Robust ASR front-end using spectral-based and discriminant features: experiments on the Aurora tasks

Carmen Benítez[a,1], *Lukas Burget*[b], *Barry Chen*[a], *Stéphane Dupont*[a], *Hari Garudadri*[c], *Hynek Hermansky*[a,b], *Pratibha Jain*[b], *Sachin Kajarekar*[b], *Nelson Morgan*[a], *Sunil Sivadas*[b]

[a]International Computer Science Institute, Berkeley, California, USA,
[b]Oregon Graduate Institute of Science and Technology, Portland, USA,
[c]Qualcomm Inc., San Diego, CA, USA.

carmen,byc,dupont,morgan@icsi.berkeley.edu hgarudad@qualcomm.com
hynek,pratibha,sachin,sunil,lukas@ece.ogi.edu

Abstract

This paper describes an automatic speech recognition front-end that combines low-level robust ASR feature extraction techniques, and higher-level linear and non-linear feature transformations. The low-level algorithms use data-derived filters, mean and variance normalization of the feature vectors, and dropping of noise frames. The feature vectors are then linearly transformed using Principal Components Analysis (PCA). An Artificial Neural Network (ANN) is also used to compute features that are useful for classification of speech sounds. It is trained for phoneme probability estimation on a large corpus of noisy speech. These transformations lead to two feature streams whose vectors are concatenated and then used for speech recognition.

This method was tested on the set of speech corpora used for the "Aurora" evaluation. Using the feature stream generated without the ANN yields an overall 41% reduction of the error rate over Mel-Frequency Cepstral Coefficients (MFCC) reference features. Adding the ANN stream further reduces the error rate yielding a 46% reduction over the reference features.

1. Introduction

Voice-enabled services are being deployed. These services could be accessed through many kind of devices, including mobile devices such as cellular phones. However, speech transmitted on mobile channels can be significantly degraded. The concept of Distributed Speech Recognition (DSR) was proposed to address this issue by moving some ASR processing, like feature extraction, onto the terminal side where the speech signal is free of transmission channel degradation. These features can then be transmitted (through the data channel of the mobile communication system) to a remote ASR server.

Sensing the need to ensure compatibility between the terminal and server side ASR processing, a standartization initiative called "Aurora" was initiated within European Telecommunications Standards Institute (ETSI). Several databases were prepared and an ASR back-end based on HMM-based HTK toolkit was defined for evaluation purposes [5]. This year's "Aurora" task consists of digit string recognition on six languages and a range of noise conditions. the challenge involved is to design a front-end that gives a significant reduction in the WER compared to a conventional front-end based on Mel-cepstrum (MFCC), within the limited computational resources and restricted delay.

In this paper, we describe the front-end submitted by OGI-ICSI-QUALCOMM consortium . The proposed feature extraction scheme is based on temporal and spectral processing and uses a number of techniques, most of them being described earlier in other papers, where they were independently shown to yield improved noise robustness. First, robust MFCC-based features are computed using data-derived temporal filters [7], on-line mean and variance normalization [3] and voice activity detection. Previous work has also shown that Muli-Layer Perceptron Neural Net (MLP) can be used effectively as feature post-processor to improve robustness in speech recognition [1, 6]. We adopt this tecnhnique and use Multi-Layered Perceptrons. To limit the computations at the terminal-side, the MLP is preferably applied at the server side. The next sections will give more details on the processing using this preferred split between terminal-side and server-side processing.

2. Terminal Feature Extraction

The speech samples (8 kHz sampling rate) are windowed into overlapping 25 ms frames with a frame shift of 10 ms. A 256 point FFT is used to compute the power spectrum that is used in an emulated filter-bank composed of 23 triangular weighting functions on a mel scale. The natural logarithm is then applied to the 23 filter-bank energies.

At this point, two processes are performed. First, a Voice Activity Detector (VAD) is used to detect non-speech regions. The feature vectors are passed to an ANN trained to discriminate between speech and non-speech. The second process consists of filtering the time trajectories of the log energies for each of the 23 channels using RASTA-like filters that are optimized using Linear Discriminant Analysis (LDA) [7]. Due to the band-pass nature of these LDA-RASTA filters, the filtered signal is then downsampled by a factor 2 in time, without degradation in performance [2].

Non-speech frames are then dropped using the decisions provided by the VAD detector. Fifteen cepstral coefficients are calculated using a Discrete Cosine Transform (DCT) on these filtered log-energies, and finally, a mean and variance on-line normalization algorithm is applied to partially compensate for channel and noise distortions [3].

15 acoustic parameters are generated in this way at the terminal side for each frame. A compression algorithm is applied

[0] C. Beńitez is also with Granada University, Spain

to reduce the bit-rate before transmission over the data channel. Note that as we use the downsampled stream, the frame shift is now 20 ms. This provides empty slots in the terminal-to-server transmission increasing channel capacity. The empty slots can be used by other applications or directly translate into higher channel capacity.

This terminal side processes are described in some more detail in the following sections.

2.1. Voice Activity Detector (VAD)

The VAD uses an MLP trained to discriminate between speech and non-speech. The MLP is trained on multiple databases containing clean and noisy speech. This MLP has 6 input units, 15 hidden units and 1 output. Three analysis frames are used as input - the current frame and 2 adjacent frames. This architecture is a compromise between VAD complexity and accuracy. The output of the MLP is an estimation of the non-speech posterior probability. This probability is thresholded to zero or one, and then filtered using an 11 point median filter to remove spurious spikes. The binary decision that is obtained for each frame is later used to drop the frames that are more likely to be non-speech.

2.2. LDA-based temporal RASTA filtering

Spectral features are sensitive to environment mismatches such as background noise and channel distortion. This results in rapid degradation in performance of speech recognizers in adverse conditions. We address this problem by band-pass filtering the time trajectories of log mel-frequency energies. The filters are derived using the linear discriminant analysis (LDA) [7]. Restaurant noise from Aurora database was added to OGI-Stories at 10 dB SNR to simulate the environment mismatch. The filters were derived foreach mel-frequency band using 101 dimensional feature vectors. Each feature vector represented the current frame, 50 frames in the past and 50 frames in the future.

To meet latency requirements defined by ETSI, we approximated the filter responses by 41 tap symmetric FIR filters. The filter derived on the second band was used to filter the first two bands and the filter derived on the fourth band was used to filter the remaining bands. Figure 1 shows the impulse and frequency response of these two filters. The filter with 6 Hz cutoff is applied to Mel channels 1 and 2, and the filter with 16 Hz cutoff is applied to channels 3 to 23.

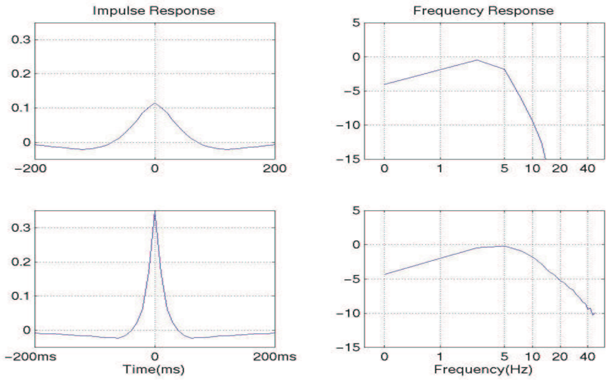


Figure 1: LDA filters impulse response and frequency response.

2.3. On-line Normalization (OLN)

Additive noise shifts the mean and reduces the variance of cepstral coefficients [3]. Therefore, online mean and variance normalization is applied. The estimates of mean and variance are initialized using the mean and variance of the first four frames of all utterances in our training set. These estimates are updated for each frame as follows.

$$m_t = m_{t-1} + \alpha(x_t - m_{t-1}), \tag{1}$$

$$\sigma_t^2 = \sigma_{t-1}^2 + \alpha((x_t - m_t)^2 - \sigma_{t-1}^2) \tag{2}$$

$$x_t' = \frac{x_t - m_t}{\sigma_t + \theta} \tag{3}$$

where x_t is the cepstral coefficient and x_t' is the normalized cepstral coefficient at time t. m_t and σ_t^2 are the estimated mean and the variance of x_t. α is an adaptation constant less than 1 to guarantee positive estimate of the variance, and θ is an empirically determined variance floor. We use $\alpha = 0.1$ and $\theta = 1.0$.

3. Server Feature Vector Generation

In [6], several feature sets were evaluated on the earlier version of the “Aurora” task. It was shown that after a MLP-based non-linear discriminative transformation, various features which were so far not well applicable, can be effectively used in GMM-based recognition systems.

In the previous evaluation, only a single database was used and both the MLP as well as the HTK back-end were trained on the same data. In the current evaluation, the task of training the MLP is harder because of the additional databases which vary in language and noise conditions. Further complicating our attempts to apply this tandem approach, the MLP cannot be trained on the target task data because the front end has to be independent of the target language, task, and noise condition.

We evaluated the tandem structure with MLPs trained on various single-language as well as multi-lingual databases. Training the MLP on the target language and task was shown to yield the typical 40% error rate reduction but training on other languages or on multi-lingual databases resulted in negligible improvements in accuracy.

However, we observed that the MLP outputs could be effectively used in conjunction with the non-transformed features. Figure 2 gives a block diagram of this structure. First, the derivatives and accelerations coefficients are computed. Then, we concatenate both non-linearly transformed features (outputs of the MLP followed by PCA, labeled 3 and 4 in Figure 2) as well as linearly transformed features (outputs from PCA, labeled 2 in Figure 2). This can significantly improve the recognition performance, at the cost of additional processing, storage, and algorithmic delay.

The speech corpus that we use to train the MLP and to compute the PCA transformation matrices consist of a portion of the Timit database downsampled to 8 kHz and artificially corrupted by various types of noises at different signal to noise ratios. In the following subsections we briefly describe the block diagram of the server side feature vector generation process.

3.1. Contextual Principal Components Analysis

Contextual PCA allows us to add contextual information without increasing the feature vector size by keeping only those coefficients that explain most of the variance. Three consecutive frames of the features are stacked together to form a 135-

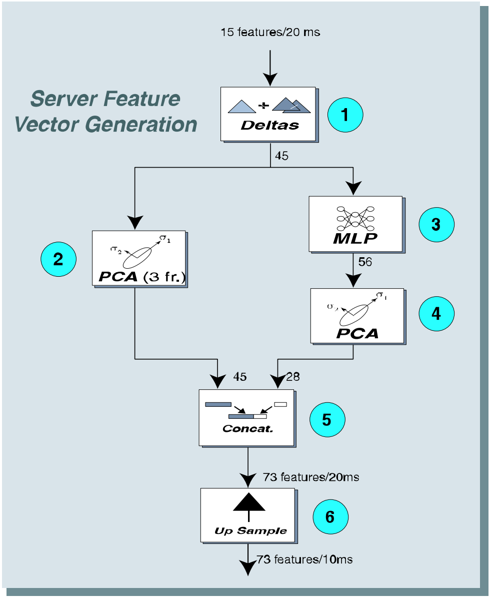


Figure 2: Block diagram of the Server Feature Vector Generation. The size of the feature vectors is given along the arrows.

dimensional vector which is projected using the 45 directions corresponding to the largest eigenvalues.

3.2. Non-Linear Transformation Using an Artificial Neural Network

The MLP provides a nonlinear mapping between feature space and phoneme posterior probability estimates. These probability estimates typically have a highly non-Gaussian distribution. However, the MLP outputs before the final nonlinearities are quite gaussian and more suitable for Gaussian mixture model in current HMM. These pre-nonlinearity outputs are then used for the subsequent PCA which is applied to decorrelate the feature space and make it more compatible with the diagonal covariance model of the HTK back-end. Further, the PCA also reduces the dimensionality of the final feature space.

Five consecutive feature frames (computed using the terminal-side front-end algorithms described earlier) are stacked together to yield a 225 dimensional input vector for the MLP. The MLP has one hidden layer consisting of 500 units with sigmoid activation functions, while the output layer consists of 56 output units with softmax activation functions. The MLP is trained on phonetic targets (56 context independent English phonemes) from the Timit database with added noise. The PCA bases are pre-computed on the same data that is used for training the MLP. As can be seen in Figure 2, The outputs after the MLP and PCA are 28 features.

3.3. Concatenation and up-sampling

Combining information from different feature streams often improves performance of recognition systems. The algorithms described in Section 2 yield a stream containing spectral information. The algorithm described in Section 3 amount to creating a stream containing phonetic discrimination information. The vectors from these two streams are concatenated to yield a 73-dimensional feature vector.

The final step before the back-end is to upsample the feature stream by two. Linear interpolation between successive frames is used to go back to the 10 ms original frame shift.

4. Experimental Setup and Results

Experiments were performed on the entire set of digit strings tasks defined by ETSI. They cover six different languages: English, Italian, Finnish, Spanish, German, and Danish.

The English task is based on a modified version of the TI-Digits database. Different noises are added in a controlled way at different signal-to-noise ratios (from clean to 0 dB). Two training modes using the same set of utterances are defined: “clean” uses only clean digits and “multicondition” uses 4 different noises. Three test sets are defined. “Set A” has the same noises as the multicondition training, resulting in matched test condition. “Set B” uses four different noises. In this case, there is a mismatch between training and test data. In “Set C”, noisy speech is filtered to simulate channel mismatch.

The other languages are taken from the corpora that were recorded as part of the SpeechDat-Car European (SDC) project [4]. These are real recordings made in cars with a setup consisting of a close talking microphone and a hands-free microphone. Three train/test configurations were defined: the well-matched condition (WM), the medium mismatched (MM) condition and the highly mismatched condition (HM). In the WM case, 70% of the entire data is used for training and 30% for testing. The training set contains all the variability that appear in the test set. In the MM case, only far microphone data is used for both training and testing. For the HM case, training data consists of close microphone recordings only while testing is done on far microphone data.

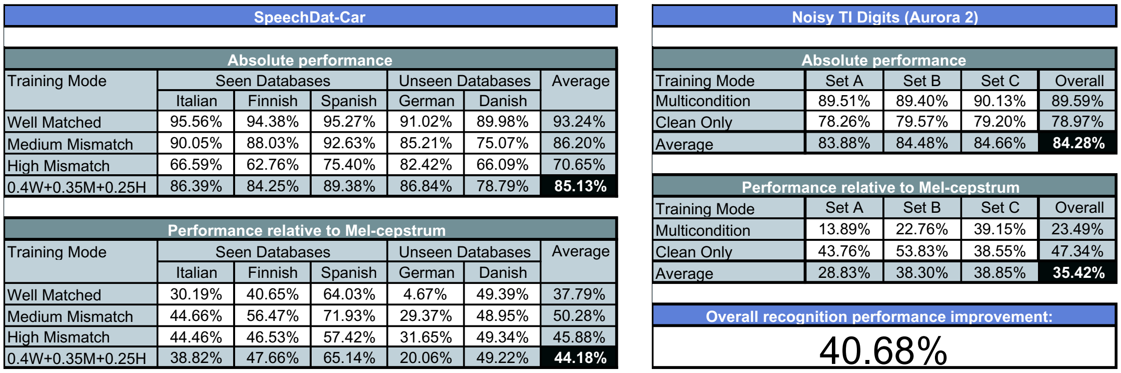
4.1. Results

The performance of the proposed front-end is evaluated by comparison to standard MFCC features. The configuration of the recognizer is fixed for all tasks. It uses the HTK toolkit (Gaussian Mixtures Model-Hidden Markov Model - GMM-HMM) recognizer where the models are words, each composed of 16 states, 3 mixture per state, and diagonal covariance matrices.

Results are presented in Table 1 for the terminal-side feature extraction algorithm (with additional derivatives and accelerations coefficients) and in Table 2 when we also use the server-side post-processing. These tables contain the absolute performance for the different databases and test conditions as well as the performance relative to the baseline MFCC.

For both systems, it can be seen that the reduction in WER is consistent within TI-Digits and SDC. The only exception is SDC-German, which is the noisiest of all SDC databases. The difference in improvement between TIDIGITS and SDC is due to multiple reasons. SDC is a realistic database and hence the baseline performance deteriorates much faster with mismatch compared to mismatch created by artificially adding noise to clean speech. The long segments of silence before the onset of speech in SDC cause a large number of insertion errors in baseline system. We observed that the VAD alone improves acuracy of the baseline on this Aurora task by more than 20%.

Let’s note finally that a coding scheme can be used before transmission of the features through the mobile data channel. A bit rate of 2400 bits/s is achievable without significant degradation in the ASR performance. The algorithmic delay of the first approach which is applied on the handset is below 250 ms, the server side processing currently adds another 80 ms to the



SpeechDat-Car

Absolute performance

Training Mode	Seen Databases			Unseen Databases		Average
	Italian	Finnish	Spanish	German	Danish	
Well Matched	95.56%	94.38%	95.27%	91.02%	89.98%	93.24%
Medium Mismatch	90.05%	88.03%	92.63%	85.21%	75.07%	86.20%
High Mismatch	66.59%	62.76%	75.40%	82.42%	66.09%	70.65%
0.4W+0.35M+0.25H	86.39%	84.25%	89.38%	86.84%	78.79%	**85.13%**

Performance relative to Mel-cepstrum

Training Mode	Seen Databases			Unseen Databases		Average
	Italian	Finnish	Spanish	German	Danish	
Well Matched	30.19%	40.65%	64.03%	4.67%	49.39%	37.79%
Medium Mismatch	44.66%	56.47%	71.93%	29.37%	48.95%	50.28%
High Mismatch	44.46%	46.53%	57.42%	31.65%	49.34%	45.88%
0.4W+0.35M+0.25H	38.82%	47.66%	65.14%	20.06%	49.22%	**44.18%**

Noisy TI Digits (Aurora 2)

Absolute performance

Training Mode	Set A	Set B	Set C	Overall
Multicondition	89.51%	89.40%	90.13%	89.59%
Clean Only	78.26%	79.57%	79.20%	78.97%
Average	83.88%	84.48%	84.66%	**84.28%**

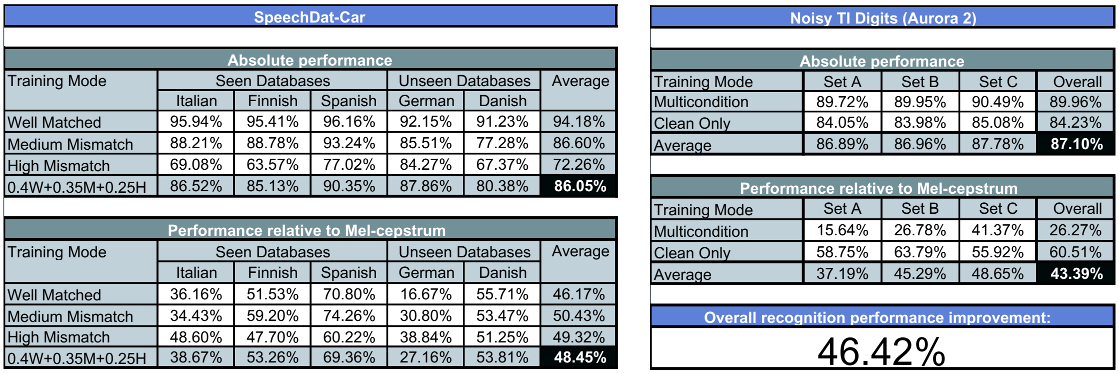
Performance relative to Mel-cepstrum

Training Mode	Set A	Set B	Set C	Overall
Multicondition	13.89%	22.76%	39.15%	23.49%
Clean Only	43.76%	53.83%	38.55%	47.34%
Average	28.83%	38.30%	38.85%	**35.42%**

Overall recognition performance improvement:

40.68%

Table 1: Terminal-side feature extraction. The absolute performance is a recognition accuracy (1 - recognition rate). The performance relative to Mel-cepstrum is the relative error rate reduction over the MFCC baseline. Weighted averages according to the ETSI-defined protocol are also provided. An overall performance improvement for the 6 different languages is also given: 40,68% reduction of the error rate over the MFCC baseline. The definition of the different test conditions is given in the text of the paper.



SpeechDat-Car

Absolute performance

Training Mode	Seen Databases			Unseen Databases		Average
	Italian	Finnish	Spanish	German	Danish	
Well Matched	95.94%	95.41%	96.16%	92.15%	91.23%	94.18%
Medium Mismatch	88.21%	88.78%	93.24%	85.51%	77.28%	86.60%
High Mismatch	69.08%	63.57%	77.02%	84.27%	67.37%	72.26%
0.4W+0.35M+0.25H	86.52%	85.13%	90.35%	87.86%	80.38%	**86.05%**

Performance relative to Mel-cepstrum

Training Mode	Seen Databases			Unseen Databases		Average
	Italian	Finnish	Spanish	German	Danish	
Well Matched	36.16%	51.53%	70.80%	16.67%	55.71%	46.17%
Medium Mismatch	34.43%	59.20%	74.26%	30.80%	53.47%	50.43%
High Mismatch	48.60%	47.70%	60.22%	38.84%	51.25%	49.32%
0.4W+0.35M+0.25H	38.67%	53.26%	69.36%	27.16%	53.81%	**48.45%**

Noisy TI Digits (Aurora 2)

Absolute performance

Training Mode	Set A	Set B	Set C	Overall
Multicondition	89.72%	89.95%	90.49%	89.96%
Clean Only	84.05%	83.98%	85.08%	84.23%
Average	86.89%	86.96%	87.78%	**87.10%**

Performance relative to Mel-cepstrum

Training Mode	Set A	Set B	Set C	Overall
Multicondition	15.64%	26.78%	41.37%	26.27%
Clean Only	58.75%	63.79%	55.92%	60.51%
Average	37.19%	45.29%	48.65%	**43.39%**

Overall recognition performance improvement:

46.42%

Table 2: Terminal-side feature extraction + server-side post-processing.

overall algorithmic latency.

5. Conclusions

In this paper, we presented a robust front-end that combines several approaches previously described in the literature. Spectral-based features are first computed. Increased robustness is obtained using data-derived temporal filters, on-line normalization of the features and voice activity detection. Moreover, an artificial neural network is used in a similar fashion than non-linear discriminant analysis to obtain features that are relevant to phoneme classification.

This method was tested on the set of speech corpora used for the “Aurora” evaluation. Using the feature stream generated without the ANN yields an overall 41% reduction of the error rate over Mel-Frequency Cepstral Coefficients (MFCC). The complexity of the handset side processing is not much higher than the complexity of the conventional MFCC. Adding the ANN stream further reduces the error rate, yielding a 46% reduction over the reference features at the expense of additional complexity and a slightly longer algorithmic latency.

Our approach could be used in tandem with conventional noise supression techniques such as spectral subtraction and Wiener filtering to further improve the performance.

6. Acknowledgments

This research was supported by DoD under MDA904-98-1-0521, by NSF under IRI-9712579, and by Qualcomm Inc.

7. References

[1] V. Fontaine, C. Ris, and J.M. Boite. Nonlinear discriminant analysis for improved speech recognition. In *EUROSPEECH*, volume 4, pages 2071–2974, Rhodes, 1997.

[2] Hynek Hermansky and Pratiba Jain. Down-sampling speech representation in ASR. In *EUROSPEECH*, Budapest, 1999.

[3] P. Jain and H. Hermansky. Improved mean and variance normalization for robust speech recognition. In *ICASSP*, Salt Lake City, 2001.

[4] A. Moreno, B.Lindberg, C.Draxler, G.Richard, K.Choukri, and J. Allen. Speechdat-car a large speech database for automotive environments. In *LREC (Language Resources and Evaluation*, Athens, 2000.

[5] David Pearce. Enabling new speech driven services for mobile devices:an overview of the etsi standards activities for distributes speech recognition front-ends. In *AVIOS 2000: The Speech Applications Conference*, San Jose, CA, 2000.

[6] S. Sharma, D. Ellis, S. Kajarekar, P. Jain, and H. Hermansky. Feature extraction using no-linear transformation for robust speech recognition on the aurora database. In *ICASSP*, volume 2, pages 1117–1120, Istambul, 2000.

[7] S. van Vuuren and H. Hermansky. Data-driven design of rasta-like filters. In *EUROSPEECH*, volume 1, pages 409–412, Rhodes, 1997.