Basic Filters for Convolutional Neural Networks: Training or Design?\footnote{This work has been supported by the Vienna Science and Technology Fund (WWTF) through project MA14-018.}

M. Dörfler, T. Grill, R. Bammer, A. Flexer
monika.doerfler@univie.ac.at

Abstract: When convolutional neural networks are used to tackle learning problems based on time series, e.g., audio data, raw one-dimensional data are commonly pre-processed to obtain spectrogram or mel-spectrogram coefficients, which are then used as input to the actual neural network. In this contribution, we investigate, both theoretically and experimentally, the influence of this pre-processing step on the network’s performance and pose the question, whether replacing it by applying adaptive or learned filters directly to the raw data, can improve learning success. The theoretical results show that approximately reproducing mel-spectrogram coefficients by applying adaptive filters and subsequent time-averaging is in principle possible. On the other hand, extensive experimental work leads to the conclusion, that the invariance induced by mel-spectrogram coefficients is both desirable and hard to infer by the learning process. Thus, the results achieved by adaptive end-to-end learning approaches are close to but slightly worse than results achieved by state-of-the-art reference architectures using standard input coefficients derived from the spectrogram.

1 Introduction

Convolutional neural networks, first introduced in learning tasks for image data [15], have revolutionized state-of-the-art results in many machine learning (ML) problems. In convolutional neural networks (CNNs), when applied to image data, all filter coefficients are usually learned. For applications to time series, such as audio data, on the other hand, it is common practice to first apply a fixed filter bank to the raw, one-dimensional data in order to generate a feature representation. In traditional audio signal processing methods, used, e.g., in music information retrieval (MIR) or speech processing, FFT-based features such as mel-spectrograms are typically used as such inputs. These first level features are two-dimensional arrays, derived from some kind of windowed Fourier transform with subsequent mel-scale averaging.

Recently, the natural question arose, what kind of filters a network would learn if it was given the raw audio input. To date, encouraging results are scarce and so far, a true end-to-end approach for music signals, i.e., acting on raw audio without any pre-processing, has not been able to outperform models based on linear-frequency spectrogram or mel-spectrogram input [5]. It has been argued that these two ubiquitous representations automatically capture invariances which are of importance for all audio signals, in particular, a kind of translation invariance in time (guaranteed by introducing the non-linear magnitude operation) and a certain stability, introduced by the mel-averaging, to frequency shifts and time-warping (cp. [1]).

In this contribution, we give a formal description of the action of mel-scale averaging on spectrogram coefficients. We show that the resulting mel-spectrogram coefficients can indeed be mimicked by applying frequency-adaptive filters, however, followed...
by time-averaging of each filter’s output. In order to obtain a close approximation to mel-spectrogram coefficients, the frequency adaptive filter bank’s output signals must each undergo a time-averaging operation and the time-averaging window is different for each channel. Note that the similarity of mel-spectrogram coefficients to the result of time-averaging wavelet coefficients has already been observed in [1], without giving a precise formulation of the connection.

We derive the necessary conditions on the filters, a different one for each bin in the mel-scale, by using the theory of Gabor multipliers and their spreading function, cf. [7]. Considering the description of an operator by means of its spreading function gives interesting insight in the nature of the correlations invoked by the application of the corresponding operator on the signal coefficients. In the case of mel-spectrogram coefficients it turns out that applying wide triangular windows in the high frequency regions actually corresponds to the application of an operator with little spreading in time. This seems to be the intuitively correct choice for audio signals such as music and speech. While a similar effect can be realized by applying wavelet or constant-Q type filters, the subsequent time-averaging alleviates the significant frequency-spreading effect introduced by rather narrow filtering windows. The observation gained from investigating the classical mel-spectrogram coefficients is thus, that time- and frequency-averaging spectrogram coefficients provide invariances which are useful in most audio classification tasks. On the other hand, relaxing the strict averaging performed by computing mel-spectrogram coefficients may intuitively open the opportunity to keep information on details which may be necessary in certain learning tasks.

In our numerical experiments we thus strive to understand how time and frequency averaging influence CNN prediction performance on realistic data sets. The observations drawn from the experiments on learning filters can be summarized as follows:

- Allowing the net to learn time-averaging leads to comparable, but not improved, prediction results compared to learning from mel-spectrogram inputs.
- The adapted time-averaging windows do not coincide with the general mathematical theory but are dependent on the specific structure of the training data.
- Tricks are required to make the CNNs adapt the feature processing stage at all. Otherwise, the classification part of the network takes over the adaptation required to minimize the target loss.

This paper is organized as follows. In the next section we introduce necessary concepts from time-frequency analysis. In Section 3, we give a formal description of the network architecture, since we haven’t found any concise exposition in the literature. Section 4 then gives the formal result linking mel-spectrogram coefficients with adaptive filter-banks. We also give a detailed analytical example based on Gaussian windows and present a numerical evaluation of the resulting approximation of mel-spectrogram coefficients. In Section 5 we report on the experiments with a real-world data set for the problem.

\[\text{This observation seems to have served as one motivation to introduce the so-called scattering transform, which consists of repeated composition of convolution, a nonlinearity in the form of taking the absolute value and time-averaging. In that framework, mel-spectrogram coefficients are interpreted as first order scattering coefficients.}\]
of singing voice detection. Finally we conclude with a discussion and perspectives in Section 6.

2 Time-Frequency concepts

The Fourier transformation of a function \( f \in \mathcal{H} \), for some Hilbertspace \( \mathcal{H} \), will be denoted by \( \mathcal{F}(f) \). We use the normalization \( \mathcal{F}(f)(\omega) = \int_{\mathbb{R}} f(t) e^{2\pi i \omega t} dt \) and denote its inverse by \( \mathcal{F}^{-1}(f)(t) = \int_{\mathbb{R}} f(\omega) e^{2\pi i \omega t} d\omega \). For \( x, \omega \in \mathbb{R} \), the translation or time shift operator of a function \( f \) is defined as

\[
T_x f(t) = f(t-x).
\]

and the modulation or frequency shift operator of a function \( f \) is defined as

\[
M_\omega f(t) = e^{2\pi i \omega t} f(t).
\]

The operators of the form \( T_x M_\omega \) or \( M_\omega T_x \) are called time-frequency shifts. To obtain local information about the frequency spectrum we define the short-time Fourier transform (STFT) of a function \( f \) with respect to a window \( g \neq 0 \), where \( f,g \in \mathcal{H} \), as

\[
\mathcal{V}_g f(b,k) = \int_{\mathbb{T}} f(t) g(t-b) e^{-2\pi i kt} dt = \mathcal{F}(f \cdot T_b g)(k).
\]

(1)

Moreover this can be written as an inner product combining the above introduced operators

\[
\mathcal{V}_g f(b,k) = \langle f, M_k T_b g \rangle.
\]

Taking the absolute value squared we obtain the spectrogram as \( S_0(b,k) = |\mathcal{V}_g f(b,k)|^2 \).

3 The structure of CNNs

The basic, modular structure of CNNs has often been described, see e.g., [10]. Here, we will give a formal statement of the specific architecture used in the experiments in this paper. This architecture has been successfully applied to several MIR tasks and seems to have a prototypical character for audio applications, cf. [11].

The most basic building block in a general neural network may be written as

\[
x_{n+1} = \sigma(A_n x_n + b_n)
\]

where \( x_n \) is the data vector, or array, in the n-th layer, \( A_n \) represents a linear operator, \( b_n \) is a vector of biases in the n-th layer and the nonlinearity \( \sigma \) is applied component-wise. Note that in each layer the array \( x_n \) may have a different dimension. Now, in the case of convolutional layers of CNNs, the matrix \( A \) has a particular structure for the convolutional layers, namely, it is a block-Toeplitz matrix, or, depending on the implementation of the filters, a concatenation of circular matrices, each representing one convolution kernel. There may be an arbitrarily high number of convolutional layers, followed by a certain number of so-called dense layers, for which \( A_n \) is again an arbitrary linear operator. In this paper, the chosen architecture comprises up to four convolutional and two or three dense layers.
Remark 3.1. Note that it has been observed in [16, 19] that in the context of scattering networks, most of the input signal’s energy is contained in the output of the first two convolutional layers. While the context and the filters here are different, this observation might be interesting also as a background for the usual choice of architecture of CNNs for audio processing.

3.1 The CNN with Spectrogram Input

The standard input in learning methods for audio signal is based on a spectrogram, either in its raw form, or after some pre-processing such as the computation of mel-spectrogram, cf. Equation (4) in Section 4, which we will consider in detail in Section 4. In any case, the input to the CNN is a matrix of size $M \times N$.

Remark 3.2. In most MIR tasks, the inputs are derived from rather short snippets, that is, about 2 to 4 seconds of sound. Considering a sampling rate of 22050 Hz, a window size of 2048 samples and a time shift parameter of 512 samples, i.e., 23 ms, the resulting spectrogram (containing positive frequencies only) is of size $M \times N = 1024 \times 130$, where the latter is the time dimension. Hence, the frequency dimension is, in some sense, over-sampled. In particular, individual bins in the higher frequency regions contain less energy and thus information than in lower regions. Computing the mel-spectrogram is a convenient and straight-forward method of reducing the information to typically 80 frequency channels by averaging over increasingly many frequency bins.

We now define the following building blocks of a typical CNN:

- **Convolution**: $S * w(m, n) := \sum_{m'} \sum_{n'} S(m', n') w(m - m', n - n')$

- **Pooling**: For $1 \leq p \leq \infty$, we define $A \times B$ pooling as the operator mapping an $M \times N$ matrix $S_0$ to a $M/A \times N/B$ matrix $S_1$ by

$$S_1(m, n) = P^{A,B}_p (m, n) = \|v^{m,n}_{S_0}\|_p$$

where $v^{m,n}_{S_0}$ is the vector consisting of the matrix entries $S_0((m-1) \cdot A + 1, \ldots, m \cdot A; (n-1) \cdot B + 1, \ldots, n \cdot B)$ for $m = 1, \ldots, M/A$, $n = 1, \ldots, N/B$. In this work, we use max-pooling, which has been the most successful choice, corresponding to $p = \infty$ in the above formula.

- **A nonlinearity** $\sigma : \mathbb{R} \mapsto \mathbb{R}$, whose action is always to be understood component-wise. In all but the last layer we use leaky rectified linear units, which allow for a small, non-zero gradient when the unit is not active:

$$\sigma(x) = \begin{cases} x & \text{if } x > 0 \\ cx & \text{otherwise} \end{cases}$$

for some $c \ll 1$.

The output layer’s nonlinearity $\sigma_o$ is a sigmoid function. Given the above definitions, we can now write the output of a convolutional layer with an input array $S^{k_n-1}_{n} \in \mathbb{R}^{M \times N}$ as
where $\mathbf{1}$ is an all-ones matrix of size $M_n \times N_n$, $b_{kn} \in \mathbb{R}^{K_n}$ and $S_{n+1}^{kn} \in \mathbb{R}^{M_n/A_n \times N_n/B_n}$ for $k_n = 1, \ldots, K_n$.

Letting $D_c$ denote the number of convolutional layers and $S_{DC-1}^{kDC-1} \in \mathbb{R}^{M_{DC} \times N_{DC} \times K_{DC-1} \times 1}$ the output of the last convolutional layer, the over-all action of a CNN with two dense layers, and considering, for simplicity, a single output unit emitting $x_{out}$, can be written as

$$x_{out} = \sigma_2(\sigma_1(A_1 \cdot S_{DC-1}^{kDC-1} + b_{DC+1}) + b_{DC+2})$$

Here, $A_1$ and $A_2$ are weight-matrices of size $N_d \times M_{DC} \times N_{DC} \times 1 \times N_d$, respectively, where $N_d$ is the number of hidden units in the first dense layer, $b_{DC+1} \in \mathbb{R}^{N_d}$ and $b_{DC+2} \in \mathbb{R}$.

### 3.2 Modifying the input matrix

As mentioned in the previous section, the spectrogram of audio is often pre-processed in order to reduce the dimensionality on the one hand, and in order to obtain a spectral representation that better fits both human perception and properties of speech and music on the other hand. Additionally, the authors in [1] pointed out that using mel-spectrogram improved stability with respect to frequency shifts or, more generally, deformations of the original audio signals, than the usage of spectrograms. However, given appropriate choice of network architecture, comparable results can usually be achieved using either the spectrogram or the mel-spectrogram, i.e., the invariance introduced by the mel-averaging can also be learned. In other respects, omitting the frequency-averaging provided by the mel-spectrogram leads to an increase in the number of weights to be learned. On the other hand, this observation raises the question, whether using filters learned directly in the time-domain, would improve the net’s ability to achieve the amount of invariance most appropriate for a particular ML task and thus increase stability. The corresponding approach then implies learning time-domain filters already in a layer prior to the first 2D-convolution. To put this remark into perspective, we note that the spectrogram may easily be interpreted as the combined (and possibly sub-sampled) output of several convolutions, since, setting $h(n) = h(-n)$, we can write

$$S_0(m, n) = \left| \sum_{n'} f(n')h(n' - n)e^{-2\pi imn'} \right|^2 = |f \ast h_m(n)|^2$$

In the two following sections we thus raise and answer two questions.

(i) Is it possible to obtain coefficients which are approximately equivalent to the well-established mel-spectrogram coefficients simply by using the ‘correct’ filters directly on the audio signal?
(ii) Can adaptivity in frequency- and time-averaging improve prediction accuracy? In particular, for a given set of frequency-adaptive filters precisely mimicking the mel-scale, can a time-averaging layer with learned averaging width improve learning performance?

4 The mel-spectrogram and basic filters

In this section, we take a detailed look at the mel-spectrogram. This representation is derived from the classical spectrogram by weighted averaging of the absolute values squared of the STFT and can undoubtedly be referred to as the most important feature set used in speech and audio processing, together with MFCCs which are directly derived from it. The number of mel-filters used varies between 80 filters between 80 Hz and 16 kHz [11] and 128 [5] or more. In order to better understand the relation between the result of mel-averaging and FFT-based analysis with flexible windows, we observe the following: denote the input signal by \( f \), the window function for generating the spectrogram by \( g \), and the mel-filters by \( \lambda \nu \) for \( \nu = 1 \ldots K \), where \( K \) is the chosen number of filters. We can then write the mel-spectrogram as

\[
\text{MS}_g(f)(b, \nu) = \sum_k |\mathcal{F}(f \cdot T_b g)(k)|^2 \cdot \lambda \nu(k). \tag{4}
\]

Andén and Mallat showed in [1], that the mel-spectrogram can be approximated by time-averaging the absolute values squared of a wavelet transform. Here, we make their considerations precise by showing that we can get a close approximation of the mel-spectrogram coefficients if we use adaptive filters.

Remark 4.1. Note that the resulting transform may be interpreted as a nonstationary Gabor transform, compare [2, 3, 13, 6].

Recall that the STFT can be understood as

\[
V_g f(b, k) = \mathcal{F}(f \cdot T_b g)(k)
\]

Proposition 4.2. Let an analysis window \( g \) and mel-filters \( \lambda \nu \) be given, for \( \nu \in \mathcal{I} \). If, for each \( \nu \), the windows \( h_\nu \) and time-averaging functions \( \varpi_\nu \) are chosen such that

\[
V_h_\nu h_\nu(x, \xi) \cdot \mathcal{F}(\varpi_\nu)(\xi) = V_g g(x, \xi) \cdot \mathcal{F}^{-1}(\lambda \nu)(x), \tag{5}
\]

then the mel-spectrogram coefficients \( \text{MS}_g(f)(b, \nu) \) can be obtained by time-averaging the filtered signal’s absolute value squared as follows:

\[
\text{MS}_g(f)(b, \nu) = \sum_l |(f \ast h_\nu)(l)|^2 \cdot \varpi_\nu(l - b) = : TA^h(f)(b, \nu). \tag{6}
\]

The proof is given in Appendix A.

While it is in general tedious to derive conditions for the optimal filters \( h_\nu \) and the time-averaging windows \( \varpi_\nu \) explicitly, we obtain a more accessible situation if we restrict the choice of windows to dilated Gaussians. In this case we set \( g(t) = \varphi_\sigma(t) = (\frac{2}{\pi})^{\frac{1}{2}} e^{-\frac{t^2}{\sigma^2}} \).

Then we have, by straight-forward computation,

\[
V_{\varphi_\sigma} \varphi_\sigma(x, \xi) = e^{-\frac{x^2}{4\sigma^2}} e^{-\frac{\xi^2}{2\sigma^2}} e^{-\pi i x \xi},
\]
which leads to the following result:

**Corollary 4.3.** Fix \( g = \varphi_\sigma \) for some scaling factor \( \sigma \), and let the filters \( \Lambda_\nu \) be given as shifted and possibly dilated versions of a basic shape (e.g., in the case of mel-filters, asymmetric triangular functions), i.e., \( \Lambda_\nu(\xi) = T_\nu D_{a_\nu}(\xi) \), for \( \nu \in \mathcal{I} \).

Assuming that each filter \( h_\nu \) is a Gaussian window \( \varphi_{\rho(\nu)} \) with a frequency-dependent dilation factor \( \rho(\nu) \) and frequency-shifted to position \( \nu \), i.e., \( h_\nu(t) = e^{2\pi i \nu t} \varphi_{\rho(\nu)}(t) \). Then condition (5) is equivalent to the following conditions in separate variables:

\[
e^{-\frac{\pi}{2} x^2 (\frac{1}{\alpha^2} - \frac{1}{\beta^2})} = F^{-1}(D_{a_\nu}(\lambda))(x) \tag{7}
\]

\[
e^{-\frac{\pi}{2} \xi^2 (\sigma - \rho(\nu))} = F(\omega_\nu)(\xi). \tag{8}
\]

**Proof:** Noting that \( V_{h_\nu} h_\nu(x, \xi) = e^{2\pi i \nu x} V_{\varphi_{\rho(\nu)}} \varphi_{\rho(\nu)}(x, \xi) \) and that \( F^{-1}(\Lambda_\nu)(x) = e^{2\pi i \nu x} F^{-1}(D_{a_\nu}(\lambda))(x) \), the result is straight-forward.

**Remark 4.4.** From the above corollary we see that a precise recovery of the mel-spectrogram coefficients by means of time-averaging of the spectrogram coefficients with respect to any dilated Gaussian window is only possible if the averaging windows \( D_{a_\nu}(\lambda) \) are themselves dilated Gaussians. However, we can easily deduce the following pointwise error estimate:

\[
| \text{MS}_g(f)(b, \nu) - TA^h(f)(b, \nu) | \leq \| V_{h_\nu} h_\nu \cdot F(\omega_\nu) - V_g g \cdot F^{-1}(\Lambda_\nu) \|_2 \tag{9}
\]

So, even if the mel-spectrogram coefficients can not precisely be recovered by adaptive filtering followed by time-averaging, a close approximation can be achieved for reasonable windows \( g \). The experiments in the next section show that the deviations resulting from adaptive filtering followed by time-averaging as opposed to mel-coefficients are small and probably negligible in the setting of CNN-based learning. However, excellent precision of approximation is only achieved in the case of full spectrograms, i.e., if no subsampling is applied and declines with increasingly coarse subsampling strategies.

To see this, note that both \( F(\omega_\nu) \) and \( F^{-1}(\Lambda_\nu) \) are periodic with period \( \frac{1}{\alpha} \) and \( \frac{1}{\beta} \), respectively, if the time- and frequency variable in (1) are sub-sampled by \( \alpha \) and \( \beta \), respectively. Obviously, the resulting aliases deteriorate the estimate in (9). In the realistic scenarios addressed in Section 5, which always involve subsampled spectrograms, the differences between mel-spectrogram coefficients and their approximation by adaptive filtering become relevant for the performance of CNN-learning.

### 4.1 Examples

In this section, we give some examples of filters \( h_\nu \) computed to obtain mel-coefficients \( \text{MS}_g(f) \) by time-averaging \( |(f * h_\nu)(l)|^2 \) as in (6). We consider Hann windows, which is the standard choice in audio processing, also applied in the computation of mel-spectrogram coefficients and their approximation in Section 5. Starting from a Hann window \( g \), we compute adaptive filters \( h_\nu \) for the first 50 bins of the mel-scale. Figure 1 shows the ambiguity functions \( V_{g_\nu} \), \( V_{h_\nu} h_\nu \), and the weighted ambiguity functions \( V_{g_\nu} g \cdot F^{-1}(\Lambda_\nu) \), \( V_{h_\nu} h_\nu \cdot F(\omega_\nu)(\xi) \), for \( \nu = 49 \), which corresponds to 2587.6 Hz.
In Figure 2, the upper plot shows the original Hann and three adapted windows, narrowing with increasing mel-number to realize the mel-averaging by adaptivity in the frequency domain. The lower plot shows the average error per bin obtained from computing the mel-spectrogram coefficients and their approximations for 200 (normally distributed) random signals. We only show relevant bins, i.e., those bins, where mel-averaging concerns more than 1 original frequency bin since the error is zero otherwise. For illustration, in Figure 3, the absolute value squared of the output of filtering a random signal $f$ with a modulated version of $g$ and $h_{14}$, i.e., one row of $|V_g f|^2$ and $|V h_{14} f|^2$, respectively, as well as the frequency-averaged (Mel) and time-averaged (Approx.) versions thereof, are shown. Finally, the lower plots in Figure 3 show the full mel-spectrogram of a synthetic signal and its approximation by using the adapted $h_v$ per bin with subsequent time-averaging.

5 Experiments on Singing Voice Detection

In Proposition 4.2 it is shown that coefficients with mel-characteristics (and other related nonlinear scales) can be closely approximated by applying appropriately chosen filters directly to raw audio data and allowing for a subsequent time-averaging step. Now, we are interested in investigating if the theoretical findings translate to typical real-world problems that have already been successfully treated with CNNs. Thereby, we are motivated by the fact that state-of-the-art results for several MIR problems are based on mel-spectrogram coefficients which show certain desirable invariance and stability properties. In particular, due to the modulus, they are invariant to translation and, due to the frequency averaging, they exhibit stability to certain deformations such as time-warping, cf. [1]. However, in general, the required invariance and stability with respect to deformations will depend on data characteristics and the learning task, cf. [17]. Hence, we start from our construction encompassing adaptive filtering followed by time-averaging and allow the time-averaging width to be learned by the network, posing the question whether adaptivity in this step can improve the network’s performance.

We need to note that, when trained on a specific problem, both the feature layers (including the adaptive time-averaging step) and the classification part of a CNN will concurrently adapt their parameters towards optimally predicting the given targets. We will discuss the implications of this behavior for our experiments in Section 5.4.

Our hypothesis is that a CNN with an architecture that is adapted to a given learning task will learn filters — in this case their time-averaging components — which alleviate the extraction of stabilities and invariance properties and are thus beneficial in the given context.

5.1 Data

We investigate the effects of learning filters directly on raw audio by revisiting the problem of singing voice detection [18] we have studied before. In the referenced publication, a CNN was tuned for maximum prediction accuracy both in the absence or presence of various forms of data augmentation.

The experiments were performed on a non-public dataset of 188 30-second audio snippets from an online music store (dataset ‘In-House A’), covering a very wide range
Figure 1: Ambiguity functions $V_g g$, $V_{h \nu} h_{\nu}$, and weighted ambiguity functions $V_g g \cdot \mathcal{F}^{-1}(\Lambda_{\nu})$, $V_{h \nu} h_{\nu} \cdot \mathcal{F}(\omega_{\nu})(\xi)$ used for the computation of adaptive filtering, for $\nu = 50$. It is clearly visible that the surplus in frequency spread introduced by the narrower window $h_{\nu}$ is removed by time-averaging. On the other hand, frequency averaging reduces the time-spread of the wider window $g$. 
Figure 2: Upper plot: (original) Hann window and adapted windows; lower plot: error in approximation of mel-spectrogram coefficients by adaptive filtering and subsequent time-averaging
of genres and origins. We used a five-fold cross-validation, for each iteration 150 files for training, the remaining 38 for evaluation. The audio was subsampled to a sampling rate of 22.05 kHz and down-mixed to mono. The precomputed mel-spectrograms were calculated using an STFT with Hamming windows, a frame length of 1024 and hop size of 315 samples (yielding 70 frames per second). We discarded the phases and applied a filter-bank with 80 triangular mel-scaled filters from 27.5 Hz to 8 kHz, then logarithmized the magnitudes (after clipping values below $10^{-7}$). Finally, each mel-band was normalized to zero mean and unit variance over the entire training set.

5.2 CNN training procedure and architecture

The training procedure used in our experiments is slightly different than in the reference publication [18]. The networks are trained on mel-spectrogram excerpts of 115 spectrogram frames (~1.6 seconds) paired with a label denoting the presence or absence of human voice in the central frame. Training is performed using stochastic gradient descent on cross-entropy error based on mini-batches of 64 randomly chosen examples. Updates to the network weights are computed using the ADAM update rule [14] with an initial learning rate of 0.001 and an adaptive scheme reducing the learning rate twice by a factor of 10 whenever the training error does not improve over three consecutive episodes of 1000 updates.

Evaluation is performed running a complete five-fold cross-validation run to obtain predictions for the whole set of training data, with this procedure repeated five times. The mean classification error (1 − accuracy) obtained with this architecture in the absence of augmentation on the ‘In-House dataset A’ is 11.6%, insignificantly below the value of 12.0% reported in the reference paper.

As described in Section 3.1, the applied CNN architecture employs three types of feed-forward neural network layers: convolutional feature processing layers convolving a stack of 2D inputs with a set of learned 2D kernels, pooling layers subsampling a stack of 2D inputs by taking the maximum over small groups of neighboring pixels, and dense classification layers flattening the input to a vector and applying a dot product with a learned weight matrix $A_j$.

The architecture used in [18] has a total number of 1.41 million weights, with the dense connections of the classification layers taking up the major share (1.28 million, or 91%). It can be expected that the actual output of the convolutional feature stage is of subordinate importance when the classification stage with its high explanatory power dominates the network.

If no data augmentation is applied, the network size — especially the classification part — can be drastically reduced while largely preserving its performance. This size reduction is possible, since, as a general rule, the necessary number of parameters determining the network is correspondent to the complexity of the training data set. Since we are interested in the impact of the convolutional feature stage’s properties, we reduce the architecture for our experiments as follows: We use four convolutional layers, two $3 \times 3$ convolutions of 32 and 16 kernels, respectively, followed by $3 \times 3$ non-overlapping max-pooling and two more $3 \times 3$ convolutions of 32 and 16 kernels, respectively, and another $3 \times 3$ pooling stage.

In the notation of (2), this corresponds to
For the classification part, we experimented with two variants: One with two dense layers of 64 and 16 units, and the other one with just one dense layer of 32 units. In both cases, the final dense layer is a single sigmoidal output unit. For the first variant, the total number of weights is 94337, with the classification stage taking up 79969 units, or 85%. The second variant features a considerably smaller classification network: the total number of weights is 34113, with the classification stage taking up 19745 units, or 58%. The different network sizes, especially the ratio of feature to classification stage allows us to analyze the influence of the different parts.

5.3 Experimental setup

In the following, we will compare the behavior of the CNNs applied to the STFT spectrogram features to features computed ad hoc from the audio signal. For that, we use Gabor filter banks with subsequent magnitude computation, time-averaging and logarithmization. The maximum kernel sizes of the Gabor filter banks are set to 1024, identical to the frame length of the previously used STFT. The kernel size equals the time support for the lowest frequency band (50 Hz) and reduces, according to the band-width requirements of the mel frequency scale, down to 94 samples for the highest band at 7740 Hz. The ad-hoc feature computation also necessitates a shift from global normalization of spectral features (unit mean and standard deviation on the entire training set) to batch-wise normalization, computed on the fly. We could verify in side experiments with the reference architecture that the use of batch normalization does not change the scores. The training examples are snippets of the audio signal with a length of $115 \times 315 + 1024 = 37249$ samples each with a hop size of 315 samples.

To judge the influence of adaptivity, two different approaches have been compared:

1. Fixed-size time-averaging after the Gabor filter banks (architecture ‘Gabor fixed-width’)

2. Adaptive time-averaging after the Gabor filter banks, learned alongside the CNN weights either with a uniform averaging length over all frequency bins (‘Gabor variable-width uniform’), or with individual adaptation per frequency bin (‘Gabor variable-width individual’).
For reasons of computational cost it is not feasible to perform a full sample-by-sample convolution for the Gabor filter bank. For our experiments, we have chosen a convolution stride for the Gabor filters of 21 samples, that is, the resulting spectrum is down-sampled along the time axis by a factor of 21. The subsequent non-overlapping averaging is computed on 15 frames each, in order to stay comparable with the STFT hop size of $315 = 21 \times 15$ samples. Preliminary tests have shown that the chosen subsampling does not significantly degrade the results. Note that the stride is a factor of about 4.5 lower than the shortest kernel support (21 vs. 94).

For the fixed-size time-averaging variant standard average-pooling is used, implemented as a $15 \times 1$ 2D-pooling layer acting on the Gabor magnitude spectrum. In this case the temporal averaging size is uniform over the frequency axis which is an approximation of the mathematical findings. The variable-size case is implemented using Hann windows with adaptive width, acting on the magnitude spectrum. The maximum time support of this Hann window is 8 times the STFT hop size, equivalent to 2520 samples. The choice of Hann in contrast to Boxcar windows (as in the fixed-size case) is motivated by its smoothness which enables adaptivity for the CNN training process. Two variants were tried: Firstly, a uniform width over all frequency bands, corresponding to the fixed-size case, and, secondly, individually adaptive sizes per band.

5.4 Experimental results

Figure 4 shows the results of our CNN experiments for the problem of singing voice detection. For our evaluations, we have switched from the simple error measure with the ‘optimal’ (in the sense of maximum accuracy) threshold per experiment to the more informative ‘area over the ROC curve’ measure (AOC), fusing classification errors for all possible thresholds into one measure. A lower measure indicates a better result. Leftmost, the reference architecture with the large network is displayed. Right from it, there are two groups of experiments: for the ‘small-two’ and ‘small-one’ networks, respectively. Each group contains one experiment for the STFT baseline (‘STFT ad-hoc’), one for a fixed-width time averaging (‘Gabor fixed-width’) and variable-width experiments with either uniform or individual averaging widths for the spectral bands.

The reference implementation in [18] uses pre-computed spectrograms, with a normalization globally on the the training set and eventual padding performed also on the spectrogram. End-to-end learning as performed in our experiments demands on-line normalization (using a batch normalization layer) and padding directly on the audio time signal. We could verify that this yields a performance equivalent with the reference experiments (independent t-test probability level 79% for equal means, $df = 8$) which is shown in Figure 4 in the first two columns (‘STFT precomputed’ and ‘STFT ad-hoc’).

We can also confirm that the results of our smaller networks (‘small-two’ and ‘small-one’, for the two- and one-classification-layer variants) are comparable to the large baseline architecture. For AOC (‘STFT ad-hoc’ case), the smaller networks score 5.22% (‘small-two’) and 5.29% (‘small-one’), respectively, compared to 4.96% of the original architecture. Although the STFT results for the two small networks are almost identical (independent t-test probability level 70%, $df = 8$), there is a general tendency for the smaller networks to show slightly greater AOC measures, as expected.

In the course of experimentation it has become apparent that the averaging widths of
the Gabor-filter models hardly adapt at all. They rather stay at the initial values, while the CNN weights adapt instead. As a trick we have boosted the widths’ gradients for the back-propagation by a factor 3 to force the width parameters to adapt at a higher rate. Higher factors have proven unfeasible, causing the adaptation to run out of bounds. Since the adaptation process is intricate, the choice of a starting value for the variable averaging length (time support of the Hann window) is important. We have tried values of 0.1, 0.2, 0.3, 0.5 of the maximum time support, with 0.2 leading to the best results.

The performance measures obtained using Gabor filters with subsequent time-averaging is significantly lower than with STFT-computed mel-spectrograms, with a t-test probability level of $p < 0.5\%$ for equal means (again $df = 8$). The different variations of Gabor-filter-based experiments perform equivalently for the ‘small-two’ and ‘small-one’ architectures, respectively. There is no observable trend or statistically significant performance difference between the ‘Gabor fixed-width’, ‘Gabor variable-width uniform’ and ‘Gabor variable-width individual’ models.

Figure 5 shows the adapted averaging widths per frequency band calculated in the CNN training process of the ‘small-two’ for the five cross-validation folds with 5 experiments each. Obviously, the widths vary considerably across experiments. The mean Pearson correlations between pairs of curves amount to 0.49, 0.45, 0.38, 0.44, 0.29, for the different folds, all representing significant correlations at a level of $p = 5\%$. Some common patterns can be discerned, e.g., the bump upwards at about 600 Hz, or the hill-valley-hill shape towards the high end of the spectrum. In fact, when taking the means of the adapted averaging widths for the different architectures in our study, it becomes clear that there is a very strong correlation of the adaptations of the two models, shown in Figure 6. The Pearson correlation coefficients between pairs of widths for the architectures lie in the range of $r = 0.86 \ldots 0.91$. The adapted time-averaging widths range from 390 to 655 samples, or 18 to 30 ms. Note that the time-averaging is performed with a Hann window, resulting in a somewhat smaller effective length, in contrast to the Boxcar window of the fixed-width case with an averaging length of 315 samples. The results of the adaptive case tend to be slightly better than the fixed averaging, but again, not significantly.

From the mathematical derivation above, one may expect the time-averaging width to increase with frequency in order to mimic the, supposedly optimal, behavior of the mel-spectrogram. It can be observed that this is not the case in practical experiments, where the structure adapts to the characteristics of the training data.

5.4.1 Adapting the mel-filters

As discussed in the previous section, experiments showed that coefficients computed by using filters which directly reflect the frequency resolution of the mel-scale are not able to improve or even reach the state-of-the art results obtained by using mel-spectrogram coefficients as input to the networks. Allowing for adaptivity in the frequency resolution itself deteriorated the results even further. We therefore assume that computing frequency-averaged coefficients from pre-computed spectrograms is advantageous at least for the learning task we focus on and thus, instead of a true end-to-end learning approach, allowed the net to adapt the frequency-filters used in the averaging step as described in (4). Indeed, using this approach, the reference results could be reproduced and a small but
significant performance improvement is achieved when adaptivity is introduced. We can observe, that the adaptation process is again consistent over different architectures, i.e., the amount of adaptivity per mel-frequency band is highly correlated (pairwise Pearson correlation > 0.99) over the three architectures studied. The deviation from the (mel-scale based) initialization is shown in Figure 7.

6 Discussion and Perspectives

In this article we posed and answered two questions concerning the application of alternative time-frequency representations for learning problems in music information retrieval. First, it has been analytically shown, under which conditions mel-spectrogram coefficients can be reproduced by applying frequency-adaptive filters followed by a time-averaging step. In practice, this procedure will always lead to approximate values. We expected the resulting approximation errors to be out-balanced by subsequent convolutional layers of the neural network and thus at least equal or, due to possible advantageous effects of adaptivity, improved performance in comparison to standard settings. However, and this answers the second question, extensive experimental studies concerning the problem of singing voice detection have not led to advantageous effects of the usage of adaptive filters. Coefficients obtained by averaging pre-computed spectrogram coefficients over frequency bins (as in mel-spectrogram coefficients, cf. (4)), consistently lead to the best results, albeit by a small margin. The reasons for this observation require further investigations and may be related to the following facts: First, the setting involving only one basic analysis window is well-established and in particular, the original window length may be optimal in the sense that the subsequent CNN architecture’s parameters, e.g., the size of the subsequent filters, is properly adapted to its characteristics. Second, the classical method to obtain mel-spectrogram coefficients, (4), leads to a block-structure of the time-frequency resolution of the resulting representation. This block-structure depends on the length, in samples, of the original analysis window, since this length determines the FFT-length in (1). Hence, the number of available frequency-bins the mel-spectrogram coefficients can be based on, depends on the analysis window’s length. It seems possible, that subsequent convolutional layers adapt more easily to this block-structure. Third, the impact of the approximation errors occurring in the Gabor-filter based approximation of mel-spectrogram coefficients is unclear and needs to be further studied analytically.

Future work on the problem of learned basic filters in MIR tasks will involve the study of the influence of the analysis window used to compute the spectrogram, cp. preliminary work in [6], both regarding the network’s expressivity and the performance of the learning process. Further, the questions whether the block-structure corresponding to mel-spectrogram coefficients is beneficial will be investigated both analytically and experimentally. Finally, the propagation and alleviation of approximation errors will be investigated relying on existing results on stability of CNNs, compare [20, 4, 12].
A Invariance Properties induced by filters: Technical Details

A.1 Proof of Proposition 1

The proof of Proposition 1 is based on the observation that the mel-spectrogram can be written via the operation of so-called STFT- or Gabor-multipliers, cf. [9], on any given function in the sense of a bilinear form. Before deriving the involved correspondence, we thus introduce this important class of operators.

Given a window function \( g \), time- and frequency-sub-sampling parameters \( \alpha, \beta \), respectively, and a function \( m : \mathbb{Z} \times \mathbb{Z} \to \mathbb{C} \), the corresponding Gabor multiplier \( G_{g^\alpha,m}^\beta \) is defined as

\[
G_{g^\alpha,m}^\beta f = \sum_k \sum_l m(k,l) \langle f, M_{k\beta} T_{\alpha g} \rangle M_{k\beta} T_{\alpha g}.
\]

We next derive the expression of a mel-spectrogram by an appropriately chosen Gabor multiplier. Using sub-sampling factors \( \alpha \) in time and \( \beta \) in frequency as before, we start from (4) and reformulate as follows:

\[
\text{MS}_g(f)(b, \nu) = \sum_k |F (f \cdot T_{b g} (\beta k))|^2 \cdot \Lambda_\nu(k)
\]

\[
= \sum_k \langle f, M_{\beta k} T_{b g} \rangle \langle f, M_{\beta k} T_{b g} \rangle \Lambda_\nu(k)
\]

\[
= (\sum_k \Lambda_\nu(k) \langle f, M_{\beta k} T_{b g} \rangle M_{\beta k} T_{b g}, f)
\]

\[
= (\sum_k \sum_l m(k,l) \langle f, M_{\beta k} T_{\alpha l g} \rangle M_{\beta k} T_{\alpha l g}, f)
\]

with \( m(k,l) = \delta(\alpha l - b) \Lambda_\nu(k) \), we see that the mel-coefficients can thus be interpreted via a Gabor multiplier as defined above, i.e., \( \text{MS}_g(f)(b, \nu) = \langle G_{g^\alpha,m}^\beta f, f \rangle \).

The next step is to switch to an alternative operator representation. Indeed, as shown in [8], every operator \( H \) can equally be written by means of its spreading function \( \eta_H \) as

\[
H f(t) = \int_\mathbb{R} \int_\mathbb{R} \eta_H(x, \xi) f(t-x) e^{2\pi i t \xi} d\xi dx.
\]

(10)

We note that two operators \( H_1, H_2 \) are equal if and only if their spreading functions coincide, see [7, 8] for details.

As shown in [7], a Gabor multiplier’s spreading function \( \eta_{g^\alpha,m}^\beta \) is given by

\[
\eta_{g^\alpha,m}^\beta(x, \xi) = \mathcal{M}(x, \xi) V_{g^\alpha,m}(x, \xi),
\]

(11)

where \( \mathcal{M}(x, \xi) \) denotes the \((\beta^{-1}, \alpha^{-1})\)-periodic symplectic Fourier transform of \( m \), i.e.,

\[
\mathcal{M}(x, \xi) = \mathcal{F}_s(m)(x, \xi) = \sum_k \sum_l m(k,l) e^{-2\pi i (\alpha l \xi - k \beta x)}.
\]

We now equally rewrite the time-averaging operation applied to a filtered signal, as defined in (6), as a Gabor multiplier. We set \( \tilde{h}(t) = h(-t) \) and have

\[
\sum_l |(f * h_\nu)(\alpha l)|^2 \cdot \omega_\nu(\alpha l - b) =
\]
\[
\sum_l \sum_n |f(n)\tilde{h}_\nu(n-\alpha l)|^2 \cdot \varpi_\nu(\alpha l-b)
\]

\[= \sum_k \sum_l |\langle f, M_{\beta k} T_{\alpha l} \tilde{h}_\nu \rangle|^2 \cdot \varpi_\nu(\alpha l-b) \delta(k)
\]

\[= \langle G_{\tilde{h}_\nu}^\alpha m_T f, f \rangle.
\]

with \(m_T(k,l) = T_b \varpi_\nu(l) \delta(k)\).

In order to compare the two operators \(G_{\delta,\alpha}^\beta m\) and \(G_{\tilde{h}_\nu}^\alpha m_T\), we compute their respective spreading functions by applying the symplectic Fourier transform to \(m = T_b \delta \cdot \Lambda_\nu\) and \(m_T = T_b \varpi_\nu \cdot \delta\), respectively, to find:

\[
\mathcal{M}(x,\xi) = \mathcal{F}_s(m)(x,\xi) = e^{-2\pi ib\xi} \cdot \mathcal{F}^{-1}(\Lambda_\nu)(x)
\]

and

\[
\mathcal{M}_T(x,\xi) = \mathcal{F}_s(m_T)(x,\xi) = e^{-2\pi ib\xi} \cdot \mathcal{F}(\varpi_\nu)(\xi)
\]

Plugging these expressions into (11) yields the condition in Proposition 1 and thus concludes its proof.

**Remark A.1.** It is interesting to interpret the action of an operator in terms of its spreading function. In view of (10), we see that the spreading function determines the amount of shift in time and frequency, which the action of the operator imposes on a function. For Gabor multipliers, if well-concentrated window functions are used, it is immediately obvious that the amount of shifting is moderate as well as determined by the window’s eccentricity. At the same time, the aliasing effects introduced by coarse sub-sampling are reflected in the periodic nature of \(\mathcal{M}\). Since, for \(\mathcal{F}^{-1}(\Lambda_\nu)\) the sub-sampling density in frequency, determined by \(\beta\), and for \(\mathcal{F}(\varpi_\nu)\) the sub-sampling density in time, determined by \(\alpha\), determine the amount of aliasing, the over-all approximation quality deteriorates with increasing sub-sampling factors.

**References**


Figure 3: Upper plot: Filtering a random signal $f$ with modulated Hann window $g$ and adapted window, and effect of frequency- (Mel) and time-averaging (Approx.). Lower plots show full mel-spectrogram of synthetic signal and approximation by adaptive filtering with subsequent time-averaging.
Figure 4: AOC measures for the problem of singing voice detection. Models compared are results for multiple runs of five-fold cross-validation on pre-computed and pre-normalized mel-spectrograms (STFT precomputed), on mel-spectrograms computed on-the-fly (STFT ad-hoc), on batch-normalized Gabor filter-banks with fixed-width temporal averaging (Gabor fixed-width), and with variable-width averaging in two variants: uniform width across all frequency bands (Gabor variable-width uniform) and individual widths (Gabor variable-width individual). Shown are individual results (gray crosses), mean (red triangles) and median (green dots) values.
Figure 5: Adapted averaging widths for different folds of the ‘small-two’ architecture per mel-frequency band. Values are shown for 5 experiments per fold (in gray), along with mean and standard deviation (red). The starting value of 0.2 is depicted in dashed line style.

Figure 6: Mean adapted averaging widths per mel-frequency band for the different architectures. The starting value of 0.2 is depicted in dashed line style.
Figure 7: Adapted frequency widths per mel-frequency band for the different architectures.