RAPID ON-LINE ENVIRONMENT COMPENSATION FOR SERVER-BASED SPEECH RECOGNITION IN NOISY MOBILE ENVIRONMENTS

Etienne Marcheret, Juan M. Huerta, Sreeram Balakrishnan

IBM T.J. Watson Research Center
1101 Kitchawan Road, Yorktown Heights NY, 10598

ABSTRACT

We present a rapid compensation technique aimed at reducing the detrimental effect of environmental noise and channel on server based mobile speech recognition. It solves two key problems for such systems: firstly how to accurately separate non-speech events (or background noise) from noise introduced by network artifacts; secondly how to reduce the latency created by the extra computation required for a codebook-based linear channel compensation technique. We address the first problem by modifying an existing energy based endpoint-detection algorithm to provide segment-type information to the compensation module. We tackle the latency issue with a codebook based scheme by employing a tree structured vector quantization technique with dynamic thresholds to avoid the computation of all codewords. Our technique is evaluated using a speech-in-car database at 3 different speeds. Our results show that our method leads to a 8.7% reduction in error rate and 35% reduction in computational cost.

1. INTRODUCTION

Speech recognition in mobile environments can be server or client based, depending on where the recognition takes place. In this paper we address the problem of robustness in a voice-activated server-based configuration. In such a case, the user’s speech is transmitted to the recognition server through a wireless channel which is typically associated with a network voice activity detection mechanism in order to minimize the usage of the channel.

While many robustness techniques have been proposed in the past, in order to implement any of these techniques in a server based system two issues have to be addressed:

- Special care has to be taken to account for network artifacts, such as network voice-activity detectors, as many noise compensation techniques use the hypothesized non-speech part of the signal to get an initial noise estimate. The voice-activity detector can introduce spurious artificial noise samples into this estimate, resulting in poor overall performance.

- The overall average system response should not be substantially degraded by the compensation routine.

We propose the integration of a compensation technique based on the well studied assumption of a linear channel and additive noise into a voice activated server based system for in car speech recognition. We modify an existing computationally efficient energy based endpoint detector that can separate out artifacts introduced by network voice activity detection (e.g. comfort noise) from the actual environmental background noise. Our endpoint combines a novel forward and backward pass through the signal to increase its segmentation accuracy, and its results are designed for direct utilization by the compensation module. While endpoint detection can feasibly exist on either side of the wireless link (i.e., in the client or the server) without loss of generality, in our experimental system we have located our endpointer on the server side for three reasons: first, it allows the client to be application independent (i.e., we are not assuming that our speech recognition system is the only application that this client can connect to), second, it allows us to design an endpointer that provides the most useful information to our specific compensation algorithm (i.e., the thresholds for noise and speech can be tailored to best suit our compensation routine), and finally, it allows for the recognizer’s endpoint detection and feature compensation to be designed independently of the network’s voice activity detection algorithm.

We first show that a baseline compensation system (using a basic codebook-based algorithm with linear channel and additive noise assumptions) can improve the accuracy of the recognizer. Then, based on observations regarding the distribution of the codeword posterior probabilities we propose a hierarchical labeling approach with dynamic thresholds, to substantially reduce the computational latency introduced by the compensation technique. This reduced-computation system compares very favorably with the original full labeling technique.

This paper is organized as follows: in Section 2 we describe our application’s acoustic environment, in Section 3 we describe our basic on-line compensation algorithm, in Section 4 we describe our proposal for a reduced computation on-line environment compensation method, and in Sections 5 and 6 we describe our experimental results and conclusions.

2. SERVER-BASED SPEECH RECOGNITION IN A NOISY MOBILE ENVIRONMENT

We address the problem of server based recognition of in-car speech, through a fixed-position far field microphone via a cellular telephony connection. In our application’s scenario, after the user establishes a connection with the server, the user interacts with a command-and-control type of system. Such systems typically have relatively small vocabulary sizes but are expected to have high accuracy and fast response time.
To get a sense of the relative magnitude of the noise in our target environment, we computed the frame SNR curves for each of the 3 speeds for which evaluation data was collected: 0 miles per hour, 30 miles per hour, and 60 miles per hour. We observed that the speed of the car directly affects the average SNR of the signal, with mean of 25, 23 and 18 dBs respectively, and with respective variance 32, 32 and 35. We collected this data through five car models, each having slightly different microphone setups.

In this kind of environment, the road noise, the engine noise and the cabin noise get combined with the speech signal and with their reverberations. The microphone and the channel further affect the signal as it gets transmitted to the server. In our experiments the microphone is voice activated. The system should be able to distinguish between regions of microphone inactivity and microphone activity under speech inactivity. This estimation is more critical if the silence regions are used to estimate noise vectors. A compensation routine integrated with the endpointer makes it possible to ameliorate the effect of a voice activated channel. We describe our approach to integrating the compensation algorithm with the endpointer in section 3.2.

3. ON-LINE NOISE AND CHANNEL COMPENSATION

In order to reduce the effect of the noise and environment on the system’s performance we implemented a simple and efficient noise and channel compensation algorithm. In this section we describe this method as well as its on-line implementation and integration with the endpointer detector.

3.1. A Simple model of the Environment

We start by assuming that the signal is affected by an additive noise whose spectral characteristics change slowly compared to the speech signal. We also assume the presence of a linear channel whose impulse response also changes very slowly w.r.t. speech signal. For a short region of speech $R$ (e.g., a few seconds),

$$z_R[m] = x_R[m] * h_R[m] + n_R[m] \tag{1}$$

where the noise $n$ and channel $h$ are stationary for that region. Given an observed segment of signal $z_R[m]$, our technique estimates the set of cepstral features (frames) corresponding to the clean speech signal: $X_K$.

Several approaches have been proposed to find a solution to the above estimation problem (e.g., VTS, CDCN [1], PMC). For simplicity we employ the VTS-0/CDCN solution, in which the ML estimate for the channel cepstral characteristic $h$ is:

$$\hat{h}_j = \frac{\sum_{i=0}^{N-1} \sum_{k=0}^{K-1} f_i[k](x_i - r_{j,k} - c_k)}{\sum_{i=0}^{N-1} \sum_{k=0}^{K-1} f_i[k]} \tag{2}$$

The above computation is performed in terms of a codebook of size $K$ (Gaussian Mixture Model) which models the clean speech. Because the computation time is proportional to the size of the codebook, reducing the number of codewords employed reduces the computational delay of the algorithm. It is well known, however, that larger codebooks perform better: it is necessary, then, to devise a judicious method to evaluate only the necessary codewords in a codebook (e.g., ACDCN [2]). $n$ can be either estimated on each region or it can be updated based on a forgetting factor $\alpha$:

$$\bar{n}_A = \alpha\bar{n}_{A-1} + (1.0 - \alpha) \sum_{i \in X_A} \frac{z_i}{|X_A|} \tag{3}$$

When $\alpha = 0.0$, $n$ is estimated based only on the current utterance segment. Depending on the proximity of the audio segments, if the noise changes slowly, then $\alpha$ could be made closer to 1.0. In this paper, $\alpha$ is set to 0.

After $\bar{h}$ and $\bar{n}$ have been obtained (we describe below how we estimate $\bar{n}$), the MMSE of each speech frame $\bar{x}_i$, is:

$$\bar{x}_i = z_i - \bar{h} - \sum_{k=0}^{K-1} f_i[k] r_{i,k}^j \tag{4}$$

Where $r_{i,k}^j$ is the VTS-0/CDCN correction for the $i^{th}$ frame and $j^{th}$ iteration, and $f_i[k]$ is the posterior for the corresponding $K^{th}$ codeword and $j^{th}$ frame. The above computation is also performed in terms of the GMM codebook.

3.2. Integration with an End-Point Detection and on-line considerations

We employ a fast energy-based endpoint detector which is normally applied in causal (forward) mode [3]. Due to details of its operation (dynamic energy thresholds), when the endpointer is employed in a voice activated channel, it is typically unable to identify regions of absolute channel inactivity (mic-off) and of no speech activity (mic-on) reliably. We explain how we solved this problem with our endpointer-integrated compensation algorithm (BEICA) described below.

Figure 1 shows an example of a signal under such a voice activated channel. Close analysis of the signal reveals that the nonspeech region at the end corresponds to a mic-on situation, while the nonspeech region at the beginning is a mic-off region. In panel (b), in the middle, the continuous line represents a forward endpoint pass while the dotted line represents a backward pass. When the line is high, the signal is deemed non-speech, and when the line is low the region is deemed speech by the endpointer. We can see that there are regions of agreement and disagreement in these forward and backward classifications. To avoid estimating $h$ and $n$ on mic-off regions, we utilize these forward and backward agreement and disagreement classifications.

3.2.1. Basic Endpoint-Integrated Compensation Alg.: BEICA

We now describe BEICA: the basic endpoint integrated compensation algorithm based on equations (2) and (4), and [3]:

**Step 1.** Given a starting point, identify in a forward pass the region of forward speech activity

**Step 2.** Where the forward speech activity detector turns off, find the backward speech activity region through a backward pass

**Step 3.** Estimate $n$ on the regions of forward and backward speech disagreement. Estimate $\bar{h}$ on the regions of forward-backward speech agreement

**Step 4.** Using $h$ and $n$ find the clean signal’s cepstra estimate for the region of speech agreement

**Step 5.** Move the starting point to where the forward endpointer
turned off and goto Step 1.

We analyzed the different variations of BEICA. The variants consist essentially of different ways in which \( n \) and \( h \) are estimated and which signal regions are compensated. Method 4 corresponds to the BEICA algorithm outlined above.

**Method 1:** Estimate \( n \) using forward-backward regions. Estimate \( h \) on forward-backward agreement. Correct all incoming data.

**Method 2:** Estimate \( n \) using only forward regions. Estimate \( h \) and correct on regions of forward-backward agreement.

**Method 3:** Estimate \( n \) using only forward regions. Esti\( m \)he \( h \) on all data. Correct all incoming data.

**Method 4:** (Original BEICA) Estimate \( n \) using forward-backward regions. Estimate \( h \) and correct on regions of forward-backward agreement.

As we will see in the Results section, in order to get the compensation algorithm to work, it is crucial to get the BEICA configuration right.

### 4. COMPUTATIONAL CONSIDERATIONS: HIERARCHICAL LABELING-BASED COMPENSATION

The size of the codebook used in the compensation algorithm directly affects its running time as well as its accuracy. We augmented our compensation algorithm to include a modified hierarchical labeler (tree based vector quantization based on [4]) to efficiently compute the Gaussian likelihoods needed in the estimation and maximization steps of the compensation routine.

The hierarchical labeler described in [4] consists of a tree of depth \( N \) in which each node is associated to a single Gaussian. The internal node’s Gaussian represents a cluster of its children. Each leaf corresponds to a Gaussian codeword in the global GMM Codebook. For details on how the tree is traversed in an ASR system see [4]. In that case, the goal is to reduce the number of states whose likelihoods are to be evaluated. In our case we want to identify a relevant subset of the codebook at each frame. We will utilize a tree with 3 levels: \( V1 \) and \( V2 \) are internal levels, and \( V3 \) is the leaf level which map to our \( K \) size codebook. The list of Gaussians start with the \( V1 \) nodes; the likelihoods of these nodes are evaluated and sorted. Nodes are added to the list of codewords until the likelihood of the new nodes are above a cutoff. The value of the likelihood cutoff is established by using the higher of two bounds

1. The Log-Likelihood of the highest scoring \( V1 \) node, minus a \( \Delta \) parameter
2. The Log-Likelihood of the score cutoff which would permit at most \( \Lambda_L \) Gaussians in the level below to be computed

Essentially cutoff 2 is not used unless the scores of the nodes in level \( L \) of the tree are so flat that more than \( \Lambda_L \) Gaussians will be evaluated in the next level.

#### 4.1. Hierarchical labeling with soft-max thresholds

We found that using a soft-max threshold instead of the cutoffs (1) and (2) above, allowed us to achieve a better accuracy and a lower average number of Gaussians computed per frame.

We define the soft-max score \( s_{i,j} \) for node \( j \) in level \( i \) of the tree as

\[
s_{i,j} = \frac{\exp(\beta L_{i,j})}{\sum_j \exp(\beta L_{i,j})}
\]

Where \( L_{i,j} \) is the log likelihood of the Gaussian associated with node \( j \) in level \( i \) and \( \beta \) is a scaling constant chosen by experiment. We define for each level in the hierarchy a soft-max cutoff \( \text{min}_j \text{softmax}_j \) and only expand those nodes for which \( s_{i,j} > \text{min}_j \text{softmax}_j \).

#### 4.2. Measuring the frame posterior instantaneous entropy

A justification for the better performance of the soft-max threshold can be found by examining the distribution of the codebook posterior probabilities \( f_i[k] \) at different speeds.

Equations (2) and (4) are computed in terms of the posterior probabilities \( f_i[k] \) using the entire GMM codebook. Figure 4 shows the histograms of the instantaneous entropy of the codebook posterior distributions for frames of a subset of the evaluation corpus at each speed. The instantaneous entropy is intended to measure the flatness of the distribution of the posteriors at each frame: the flatter this distribution is for a particular frame, the higher the instantaneous entropy. The instantaneous entropy for frame \( x_t \) is defined in terms of the codeword posterior probabilities \( p(k|z_t) \) as:

\[
d(x_t) = \sum_k p(k|z_t) \log(p(k|z_t))
\]

The normalized histograms of the instantaneous entropy in figure 3 show that the higher the car speed (noisier) the higher the location of the peak. Similarly, the lower the noise level is, the lower the peak. This implies that the cumulative weight in the posterior curve (and thus, Eqs. 4 and 2) is dominated by a few codewords in quiet conditions. The soft-max threshold will naturally take this into account since it is based on Gaussian scores that will behave in a similar way to the codebook scores. Hence if only a few code-words dominate the posterior mass then only a few nodes in the Hierarchical tree are going to have a soft-max score that is above the threshold.
5. EXPERIMENTAL RESULTS

Our recognition system consists of HMM models with 156K Gaussians and just over 1000 states trained on general telephony data. Our testing corpus consists of data recorded over 5 car models, coming from 97 recording sessions, each reading 80 utterances. These recording session occurred at 0, 30 or 60 miles per hour. We used a 39 dimensional LDA based feature generated from 13 dimensional cepstra produced at 100 frames per second. Table 1, shows in the first column the baseline experimental results, consisting of the recognition of the uncompensated observed signal using the HMM models described above. The baseline average error rate for the three speeds is 13.44%.

**Table 1.** Full labeling compensation word error rates. Effect of different end-pointer integration methods.

<table>
<thead>
<tr>
<th>Speed</th>
<th>Bsln</th>
<th>FL Gpf</th>
<th>HL Gpf</th>
<th>%CS</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>9.71</td>
<td>9.39</td>
<td>9.17</td>
<td>151</td>
</tr>
<tr>
<td>30</td>
<td>12.27</td>
<td>11.95</td>
<td>11.97</td>
<td>167</td>
</tr>
<tr>
<td>60</td>
<td>18.33</td>
<td>16.95</td>
<td>16.31</td>
<td>176</td>
</tr>
<tr>
<td>Avg</td>
<td>13.44</td>
<td>12.76</td>
<td>12.28</td>
<td>165</td>
</tr>
</tbody>
</table>

**Table 2.** Hierarchical-Labeling compensation results, average number of Gaussians per frame (Gpf) evaluated, and percentual computational savings

<table>
<thead>
<tr>
<th>Speed</th>
<th>Bsln</th>
<th>FL Gpf</th>
<th>HL Gpf</th>
<th>%CS</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>256</td>
<td>151</td>
<td>41</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>256</td>
<td>167</td>
<td>35</td>
<td></td>
</tr>
<tr>
<td>60</td>
<td>256</td>
<td>176</td>
<td>30</td>
<td></td>
</tr>
<tr>
<td>Avg</td>
<td>256</td>
<td>165</td>
<td>35</td>
<td></td>
</tr>
</tbody>
</table>

SNR is 41, 35 and 31% respectively.

6. CONCLUSIONS

We described a compensation technique integrated with an on-line endpoint detector for rapid robust ASR in mobile environments. In order for our basic compensation technique to work, it needs to be carefully integrated with the endpoint detector. When this is done, we reduced the WER from 13.44% to 12.76% using full labeling compensation. We observed that using hierarchical labeling not only reduces the average number of codewords evaluated per frame, but it also further reduces the average WER to 12.28%. This is an encouraging result; future directions will include further exploitation of hierarchical labeling in compensation routines.

7. REFERENCES