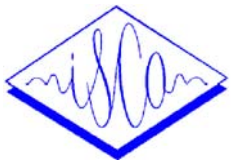
ISCA Archive
http://www.isca-speech.org/archive



INTERSPEECH 2004 - ICSLP
8th International Conference on Spoken Language Processing
ICC Jeju, Jeju Island, Korea
October 4-8, 2004

Memory and Computation Reduction for Embedded ASR Systems

Sangbae Jeong[1], *Icksang Han*[2], *Eugene Jon*[3] *and Jeongsu Kim*[4]

Human Computer Interaction Lab.
Samsung Advanced Institute of Technology
{sangbae.jeong[1], hanis[2], eugene95[3], jeongsu.kim[4]}@samsung.com

Abstract

Nowadays, the performance of mobile devices such as personal digital assistants (PDA) or cell phones are rapidly improving. This improvement increased the demand for various functions in mobile devices. Among them, embedded speech recognizers are one of the major research topics. To guarantee an acceptable performance, the efficient usage of the hardware resources is very important. In this paper, we introduce our embedded speech recognizer and several important technologies implemented in it used to reduce hardware resources.

1. Introduction

The number of people using mobile devices such as cell-phones and PDA's is steadily increasing. Recently, they are used multi-purposely with applications such as Internet, E-mail, navigations, etc. However, most people find the input interface of these mobile devices inconvenient. This is because state of the art technology has made devices too small to put full size keyboards on them. To alleviate this inconvenience, many buttons and touch-screen methods have been used. However, these methods are not as convenient and natural as speech recognition. In addition to the inconvenience, the operation of mobile devices in car environments is very dangerous due to the possibility of distracting the driver. In these points of view, the control of mobile devices by speech is very useful[1].

Many researchers have been studying embedded speech recognizers for these reasons. The major problem inherent in implementing automatic speech recognition (ASR) systems to mobile devices concerns the memory and the computation speed, because the more complex the recognition tasks are, the more computation power and memory they require. Therefore, in order to guarantee similar performance to what we can obtain in PC's or workstations, we used various methods to reduce the memory and the computation load.

We can divide an ASR system roughly into front-end (noise reduction and speech detection), feature extraction, and the Viterbi search engine. Among these modules, the Viterbi search engine requires the most hardware resource. Nowadays, the concept of distributed speech recognition (DSR) systems has been proposed[2][3]. In DSR systems, noise reduction, speech detection, and feature extraction are computed in mobile devices and the Viterbi search is computed in servers. The European Telecommunication Standard Institute (ETSI) and The International Telecommunication Union (ITU) are currently defining the standard feature parameters and communication protocols, by which any mobile device can communicate with any ASR server. The DSR concept is applicable to large vocabulary recognition such as address recognition, for navigation by a global positioning system (GPS). However, small-size vocabulary recognition tasks on mobile devices are still required to make user-friendly systems.

HCI Lab. in SAIT has been working to develop embedded ASR systems for name-dialing, digit-dialing in PDA's and cell-phones manufactured by Samsung Electronics Co. Ltd. (SEC). This paper is for the introduction of the embedded ASR system of HCI Lab and mainly focuses on the reduction of hardware resources necessary for ASR.

The remainder of this paper is composed as follows. In section 2, we introduce the embedded speech recognition systems of HCI Lab, In section 3, we explain the methods adopted to reduce memory and computation loads. In section 4 we explain the experiments and the results. Finally, in section 5 we conclude this paper.

2. Overview of the embedded ASR system

The main tasks of the embedded ASR system of HCI Lab. are name-dialing and digit-dialing. In addition to speech-activated dialing functions, the system can recognize 25 commands. The commands include Internet access, confirmation of the number of the recent incoming call and today's schedule, etc. In name-dialing, it can recognize 200 registered Korean names. In digit-dialing, it performs Korean continuous digit recognition in three stages. Several post-processing algorithms are also implemented in order to reduce substitution and deletion/insertion errors, which are very frequent due to the characteristics of Korean digits. Our system was ported to a SCH-M400 PDA, equipped with an IntelXscale 400MHz CPU and 64MB of read-only memory (ROM) and random-access memory (RAM), respectively.

3. Resource reduction methods

To reduce the required memory and the computation load, we focused our research on the hidden Markov model (HMM) parameters, noise reduction methods, and likelihood computation. The details are as follows.

3.1. HMM parameters

3.1.1. State-tying

State-tying techniques are usually used when the data for the acoustic contexts are not enough for normal HMM training. In many cases, we can improve the recognition performance by using this method. However, we were able to use state-tying to reduce the RAM and the ROM size without loss of the recognition performance, because we were given

sufficient training data. The state-tying used in our study was a kind of bottom-up method. To tie acoustically similar states, we first train the semi-continuous HMMs. Then, information loss (*IL*) is calculated as follows.

$$IL = \left| E_1 + E_2 - 2E_m \right| \tag{1}$$

Where, E_1 and E_2 are each entropies of the two comparing states' probability distribution functions (PDF), and E_{m} is the entropy of the mean PDF. The mean entropy is calculated by the equally-weighted sum of the two PDF's. As shown in Eq. (1), the more similar the PDF's are, the less information loss we obtain. Figure 1 shows an illustration of the calculation of the information loss. Finally, the groups of states are found, where each pair of states has information loss less than the prefix threshold. After tied states are made in such a way, they are shared when we perform HMM-training and speech recognition.

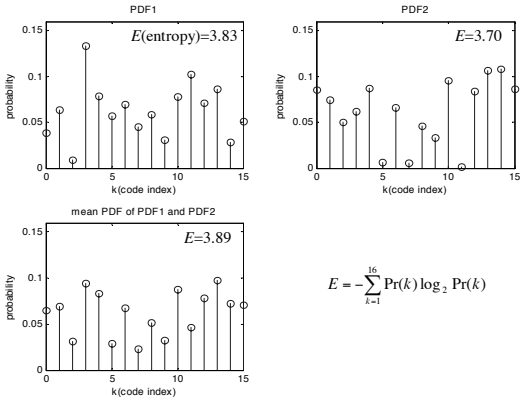


Figure 1 : Illustration of information loss computation

3.1.2. Modified global variance

It is well known that the variance vectors of each mixture in HMM states are less sensitive to the probability computation than the mean vectors. So, the variance vectors are usually replaced with a particular value, a global variance (GV) vector, when the resource of a recognition device is limited like the case of PDA's or mobile cellular phones. But, when the GV vector is used, the recognition performance degrades somewhat. In order to mitigate the degradation, we proposed a modified global variance method, in which each Gaussian PDF has a unique weighting value[4]. The weighting value is multiplied to the GV. As a result, we obtain the modified GV's, which are very similar to the original variance values. The weighting value for a specific Gaussian PDF is computed in the minimum mean-squared-error sense. The cost function to minimize is shown in Eq. (2).

$$J = \sum_{k=1}^{D} (\sigma^2(k) - \alpha\sigma_{GV}^2(k))^2 \tag{2}$$

Where, D is the dimension of the feature vector and, $\sigma^2(k)$, $\sigma_{GV}^2(k)$, α are the original variance, the global variance, and the weighting value, respectively. Because the cost function J is a second order equation with respect to α, the optimal weighting value can be obtained by taking derivative on both sides of Eq. (2) and set it equal to zero. Eq. (3) shows the result.

$$\alpha^* = \frac{\sum_{k=1}^{D} \sigma_{GV}^2(k)\sigma^2(k)}{\sum_{k=1}^{D} \sigma_{GV}^4(k)} \tag{2}$$

The optimal weighting values are computed for each feature stream, that is, the static, the delta, and the delta-delta cepstrum in our real applications. The approximation of the original variances using a global variance and its optimal weighting value is depicted in Figure 2.

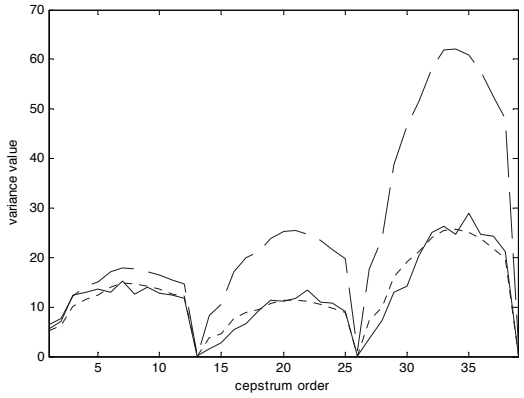


Figure 2 : Illustration of cepstal curve fitting by original variance and its optimal weighting values(solid : original variance, dotted : modified GV, dashed : global variance)

3.1.3. Scalar quantization

When we compute the likelihood of a certain feature vector, Eq. (3) is the basic computation unit, for the PDF's of HMM states are composed of many Gaussians mixtures.

$$\frac{1}{\alpha^*} \sum_{k} \frac{(c(k) - \mu(k))^2}{2\sigma_{GV}^2(k)} \tag{3}$$

Where, $c(k)$ and $\mu(k)$ are the cepstrum and the mean value. If Eq. (3) is used to compute the likelihood without any modification, the computation amount is burdensome because it requires multiple division calculations for each Gaussian mixture, which are time-consuming operations. To reduce the division operations, we devised the scalar quantization of feature vectors and mean vectors. Before we calculate Eq. (3), the global variance estimated with a large number of feature

vectors. So, if we quantize the feature vectors and the mean vectors, the number of divisions can be reduced to only once. After the scalar quantization, Eq. (4) is equivalent to Eq. (3).

$$\frac{1}{2\alpha^*}\sum_k \mathrm{DIST}\big(\mathrm{SQ}(c(k)),\mathrm{SQ}(\mu(k))\big) \qquad (4)$$

Where, $\mathrm{SQ}(\bullet)$ returns the index of the quantized value of its input, and $\mathrm{DIST}(\bullet)$ loads the squared and GV-divided value of the difference of the two quantized values indexed by the SQ functions. Actually, scalar quantization replaces the multiplication and division in Eq. (3) with index-searching and memory loading, which are much simpler tasks. Because cepstral coefficients have normal distributions in most cases, the scalar quantizer should be non-uniform. Hence, it has higher resolution for smaller absolute values and lower resolution for larger absolute values. Figure 3 shows the characteristic of our scalar quantizer. In addition to the computation time reduction, the scalar quantization can save a lot of memory units. Before quantization, we must consume 4 bytes per each HMM parameter to guarantee enough resolution. However, 128-level scalar quantization was shown sufficient; it does not degrade recognition performance. Therefore, we can reduce 75% of the memory usage related with HMM parameters.

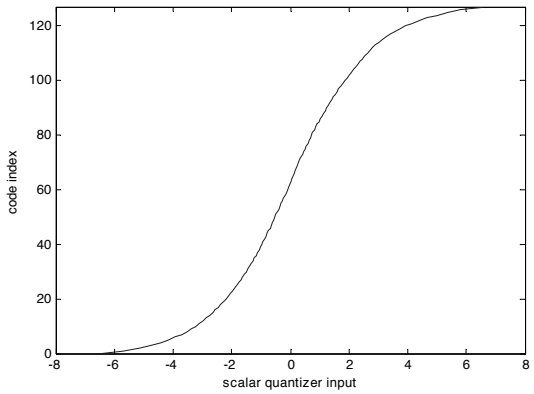


Figure 3 : Input-output characteristic of the scalar quantizer

3.2. Noise reduction

In noise cancellation methods based on single microphone approaches, we found that Kalman filter-based ones showed the best performance. In the Kalman filter applications, the Kalman gain estimation is important but has a large computation load. The Kalman gain is estimated by using the error covariance matrix and the error is defined by $\mathbf{s}_n - \hat{\mathbf{s}}_n$. Where, $\mathbf{s}_n$ and $\hat{\mathbf{s}}_n$ are the clean speech vector and its estimation at time *n*. To estimate the Kalman gain correctly, all elements of the error covariance matrix should be calculated. But this requires much computation time. So, we used only diagonal components of the error covariance matrix to compute the Kalman gain with the motivation that the diagonal elements are significantly more important than the off-diagonal ones. In many speech recognition tests, we identified that this simplification does not affect the performance

3.3. Likelihood computation

It is widely known that compared with the output observation probability of a given HMM state, its transition probabilities have little effect on the likelihood value. Hence, we do not take the transition probabilities into account. Besides ignoring the state transition probabilities, we used another intelligent technique, which skips the calculation of the likelihood at some states. We need not calculate the likelihood for every analysis frame while the frame is in a spectrally stationary interval. In this stationary interval, we calculate the likelihood every two frames. The stationary interval is checked by the beam-pruning occurrence in the Viterbi forward search for the corresponding HMM states.

4. Experiments and results

Basically, our recognizer is an HMM-based one. The 25 commands and Korean names were modeled by pre-selected biphones and 46 monophones. We chose 258 biphones most frequently used in Korean names. Head-Body-Tail (HBT)-based acoustic modeling was used for digit recognition because it showed better performance in our various experiments than triphone- or biphone-based modeling did.

The number of head and tail contexts we defined were 7 each and we obtained 165 subwords for digit recognition. To improve the recognition performance, we trained all HMMs gender-dependently and both gender models were used in our recognizer. Therefore, the total required number of subwords was doubled. Each subword except the body models has 5 states including 2 null states. The body models have 4 states including 2 null states. All subwords have the well-known left-to-right topology without state-skipping. We inserted a short pause model which can be skipped between digit models because users may or may not give pause briefly when they utter continuous digits. Most floating-point operations of our recognition system were converted to fixed-point versions to make the recognition process faster.

All training and test data has 8 kHz sampling rate and 16-bit resolution. They were all collected using SCH-M400 PDA's, in order to avoid performance degradation from device mismatch. To train biphones and monophones for Korean name recognition, 33770 names were recorded from 228 speakers. The digit recognition training DB consisted of 101665 4-digit continuous utterances spoken by 316 speakers.

The recognition feature parameters in our study are the 12^{th} order Mel-frequency cepstral coefficients (MFCC), basically 1 log energy. Finally, 39^{th} order feature parameters are constructed by the basic 13 coefficients and their first and second derivatives and applied to the recognition and the training process. The feature parameters are extracted every 10ms. Each analysis frame has 240 samples. So, after appending 16 zeros to each analysis frame, 256-point fast Fourier transform (FFT) is performed to extract the feature parameters.

To evaluate the recognition rates, we collected 1000 utterances for the name recognition and the digit recognition, respectively. The utterances were spoken by 10 males and 10 females. We also collected the utterances in a car noise

environment by the same speakers. The average SNR's are 27.2 dB, 10.4 dB and 5.7 dB for office, idle in-car, running in-car environments, respectively. When we evaluated the performance in car noise environments, we only considered stationary noises such as engine noises, wheel friction noises, etc.

Table 1 shows the baseline recognition rates of our system for office, idle in-car, running in-car environments.

Table 1 : Baseline recognition rate

	office	in-car (idle)	in-car (running)
200 Korean names	94.7%	92.9%	90.5%
4-digit continuous utterances	94.4%	92.4%	88.3%

Table 2 summarizes RAM/ROM usage for each recognition task. Figures in Table 2 include memory usage by the speech recognizer. MGVSQ means modified global variance and scalar quantization, which was explained in section 3.1.

Table 2 : RAM/ROM usage

	200 Korean names	4-digit continuous utterances
baseline	**3.01MB/2.33MB**	**1.80MB/1.35MB**
state-tying	2.13MB/1.67MB	1.44MB/0.98MB
MGVSQ	1.20MB/0.60MB	0.84MB/0.39MB
state-tying + MGVSQ	1.06MB/0.46MB	0.77MB/0.32MB

By applying the state-tying technique in section 3.3, we reduced about 30% of HMM states without any significant performance degradation

The response time by each algorithm is shown in Table 3.

Table 3 : Response time

	200 Korean names	4-digit continuous utterances
baseline	**5.2 sec.**	**5.5 sec.**
state-tying	5.0 sec.	5.2 sec.
MGVSQ	1.2 sec.	1.3 sec.
state-tying + MGVSQ	1.1 sec.	1.2 sec.

Each figure in Table 3 includes the effects of the noise canceller and the efficient likelihood computation technique. The noise cancellation increases 0.1 second of response time but it reduces about 35% of computation amount compared with the original Kalman filter algorithm. The efficient likelihood computation technique also reduces about 30% of response time consistently.

Table 4 summarizes the decreases of recognition rates in office environments.

Table 4 : Decreases of recognition rates

	200 Korean names	4-digit continuous utterances
state-tying	94.3%	94.0%
MGVSQ	94.5%	94.2%
state-tying + MGVSQ	94.0%	93.8%

Although we omitted the evaluation results in car noise environments, their patterns were similar to the results in Table 4 and the performance decreases was not serious.

In our experiments, MGVSQ was very effective in reducing hardware resources; it reduces 53%(71%) of RAM(ROM) size and 76% of response time. State-tying did not reduce response time significantly, because it does not affect the search network structure in the Viterbi search algorithm. Although state-tying reduces likelihood computation, it does not decrease response time greatly because the frequency of the beam-pruning is decreased. This is because the tied states generate the same likelihood values and consequently, their differences at the nodes in the search network become smaller.

5. Conclusions

In this paper, we introduced the embedded ASR system of HCI Lab. in SAIT. Also, the key technologies implemented to reduce necessary RAM/ROM sizes and to make the system faster were covered. As shown in the experiment results, our embedded ASR system became much lighter and faster by using the reduction techniques. We expected a trade-off with recognition performance, but we could not find any significant degradation.

For further research, we are planning to develop large vocabulary embedded ASR systems, which can recognize up to 10,000 isolated words. This should make ASR possible without DSR concepts in GPS-navigation systems, which have disadvantages such as response time delay, service charge, etc.

6. References

[1] Li, D., Wang, K., "Distributed speech processing in MiPad's multimodal user interface", *IEEE Trans. Speech and Audio Proc*., 10(8):605-719, 2002..

[2] Diaz, J. C., Rodriguez, J. M., "Robust voice recognition as a distributed service**",** *Emerging Technologies and Factory Automation*, Vol.2, pp.571-575, 2000.

[3] Raj, B., Migal, J., "Distributed speech recognition with codec parameters", *IEEE Workshop on ASRU*, pp.127-130, 2001.

[4] Han, I., Jeong, S. and Jon, E., "System and method of reducing memory requirement for embedded ASR", submitted to Korea patent(P20040013815)