Acoustic Modeling from Frequency-Domain Representations of Speech

Pegah Ghahremani[1], *Hossein Hadian*[1,3], *Hang Lv*[1,4], *Daniel Povey*[1,2], *Sanjeev Khudanpur*[1,2]

[1]Center of Language and Speech Processing
[2]Human Language Technology Center Of Excellence,
Johns Hopkins University, Baltimore, MD
[3]Department of Computer Engineering, Sharif University of Technology, Iran
[4]School of Computer Science, Northwestern Polytechnical University, Xian, China
{pghahre1,hhadian,khudanpur}@jhu.edu, dpovey@gmail.com,hanglv@nwpu-aslp.org

Abstract

In recent years, different studies have proposed new methods for DNN-based feature extraction and joint acoustic model training and feature learning from raw waveform for large vocabulary speech recognition. However, conventional pre-processed methods such as MFCC and PLP are still preferred in the state-of-the-art speech recognition systems as they are perceived to be more robust. Besides, the raw waveform methods – most of which are based on the time-domain signal – do not significantly outperform the conventional methods. In this paper, we propose a frequency-domain feature-learning layer which can allow acoustic model training directly from the waveform. The main distinctions from previous works are a new normalization block and a short-range constraint on the filter weights. The proposed setup achieves consistent performance improvements compared to the baseline MFCC and log-Mel features as well as other proposed time and frequency domain setups on different LVCSR tasks. Finally, based on the learned filters in our feature-learning layer, we propose a new set of analytic filters using polynomial approximation, which outperforms log-Mel filters significantly while being equally fast.

Index Terms: filter bank learning, acoustic modeling

1. Introduction

Feature extraction is a crucial part of automatic speech recognition (ASR) systems. The most commonly used conventional methods for feature extraction are MFCC [1], PLP [2], and log-Mel filter-bank. These are hand-crafted based on physiological models of the human auditory system and are not guaranteed to be optimal for the current DNN-based ASR models. In contrast, data-driven feature extraction methods aim to use the training data to learn a feature extractor. Alternatively, raw waveform acoustic modeling techniques employ deep neural networks to enable joint acoustic modeling and feature extraction.

During recent years, other parts of ASR systems such as acoustic modeling and language modeling have greatly evolved with the advent of deep neural networks. However, data-driven feature extraction methods have not significantly outperformed conventional features on LVCSR tasks. As a result, most state-of-the-art ASR systems still use conventional methods such as MFCC for feature extraction. Part of the reason might be that the data-driven representations can overfit to the training data used for feature learning and thus may not generalize well to mismatched acoustic conditions.

In the work presented here, we simplify our previous approach [3] (i.e. time-domain feature learning) by operating in the frequency domain. That is, we include a Fourier transform layer in the network and let the network learn the filter-banks in the frequency domain. Frequency-domain feature learning has been previously used in [4] and [5], however, we propose a new normalization layer which helps with stabilization and better convergence of the filters. Additionally, we employ a different weight constraint approach which further improves the results. We use the proposed frequency-domain layer in the state-of-the-art LF-MMI setup and show significant word error rate improvements on various well-known large vocabulary databases. Finally, based on the learned filters in our frequency-domain layer, we propose an analytic set of filters which enable faster training of the acoustic model while delivering the same results as the proposed setup.

Time-domain feature learning is explained in Section 2. In Section 3, our proposed frequency-domain approach, as well as previous works on frequency-domain feature learning, is described. The experiments and results are presented in Section 4, and some conclusions appear in Section 5.

2. Time domain feature learning

Most of the data-driven feature learning approaches in recent years have attempted to do feature learning directly from the time-domain waveform. [6] trained a DNN acoustic model on waveforms and showed that auditory-like filters can be learned using fully connected deep neural networks. Other works usually use time convolution layers, which share weights across time shifts [7, 8, 9].

The first layer in a time-domain feature learning setup is usually a time-convolution layer, which is like a finite impulse-response filter-bank followed by a nonlinearity. This layer is expected to approximate the standard filter-banks, which is often implemented as filters followed by rectification and averaging over a small window. The output of this layer can be referred to as time-frequency representation. Next, the rectification or absolute function is applied to the output of the convolution filters and the log compression is used on the absolute value of the filter outputs to reduce the feature dynamic range. To the best of our knowledge, most of the reported results show performance degradation when using time-domain feature learning and [9] and [3] are the works where raw waveform setup slightly outperforms the conventional features. [3] proposed a new nonlinearity to aggregate filter outputs leading to results competitive with the state of the art baseline systems.

3. Frequency domain feature learning

3.1. Previous works

In contrast to time-domain feature learning where the inputs to the CNN and filter bank layers are raw speech samples, in the frequency-domain feature learning the samples are passed through a Fourier-transform layer first [5, 4, 10].

In this study, we adopt a similar frequency-domain approach but with a few major differences. Specifically, we use an extra normalization block and we constrain the weights in the filter-bank layer to a short-range. The details of our setup will be explained in the following subsections.

3.2. Proposed feature extraction block

The overall process of feature learning in our setup is shown in Figure 1. The input features of the neural network are non-overlapping 10ms segments of the raw waveform signal. Each segment is represented by a vector of amplitude values (e.g. for 8kHz speech, the features will be 80-dimensional). Unlike acoustic modeling from time-domain [3], there is no need for input normalization in the frequency-domain setup. As shown in Figure 1, the input features are first passed through a pre-processing layer which performs pre-emphasis and DC-removal. Then they go through the Fourier transform layer which is implemented using sine/cosine transforms. L2-normalization is also applied on the output of Fourier transform. The next step is the normalization block which is explained in Section 3.3. After normalization, there is the main filter-bank layer. Implementation-wise, the filter-bank layer is an NxM weight matrix (i.e. a linear transform), where each row represents an M-point filter. The weights in this matrix are constrained according to Equation 1 which is applied after updating the parameters of the filter bank for each mini-batch during training.

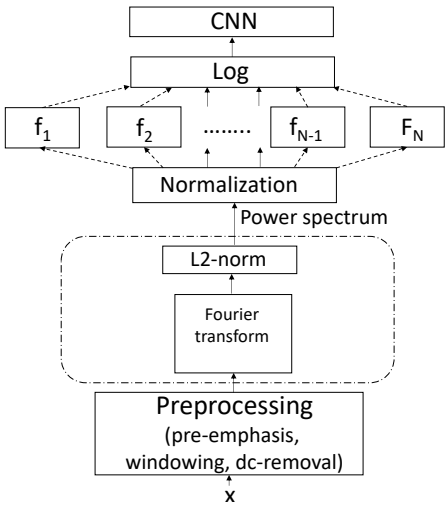

Figure 1: *Frequency-domain feature extraction setup*

$$\mathcal{W}_{ij}^{'} = \max(\alpha_1, \min(\mathcal{W}_{ij}, \alpha_2)) \quad \alpha_1 < \alpha_2 \qquad (1)$$

We tried different values for α_1 and α_2 and found out that 0 and 1 give the best results. Table 1 compares the different constraints we tried. We also compared this method with the proposed method in [5], where the parameters are constrained to be positive by using exponentiation as $\exp(W_{ij})$ but found that our approach was more effective.

The filter-bank layer is followed by log compression which is a common practice in DNN acoustic modeling, where the log compression helps to reduce the dynamic range of the filter-banks. We investigated two common log methods: (1) clipped log (i.e. $\log(\max(\delta, x))$) and (2) stabilized log (i.e $\log(x + \delta)$) and found out that clipped log was more effective which is what we use in this setup. Finally, the log-filter-bank features are passed to a CNN layer. We use a 2-dim convolution layer with 32 filters with size $3X3$, with time stride 2 instead of pooling with factor 2 in this setup.

Table 1: *Effect of different filter-bank constraint methods*

Method		WER
Proposed weight constraint (α_1, α_2)	$(-\infty, \infty)$	15.9
	$(-\infty, 1)$	16.0
	$(0, \infty)$	14.5
	$(0, 1)$	**14.3**
exponential weights[5]		15.3

3.3. Normalization block

As suggested in [5], applying normalization before filter learning is beneficial. Distribution of the inputs can change during training and the first layer of the network is more sensitive to these changes which can slow down training or make it unstable. Therefore, we normalize the input power spectrum which helps to stabilize training and to better train narrow-band *filter-bank*s. As shown in Figure 1, the inputs to the filter learning stage are normalized. This is shown in more details in Figure 2. Specifically, we first transform the power spectrum features to log-space, where batch normalization is applied, normalizing the features over a mini-batch. We use batch normalization proposed in [11] which allows to use much larger learning rates. After batch normalization, the outputs are normalized globally using mean and variance parameters, that are jointly learned with other parameters during training. Finally, the parameters are transformed back into normal space using the exponential function.

We examine the effect of each component in the normalization block in Table 2. It can be seen that doing the normalization in log-space is crucial. Besides, batch-normalization has a significant effect on the final word error rate too.

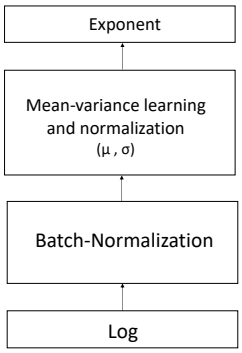

Figure 2: *Normalization block*

Table 2: *Effect of different components in normalization block*

log-domain	batch-norm	global norm	WER
✓	✓	✓	14.3
✓	✓	✗	14.6
✗	✓	✓	17.2
✓	✗	✓	15.0

3.4. Analytic filter approximation

In this section, we propose a new set of analytic filters for narrow-band data. The new filters are approximated using the filters learned in the filter-bank layer on the $8kHz$ Switchboard, which is trained (separately) using 40, 100 and 200 filters.

The filter shapes learned in the filter-bank layer are close to cosine type filters, therefore we use the cosine function for estimating the analytic filters. The formula used for filter estimation is shown in Equation 2, where each filter is specified using a center frequency f_c and a bandwidth w. As can be seen, the filters estimated using this formula have the same energy.

$$\begin{cases} \frac{\pi}{2w}\cos(\frac{\pi(x-f_c)}{w}) & f_c - \frac{w}{2} \leq x \leq f_c + \frac{w}{2} \\ 0 & else \end{cases} \quad (2)$$

The center frequencies f_c are estimated using a 4^{th} order polynomial which is in turn approximated using least-square error minimization on the center frequencies for the 40, 100 and 200 learned filters. The approximated polynomial is shown in Equation 3. Nyquist and f are in Hz and M is the number of filters.

$$f_c(i) = a_1 f^4 + a_2 f^3 + a_3 f^2 + a_4 f + a_5 \quad (3)$$
$$f = \frac{i \times Nyquist}{M}$$
$$(a_1, a_2, a_3, a_4) = (1.6e^{-11}, -7.4e^{-8}, 2.2e^{-4}, 0.23, 0)$$

To measure the bandwidths for the learned filters, we considered two approaches: noise-equivalent bandwidth estimation, in which the bandwidth for filter $\mathbf{u}$ is computed as $\sum_i u_i^2/(\max_j u_j)^2\delta_f$, where $\delta_f = \frac{Nyquist}{N}$ and N is the number of FFT bins; and percentile-based bandwidth estimation, where the bandwidth is the difference in frequency between the 25% and 75% percentiles of the mass of the distribution for filter $\mathbf{u}$. It can be shown mathematically that the proposed filters have a bandwidth of w according to the noise-equivalent formula. We estimate the bandwidths for the analytic filters as a piece-wise linear function of the center frequencies. This piece-wise linear function is in turn approximated using the bandwidths of the learned filters (on the 8kHz Switchboard). The plot of the filter bandwidth versus center frequency for the learned and approximated filters are shown in Figure 3. As can be seen, the filters learned in the filter-bank layer have higher bandwidth (and thus larger overlap) compared to the Mel filters. Also, the optimal filter bandwidth seems to stay constant as the number of filters is increased, which is not how triangular Mel filter-banks are usually set up. The bandwidth of the Mel filters are set by aligning the endpoint of the triangle and it is not determined using proper optimization. The proposed approximated filters are wider and overlap with more neighboring filters.

4. Results

In this section, we compare our proposed frequency-domain setup with the time-domain setup proposed in [3] trained on the $300hrs$ Switchboard task. We evaluate on the full Hub5 '00 set (also called "eval2000"). [1] We also compare with two conventional baselines: MFCC and log-Mel filter-bank features. The MFCC baseline system uses spliced 40-dim MFCC feature vectors followed by an LDA layer. Note that the results for 40-dim and 80-dim MFCC features were the same (not shown). Mel features are generated by passing the power spectrum through a set of Mel-filters and log-Mel filter-bank features are generated by applying a log compression on the Mel features. The log-Mel features – as well as all other feature learning layers we are comparing here – are followed by a CNN layer. The rest of the network structure is the same in all experiments (i.e. after the LDA or the CNN layer). Specifically, we use blocks of TDNN layers [13] followed by batch-normalization [11] and rectified linear units.

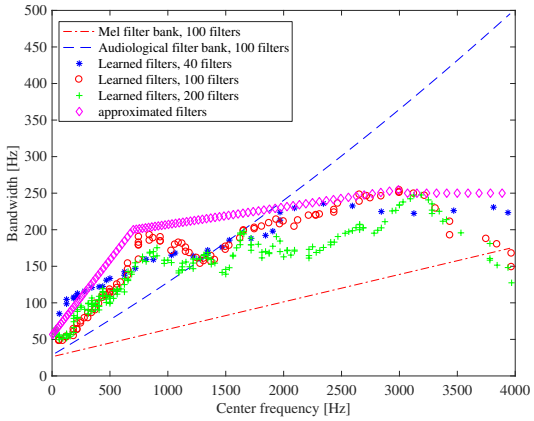

Figure 3: *Filter bandwidth vs. center frequency for different filter-banks.*

The results are shown in Table 3. The time-domain feature extraction setup used in the 3^{rd} row of this table is similar to [3]. We also show the results of training separate filters on real and imaginary parts of the Fourier transform as done in the Complex Linear Projection (CLP) method proposed in [10]. Specifically, we train two separate filter-banks $\mathcal{W}_\mathcal{R}$ and $\mathcal{W}_\mathcal{I}$ on the real and imaginary parts of Fourier transform of the signal and the real and complex parts of the output are computed as $\mathcal{W}_\mathcal{R}\mathcal{X}_\mathcal{R} - \mathcal{W}_\mathcal{I}\mathcal{X}_\mathcal{I}$ and $\mathcal{W}_\mathcal{R}\mathcal{X}_\mathcal{I} + \mathcal{W}_\mathcal{I}\mathcal{X}_\mathcal{R}$ and $L2$-norm followed by log nonlinearity is used to compute the log-filter-bank features. We can see that our proposed frequency-domain setup outperforms other frequency-domain and time-domain setups and conventional methods. We used 40, 100, and 200 filters in the filter-bank layer in our setup and all cases led to the same result shown in the table (i.e. 14.3).

4.1. Filter analysis

Figure 4 shows the filter-bank weights learned for the proposed frequency-domain setup with and without normalization. It can be seen that normalization helps in learning less noisy filters.

The filters learned in the filer-bank layer are usually interpreted as band-pass impulse response. One of the main issues in time-domain filter learning is that the filters are not usually narrow-banded and regularization is necessary. We use L_1 regularization on the Fourier transform of the filters learned in the time-domain setup which is helpful in learning narrow-band

[1] We perform all the experiments using the Kaldi speech recognition toolkit[12].

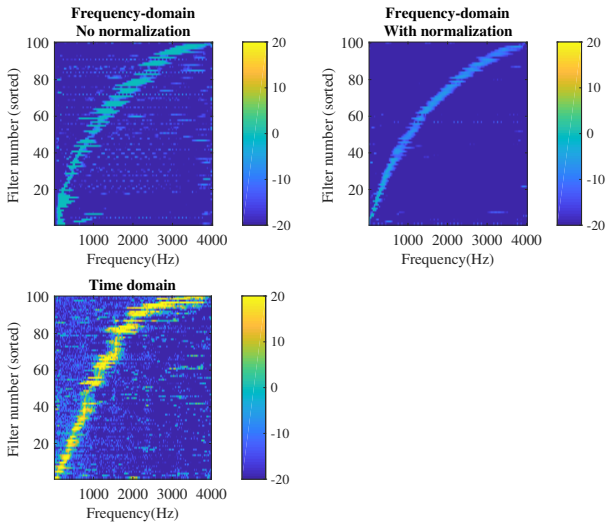

Figure 4: *Magnitude response of learned filter ordered in center frequency.*

filters. As can be seen in Figure 4, this issue is alleviated in frequency-domain filter learning and the filter banks learned in this domain are narrow-banded and none of the filters shows multiple pass-bands. We apply L_2-regularization on filter bank weights in the frequency-domain and CLP setups.

Table 3: *Frequency-domain vs. Time-domain*

Method	WER	
	eval2000	rt03
40-dim MFCC	14.9	17.8
log-Mel fbank*	15.1	18.5
Time-domain setup [3]*	14.4	17.4
Time-domain setup [9]*	15.2	18.2
Proposed Frequency-domain setup*	**14.3**	**17.0**
CLP*	14.9	17.6

* CNN layer added after log filter-banks.

4.2. Analytic filters

To evaluate the proposed analytic filters (in Section 3.4), we set the filters in the filter-bank layer (in the DNN) using the proposed analytic set of filters and train the DNN while the filters are fixed (we still use the normalization block). The results are presented in Table 4. We can see the proposed analytic filters have outperformed the proposed frequency-domain filters based on which they are approximated. This might be because they are fixed during the training.

Table 4: *Frequency-domain setup vs. proposed analytic filters*

Method	WER	
	eval2000	rt03
40-dim MFCC	14.9	17.8
Proposed Frequency-domain setup	14.3	17.0
Proposed analytic filters	**14.2**	**16.8**

4.3. Performance on various LVCSR tasks

Finally we evaluate our proposed frequency-domain setup on various databases, namely Tedlium [14], and AMI IHM and SDM [15], Wall Street Journal [16] and Librispeech [17]. The results are shown in Table 5. The amount of training data for filter learning varies from 80-1000 hours across these tasks. The baselines are the state of the art TDNN models trained on conventional 40-dim MFCC features. We use 100 filters in in 8kHz tasks and 200 filters for the 16kHz tasks. The results on Librispeech are obtained by rescoring with $4-$gram language model. We use the same CNN layer as described in Section 3.2 in all the experiments. An average relative improvement of 1% to 7% is observed over the conventional state-of-the-art MFCC based models.

Table 5: *Performance of the proposed frequency-domain setup on various databases.*

Database	Test set	Baseline	Proposed setup
Switchboard	eval2000	14.9	14.3
	rt03	17.8	17.0
Wall Street Journal	eval92	2.6	2.4
	dev93	4.7	4.6
TED-LIUM	test	8.8	8.5
	dev	8.3	8.0
AMI-IHM	eval	20	19.9
	dev	20	19.5
AMI-SDM	eval	39.6	38.9
	eval	35.8	34.9
Librispeech	dev-other	10.6	9.7
	test-other	10.9	10.2

5. Conclusions and future work

In this study, we presented our work on joint feature learning and acoustic modeling in the state-of-the-art lattice-free MMI framework. Specifically, we introduced a new frequency-domain feature learning layer which improves the WER for the baseline MFCC setup from 14.9% to 14.3% on the 300hrs Switchboard task by employing a new normalization block and a short- range weight constraint. Furthermore, we did comparison among different well-known data driven feature learning approaches. We also evaluated our proposed frequency-domain setup on various narrow-band and wide-band LVCSR databases and achieved consistent improvements ranging from 1% to 7% relative reduction in WER.

Inspired by the learned features, we proposed a new set of analytic filters for narrow-band data. We used a 4^{th} order polynomial to approximate the center frequencies based on the learned filters in the proposed frequency-domain setup. We also estimated the bandwidths for the analytic filters using a piecewise linear function of the center frequencies. The important observation is that the optimal filter bandwidth stays constant as the number of filters is increased; this is not how triangular Mel filter-banks are set up. Using the proposed analytic filters led to a WER of 14.2 on the 300hrs Switchboard task which is an improvement over the proposed setup itself. As an added benefit, these analytic filters are considerably faster at runtime, as they are pre-computed and fixed.

6. References

[1] P. Mermelstein, "Distance measures for speech recognition, psychological and instrumental," *Pattern recognition and artificial intelligence*, vol. 116, pp. 374–388, 1976.

[2] H. Hermansky, "Perceptual linear predictive (plp) analysis of speech," *the Journal of the Acoustical Society of America*, vol. 87, no. 4, pp. 1738–1752, 1990.

[3] P. Ghahremani, V. Manohar, D. Povey, and S. Khudanpur, "Acoustic modelling from the signal domain using cnns." in *INTERSPEECH*, 2016, pp. 3434–3438.

[4] T. N. Sainath, B. Kingsbury, A.-r. Mohamed, and B. Ramabhadran, "Learning filter banks within a deep neural network framework," in *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*. IEEE, 2013, pp. 297–302.

[5] T. N. Sainath, B. Kingsbury, A.-r. Mohamed, G. Saon, and B. Ramabhadran, "Improvements to filterbank and delta learning within a deep neural network framework," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 6839–6843.

[6] Z. Tüske, P. Golik, R. Schlüter, and H. Ney, "Acoustic modeling with deep neural networks using raw time signal for lvcsr," in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[7] Y. Hoshen, R. J. Weiss, and K. W. Wilson, "Speech acoustic modeling from raw multichannel waveforms," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 4624–4628.

[8] D. Palaz, M. Magimai.-Doss, and R. Collobert, "Analysis of cnn-based speech recognition system using raw speech as input," Idiap, Tech. Rep., 2015.

[9] T. N. Sainath, R. J. Weiss, A. Senior, K. W. Wilson, and O. Vinyals, "Learning the speech front-end with raw waveform cldnns," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[10] E. Variani, T. N. Sainath, I. Shafran, and M. Bacchiani, "Complex linear projection (clp): A discriminative approach to joint feature extraction and acoustic modeling." in *INTERSPEECH*, 2016, pp. 808–812.

[11] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*, 2015, pp. 448–456.

[12] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*, no. EPFL-CONF-192584. IEEE Signal Processing Society, 2011.

[13] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. J. Lang, "Phoneme recognition using time-delay neural networks," in *Readings in Speech Recognition*. Elsevier, 1990, pp. 393–404.

[14] A. Rousseau, P. Deléglise, and Y. Esteve, "Ted-lium: an automatic speech recognition dedicated corpus." in *LREC*, 2012, pp. 125–129.

[15] I. McCowan, J. Carletta, W. Kraaij, S. Ashby, S. Bourban, M. Flynn, M. Guillemot, T. Hain, J. Kadlec, V. Karaiskos *et al.*, "The ami meeting corpus," in *Proceedings of the 5th International Conference on Methods and Techniques in Behavioral Research*, vol. 88, 2005, p. 100.

[16] D. B. Paul and J. M. Baker, "The design for the wall street journal-based csr corpus," in *Proceedings of the workshop on Speech and Natural Language*. Association for Computational Linguistics, 1992, pp. 357–362.

[17] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an asr corpus based on public domain audio books," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 5206–5210.