Multiplierless Refinement Scheme for Interval Calculation of GMM-based Classification

Hidenori Watanabe, Shogo Muramatsu and Hisakazu Kikuchi
Niigata University, 8050 2-no-cho, Ikarashi, Nishi-ku, Niigata 950-2181, Japan
E-mail: hide-w@telecom0.eng.niigata-u.ac.jp Tel: +81-25-262-6746

***Abstract*—In this work, an efficient refinement scheme is proposed for interval calculation of GMM-based classification. The proposed refinement scheme reduces the circuit area and accelerates the throughput from the original. The proposed scheme consists of scaling for the exponential part and shifting of precalculated values of the fractional part. As a result, the refinement process can be constructed with a multiplierless architecture. An experimental implementation on FPGA shows that the proposed method reduces the circuit area under 52.2% and accelerates the throughput over 117.6% from the architecture with the original interval refinement process.**

I. Introduction

Gaussian mixture model(GMM)-based classification, a method of pattern recognition, is widely accepted in many applications such as medical science, accident prevention, monitoring systems and so on. As an advanced monitoring system, wireless sensor networks are expected[1]. A wireless sensor network is constructed with many sensor nodes that consist of sensor, wireless communication module, and battery. It is also possible to mount a camera as a sensor on a sensor node.

Such sensor nodes must have low power consumption because they are assumed to be driven by battery. Sensor nodes with camera is prone to waste high power when the sensor nodes transmit the data to a fusion center or other nodes. To reduce the transmission power, pattern classification can be suitably used. With some classification technique, sensor nodes are required to send only a few significant data.

From these backgrounds, we proposed an interval calculation technique for GMM-based classification as a previous work[2]. The method consists of an initial decision process, refinement process and termination. The initial decision process makes decision with quite simple operations. The refinement process improves the precision of the interval when the initial decision process fails to classify the data. The original refinement process is realized with look-up tables and multipliers. However, there is a possibility that the cost of refinement process becomes larger than other approaches such as Chebychev approximation or CORDIC when many refinement steps are required.

This work proposes a novel refinement process to reduce the calculation cost and shows the significance of reducing the circuit size and accelerating the throughput on FPGA.

II. Review of GMM-based classification

In this section, Bayesian decision rule and a pattern classification procedure using GMM are reviewed.

A. Bayesian decision rule

Bayesian decision rule uses Bayes theorem[3]. Bayes theorem is expressed as

$$P(C|\mathbf{x}) = \frac{P(C)p(\mathbf{x}|C)}{p(\mathbf{x})}, \tag{1}$$

where C is a class and $\mathbf{x}$ is a feature vector. $P(C|\mathbf{x})$, $P(C)$ and $p(\mathbf{x}|C)$ are a posterior probability, prior probability and likelihood function, respectively.

Bayesian decision rule tells us that the optimum selection is given by

$$if \max_{i=0,\cdots,N-1}\{P(C_i|\mathbf{x})\} = P(C_n|\mathbf{x}) \Rightarrow \mathbf{x} \in \text{class } n, \tag{2}$$

where N is the number of classes. From Eq. (1), Eq.(2) can be rewritten as

$$if \max_{i=0,\cdots,N-1}\{P(C_i)p(\mathbf{x}|C_i)\} = P(C_n)p(\mathbf{x}|C_n) \Rightarrow \mathbf{x} \in \text{class } n. \tag{3}$$

Note that $p(\mathbf{x})$ can be omitted because the posterior probability has $p(\mathbf{x})$ in denominator as a common factor.

B. Gaussian mixture model (GMM)

GMM is a linear combination of Gaussian distributions. GMM has more flexibility than Gaussian distribution and can afford to represent multi-modality.

For a given feature vector $\mathbf{x}$, Gaussian distribution $\mathcal{N}(\mathbf{x}|\mu, \mathbf{\Sigma})$ is defined by

$$\mathcal{N}(\mathbf{x}|\mu, \mathbf{\Sigma}) = \frac{1}{(2\pi)^{\frac{D}{2}}|\mathbf{\Sigma}|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\mu)^T\mathbf{\Sigma}^{-1}(\mathbf{x}-\mu)\right\}, \tag{4}$$

where D is the number of variables, μ is a $D \times 1$ mean vector, and $\mathbf{\Sigma}$ is a $D \times D$ covariance matrix.

From Eq. (4), GMM $\mathcal{M}(\mathbf{x}|\mathbf{\Theta})$ is defined by

$$\mathcal{M}(\mathbf{x}|\mathbf{\Theta}) = \sum_{n=0}^{N-1} \alpha_n \mathcal{N}(\mathbf{x}|\mu_n, \mathbf{\Sigma}_n), \tag{5}$$

where N is the number of distributions, α_n is the mixture ratio of the n-th distribution, Θ is a set of parameters $\{\{\alpha_0, \cdots, \alpha_{N-1}\}, \{\mu_0, \cdots, \mu_{N-1}\}, \{\mathbf{\Sigma}_0, \cdots, \mathbf{\Sigma}_{N-1}\}\}$. The mixture ratios $\{\alpha_n\}$ require the conditions $\sum_{n=0}^{N-1} \alpha_n = 1$ and $0 \leq \alpha_n \leq 1$.

C. Decision rule for GMM-based classification

GMM-based classification is realized by using GMM $\mathcal{M}(\mathbf{x}|\mathbf{\Theta})$ as the probability density function $P(\mathbf{x}|C_i)$ in the Bayesian decision rule. From Eq. (3), the decision rule for the GMM-based classification is given by

$$if \max_{i=0,\cdots,N-1} \{P(C_i)\mathcal{M}(\mathbf{x}|\mathbf{\Theta}_i)\} = P(C_n)\mathcal{M}(\mathbf{x}|\mathbf{\Theta}_n) \Rightarrow \mathbf{x} \in \text{class } n. \quad (6)$$

III. Review of interval calculation for GMM-based classification

Our proposed interval calculation for GMM-based classification is an algorithm that reduces the computational cost by adaptively control the computational accuracy without violating the decision rule. GMM-based classification has high computational cost arithmetics in the calculation of Mahalanobis distance and exponential function. The interval calculation scheme is effective for reducing the cost of exponential function.

The interval calculation consists of an initial decision, refinement and termination process. In this section the initial decision and refinement are reviewed.

A. Initial decision process

The initial decision process uses powers of two instead of exponential.

For the sake of convenience, let

$$P(C)\mathcal{M}(\mathbf{x}|\mathbf{\Theta}) = \sum_{n=0}^{N-1} K_n \exp\{-z_n(\mathbf{x})\}, \quad (7)$$

where K_n and $z_n(\mathbf{x})$ are

$$K_n = \frac{P(C)\alpha_n}{(2\pi)^{\frac{D}{2}} |\mathbf{\Sigma}_n|^{\frac{1}{2}}}, \quad (8)$$

$$z_n(\mathbf{x}) = \frac{1}{2}(\mathbf{x} - \mu_n)^T \mathbf{\Sigma}_n^{-1}(\mathbf{x} - \mu_n), \quad (9)$$

respectively, [4]. Note that K_n and $z_n(\mathbf{x})$ are nonnegative.

The interval calculation scheme uses the fact that the exponential function can be rewritten as

$$\exp\{-z_n(\mathbf{x})\} = 2^{-z_n(\mathbf{x})\log_2 e}, \quad (10)$$

where e is the Napier's constant (≈ 2.718). Consequently, Eq. (10) leads an inequation

$$2^{-\lfloor z_n(\mathbf{x})\log_2 e \rfloor - 1} \leq 2^{-z_n(\mathbf{x})\log_2 e} \leq 2^{-\lfloor z_n(\mathbf{x})\log_2 e \rfloor}, \quad (11)$$

where $\lfloor x \rfloor$ is the integer less than or equal to x.

One of the advantages in the forms of $2^{-(\text{integer})}$ is a familiarity with digital logic circuits. In a fixed-point implementation, $2^{-(\text{integer})}$ is achieved by a right bit shift operation. In a floating-point implementation, it is achieved by a subtraction of the exponential part.

From these facts, the upper and lower boundary of the interval are respectively obtained by

$$g_k^{\text{upper}}(\mathbf{x}) = \sum_{n=0}^{N-1} K_{k,n} 2^{-\lfloor z_{k,n}(\mathbf{x})\log_2 e \rfloor}, \quad (12)$$

$$g_k^{\text{lower}}(\mathbf{x}) = \sum_{n=0}^{N-1} K_{k,n} 2^{-\lfloor z_{k,n}(\mathbf{x})\log_2 e \rfloor - 1}. \quad (13)$$

The decision rule in the initial decision process becomes

$$if\ \{g_k^{\text{lower}}(\mathbf{x}) - g_i^{\text{upper}}(\mathbf{x})\} > 0\ for\ all\ i\ \Rightarrow \mathbf{x} \in \text{class } k, \quad (14)$$

where $k, i = 0, 1, \cdots, N-1$.

It, however, may happen that all of $g_k^{\text{lower}}(\mathbf{x})$ are lower than $g_i^{\text{upper}}(\mathbf{x})$ $for\ i \neq k$. We refer to this situation as undecision and the refinement process is required for making decision.

B. Refinement process

The refinement process uses the fractional part in the exponential part to make the interval narrower. The fractional part β is obtained by

$$\beta = z_{k,n}(\mathbf{x})\log_2 e - \lfloor z_{k,n}(\mathbf{x})\log_2 e \rfloor. \quad (15)$$

Operation of $2^{-\beta}$ cannot be realized by simple bit shift or subtraction of the exponential part. In the original refinement process, some multipliers and look-up tables are used.

When β is represented in L-bits, $2^{-\beta}$ is approximated by

$$2^{-\beta} = 2^{-\sum_{l=1}^{L} \beta_l} = \prod_{l=1}^{L} 2^{-\beta_l \cdot 2^l} = \prod_{l=1}^{L} T[l]^{\beta_l}, \quad (16)$$

where the β_l is the l-th bit of β and $\beta_l \in \{0, 1\}$. $T[l]$ is a constant computed by $T[l] = 2^{-2^l}$.

The interval boundaries, $g_k^{\text{upper}}(\mathbf{x})$ and $g_k^{\text{lower}}(\mathbf{x})$ are refined with the following update operations:

$$g_{k,j}^{\text{upper}}(\mathbf{x}) = g_{k,j-1}^{\text{upper}}(\mathbf{x}) \cdot T[j]^{\beta_j}, \quad (17)$$

$$g_{k,j}^{\text{lower}}(\mathbf{x}) = g_{k,j-1}^{\text{lower}}(\mathbf{x}) \cdot T[j]^{-\beta_j}, \quad (18)$$

where j is the number of updates, $j = 1, 2, \cdots L$, $g_{k,0}^{\text{upper}}(\mathbf{x}) = g_k^{\text{upper}}(\mathbf{x})$ and $g_{k,0}^{\text{lower}}(\mathbf{x}) = g_k^{\text{lower}}(\mathbf{x})$.

The decision rule in the refinement process is given by

$$if\ \{g_{k,j}^{\text{lower}}(\mathbf{x}) - g_{i,j}^{\text{upper}}(\mathbf{x})\} > 0,\ for\ all\ i,\ i \neq k \Rightarrow \mathbf{x} \in \text{class } k, \quad (19)$$

where $k, i = 0, 1, \cdots, N-1$.

IV. Multiplierless refinement scheme

The main issue of the interval calculation is the cost of the refinement process. The refinement process requires N multiplication when N-times refinements are executed. It remains possible that the cost of interval calculation becomes higher than the conventional exponent calculation when many refinement steps are required.

In this section, a fast refinement scheme is proposed. The scheme can reduce K multiplications and refinement steps by using newly introduced look-up tables.

A. Calculation of shift amount

The factor $K2^{-z(\mathbf{x})\log_2 e}$ can be rewritten as

$$K2^{-z(\mathbf{x})\log_2 e} = 2^{\log_2 K}\cdot 2^{-z(\mathbf{x})\log_2 e} = 2^{-\{-\log_2 K + z(\mathbf{x})\log_2 e\}}, \tag{20}$$

and we can define an shift amount $q_{k,n}$ by

$$q_{k,n}(\mathbf{x}) = \lfloor -\log_2 K_{k,n} + z_{k,n}(\mathbf{x})\log_2 e \rfloor. \tag{21}$$

If $\log_2 K_{k,n} > z_{k,n}(\mathbf{x})\log_2 e$, the shift amount $q_{k,n}(\mathbf{x})$ become greater than 1. It means that left bit-shift is required in the fixed-point implementation as well as right bit shift. It makes the fixed-point implementation complex. To avoid the left bit shift, let us normalized $K_{k,n}$ as

$$\overline{K}_{k,n} = K_{k,n}/\max\{K_{k,n}\}. \tag{22}$$

Note that $\log_2 \overline{K}_{k,n}$ is not positive because $\overline{K}_{k,n}$ is less than or equal to 1. Equation (22) is based on the fact that Eq. (6) can be rewritten as

$$if \max_{i=0,\cdots,N-1}\left\{\frac{P(C_i)\mathcal{M}(\mathbf{x}|\mathbf{\Theta}_i)}{U}\right\} = \frac{P(C_n)\mathcal{M}(\mathbf{x}|\mathbf{\Theta}_n)}{U} \Rightarrow \mathbf{x}\in \text{class } n, \tag{23}$$

where U is a positive value.

From Eq. (22), the shift amount $q_{k,n}(\mathbf{x})$ is rewritten as

$$q_{k,n}(\mathbf{x}) = \lfloor -\log_2 \overline{K}_{k,n} + z_{k,n}(\mathbf{x})\log_2 e \rfloor. \tag{24}$$

Equation (24) can be realized only by some right bit shifts in the fixed-point implementation.

B. Look-up table for multiplierless interval calculation

The look-up table of our proposed multiplierless scheme is different from the original one.

In the multiplierless scheme, the fractional part β of the exponential part is given by

$$\beta = -\log_2 \overline{K} + z(\mathbf{x})\log_2 e - \lfloor -\log_2 \overline{K} + z(\mathbf{x})\log_2 e \rfloor, \tag{25}$$

and a function $s(\beta)$ is defined as

$$s(\beta) = 2^{-\beta}. \tag{26}$$

In the multiplierless scheme, function $s(\beta)$ is realized by look-up table .

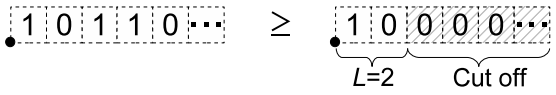

Fig. 1. Relation between the true value (left-hand side) and round-down value (right-hand side). Round-down value becomes 0.5.

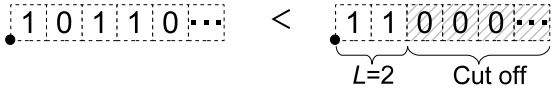

Fig. 2. Relation between the true value (left-hand side) and round-up value (right-hand side). Round-up value becomes 0.75.

C. Refinement process with shift amount

The shift amount and the table from Sec. IV-A and IV-B make it possible to reduce the multipliers.

With Eqs.(24) and (26), the refined interval operation is realized by

$$g_k^{\text{upper}}(\mathbf{x}) = \sum_{n=0}^{N-1} h_{k,n}^{\text{upper}}(\mathbf{x}), \tag{27}$$

$$g_k^{\text{lower}}(\mathbf{x}) = \sum_{n=0}^{N-1} h_{k,n}^{\text{lower}}(\mathbf{x}), \tag{28}$$

where

$$h_{k,n}^{\text{upper}}(\mathbf{x}) = s(\beta_{k,n}) \gg q_{k,n}(\mathbf{x}), \tag{29}$$

and

$$h_{k,n}^{\text{lower}}(\mathbf{x}) = \begin{cases} s(0) \gg (q_{k,n}(\mathbf{x})+1) & (\beta_{k,n} = 1-2^{-L}) \\ s(\beta_{k,n}+2^{-L}) \gg q_{k,n}(\mathbf{x}) & (otherwise) \end{cases}. \tag{30}$$

Note that there is no multiplier in Eqs. (29) and (30).

Equation (30) is based on the following inequation:

$$\sum_{l=1}^{L}\beta_l \le \sum_{l=1}^{\infty}\beta_l < \left(\sum_{l=1}^{L}\beta_l\right) + 2^{-L}, \tag{31}$$

where the l-th bit of β is expressed by β_l, $\sum_{l=1}^{L}\beta_l$ is the L-bit round-down of β and $\sum_{l=1}^{\infty}\beta_l$ is the true value of β. Figures 1 and 2 show some examples for $L = 2$ and true value 0.6875.

D. Decision rule in the proposed refinement scheme

The decision rule is given by

$$if\ \{g_k^{\text{lower}}(\mathbf{x}) - g_i^{\text{upper}}(\mathbf{x})\} > 0\ for\ all\ i \Rightarrow \mathbf{x}\in \text{class } k, \tag{32}$$

where $k, i = 0, 1, \cdots, N-1$.

V. Performance evaluation on FPGA

In this section, in order to verify the significance of the novel refinement scheme, let us evaluate the circuit size of GMM-based classifier with the interval calculation in FPGA is evaluated.

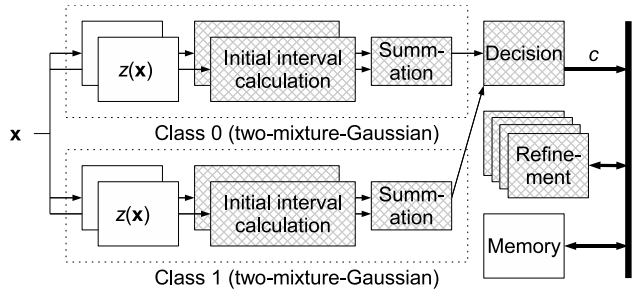

Fig. 3. Original interval calculation classifier

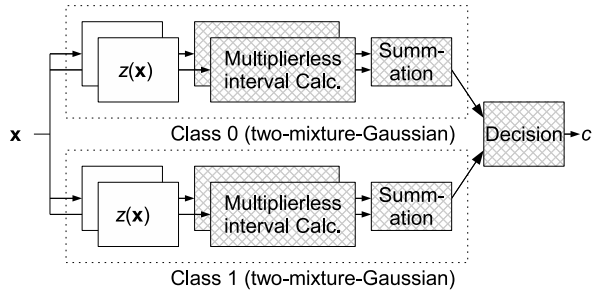

Fig. 4. Proposed interval calculation classifier

A. Implementation condition

In the followings, we summarize the design condition for a GMM-based classifier with the interval calculation

- Target FPGA: Xilinx XC3SD3400
- Development environment: Xilinx ISE 10.1
- Optimization: speed
- Data type: 32-bit and 64-bit fixed-point
- The number of classes: 2
- The number of distributions: 2
- Embedded RAMs and multipliers are not used

The original refinement scheme was estimated by adding four multipliers to the initial decision process. The multipliers were generated by Xilinx Core Generator and they were configured as parallel architecture.

Figures 3 and 4 show the block diagrams of the circuits in this evaluation, where the hatched blocks denote the target in this evaluation.

B. synthesis report

Tables I-IV show the synthesis results of the number of flip-flops, look-up tables (as elements of FPGA), max clock period, throughput, and clock latency. The values in parentheses show the ratio of the improvement from the proposed scheme to the original scheme. Symbols that $+$ or $-$ on Table III and IV mean uncertainty of the estimated values because the original refinement process is adaptive scheme, and the value shows the best case where only a few refinement processes are required in the interval calculation.

From these results, it was confirmed that the multiplierless refinement scheme for the interval calculation can reduce the circuit area under 52.2% and can accelerate the throughput over 117.6%.

TABLE I
RESULTS OF LOGIC SYNTHESIS FOR 32-BIT IMPLEMENTATION

	FFs	LUTs	Max clock period (MHz)
Prop. 1-bit	2690(42.2%)	2423(40.9%)	180.343(117.6%)
Prop. 2-bit	3038(47.6%)	2778(46.9%)	180.343(117.6%)
Prop. 3-bit	3274(51.3%)	3031(51.1%)	180.343(117.6%)
Prop. 4-bit	3302(51.8%)	3095(52.2%)	180.343(117.6%)
Orig. refine.	6380	5928	153.374

TABLE II
RESULTS OF LOGIC SYNTHESIS FOR 64-BIT IMPLEMENTATION

	FFs	LUTs	Max clock period (MHz)
Prop. 1-bit	4385(22.2%)	4043(21.0%)	178.253(171.84%)
Prop. 2-bit	5078(25.7%)	4720(24.5%)	178.253(171.84%)
Prop. 3-bit	5547(28.0%)	5188(27.0%)	178.253(171.84%)
Prop. 4-bit	5615(28.39%)	5320(27.7%)	178.253(171.84%)
Orig. refine.	19779	19232	103.734

TABLE III
RESULTS OF THROUGHPUT AND CLOCK LATENCY FOR 32-BIT IMPLEMENTATION

	Throughput($\times 10^6$)	Latency
Prop. 1-bit	180.343(117.6+%)	12
Prop. 2-bit	180.343(117.6+%)	12
Prop. 3-bit	180.343(117.6+%)	12
Prop. 4-bit	180.343(117.6+%)	12
Orig. refine.	153.374-	9+

TABLE IV
RESULTS OF THROUGHPUT AND CLOCK LATENCY FOR 64-BIT IMPLEMENTATION

	Throughput($\times 10^6$)	Latency
Prop. 1-bit	178.253(171.84+%)	13
Prop. 2-bit	178.253(171.84+%)	13
Prop. 3-bit	178.253(171.84+%)	13
Prop. 4-bit	178.253(171.84+%)	13
Orig. refine.	103.734-	11+

VI. CONCLUSION

In this work, an efficient refinement scheme was proposed for interval calculation of GMM-based classification. The proposed scheme consists of scaling for the exponential part and shifting of pre-calculated values of the fractional part. As a result, the refinement process could be made by multiplierless architecture. By the experimental implementation on FPGA, it was confirmed that the proposed method reduced the circuit area under 52.2% and accelerated the throughput over 117.6%. Future task is implementation for classification applications.

REFERENCES

[1] Ministry of internal affairs and communications, "White Paper on telecommunications 2006", p. 219, 2006. http://www.johotsusintokei.soumu.go.jp/whitepaper/ja/h18/pdf/index.html

[2] Shogo Muramatsu and Hidenori Watanabe, "FAST ALGORITHM FOR GMM-BASED PATTERN CLASSIFIER," *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing(ICASSP2009)*, pp.633-636, Taipei, April 2009.

[3] Christopher M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.

[4] Minghua Shi, A. Bermak, S. Chandrasekaran, and A. Amira, "An efficient FPGA implementation of Gaussian mixture models-based classifier using distributed arithmetic," in *IEEE Proceedings of International Conference on Electronics, Circuits and Systems '06*, pp.1276-1279, Dec. 2006.