EFFECTIVE JOINT TRAINING OF DENOISING FEATURE SPACE TRANSFORMS AND NEURAL NETWORK BASED ACOUSTIC MODELS

Takashi Fukuda, Osamu Ichikawa
Gakuto Kurata, Ryuki Tachibana

IBM Watson Multimodal
Chuo-ku Hakozaki, Tokyo, 103-8510, JAPAN
{fukuda, ichikaw, gakuto, ryuki}@jp.ibm.com

Samuel Thomas, Bhuvana Ramabhadran

IBM Watson Multimodal
Yorktown Heights, NY 10598, US
{sthomas, bhuvana}@us.ibm.com

ABSTRACT

Neural Network (NN) based acoustic frontends, such as denoising autoencoders, are actively being investigated to improve the robustness of NN based acoustic models to various noise conditions. In recent work the joint training of such frontends with backend NNs has been shown to significantly improve speech recognition performance. In this paper, we propose an effective algorithm to jointly train such a denoising feature space transform and a NN based acoustic model with various kinds of data. Our proposed method first pretrains a Convolutional Neural Network (CNN) based denoising frontend and then jointly trains this frontend with a NN backend acoustic model. In the unsupervised pretraining stage, the frontend is designed to estimate clean log Mel-filterbank features from noisy log-power spectral input features. A subsequent multi-stage training of the proposed frontend, with the dropout technique applied only at the joint layer between the frontend and backend NNs, leads to significant improvements in the overall performance. On the Aurora-4 task, our proposed system achieves an average WER of 9.98%. This is a 9.0% relative improvement over one of the best reported speaker independent baseline system's performance. A final semi-supervised adaptation of the frontend NN, similar to feature space adaptation, reduces the average WER to 7.39%, a further relative WER improvement of 25%.

Index Terms— Speech recognition, neural network, CNN, joint training, denoising autoencoder

1. INTRODUCTION

Despite recent significant advances in acoustic modeling with Deep Neural Networks (DNNs) [1, 2, 3, 4], Automatic Speech Recognition (ASR) systems are still not robust enough to deal with noise, speaker and domain variabilities unseen during training. To improve speech recognition performances in these settings, four broad classes of techniques are actively being pursued with DNN based acoustic models - feature compensation or signal enhancement, feature or model space adaptation, data augmentation followed by multicondition style training and training with side information about undesired variabilities in the signal.

Under the first class of techniques, DNNs are trained using noise robust feature representations [5, 6, 7, 8] compensated for additive and convolutive distortions. In addition to feature level compensation, signal denoising or enhancement techniques like Weiner filtering, spectral subtraction, non-negative matrix factorization are often used as well [9, 10, 11, 12, 13, 14]. The second class of methods are generally variants or extensions of adaptation techniques previously studied and employed with Gaussian Mixture Models (GMMs). In conjunction with GMMs, DNNs in this case are trained on features transformed using techniques like feature-space MLLR [15] or the weights and biases of the DNNs are adapted similar to the adaptation of Gaussian means and variances [16]. Neural network based regularization schemes like modifications to network non-linearities [17] can also be placed under this category of techniques.

More recently significant performance gains have been observed under the third class of techniques using multicondition sytle training with neural networks after data augmentation with real and artificially created noises [18]. Although this approach increases network training complexities, it can be combined with noise robust feature representations or compensation techniques described earlier. In the fourth class of techniques, information about undesired noise and speaker variabilities is provided to allow the network to automatically learn normalization transformations during training. In this approach, i-vectors are concatenated along with traditional acoustic features for training DNN models [19]. Similar to i-vector representations, estimates of noise [20] and speaker codes [21] have also been explicitly provided to DNNs for noise and speaker compensation.

Interestingly the above mentioned broad classes overlap significantly as many recent neural network based robustness techniques build on several of these categories together [22, 23, 24, 25]. In this paper we develop a neural network feature frontend that combines techniques from several of the above categories as well. We begin by first training a denoising autoencoder as a frontend NN to learn denoising feature-spece transforms using multicondition style training. Autoencoders have traditionally been used as models for initializing and pretraining deep neural networks to learn useful representations [26]. However more recently these networks have been shown to be useful as denoising frontends that estimate clean acoustic features from noisy inputs [27, 28]. After learning denoising feature space transforms, the frontend is combined with a neural network based acoustic model and jointly trained similar to the approaches in [29, 30]. In this paper we focus on a speaker-independent system and propose an effective multi-stage training algorithm for denoising frontends. After pre-training the frontend NN alone with a parallel corpus, the frontend is jointly trained along with the backend NN. The joint training updates network parameters of the entire NN, including the denoising frontend, with a dropout technique for specific frequency bands at the joining layer and refines the denoising frontend to better fit the backend NN. During test time, similar to fMLLR, we learn transforms that further match the test data using semi-supervised transcripts from sucessive decodes.

ASR experiments in this paper are performed on the Aurora 4 task [31] - a medium vocabulary task, based on the Wall Street Journal corpus. Using the task's multicondition experimental framework which utilizes a variety of noise types for train and several test sets

containing both seen and unseen noise distortions, we investigate the usefulness of our proposed training algorithm for joint NN based denoising frontend and backend acoustic models.

2. DENOISING AUTOENCODERS

To reconstruct clean acoustic features x_n from corrupted input acoustic features $\hat{x_n}$, autoencoders are trained on parallel noisy and clean corpora to minimize the mean squared error loss function $||y_n - x_n||^2$ between the "cleaned up" features y_n and the actual clean features. The mapping layer (encoding layer) of typical denoising autoencoders have the form:

$$h_i(z_i) = f(W_i z_i + b_i), \quad (1)$$

where z_i is the input to the i-th hidden layer. W_i and b_i are the weight matrix and bias vector, respectively. $f()$ is a non-linearity such as a sigmoid, tanh, or ReLU. A regularization term is often included in the loss function to prevent over-fitting. Acoustic features $\hat{x_n}$ and x_n can also include neighboring left and right frames as the acoustic context. Denoising autoencoders are typically configured with fully connected networks and use the same feature representations in inputs and outputs.

As an extension to traditional training techniques, these frontends have also been combined and jointly trained with neural network based acoustic models [29, 30]. In these jointly trained models, the output layer of the NN denoiser frontend is treated as the input layer of the backend acoustic model and integrated as a hidden layer of the whole network. After joint training, since the frontend network is refined to better fit the backend NN, the combined networks have shown to yield better classification performances.

3. JOINTLY TRAINED DENOISING FRONTEND AND ACOUSTIC MODELS

3.1. CNN-based denoising frontend

Figure 1 illustrates our proposed denoising frontend and acoustic model backend framework. Unlike other denoising autoencoders, our NN-based frontend is a CNN designed to estimate clean log Mel-filterbank features from noisy log power spectrum features. We hypothesize that such a frontend will learn both feature-space and denoising transforms that not only denoise the signal but also reduce the high dimensional noisy log-power spectrum features to lower dimensional features. As shown in Figure 1, our frontend not only has convolutional layers but also fully connected layers. To allow for seamless integration with the backend AM, the predicted targets of denoising frontends have sufficient acoustic context. Since each convolutional layer filter intrinsically has the capability to extract important variational components while removing unnecessary components from the acoustic features [32], CNN-based frontend NNs realize better feature space transforms than DNN-based frontends.

Figure 2 illustrates examples of 9×9 filters obtained just after training the denoising frontend and those after the joint training. In the figure, red and blue colors represent positive and negative values, respectively. Many filters in the first convolutional layer right after training denoising frontend NN have complicated geometric patterns reflecting both noise components in training data and information that is necessary to map features to lower dimension. Figure 2 suggests that the joint training with targets corresponding to phoneme context dependent states, significantly changes some of the filters so that they capture N-th order spectral patterns in time and frequency

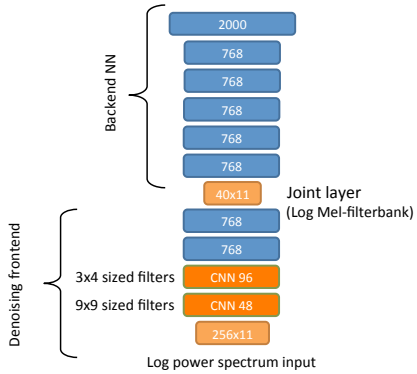

Fig. 1. Denoising feature-space transform and backend acoustic NN.

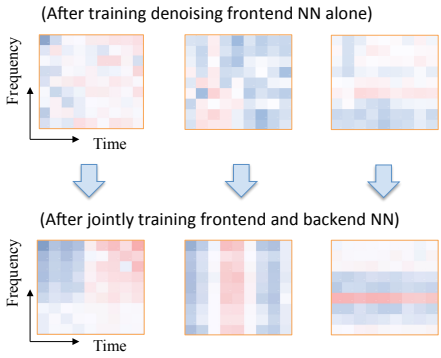

Fig. 2. Examples of 9×9 filters in frontend NN before and after joint training.

plane such as delta features, which are important for phone classification. Approximately half of the filters after the joint training, retain characteristics similar to those learnt after training the denoising frontend.

3.2. Multi stage training strategy for denoisng frontend

When we use multiple frames as the targets of the frontend NNs, it is observed that the output features often have strong correlation between frames and are overly smoothened. As shown in Figure 3, to circumvent this issue we first pretrain the frontend network with a single frame target before expanding the target to N-frames and retraining it. Weights obtained in the first stage of training (with single frame targets) are used as initial weights for the second stage. We expect the trained frontend NN to emphasize the center frame from among the concatenated contexual frames while learning denoising and other feature transforms required to characterize each phoneme.

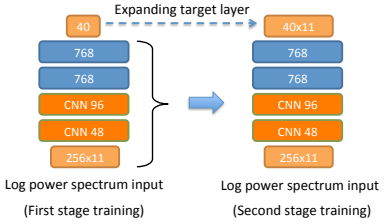

Fig. 3. Multi stage training strategy of denoising NN.

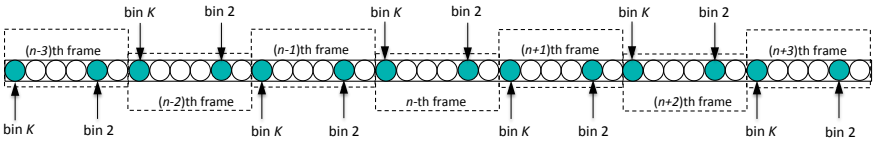

Fig. 4. Dropout at joint layer.

Table 1. Baseline Performance (WER%) of CNN systems. # of layers indicates the total number of hidden layers consisting of convolutional layers and fully connected layers.

System	# of layers	# of params	A	B	C	D	AVG
CNN-1024	7L	8.7 M	4.30	8.23	6.93	15.91	11.15
CNN-768	10L	7.7 M	4.33	7.95	6.89	15.78	10.97

3.3. Frequency-wise dropout at joint layer

When training neural networks on a limited amount of data it has been shown that randomly zeroing, or "dropping out" a fixed percentage of outputs of a given layer can improve test set performance significantly, since dropout training discourages detectors in the network from co-adapting. This is turn limits the capacity of a network and prevents over fitting [33]. In the current setting we hypothesize that incorporating dropout training at the joint layer where the frontend and backend models are combined, can further reduce the detrimental effects from over-smoothing of output frames. We propose applying dropout for specific frequency bands only at the joint layer as shown in Figure 4. We hypothesize that applying dropout will result in a more robust NN being trained as it simulates the effect of missing information by making the zeroed frequency bands unavailable, thereby forcing other parts of the NN to compensate for the missing information (similar to missing feature theory [34]).

4. EXPERIMENTS

4.1. Baseline evaluations

The proposed techniques are evaluated using a series of experiments on Aurora 4 - a medium vocabulary task, based on the Wall Street Journal corpus [31]. Neural network based acoustic models in our experiments are trained on standard train and test data partitions for the task similar to [35]. Test results are reported on 4 subsets commonly referred to as clean (test set A), noisy (test set B), clean with channel distortion (test set C) and noisy with channel distortion (test set D).

CNN-based baselines with different numbers of hidden layers and units are built for the multi-condition task. Baseline CNN systems use 40 dimensional log Mel-frequency spectra augmented with Δ and $\Delta\Delta$s as inputs. The log Mel-frequency spectra are extracted by first applying mel scale integrators on power spectral estimates in short analysis windows (25 ms) of the signal followed by the log transform. Each frame of speech is also appended with a context of 11 frames after applying a speaker independent global mean and variance normalization. The CNN baselines use two convolutional layers with 128 and 256 hidden nodes each in addition to five and eight fully connected layers with 1024 and 768 units per layer, respectively. The CNN baseline systems with ReLU non-linearity estimate posterior probabilities of 2000 output targets. When ReLU non-linearity is used, fixed dropout of 20% are applied on the all fully connected hidden layers. All of the 128 nodes in the first feature extracting layer are attached with 9×9 filters that are two dimensionally convolved with the input log Mel-filterbank representations. The second feature extracting layer with 256 nodes has a similar set of 3×4 filters that processes the non-linear activations after max pooling from the preceding layer. The non-linear outputs from the second feature extracting layer are then passed onto the subsequent fully connected layers. The CNNs are discriminatively pre-trained before being fully trained to convergence. These baseline systems correspond to state-of-the-art systems for this task [35].

After training, the CNN models are decoded with the task-standard WSJ0 bigram language model. Table 1 shows the baseline results. Both the "CNN-768" and "CNN-1024" baseline systems have almost the same numbers of network parameters. The "CNN-768" system in particular has a similar topology to our proposed system combined with frontend NN as illustrated in Figure 1.

4.2. DNN-based vs. CNN-based denoising frontend

In our first set of experiments we separately train frontend NNs and backend acoustic models to compare the performance of a CNN-based denoiser with a DNN-based one. The CNN-based denoising frontend has two convolutional layers with 48 and 96 hidden nodes respectively. The following two fully connected layers have 1024 hidden units each. While all the hidden nodes in the first convolutional layer are connected to 9×9 filters, the nodes in the second layers are connected to 4×3 filters. Max-pooling in frequency is applied to the subsampling layers with a pooling size of 5 for the first layer and 2 for the second. The denoiser is trained to estimate 40-dimensional clean log Mel-filterbank features provided as input without any additional feature context. No multi-stage training of the frontend NN addressed in Section 3.2 was applied here. In all these experiments, the backend AM is trained with log Mel-filterbank features from the trained denoiser frontend with a context of $\pm$ 5 frames.

The denoiser frontend is trained with the mean square error (MSE) criterion to convergence using stochastic gradient descent on parallel training data available in the Aurora setup. The parallel training data is fully randomized at the frame level and partitioned into minibatches of 250 frames. Prior to complete training, the fully connected layers of the network are grown incrementally with layer-wise MSE based pre-training on one pass of the data. The backend NN acoustic models have architectures with 1024 hidden units for the five fully connected layers. In this section, the backend NN was trained with sigmoid non-linearity and no dropout was applied. Table 2 shows the results from this set of experiments. As

Table 2. Performance (WER%) using separately trained denoisers and a fully connected DNN backend acoustic model.

Frontend NN	A	B	C	D	AVG
DNN	5.75	10.59	12.16	22.76	15.57
CNN	5.29	9.82	9.75	21.26	14.41

can be seen from the table, the CNN-based denoiser performs better than the DNN-based one, suggesting that the CNN frontend is more effective.

4.3. Multi-stage training strategy for denoising NN

In this subsection, we evaluate the training strategy of denoising NNs proposed in Section 3.2. The topology of the CNN-based denoising frontend is the same as Section 4.2 except for using 768 hidden units in the fully connected layers. The denoiser is trained with both single and multi-stage training strategies to estimate 40-dimensional clean log Mel-filterbank outputs with a context of $\pm$ 5 frames from 256-dimensional log spectral features. The frontend NN and backend NN are combined as shown in Figure 1 and jointly trained after the denoiser is constructed. The size of hidden units in the backend NN portion is also changed from 1024 to 768, considering the total number of hidden layers that make up the frontend and backend. During the joint training the weights of the NN backend and the CNN frontend were both updated. Table 3 compares the proposed single stage and multi-stage strategies. In the single stage system the frontend NN is trained with multiple frame targets without any intermediate context expansion. As can be seen from the table, the CNN-based denoiser trained with the multi-stage strategy performs better than that trained with the single stage strategy. Compared to the AM without any denoisers shown in Table 1, CNN-based frontend trained with the multi-stage strategy improved performance in Set A, B, and C.

Table 3. Comparison of a single and a multi-stage training strategy for denoising NN.

Training scheme	A	B	C	D	AVG
Single stage	3.79	7.11	6.99	16.11	10.72
Multi-stage	4.05	6.97	6.65	15.88	10.56

4.4. Results on dropout for specific frequency bands

In our third set of experiments we compare dropout techniques for the joint training of frontend and backend NNs. The architecture of the combined NN is the same as described in the previous section. The output layer of the CNN frontend is connected to the input layer of backend NN acoustic models via a hidden layer referred to as the joint layer.

As shown in Table 4, using dropout only at the joint layer with the algorithm described in Section 3.3 significantly improves the performance on all of the test cases compared to the systems without using any dropout. Dropout for all hidden layers of the backend NN ("All layers" in the Table 4) also shows better performance than a no dropout system, but the improvement is marginal. The final jointly trained NN has 2 convolutional layers and 8 fully connected layers, which is a similar topology to the baseline "CNN-768" in Table 1. These results clearly show the advantage of the proposed training algorithm. Table 4 also shows the results of applying sequence training to our best system. The final sequence trained system achieved 9.0% relative improvement compared to the best baseline system using ReLU non-linearity and bigram LM. This is one of the best recognition performances reported in the literature in comparison with similar speaker-independent systems without using any noise adaptive training (NAT) [36].

4.5. Test time adaptation of frontend and acoustic model

This section investigates the impact of semi-supervised adaptation in combination with the proposed method. Table 5 illustrates the performance obtained when weight decay based semi-supervised adaptation (WDA) [37] is applied to the test data set. Three scenarios were studied wherein, all layers, the denoising frontend layers, and only CNN denoising layers, were adapted. We hypothesize that adapting just the frontend CNN layers can have the effect of a feature space transform applied to the entire test data set. As no speaker information is available for this semi-supervised adaptation, this step serves as a means to further adapt not to test speakers [38] but to the noise and channel effects of the entire test data. As seen in Tables 5 and 6, WDA of the frontend denoising layers yeields the best performance. Four iterations of WDA with improved transcripts further reduced the WER down to 7.39%, which is better than that for baseline CNN system at 8.10%. In contrast, the oracle WER, i.e., WDA using supervised transcripts yields an average WER of 4.92%. Since the frontend layers in our system serve as logmel feature extractors, adaptation of the entire frontend shows the best reduction in WER.

Table 4. Comparison of a dropout strategy during joint training.

Nature of dropout	A	B	C	D	AVG
All layers	4.18	7.54	6.54	15.02	10.46
Joint layer	3.83	6.68	6.20	15.67	10.29
+ Sequence training	3.70	6.58	6.18	15.06	9.98

Table 5. Weight-decay based semi-supervised adaptation (WDA) assuming a batch mode scenario.

System	A	B	C	D	AVG
WDA-AL (All Layers)	3.61	5.75	5.64	12.82	8.62
WDA-FL (Frontend L.)	3.55	5.82	5.55	12.34	8.43
WDA-CL (CNN L.)	3.64	6.41	6.00	14.31	9.57

Table 6. Adaptation results with improved decodes and alignments.

System	A	B	C	D	AVG
WDA: Baseline CNN + 4 repeats	4.26	6.37	5.79	10.85	8.10
WDA-FL + 4 repeats	3.62	5.58	5.55	10.13	7.39
Supervised WDA-FL	2.30	3.43	2.78	7.22	4.92

5. CONCLUSIONS

In this paper we have developed a novel framework for jointly training data-driven feature transforms along with neural network acoustic models for robust ASR and have illustrated several techniques to improve the proposed feature transforms. Novel contributions of this paper include:

a. Techniques that learn and refine transforms in 3 distinct stages using various kinds of data. Unlike previous approaches, in the first step, denoising transforms are learnt in a completely unsupervised fashion using parallel clean and noisy data. The transforms are then refined in a fully supervised second stage in conjunction with training of a NN based backend acoustic model. In the third step, using semi-supervised transcripts, the learnt transforms are finally adapted to test conditions.

b. Training of the denoising frontend first with a single frame target and then expanding to a multiple frame target to avoid an over smoothing between frames,

c. Introduction of frequency-wise dropout at the combination layer while jointly training the frontend transforms and backend acoustic model NN,

d. Application of a weight decay based adaptation technique to the denoising frontend using semi-supervised data at test time.

6. REFERENCES

[1] T. N. Sainath, B. Kingsbury, B. Ramabhadran, P. Fousek, P. Novak, and A. R. Mohamed, "Making deep belief networks effective for large vocabulary continuous speech recognition," in *IEEE Automatic Speech Recognition and Understanding (ASRU)*, 2011, pp. 30–35.

[2] F. Seide, G. Li, and D. Yu, "Conversational speech transcription using context-dependent deep neural networks." in *Interspeech*, 2011, pp. 437–440.

[3] H. Soltau, G. Saon, and T. N. Sainath, "Joint training of convolutional and non-convolutional neural networks." in *IEEE Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 5572–5576.

[4] G. Saon, H.-K. J. Kuo, S. Rennie, and M. Picheny, "The IBM 2015 English conversational telephone speech recognition system," *arXiv preprint arXiv:1505.05899*, 2015.

[5] O. Kalinli, N. L. Seltzer, and A. Acero, "Noise adaptive training using a vector Taylor series approach for noise robust automatic speech recognition," in *IEEE Acoustics, Speech and Signal Processing (ICASSP)*, 2009, pp. 3825–3828.

[6] V. Mitra, H. Franco, M. Graciarena, and A. Mandal, "Normalized amplitude modulation features for large vocabulary noise-robust speech recognition," in *IEEE Acoustics, Speech and Signal Processing (ICASSP)*, 2012, pp. 4117–4120.

[7] S. Ganapathy, S. Thomas, and H. Hermansky, "Robust spectro-temporal features based on autoregressive models of Hilbert envelopes," in *IEEE Acoustics, Speech and Signal Processing (ICASSP)*, 2010, pp. 4286–4289.

[8] C. Kim and R. M. Stern, "Power-normalized cepstral coefficients (PNCC) for robust speech recognition," in *IEEE Acoustics, Speech and Signal Processing (ICASSP)*, 2012, pp. 4101–4104.

[9] S. F. Boll, "Suppression of acoustic noise in speech using spectral subtraction," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 27, no. 2, pp. 113–120, 1979.

[10] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Advances in neural information processing systems*, 2001, pp. 556–562.

[11] D. Macho, "Evaluation of a noise-robust DSR front-end on Aurora databases," in *ICSLP*, 2002, pp. 17–20.

[12] D. Yu, L. Deng, J. Droppo, J. Wu, Y. Gong, and A. Acero, "A minimum-mean-square-error noise reduction algorithm on Mel-frequency cepstra for robust speech recognition," in *IEEE Acoustics, Speech and Signal Processing (ICASSP)*, 2008, pp. 4041–4044.

[13] B. Li, Y. Tsao, and K. C. Sim, "An investigation of spectral restoration algorithms for deep neural networks based noise robust speech recognition." in *Interspeech*, 2013, pp. 3002–3006.

[14] M. Delcroix, Y. Kubo, T. Nakatani, and A. Nakamura, "Is speech enhancement pre-processing still relevant when using deep neural networks for acoustic modeling?" in *Interspeech*, 2013, pp. 2992–2996.

[15] M. J. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," *Computer speech & language*, vol. 12, no. 2, pp. 75–98, 1998.

[16] K. Yao, D. Yu, F. Seide, H. Su, L. Deng, and Y. Gong, "Adaptation of context-dependent deep neural networks for automatic speech recognition," in *IEEE Spoken Language Technology Workshop (SLT)*, 2012, pp. 366–369.

[17] S. Sivadas, Z. Wu, and M. Bin, "Investigation of parametric rectified linear units for noise robust speech recognition," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[18] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Interspeech*, 2015.

[19] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using i-vectors." in *IEEE Automatic Speech Recognition and Understanding (ASRU)*, 2013, pp. 55–59.

[20] M. L. Seltzer, D. Yu, and Y. Wang, "An investigation of deep neural networks for noise robust speech recognition," in *IEEE Acoustics, Speech and Signal Processing (ICASSP)*, 2013, pp. 7398–7402.

[21] O. Abdel-Hamid and H. Jiang, "Fast speaker adaptation of hybrid NN/HMM model for speech recognition based on discriminative learning of speaker code," in *IEEE Acoustics, Speech and Signal Processing (ICASSP)*, 2013, pp. 7942–7946.

[22] A. L. Maas, Q. V. Le, T. M. O'Neil, O. Vinyals, P. Nguyen, and A. Y. Ng, "Recurrent neural networks for noise reduction in robust asr." in *Interspeech*, 2012, pp. 22–25.

[23] T. Yoshioka, K. Ohnishi, F. Fang, and T. Nakatani, "Noise robust speech recognition using recent developments in neural networks for computer vision," in *IEEE Acoustics, Speech and Signal Processing (ICASSP)*, 2016.

[24] J. T. Geige, Z. Zhang, F. Weninger, B. Schuller, and G. Rigoll, "Robust speech recognition using long short-term memory recurrent neural networks for hybrid acoustic modeling," in *Interspeech*, 2014.

[25] Y. Zhang, G. Chen, D. Yu, K. Yao, S. Khudanpur, and J. Glass, "Highway long short-term memory rnns for distant speech recognition," in *IEEE Acoustics, Speech and Signal Processing (ICASSP)*, 2016.

[26] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *The Journal of Machine Learning Research*, vol. 11, pp. 3371–3408, 2010.

[27] R. Hsiao, J. Ma, W. Hartmann, M. Karafiat, F. Grézl, L. Burget, I. Szoke, J. Cernocky, S. Watanabe, and Z. Chen, "Robust speech recognition in unknown reverberant and noisy conditions," in *IEEE Automatic Speech Recognition and Understanding (ASRU)*, 2015.

[28] M. Mimura, S. Sakai, and T. Kawahara, "Joint optimization of denoising autoencoder and dnn acoustic model based on multi-target learning for noisy speech recognition," in *Interspeech 2016*, 2016, pp. 3803–3807.

[29] A. Narayanan and D. Wang, "Joint noise adaptive training for robust automatic speech recognition," in *IEEE Acoustics, Speech and Signal Processing (ICASSP)*, 2014, pp. 2504–2508.

[30] T. Gao, J. Du, L.-R. Dai, and C.-H. Lee, "Joint training of front-end and back-end deep neural networks for robust speech recognition," in *IEEE Acoustics, Speech and Signal Processing (ICASSP)*, 2015, pp. 4375–4379.

[31] N. Parihar and J. Picone, "Aurora Working Group: DSP Front-end and LVCSR Evaluation AU/384/02," Inst. for Signal and Information Processing, Mississippi State University, Tech. Rep., 2002.

[32] T. Fukuda, O. Ichikawa, and R. Tachibana, "Convolutional neural network pre-trained with projection matrices on linear discriminant analysis," in *IEEE Acoustics, Speech and Signal Processing (ICASSP)*, 2016, pp. 5345–5349.

[33] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[34] R. P. Lippmann and B. A. Carlson, "Using missing feature theory to actively select features for robust speech recognition with interruptions, filtering and noise," in *Eurospeech*, vol. 97, 1997, pp. 37–40.

[35] S. Rennie, V. Goel, and S. Thomas, "Annealed dropout training of deep networks," in *SLT*, 2014, pp. 159–164.

[36] Y. Qian, M. Yin, Y. You, and K. Yu, "Multi-task joint-learning of deep neural networkds for robust speech recognition," in *ASRU*, 2015.

[37] M. Suzuki, R. Tachibana, S. Thomas, B. Ramabhadran, and G. Saon, "Domain adaptation of cnn based acoustic models under limited resource settings," in *Interspeech*, 2016.

[38] L. Samarakoon and K. C. Sim, "Multi-attribute factorized hidden layer adaptation for dnn acoustic models," in *Interspeech*, 2016.