EVALUATION OF ROBUST SPEECH RECOGNITION ALGORITHMS FOR DISTRIBUTED SPEECH RECOGNITION IN A NOISY AUTOMOBILE ENVIRONMENT

Hong Kook Kim and Richard C. Rose

AT&T Labs-Research, 180 Park Avenue, Florham Park, NJ 07932, USA
E-mail: {hkkim, rose}@research.att.com

ABSTRACT

In this paper, we evaluate the performance of several robust speech recognition algorithms in a noisy automobile environment as characterized by the Finnish SpeechDat–Car ASR task [1]. By applying acoustic feature compensation, model compensation, and speech detection algorithms to this task, a 51% reduction in word error rate (WER) was obtained relative to the ETSI standard ASR front–end. In addition, these same techniques achieved an average 35% WER reduction for clean condition training and multiple condition training on a simulated speech–in–noise task as characterized by the Aurora 2 ASR task [2]. The paper also presents alternatives for how these algorithms can be implemented in a distributed speech recognition framework.

1. INTRODUCTION

With the increased use of mobile wireless devices the need for automatic speech recognition (ASR) over wireless communications channels has increased tremendously. A distributed speech recognition (DSR) approach can provide an efficient solution to wireless voice access in that speech algorithms can be distributed between client and network based implementations depending on the complexity and latency of the algorithms [2]. As part of an effort to standardize the feature analysis algorithm in a DSR framework, the ETSI Aurora group has released several standard databases for developing a new front–end algorithm and compares its performance to that of an existing ETSI standard front–end [3].

In this paper, we first demonstrate the effectiveness of several speech algorithms by evaluating their performance on the Aurora 2 database and the Finnish SpeechDat–Car database which is a subset of the Aurora 3 database. The speech algorithms considered are the cepstrum–domain acoustic feature compensation technique [4], cepstrum mean subtraction [5], and the cepstrum–domain model compensation [6]. We also propose a speech segmentation method which can be deployed in the network side of a client–server based DSR system. A network based implementation is possible because the speech segmentation method is performed using the reconstructed speech from mel–frequency cepstral coefficients (MFCCs) and pitch values [7]. In addition, we address the general problem of how these speech algorithms can be configured in a client-server framework.

In the following sections, we describe the database and ASR system in Section 2. In Section 3, we demonstrate the ASR performance of each of the three algorithms on the Finnish SpeechDat–Car database, where the algorithms are the feature compensation technique, the model compensation algorithm, and the speech segmentation algorithm. In Section 4, the performance of the procedure which combines all the algorithms is compared to that using the ETSI standard front–end on the Finnish SpeechDat–Car and the Aurora 2 database. In Section 5, we address how to configure these algorithms in the client-server model. Finally, we conclude our work in Section 6.

2. DATABASE AND ASR SYSTEM

In this section, we describe the Finnish SpeechDat–Car database and a baseline ASR system configuration that was defined by the ETSI Aurora standard group. The database and the system configuration are used in the following sections for evaluating each of the speech algorithms.

The Finnish SpeechDat–Car database is used in our study [1]. This is a subset of the Aurora 3 database. The standard evaluation procedure as defined for the Aurora 3 task was followed [8], Finnish digit strings were recorded using close–talking microphone and hands–free microphones. There were three conditions: quiet, low noise, and high noise. Quiet condition refers to a car being completely stopped or idling. Low noise refers to the condition where a car was driven at speeds of 40–60 km/h with without window open. High noise refers to high speed driving at speeds of 100–120 km/h, often with music playing. The database was divided for the simulation of three different training–testing conditions including well–matched (WM), medium–mismatch (MM), and high–mismatch (HM). For the detailed explanation of the database, refer to [1].

In this work, each word was modeled by a simple left–to–right 16–state 3–mixture per state whole word hidden Markov model (HMM). In addition, a 3–state 6–mixture silence model and an 1–state 6–mixture pause model were used to model the beginning and ending of each digit string and the between–word pause, respectively. The existing ETSI standard front–end extracts 12 MFCCs and energy once every 10 ms segment of speech signal, and adds delta and delta–delta components to MFCCs and energy. In Section 4, the performance of the ETSI standard front–end will be compared to the techniques described in this paper.

3. SPEECH ALGORITHMS

In this section, we introduce acoustic feature and model compensation techniques for improving ASR performance in noisy environments as represented by the Aurora databases. Fig. 1 shows an ASR system incorporating compensation techniques. The front–end extracts acoustic feature vectors from speech signals and then these vectors are compensated and normalized for mitigating background noise and channel effects. Moreover, the front–end provides a pitch value per analysis frame for the speech segmentation algorithm which will be explained in Section 3.3. Next, model compensation is applied to the HMM model parameters for
compensating the mismatch between training and testing environments. The estimates of noise model parameters for model compensation can be obtained from the acoustic feature vectors [6].

In the baseline front–end, the standard MFCC analysis is performed, and first and second difference MFCCs are concatenated to the 13 static MFCCs to make 39-dimensional MFCC vectors. This is similar to the ETSI standard front–end with minor differences in the structure of the filter-bank and the computation of energy. The first column of Table 1 shows the word error rates (WER) of the baseline front–end under the WM, MM, and HM conditions. The baseline front–end under MM and HM conditions gave larger WERs than under the WM condition because training and testing in the MM and HM conditions were done under different driving conditions and/or different microphone conditions.

3.1. Feature Compensation

3.1.1. Cepstrum subtraction method

The cepstrum subtraction method (CSM) [4] is applied as an acoustic feature compensation technique to speech signals. CSM decomposes noise-corrupted speech into clean speech and noise components in the cepstrum domain without any explicit SNR estimation. CSM was shown to significantly improve word accuracy in noisy environments [4], especially on the Aurora 2 database.

The second column of Table 1 shows the WERs obtained when CSM was included in the baseline. CSM significantly reduced WER by 50% for the HM condition. However, CSM slightly degraded the performance under the MM condition. This is because half of the training data for the MM condition were recorded under quiet conditions and CSM introduced minor distortion in the clean speech signals. On the other hand, CSM also reduced WER for the WM condition where training and testing data were collected under various recording conditions. Overall, CSM reduced the average WER by 30% as shown in the last row of Table 1.

3.1.2. Cepstrum normalization

Acoustic feature vectors are further normalized by using cepstrum mean subtraction (CMS). In addition, the energy parameter is normalized so that the maximum value over one utterance is zero.

Table 2 shows WERs of the baseline front–end, the baseline front–end with CSM (baseline+CMS) and the CSM front–end with CMS (CSM+CMS). As expected, applying CMS to the baseline front–end and the CSM front–end yielded great reduction in WER under the MM and HM conditions rather than the WM condition. CSM+CMS provided smaller WER than baseline+CMS. As a result, CSM+CMS provided an average WER reduction of 51% compared to the baseline front–end.

3.2. Model Compensation

In this subsection, we further improve the ASR performance by applying the cepstrum–domain model combination method [6] which compensates the mean and covariance matrices of HMMs. The method requires the estimation of the mean and covariance of a noise model. In this work, CSM is applied to extract the noise feature vectors, where a noise feature vector is obtained for each analysis frame [6]. The parameters of a single mixture Gaussian noise model are estimated for each utterance. The model compensation is easily implemented by adding the means and covariance matrices of the speech HMM, $\lambda_s$, and those of the noise model, $\lambda_n$, to obtain the noise–corrupted HMM, $\lambda_{nc}$.

In the cepstrum–domain model combination method, we can compensate either the mean vectors of $\lambda_s$ alone (Mean–Only) or the mean and covariance matrices of $\lambda_s$ (Mean–Cov). Table 3 shows the WERs after applying the model compensation technique in two modes: mean–only compensation (CSM+CMS+Mean–Only) and mean–covariance compensation (CSM+CMS+Mean–Cov). In order to compare the WERs of the model combination techniques to that without model compensation, we displayed the WERs of the baseline and CSM+CMS in the first and second columns of Table 3, respectively. Compared to CSM+CMS, the two model compensation techniques gave significant reduction of WER under all the conditions. For the WM condition, the performance of the Mean–Only compensation was slightly better than that of the Mean–Cov compensation. In particular, a big improvement was achieved in the MM and HM conditions because these conditions represented the environments of mismatched acoustic noise and transducer conditions.

3.3. Speech Segmentation

In this subsection, we show the effect of speech segmentation on the Finnish database. It is well–known that accurate boundary detection should improve recognition accuracy and reduce decoding time for ASR in severe background noise. Here, we propose a speech segmentation algorithm which fully utilizes the recon-

![Fig. 1. Speech recognition system and robust speech recognition algorithms which includes acoustic feature compensation, feature normalization, model compensation, and speech segmentation.](image-url)
In this section, we address the problem of configuring the speech algorithms described in Section 3 in the DSR framework. In DSR, it is necessary to assign different portions of these speech algorithms to be implemented on the client and the server. Fig. 3 shows the WERs for the baseline front-end and the front-ends with feature and/or model compensation. ETSI standard front-end, endpoints were obtained for the utterances by forced Viterbi alignment of the reference HMMs with the uncorrupted version of the utterances [10]. For our front-end, segmentation was performed using the unsupervised segmentation algorithm described in Section 3.3. Comparing two front-ends, our front-end was better under the WM condition, but worse under the MM condition than the ETSI standard front-end. This was caused by the performance of speech segmentation because automatic segmentation used in this work had more errors in the severe noise conditions than in the clean condition. However, after applying the speech algorithms described in Section 3, we achieved an average WER reduction of 51%. Significant improvement was obtained in all the conditions.

## 4. RESULTS ON THE AURORA DATABASES

In this subsection, the techniques described in Section 3 are applied to the Aurora 2 database. The database and the ASR system configuration for the Aurora 2 task were described in [2]. There were two training conditions: clean–condition training and multi–condition training. The former used only clean speech signals for training HMMs, but the latter used noisy speech signals collected various background environments for training. In other words, the multi–condition training corresponds to the WM condition in the Aurora 3 database.

Table 6 shows the WERs for the ETSI standard front-end and the proposed procedure including the robust speech recognition algorithms in Section 3. The first two columns were WERs obtained for the clean–condition training. In the clean–condition training, the proposed procedure reduced average WER by 56% compared to the ETSI standard front-end. The major improvement was obtained by applying CSM, CMS, and model compensation. Speech segmentation did not improve the performance at all. The third and fourth columns were WERs obtained for the multi–condition training. Here, we did not apply model compensation since it degraded ASR performance. This is because the data for the multi–condition training already included noise information on the parameters of HMMs. In other words, we only applied CSM, CMS, and speech segmentation. Similar to the clean–condition training, the speech segmentation algorithm had no effect on the WER for the multi–condition training. As a result, we obtained an average WER reduction of 13% mainly due to CSM and CMS.
Table 5. Performance comparison of the ETSI standard front–end and the proposed procedure on the Finnish SpeechDat–Car database.

<table>
<thead>
<tr>
<th>Condition</th>
<th>ETSI front–end</th>
<th>baseline front–end</th>
<th>proposed procedure</th>
</tr>
</thead>
<tbody>
<tr>
<td>WM</td>
<td>7.26</td>
<td>5.90</td>
<td>3.81</td>
</tr>
<tr>
<td>MM</td>
<td>19.49</td>
<td>24.56</td>
<td>11.42</td>
</tr>
<tr>
<td>HM</td>
<td>59.47</td>
<td>53.85</td>
<td>17.49</td>
</tr>
<tr>
<td>Avg</td>
<td>24.39</td>
<td>24.42</td>
<td>9.89</td>
</tr>
</tbody>
</table>

WER Reduction (%)<br>17.40 | 11.42 | 41.94 | 17.49 | 13.63 | 51.15 | 3.81 | 9.89 | - | 10.88 | 56.77 | 41.26 | 53.85 |

Table 6. Performance comparison of the ETSI standard front–end and the proposed procedure for the clean–condition training and the multi–condition training defined in the Aurora 2 database.

<table>
<thead>
<tr>
<th>Condition</th>
<th>Clean Training</th>
<th>Multi Training</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>ETSI front–end</td>
<td>proposed procedure</td>
</tr>
<tr>
<td>Set A</td>
<td>41.26</td>
<td>18.74</td>
</tr>
<tr>
<td>Set B</td>
<td>46.60</td>
<td>17.40</td>
</tr>
<tr>
<td>Set C</td>
<td>34.00</td>
<td>16.93</td>
</tr>
<tr>
<td>Avg.</td>
<td>41.94</td>
<td>17.84</td>
</tr>
</tbody>
</table>

WER Reduction (%)<br>- | 56.77 | - | 13.63 |

a possible configuration of the speech algorithms. The client has a front–end algorithm with frame–based feature compensation, pitch extraction for speech segmentation, and noise model estimation for model combination. The transmission parameters for each analysis frame are 13 MFCCs and a pitch interval [7][11]. Also, mean and covariance matrices for the estimated noise model are transmitted to the server, but need be updated only infrequently. The transmission rate for the noise model parameters may vary depending on the application. Speech segmentation, channel compensation, and model combination are performed on the server and are dependent on the parameters transmitted over the channel from the client.

For the transmission of features, we can compress 13 MFCCs at a rate of 44 bit/frame [3] and a pitch value can be represented by 4–5 bits [11][12]. Since the frame rate is 100 Hz, the transmission rate for MFCCs and pitch is 4.9 kbit/s. In addition, we have to send the noise model parameters to the server. The noise model is assumed to be a single mixture Gaussian with a diagonal covariance matrix, where the dimension of Gaussian distribution is 39 including static, delta, and delta-delta. Then, we need to compress 39-dimensional mean vector and covariance matrix. If the same dimensional quantizer used for quantizing MFCC is available, the noise model parameters can be compressed with 264 bits. If we assume that background noise is stationary over a 500 ms interval, it is sufficient to update the noise model parameters and transmit the updated parameters at that rate instead of sending them once every frame. In other words, we need additional bit rate of 0.528 kbit/s for the noise model parameters. As a result, a possible transmission rate for the configuration of Fig. 3 is 5.428 kbit/s for all parameters including MFCCs, pitch, and noise model parameters.

6. CONCLUSION

We have evaluated the performance of several speech algorithms on a subset of the Aurora 3 database. The algorithms considered were acoustic feature compensation, feature normalization, model compensation, and speech segmentation. A comparison was made between the performance of our algorithms applied to un–endpointed utterances and the ETSI standard front–end applied to utterances that were endpointed using the supervised procedure referred to in Section 4.1. The combination of all the algorithms reduced the average word error rate (WER) by 51% compared to the ETSI standard front–end. Experiments on the Aurora 2 database showed that these algorithms provided WER reduction of 56% and 13% for clean–condition training and multi–condition training, respectively. Finally, we addressed how these speech algorithms could be configured in the DSR framework by considering complexity and latency of each algorithm, and gave a typical bit rate for transmitting parameters from the client to the server.

7. REFERENCES