Robust CNN-based Speech Recognition With Gabor Filter Kernels

Shuo-Yiin Chang [1,2], *Nelson Morgan*[1,2]

[1] EECS Department, University of California-Berkeley, Berkeley, CA, USA
[2] International Computer Science Institute, Berkeley, CA, USA
shuoyiin@icsi.berkeley.edu, morgan@icsi.berkeley.edu

Abstract

As has been extensively shown, acoustic features for speech recognition can be learned from neural networks with multiple hidden layers. However, the learned transformations may not sufficiently generalize to test sets that have a significant mismatch to the training data. Gabor features, on the other hand, are generated from spectro-temporal filters designed to model human auditory processing. In previous work, these features are used as inputs to neural networks, which improved word accuracy for speech recognition in the presence of noise. Here we propose a neural network architecture called a Gabor Convolutional Neural Network (GCNN) that incorporates Gabor functions into convolutional filter kernels. In this architecture, a variety of Gabor features served as the multiple feature maps of the convolutional layer. The filter coefficients are further tuned by back-propagation training. Experiments used two noisy versions of the WSJ corpus: Aurora 4, and RATS re-noised WSJ. In both cases, the proposed architecture performs better than other noise-robust features that we have tried, namely, ETSI-AFE, PNCC, Gabor features without the CNN-based approach, and our best neural network features that don't incorporate Gabor functions.

Index Terms: speech recognition, convolutional neural network, Aurora 4, Gabor filter

1. Introduction

Neural networks have been used successfully for HMM-based speech recognition for more than two decades [1]. In that approach, network outputs were used as posteriors to derive emission probabilities for hidden Markov models (HMMs). Later, a number of researchers made use of network outputs as features for HMM observations (tandem) [2][3]. Both of these approaches have been used in more recent "deep" learning methods that have been designed to effectively incorporate more layers, and in particular have been successfully applied to automatic speech recognition (ASR) [4][5][6].

In a typical system, cepstral coefficients or short-term spectra are generated as input to a (deep) neural network or convolutional neural network (CNN) [7][8][9]. While trained features could effectively reduce WER on matched testing set, they might be specialized. Unlike cepstral coefficients or spectra, Gabor processing generates a large number of features based on prior knowledge of auditory models before neural network training. Previously, Gabor features have demonstrated substantial noise robustness [10][11], providing a different solution to most existing robustness methods focusing on compensating the difference of clean and noisy speech in the model [12][13] or the features [14][15][16]. In this paper, we propose a neural network architecture, referred as the *Gabor convolutional neural network*, incorporating the process of 2D Gabor filtering into the filter kernel of a typical convolutional neural network.

Gabor functions have been successfully incorporated as an approximation for spectro-temporal patterns referred to as spectro-temporal receptive fields, STRFs [17]. These approaches define a series of spectral, temporal, and spectro-temporal modulation filters that can be seen as roughly modeling neural firing patterns for particular spectro-temporal signal components. Aside from any biological interpretation, they provide a wide range of transformations of the time-frequency plane. In particular, purely temporal features such as TRAPS [18] and HATS [19] can be regarded as special cases of spectro-temporal features. More recently, Gabor filters have been used as input for deep neural networks (DNN) to generate Gabor-DNN features for improved speech recognition [20][21].

Here, we report integrating pre-defined Gabor filters and trained convolutional neural networks to generate a more robust feature called GCNN. Typical CNN architectures use shared weights to filter a receptive field, modeling the local characteristics of a spectrum. This filtering process permits us to integrate 2D Gabor filters into the CNN topology. We modified the receptive fields of the CNN, with several time and frequency supports conforming to Gabor filter characteristics. The modified CNN includes Gabor filter coefficients as the initial filters at the lowest layer, and performs fine-tuning to optimize the coefficients by back propagation training. In the experiments, the proposed GCNN features perform better than both Gabor-DNN and CNN features, where the former kept Gabor coefficients untrained while the latter used trained filters without Gabor modeling. Also, pooling reduced word error rate effectively for recognition of noisy speech.

2. Proposed Method

2.1. Power-Normalized Spectrum

Either Gabor filters or CNNs implement filters of local frequency characteristics, so some function of the short-term power spectrum was employed as input. Although the Mel-spectrum has been successfully used in many experiments, it is easily corrupted in the presence of noise. In this paper, the features are generated from a more robust spectro-temporal representation called *power-normalized spectrum*, (PNS) which uses the feature generation algorithm based on PNCC [16]. Unlike mel spectrum, the short-term spectrum is integrated using gammatone auditory filters equally spaced on the equivalent rectangular bandwidth (ERB) scale. Second, medium-duration power bias is subtracted, where the bias level calculation was based on the ratio of arithmetic mean and geometric mean (AM/GM ratio) of the medium duration power, which is motivated by a decrease of the noise power for a decreasing AM/GM ratio [16]. Finally, a power nonlinearity with an exponent of 0.1 replaces the logarithm nonlinearity used for compression in mel cepstra.

2.2. Gabor features

Here we describe our Gabor filter design and the corresponding Gabor features. Gabor filters capture localized regions of spectrum and temporal information over a broader time interval. We implement Gabor filters as the product of a complex sinusoid and a Hanning envelope. The complex sinusoid (with time modulation frequency ω_n and spectral modulation frequency ω_k) is represented as:

$$s(n,k) = \exp[-j(\omega_n n + \omega_k k)] \quad (1)$$

while the Hanning envelope is given as Eq. 2 (where W_n and W_k denote window length)

$$h(n,k) = [\frac{1}{2}(1-\cos(\frac{2\pi n}{W_n+1}))][\frac{1}{2}(1-\cos(\frac{2\pi k}{W_k+1}))] \quad (2)$$

The temporal window W_n is inversely proportion to the temporal modulation frequency ω_n; similarly, the spectral window W_k is inversely proportional to the spectral modulation frequency ω_k. The periodicity of the carrier function is defined by the radian frequencies ω_n and ω_k, which permits the Gabor function to be tuned to a particular direction of spectro-temporal modulation. The Gabor filter bank used here has been adapted from [20] and consists of 59 different characteristics. A subset of the possible combinations are used to avoid high correlations of feature components, resulting in an 814-dimensional feature. The narrow filters capture the rapidly time-varying part of the spectrum, while wide filters capture the coarse representation of the speech dynamic, as shown in Fig. 1. Similarly, a variety of "tall" and "short" filters corresponding to different spectral modulation frequencies generate features capturing different spectral dynamics.

In the experiments reported here, Gabor features were obtained by convolving power normalized spectrum and Gabor filters. Previously, we exploited Gabor features as input to a DNN. In that scheme, the Gabor filter coefficients were fixed while the neural network parameters were trained.

2.3. Convolutional Neural Network

Unlike Gabor filters, localized filters in convolutional neural networks are typically designed using supervised training. A common CNN topology was proposed by LeCun et.[22] and was widely used in computer vision. Recently, researchers have applied CNNs to speech recognition. The typical convolutional neural network consists of a convolutional layer, a subsampling layer, and fully connected layers as shown

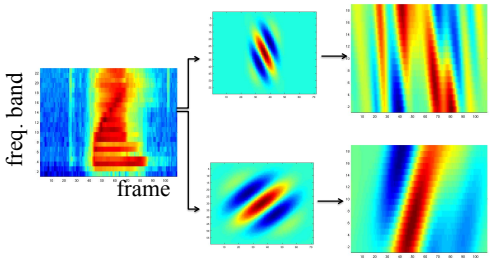

Figure 1: Above: a narrow filter and corresponding filter output; below: a wide filter and filter output.

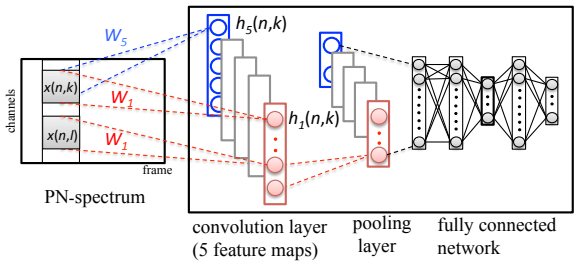

Figure 2: Basic convolutional neural network topology

in Fig. 2. Each CNN training case consists of entire frequency bands of successive frames. In the convolutional layer, the receptive field of each neuron is connected to a local subset of frequency bands. A set of neurons with receptive fields shifted in frequency share the same kernel (weights). Stacks of these neurons that cover features of the kernel along entire frequency bands constitute a feature map. Typically, a convolutional layer is composed of multiple feature maps determined by different kernels, as depicted in Fig. 2. Activation of each neuron is computed by multiplication of a local receptive field with the weights, adding a bias and applying a nonlinear function:

$$h_m(n,k) = \theta(\sum_{i=-N}^{N}\sum_{j=-K}^{K} W_m(i,j)\cdot x(n+i,k+j) + b_m) \quad (3)$$
$$= \theta(W_m(-n,-k) * x(n,k) + b_m)$$

where $h_m(n,k)$ represents the neuron of m^{th} feature map, whose receptive field is $2K+1$ (bands) by $2N+1$ (frames) matrix centered at current band of the frame, $x(n,k)$. The connection weights W_m perform filtering on the receptive field where the indices of the filter coefficients are flipped from the weight indices, both vertically and horizontally as shown in Eq. 3. b_m and $\theta(.)$ are the bias term and sigmoid function respectively.

In common CNN topology, there is a pooling layer performing down-sampling after the convolutional layer. Typically, pooling takes the maximum value from a window of activations in a convolutional layer, where the window size is called pooling size. For example, the pooling size in Fig. 2 is 2. The max-pooling layer can effectively reduce the size of the convolutional layer, and remove variance along convolutional bands. A fully connected network is then added after the pooling layer to integrate the pooled features.

2.4. Convolutional Neural Network using Gabor filter

To model Gabor filtering in CNNs, we made two modifications to Eq. 3. First of all, we used linear instead of sigmoid activations. The bias term was enforced to be zero. In this case, the neurons of convolutional layer are just the filter outputs of the receptive field. Second, Gabor features consist of filters with several different time and frequency band supports. To meet the filter design, we modified the receptive field size to give the same supports as the Gabor filters, instead of using a fixed receptive field size.

The Gabor filter coefficients are then incorporated into the initial weights W_m in Eq. (1). The initial feature maps were Gabor features. The modified topology is depicted in Fig. 3. Unlike Gabor-DNN features, filter coefficients were no longer untrained. Compared to a typical CNN, we used Gabor filter

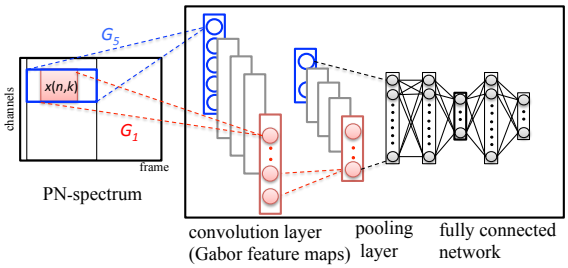

Figure 3:Gabor convolutional neural network topology

characteristics to define linear activated feature maps with several time and frequency resolutions to initialize back propagation training, and potentially avoid over fitting to training data. A max-pooling layer follows the convolutional layer to down-sample and smooth the Gabor features.

3. Experimental Setup

The proposed approach was evaluated using two noisy versions of WSJ: (1) Aurora 4 [23] and (2) RATS "re-noised" Wall Street Journal (WSJ) speech. The Aurora 4 dataset provides both a clean training set and a multi-condition training set. The clean training set is taken from 7138 utterances of WSJ0 SI-84 dataset (83 speakers) where the data was recorded using a Sennheiser microphone. The multi-condition training set contains the same number of utterances as the clean training set, while half of the utterances were recorded by a secondary microphone. Six noise types (car, babble, restaurant, street, airport and train) at SNRs between 10dB and 20 dB were randomly added to three-fourths of utterances from both microphone types. The evaluation set is based on 166 utterances of Nov'92 5k evaluation set (8 speakers), and is composed of 14 subsets: clean and 6 noise corrupted sets for data recorded by both microphone types. The noise types are the same as those used for the multi-condition training set, but were chosen with an SNR between 5 and 15 dB. The 14 subsets are grouped into 4 sets: clean, noisy, clean with microphone distortion and noisy with microphone distortion, which are referred as A, B, C and D respectively.

For "RATS re-noised WSJ," we started out with data taken from WSJ1 dataset (284 speakers) for training and the WSJ-eval94 dataset (20 speakers) for testing. Estimated additive and channel noise from degraded recordings was applied to both training and testing dataset using the "renoiser" tool [24]. Designed for use in the DARPA RATS project, the system analyzes data from RATS rebroadcast example signals (in this case, LDC2011E20) to estimate the noise characteristic including SNRs and frequency-shifts; the original data is described in [25] and consists of a variety of continuous speech sources that have been transmitted and received over 8 different radio channels, resulting in significant signal degradations. The 8 radio channel characteristics are specified in Table 1. We applied the same noise characteristics to WSJ data to generate the "RATS re-noised WSJ". In this case, the training data was obtained from 51.2 hours of WSJ1 dataset with clean channel and channel G (the channel with highest SNR). Testing data was 0.8 hours of WSJ-eval94 for each channel. The results reported here are WERs averaging over clean and 8 noisy channels.

For both Aurora 4 and RATS re-noised WSJ, the acoustic models used cross-word triphones estimated with maximum likelihood. The resulting triphone states were clustered to 2500 tied states, each of which was modeled by 16 components of a Gaussian mixture model. We used version 0.6 of the CMU pronunciation dictionary and the standard 5k bigram language model created at Lincoln Labs for the 1992 evaluation. Unless otherwise specified, mean normalization was performed for the features, while vocal tract length normalization (VTLN) and adaption techniques such as maximum likelihood linear regression (MLLR) were not employed for these tests.

The fully connected deep neural networks were trained with a 6-layer bottleneck structure with a bottleneck (25 units) in the fifth hidden layer. The output layer consisted of 41 context-independent phonetic targets. 39-d cepstral coefficients or 814-d Gabor features with 9 successive frames were used as input for a fully connected deep neural network. Restricted Boltzman machine (RBM) pre-training [26] was employed to initialize the parameters of the neural network. For back propagation following the pre-training, we began with a learning rate of .008 and reduced the learning rate by factors of two once cross-validation indicated limited progresses with each learning rate, and continued until cross-validation showed essentially no further progress.

For the CNN topology, we used 120 filters for the convolutional layer. The filter size was 9 frequency bands with 15 successive frames. We used a pooling size of 6 convolutional bands with stride 2 (overlap by 4), which reduced dimensionality by a factor of 2. This layer was fed to a 5-layer fully connected bottleneck structure. In the GCNN architecture, the time support for each filter kernel ranges from 7 to 99 frames, and frequency support ranges from 7 to 40 bands. 59 of the filters were initialized as Gabor filter coefficients, and the other 61 filters were randomly initialized. The rest of network set up is the same as for the CNN. For both CNN and GCNN architecture, 40-d power normalized spectrum was used as input. We didn't use delta and acceleration coefficients to be consistent with Gabor filter input. Back propagation strategy is the same as used for the DNN, while no pre-training was performed. For fair comparison, the number of free parameters of the neural networks were constrained to roughly 3.5M by controlling the hidden layer size.

MFCCs were concatenated with the fully connected deep neural network or convolutional neural network trained features, resulting in a 64-dimensional feature vector. Also, means and variances were normalized per utterance before HMM training and testing for all the features described here.

	Microphone	SNR	Frequency shift
Channel A	Motorola HT1250	15.6	0
Channel B	Midland GXT1050	6.2	0
Channel C	Midland GXT1050	6.0	0
Channel D	Galaxy DX2547	3.5	180.9 Hz
Channel E	Icom IC-F70D	0.9	0
Channel F	Trisquare TSX300	3.0	0
Channel G	Vostek LX-3000	18.7	0
Channel H	Magnum 1012 HT	3.0	120.7 Hz

Table 1: channel characteristic in RATS WSJ

4. Results and Discussion

We first present a series of baseline results for RATS re-noised WSJ and Aurora 4 results using the clean training set.

In Table 2, PNCC was better than other feature baselines for both cases (on average). As a result we used PNCC or PN spectrum based features for the experiments that followed.

Feature	RATS WSJ	Aurora 4				
		A	B	C	D	Avg
MFCC	62.4	8.2	36.7	25.5	52.1	40.5
MFCC(CMVN)	61.8	7.9	30.3	22.8	48.2	35.8
ETSI-AFE	59.4	8.8	24.5	24.7	38.9	29.5
PNCC	58.8	9.3	22.1	23.3	37.5	27.9

Table 2: MFCC, AFE and PNCC baseline

In Table 3, we compare a series of trained features using fully connected neural network or convolutional neural network. First of all, the trained features of Table 3 are better than untrained features of Table 2. In Table 3, Gabor-DNN was better than PNCC-DNN except for the clean set (A). Next, we compare Gabor-DNN with PNS-CNN and PNS-GCNN without pooling layer. Without pooling, the inputs of fully connected network are feature maps of convolutional layer. Therefore, these rows function as a comparison between different sets of spectro-temporal filters. Gabor-DNN used filters with variable size, but totally handcrafted. PNS-CNN, on the other hand, learned filter with fixed size and trained on limited data. PNS-GCNN has filters with variable size. Also, the trained filters were initialized with handcrafted filters. In Table 3, PNS-GCNN was better than the other two features, although the differences are small. The larger effects visible in the table show the effects of pooling, and the cumulative effects of pooling and using GCNN instead of CNN. In particular, max pooling provides a significant improvement for both RATS WSJ and Aurora 4 (especially for noisy set (B) and noisy set with channel distortion (D)), and particularly with pooling, using Gabor filters to help design the CNN has a good effect.

The trained CNN filters are composed of several vertical (spectral) and horizontal (temporal) filters, as the examples in Fig. 4 show. However, the filters have very low correlation to the diagonal Gabor filter, such as the diagonal filter in Fig. 5(left) while the diagonal filter would be kept and tuned in GCNN topology and the final filter is shown in Fig. 5(right). Thus, diagonal filtering is another factor distinguishing trained filters with Gabor initialization and those with random initialization.

Feature	RATS WSJ	Aurora 4				
		A	B	C	D	Avg
PNCC-DNN	54.8	6.6	20.4	18.7	36.6	26.2
Gabor-DNN	53.9	6.9	18.9	17.8	35.4	25
PNS-CNN no pooling	54.1	6.3	19.8	17.9	35.8	25.5
PNS-GCNN no pooling	51.6	6.5	18.3	18.5	34.6	24.4
PNS-CNN with pooling	51.5	6.2	18.8	15.6	33.8	24.1
PNS-GCNN with pooling	49.8	6.3	17.8	15.7	32.1	22.9

Table 3: WER for neural network features, clean training, noisy test.

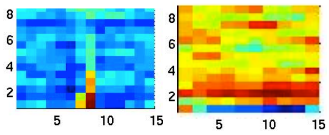

Figure 4: a vertical and a horizontal filter example

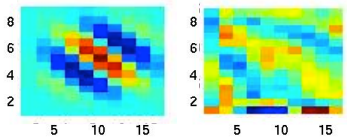

Figure 5: (left): diagonal Gabor filter, (right): GCNN tuned diagonal Gabor filter

In addition to the experiments with mismatched training and testing, we also used the multi-condition training set for Aurora 4. We choose the distinguished features, Gabor-DNN, PNS-CNN and PNS-GCNN comparing with two baselines ETSI-AFE and PNCC. The results are shown in Table 4, where the proposed PNS-GCNN could achieve 16.6% WER. This was achieved without VTLN, MLLR, or other modeling enhancements.

Feature	Aurora 4				
	A	B	C	D	Avg
ETSI-AFE	10.6	18.6	19.7	30.9	23.4
PNCC	10.5	17.4	19.1	30	22.5
Gabor-DNN	8.4	14.2	14.3	25.8	18.8
PNS-CNN with pooling	7.4	13.4	12.8	24.7	17.8
PNS-GCNN with pooling	7.3	12.8	12.1	22.7	16.6

Table 4: WER for multi-condition training set.

5. Conclusion

In this paper, we proposed a robust CNN architecture integrating Gabor filter design. The proposed GCNN architecture learned local features with multiple temporal and spectral resolutions with Gabor filter initialization, both for structure and initial weights. The filter coefficients are further optimized by back propagation training. A maximum pooling layer also gave significant improvement in our experiments Our results indicated the proposed GCNN feature achieve the best results among other noise robust feature and neural network feature for the two noisy WSJ corpus.

6. Acknowledgements

We thank Adam Janin and Dan Ellis for the helpful discussion. We thank Hans-Günter Hirsch for help in setting up the Aurora 4. We also thank Chanwoo Kim and Richard Stern for use of PNCC. This material is based on work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. D10PC20024. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the view of the DARPA or its Contracting Agent, the U.S. Department of the Interior, National Business Center, Acquisition & Property Management Division, Southwest Branch.

7. References

[1] H. Bourlard and N. Morgan, “Connectionist Speech Recognition: A Hybrid Approach”, Kluwer Press, 1993

[2] H. Hermansky, D. P. W. Ellis, and S. Sharma, “Tandem connectionist feature extraction for conventional HMM systems,” in Proc. IEEE Int. Conf. Acoust., Speech, Signal Process., 2000, vol. 3, pp. 1635–1638.

[3] Q. Zhu, B. Chen, N. Morgan, and A. Stolcke, “On using MLP features in LVCSR,” in Proc. Interspeech, 2004, pp. 921–924

[4] A. Mohamed, G. Dahl, and G. Hinton, “Acoustic modeling using deep belief networks,” Audio, Speech, and Language Processing, IEEE Transactions on, vol. 20, no. 1, pp. 14 –22, Jan. 2012.

[5] G. Hinton, L. Deng, D. Yu, G. E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, and T. N. Sainath, “Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups,” Signal Processing Magazine, IEEE, vol. 29, no. 6, p. 8297, 2012.

[6] M. Seltzer, D. Yu and Y. Wang, “An investigation of deep neural networks for noise robust speech recognition” in Proc. ICASSP, pp. 7398–7402, 2013.

[7] O. Abdel-Hamid, L. Deng, and D. Yu. “Exploring convolutional neural network structures and optimization for speech recognition,” Proc. Interspeech, 2013

[8] T. N. Sainath, A. Mohamed, B. Kingsbury and B. Ramabhadran “Deep convolutional neural networks for LVCSR” in Automatic Speech Recognition and Understanding (ASRU), 2013 Proc. p. 315-320

[9] K. Vesely, M. Karafiat, and Frantisek Grezl, “Convolutive bottleneck network features for LVCSR,” in Automatic Speech Recognition and Understanding (ASRU), 2011 IEEE Workshop on. IEEE, 2011 pp. 42–47.

[10] S.V Ravuri and N. Morgan ”Easy does it: robust spectro-temporal many-stream asr without fine tuning streams”, Proc. ICSASP 2012, pp. 4309-4312

[11] Y. Shao, Z. Jin, D. Wang and S. Srinivasan “An auditory-based feature for robust speech recognition” Proc. Interspeech 2009, Sep 2009, pp. 4625-4628

[12] O. Kalinli, M.L. Seltzer, and A. Acero, “Noise adaptive training using a vector Taylor series approach for noise robust automatic speech recognition,” in Proc. ICASSP, 2009, pp. 3825-3828

[13] F. Flego and M. J. F. Gales, “Factor Analysis Based VTS Discriminative Adaptive Training” Proc. ICASSP. IEEE, 2012, pp. 4669–4672.

[14] H. Franco, M. Graciarena, and A. Mandal, “Normalized amplitude modulation features for large vocabulary noise-robust speech recognition”, Proc. ICASSP 2012, March 2012, pp. 4117-4120

[15] Speech Processing, Transmission and Quality Aspects (STQ); Distributed Speech Recognition; Adv. Front-end Feature Extraction Algorithm; Compression Algorithms, ETSI ES 202 050 Ver. 1.1.5, 2007

[16] C. Kim and R. M. Stern, “Feature extraction for robust speech recognition based on maximizing the sharpness of the power distribution and on power flooring”, in Proc. ICASSP, pp. 4574–4577, 2010.

[17] M. Kleinschmidt, “Localized spectro-temporal features for automatic speech recognition,” in Proc. of Eurospeech, 2003, Sep 2003, pp. 2573–2576.

[18] H. Hermansky and S. Sharma, “Temporal patterns (TRAPs) in ASR of noisy speech,” Proc. ICASSP 1999, March 1999, pp. 289-292 vol. 1.

[19] B.Y. Chen, Q. Zhu, and N. Morgan, “A Neural Network for Learning Long-Term Temporal Features for Speech Recognition,” Proc. ICASSP 2005, March 2005, pp. 945-948

[20] S.Y. Chang, N. Morgan “Informative spectro-temporal bottleneck features for noise-robust speech recognition”, Proc. Interspeech 2013

[21] E. Bocchieri and D. Dimitriadis “Investigating deep neural network based transforms of robust audio features for LVCSR” in Proc. ICASSP, pp. 6709–6713, 2013.

[22] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” Proceedings of the IEEE, 86(11), 2278–2324, 1998.

[23] N. Parihar, J. Picone, D. Pearce, H.G. Hirsch, "Performance analysis of the Aurora large vocabulary baseline system," Proceedings of the European Signal Processing Conference, Vienna, Austria, 2004.

[24] “Renoiser web page,” http://labrosa.ee.columbia.edu/projects/renoiser/create_wsj.html

[25] K.Walker and S. Strassel, “The rats radio traffic collection system,” in Proc. of ISCA Odyssey, 2012

[26] G. Hinton, “A practical guide to training restricted Boltzmann machines,” Tech. Rep. UTML TR 2010-003, University of Toronto, 2010.