EE678: Wavelets

R&D Project Report

Adaptive Reconstruction Filter-Banks using Autoencoders

Group 6:
Rohit M A (183076001)
Shreeharsha B S (18307R002)
Kawal Jeet Singh (194070012)

Course Instructor:
Prof. V.M. Gadre

Department of Electrical Engineering
Indian Institute Of Technology Bombay
June 2020

1 Introduction

Wavelet-based methods are known to be better suited for analysis of transient signal structures than the more ubiquitous Fourier-based counterparts. While a Fourier analysis decomposes signals into sines and cosines, the wavelet method uses piece-wise continuous functions at various scales as bases. This helps provide improved time and frequency resolution and is therefore more effective in detecting transient patterns like spikes and bursts in 1D signals, and edges in 2D [7]. Moreover, in the case of audio signals, the wavelet decomposition results in better localisation in time for high frequency content and better frequency selectivity at low frequencies, and is thus also appropriate because of its close similarity to the way human auditory perception works [5].

The various time-frequency transforms like the Short-Time Fourier Transform (STFT), Discrete Wavelet Transform (DWT), etc., have been used to not only facilitate better signal analysis and modification in tasks like signal compression and denoising, but also in machine learning tasks, by offering a more suitable representation from which to extract features. This is especially true of the audio signal processing field, where the time-domain waveform is far too raw to work with and the input features are designed from a more compact version of a suitable time-frequency map of the signal that makes patterns more apparent. For instance, for most speech and music related tasks, the magnitude spectrogram is the common representation and is often further transformed to a more perceptual and compact mel-scale spectrogram.

However, the foregoing of a hand-crafted signal processing based feature extraction step in favour of "learning" features directly from data (a time-frequency map in case of 1D signals, and the signal itself in case of 2D image data) has been the norm for quite some time now. But this does not always leave a model amenable to adequate interpretation. In some cases, especially for 1D data, learning features from the raw waveform can also be extremely computationally expensive [3]. It is generally assumed that simpler models like regression and decision trees, have higher interpretability than complex architectures involving several layers of deep neural networks. Providing interpretability to these deep learning techniques is therefore now an evolving field of research. It is also important to understand what a model has learnt and ensure that the model predictions on a test example are not just due to exploiting certain artifacts in the test data [8].

Given this issue of model interpretability and the constant need for a concise representation of signals that accurately preserves the attributes important to the task at hand, recent advances in this field therefore have involved designing an adaptive "front-end" neural network to also learn a representation from which subsequent layers learn features [13, 4]. It has been shown that learning such a task-specific representation can overcome some limitations of existing transforms and make it more suitable for the dataset and task being

addressed. In this work, we attempt to undertake the designing of such a neural network capable of learning a time-frequency representation, by drawing inspiration from the wavelet-based perfect reconstruction filter banks (shown in figure 1) consisting of FIR filters (e.g., a Haar filter bank). In particular, we examine the autoencoder architecture implemented using fully connected and convolutional layers. We attempt to provide some interpretation to the layers of the autoencoder, specifically on the orthogonality of the filters involved.

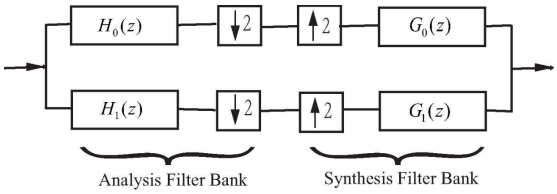


Figure 1: A 2-channel analysis-synthesis filter bank

2 Background and Motivation

Wavelet analyses have found use in a number of research areas involving various kinds of one and two dimensional signals. Since the present work happens to be more on audio signals, we first review some of the past work in the audio signal processing field. This is to help provide a better perspective about the motivation for this work and the directions in which it can be extended.

The ability of a wavelet decomposition to localize the signal in a few levels of decomposition was first exploited to improve audio compression and coding schemes [9] for communication and storage. This naturally then extended to audio indexing for query-by-example tasks where the search query was an audio snippet and similar audios had to be fetched. Wavelet transform based methods were used here to obtain a better and more compact signal representation and thus make searching more efficient [10]. The filter-bank implementation of the DWT made it particularly attractive to computer music researchers who used it to modify coefficients at particular levels to result in interesting re-synthesized sounds [6]. The decomposition into levels with more perceptually accurate time and frequency resolutions offered by the DWT also led to its use as a pre-processing step to extracting features for various tasks. For example, in [11], the mean and standard deviation of the coefficient values in each sub-band were used to uniquely characterise sounds from a few broad categories like human speech, music, noise, etc. Along similar lines, in [12], sub-bands corresponding to specific empirically derived decomposition levels were used to identify locations of the strokes of 2 different kinds of drums - the lower band for the bass drum and the higher for the snare drum, in a stream of audio.

In our work, we choose to explore the problem of accurately modelling percussive drum sounds, having different spectral characteristics, in audio signals. Such a model could then be useful in various tasks like - identifying occurrences of these events in test signals, providing better control for desired modifications before re-synthesizing an input signal, synthesis of signals from compressed information, etc. While these tasks are all fairly well-studied and researched, what we are interested in is in using an adaptive method to achieve the decomposition that allows learning filters from the data, in devising ways to influence this approach with principles from the traditional methods, and to investigate if doing so offers any advantages.

Our main motivation for trying to find ways to connect the traditional wavelet-based and a modern learning-based paradigm is based on the observation that the learned filters in some neural network models designed for tasks such as audio source separation or image classification, appear similar to wavelet bases - localised in time and frequency [13]. A survey of more literature along these lines led us to the more recent and closely related work by Khan et. al. [4], where, a regular CNN preceded by a novel wavelet deconvolution (WD) layer was used for an automatic speech recognition(ASR) task. The WD layer was responsible for producing an optimized spectral decomposition by learning the best scale values to use in the decomposition. The wavelet used was a sampled Gaussian wavelet. The CNN layers on top of this layer were tasked with learning the appropriate features and producing an output. The scale values were randomly initialised and updated using back-propagation. It turned out that in the process, the filters learnt in the WD layer were remarkably similar to the mel-filters used quite frequently in ASR.

3 Dataset

For our present work, we report the experiments performed on a dataset of isolated drum sound recordings. These sounds are a good example of transients that could be harder to model using Fourier methods due to the sharp and percussive nature of the onset and the spread of energy across the frequency spectrum. We note that some alternative datasets could include sounds from natural scenes (e.g. gunshot, door closing, vehicle ignition, etc.), non-stationary noise signals in speech, or images containing edges.

We make use of the IDMT-SMT drums dataset [1]. The dataset consists of single channel wav files sampled at 44.1 kHz of three different kinds of drums (snare, kick and high-hat) totalling a duration of about 2 hours. Each file contains several well-separated repetitions of a single drum strike. Segmenting all these recordings gives us about 1765 single drum strike sounds in all (about 650 for the kick and high-hat and about 450 for the snare drum). Each such sound is between 1 and 2 seconds long. We set aside about 25% of the recordings

as the validation set and use the remaining for training.

4 Experiments

Autoencoders are neural networks used for learning low-dimensional representations of their inputs. The target outputs for an autoencoder are usually the inputs themselves, making the whole structure similar to a reconstruction filter bank. The model is trained to minimize the error in reconstruction along with optional regularizing constraints. The general architecture of an autoencoder is shown in Figure 2. The encoder and decoder layers are designed to perform inverse operations, and can either consist of fully-connected or convolutional layers. The input and outputs may also be one or two dimensional. The latent variable z provides a compressed representation of the input that could be further processed before passing on to the decoder. This is similar to modifying the wavelet coefficients (e.g., applying a threshold or adding a gain) before resynthesizing the input. We therefore see a good amount of similarity between the structures of the autoencoder and a reconstruction filter bank.

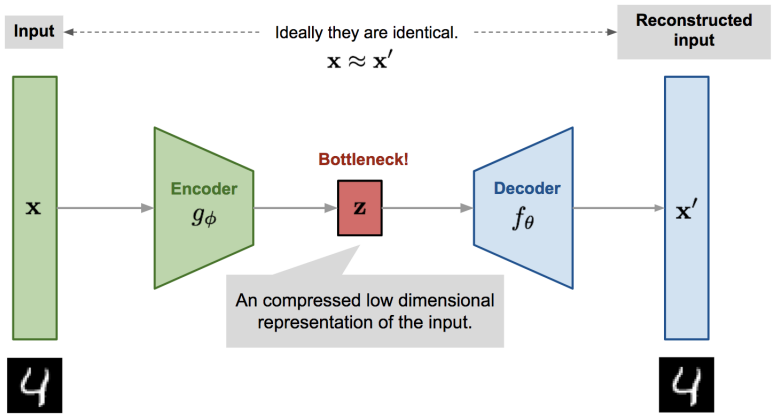


Figure 2: The general form of an autoencoder model [2]

In this work, we examine two kinds of autoencoders:

1. Fully-connected layers in the encoder and decoder (FC-AE)
2. 1-D convolutional and maximum pooling layers in the encoder and decoder (CNN-AE)

The FC-AE is the simplest form of an autoencoder, implemented here as a sequence of 4 layers - input, 2 hidden and output. The layers are fully-connected and contain 22050, 8000, 2048, and 512 neurons with ReLU activations respectively, in the encoder. The decoder is the same but with the sequence of layers mirrored. Because it is unwieldy to present an entire signal as input to the

model (which would require an exceptionally large number of input neurons), we feed in 0.5 second chunks (22050 samples) of a drum recording to the encoder at every iteration and move along the recording with no overlaps. This is similar to passing short windows of an input signal to a perfect reconstruction filter bank (except for the non-linearity). The loss function minimized is the mean squared error (MSE) loss between the input and reconstructed 0.5 second chunks. The model is trained for 200 epochs using the Adam optimizer with an initial learning rate of 0.001.

The CNN-AE has the following architecture. A 1-D convolutional layer with N filters each of length L and a max pooling layer of kernel length M form the encoder, and a max un-pooling layer and a transpose convolutional layer of the same dimensions (as the encoder layers) form the decoder. The architecture therefore resembles an N-channel analysis-synthesis filter bank, with L-length FIR filters, and down- and upsampling factors of M. We experiment with three values for N - 1, 2 and 10, and three values for L - 32, 256 and 1024. The case with $N = 1$ is not entirely meaningful and is only included as a means to verify the model operation, and to see what the learned filter appears like (should resemble an all-pass filter). The value of M is always set to be equal to N. The input is an entire audio signal containing a single drum hit sound, with which the convolutional layer filters are convolved (similar to a regular convolution operation). The only learnable parameters are the N filters in each of the convolutional layers of the encoder and decoder. The model is trained by minimizing the mean squared error (MSE) loss between the input signal and the output of the decoder (re-synthesized signal). Training is carried out for a maximum of 300 epochs using the Adam optimizer with an initial learning rate of 0.001.

In the multi-resolution analysis design method of a perfect reconstruction 2-band filterbank, certain conditions are imposed on the analysis and synthesis filters, with only the low-pass analysis filter being designed independently. These are given in Equation 1.

$$H_1(z) = z^{-(L-1)} H_0(-z^{-1}) \tag{1a}$$

$$G_0(z) = \pm H_1(-z) \tag{1b}$$

$$G_1(z) = \mp H_0(-z) \tag{1c}$$

To effect a similar structure on the filters of the CNN-AE we employ the first of these constraints - the two analysis filters being conjugate quadrature, as a regularizing term on the learned weights of the model's encoder and decoder filters, for the case with $N = 2$. This changes the loss function to that given in Equation 2 (Equation 2b applies similarly to $w_{de,1}$ as well)

$$\mathscr{L}(x, x') = \frac{1}{S}||x - x'||_2^2 + \frac{1}{L}||w_{en,0} - \hat{w}_{en,1}||_2^2 + \frac{1}{L}||w_{de,0} - \hat{w}_{de,1}||_2^2 \tag{2a}$$

$$\hat{w}_{en,1}[l] = (-1)^l w_{en,1}[L-l] \tag{2b}$$

where *x, x'* are the input and output signals, S is the length of the input, L is the filter length, and $w_{en,0}$, $w_{en,1}$ & $w_{de,0}$, $w_{de,1}$ are the two L-length convolutional layer filters of the encoder and decoder respectively. While this regularization term is easy to implement for the case with two filters, it is not so straightforward for ten filters, in which case we try to simply impose an orthogonality constraint between the filters as given in Equation 3.

$$\mathscr{L}(x, x') = \frac{1}{S}||x - x'||_2^2 + \frac{1}{N^2}||W_{en}W_{en}^T - I||_2^2 + \frac{1}{N^2}||W_{de}W_{de}^T - I||_2^2 \tag{3}$$

where W_{en}, W_{de} are the encoder and decoder weight matrices of dimensions N x L and I is an NxN identity matrix.

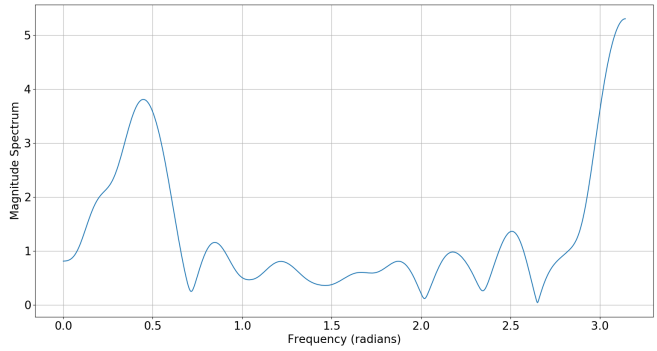
We then evaluate the trained models by observing the magnitude spectra of the encoder and decoder filters, and perceptually assessing the quality of reconstruction for an audio sample in the validation set. These are presented next.

5 Results and Discussion

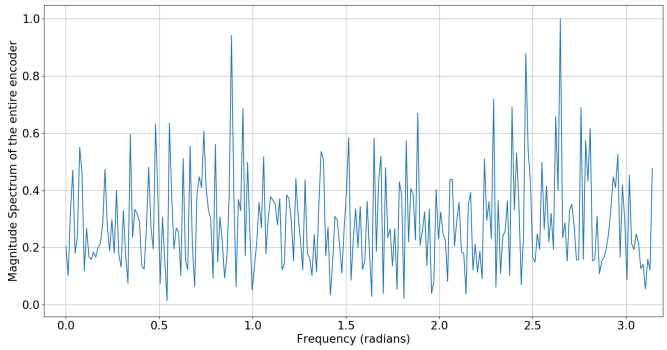
The FC-AE performs very poorly in reconstructing the original sound (an example can be found here[1]). Due to the inputs being only 0.5 seconds long and with no additional temporal structure imposed between successive 0.5 second chunks from the same signal, the reconstruction produces drum bursts in all the chunks, even in those that do not contain any. This could be because the overall MSE loss gets naively minimized by catering more to the energetic chunks containing drum strikes, with the quieter chunks end up not having enough of an effect on the loss. Although individual filters in a specific FC-AE layer have curious band pass characteristics (Figure 3a), the overall encoder's impulse response (Figure 3b) is not easy to interpret. There is also no interaction between any of the filters within the same layer in the encoder or decoder. This is precisely why CNNs are used - to provide temporal structure by using the same filter across the input.

The CNN-AE fares much better and manages to reconstruct the input quite well as shown in Figure 4 (audio examples provided here[2]). The encoder and decoder filter magnitude spectra with only the MSE loss are shown in Figure 5 and with the regularized loss in Figure 6, for the case with L=256. The responses are plotted with offsets to improve readability in Figures 5 and 6. We see that with a single filter, the reconstruction has faint vertical bursts of energy towards the end, and the filters do resemble an all pass response, but with some

[1] Audio examples: https://bit.ly/2MGegwN
[2] Audio examples: https://bit.ly/2UrVNbj



(a)



(b)

Figure 3: The magnitude spectra of (a) a filter from one of the layers in the encoder and (b) the impulse response of the entire encoder.

small, abrupt peaks and dips throughout. With two filters, the reconstruction is slightly better but the filters do not really have any interesting nature and both the filters in the encoder and decoder are similar. With ten filters, while there are a couple of filters with similar responses, each of the rest seem to capture a different band of the frequency spectrum but not in any well-structured manner. The reconstruction spectrogram also seems to have bands of lesser stationary noise (present throughout in the other two cases). With the regularized loss, we see that in the case with two filters, the filters indeed end up being complementary. The reconstruction however ends up being noisier and in particular having a constant band of more energetic noise close to and below the cut-off frequency. And with ten filters, the regularization does not help as much in bringing a better defined structure. Similar results were observed for the cases with $L = 32$ and 1024, with the filter magnitude spectra being smoother and noisier respectively.

All the CNN-AE model variants above have fairly simple architectures and do not possess high capacity. A good reconstruction despite that is perhaps

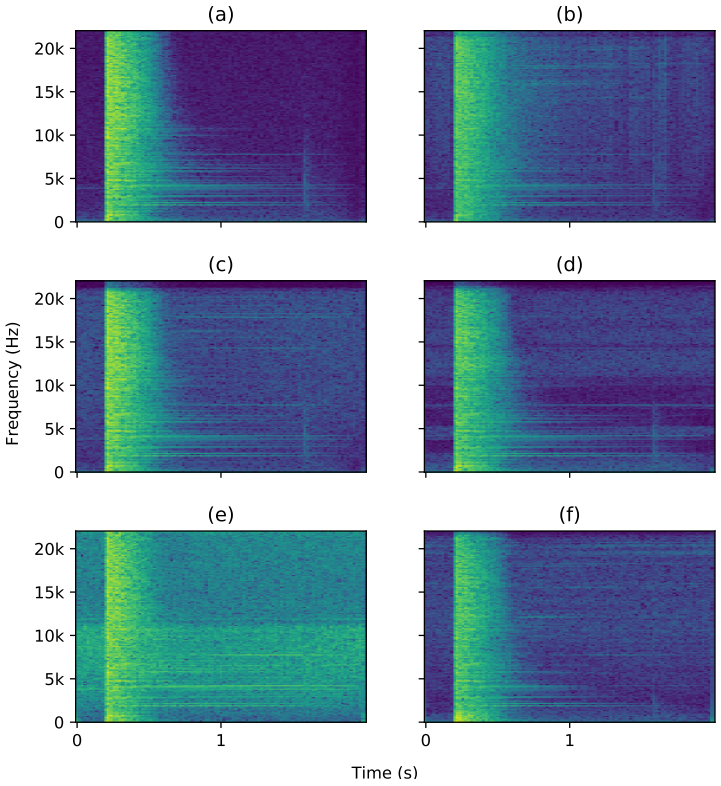
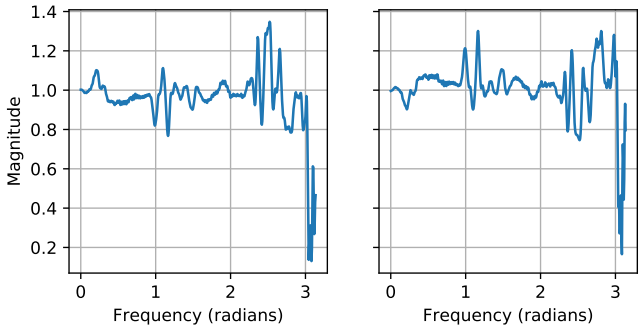
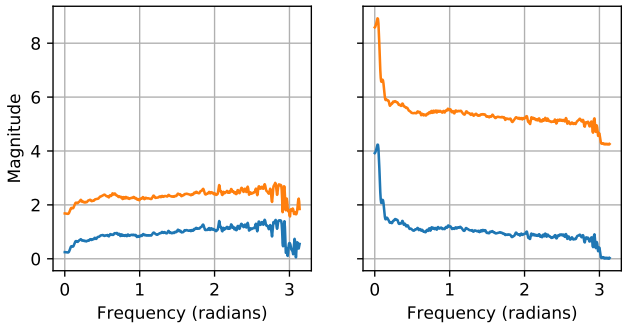


Figure 4: Magnitude spectrograms of the original and CNN-AE reconstructions of an audio sample in the validation set. (a)Original (b) N=1 (c) N=2 (d) N=10 (e) N=2 with regularised loss (f) N=10 with regularised loss.

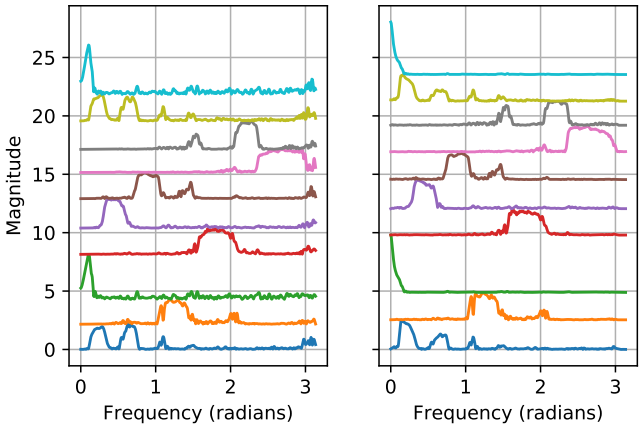
simply because the encoder does not perform any significantly lossy operations (the max-pooling is lossy but perhaps not extremely because of the small pooling sizes). The goal of the preliminary investigation carried out in this work was to simply establish how an autoencoder model resembles a reconstruction filterbank and how the learned filters can be constrained to have properties that come naturally in the classical filterbank design approach. This makes the model more interpretable. However, the more important next step is to utilise this model to perform a more compact encoding, on which more useful, and possibly non-linear operations can be performed. This could be achieved by adding a few fully connected layers at the center of the autoencoder to compress the signal after the analysis and downsampling, and then either reconstructing the input, or using the compressed representation as a feature vector to perform some classification tasks.



(a)

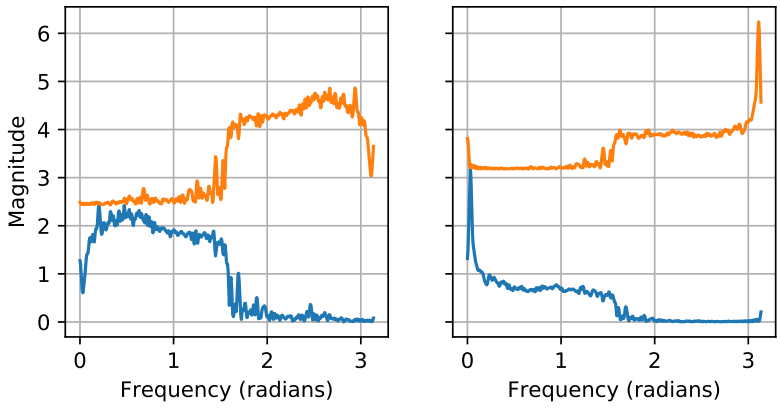


(b)

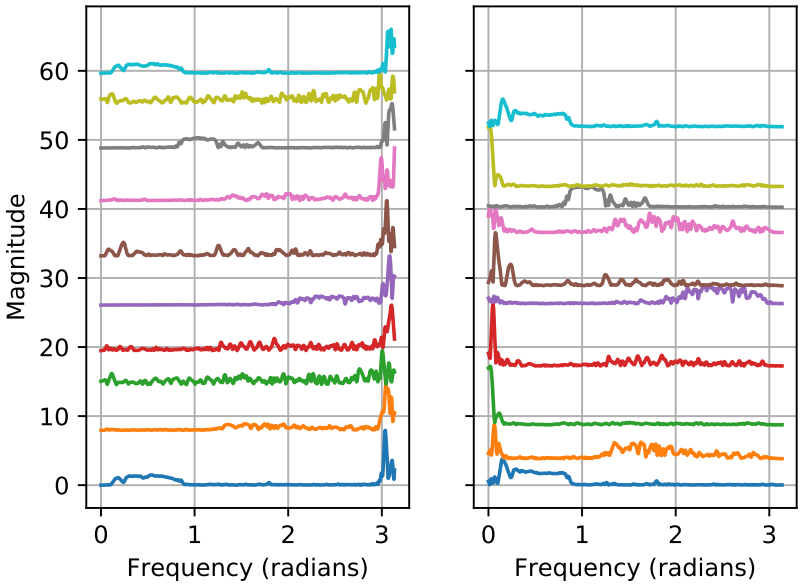


(c)

Figure 5: The encoder(left) and decoder(right) filter magnitude spectra for different cases (a) N=1 (b) N=2 (c) N=10



(a)



(b)

Figure 6: The encoder(left) and decoder(right) filter magnitude spectra with regularized losses (a) N=2 (b) N=10

6 Conclusions

In this work we have first presented a background of the applications of wavelet-based methods for the analysis of transient structures particularly in the case of audio signals. With the larger goal of addressing the common problem of choosing the best wavelet for a given task, we set out to devise ways to learn the filters to better suit the data and task at hand. We achieved this using an adaptive filterbank implemented as an autoencoder model. We observed that a fully connected model performs poorly and also does not have a form that is similar to a FIR filterbank. The convolutional model however is not only more intuitive, but also performs the reconstruction quite well. Further, by regularizing the reconstruction loss with constraints on the weights based on principles from the classical approach, we find that the filters indeed end up possessing properties we desire. However, we note that a mere signal reconstruction model is of no real use if the analysis does not result in a transformation that makes signal modification or identification easier. Future work would therefore involve trying to compare this transformation more objectively with a traditional approach by evaluating the performance on a task like classifying the presented input sound into one of a few categories, using the analysis coefficients.

References

[1] Drums - Fraunhofer IDMT . https://www.idmt.fraunhofer.de/en/business_units/m2d/smt/drums.html.

[2] From autoencoder to beta-vae. https://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html.

[3] Jesse Engel, Cinjon Resnick, Adam Roberts, Sander Dieleman, Mohammad Norouzi, Douglas Eck, and Karen Simonyan. Neural audio synthesis of musical notes with wavenet autoencoders. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1068–1077. JMLR. org, 2017.

[4] Haidar Khan and Bulent Yener. Learning filter widths of spectral decompositions with wavelets. In *Advances in Neural Information Processing Systems*, pages 4601–4612, 2018.

[5] Richard Kronland-Martinet. The wavelet transform for analysis, synthesis, and processing of speech and music sounds. *Computer Music Journal*, 12(4):11–20, 1988.

[6] Panos E Kudumakis and Mark B Sandler. Synthesis of audio signals using the wavelet transform. In *IEE Colloquium on Audio DSP-Circuits and Systems (Digest No. 1993/219)*, pages 4–1. IET, 1993.

[7] Stéphane Mallat. *A wavelet tour of signal processing*. Elsevier, 1999.

[8] Grégoire Montavon, Wojciech Samek, and Klaus-Robert Müller. Methods for interpreting and understanding deep neural networks. *Digital Signal Processing*, 73:1–15, 2018.

[9] Deepen Sinha and Ahmed H Tewfik. Low bit rate transparent audio compression using adapted wavelets. *IEEE Transactions on signal processing*, 41(12):3463–3479, 1993.

[10] S. R. Subramanya, R. Simha, B. Narahari, and A. Youssef. Transform-based indexing of audio data for multimedia databases. In *Proceedings of IEEE International Conference on Multimedia Computing and Systems*, pages 211–218, 1997.

[11] George Tzanetakis, Georg Essl, and Perry Cook. Audio analysis using the discrete wavelet transform. In *Proc. Conf. in Acoustics and Music Theory Applications*, volume 66, 2001.

[12] George Tzanetakis, Ajay Kapur, and Richard I McWalter. Subband-based drum transcription for audio signals. In *2005 IEEE 7th Workshop on Multimedia Signal Processing*, pages 1–4. IEEE.

[13] Shrikant Venkataramani, Jonah Casebeer, and Paris Smaragdis. Adaptive front-ends for end-to-end source separation. In *31st Conference on Neural Information Processing Systems (NIPS)*, 2017.