong Gong, Emilia Gómez, and Xavier Serra. Timbre Analysis of Music Audio Signals with Convolutional Neural Networks. 2017.

[12] Philippe Hamel and Douglas Eck. Learning Features from Music Audio with Deep Belief Networks. *International Society for Music Information Retrieval Conference (ISMIR)*, (Ismir):339–344, 2010.

[13] Honglak Lee, Pt Pham, Y Largman, and Ay Ng. Unsupervised feature learning for audio classification using convolutional deep belief networks. *Nips*, pages 1–9, 2009.

[14] Eric J Humphrey, Aron P Glennon, and Juan Pablo Bello. Non-linear semantic embedding for organizing large instrument sample libraries. In *Machine Learning and Applications and Workshops (ICMLA), 2011 10th International Conference on*, volume 2, pages 142–147. IEEE, 2011.

[15] John Thickstun, Zaid Harchaoui, and Sham Kakade. Learning Features of Music from Scratch. pages 1–14, 2016.

[16] Soroush Mehri, Kundan Kumar, Ishaan Gulrajani, Rithesh Kumar, Shubham Jain, Jose Sotelo, Aaron Courville, and Yoshua Bengio. SampleRNN: An Unconditional End-to-End Neural Audio Generation Model. pages 1–11, 2016.

[17] Jesse Engel, Cinjon Resnick, Adam Roberts, Sander Dieleman, Douglas Eck, Karen Simonyan, and Mohammad Norouzi. Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders. 2017.

[18] Andy M Sarroff and Michael Casey. Musical Audio Synthesis Using Autoencoding Neural Nets. *Proceedings of the International Computer Music Conference*, 1(September):14–20, 2014.

[19] Shinji Takaki and Junichi Yamagishi. A deep auto-encoder based low-dimensional feature extraction from fft spectral envelopes for statistical parametric speech synthesis. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pages 5535–5539. IEEE, 2016.

[20] Spyridon Stasis, Ryan Stables, and Jason Hockman. A Model for Adaptive Reduced-Dimensionality Equalisation. *Proc. of the 18th Int. Conference on Digital Audio Effects (DAFx-15)*, pages 1–6, 2015.

[21] Takaaki Ishii, Hiroki Komiyama, Takahiro Shinozaki, Yasuo Horiuchi, and Shingo Kuroiwa. Reverberant speech recognition based on denoising autoencoder. In *Interspeech*, pages 3512–3516, 2013.

[22] Emmanouil Benetos, Margarita Kotti, and Constantine Kotropoulos. Musical in-

strument classification using non-negative matrix factorization algorithms and subset feature selection. In *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, volume 5, pages V–V. IEEE, 2006.

[23] Pablo Sprechmann, AM Bronstein, and Guillermo Sapiro. Supervised non-negative matrix factorization for audio source separation. *Vista.Eng.Tau.Ac.Il*, pages 1–14, 2014.

[24] G. E. Hinton. Reducing the Dimensionality of Data with Neural Networks. *Science*, 313(5786):504–507, 2006.

[25] Matthias Scholz, Martin Fraunholz, and Joachim Selbig. Nonlinear principal component analysis: neural network models and applications. *Principal manifolds for data visualization and Dimension Reduction*, pages 45–68, 2008.

[26] Pascal Vincent PASCALVINCENT and Hugo Larochelle LAROCHEH. Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion Pierre-Antoine Manzagol. *Journal of Machine Learning Research*, 11:3371–3408, 2010.

[27] Daniel Griffin and Jae Lim. Signal estimation from modified short-time fourier transform. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 32(2):236–243, 1984.

[28] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. pages 1–15, 2014.

[29] Gérard Grisey. *Partiels [: pour 18 musiciens: partitura*. Ricordi, 1976.