SPEAKER RECOGNITION FROM RAW WAVEFORM WITH SINCNET

*Mirco Ravanelli, Yoshua Bengio**

Mila, Université de Montréal, *CIFAR Fellow

ABSTRACT

Deep learning is progressively gaining popularity as a viable alternative to i-vectors for speaker recognition. Promising results have been recently obtained with Convolutional Neural Networks (CNNs) when fed by raw speech samples directly. Rather than employing standard hand-crafted features, the latter CNNs learn low-level speech representations from waveforms, potentially allowing the network to better capture important narrow-band speaker characteristics such as pitch and formants. Proper design of the neural network is crucial to achieve this goal.

This paper proposes a novel CNN architecture, called *SincNet*, that encourages the first convolutional layer to discover more meaningful filters. SincNet is based on parametrized sinc functions, which implement band-pass filters. In contrast to standard CNNs, that learn all elements of each filter, only low and high cutoff frequencies are directly learned from data with the proposed method. This offers a very compact and efficient way to derive a customized filter bank specifically tuned for the desired application.

Our experiments, conducted on both speaker identification and speaker verification tasks, show that the proposed architecture converges faster and performs better than a standard CNN on raw waveforms.

Index Terms— speaker recognition, convolutional neural networks, raw samples.

1. INTRODUCTION

Speaker recognition is a very active research area with notable applications in various fields such as biometric authentication, forensics, security, speech recognition, and speaker diarization, which has contributed to steady interest towards this discipline [1]. Most state-of-the-art solutions are based on the i-vector representation of speech segments [2], which contributed to significant improvements over previous Gaussian Mixture Model-Universal Background Models (GMM-UBMs) [3]. Deep learning has shown remarkable success in numerous speech tasks [4–7], including recent studies in speaker recognition [8, 9]. Deep Neural Networks (DNNs) have been used within the i-vector framework to compute Baum-Welch statistics [10], or for frame-level feature extraction [11]. DNNs have also been proposed for direct discriminative speaker classification, as witnessed by the recent literature on this topic [12–15]. Most of past attempts, however, employed hand-crafted features such as FBANK and MFCC coefficients [12, 16, 17]. These engineered features are originally designed from perceptual evidence and there are no guarantees that such representations are optimal for all speech-related tasks. Standard features, for instance, smooth the speech spectrum, possibly hindering the extraction of crucial narrow-band speaker characteristics such as pitch and formants. To mitigate this drawback, some recent works have proposed directly feeding the network with spectrogram bins [18–20] or even with raw waveforms [21–30]. CNNs are the most popular architecture for processing raw speech samples, since weight sharing, local filters, and pooling help discover robust and invariant representations.

We believe that one of the most critical part of current waveform-based CNNs is the first convolutional layer. This layer not only deals with high-dimensional inputs, but is also more affected by vanishing gradient problems, especially when employing very deep architectures. The filters learned by the CNN often take noisy and incongruous multi-band shapes, especially when few training samples are available. These filters certainly make some sense for the neural network, but do not appeal to human intuition, nor appear to lead to an efficient representation of the speech signal.

To help the CNNs discover more meaningful filters in the input layer, this paper proposes to add some constraints on their shape. Compared to standard CNNs, where the filter-bank characteristics depend on several parameters (each element of the filter vector is directly learned), the SincNet convolves the waveform with a set of parametrized sinc functions that implement band-pass filters. The low and high cutoff frequencies are the only parameters of the filter learned from data. This solution still offers considerable flexibility, but forces the network to focus on high-level tunable parameters with broad impact on the shape and bandwidth of the resulting filter.

Our experiments are carried out under challenging but realistic conditions, characterized by minimal training data (i.e., 12-15 seconds for each speaker) and short test sentences (lasting from 2 to 6 seconds). Results achieved on a variety of datasets, show that the proposed SincNet converges faster and achieves better end task performance than a more standard CNN. Under the considered experimental setting, our archi-

tecture also outperforms a more traditional speaker recognition system based on i-vectors.

The remainder of the paper is organized as follows. The SincNet architecture is described in Sec. 2. Sec. 3 discusses the relation to prior work. The experimental setup and results are outlined in Sec. 4 and Sec. 5 respectively. Finally, Sec. 6 discusses our conclusions.

2. THE SINCNET ARCHITECTURE

The first layer of a standard CNN performs a set of time-domain convolutions between the input waveform and some Finite Impulse Response (FIR) filters [31]. Each convolution is defined as follows[1]:

$$y[n] = x[n] * h[n] = \sum_{l=0}^{L-1} x[l] \cdot h[n-l] \quad (1)$$

where $x[n]$ is a chunk of the speech signal, $h[n]$ is the filter of length L, and $y[n]$ is the filtered output. In standard CNNs, all the L elements (taps) of each filter are learned from data. Conversely, the proposed SincNet (depicted in Fig. 1) performs the convolution with a predefined function g that depends on few learnable parameters θ only, as highlighted in the following equation:

$$y[n] = x[n] * g[n, \theta] \quad (2)$$

A reasonable choice, inspired by standard filtering in digital signal processing, is to define g such that a filter-bank composed of rectangular bandpass filters is employed. In the frequency domain, the magnitude of a generic bandpass filter can be written as the difference between two low-pass filters:

$$G[f, f_1, f_2] = rect\Big(\frac{f}{2f_2}\Big) - rect\Big(\frac{f}{2f_1}\Big), \quad (3)$$

where f_1 and f_2 are the learned low and high cutoff frequencies, and $rect(\cdot)$ is the rectangular function in the magnitude frequency domain[2]. After returning to the time domain (using the inverse Fourier transform [31]), the reference function g becomes:

$$g[n, f_1, f_2] = 2f_2 sinc(2\pi f_2 n) - 2f_1 sinc(2\pi f_1 n), \quad (4)$$

where the sinc function is defined as $sinc(x) = sin(x)/x$.

The cut-off frequencies can be initialized randomly in the range $[0, f_s/2]$, where f_s represents the sampling frequency of the input signal. As an alternative, filters can be initialized with the cutoff frequencies of the mel-scale filter-bank, which has the advantage of directly allocating more filters in the lower part of the spectrum, where many crucial clues about the speaker identity are located. To ensure $f_1 \geq 0$ and $f_2 \geq f_1$, the previous equation is actually fed by the following parameters:

$$f_1^{abs} = |f_1| \quad (5)$$

$$f_2^{abs} = f_1 + |f_2 - f_1| \quad (6)$$

Note that no bounds have been imposed to force f_2 to be smaller than the Nyquist frequency, since we observed that this constraint is naturally fulfilled during training. Moreover, the gain of each filter is not learned at this level. This parameter is managed by the subsequent layers, which can easily attribute more or less importance to each filter output.

An ideal bandpass filter (i.e., a filter where the passband is perfectly flat and the attenuation in the stopband is infinite) requires an infinite number of elements L. Any truncation of g thus inevitably leads to an approximation of the ideal filter, characterized by ripples in the passband and limited attenuation in the stopband. A popular solution to mitigate this issue is windowing [31]. Windowing is performed by multiplying the truncated function g with a window function w, which

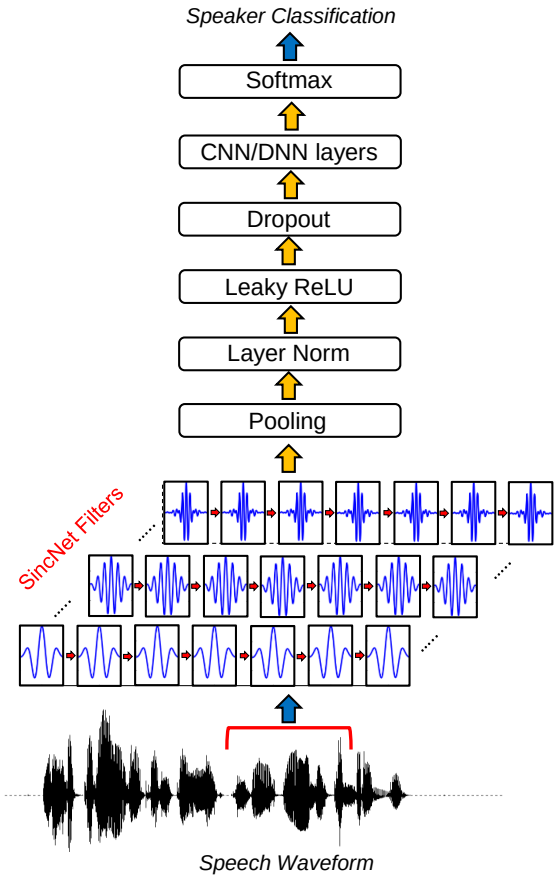


Fig. 1: Architecture of SincNet.

[1]Most deep learning toolkits actually compute *correlation* rather than *convolution*. The obtained flipped (mirrored) filters do not affect the results.

[2]The phase of the $rect(\cdot)$ function is considered to be linear.

aims to smooth out the abrupt discontinuities at the ends of g:

$$g_w[n, f_1, f_2] = g[n, f_1, f_2] \cdot w[n]. \quad (7)$$

This paper uses the popular Hamming window [32], defined as follows:

$$w[n] = 0.54 - 0.46 \cdot cos\Big(\frac{2\pi n}{L}\Big). \quad (8)$$

The Hamming window is particularly suitable to achieve high frequency selectivity [32]. However, results not reported here reveals no significant performance difference when adopting other functions, such as Hann, Blackman and Kaiser windows.

All operations involved in SincNet are fully differentiable and the cutoff frequencies of the filters can be jointly optimized with other CNN parameters using Stochastic Gradient Descent (SGD) or other gradient-based optimization routines. As shown in Fig. 1, a standard CNN pipeline (pooling, normalization, activations, dropout) can be employed after the first sinc-based convolution. Multiple standard convolutional or fully-connected layers can then be stacked together to finally perform a speaker classification with a softmax classifier.

2.1. Model properties

The proposed SincNet has some remarkable properties:

- **Fast Convergence:** SincNet forces the network to focus only on the filter parameters with major impact on performance. The proposed approach actually implements a natural inductive bias, utilizing knowledge about the filter shape (similar to feature extraction methods generally deployed on this task) while retaining flexibility to adapt to data. This prior knowledge makes learning the filter characteristics much easier, helping SincNet to converge significantly faster to a better solution.

- **Few Parameters:** SincNet drastically reduces the number of parameters in the first convolutional layer. For instance, if we consider a layer composed of F filters of length L, a standard CNN employs $F \cdot L$ parameters, against the $2F$ considered by SincNet. If $F = 80$ and $L = 100$, we employ 8k parameters for the CNN and only 160 for SincNet. Moreover, if we double the filter length L, a standard CNN doubles its parameter count (e.g., we go from 8k to 16k), while SincNet has an unchanged parameter count (only two parameters are employed for each filter, regardless its length L). This offers the possibility to derive very selective filters with many taps, without actually adding parameters to the optimization problem. Moreover, the compactness of the SincNet architecture makes it suitable in the few sample regime.

- **Computational Efficiency**: The proposed function g is symmetric. This means we can perform convolution in a very efficient way by only considering one side of the filter and inheriting the results for the other half. This saves 50% of the first-layer computation over a standard CNN.

- **Interpretability**: The SincNet feature maps obtained in the first convolutional layer are definitely more interpretable and human-readable than other approaches. The filter bank, in fact, only depends on parameters with a clear physical meaning.

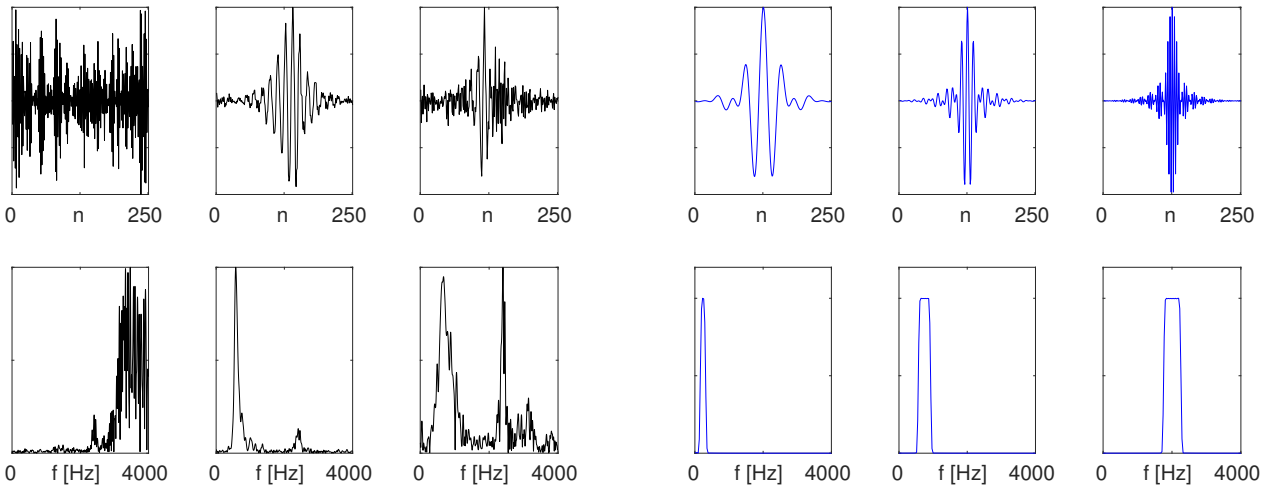
3. RELATED WORK

Several works have recently explored the use of low-level speech representations to process audio and speech with CNNs. Most prior attempts exploit magnitude spectrogram features [18–20, 33–35]. Although spectrograms retain more information than standard hand-crafted features, their design still requires careful tuning of some crucial hyper-parameters, such as the duration, overlap, and typology of the frame window, as well as the number of frequency bins. For this reason, a more recent trend is to directly learn from raw waveforms, thus completely avoiding any feature extraction step. This approach has shown promise in speech [21–25], including emotion tasks [26], speaker recognition [28], spoofing detection [27], and speech synthesis [29, 30]. Similar to SincNet, some previous works have proposed to add constraints on the CNN filters, for instance forcing them to work on specific bands [33, 34]. Differently from the proposed approach, the latter works operate on spectrogram features and still learn all the L elements of the CNN filters. An idea related to the proposed method has been recently explored in [35], where a set of parameterized Gaussian filters are employed. This approach operates on the spectrogram domain, while SincNet directly considers raw time domain waveform.

To the best of our knowledge, this study is the first to show the effectiveness of the proposed sinc filters for time-domain audio processing from raw waveforms using convolutional neural networks. Several past works target speech recognition, while our study specifically considers a speaker recognition application. The compact filters learned by SincNet are particularly suitable for speaker recognition tasks, especially in a realistic scenario characterized by few seconds of training data for each speaker and short sentences for testing.

4. EXPERIMENTAL SETUP

The proposed SincNet has been evaluated on different corpora and compared to numerous speaker recognition baselines. In the spirit of reproducible research, we perform most experiments using publicly available data such as Librispeech, and



(a) CNN Filters

(b) SincNet Filters

Fig. 2: Examples of filters learned by a standard CNN and by the proposed SincNet (using the Librispeech corpus). The first row reports the filters in the time domain, while the second one shows their magnitude frequency response.

release the code of SincNet on GitHub[3]. In the following sections, an overview of the experimental settings is provided.

4.1. Corpora

To provide experimental evidence on datasets characterized by different numbers of speakers, this paper considers the TIMIT (462 spks, *train* chunk) [36] and Librispeech (2484 spks) [37] corpora. Non-speech intervals at the beginning and end of each sentence were removed. The Librispeech sentences with internal silences lasting more than 125 ms were split into multiple chunks. To address text-independent speaker recognition, the calibration sentences of TIMIT (i.e., the utterances with the same text for all speakers) have been removed. For the latter dataset, five sentences for each speaker were used for training, while the remaining three were used for test. For the Librispeech corpus, the training and test material have been randomly selected to exploit 12-15 seconds of training material for each speaker and test sentences lasting 2-6 seconds.

4.2. SincNet Setup

The waveform of each speech sentence was split into chunks of 200 ms (with 10 ms overlap), which were fed into the SincNet architecture. The first layer performs sinc-based convolutions as described in Sec. 2, using 80 filters of length $L = 251$ samples. The architecture then employs two standard convolutional layers, both using 60 filters of length 5. Layer normalization [38] was used for both the input samples and for all convolutional layers (including the SincNet input layer). Next, three fully-connected layers composed of 2048 neurons and normalized with batch normalization [39] were applied. All hidden layers use leaky-ReLU [40] non-linearities. The parameters of the sinc-layer were initialized using mel-scale cutoff frequencies, while the rest of the network was initialized with the well-known "Glorot" initialization scheme [41]. Frame-level speaker classification was obtained by applying a softmax classifier, providing a set of posterior probabilities over the targeted speakers. A sentence-level classification was simply derived by averaging the frame predictions and voting for the speaker which maximizes the average posterior.

Training used the RMSprop optimizer, with a learning rate $lr = 0.001$, $\alpha = 0.95$, $\epsilon = 10^{-}7$, and minibatches of size 128. All the hyper-parameters of the architecture were tuned on TIMIT, then inherited for Librispeech as well.

The speaker verification system was derived from the speaker-id neural network considering two possible setups. First, we consider the *d-vector* framework [12, 20], which relies on the output of the last hidden layer and computes the cosine distance between test and the claimed speaker d-vectors. As an alternative solution (denoted in the following as *DNN-class*), the speaker verification system can directly take the softmax posterior score corresponding to the claimed identity. The two approaches will be compared in Sec. 5.

To perform an accurate evaluation, 10 utterances from impostors were randomly selected for each sentence coming from a genuine speaker. Impostors were taken from a speaker pool different from that used for training the speaker id network.

[3] at `https://github.com/mravanelli/SincNet/`.

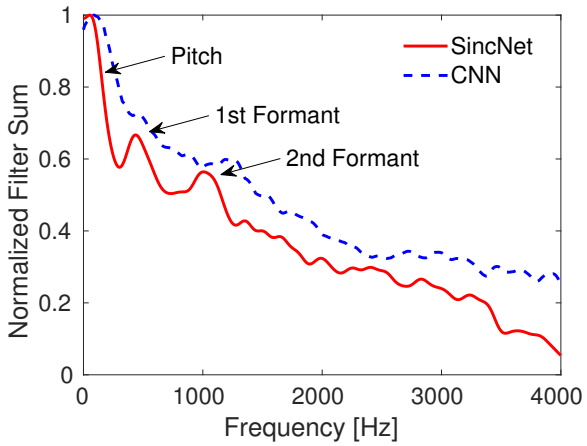


Fig. 3: Cumulative frequency response of the SincNet filters.

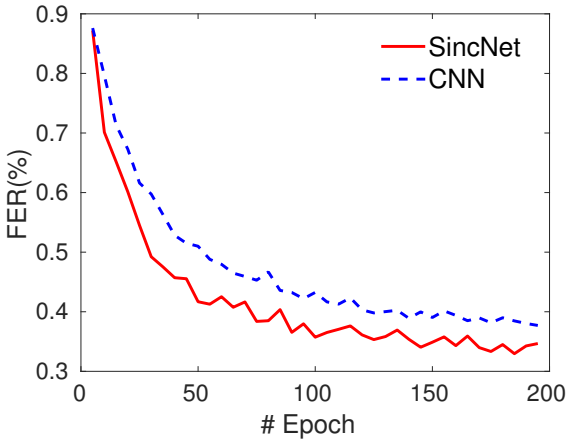


Fig. 4: Frame Error Rate (%) of SincNet and CNN models over various training epochs. Results are reported on TIMIT.

4.3. Baseline Setups

We compared SincNet with several alternative systems. First, we considered a standard CNN fed by the raw waveform. This network is based on the same architecture as SincNet, but replacing the sinc-based convolution with a standard one.

A comparison with popular hand-crafted features was also performed. To this end, we computed 39 MFCCs (13 static+Δ+$\Delta\Delta$) and 40 FBANKs using the Kaldi toolkit [42]. These features, computed every 25 ms with 10 ms overlap, were gathered to form a context window of approximately 200 ms (i.e., a context similar to that of the considered waveform-based neural network). A CNN was used for FBANK features, while a Multi-Layer Perceptron (MLP) was used for MFCCs[4]. Layer normalization was used for the FBANK network, while batch normalization was employed for the MFCC one. The hyper-parameters of these networks were also tuned using the aforementioned approach.

For speaker verification experiments, we also considered an i-vector baseline. The i-vector system was implemented with the SIDEKIT toolkit [43]. The GMM-UBM model, the Total Variability (TV) matrix, and the Probabilistic Linear Discriminant Analysis (PLDA) were trained on the Librispeech data (avoiding test and enrollment sentences). GMM-UBM was composed of 2048 Gaussians, and the rank of the TV and PLDA eigenvoice matrix was 400. The enrollment and test phase is conducted on Librispeech using the same set of speech segments used for DNN experiments.

5. RESULTS

This section reports the experimental validation of the proposed SincNet. First, we perform a comparison between the filters learned by a SincNet and by a standard CNN. We then compare our architecture with other competitive systems on both speaker identification and verification tasks.

5.1. Filter Analysis

Inspecting the learned filters is a valuable practice that provides insight into what the network is actually learning. Fig. 2 shows some examples of filters learned by a standard CNN (Fig. 2a) and by the proposed SincNet (Fig. 2b) using the Librispeech dataset (the frequency response is plotted between 0 and 4 kHz). As observed in the figures, the standard CNN does not always learn filters with a well-defined frequency response. In some cases the frequency response looks noisy (see the first filter of Fig. 2a), while in others assuming multi-band shapes (see the third filter of the CNN plot). SincNet, instead, is specifically designed to implement rectangular band-pass filters, leading to more meaningful CNN filters.

Beyond a qualitative inspection, it is important to highlight which frequency bands are covered by the learned filters. Fig. 3 shows the cumulative frequency response of the filters learned by SincNet and CNN. Interestingly, there are three main peaks which clearly stand out from the SincNet plot (see the red line in the figure). The first one corresponds to the pitch region (the average pitch is 133 Hz for a male and 234 for a female). The second peak (approximatively located at 500 Hz) mainly captures first formants, whose average value over the various English vowels is indeed 500 Hz. Finally, the third peak (ranging from 900 to 1400 Hz) captures some important second formants, such as the second formant of the vowel /a/, which is located on average at 1100 Hz. This filter-bank configuration indicates that SincNet has successfully adapted its characteristics to address speaker identification. Conversely, the standard CNN does not exhibit such a meaningful pattern: the CNN filters tend to correctly focus on the lower part of the spectrum, but peaks tuned on first and

[4]CNNs exploit local correlation across features and cannot be effectively used with uncorrelated MFCC features.

second formants do not clearly appear. As one can observe from Fig. 3, the CNN curve stands above the SincNet one. SincNet, in fact, learns filters that are, on average, more selective than CNN ones, possibly better capturing narrow-band speaker clues.

5.2. Speaker Identification

Fig. 4 shows the learning curves of SincNet compared with that of a standard CNN. These results, achieved on the TIMIT dataset, highlight a faster decrease of the Frame Error Rate ($FER\%$) when SincNet is used. Moreover, SincNet converges to better performance leading to a FER of 33.0% against a FER of 37.7% achieved with the CNN baseline.

	TIMIT	LibriSpeech
DNN-MFCC	0.99	2.02
CNN-FBANK	0.86	1.55
CNN-Raw	1.65	1.00
SINCNET	**0.85**	**0.96**

Table 1: Sentence Error Rate (SER%) of speaker identification systems trained on TIMIT (462 spks) and Librispeech (2484 spks) datasets. SincNets outperform the competing alternatives.

Table 1 reports the achieved Sentence Error Rates (SER%). The table shows that SincNet outperforms other systems on both TIMIT and Librispeech datasets. The gap with a standard CNN fed by raw waveform is particularly large on TIMIT, confirming the effectiveness of SincNet when few training data are available. Although this gap is reduced when LibriSpeech is used, we still observe a 4% relative improvement that is also obtained with faster convergence (1200 vs 1800 epochs). Standard FBANKs provide results comparable to SincNet only on TIMIT, but are significantly worse than our architecture when using Librispech. With few training data, the network cannot discover filters much better than FBANKs, but with more data a customized filter-bank is learned and exploited to improve the performance.

5.3. Speaker Verification

As a last experiment, we extend our validation to speaker verification. Table 2 reports the Equal Error Rate (EER%) achieved with the Librispeech corpus. All DNN models show promising performance, leading to an EER lower than 1% in all cases. The table also highlights that SincNet outperforms the other models, showing a relative performance improvement of about 11% over the standard CNN model. *DNN-class* models perform significantly better than d-vectors. Despite the effectiveness of the later approach, a novel DNN model must be trained (or fine-tuned) for each new speaker added into the pool [28]. This makes this approach better performing, but less flexible than d-vectors.

	d-vector	DNN-class
DNN-MFCC	0.88	0.72
CNN-FBANK	0.60	0.37
CNN-Raw	0.58	0.36
SINCNET	**0.51**	**0.32**

Table 2: Speaker Verification Equal Error Rate (EER%) on Librispeech datasets over different systems. SincNets outperform the competing alternatives.

For the sake of completeness, experiments have also been conducted with standard i-vectors. Although a detailed comparison with this technology is out of the scope of this paper, it is worth noting that our best i-vector system achieves a EER=1.1%, rather far from what achieved with DNN systems. It is well-known in the literature that i-vectors provide competitive performance when more training material is used for each speaker and when longer test sentences are employed [44–46]. Under the challenging conditions faced in this work, neural networks achieve better generalization.

6. CONCLUSIONS AND FUTURE WORK

This paper proposed SincNet, a neural architecture for directly processing waveform audio. Our model, inspired by the way filtering is conducted in digital signal processing, imposes constraints on the filter shapes through efficient parameterization. SincNet has been extensively evaluated on challenging speaker identification and verification tasks, showing performance benefits for all considered corpora.

Beyond performance improvements, SincNet also significantly improves convergence speed over a standard CNN, and is more computationally efficient due to exploitation of filter symmetry. Analysis of the SincNet filters reveals that the learned filter-bank is tuned to precisely extract some known important speaker characteristics, such as pitch and formants. In future work, we would like to evaluate SincNet on other popular speaker recognition tasks, such as VoxCeleb. Although this study targeted speaker recognition only, we believe that the proposed approach defines a general paradigm to process time-series and can be applied in numerous other fields. Our future effort will be thus devoted to extending to other tasks, such as speech recognition, emotion recognition, speech separation, and music processing.

Acknowledgement

We would like to thank Gautam Bhattacharya, Kyle Kastner, Titouan Parcollet, Dmitriy Serdyuk, Maurizio Omologo, and Renato De Mori for their helpful comments. This research was enabled in part by support provided by Calcul Québec and Compute Canada.

7. REFERENCES

[1] H. Beigi, *Fundamentals of Speaker Recognition*, Springer, 2011.

[2] N. Dehak, P. J. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2011.

[3] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted Gaussian mixture models," *Digital Signal Processing*, vol. 10, no. 1–3, pp. 19–41, 2000.

[4] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, MIT Press, 2016.

[5] D. Yu and L. Deng, *Automatic Speech Recognition - A Deep Learning Approach*, Springer, 2015.

[6] G. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large vocabulary speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, no. 1, pp. 30–42, 2012.

[7] M. Ravanelli, *Deep learning for Distant Speech Recognition*, PhD Thesis, Unitn, 2017.

[8] M. McLaren, Y. Lei, and L. Ferrer, "Advances in deep neural network approaches to speaker recognition," in *Proc. of ICASSP*, 2015, pp. 4814–4818.

[9] F. Richardson, D. Reynolds, and N. Dehak, "Deep neural network approaches to speaker and language recognition," *IEEE Signal Processing Letters*, vol. 22, no. 10, pp. 1671–1675, 2015.

[10] P. Kenny, V. Gupta, T. Stafylakis, P. Ouellet, and J. Alam, "Deep neural networks for extracting baum-welch statistics for speaker recognition," in *Proc. of Speaker Odyssey*, 2014.

[11] S. Yaman, J. W. Pelecanos, and R. Sarikaya, "Bottleneck features for speaker recognition," in *Proc. of Speaker Odyssey*, 2012, pp. 105–108.

[12] E. Variani, X. Lei, E. McDermott, I. L. Moreno, and J. Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," in *Proc. of ICASSP*, 2014, pp. 4052–4056.

[13] G. Heigold, I. Moreno, S. Bengio, and N. Shazeer, "End-to-end text-dependent speaker verification," in *Proc. of ICASSP*, 2016, pp. 5115–5119.

[14] D. Snyder, P. Ghahremani, D. Povey, D. Romero, Y. Carmiel, and S. Khudanpur, "Deep neural network-based speaker embeddings for end-to-end speaker verification," in *Proc. of SLT*, 2016, pp. 165–170.

[15] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," in *Proc. of ICASSP*, 2018.

[16] F. Richardson, D. A. Reynolds, and N. Dehak, "A unified deep neural network for speaker and language recognition," in *Proc. of Interspeech*, 2015, pp. 1146–1150.

[17] D. Snyder, D. Garcia-Romero, D. Povey, and S. Khudanpur, "Deep neural network embeddings for text-independent speaker verification," in *Proc. of Interspeech*, 2017, pp. 999–1003.

[18] C. Zhang, K. Koishida, and J. Hansen, "Text-independent speaker verification based on triplet convolutional neural network embeddings," *IEEE/ACM Trans. Audio, Speech and Lang. Proc.*, vol. 26, no. 9, pp. 1633–1644, 2018.

[19] G. Bhattacharya, J. Alam, and P. Kenny, "Deep speaker embeddings for short-duration speaker verification," in *Proc. of Interspeech*, 2017, pp. 1517–1521.

[20] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: a large-scale speaker identification dataset," in *Proc. of Interspech*, 2017.

[21] D. Palaz, M. Magimai-Doss, and R. Collobert, "Analysis of CNN-based speech recognition system using raw speech as input," in *Proc. of Interspeech*, 2015.

[22] T. N. Sainath, R. J. Weiss, A. W. Senior, K. W. Wilson, and O. Vinyals, "Learning the speech front-end with raw waveform CLDNNs," in *Proc. of Interspeech*, 2015.

[23] Y. Hoshen, R. Weiss, and K. W. Wilson, "Speech acoustic modeling from raw multichannel waveforms," in *Proc. of ICASSP*, 2015.

[24] T. N. Sainath, R. J. Weiss, K. W. Wilson, A. Narayanan, M. Bacchiani, and A. Senior, "Speaker localization and microphone spacing invariant acoustic modeling from raw multichannel waveforms," in *Proc. of ASRU*, 2015.

[25] Z. Tüske, P. Golik, R. Schlüter, and H. Ney, "Acoustic modeling with deep neural networks using raw time signal for LVCSR," in *Proc. of Interspeech*, 2014.

[26] G. Trigeorgis, F. Ringeval, R. Brueckner, E. Marchi, M. A. Nicolaou, B. Schuller, and S. Zafeiriou, "Adieu features? end-to-end speech emotion recognition using a deep convolutional recurrent network," in *Proc. of ICASSP*, 2016, pp. 5200–5204.

[27] H. Dinkel, N. Chen, Y. Qian, and K. Yu, "End-to-end spoofing detection with raw waveform CLDNNS," *Proc. of ICASSP*, pp. 4860–4864, 2017.

[28] H. Muckenhirn, M. Magimai-Doss, and S. Marcel, "Towards directly modeling raw speech signal for speaker verification using CNNs," in *Proc. of ICASSP*, 2018.

[29] 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," in *Arxiv*, 2016.

[30] S. Mehri, K. Kumar, I. Gulrajani, R. Kumar, S. Jain, J. Sotelo, A. C. Courville, and Y. Bengio, "Samplernn: An unconditional end-to-end neural audio generation model," *CoRR*, vol. abs/1612.07837, 2016.

[31] L. R. Rabiner and R. W. Schafer, *Theory and Applications of Digital Speech Processing*, Prentice Hall, NJ, 2011.

[32] S. K. Mitra, *Digital Signal Processing*, McGraw-Hill, 2005.

[33] T. N. Sainath, B. Kingsbury, A. R. Mohamed, and B. Ramabhadran, "Learning filter banks within a deep neural network framework," in *Proc. of ASRU*, 2013, pp. 297–302.

[34] H. Yu, Z. H. Tan, Y. Zhang, Z. Ma, and J. Guo, "DNN Filter Bank Cepstral Coefficients for Spoofing Detection," *IEEE Access*, vol. 5, pp. 4779–4787, 2017.

[35] H. Seki, K. Yamamoto, and S. Nakagawa, "A deep neural network integrated with filterbank learning for speech recognition," in *Proc. of ICASSP*, 2017, pp. 5480–5484.

[36] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscus, D. S. Pallett, and N. L. Dahlgren, "DARPA TIMIT Acoustic Phonetic Continuous Speech Corpus CDROM," 1993.

[37] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An ASR corpus based on public domain audio books," in *Proc. of ICASSP*, 2015, pp. 5206–5210.

[38] J. Ba, R. Kiros, and G. E. Hinton, "Layer normalization," *CoRR*, vol. abs/1607.06450, 2016.

[39] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. of ICML*, 2015, pp. 448–456.

[40] A. L. Maas, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *Proc. of ICML*, 2013.

[41] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. of AISTATS*, 2010, pp. 249–256.

[42] D. Povey et al., "The Kaldi Speech Recognition Toolkit," in *Proc. of ASRU*, 2011.

[43] A. Larcher, K. A. Lee, and S. Meignier, "An extensible speaker identification sidekit in python," in *Proc. of ICASSP*, 2016, pp. 5095–5099.

[44] A. K. Sarkar, D Matrouf, P.M. Bousquet, and J.F. Bonastre, "Study of the effect of i-vector modeling on short and mismatch utterance duration for speaker verification," in *Proc. of Interspeech*, 2012, pp. 2662–2665.

[45] R. Travadi, M. Van Segbroeck, and S. Narayanan, "Modified-prior i-Vector Estimation for Language Identification of Short Duration Utterances," in *Proc. of Interspeech*, 2014, pp. 3037–3041.

[46] A. Kanagasundaram, R. Vogt, D. Dean, S. Sridharan, and M. Mason, "i-vector based speaker recognition on short utterances," in *Proc. of Interspeech*, 2011, pp. 2341–2344.