Audio Engineering Society

Convention Paper 10634

Presented at the 153rd Convention
2022 October

Comparison of Audio Spectral Features in a Convolutional Neural Network

Greg Vines and Elias Nemer

San Diego, CA, U.S.A.

Correspondence should be addressed to G. Vines (gvines@ieee.org)

ABSTRACT

Time-Frequency transformation and spectral representations of audio signals are commonly used in various machine learning applications. Typically the Mel-Spectrogram is used to create the input features to the network justified by the Mel scale's human auditory system basis. In this paper, we compare several spectral features in a gender detection speech model comparing their performance and showing that the Mel-Spectrogram is not always the best choice for input features.

1. Introduction

Two dimensional representations of audio where frequency or spectral features are computed against time are often used in neural networks because they allow modelling characteristics of the auditory system, such as masking, perception, or resolution. Having a two dimensional input allows audio applications to leverage the progress made in the image processing field.

Spectral data such as the *Mel*-Spectrogram [1] (Melgram), Chromagram [2] or Cochleogram [3] are some examples of such Time Frequency (TF) representations. They are often used in applications where estimation [4], classification [5][6] or detection is sought.

While there have been some networks which use time-samples as input, such as WaveNet [7], the majority of recent published work still uses spectral features as input to the models as the results have shown that they provide superior performance to taking in raw audio samples [8]. However, most audio neural networks still use Melgrams as their input features. For example, in the 2020 DCASE Task 1, Low Complexity Scene Classification challenge [9] all ten of the top performing entries used log Mel energies, or a variation thereof.

Because a variety of transforms can be used and easily incorporated into the networks early layers [10][11], the natural question is which transform is the best. This paper investigates several spectral representations comparing their relative performances. In addition to the Mel-Spectrogram, the Chromagram, Cochleogram, and Short Time Fourier Transform (STFT) are implemented in a Gender Detection model using a Convolutional Neural Network (CNN).

2. Model Description and Definition

The `Pytorch` framework [12] was used in this work because it provided a flexible and powerful development environment. The model was based on the VGG image classification model using a scaled down network appropriate to the input feature size [13].

Each time-frequency transform is constructed to have 64 columns each with 64 values. This allows all approaches to use the same neural network and eases comparison of the results.

The network was modelled after the VGG networks, but with a smaller structure. The pattern of two 2D convolutional layers, Batch Norm, and `ReLU` modules followed by a max pool layer was repeated four times. The output of these layers was $4x4x96$ which was fed into three fully connected layers with `ReLU` and dropout in between. The output of the fully connected layer yielded the three classes which were fed into a `softmax` function.

The resulting network has eight 2D convolutional layers followed by three linear layers for a total of 708,291 parameters. Figure 1 illustrates the structure of the network.

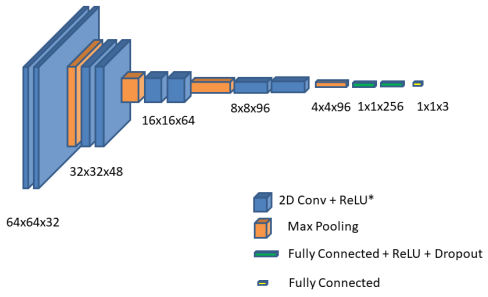

Figure 1: Network Architecture

3. Input Feature Generation

A. Mel-Spectrogram

a. Definition

A Spectrogram is a convenient way to visualize the magnitude of the DFT points for each analysis frame. A spectrogram is a 3D plot with the *x* and *y* axis being the time and frequency indices respectively. The third dimension is often replaced with an intensity value so that the entire plot is viewed in 2D.

The *Mel* scale came about with the realization that human hearing does not perceive frequency on a linear, but a more logarithmic-like, scale [14]. The *Mel* scale was developed based on having a unit of pitch such that equal distances in pitch sounded equally distant to the listener. A transformation from frequency in Hz to the *Mel* pitch scale [15] is given by:

$$m_{mel}(f) = 2595\log_{10}\left(1+\frac{f}{700}\right) \quad (1)$$

With the reverse transformation from Mel to Hz as:

$$f_{Hz}(m) = 700\left(10^{m/2595}-1\right) \quad (2)$$

b. Functional Implementation

The *Mel*-Spectrogram is generated by first computing a DFT on the signal, using a given frame length (N) and an overlap. We then compute the so-called Mel-scale filterbank, which is a set of M filters that are applied to the spectrogram. Each one of these filters combines the energy of a number of frequency points representing the *Mel* unit. Each vector representing the filter is mostly zero except for a small segment of the spectrum.

c. Determining the Mel Filters

The span and values of each filter coefficient is a function of the sampling frequency, the number of DFT points, and the desired bandwidth of the signal to cover. Below is the summary of the key steps detailed in [16].

- Define the frequency range and compute the end points in *Hz* and *Mels*.
- Decide how many filterbanks are needed, and define equally spaced frequencies in *Mels* using Eq. 1.
- Convert these frequencies to *Hz* using Eq. 2.

- Translate these frequencies to bins, based on the STFT size and generate the set $f(m)$ in bins for $m = 1, ..., M$.
- Construct the M filters: $H_m(k) \quad m = 1, ..., M$

$$H_m(k) = \begin{cases} 0 & \text{..... for .....} \quad k < f(m-1) \\ \dfrac{k - f(m-1)}{f(m) - f(m-1)} & f(m-1) \le k \le f(m) \\ \dfrac{f(m+1) - k}{f(m+1) - f(m)} & f(m) \le k \le f(m+1) \\ 0 & k > f(m+1) \\ 1 & k = f(m) \end{cases}$$

In this work the Melgram filter from the `Librosa` library is used.

B. Chromagram

a. Definition

In music analysis, pitch is a perceptual property of sounds and allows their ordering on a frequency-related scale. It is this quality that makes it possible to judge sounds as "higher" or "lower" in pitch. The human perception of pitch is periodic in the sense that two pitches are perceived as similar in “color”, or harmonic role, if they differ by one or several octaves. Based on this observation, a pitch can be separated into two components, which are referred to as tone height and Chroma, and sometimes called the pitch helix [17]. The Chroma, or color, captures the harmonic and melodic characteristics of music, and is robust to changes in timbre and instrumentation. Such features are commonly used in music analysis, where the pitches can be meaningfully categorized. Assuming the equal-tempered scale, one considers twelve Chroma values represented by the set {C, C♯, D, D♯, E , F, F♯, G, G♯, A, A♯, B} which are the twelve pitch spelling attributes used in Western music notation.

From a signal processing perspective, the term Chroma vector, or Chromagram, relates to the twelve pitch classes. It is typically a 12-element feature vector, though it may contain more elements, and indicates how much energy from each pitch class {C, C#,D, D#, E, ..., B} is present in the signal.

b. Functional Implementation

The conversion of an audio recording into a Chroma representation, or Chromagram, may be performed by using the Short-Time Fourier Transform (STFT) in combination with binning strategies [18]. The idea is to aggregate all spectral information that relates to a given pitch class into a single coefficient.

c. Determining the Chroma Filters

The Chroma filters are somewhat similar to the *Mel* filters described earlier. They combine the energies of a selected set of frequency bins to yield the desired filter output. In this case each Chroma filter combines the energy of one pitch class, for example the C# pitch class, from the various bins of the DFT vector which conceptually represent the various octaves.

In this work, we make use of the `Librosa` library function to synthesize such filters, given the number of FFT points, sampling rate, and the number of filters. An example is given in Fig. 2 below showing four of 24 filters.

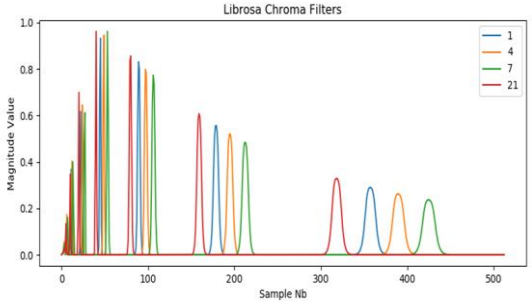

Figure 2: Subset of Chroma Filters (from `Librosa`)

C. Cochleogram

a. Background

The use of the Cochleogram stems from the attempt to model the response of the human auditory system, namely the inner ear or cochlea. It was found that each point along the basilar membrane (BM) inside the cochlea responds best to a range of frequencies.

This movement is sensed by the inner hair cells [3] which translate it into neural signals to the brain. The resonant frequency along the cochlea varies smoothly from high frequencies at the base to low frequencies at the apex.

A common way to describe this frequency specific behaviour is to model it as a bank of parallel auditory filters (AF). The AF are highly overlapping since each point of the BM defines its own AF, with a critical frequency determined by its distance from the apex.

b. Cochleogram Implementation

Auditory filters can be modeled in the frequency domain. The `PyCochleogram` [19] functionality used in this work is a frequency-based Python implementation of the Cochleogram. The various steps carried out in [19] are illustrated in Fig. 3. The incoming signal $s(n)$ is transformed to the frequency domain $S(f)$ and multiplied by the frequency representation $H_m(f)$ of the filters. A Hilbert transform is applied to the frequency domain outputs $Y_m(f)$ and creates the so-called analytical signals. An inverse-DFT brings the results back to a complex time-domain representation $x_m(n)$. From these real and imaginary components, the envelope of each filter output is computed and yields, after a power operator, the Cochleogram value. Note that for each N input data points, $s(n)$, there are also N data points at the $x_m(n)$ output of each filter, which is down-sampled from 512 to obtain 64 samples.

D. Short Time Fourier Transform

a. Definition

The Short Time Fourier Transform (STFT) provides the Fourier spectrum on shorter time segments. A windowed subset of the source data is transformed with an FFT to provide a frequency representation of a short time interval of the original signal. The STFT can be written for the discrete time case as:

$$Y(k)\big|_m = \sum_{n=-\infty}^{\infty} x[n]\,w[n-m]\,e^{-j\cdot k\cdot n\cdot\frac{2\pi}{N}} \quad k = 0,.,N-1 \quad (3)$$

Where $x[n]$ is the function to be transformed, $w[n]$ is the window function, m is the sample offset for the time segment, k the discrete frequency index and N the transform size.

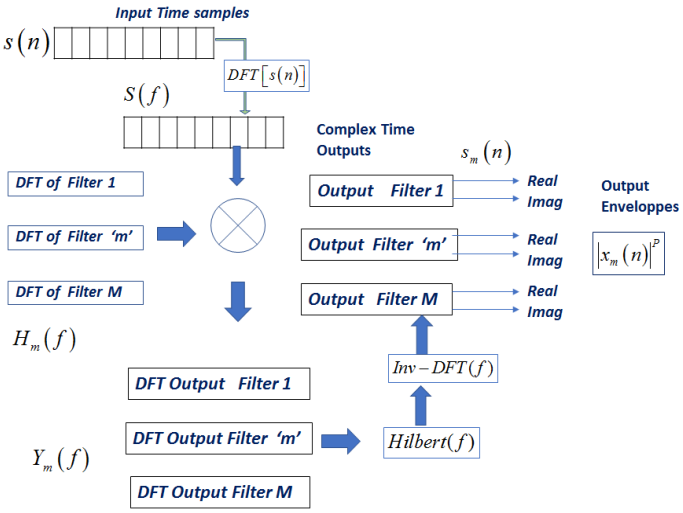

Figure 3: Frequency-based `PyCochleogram`

b. Functional Implementation

In this work a Hanning window and the `Librosa` STFT was used. A 1024 sample FFT was input and the average magnitude was taken over every 8 samples to yield 64 output values per STFT.

4. Network Training

A. Optimizers and Loss Functions

The `Adam` optimizer was used with all input features [20]. The Stochastic Gradient Descent (SGD) was tested but yielded inferior results. The default values for `Adam` parameters were used in most cases except where noted in the results section.

Several loss functions were tested with each input feature and the `MSELoss()` and `SmoothL1Loss()`

functions performed the best. The other loss functions tested were `CrossEntropyLoss`(), `NLLLoss`(), and `BCEWithLogistsLoss`(). All loss functions came from the `torch.nn` package [21].

5. Datasets

Publically available data sets were used for all samples. Because the speech files included some gaps, any segment without speech longer than 0.5 second was removed.

All data which was not single channel and 16 KHz sampling rate was converted to this format. Each transform processed 1024 samples to generate the 64 input feature values to the network. Subsequent samples were taken with a hop size of 512, thereby having each sample overlap with its neighbours by 50%. The 64 samples in each input spanned 2.048 seconds.

Table 1 shows the training set as created from the VOX I 'Male' and 'Female' samples [22] and a combination of 99 Sound Effects [23], ESC-50 [24] and non-vocal music.

Dataset	Files	samples
Vox 1 (2 Minute files) Female	15	1740
Vox 1 (2 Minute files) Male	25	1740
Vox 1 (5 Minute files) Female	15	4365
Vox 1 (5 Minute files) Male	25	4365
99 Sound Effects	99	982
ESC-50	1980	5940
NonVocalMusic (instrumental)	1	195

Table 1: Training Set

Because there were more 'Male' than 'Female' samples, the 'Male' samples were selected randomly from the total to end up with the same number for each. The final Training set was comprised of 6105 'Male', 6105 'Female', and 7117 Neither samples.

The validation set was generated from a subset of VOX 1 with different speakers from the training set and non-speech files from the BBC Sound Effects [25] site and is shown in Table 2.

Dataset	Files	Samples
BBC Sound Effects	72	5606
Vox 1 (10 Minute files) Female	4	2100
Vox 1 (10 Minute files) Male	4	2100

Table 2: Validation Set

Because there were a different number of 'Male' and 'Female' samples, 2100 were selected randomly from each to obtain the final set. The final Validation set included 2100 'Male', 2100 'Female', and 5606 Neither samples.

The Test set was generated from the `Librispeech` Test set [26] for speech samples and samples from the BBC Sound Effects not used in the Validation set and is listed in Table 3.

Dataset	Files	Samples
BBC Sound Effects	95	7335
Librispeech Test set	40	14133

Table 3: Test Set

The final Test set included approximately 7000 'Male', 7000 'Female', and 7335 'Neither' samples.

6. Results

Each of the data sets was processed for each input feature transform to create transformed training, validation, and testing datasets for the network. During the training of each input feature, different optimizers, loss functions, batch normalization, and input normalization were tried. Variations were seen in successive trainings so when the best performing model for each input feature transform was selected, ten additional trainings were made for each configuration to average out variations as reported below.

Training times for each input feature type were essentially the same because the networks were identical and the differences in the features were in

how the features were computed. Each training was run for 200 epochs, and the model with the lowest validation loss was chosen. Training took approximately 6.6 seconds per epoch on an NVIDIA RTX 2080.

For each model a count of True Positive (TP) True Negative (TN), False Positive (FP), and False Negative (FN) was kept. From these counts, metrics were computed as:

$$precision = \frac{TP}{TP + FP} \tag{4}$$

$$recall = \frac{TP}{TP + FN} \tag{5}$$

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{6}$$

$$F1 = 2\frac{precision \times recall}{precision + recall} \tag{7}$$

Each model was evaluated using the test data set for 'Precision', 'Recall', 'Accuracy' and 'F1' score. Eleven models were trained and the average scores are given below.

A. Melgram

The model trained with the Melgram transformed data gave the best results using the `MSE` loss and `Adam` optimizer with a learning rate of `0.0002`, betas were (`0.95, 0.999`), and `weight_decay` of `1e-06`. All other parameters were the default values. Averaged test results for the Melgram based models are given in Table 4.

Class	Precision	Recall	Accuracy	F1
Male	0.8680	0.8615	0.9123	0.8644
Female	0.8988	0.8416	0.9181	0.8689
Neither	0.9141	0.9745	0.9587	0.9430

Table 4: Melgram Results

B. Chromagram

The model trained with the Chromagram transformed data gave the best results with the `SmoothL1Loss()` and `Adam` optimizer with the learning rate equal to `0.0001`. All other parameters were the default values. Table 5 shows testing results from the Chromagram models.

Class	Precision	Recall	Accuracy	F1
Male	0.8696	0.8208	0.8998	0.8440
Female	0.8626	0.8671	0.9109	0.8645
Neither	0.9401	0.9850	0.9734	0.8902

Table 5. Chromagram Results

C. Cochleogram

The model trained with the Cochleogram transform data gave the best results with the `SmoothL1Loss` and Batch Normalization. All `Adam` optimizer parameters were the default values, except the learning rate was `0.0001`. Test results from the Cochleogram trained models are in Table 6.

Class	Precision	Recall	Accuracy	F1
Male	0.8684	0.9890	0.9464	0.9245
Female	0.9911	0.8455	0.9468	0.9120
Neither	0.9933	0.9983	0.9971	0.9958

Table 6. Cochleogram Results

D. STFT

The model trained by the STFT transformed data which gave the best results used the `SmoothL1Loss()` function, Batch Normalization was applied to each convolutional layer and the `Adam` optimizer with all default values. Table 7 provides the test results from the STFT model.

Class	Precision	Recall	Accuracy	F1
Male	0.8888	0.9695	0.9496	0.9271
Female	0.9672	0.8729	0.9485	0.9172
Neither	0.9941	0.9983	0.9973	0.9962

Table 7: STFT Results

E. Comparison

The average F1 scores for all models are given in Table 8.

Melgram	Chromagram	Cochleogram	STFT
0.8921	0.8902	0.9441	0.9468

Table 8: F1 Score Comparison

With the STFT and Cochleogram based models performing the best with the Chromagram and Melgram at the bottom. Because there was significant variation between trainings, all of the averaged F1 scores are shown in Figure 4.

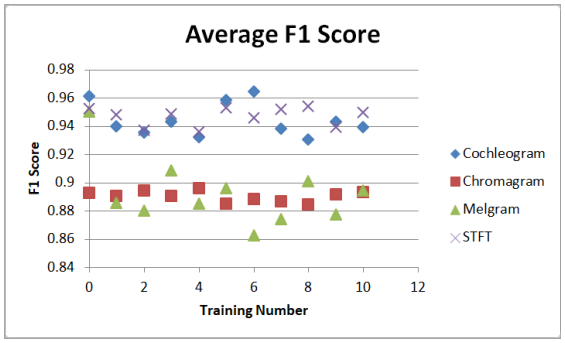

Figure 4: Average F1 Scores for all Trainings

The single best performing model was based on the Cochleogram input features. In the classical thinking of audio signal processing, it would seem that analysing the pitch excursions and ranges would be sufficient to differentiate between male and female voices. In this line of reasoning, any TF representation would be equally informative in that it clearly shows the pitch harmonics across time. The results of this study suggest that other patterns in the TF domain are valuable in achieving an accurate classification. The Cochleogram, being modelled after the human cochlea, seems to highlight such patterns better than any other representation.

7. Conclusion

This paper evaluated neural network performance when a model is trained with different spectral representations of an audio signal. The evaluation was done with a gender detection model based on a VGG style CNN. The Mel-Spectrogram, Chromagram, Cochleogram, and STFT were used to create the input features to the network. The results show that for this model, the Cochleogram and STFT features performed the best.

The issue of what constitutes an optimal set of input features for an audio neural network is not a simple one. It is worth evaluating alternatives to the commonly used Mel-Spectrogram, for the specific application and the network structure sought.

References

[1] K. Prahallad. “Spectrogram, Cepstrum and Mel-Frequency Analysis”. http://www.speech.cs.cmu.edu/15-492/slides/03_mfcc.pdf.

[2] Chroma Feature. https://en.wikipedia.org/wiki/Chroma_feature

[3] R. F. Lyon, "A Computational Model of Filtering, Detection, and Compression in the Cochlea", in *IEEE International Conference on Acoustics, Speech and Signal Processing* (*ICASSP*), pp. 1282-1285. (1982).

[4] R. Martin, “Noise Power Spectral Density Estimation Based on Optimal Smoothing and Minimum Statistics,” *IEEE Transactions on Speech and Audio Processing*, vol. 9, no. 5, pp. 504–512, 2001.

[5] B. Gao, W. Woo, L.C. Khor. “Cochleagram-based Audio Pattern Separation Using Two-dimensional Non-Negative Matrix Factorization with Automatic Sparsity Adaptation”. *Journal of Acoustical Society of America*. 2014 Mar;135(3):1171-85. doi: 10.1121/1.4864294. PMID: 24606260.

[6] R. V. Sharan, T. J. Moir, “Acoustic Event Recognition Using Cochleagram Image and Convolutional Neural Networks”, *Applied Acoustics*, Volume 148, 2019, Pages 62-66, ISSN 0003-682X,

[7] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, ``WaveNet: A generative Model for Raw Audio," 2016, arXiv:1609.03499. [Online]. Available: http://arxiv.org/abs/1609.03499

[8] http://dcase.community/challenge2020/task-acoustic-scene-classification-results-b

[9] “Low-Complexity Acoustic Scene Classification.” DCASE 2020, http://dcase.community/challenge2020/task-acoustic-scene-classification-results-b.

[10] E. Nemer, "Audio Cochleogram with Analysis and Synthesis Banks Using 1D Convolutional Networks", in *1st International Conference on Artificial Intelligence and Data Analytics* (CAIDA), 2021, doi: 10.1109/CAIDA51941.2021.9425342.

[11] E. Nemer, G. Vines. “1D Convolutional Layers to Create Frequency-Based Spectral Features for Audio Networks”, *AES 153rd convention*. 2022.

[12] A. Paszke *et al*. “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In H. Wallach *et al*., eds. Advances in Neural Information Processing Systems 32. Curran Associates, Inc., pp. 8024–8035. 2019 Available

at: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.
[13] https://github.com/pytorch/vision/blob/master/torchvision/models/vgg.py
[14] S. Umesh, L. Cohen, and D. Nelson, ``Fitting the Mel scale," in Proc. *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 1, Mar. 1999, pp. 217-220.
[15] D. O'Shaughnessy, Speech Communications: Human and Machine. New York, NY, USA: Wiley, 1987.
[16] H. Fayek "Speech Processing for Machine Learning", online at: https://haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html.
[17] M. A. Bartsch and G. H. Wakefield, "Audio Thumbnailing of Popular Music Using Chroma-based Representations," in *IEEE Transactions on Multimedia*, vol. 7, no. 1, pp. 96-104, Feb. 2005, doi: 10.1109/TMM.2004.840597.
[18] E. Gómez, Emilia. "Tonal Description of Music Audio Signals". PHD Thesis, UPF Barcelona, Spain. 2006.
[19] R. Gonzales. "Pycochleogram: Generate Cochleagrams Natively in Python." Documentation and code available online at: https://pycochleagram.readthedocs.io/en/latest/index.html.
[20] D. P. Kingma, J. Ba, "Adam: A Method for Stochastic Optimization", online at: https://arxiv.org/abs/1412.6980
[21] https://pytorch.org/docs/stable/optim.html
[22] https://www.robots.ox.ac.uk/~vgg/data/voxceleb/
[23] https://99sounds.org/
[24] https://www.researchgate.net/publication/305854186_ESC_Dataset_for_Environmental_Sound_Classification
[25] https://sound-effects.bbcrewind.co.uk/
[26] https://www.openslr.org/12