AUTOENCODERS FOR MUSIC SOUND SYNTHESIS: A COMPARISON OF LINEAR, SHALLOW, DEEP AND VARIATIONAL MODELS

Fanny Roche[1,2]   **Thomas Hueber**[1]   **Samuel Limier**[2]   **Laurent Girin**[1,3]

[1] Univ. Grenoble Alpes, CNRS, Grenoble INP, GIPSA-lab, 38000 Grenoble, France
[2] Arturia, 38240 Meylan, France
[3] INRIA, Perception Team, Montbonnot, France

`fanny.roche@gipsa-lab.fr`

ABSTRACT

This study investigates the use of non-linear unsupervised dimensionality reduction techniques to compress a music dataset into a low-dimensional representation, and its use for the synthesis of new sounds. We systematically compare (shallow) autoencoders (AE) and deep autoencoders (DAE) with principal component analysis (PCA) for representing the high-resolution short-term amplitude spectrum of a large and dense dataset of music notes into a lower-dimensional vector (and then convert it back to a synthetic amplitude spectrum used for sound resynthesis). In addition, we report results obtained with variational autoencoders (VAE) which to our knowledge have never been considered for processing musical sounds. Our experiments were conducted on the publicly available multi-instrument and multi-pitch database NSynth. Interestingly and contrary to the recent literature on image processing, they showed that PCA systematically outperforms shallow AE and that only a deep architecture (DAE) can lead to a lower reconstruction error. The optimization criterion in deep VAE being the sum of the reconstruction error and a regularization term, it naturally leads to a lower reconstruction accuracy than DAE but we show that VAEs are still able to outperform PCA while providing a low-dimensional latent space with nice "usability" properties.

1. INTRODUCTION

Music sound synthesis is a research field that has interested labs for more than four decades and is still very active. Many different methods for sound generation in music exist and are commonly used in synthesizers such as additive synthesis, subtractive synthesis, frequency modulation, wavetable synthesis or physical modeling [17].

For a few years, researchers have started to investigate new synthesis methods using data-driven machine learning techniques and artificial intelligence, in particular artificial neural networks (ANNs) [6, 7, 20]. Yet, the application of ANNs to music sounds synthesis is still largely open compared to other music applications where machine learning has already been extensively used, such as music recommendation, genre classification, or instrument recog-

nition, and also compared to image synthesis [11, 15, 22]. Only a few studies in audio processing have been proposed, with a general principle that is similar to image synthesis/transformation: projection of the signal space into a low-dimension latent space (encoding or embedding), modification of the latent coefficients, and inverse transformation of the modified latent coefficients into the original signal space (decoding).

The first study we found to apply this principle on music signals is [20]. The author used autoencoders to process normalized magnitude spectra randomly extracted from a dataset of 8,000 songs from 10 different musical genres. An autoencoder (AE) is a specific ANN architecture which is trained to reconstruct the input at the output layer, after passing through the latent space (see Section 2.2.2). The output audio signal is reconstructed using the decoded magnitude spectra and the phase of the original signal. Although the principle of AE-based audio synthesis was validated, limited reconstruction accuracy was achieved. In [6] the authors tried to improve the results obtained in [20] by using a different optimizer and a more controled database of sounds (generated with a MicroKORG synthesizer). In both [20] and [6], the authors tried various topologies of the networks going from shallow AEs to deep AEs (DAEs) with up to 9 layers, different activation functions and different numbers of neurons per layer. Evaluation was made by computing the mean squared error (MSE) between the original and the reconstructed magnitude spectra. However, no comparisons with other types of dimensionality reduction techniques, and in particular with linear techniques such as principal component analysis (PCA), were reported in these papers. It is thus difficult to have a quantitative evaluation and to understand how well-suited these models are for such task as audio generation (beginning with analysis-resynthesis without transformation).

Google Magenta's team took inspiration from the WaveNet speech synthesizer [21] (itself adapted from PixelRNN [22] used in image processing) to propose NSynth, an audio synthesis method using a WaveNet autoencoder [7]. The authors investigated the use of this model to find a high-level latent space well-suited for interpolation between instruments. Their autoencoder is conditioned on pitch and is fed with raw audio from their large-scale multi-

instrument and multi-pitch database (the NSynth dataset). The model is composed of a temporal encoder made of a 30-layer residual network of dilated convolutions followed by 1x1 convolutions, and a WaveNet decoder producing raw audio similar to [21]. This model leads to a latent embedding of size 125x16 which is then upsampled before being given to the decoder. Qualitative evaluation reports that NSynth is able to reconstruct the signal better than a baseline model (a convolutional autoencoder applied on normalized log-magnitude power spectra) and that the extracted latent dimensions are more musically interesting for instrument "morphing". Moreover, the use of raw audio samples permits to avoid the need to use the original signal phase when resynthesizing sounds from spectrograms as in [6, 20]. However, here again, the evaluation was only qualitative and the NSynth model was not compared to classical autoencoders or PCA. Also, this model may be difficult to train for non-experts and requires a lot of computational power.

Another popular technique to synthesize data using deep learning is the so-called variational autoencoder (VAE), which can be seen as a probabilistic version of an AE conditioned on the latent space [15]. In addition, the distribution of the latent variables can be controlled, so that they are assumed to be more "meaningful" for signal resynthesis, compared to the bottleneck layer outputs of regular AEs (i.e. compared to AE, each latent variable of VAE may control more clearly one single perceptual signal dimension). All this makes VAEs well suited for signal synthesis and they were originally applied to image generation. Recent studies proposed to use VAEs for the modeling and transformation of speech signals [3, 13]. In these studies, VAEs were applied to log-spectrograms of speech signals with the purpose of using the extracted latent space for modeling and modifying some attributes of speech and/or speakers. Here again, no comparison with linear techniques was done.

The goal of the current paper is to present new contributions to music sound analysis-synthesis in the continuation of the above-presented studies. First, we report the results of our own experiments conducted with AEs and DAEs applied on high-resolution short-term amplitude spectra, and using (part of) the NSynth dataset. In contrast to all the above-mentioned studies, we systematically compare the results with the ones obtained with PCA. Interestingly and contrary to the recent literature on image processing, in our experiments PCA systematically outperforms shallow AE, and only deep architectures (DAEs) lead to a lower reconstruction error. In addition, we report results obtained with (deep) VAEs, which, to our knowledge, were never considered for processing musical sounds. The optimization criterion in VAEs is the sum of a reconstruction error and a regularization term. This naturally leads to lower reconstruction accuracy than DAEs (with similar architecture dimension) but we show that VAEs are still able to outperform PCA while (theoretically) providing a low-dimensional latent space with nice "usability" properties.

2. METHODOLOGY

The global methodology we applied for (V)AE-based analysis-transformation-synthesis of audio signals is in line with previous works [3, 6, 13, 20]. It is illustrated in Figure 1 and is described in the next subsections.

2.1 Analysis-Synthesis

First, a Short-Term Fourier Transform (STFT) analysis is performed on the input audio signal. The magnitude spectra are sent to the model (encoder input) on a frame-by-frame basis, and the phase spectra are stored for the synthesis stage. After possible modifications of the extracted latent variables (at the (V)AE bottleneck layer output, see next subsection), the magnitude spectra is reconstructed by the decoder. Finally, the audio signal is synthesized by combining the decoded magnitude spectra with the original phase spectra, and by applying inverse STFT with overlap-add. Technical details are given in Section 3.2.

2.2 Dimensionality Reduction Techniques

2.2.1 Principal Component Analysis

As a baseline, we investigated the use of Principal Component Analysis (PCA) to reduce the dimensionality of log-magnitude spectra feature vector. PCA is the optimal linear orthogonal transformation that provides a new coordinate system (i.e. the latent space) in which basis vectors follow modes of greatest variance in the original data [2].

2.2.2 Autoencoder

An autoencoder [10] is a specific kind of ANN traditionally used for dimensionality reduction thanks to its diabolo shape, see Figure 2 (top). It is composed of an encoder and a decoder. The encoder maps a high-dimensional low-level input data vector $\mathbf{x}$ into a low-dimensional higher-level latent vector $\mathbf{z}$, which is assumed to nicely encode properties or attributes of $\mathbf{x}$. The mapping writes:

$$\mathbf{z} = f_{enc}(\mathbf{W}_{enc}\,\mathbf{x} + \mathbf{b}_{enc}),$$

where f_{enc} is a non-linear activation function, $\mathbf{W}_{enc}$ is a weight matrix and $\mathbf{b}_{enc}$ is a bias vector. Similarly, the decoder reconstructs an estimate $\hat{\mathbf{x}}$ of the input vector $\mathbf{x}$ from the latent vector $\mathbf{z}$ with:

$$\hat{\mathbf{x}} = f_{dec}(\mathbf{W}_{dec}\,\mathbf{z} + \mathbf{b}_{dec}).$$

For regression tasks (such as the one considered in this study), a linear activation function is generally used for the output layer.

During the training of the model, the weights matrices and the bias vectors are learned by minimizing some cost function over a (large) training dataset. The cost function that is commonly used is the mean squared error (MSE) between the input $\mathbf{x}$ and the output $\hat{\mathbf{x}}$.

The model can be improved by adding hidden layers in both the encoder and decoder to create a deep autoencoder (DAE), as illustrated in Figure 2 (bottom). This aims at extracting even more abstract latent features. In this case,
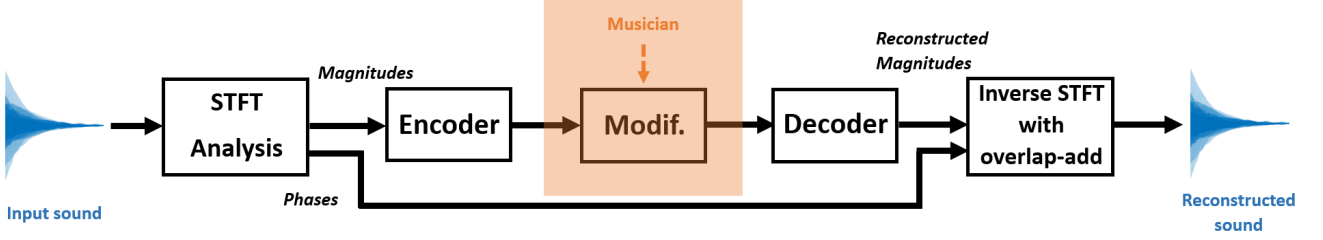
Figure 1. Global diagram of the sound analysis-transformation-synthesis process.

end-to-end training of the deep model can be difficult, as it can get stuck in a local minimum. A solution is to proceed to a layer-wise training by considering the DAE as a stack of shallow AEs [1, 12] before fine-tuning the entire model.
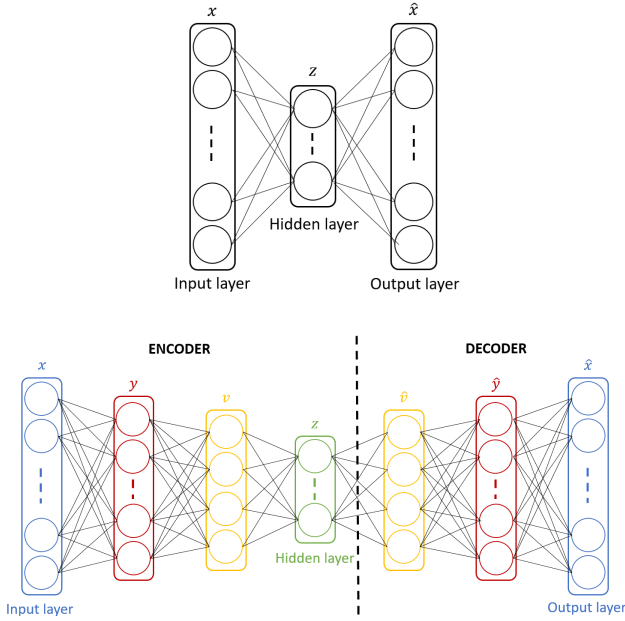

Figure 2. General architecture of a shallow autoencoder (AE, top) and a deep autoencoder (DAE, bottom).

2.2.3 *Variational Autoencoder*

As mentioned in the introduction, variational autoencoders (VAEs) can be seen as probabilistic AEs. A VAE delivers a (parametric) model of data distribution:

$$p_\theta(\mathbf{x}, \mathbf{z}) = p_\theta(\mathbf{x}|\mathbf{z})p_\theta(\mathbf{z}).$$

θ denotes the set of distribution parameters. In the present context, the likelihood function $p_\theta(\mathbf{x}|\mathbf{z})$ plays the role of a probabilistic decoder which models how the generation of observed data $\mathbf{x}$ is conditioned on the latent data $\mathbf{z}$. The prior distribution $p_\theta(\mathbf{z})$ is used to structure (or regularize) the latent space. Typically a standard Gaussian distribution is used [15]:

$$p_\theta(\mathbf{z}) = \mathcal{N}(\mathbf{z}; \mathbf{0}, \mathbf{I}),$$

where $\mathbf{I}$ is the identity matrix. This forces the latent coefficients to be orthogonal and with similar range. The likelihood $p_\theta(\mathbf{x}|\mathbf{z})$ is usually defined as a Gaussian density:

$$p_\theta(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_\theta(\mathbf{z}), \boldsymbol{\sigma}_\theta^2(\mathbf{z})), \qquad (1)$$

where $\boldsymbol{\mu}_\theta(\mathbf{z})$ and $\boldsymbol{\sigma}_\theta(\mathbf{z})$ are the outputs of the decoder ANN (hence the parameter set θ is composed of $\mathbf{W}_{dec}$ and $\mathbf{b}_{dec}$). Note that $\boldsymbol{\sigma}_\theta^2(\mathbf{z})$ indifferently denotes the covariance matrix of the distribution, which is assumed diagonal, or the vector of diagonal entries.

The exact posterior distribution $p_\theta(\mathbf{z}|\mathbf{x})$ corresponding to the above model is intractable. It is approximated with a tractable parametric model $q_\phi(\mathbf{z}|\mathbf{x})$ that will play the role of the corresponding probabilistic encoder. This model generally has a form similar to the probabilistic decoder:

$$q_\phi(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\mathbf{z}; \tilde{\boldsymbol{\mu}}_\phi(\mathbf{x}), \tilde{\boldsymbol{\sigma}}_\phi^2(\mathbf{x})),$$

where $\tilde{\boldsymbol{\mu}}_\phi(\mathbf{x})$ and $\tilde{\boldsymbol{\sigma}}_\phi(\mathbf{x})$ are the outputs of the encoder ANN (the parameter set ϕ is composed of $\mathbf{W}_{enc}$ and $\mathbf{b}_{enc}$; $\tilde{\boldsymbol{\sigma}}_\phi^2(\mathbf{x})$ is diagonal or is the vector of diagonal entries).

Training of the VAE model, i.e. estimation of θ and ϕ, is made by maximizing the marginal log-likelihood $\log p_\theta(\mathbf{x})$ over a large training dataset of vectors $\mathbf{x}$. It can be shown that the marginal log-likelihood writes [15]:

$$\log p_\theta(\mathbf{x}) = \mathrm{D}_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}|\mathbf{x})) + \mathcal{L}(\phi, \theta, \mathbf{x}),$$

where $\mathrm{D}_{\mathrm{KL}} \geq 0$ denotes the Kullback-Leibler divergence (KLD) and $\mathcal{L}(\phi, \theta, \mathbf{x})$ is the variational lower bound (VLB) given by:

$$\mathcal{L}(\phi, \theta, \mathbf{x}) = \underbrace{-\mathrm{D}_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z}))}_{\text{regularization}}$$
$$+ \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})]}_{\text{reconstruction accuracy}}. \quad (2)$$

In practice, the model is trained by maximizing $\mathcal{L}(\phi, \theta, \mathbf{x})$ with respect to parameters ϕ and θ. We can see that the VLB is the sum of two terms. The first term acts as a regularizer encouraging the approximate posterior $q_\phi(\mathbf{z}|\mathbf{x})$ to be as close as possible to the prior $p_\theta(\mathbf{z})$. The second term represents the average reconstruction accuracy. Since this term is difficult to compute analytically, it is approximated using a Monte Carlo estimate:

$$\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] \simeq \frac{1}{L}\sum_{l=1}^{L} \log p_\theta(\mathbf{x}|\mathbf{z}_l),$$

where $\mathbf{z}_l$ are samples drawn from $q_\phi(\mathbf{z}|\mathbf{x})$. Also, in order to make the gradient of the above form differentiable with respect to ϕ, the so-called reparameterization trick was introduced in [15]: The sampling is turned into a deterministic mapping that depends on ϕ combined with a sampling

independent of ϕ: $\mathbf{z}_l = \tilde{\boldsymbol{\mu}}_\phi(\mathbf{x}) + \tilde{\boldsymbol{\sigma}}_\phi(\mathbf{x}) \odot \boldsymbol{\epsilon}_l$, where $\odot$ represents element-wise multiplication and $\boldsymbol{\epsilon}_l \sim \mathcal{N}(0, \mathbf{I})$.

In [3, 13], not much information is given about the precise choice of $p_\theta(\mathbf{x}|\mathbf{z})$. All that we know so far is that, in every case, including our study, $\mathbf{x}$ is a vector of real-valued spectral magnitudes (or magnitudes squarred or log-magnitudes), possibly normalized, and phases are processed separately. Choosing the Gaussian model (1) leads to a reconstruction error term in (2) of the form (for one data sample and up to a constant):

$$\log p_\theta(\mathbf{x}|\mathbf{z}) = -\sum_{d=1}^{D} \left(\log \sigma_{\theta,d}^2(\mathbf{z}) + \frac{(x_d - \mu_{\theta,d}(\mathbf{z}))^2}{2\sigma_{\theta,d}^2(\mathbf{z})} \right), \quad (3)$$

where the subscript d denotes the d-th entry of a vector, and D is the dimension of the data vector $\mathbf{x}$. In the case where the variance $\sigma_{\theta,d}^2(\mathbf{z})$ is fixed and set to be equal at every entry d, (3) becomes proportional to a mean square error (MSE) between inputs $\mathbf{x}$ and decoder outputs $\boldsymbol{\mu}_\theta(\mathbf{x})$, which simplifies much the training and continues to make sense as a reconstruction criterion. Alternately, the case with $\mu_{\theta,d}(\mathbf{z}) = 0, \forall d$, and $\sigma_{\theta,d}^2(\mathbf{z})$ let free also makes sense: (3) then becomes proportional to the Itakura-Saito distance (ISD) between $\mathbf{x}$ and $\boldsymbol{\sigma}_\theta^2(\mathbf{z})$. And it was shown in [8] that, if the input vector $\mathbf{x}$ is composed of linear-scale squared magnitudes, minimizing this ISD corresponds to maximum-likelihood estimation of $\boldsymbol{\sigma}_\theta^2(\mathbf{z})$ under the assumption of a zero-mean proper complex Gaussian model for the STFT coefficient vector corresponding to $\mathbf{x}$ (here with variance equal to $2\boldsymbol{\sigma}_\theta^2(\mathbf{z})$). Such model and associated fitting procedure have been used extensively in audio processing, especially in the context of non-negative matrix factorization (NMF) modeling of audio power spectrograms, and applications to audio source separation, see e.g. [9,16,18]. The ISD has a scale-invariance property that has been claimed to be of interest in the audio processing context of modeling power spectra [8].

In our present work, we chose to set $\sigma_{\theta,d}^2(\mathbf{z}) = 1, \forall d$ and estimate $\boldsymbol{\mu}_\theta(\mathbf{x})$, even if this is not supported by a clear interpretation in terms of underlying statistical framework. This is because in our experiments the use of the ISD generally led to poor reconstruction accuracy in term of MSE (which is not surprising) and audio quality, whatever the balance between reconstruction and regularization in the VLB (see below). In other words, we deliberately turn the reconstruction term of the VLB into an MSE to guarantee satisfying reconstruction in the MSE sense and good signal quality.

In addition, as discussed in [3], we had to find an appropriate balance between the reconstruction term and the regularization term of the VLB in order to obtain a nice trade-off between output signal quality and "compactness"/orthogonality of the latent control coefficients $\mathbf{z}$. Indeed, if the reconstruction term is too strong relatively to the regularization term, then the distribution of the latent space will be far from the prior $p_\theta(\mathbf{z})$, somehow turning the VAE into an AE. Reversely, if it is too weak, then the model can get stuck in a state where most latent coefficients are saturated or inactive and lead to poor reconstruc-

tion. The balance was controlled using a weighted version of (2):

$$\mathcal{L}(\phi, \theta, \beta, \mathbf{x}) = -\beta\ \mathrm{D_{KL}}(q_\phi(\mathbf{z}|\mathbf{x})||p_\theta(\mathbf{z})) \\ + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})], \quad (4)$$

where β is a weight set empirically so that the KLD term and the reconstruction error are in the same range. A deeper investigation of the use of ISD or the general form (3) in the present analysis-synthesis framework, in connection with perceptual aspects, is left for future works.

3. EXPERIMENTS

3.1 Dataset

In this study, we used the NSynth dataset introduced in [7]. This is a large database (more than 30 GB) of 4s long monophonic music sounds sampled at 16 kHz. They represent 1,006 different instruments generating notes with different pitches (from MIDI 21 to 108) and different velocities (5 different levels from 25 to 127). To generate these samples different methods were used: Some acoustic and electronic instruments were recorded and some others were synthesized artificially. The dataset is labeled using different criteria: i. instrument family (e.g., keyboard, guitar, synth_lead, reed), ii. source (acoustic, electronic or synthetic), iii. instrument index within the instrument family, iv. pitch value, and v. velocity value. Some other labels qualitatively describe the samples, e.g. brightness or distortion, but they were not of any use in our work.

To train our models, we limited the database to a set of 1,000 different sounds chosen randomly from this NSynth database, representing all families of instruments, different pitches and different velocity values. We split this dataset into a training set (80%) and testing set (20%). During the training phase, 20% of the training set was kept for validation. All the results presented in this section were obtained using k-fold cross-validation with $k = 5$.

3.2 Data Pre-Processing

For magnitude and phase short-term spectra extraction, we applied a 1024-point STFT to the input signal using a sliding Hamming window with 50% overlap. Frames corresponding to silence segments were removed. The corresponding 513-point positive-frequency magnitude spectra were then converted to log-scale and normalized in energy: We fixed the maximum of each log-spectrum input vector to 0 dB (the energy coefficient was stored to be used for signal reconstruction). Then, the log-spectra were thresholded, i.e. every log-magnitude below a fixed threshold was set to the threshold value. Finally they were normalized in between -1 and 1, which is a usual data reshaping procedure for ANNs inputs. 3 threshold values were tested: -80 dB, -90 dB and -100 dB. Corresponding denormalization, log-to-linear conversion and energy equalization were applied after the decoder, before signal reconstruction with transmitted phases and inverse STFT with overlap-add.

4

3.3 Autoencoders Implementations

We tried different types of autoencoders: AE, DAE and VAE. For all the models we investigated several encoding dimensions from 4 to 100 (with a fine-grained sampling for encoding dimension lower than 16). Different architectures were tested for the DAEs: [513, 128, enc, 128, 513], [513, 256, enc, 256, 513] and [513, 256, 128, enc, 128, 256, 513] (enc being the size of the latent layer). The architecture we used for the VAE was [513, 128, enc, 128, 513]. For all the neural models, we tested different pairs of activation functions for the hidden layers and output layer, respectively: (tanh, linear), (sigmoid, linear) and (tanh, sigmoid).

AE, DAEs and VAEs were implemented using the *Keras* deep learning toolkit [4] (we used the *scikit-learn* [19] toolkit for the PCA). Training was performed using the Adam optimizer [14] with a learning rate of 10^{-3} over 600 epochs with early stopping criterion and a batch size of 512. The DAEs were trained in two different ways, with and without layer-wise training.

3.4 Experimental Results

Figure 3 shows the evolution of the reconstruction error (RMSE in dB) obtained with PCA, AE and DAEs on the test set (averaged over the 5 folds of the cross-validation procedure), as a function of the dimension of the latent space. The results obtained with VAE (using the same protocol) are shown in Figure 4. For the sake of clarity, we present here only the results obtained for i) a threshold of -100 dB applied on the log-spectra, and ii) a restricted set of the tested AE, DAE and VAE architectures (listed in the legends of the figures). Similar trends were observed for other thresholds and other tested architectures. For each considered dimension of the latent space, a 95% confidence interval of each reconstruction error was obtained by conducting paired t-test, considering each sound (i.e. each audio file) of the test set as an independent sample.
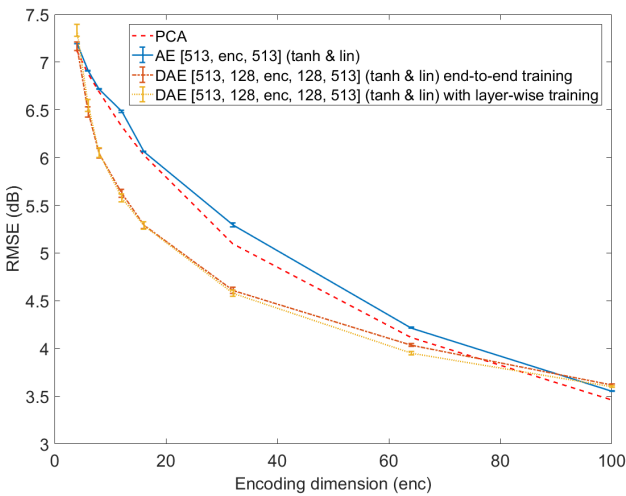

Figure 3. Reconstruction error (RMSE in dB) as a function of latent space dimension obtained with PCA, AE and DAE (with and without layer-wise training).
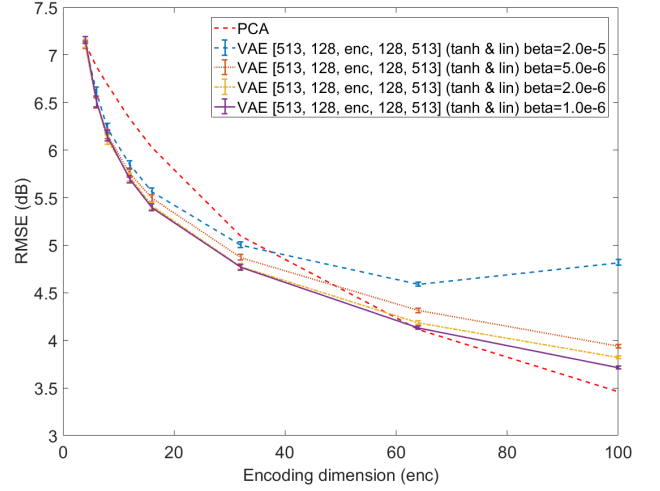

Figure 4. Reconstruction error (RMSE in dB) as a function of latent space dimension obtained with VAEs (the performance obtained with PCA is also recalled).

As expected, the RMSE decreases with the dimension of the latent space for all methods. Interestingly, PCA systematically outperforms (or at worst equals) shallow AE. This somehow contradicts recent studies on image compression for which a better reconstruction is obtained with AE compared to PCA [12]. To confirm this unexpected result, we replicated our PCA vs. AE experiment on the MNIST image dataset, using the same AE implementation and a standard image preprocessing (i.e. vectorization of 28×28 pixels gray scale image into a 784-dimensional feature vector). In accordance with the literature, best performance was systematically obtained with AE (for any considered dimension of the latent space). This difference of AE's behavior when considering audio and image data was unexpected and may be a first experimental contribution of the present study. Then, contrary to (shallow) AE, DAEs systematically outperform PCA (and thus AE), with up to more than 12% improvement (for $enc = 16$). Our experiments did not reveal notable benefit of layer-by-layer DAE training over end-to-end training. Importantly, for small dimensions of the latent space (e.g. smaller than 16), RMSE obtained with DAE decreases much faster than with PCA and AE. These results confirm the benefits of a deep architecture able to extract high-level abstractions and compress efficiently the audio space. This is of great interest for sound synthesis for which the latent space has to be kept as low-dimensional as possible (while maintaining a good reconstruction accuracy) in order to be "controlled" by a musician.

Figure 4 shows that the overall performances of VAEs are in between DAEs and PCA (and thus AE). Let recall that minimizing the reconstruction accuracy is not the only "goal" of VAE which also aims at constraining the distribution of the latent space. As shown in Figure 4, the parameter β, which balances regularization and reconstruction accuracy in (4), plays a major role. As expected, high β values foster regularization at the expense of reconstruction accuracy. However, with $\beta \leqslant 2.10^{-6}$ the VAE clearly
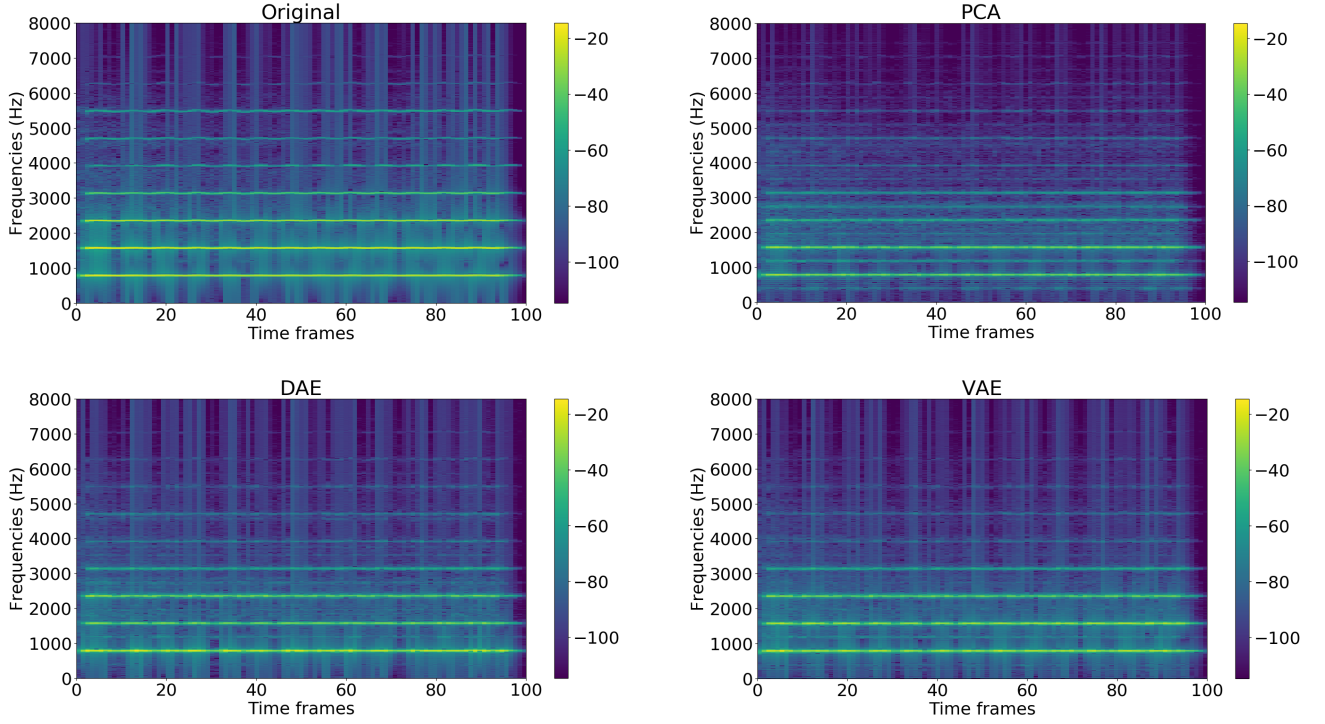
Figure 5. Examples of original and reconstructed log-spectrograms for a reed sound (files with basename reed_acoustic_029-079-050 provided as supplementary materials): original (top left), reconstruction with PCA (top right), DAE (bottom left), VAE (bottom right), for a 32-dimensional latent space.

outperforms PCA, e.g. up to 10% for $enc = 12$ or 16.

Finally, Figure 5 displays the log-spectrogram of a reed sound example, reconstructed using PCA, DAE or VAE. Contrary to PCA, DAE reconstructs very well the 4 first harmonics and we can clearly distinguish up to 7 harmonics. For PCA reconstruction, some artifacts lie below the first harmonic and between the first and second harmonic which are not generated by the DAE. For the VAE, the reconstructed spectrogram, obtained with $\beta = 1.10^{-6}$, is very close to the one obtained with the DAE, as in-between harmonics artifacts are not visible and the first harmonics are well-rendered. Examples of audio samples reconstructed with the different models can be found at `https://goo.gl/GTDnGH`.

4. CONCLUSIONS AND PERSPECTIVES

In this study, we investigated dimensionality reduction based on autoencoders to extract latent dimensions from a music sound dataset. Our final goal is to provide a musician with a new way to generate sound textures by exploring a perceptually meaningful low-dimensional space. From the experiments conducted on a subset of the publicly available database NSynth, we can draw the following conclusions: i) contrary to the literature on image processing, shallow autoencoders (AEs) do not here outperform principal component analysis (in terms of reconstruction accuracy), ii) best reconstruction performance is always obtained with deep autoencoders (DAE), iii) variational autoencoders lead to a fair reconstruction accuracy while

constraining the statistical properties of the latent space (i.e. maintaining some kind of orthogonality between encoding dimensions and limiting the range of the latent coefficients), which makes them good candidates for our targeted application.

In line with the latest conclusion, future works will mainly focus on VAE and in particular, i) a better understanding of the relationships between both the regularization and reconstruction terms in the loss function and the perceptual meaning of the resulting latent dimensions, and ii) the extension of the proposed static model to a dynamic one (such as the one recently proposed in [5]) in order to take into account the temporal characteristics of a musical sound (and thus enable the musician to modify them). Finally, experiments will be conducted with Generative Adversarial Networks [11] which is another popular technique for data generation powered by machine learning.

5. ACKNOWLEDGMENT

6. REFERENCES

[1] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle. Greedy layer-wise training of deep networks. In *Advances in Neural Information Processing Systems 19 (NIPS)*. Vancouver, Canada, 2007.

[2] C. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[3] M. Blaauw and J. Bonada. Modeling and transforming speech using variational autoencoders. In *Conf. of the Int. Speech Communication Association (Interspeech*, San Francisco, CA, 2016.

[4] F. Chollet et al. Keras. `https://keras.io`, 2015.

[5] J. Chung, K. Kastner, L. Dinh, K. Goel, A. C. Courville, and Y. Bengio. A recurrent latent variable model for sequential data. In *Advances in neural information processing systems*, pages 2980–2988, 2015.

[6] J. Colonel, C. Curro, and S. Keene. Improving neural net auto encoders for music synthesis. In *Audio Engineering Society Convention*, New-York, NY, 2017.

[7] J. Engel, C. Resnick, A. Roberts, S. Dieleman, D. Eck, K. Simonyan, and M. Norouzi. Neural audio synthesis of musical notes with wavenet autoencoders. *arXiv preprint arXiv:1704.01279*, 2017.

[8] C. Févotte, N. Bertin, and J.-L. Durrieu. Nonnegative matrix factorization with the itakura-saito divergence: With application to music analysis. *Neural computation*, 21(3):793–830, 2009.

[9] T. Gerber, M. Dutasta, L. Girin, and C. Févotte. Professionally-produced music separation guided by covers. In *International Society for Music Information Retrieval Conference (ISMIR)*, Porto, Portugal, 2012.

[10] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27 (NIPS)*. Montreal, Canada, 2014.

[12] G. Hinton and R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[13] W.-N. Hsu, Y. Zhang, and J. Glass. Learning latent representations for speech generation and transformation. *arXiv preprint arXiv:1704.04222*, 2017.

[14] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[15] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[16] D. Kounades-Bastian, L. Girin, X. Alameda-Pineda, S. Gannot, and R. Horaud. An inverse-gamma source variance prior with factorized parameterization for audio source separation. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Shanghai, China, 2016.

[17] E. Miranda. *Computer Sound Design: Synthesis Techniques and Programming*. Music Technology series. Focal Press, 2002.

[18] A. Ozerov and C. Févotte. Multichannel nonnegative matrix factorization in convolutive mixtures for audio source separation. *IEEE Transactions on Audio, Speech and Language Processing*, 18(3):550–563, 2010.

[19] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[20] A. Sarroff and M. Casey. Musical audio synthesis using autoencoding neural nets. In *Joint International Computer Music Conference (ICMC) and Sound & Music Computing conference (SMC)*, Athens, Greece, 2014.

[21] A. Van Den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[22] A. van den Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel recurrent neural networks. *arXiv preprint arXiv:1601.06759*, 2016.