ON THE USE OF SCALAR QUANTIZATION FOR FAST HMM COMPUTATION

Shigeki Sagayama and *Satoshi Takahashi*

NTT Human Interface Laboratories
1-2356 Take, Yokosuka-shi, Kanagawa, 238-03 Japan

ABSTRACT

This paper describes an algorithm for reducing the amount of arithmetic operations in the likelihood computation of continous mixture HMM (CMHMM) with diagonal covariance matrices while retaining high performance. The key points are the use of the scalar quantization of the input observation vector components and table look-up. These make multiplication, squaring and division operations entirely unnecessary in the whole HMM computation (i.e., output probability calculation and trellis/Viterbi computation). It is experimentally proved in an large-vocabulary isolated word recognition task that scalar quantization into no less than 16 levels does not cause significant degradation in the speech recognition performance. Scalar quantization is also utilized in the computation truncation for unlikely distributions; the total number of distribution likelihood computations can be reduced by 66% with only a slight performance degradation. This "multiplication-free" HMM algorithm has high potentiality in speech recognition applications on personal computers.

1. INTRODUCTION

The HMM state output probability (more accurately, "probability density") is often the most computationally expensive part of HMM-based speech recognition. In typical cases of an actual speech recognition system, it consumes 45–65% of the time for entire speech recognition in contrast with LPC cepstrum analysis which consumes only 0.8%. Conventionally, the discrete HMMs based on VQ (vector quantization) have been considered as a fast HMM method, although it still requires a considerable number of distance computations and the performance may be slightly lower than that of CMHMMs. This paper aims both at realizing the high performance inherent in CMHMM and low computational cost by cutting down arithmetic operations through the full use of scalar quantization.

The key idea of this paper came up as follows. It has been experimentally shown that 10-th order LSP parameters with 4-bit quantization can represent LPC speech spectra at an average spectral distortion of 0.77 dB[1]. Since cepstrum is even more advantageous than LSP in terms of logarithmic spectral distortion (according to Parseval's theorem), it is expected that 4-bit-quantized cepstral coefficients are sufficient in representing the input speech features for speech recognition.

Authors' E-mail addresses: {saga,taka}@nttspch.hil.ntt.jp

2. HMM STATE OUTPUT PROBABILITY COMPUTATION

In continuous mixture HMMs (CMHMMs) with diagonal covariance matrices, whether in tied or non-tied mixture structures, the output probability, $b_s(\mathbf{x}_t)$, of state s for the given p-dimensional input observation vector, $\mathbf{x} = (x_1, x_2, \cdots, x_p)^T$, is described by:

$$\begin{aligned} b_s(\mathbf{x}_t) & \\ &= \sum_{k \in K_s} w_{ks} \mathcal{N}_k(\mathbf{x}_t) \\ &= \sum_{k \in K_s} w_{ks} \frac{1}{\prod_{i=1}^{p} \sqrt{2\pi\sigma_{ki}^2}} \exp\left\{ -\sum_{i=1}^{p} \frac{(x_i - \mu_{ki})^2}{2\sigma_{ki}^2} \right\} \end{aligned}$$

where K_s denotes the subset of individual distributions consisting of the mixture probability density for state s, and w_{ks} denotes the mixture weight coefficient in state s for the kth individual normal distribution, $\mathcal{N}_k(\mathbf{x}_t)$, with mean, μ_{ki}, and variance, σ_{ki}^2, for each observation vector component, x_{ti}. In practical systems, this probability is often calculated in the logarithmic domain, since its (sign-inverted) logarithmic expression is simplified in the form given in Eqs (1,2).

In these equations, A_{ki} $(i = 1, 2, \cdots, p)$, B_k, W_{ks} $(s = 1, 2, \cdots, S)$ can be computed beforehand for all $k = 1, 2, \cdots, K$. The "addlog" operation (an operator : $(\log \sum \exp)$, i.e., $\mathrm{addlog}(x, y) \equiv \log(\exp x + \exp y)$) can be well approximated by the larger value or by log-table look-up and a few additions if their values are close to each other [2], i.e.,

$$\mathrm{addlog}(x, y) \approx \begin{cases} x & \text{if } x \gg y \\ x + \log(1 + e^{y-x}) & \text{if } x \gtrsim y \\ y + \log(1 + e^{x-y}) & \text{if } x \lesssim y \\ y & \text{if } x \ll y \end{cases}$$

$$
\begin{aligned}
-\log b_s(\mathbf{x}) &= \underbrace{\log \sum_{k \in K_s} \exp}_{\text{"addlog" operation}} \left\{ \overbrace{\sum_{i=1}^{p} \underbrace{\frac{(x_i-\mu_{ki})^2}{2\sigma_{ki}^2}}_{Q_{ki}(x_i)} + \underbrace{\frac{p}{2}\log 2\pi + \frac{1}{2}\sum_{i=1}^{p}\log\sigma_{ki}^2}_{B_k}}^{-\log \text{ value of normal distribution } \mathcal{N}_k(\mathbf{x}_t)} \underbrace{-\log w_{ks}}_{-W_{ks}} \right\} \\
&= \operatorname*{addlog}_{k \in K_s} \left\{ \sum_{i=1}^{p} A_{ki}(x_i-\mu_{ki})^2 + B_k + W_{ks} \right\} \qquad (1)
\end{aligned}
$$

$$
\text{where} \quad A_{ki} = \frac{1}{2\sigma_{ki}^2},\ i=1,2,\cdots,p\ ; \quad B_k = \frac{p}{2}\log 2\pi + \frac{1}{2}\sum_{i=1}^{p}\log\sigma_{ki}^2\ ; \quad W_{ks} = -\log w_{ks},\ s=1,2,\cdots,S \qquad (2)
$$

which requires a log table for values between 1 and 2.

3. SCALAR QUANTIZATION FOR FAST HMM COMPUTATION

3.1. Algorithm

The most computationally intensive part of HMM can be alleviated by table look-up if the input vector is scalar-quantized, i.e., the quantized value, $\tilde{x}_i$ is chosen from $\xi_{ij}(j = 1,2,\cdots,q)$ for each vector component, x_i $(i = 1,2,\cdots,p)$, and if the values of a quadratic form, $Q_{ki}(\xi_{ij}) = A_{ki}(\xi_{ij}-\mu_{ki})^2$, for discrete values of x_i have been calculated beforehand and stored in a table. The required arithmetic operations are as follows:

$$
\begin{aligned}
P_k(\mathbf{x}) &\approx \sum_{i=1}^{p} Q_{ki}(\tilde{x}_i) + B_k \qquad (3) \\
-\log b_s(\mathbf{x}) &\approx \operatorname*{addlog}_{k \in K_s} \{P_k(\mathbf{x}) + W_{ks}\} \qquad (4)
\end{aligned}
$$

which include only table look-ups and additions, and no multiplications/squares or divisions. The entire procedure is summarized in Figure1.

After spectral analysis (e.g., LPC cepstral analysis), scalar quantization can be rapidly executed simply by `float`-to-`int` conversion or, in the case of non-linear quantization, by a binary search algorithm. This is the big advantage of scalar quantization and is essentially different from vector quantization which requires a number of distance calculations.

Table 1 compares the required amount of arithmetic operations for all states per frame required by different approaches to HMM output probability density computation. (Often in practical HMM operation, only "active" states are calculated every frame.) It is seen that the proposed method requires few arithmetic operations.

3.2. Experimental evaluation

The algorithm was implemented and tested on an isolated word recognition task with a vocabulary size of 2620 words using 524 words each uttered by 4 different speakers. The speech analysis conditions were 12kHz sampling, a 256-point Hamming window with 8ms frame shift, 16-th order LPC analysis, 33 feature parameters including the delta log power and the 16 LPC-cepstral and delta-cepstral coefficients.

For simplicity, the quantizer design was done by equally dividing into q levels the approximate parameter distribution range of each feature vector components, x_{ti} $(i = 1,2,\cdots,p)$, given by:

$$
\left[\min_{k=1,2,\cdots,K}(\mu_{ki} - 3\sigma_{ki}),\ \max_{k=1,2,\cdots,K}(\mu_{ki} + 3\sigma_{ki}) \right]
$$

to determine α and β in the quantization rule:

$$
\mathtt{int}\{\alpha_i(x_{ti} - \beta_i)\}.
$$

Alternatively, one can provide a set of non-uniform scalar quantizers for better performance, considering the feature component distribution densities and using a fast binary search algorithm.

Table 3 shows the average word accuracy as a function of scalar quantization resolution, q (i.e., the number of quantization levels). No performance degradation is observed for quantization levels down to $q = 16$.

4. SCALAR QUANTIZATION FOR LOW PROBABILITY TRUNCATION

4.1. Algorithm

Obviously, if any of term in Eq. 3 is large, the log probability of the distribution also becomes large which equivalently means a low value of distribution probability $\mathcal{N}_k(\mathbf{x})$. Since a very low output probability is unlikely to contribute to the state output probability (Eq. 4) and total HMM likelihood accumulation (i.e., trellis or Viterbi), this irrelevant computation can be neglected without completing the full computation of Eq. 3. This situation is entirely predictable from the scalar-quantized values of the input vector. In a practical implementation, the scalar-quantized input components refer to a table (which is, actually, a bit pattern)

1. Design a scalar quantizer for each each of feature vector components.
2. Compute discrete values $\xi_{ij}, (i = 1, 2, \cdots, p;\ \ j = 1, 2, \cdots, q)$ of components of $\mathbf{x}$ levels.
3. For all Gaussian distributions $(k = 1, 2, \cdots, K)$, for all vector components with all possible quantization levels $\xi_{ij}, (i = 1, 2, \cdots, p;\ \ j = 1, 2, \cdots, q)$ of the observation vector $\mathbf{x}_t$, compute $Q_{ki}(\xi_{ij}) = \frac{(\xi_{ij} - \mu_{ki})^2}{2\sigma_{ki}^2}$, $(i = 1, 2, \cdots, p\ ;\ \ j = 1, 2, \cdots, q\ ;\ \ k = 1, 2, \cdots, K)$ and memorize them in a (pqK)-word table.
4. For all Gaussian distributions $(k = 1, 2, \cdots, K)$, compute $B_k = \frac{p}{2} \log 2\pi + \frac{1}{2} \sum_{i=1}^{p} \log \sigma_{ki}^2$ and memorize them in a K-word table. Also, for all states $(s = 1, 2, \cdots, S)$, and for all mixture components $(k = 1, 2, \cdots, K)$, compute the sign-inverted logarithmic mixture weights $W_{ks} = -\log w_{ks}$ and memorize them in a KS-word table.
5. For every input frame, scalar-quantize the observation input vector to obtain discrete values $\tilde{x}_{ti},\ \ i = 1, 2, \cdots, p$.
6. Refer to the table to obtain $Q_{ki}(\tilde{x}_{ti}),\ \ \ k = 1, 2, \cdots, K,\ \ \ i = 1, 2, \cdots, p$ and compute $P_k(\mathbf{x}_t) \approx \sum_{i=1}^{p} Q_{ki}(\tilde{x}_{ti}) + B_k$.
7. For all states $(s = 1, 2, \cdots, S)$, do "addlog" operations: $-\log b_s(\mathbf{x}_t) = \mathrm{addlog}_{k \in K_s} \{\ P_k(\mathbf{x}_t) + W_k\ \}$ to obtain the mixture probabilities. The resulted output probabilities are shared by distinct allophones or phonemes. To further reduce the addlog operation, it can be approximated by "max".

Figure 1: The procedure of fast HMM output probability computation based on scalar quantization and table look-up

1. 2. 3. and 4. (Same as Figure 1.)
5. For each of all vector components with all possible quantization levels $\xi_{ij}, (i = 1, 2, \cdots, p\ ;\ \ j = 1, 2, \cdots, q)$, create a K-bit word, `BitTable[`i`][`j`]`, whose k-th bit represents with 1 or 0 whether ξ_{ij} is "inside" or "outside" the distribution k.
6. For every input frame, scalar-quantize the observation input vector to obtain discrete code values $\tilde{x}_{ti},\ \ i = 1, 2, \cdots, p$. (Same as step 5 in Figure 1.)
7. Take the bitwise *logical AND* operation of bit patterns, $\mathtt{Pattern} = \bigwedge_{i=1}^{p} \mathtt{BitTable}[i][\tilde{x}_{ti}]$, to obtain a bit pattern indicating "active" distributions.
8. Apply steps 6 and 7 in Figure 1 to the k-th distribution only if the k-th bit of `Pattern` is 1. Otherwise, give a large constant to the sign-inverted log likelihood.

Figure 2: The procedure of truncation of HMM output probability computation based on scalar quantization

and quickly check whether they are all within the relevant range or not, referring to a truncation table (bit pattern). If any input vector component is found outside the relevant range, Eq. 4 is substituted with a fixed large value. The summary of the procedure is shown in Figure2.

A similar idea based on vector quantization was proposed by Bocchieri[3] though it still required large computational cost for distance computation, unlike our approach.

4.2. Experimental evaluation

This technique was tested on the same task as that in the previous section. For the i-th vector component, whether "inside" or "outside" the distribution k was determined as whether inside or outside the range $(\mu_{ki} \pm 5\sigma_{ki})$ in this experiment.

Table 3 compares the numbers of actual occurences of evaluating distribution probabilities throughout the task of 2620-word speech recognition using 524 utterances per speaker. Only slight performance degradation is seen in the "truncated evaluation", while the total number of arithmetic operations has been cut down by 65%.

5. CONCLUSION AND DISCUSSION

This paper proposed the use of scalar quantization to completely eliminate multiplication (or squaring, division) operations from the whole HMM computation and to dispense with irrelevant probability computations. The idea was implemented and tested on a large-vocabulary word recognition task. We determined that 16-level scalar quantization is sufficient to assure high recognition performance. 65% of the distribution probability computation can be cut without any significant performance degradation.

Being entirely "multiplication-free", this approach has high potentiality for various applications. Since HMM-based speech recognition can be done mainly by fixed-point addition and subtraction operations only,

Table 1: The number of arithmetic operations required for HMM output probabilities for all states per frame

	multiplication, and division	addition and subtraction	table access	typical count of arithmetic operations †
Discrete (VQ-based) HMM	pq	pq	S	32768 $\times$, 32768 $\pm$
Discrete (Tree VQ-based) HMM	$2p \log_2 q$	$2p \log_2 q$	S	640 $\times$, 640 $\pm$
Continuous Mixture HMM	$2pK$	$2pK + 3MS$	0	128000 $\times$, 135200 $\pm$
Scalar-Quantized CMHMM (proposed)	0	$pK + 3MS$	pK	71200 $\pm$

where p = observation parameter vector dimension, q = VQ codebook size, K = the total number of Gaussian distributions, S = the total number of states, M = the number of mixtures for each state. †A typical case is estimated with $p = 32, q = 1024, K = 2000, M = 4,$ and $S = 600$.

Table 2: Speech recognition performance with scalar-quantized HMM computation (evaluated using speaker-independent 2620-word recognition tested by 4 (2 male + 2 female) speakers each uttering 524 words).

quantization resolution	no quantization	scalar-quantized 64 levels	scalar-quantized 32 levels	scalar-quantized 16 levels	scalar-quantized 8 levels
word accuracy (%) (averaged over 4 speakers)	85.4	85.1	85.2	85.7	81.2

Table 3: Comparison between distribution evaluation counts in full and truncated evaluations in typical isolated word recognition (2620-word recognition tested using 524 words uttered by each of 4 (2 male + 2 female) speakers with 64-level scalar quantization)

speaker	full evaluation		truncated evaluation		reduction rate
	eval count	word accuracy (%)	eval count	word accuracy (%)	
MMS (male)	82243200	85.1	28086072	84.5	0.66
MMY (male)	85662720	80.5	25915436	79.3	0.69
FKS (female)	89856000	86.5	32204265	85.5	0.64
FYN (female)	100089600	88.4	34716610	87.4	0.65
average		85.1		84.2	0.66

except for the speech analysis stage, this algorithm best fits small computers without a floating point accelerator or parallel processing capability. (As for modern engineering workstations equiped with highly sophisticated pipelined floating-point arithmetics, floating point operations may be rather faster than table look-up.) This idea can be also helpful in the design of speech recognition LSI chips and algorithms for fixed-point DSPs.

Future works will include:

- Optimization of quantization levels for cepstral components depending on their distributions. Non-uniform scalar quantization may increase the performance.
- Combination with the four-level tied-structure HMMs[4] which can share computed results in the parameter, distribution, state, and model levels.

6. ACKNOWLEDGEMENT

The authors would like to thank Nobutoshi Shida, Waseda University, for his contribution in implementation and evaluation of the algorithm during his work experiance at NTT in the summer of 1994, and Takatoshi Jitsuhiro, NTT Human Interface Labs, for providing us HMM-related software modules.

7. REFERENCES

[1] F. Itakura and N. Sugamura: "LSP Speech Synthesizer, Its Principle and Implementation," Technical Report of Acoustical Society of Japan, S79-46, pp. 349–356, Nov. 1979. (in Japanese)

[2] E. E. Schwartzlander, Jr. and A. G. Alexopoulos: "The Sign/Logarithm Number System," IEEE Transactions on Computers, C-24, pp. 1238–1242, 1975.

[3] E. Bocchieri: "Vector Quantization for the Efficient Computation of Continuous Density Likelihoods," Proc. ICASSP93 (Minneapolis), pp. II-692–695, 1993.

[4] S. Takahashi and S. Sagayama: "Four-Level Tied Structure for Efficient Representation of Acoustic Modeling," Proc. ICASSP95 (Detroit), this issue.