Learnable filter-banks for CNN-based audio applications

Helena Peic Tukuljac[1], Benjamin Ricaud*[1,2], Nicolas Aspert[1], and Laurent Colbois[3]

[1]LTS2, École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland
[2]Dept. of Physics and Technology, UiT The Arctic University of Norway, Tromsø, Norway
[3]Idiap Research Institute, Martigny, Switzerland

Abstract

We investigate the design of a convolutional layer where kernels are parameterized functions. This layer aims at being the input layer of convolutional neural networks for audio applications or applications involving time-series. The kernels are defined as one-dimensional functions having a band-pass filter shape, with a limited number of trainable parameters. Building on the literature on this topic, we confirm that networks having such an input layer can achieve state-of-the-art accuracy on several audio classification tasks. We explore the effect of different parameters on the network accuracy and learning ability. This approach reduces the number of weights to be trained and enables larger kernel sizes, an advantage for audio applications. Furthermore, the learned filters bring additional interpretability and a better understanding of the audio properties exploited by the network.

1 Introduction

In audio signal processing, time-frequency representations such as spectrograms are central tools. They have an intuitive interpretation and reveal insightful information to the human expert. It is not a surprise that many deep learning approaches to audio signals use such representations as well [5, 26]. It is also convenient as most of the deep network architectures have been developed for image processing and require 2D arrays of values as inputs. The network learns to detect time-frequency patterns, similarly to what is done on images. Depending on the task, it may then output a classification of a sound [25, 28], a denoised signal [15] or separated sources [4].

These representations are conventionally made using several types of transformations. In turn, each transformation may have several parameters that influence the representation. Until recently these transformations and their parameters were carefully chosen using expert knowledge.

The recent success of end-to-end learning where the raw audio file is the input of the network (e.g. Wavenet: [18, 22, 19, 30], Tasnet: [16, 17]), and more recently LEAF [34], demonstrates the efficiency of this approach for a variety of audio tasks. In this setting, one-dimensional convolutions are applied to raw audio signals and the network creates its own representation by learning the convolution kernels. However, kernel size needs to be much larger than the one used for image applications. Indeed, at a sampling rate of 44kHz, 44 samples represent 1 ms of audio signal. To capture audio patterns that have duration of 10, 100 ms or more, in particular low frequency patterns, either large kernels are needed or deeper convolutional architectures (to allow for combinations of kernels at many different positions in time). Both solutions lead to a large increase in the number of parameters to be learned and hence require more training time and more data. Dilated convolutions or "atrous" convolution employed in Wavenet have been introduced in order to increase the time length of the kernel without increasing the number of weights to learn. Finding alternative ways for unlocking the time-length limit is an important challenge for raw audio processing in deep learning.

Replacing free kernels by parameterized filters,

*Corresponding Author: benjamin.ricaud@uit.no

https://doi.org/10.7557/18.6279

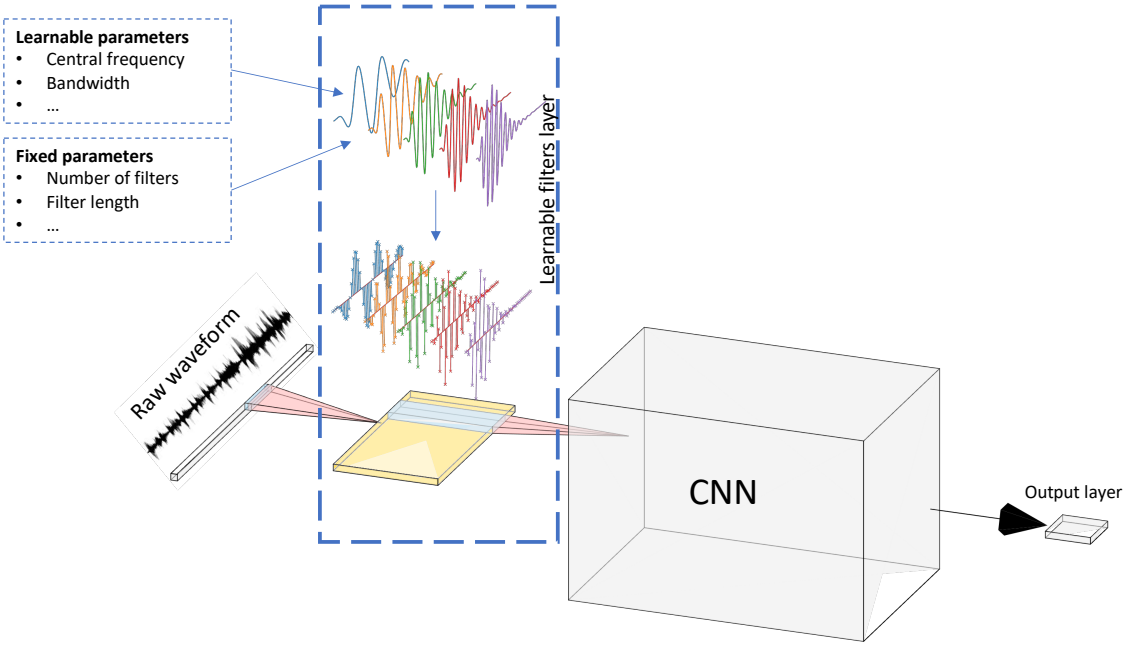

Figure 1: Network architecture using learnable filters. The first layer (in yellow) is a convolution layer where the kernels are defined as functions (colored waveforms) with learnable parameters, such as frequency and bandwidth. This functions are defined on a continuous domain and are then discretized to fit in the convolutional layer. The rest of the convolutional neural network (CNN) can have any standard architecture.

were the parameters are learnt, is an alternative way for reducing the computational burden. This is what we propose to investigate in the present work. The free kernels are replaced by filters with a few parameters in the first layer of the network, as shown in Fig. 1.

Learning parametric filters is halfway between 1) learning a standard convolutional layer, where all the weights of the kernels are learnable and unconstrained and 2) having a layer of kernels being fixed functions, where only the combination of these predefined functions may be learnt. The first approach is the most versatile but is computationally intensive and more prone to overfitting. The second approach used for example in the Scattering transform [3, 1, 6], or in [11] benefits from an inductive bias through the chosen kernel functions but is less flexible. The concept of learning filters aims at making an ideal compromise between flexibility and inductive bias. It has been first introduced in [29], [27], [33] and [12]. The first one introduces Gaussian filters in the input layer. Parameters are the amplitude, the Gaussian width and the modulation frequency. An increase of the classification accuracy is reported with the learned parameters. However, the filter learning is seen as a fine-tuning of the network after the first training pass with fixed Gaussian parameters. In the present work, the filter layer is fully integrated in the learning process, the parameters are learned from the beginning. In [27], the authors introduce a layer, called *SincNet*, made of sine modulated functions that approximate band-pass rectangular windows in the frequency domain. The learned parameters are the minimal and maximal cut-off frequencies of each band-pass filter. One of the main results is given by the cumulative frequency response of the SincNet filters. The network tends to focus more on particular regions of the frequency space, where formants are localized. This is interesting, as it shows

how the parameterized filters enable a precise interpretation of the learning and underline particular spectral properties of the data. The present work goes further in this direction. Eventually, [12] introduce Wavelet filter banks learned for speech recognition. Each kernel is a Wavelet defined by a single parameter, its scale. It shows evidences both of the efficiency of this approach and of the possibility to interpret the shape of the learned kernels. We compare the efficiency of the Wavelet filters with several other modulated windows and show that the former under-performs on audio signals. More recently and in line with our approach, [14] present complementary results, on a different dataset, with a focus on the sinc-square function, learning either the frequency or the bandwidth of the filters. Learnable Gabor functions combined with a modulus layer and a learnable PCEN layer showed state-of-the-art performance [34]. Comparing the effect of replacing a standard convolutional layer by a set of gammatone filters, [8] show an increase in the accuracy of a speech separation task. This suggests that an hybrid approach of learned gammatone filters would combine the best of both worlds.

We propose several parameterized functions and compare them to recent works on the same topics that use learnable filters. We confirm that this approach reaches state-of-the-art accuracy and even improves the accuracy on several audio classification tasks. We explore the influence of different parameters on the learning, such as the numbers of kernels and their length. Our classification experiments show that the number of filters required to obtain the best results remains small, around 20-30. We also demonstrate that the performances of different functions proposed in audio signal processing (modulated Gaussian, Gammatone) give close results and are better than Wavelets at classifying sounds. Last but not least, a relationship between the central frequency of the filter and its temporal width emerges with the learning. We provide evidences that the network converges to an auditory frequency spacing, close to the ERB (Equivalent Rectangular Bandwidth) and Bark scales found in psycho-acoustic studies [35, 9].

2 Learnable filter banks

We call the parameterized kernels in the convolutional layer *filters*, making a parallel with filters in signal processing. Indeed, these functions have the property of being band-pass filters and are well known in audio signal processing. One of the trainable parameters of each filter is the central frequency of the band-pass filter. The second parameter is the bandwidth of the filter (or a quantity closely related to it). Hence this set of filters forms a filter bank where the frequency and bandwidth of the filters may be adapted to the data and to the learning task. Note that the learned filterbank may not cover the entire spectrum but should focus on important spectral regions that are the most discriminative for classification.

We call the convolutional layer made of learnable filters, Learnable Filter (LF) layer. The input of the LF layer is a 1D audio signal and the output is a 2D representation. The output representation axes are time and filter number. Since each filter is associated to a particular frequency band, this 2D representation can be seen as a time-frequency one (or time-scale in the case of Wavelets). Initializing the filters by increasing frequencies (or scales), we can influence the frequency ordering to follow the filter number.

In all the definitions, N denotes the filter length and n is the variable (sample number). The time in seconds is expressed using the sampling frequency f_s with $t = n/f_s$. The frequency in Hertz is defined by $f \times f_s$, where $f \in [0, 0.5]$ is the normalized frequency in the formulas.

Mexican hat Wavelet. In order to compare to the state-of-the-art, we use the Mexican hat Wavelet introduced in the paper by [12]:

$$w(n) = \frac{2}{\pi^{1/4}\sqrt{3s}} \left(\frac{n^2}{s^2} - 1 \right) e^{-\frac{n^2}{s^2}}, \qquad (1)$$

with $n \in [-N/2, (N-1)/2]$ and $s > 0$ being the scale parameter.

Gaussian filter. Here, $n \in [-N/2, (N-1)/2]$. The Gaussian filter, also used in [29, 34], g is defined as follows:

$$g(n) = \sqrt{\frac{2}{\sqrt{\pi}\sigma}} e^{-\frac{n^2}{2\sigma^2}} \left(\cos(2\pi f n) + i \sin(2\pi f n) \right). \qquad (2)$$

The parameter $\sigma > 0$ is the variance of the Gaussian (temporal window width) and f is the oscillating frequency. It is a complex-valued function that we split into its real and imaginary parts. For each f and σ two kernels are created, one with the cosine modulation and one with the sine one.

Gammatone filter. The Gammatone filter [7, 24, 10] is another example of kernel. It is defined on the interval $n \in [0, N-1]$ as :

$$h(n) = A(\gamma, b) n^{\gamma-1} e^{-2\pi b n} \cos(2\pi f n) \qquad (3)$$

where A is the normalization[1], $A(\gamma, b) = \sqrt{2(4\pi b)^{(2\gamma+1)}/\Gamma(2\gamma+1)}$. The parameter γ is the order of the Gammatone. It can be learned or fixed to e.g. 2 or 4. These two latter values are the best suited ones for modeling the human hearing related filter bank [23]. The other learnable parameters are b, related to the width of the function, and f the frequency. The symbol Γ denotes the Gamma function. The bandwidth B of h depends linearly on b and is given by the following formula [7]:

$$B(\gamma, b) = 2(2^{1/\gamma} - 1)^{1/2} b. \qquad (4)$$

Remark 1: All the functions are defined and normalized in the continuous domain. In our application, the filters are discretized and truncated in order to be implemented in the convolution layer. Since they all vanish away from zero, it remains a good approximation, provided that the function's width does not exceed the fixed filter length N.

Remark 2: The modulated window functions are defined with a cosine (real part) and a sine (imaginary part) term, relating them to the Fourier transform, the spectral domain and the standard definition of filters in signal processing. For the sake of simplicity, in our experiments, we have chosen to use only the cosine term. The absence of the sine term did not affect the accuracy of our classification results. The network is able to adapt and detect discriminative patterns with a shifted cosine modulation.

Remark 3: It is important to distinguish the filter length N from the filter temporal width σ or b (or s for the scale). The filter length is fixed, can not be learned and is the size of the vector on which the filter is defined. The temporal width is learned and specifies the spread of the function over the vector of size N.

[1]This is an approximation of the normalization obtained by computing the integral of the continuous function $t^{\gamma-1}e^{-2\pi bt}$, using the following result: $\int_0^\infty t^n e^{-bt} dt = \frac{\Gamma(n+1)}{b^{n+1}}$.

3 Experiments

We apply our LF layer to several classification tasks described in the following sub-sections. We assess it on standard tasks found in the literature presented in the introduction. We have chosen 2 freely available speech datasets: *AudioMNIST* [2] and *Google Speech Commands v2* [32]. Both datasets contain words pronounced by different speakers. These datasets are dedicated to limited-vocabulary speech recognition tasks and the goal is to train the network to correctly recognize the word present in each audio sequence.

In order to compare the impact of the LF layer on the learning and classification results, we use existing network architectures and modify the first layer. For networks with raw audio input, the first convolutional layer (performing a standard 1D convolution) is replaced by our proposed parameterized convolution layer, as illustrated in Fig. 1. Our layer is then followed by a non-linear ReLU activation function. A stride parameter is available allowing to define the overlap in time of consecutive convolutions. The code needed to reproduce the experiments is publicly available on GitHub[2].

3.1 AudioMNIST Results

The original AudioMNIST paper [2] performs digit classification using raw audio as input to a network called AudioNet. The code[3] supplied with the paper has been re-used to perform 5-fold validation on the data. AudioNet is made of six convolutional layers, each convolution being followed by a max-pooling layer, and two dense layers, connected to an output layer. In all tests performed using this dataset, the models were trained using the Adam optimizer with default parameters during 50 epochs. Batch size used was set to 256 and loss function used was the categorical cross-entropy.

[2]`https://github.com/epfl-lts2/learnable-filterbanks`

[3]`https://github.com/soerenab/AudioMNIST`

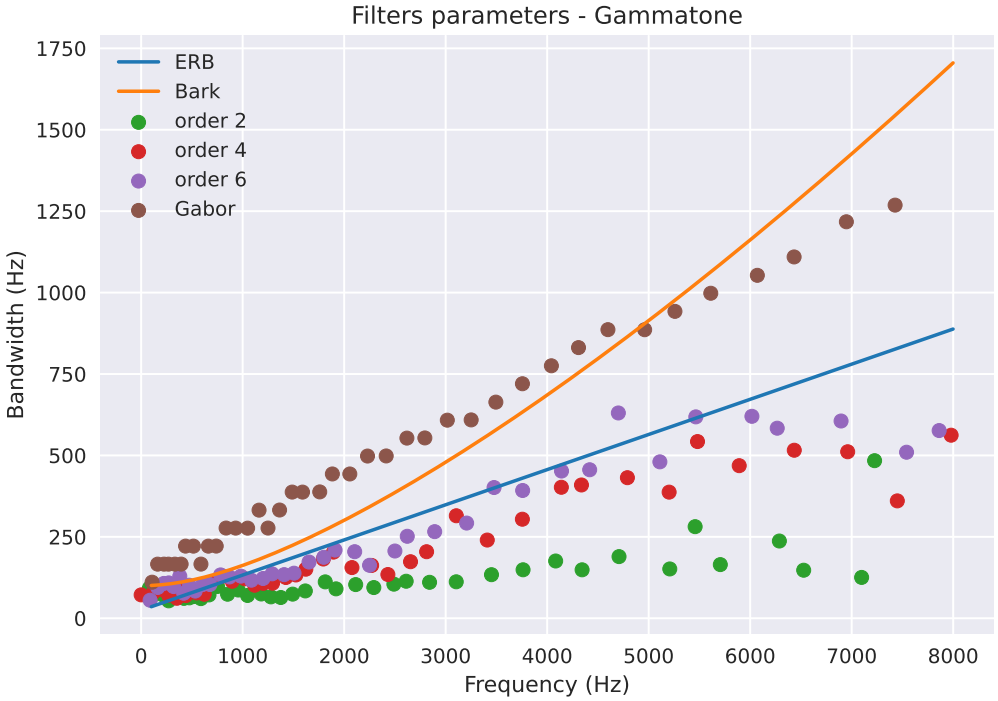

Figure 2: Bandwidth and frequency of learned filters.The curves are the psycho-acoustical relationships given by the ERB and Bark scales.

Test accuracy was then computed after this training phase and the same process was repeated for each fold.

On the AudioMNIST dataset sampled at 8 kHz, AudioNet has ca. 17 million trainable parameters. The original paper from [2] claims an accuracy of 92.53%±2.04%, whereas our implementation of AudioNet using Keras and Adam optimizer (instead of SGD in the original paper) yields an average accuracy of 94.9%±1.54%, which is already a significant improvement. We performed the same 5-fold validation using a modified version of AudioNet where the first convolutional layer is replaced by a LF layer. This layer consists in 32 4th-order Gammatone filters of length 80 (corresponding to 10 ms at 8 kHz). The stride has been set such that the overlap between two consecutive convolution steps is equal to 75%. In this modified network, the number of trainable parameters drops to ca. 3.5 million trainable parameters, i.e. a reduction in size by a factor 5. Using the LF-enabled AudioNet the average accuracy increases to $96.8\% \pm 1.22\%$.

Another LF-enabled network was used to perform the classification task on AudioMNIST. The architecture is derived from the raw waveform model *SampleCNN* introduced in [13]. Despite its much smaller number of trainable parameters (ca. 300'000), its average accuracy improves to $98.0\% \pm 0.41\%$. For the sake of completeness, we also trained this network, replacing the Gammatone filters by the learned wavelets as in [12], and the learned SincNet filters from [27]. A summary of all results achieved using AudioMNIST can be found in Table 1.

3.2 Google Speech Command

The Google Speech Command dataset [32] provides similar data to the AudioMNIST one, with a larger number of classes (35). As done in [34] and its ac-

Table 1: AudioMNIST mean test accuracy

Network	# Trainable parameters	Avg. accuracy
AudioNet	17 M	$94.9\% \pm 1.54\%$
LF-AudioNet	3.5 M	$96.8\% \pm 1.22\%$
LF-custom (Gammatone)	**300 k**	$\mathbf{98.0\% \pm 0.41\%}$
LF-custom (SincNet)	300 k	$97.2\% \pm 1.0\%$
LF-custom (Wavelet)	300 k	$89.9\% \pm 1.18\%$

companying code, we used the pre-defined dataset from Tensorflow which reduces the number of labels to 12, by merging a number of samples into an *unknown* class.

Given that Google Speech Commands does not possess pre-defined folds for n-fold validation, the experiments were repeated 3 times in order to compute the mean accuracy. In [34], the authors train a learnable parametric frontend similar to the one introduced in this paper. Their framework, called "LEAF", consists in a frontend, a convolutional network, and a final layer adapted to the number of classes in the dataset. The frontend is made of a learnable Gabor filter bank, a learnable pooling function, and a learnable smooth compression function. We reproduced the experimental setting from [34], using a frontend made of 40 order 4 Gammatone filters, overlapping by 80% and having a length representing 25 ms. In one experiment, we did not use the learnable pooling and compression methods present in LEAF and in the other we did use the complete LEAF pipeline. The convolutional network, based on EfficientNet-B0 [31], had been trained using the Adam optimizer during 30 epochs with batches of 128 and 256 samples, and using learning rate reduction on plateaus. The resulting network has ca. 3.5 million trainable parameters. The test accuracy from [34] using the complete LEAF model with Gabor filters is $\mathbf{93.4\% \pm 0.3}$. In our experiments, we observed that using Gammatones over the complete LEAF pipeline lead to results very close to the ones achieved with Gabor filters, i.e. ca. 93% of test accuracy. Using the simpler version without learnable pooling and compression, test accuracy improves to $\mathbf{94.31\% \pm 0.1}$, when using batches of 128 samples.

3.3 Properties of learned filters

The learned parameters of the LF filters can reveal insights about the data and the learning process. As stated in the introduction, several studies have shown a tendency governing the spacing in frequency of their learned kernels. The spacing becomes exponentially large as the frequency increases, following what is called a Mel scale [21]. This is in agreement with psycho-acoustics tests on the human cochlear system. In order to go further in this direction, we investigate 1) the frequency spacing and 2) we test the relationship between the temporal width of the filters and their central frequency. Indeed, psycho-acoustic models (the equivalent rectangular bandwidth (ERB) model [9] and the Bark model [35]) provide such a relationship. This is made possible by our approach where the temporal width as well as the filter central frequency are well defined for each filter.

Bandwidth and frequency. The learned filter banks can be compared to filter banks modeling the human auditory system. Two main models can be found in the literature, the Equivalent Rectangular Bandwidth (ERB) model [9], and the Bark model [35]. The ERB and Bark curves are plotted on Fig. 2, together with the learned parameters of the Gammatone filters initialized with different orders, and the Gabor filters. All the filters have been trained using the LEAF network and the Google Speech dataset, used in section 3.2. We observe a good agreement between the ERB curve and the learned Gammatone filters of order 4 and 6. The agreement is even stronger below 2 kHz. Gammatone filters of order 2 and Gabor filters do not exhibit this behavior and do not follow neither the ERB, nor the Bark curves, while keeping a similar test accuracy on the Google speech commands

dataset. In [27], the authors show that for a neural network applied to a speech dataset, the focus of the learning is situated around the pitch frequency located at 130Hz (male) and 230Hz (female), and the first and second formants (i.e. resonances of the vocal tract [20]), which are around 500Hz and 1kHz respectively. This is exactly the frequency region where our learned filters match the ERB scale.

Frequency spacing. To show the importance of the frequency spacing, we initialized the LF layer with a linear frequency spacing from 0 to the Nyquist frequency. After the learning phase, the filter frequencies evolved and moved away from their initial value as can be seen on Fig. 3. The frequency distribution is not exponential (as in the case of the Mel scale) but we can point out several interesting facts. Firstly, the final curve is flatter than the initialization in the range 0-2kHz (more filters in this range). It shows that the network tends to favour filters with a band-pass in this range for its discriminative process. Secondly, beyond 4kHz, the filters stay close to their original value. This suggests that there is not enough meaningful information in this frequency range for a correct learning. This is indeed the case for speech dataset where we found that the main information resides below 4kHz.

Figure 3: Frequency distribution of the filters before (straight line) and after training (green curve)

4 Conclusion

Decades of research in audio signal processing have brought us an important knowledge about sounds, speech and audio information. This knowledge may be inserted within neural networks as a priori information and turned into efficient inductive biases. This is what we show with the example of the LF layer, a layer of parameterized filters adapted to the extraction of audio information. Moreover, the trained network possesses properties than can, in turn, bring new insights about audio data back to the audio signal processing community. For example, the optimal relationship between frequency and bandwidth seems to be influenced by the envelop shape in a non-trivial manner.

Future work in this direction and further developments of convolutions with parameterized functions may lead to important progress both in deep learning and audio signal processing. The reduction of the number of trainable parameters decreases the network complexity, along with the training time. It also enables a better interpretation of the network adaptation to the data.

References

[1] J. Andén and S. Mallat. Deep scattering spectrum. *IEEE Transactions on Signal Processing*, 62(16):4114–4128, 2014. doi:10.1109/TSP.2014.2326991.

[2] S. Becker, M. Ackermann, S. Lapuschkin, K. Müller, and W. Samek. Interpreting and explaining deep neural networks for classification of audio signals. *CoRR*, abs/1807.03418, 2018. URL `http://arxiv.org/abs/1807.03418`.

[3] J. Bruna and S. Mallat. Invariant scattering convolution networks. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1872–1886, 2013. doi:10.1109/TPAMI.2012.230.

[4] P. Chandna, M. Miron, J. Janer, and E. Gómez. Monoaural audio source separation using deep convolutional neural networks. In *International conference on latent variable analysis and signal separation*, pages 258–266. Springer, 2017. doi:10.1007/978-3-319-53547-0_25.

[5] K. Choi, G. Fazekas, K. Cho, and M. Sandler. A tutorial on deep learning for music information retrieval. *arXiv preprint arXiv:1709.04396*, 2017.

[6] F. Cotter and N. G. Kingsbury. A learnable Scatternet: Locally invariant convolutional layers. *2019 IEEE International Conference on Image Processing (ICIP)*, pages 350–354, 2019. doi:10.1109/ICIP.2019.8802977.

[7] A. Darling. Properties and implementation of the gammatone filter: a tutorial. *Speech Hearing and Language, Work in Progress, University College London, Department of Phonetics and Linguistics*, pages 43–61, 1991.

[8] D. Ditter and T. Gerkmann. A multi-phase gammatone filterbank for speech separation via TasNet. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 36–40. IEEE, 2020. doi:10.1109/ICASSP40776.2020.9053602.

[9] B. R. Glasberg and B. C. Moore. Derivation of auditory filter shapes from notched-noise data. *Hearing research*, 47(1-2):103–138, 1990. doi:10.1016/0378-5955(90)90170-T.

[10] V. Hohmann. Frequency analysis and synthesis using a gammatone filterbank. *Acta Acustica united with Acustica*, 88(3):433–442, 2002.

[11] J.-H. Jacobsen, J. van Gemert, Z. Lou, and A. W. Smeulders. Structured receptive fields in CNNs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2610–2619, 2016. doi:10.1109/CVPR.2016.286.

[12] H. Khan and B. Yener. Learning filter widths of spectral decompositions with wavelets. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 4601–4612. Curran Associates, Inc., 2018. doi:10.5555/3327345.3327371.

[13] T. Kim, J. Lee, and J. Nam. Comparison and analysis of SampleCNN architectures for audio classification. *IEEE Journal of Selected Topics in Signal Processing*, 13(2):285–297, May 2019. doi:10.1109/JSTSP.2019.2909479.

[14] E. Loweimi, P. Bell, and S. Renals. On learning interpretable CNNs with parametric modulated kernel-based filters. In *Proc. Interspeech*, pages 3480–3484, 2019. doi:10.21437/Interspeech.2019-1257.

[15] X. Lu, Y. Tsao, S. Matsuda, and C. Hori. Speech enhancement based on deep denoising autoencoder. In *Interspeech*, pages 436–440, 2013. doi:10.21437/Interspeech.2013-130.

[16] Y. Luo and N. Mesgarani. TasNet: time-domain audio separation network for real-time, single-channel speech separation. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 696–700. IEEE, 2018. doi:10.1109/ICASSP.2018.8462116.

[17] Y. Luo and N. Mesgarani. Conv-TasNet: Surpassing ideal time–frequency magnitude masking for speech separation. *IEEE/ACM transactions on audio, speech, and language processing*, 27(8):1256–1266, 2019. doi:10.1109/TASLP.2019.2915167.

[18] A. v. d. 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.

[19] A. v. d. Oord, Y. Li, I. Babuschkin, K. Simonyan, O. Vinyals, K. Kavukcuoglu, G. v. d. Driessche, E. Lockhart, L. C. Cobo, F. Stimberg, et al. Parallel wavenet: Fast high-fidelity speech synthesis. *arXiv preprint arXiv:1711.10433*, 2017.

[20] D. O'Shaughnessy. *Speech Communications: Human and Machine*, chapter 3, page 45. Wiley-IEEE Press, 2000. doi:10.1109/9780470546475.

[21] D. O'Shaughnessy. *Speech Communications: Human and Machine*, chapter 4, pages 127–128. Wiley-IEEE Press, 2000. doi:10.1109/9780470546475.

[22] T. L. Paine, P. Khorrami, S. Chang, Y. Zhang, P. Ramachandran, M. A. Hasegawa-Johnson, and T. S. Huang. Fast wavenet generation

algorithm. *arXiv preprint arXiv:1611.09482*, 2016.

[23] R. D. Patterson, I. Nimmo-Smith, J. Holdsworth, and P. Rice. An efficient auditory filterbank based on the gammatone function. In *a meeting of the IOC Speech Group on Auditory Modelling at RSRE*, volume 2, 1987.

[24] R. D. Patterson, K. Robinson, J. Holdsworth, D. McKeown, C. Zhang, and M. Allerhand. Complex sounds and auditory images. In *Auditory physiology and perception*, pages 429–446. Elsevier, 1992. doi:10.1016/B978-0-08-041847-6.50054-X.

[25] K. J. Piczak. Environmental sound classification with convolutional neural networks. In *2015 IEEE 25th International Workshop on Machine Learning for Signal Processing (MLSP)*, pages 1–6. IEEE, 2015. doi:10.1109/MLSP.2015.7324337.

[26] H. Purwins, B. Li, T. Virtanen, J. Schlüter, S.-Y. Chang, and T. Sainath. Deep learning for audio signal processing. *IEEE Journal of Selected Topics in Signal Processing*, 13(2):206–219, 2019. doi:10.1109/JSTSP.2019.2908700.

[27] M. Ravanelli and Y. Bengio. Speaker recognition from raw waveform with SincNet. *2018 IEEE Spoken Language Technology Workshop (SLT)*, pages 1021–1028, 2018. doi:10.1109/SLT.2018.8639585.

[28] J. Salamon and J. P. Bello. Deep convolutional neural networks and data augmentation for environmental sound classification. *IEEE Signal Processing Letters*, 24(3):279–283, 2017. doi:10.1109/LSP.2017.2657381.

[29] H. Seki, K. Yamamoto, and S. Nakagawa. A deep neural network integrated with filterbank learning for speech recognition. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5480–5484, March 2017. doi:10.1109/ICASSP.2017.7953204.

[30] D. Stoller, S. Ewert, and S. Dixon. Wave-u-net: A multi-scale neural network for end-to-end audio source separation. *arXiv preprint arXiv:1806.03185*, 2018. doi:10.5281/ZENODO.1492417.

[31] M. Tan and Q. Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6105–6114. PMLR, 09–15 Jun 2019. URL `https://proceedings.mlr.press/v97/tan19a.html`.

[32] P. Warden. Speech commands: A dataset for limited-vocabulary speech recognition. *CoRR*, abs/1804.03209, 2018. URL `http://arxiv.org/abs/1804.03209`.

[33] N. Zeghidour, N. Usunier, I. Kokkinos, T. Schaiz, G. Synnaeve, and E. Dupoux. Learning filterbanks from raw speech for phone recognition. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5509–5513. IEEE, 2018.

[34] N. Zeghidour, O. Teboul, F. de Chaumont Quitry, and M. Tagliasacchi. LEAF: A learnable frontend for audio classification. *ICLR*, 2021.

[35] E. Zwicker and E. Terhardt. Analytical expressions for critical-band rate and critical bandwidth as a function of frequency. *The Journal of the Acoustical Society of America*, 68 (5):1523–1525, 1980. doi:10.1121/1.385079.