ROBUST SPEECH RECOGNITION IN UNKNOWN REVERBERANT AND NOISY CONDITIONS

Roger Hsiao[1], Jeff Ma[1], William Hartmann[1], Martin Karafiát[2], František Grézl[2], Lukáš Burget[2], Igor Szöke[2], Jan "Honza" Černocký[2], Shinji Watanabe[3], Zhuo Chen[3], Sri Harish Mallidi[4], Hynek Hermansky[4], Stavros Tsakalidis[1] and Richard Schwartz[1]

[1]Raytheon BBN Technologies, Cambridge, MA, USA
[2]Brno University of Technology, Speech@FIT and IT4I Center of Excellence, Czech Republic
[3]Mitsubishi Electric Research Laboratories, Cambridge, MA, USA
[4]Johns Hopkins University, Baltimore, Maryland USA

ABSTRACT

In this paper, we describe our work on the ASpIRE (Automatic Speech recognition In Reverberant Environments) challenge, which aims to assess the robustness of automatic speech recognition (ASR) systems. The main characteristic of the challenge is developing a high-performance system without access to matched training and development data. While the evaluation data are recorded with far-field microphones in noisy and reverberant rooms, the training data are telephone speech and close talking. Our approach to this challenge includes speech enhancement, neural network methods and acoustic model adaptation, We show that these techniques can successfully alleviate the performance degradation due to noisy audio and data mismatch.

Index Terms— ASpIRE challenge, robust speech recognition

1. INTRODUCTION

The ASpIRE (Automatic Speech recognition In Reverberant Environments) challenge aims to assess robustness of automatic speech recognition (ASR) systems to a variety of acoustic environments and recording scenarios [1]. The evaluation data are recorded with far-field microphones in noisy and reverberant rooms. The main characteristic of the challenge is developing a high-performance system without having access to matched training and development data.

In order to minimize the mismatch between the training and test conditions we explored three methods. The first method is motivated by the fact that models trained on a specific corpus will perform well when incorporated into an ASR system evaluated on new speech of a similar nature. The challenge is to create a corpus that matches unseen reverberant and noisy conditions. While this is a nearly unattainable goal, the hope is that by adding enough variability in the training data, the model—trained on such data—will be more robust to unseen conditions. This method is the most straightforward but requires training and using a separate acoustic model for noisy and reverberant conditions.

A reciprocal approach that alleviates the need of training a noisy acoustic model is to enhance (i.e. de-noise and de-reverberate) the test audio. This method utilizes a Neural Network (NN) auto-encoder for speech enhancement [2, 3, 4]. In simple terms, a NN is trained using a parallel clean-to-degraded corpus to learn a transformation that maps degraded speech to clean. Thus, a typical (i.e. trained on clean corpus) ASR system can be used to process the enhanced test audio.

The final method is using neural network based acoustic modeling techniques to enhance the performance of a typical ASR system. In this work, we first explore feature extraction techniques based on Stacked bottleneck (SBN) hierarcy; a tandem structure with features generated by a hierarchy of two Neural Networks (NN) [5]. We employ this SBN scheme so that the first NN is responsible for low-level feature extraction and the second one fits the features to the following discriminatively trained acoustic model. In addition to the feature extraction, we also investigate a unsupervised adaptation technique for neural network based on a linear least squares method (LLS) [6]. In this paper, we show that the adaptation technique is effective to alleviate the data mismatch issues for NN based acoustic models.

All three methods rely on a parallel artificially-generated clean-to-degraded (i.e. reverberant and noisy) corpus. Since the essence of the challenge is robustness to various unseen test conditions the crux of the problem is how to synthesize data that would be close to the target speech conditions. We employed a large and diverse noising and reverberation procedure first presented in [7].

This paper is organized as follows: In Section 2 we describe the ASpIRE data and our approach to augmenting the training data to improve the system performance. Section 3 explains our method for audio enhancement using a neural network auto-encoder. We then describe our systems in Sec-

Table 1. Amount of training and testing data.

Data-set	No. of conversation sides	Size [h]
Fisher 1+2	23398	1800
ASpIRE Dev	30	3.4
ASpIRE Dev-test	60	5.8
ASpIRE Eval	120	6.1

tion 5 and conclude our work in Section 6.

2. THE DATA AND ITS AUGMENTATION

Fisher English database Part 1 and 2, comprised of over 20k telephone conversation sides, was the only data allowed for training in the ASpRIRE challenge. Three sets of test data were defined by the ASpIRE challenge. The ASpIRE dev set contains 30 recordings from various rooms and noisy conditions recorded with 16 kHz sampling rate (as opposed to the 8 kHz telephone data in Fisher). Similarly, the dev-test set and evaluation set contain 60 and 120 recordings respectively in a similar condition. Therefore, all test data were first down-sampled to 8 kHz to match the training data. For sizes of the data-sets, see Table 1.

2.1. Noising

Our training data was processed by artificially adding the following types of noises:

- real fan stationary noises: 115 samples (4 minutes long) were taken from the Freesound library[1]. These samples belong to categories: fan, AC, hvac, street, ventilation. Their character is stationary (sound of fan or AC).

- real background stationary noises: 170 samples (4 minutes long) from Freesound. These samples belong to categories: city, fan, AC, restaurant, shop, crowd, library, office, workshop. Their character is mainly stationary, with some minor portion of transient noises and babbling.

- real background transient noises: 60 samples (4 minutes long) from Freesound. These samples belong to categories: dishes, motor, workshop, doors, city, keyboard, library, office. The character is mainly transient, with some minor portion of stationary noises.

- babbling noises: 25 samples (4 minutes long), each created by merging speech from 100 random speakers from Fisher database using speech activity detector.

- ASpIRE noises: 140 samples (10-60 second long) of noises extracted from ASpIRE dev data using speech activity detector. This was conforming to the evaluation rules.

- Artificial noises: 7 samples (4 minutes long) of artificial generated noises: various spectral modifications of white noise + 50 and 100 Hz hum.

[1]http://www.freesound.org

2.2. Reverberation

We generated artificial room impulse responses (IR) using ”Room Impulse Response Generator” tool from E. Habets[2]. The tool can model the size of a room (3 dimensions), reflectivity of each wall, type of microphone, position of source and microphone, orientation of microphone toward the audio source, and number of bounces (reflections) of the signal. Each room model consists of a pair of IR. One is used to reverberate (convolution with IR) the speech signal and the other is used to reverberate the noise signal that are then mixed into a single recording. Only the coordinates of the audio sources (speech/noise) differ for each of the IRs in each pair. We randomly set all parameters of the room for each room model.

2.3. Composition of the training set

We used the `fant` tool [8] to mix reverberated speech and reverberated noise with a given SNR. The speech signal was compensated for the delay caused by the reverberation (to match the timing with the original one). The following training datasets with artificially corrupted speech were created:

Large rooms dataset consists of 1800 hours of clean Fisher data augmented with another 3 copies of artificially corrupted Fisher data. IRs were generated for rooms where each dimension was limited to the range of 2–22 meters. Noises were added at SNRs ranging from 0dB to 45dB. The noise types used are: real fan stationary noises, real background stationary noises, babbling noises, and artificial noises.

Small rooms is a dataset similar to *large rooms* with the addition of real background transient noises and ASpIRE noises. After listening to the data from *large rooms* dataset and comparing it with the relatively less reverberant ASpIRE dev data, we also decided to limit the room dimensions to the range of 2–5 meters.

Auto-encoder training dataset is similar to *small rooms*. Two noises were always added into one recording: one random stationary noise and one random transient noise.

Enhanced dataset uses room dimensions 2–5 meters and real fan stationary noises, real background stationary noises, babbling noises, artificial noises added to speech at SNRs ranging from 15–45dB.This data is further enhanced (cleaned) by the auto-encoder described in section 3. This training was created in order to learn the artifacts, which

[2]http://www.audiolabs-erlangen.de/content/05-fau/professor/00-habets/05-software/01-rir-generator/rir_generator.pdf

can be introduced into the speech signal during the speech enhancement.

3. AUDIO ENHANCEMENT BY DNN AUTO-ENCODER

The role of the auto-encoder is to enhance (de-noise and de-reverberate) the speech signal. It is trained on the artificially created parallel clean-noisy Fisher corpora as described in the previous section. The input to the NN is 129 dimensional vectors of log spectra stacked over 31 frames (i.e. 3999 dimensional vector). The desired output is a 129 dimensional vector (again log spectrum) corresponding to the clean version of the central input frame. A standard feed-forward architecture is used: 3999 inputs, 3 hidden layers with 1500 neurons, 129 outputs, and tanh nonlinearities in the hidden layers. The NN is initialized in such a way that it (approximately) passes its input to the output and it is trained using conventional stochastic gradient descent to minimize the MSE objective.

We have experimented with different strategies of normalizing NN input and output. To achieve good performance, utterance level mean and variance normalization is applied to both the NN input and the desired NN output. To synthesize the cleaned speech log spectrum, the NN output is de-normalized based on the global mean and variance of clean speech. Prior to passing the data through the NN, the ASpIRE data is passed through a filter to simulate the characteristics of a telephone channel.

4. SYSTEM DESCRIPTION

4.1. BUT GMM System

The BUT ASR system was a GMM-HMM with cross-word tied-states triphones. Twelve Gaussian mixtures per state were trained from scratch using mix-up maximum likelihood training. Final word transcriptions were decoded using a 3-gram Language Model (LM) trained only on the Fisher corpus. The GMM-HMM was trained via MPE. The features used by the BUT GMM system consist of 39-dimensional PLP-HLDA features, 30-dimensional Stacked Bottleneck Neural Network (SBN) features, and four pitch features. These features are combined using the region dependent transform (RDT) method [9]. The GMM and the SBN features are trained on the augmented Fisher corpus consisting of artificial noises described in Section 2.

4.2. BBN GMM System

The BBN GMM training was similar to the procedure described in previous work [9, 10]. We use multi-layer perceptron (MLP) and PLP features. The MLP and PLP features are first projected to a lower dimensional feature vector through the RDT method [9]. The GMM models are finally trained on the RDT-projected features. In the training for this work we made a few changes,

- As described in [10], frequency domain linear prediction (FDLP) features [11] were used to train MLP. The FDLP features were extracted with a 10-second long window, producing a 476-dimensional vector. Such high-dimensional features not only require large storage spaces but also could cause I/O burdens during training, especially when the amount of training data becomes big. Instead, we used Mel filter bank (MelFB) features to train the MLP. For each frame, we compute 17 MelFB features and the energy with a 25ms window. The final feature vector was produced by concatenating 31 frames, giving a 558-dimensional vector (31x18). 17 MelFB features were computed
- Rather than training with the SBN features described in [12, 10], we trained only one MLP for generating bottleneck features, reducing the MLP training time by half. The configuration of the MLP was 558x1500x1500x150x1500xN, where N is the number of cross-word state clusters and the third hidden layer is the bottleneck layer. From this MLP, 150-dimensional bottleneck features were generated.
- Instead of concatenating 11 PLP frames, we concatenated 15 frames. Also, we increased the dimension of the RDT-projected features from 46 to 60.

The changes made in the generation of MLP features are mainly due to the time limit of the ASpIRE evaluation.

Compared to the BUT system, the BBN GMM system is trained on the original Fisher corpus without the artificial noises. We used the entire Fisher training data (1800 hours) to train the MLP features. The GMM acoustic models were trained on 800 hours of Fisher. The lexicon and language model were derived from the entire Fisher training transcripts. All models were based on cross-word tied-states quinphones. The GMM models were trained using MPE. The language model is similar to the BUT system which is a 3gram model trained on Fisher corpus. Decoding was performed with unsupervised speaker adaptation. We also utilized cross-adaptation between models. The models were used to decode the various versions of the enhanced ASpIRE audio.

4.3. BBN DNN and Adaptation

The BBN DNN acoustic model uses the same 46-dimension RDT features used by the BBN GMM system. The input to the DNN concatenates 11 frames to form a 506-dimension input. The DNN also consists of six hidden layers and each layer contains 2048 sigmoid units. The DNN is first initialized by a layer-wise discriminative pretraining procedure.

Then we perform cross entropy training and one iteration of sequence discriminative training.

To deal with the data mismatch between the train set and the test, we perform unsupervised speaker adaptation using the linear least square algorithm [6] (LLS). To describe our LLS adaptation algorithm, we define the DNN using the following notions: a DNN may contain $N + 2$ layers where the first layer (indexed with 0) is the input layer and the last layer (indexed with $N + 1$) is the output layer. Layer 1 to N are the hidden layers and each layer contains K_i units. Each unit has an activation function. Mathematically, each layer of the DNN can be evaluated by the following equations,

$$\begin{aligned} z_t^i &= f_i(y_t^i) && (1)\\ y_t^i &= A_i x_t^i + b_i && (2)\\ x_t^i &= z_t^{i-1} \quad \text{if } i > 0 && (3) \end{aligned}$$

where z_t^i is the output of the i-th layer; f_i is the activation function of the i-th layer, say a sigmoid function for the hidden layers and a softmax function for the output layer; A_i and b_i are the weights of the i-th layer; x_t^i is the input to the i-th layer and z_t^{i-1} is the output of the $(i-1)$-th layer which serves as an input to the i-th layer.

Suppose E is the objective function, say cross entropy function, gradient descent would compute $\frac{\partial E}{\partial A_i}$ and $\frac{\partial E}{\partial b_i}$ and optimize the model parameters directly. However, LLS decomposes the optimization problem into two steps. In the first step, it uses gradient descent to optimize the internal representation of a DNN, i.e. y_t^i. Once the internal representation is optimized, we solve a linear least square problem to optimize the DNN weights, i.e. $W_i \equiv [A_i, b_i]$, so that it is closest to the optimized internal representation.

Using the cross entropy function as an example, we compute

$$\begin{aligned} \frac{\partial E}{\partial y_t^i} &= \frac{\partial E}{\partial z_t^i}\frac{\partial z_t^i}{\partial y_t^i} && (4)\\ \hat{y}_t^i &= y_t^i - \lambda \frac{\partial E}{\partial y_t^i} && (5) \end{aligned}$$

so y_t^i can be optimized by back-propagation. The target internal representation $\hat{y}_t^i$ is then approximated by,

$$\begin{aligned} W_i^* &= \arg\min_{W_i} \frac{1}{2}\sum_t ||\hat{y}_t^i - W'\tilde{x}_t^i||_2 && (6)\\ &= (\sum_t \hat{y}_t^i x_t^{i'})(\sum_t \tilde{x}_t^i \tilde{x}_t^{i'})^+ && (7)\\ &\equiv \hat{Y}_T^i (X_T^i)^+ && (8) \end{aligned}$$

There are some advantages of using LLS algorithm for unsupervised adaptation. To solve equation 8, one has to perform pseudo inverse for X_T^i which is not full rank. As a result, the optimization is implicitly regularized by the Frobenius norm which prefers a sparse solution. Also, in our previous work [6], we found that the LLS algorithm is robust for unsupervised adaptation and it is not sensitive to the learning rate used for optimizing y_t^i. In this work, we use the LLS algorithm to adapt the entire network, which is different from [6] where we only adapt the last layer of the DNN.

5. EXPERIMENTAL RESULTS

5.1. Experiments on Audio Enhancement and Data Augmentation

For the BBN GMM system, we enhanced the ASpIRE audio using the techniques described in Section 3. In addition, we experimented with several variants of the normalization and noise enhancement procedure:

- small_rooms_MVN1: Utterance level mean and variance normalization is applied independently to each of the 129 dimensions of the log spectra. Both the NN input and the desired NN output are normalized. To synthesize the cleaned speech, the NN output is unnormalized based on the global mean and variance of clean speech.
- small_rooms_MVN2: the same as the previous case, except that the NN is trained with more iterations to achieve better convergence.
- small_rooms_MN1: Utterance level mean normalization but global variance normalization is applied
- small_stat_aspi_tran_rooms_MVN1: same processing as for small_rooms_MVN1, but the noisy data also includes transient noises.
- small_stat_aspi_tran_3ch_midhigh_MVN1: same processing as for small_rooms_MVN1, but the noisy data also includes transient noises and lower SNR levels.

Table 2. WER(%) of the BBN GMM system with different audio enhancement procedures.

System	Enhancement	Dev
GMM	small_rooms_MVN1	37.8
GMM	small_rooms_MVN2	37.6
GMM	small_rooms_MN1	37.6
GMM	small_stat_aspi_tran_rooms_MVN1	37.9
GMM	small_stat_aspi_tran_3ch_midhigh_MVN1	38.0

Table 2 shows the WERs of the BBN GMM system on the ASpIRE dev set using different procedures for audio enhancement. The best WER (37.6%) is obtained by using the small_rooms_MVN2 procedure.

Table 3. WER(%) of the BUT GMM system with different data augmentation and audio enhancement procedures.

SBN	GMM	Enhancement	Dev
Clean	Clean	No	42.3
Large	Large	No	35.6
Clean	Clean	Yes	37.9
Small	Clean	No	38.2
Enhanced	Clean	Yes	36.7

For data augmentation, we evaluated the large rooms and small rooms augmented data sets using the BUT GMM system. Table 3 shows the performance on the ASpIRE dev set. The results demonstrate that training on the noisy data gives the best performance. However, this requires the processing of the augmented data, a potentially significant computational cost. In this experiment, the large rooms data set consists of three copies of artificially corrupted Fisher data, containing over 5000 hours of audio data. In contrast, the systems using enhanced audio had comparable performance while the size of the train set did not increase three fold.

5.2. Cross Adaptation for DNN

We evaluated the performance of the DNN acoustic model and study whether unsupervised adaptation could alleviate the degradation due to noisy environment. Similar to the BBN GMM systems, the DNN system is trained on clean audio and tested on enhanced audio . Prior to training the DNN system, the BBN GMM systems were used to perform feature MLLR (fMLLR) for speaker adaptation. In this experiment, we focused on the effect of cross adaptation using the LLS algorithm. For this DNN, we used the audio enhanced by the small_rooms_MVN2 procedure which yields the best results in the GMM experiments.

Table 4. WER(%) of unsupervised speaker adaptation for the BBN DNN system.

System	Algorithm	Xadapt	Dev	Dev-test
DNN	fMLLR	GMM	38.6	33.8
DNN	fMLLR	BUT	38.3	33.5
DNN	fMLLR+LLS	GMM	37.5	30.6
DNN	fMLLR+LLS	BUT	36.4	30.2

The results in Table 4 show that our proposed LLS algorithm can improve the cross adaptation performance from 38.3% to 36.4% on the dev set, and from 33.5% to 30.2% WER on the dev-test set. This result suggests the DNN can benefit from unsupervised adaptation in noisy and mismatched data conditions.

Table 5. Results (WER %) using ROVER for system combination. The first three results use the standard ROVER approach. The final result weights the confidence scores from the individual systems before combination.

Systems	Dev	Dev-test
Unweighted Combination		
1	35.6	30.3
2	33.5	29.0
7	32.4	27.4
Weighted Combination		
7	32.2	27.4

5.3. ROVER and System Combination

We performed system combination using ROVER with the maxconf setting. The alpha and null confidence parameters were tuned on the development set. In ROVER, each system is treated equally. The final output is chosen through a voting scheme at the word level. We noticed that combining large numbers of systems does not always improve performance. To address this issue, we also explored weighting the confidence scores of the individual systems based on their WER. While this produced improvements on the development set, they did not carry over to the evaluation set. Results can be seen in Table 5. The two system combination refers to the cross-adapted BBN GMM and the BUT GMM. The seven system combination further adds the BNN DNN system, a BUT GMM decoding of enhanced data, and three BBN GMM decodings of various enhanced audio. Combining system outputs significantly improved performance over a single system.

Table 6. WER(%) of the BBN GMM, BBN DNN and BUT GMM systems on ASpIRE evaluation set.

System	Combine	Eval
BBN GMM	xadapt BUT GMM	46.8
BBN DNN	xadapt BUT GMM	47.1
BUT GMM	-	46.3
BBN GMM + BUT GMM	ROVER	43.9

Table 6 summarizes the performance of BBN GMM, BBN DNN and BUT GMM system on the ASpIRE evaluation set.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we describe our work in the ASpIRE challenge. We experiment and evaluate different approaches to tackling the performance degradation due to noise and data mismatch. Our approaches include audio enhancement, data augmentation, unsupervised DNN adaptation, and system combination.

They show significant improvement over the baseline. In the future, we will explore combining audio enhancement and data augmentation, and also more neural network based adaptation techniques.

7. ACKNOWLEDGMENTS

We would like to thank all members of the BBN Speech and Language group for useful discussions. We would also like to acknowledge the help of Hakan Erdogan, Scott Wisdom and John R Hershey from Mitsubishi Electric Research Laboratories (MERL).

8. REFERENCES

[1] Mary Harper, "The Automatic Speech recognition In Reverberant Environments (ASpIRE) Challenge," in *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding*, 2015.

[2] Y. Xu, J. Du, L.-R. Dai, and C.-H. Lee, "An Experimental Study on Speech Enhancement based on Deep Neural Networks," *IEEE Signal processing letters*, vol. 21, no. 1, Jan 2014.

[3] Y. Xu, J. Du, L.-R. Dai, and C.-H. Lee, "Global Variance Equalization for Improving Deep Neural Network based Speech Enhancement," in *IEEE China Summit and International Conference on Signal and Information Processing (ChinaSIP)*, 2014, pp. 71–75.

[4] M. Mimura, S. Sakai, and T. Kawahara, "Reverberant Speech Recognition Combining Deep Neural Networks and Deep Autoencoders," in *Reverb Challenge Workshop*, 2014.

[5] F. Grézl, M. Karafiát, and L. Burget, "Investigation into Bottle-neck Features for Meeting Speech Recognition," in *Proceedings of the INTERSPEECH*, 2009.

[6] R. Hsiao, S. Tsakalidis T. Ng, L. Nguyen, and R. Schwartz, "Unsupervised Adaptation for Deep Neural Network using Linear Least Square Method," in *Proceedings of the INTERSPEECH*, 2015.

[7] Martin Karafiát, František Grézl, Lukáš Burget, Igor Szöke, and Jan "Honza" Černocký, "Three ways to adapt a CTS recognizer to unseen reverberated speech in BUT system for the ASpIRE challenge," in *Proceedings of the INTERSPEECH*, 2015.

[8] H. Hirsch and H. Finster, "The Simulation of Realistic Acoustic Input Scenarios for Speech Recognition Systems," in *Proceedings of the INTERSPEECH*, 2015.

[9] T. Ng, B. Zhang, S. Matsoukas, and L. Nguyen, "Region Dependent Transform on MLP Features for Speech Recognition," in *Proceedings of the INTERSPEECH*, 2011, pp. 221–224.

[10] T. Ng, R. Hsiao, L. Zhang, D. Karakos, S. H. Mallidi, M. Karafiát, K. Veselý, I. Szöke, B. Zhang, L. Nguyen, and R. Schwartz, "Progress in the BBN Keyword Search System for the DARPA RATS Program," in *Proceedings of the INTERSPEECH*, 2014, pp. 959–962.

[11] S. Ganapathy, S. Thomas, and H. Hermansky, "Static and dynamic modulation spectrum for speech recognition," in *Proceedings of the INTERSPEECH*, 2009.

[12] F. Grézl, M. Karafiát, and L. Burget, "Investigation into Bottleneck Featrues for Meeting Speech Recognition," in *Proceedings of the INTERSPEECH*, 2013, pp. 2589–2593.