Floating-point to Fixed-point Conversion of Tropical Wood Recognition System Classifier

Enas Dhuhri Kusuma, Rubiyah Yusof, and Mohammad Fauzi Othman

***Abstract*—**Automated wood recognition system is computationally expensive and require large data processing. The most time and resources consuming part of the system is classifier. Linear Discriminant Analysis (LDA) is used as classifier in existing wood recognition system on training stage and testing stage. LDA implementation to hardware is required for accelerating classifier performance. In hardware implementation, one of its important processes is floating-point to fixed-point conversion. Fixed-point number operations requires less computation complexities and hardware resources than floating-point. Fixed-point conversion is essensially process to determine optimum wordlengths of every involved variables. Total wordlength consists of integer length and fraction length. The optimum set of wordlengths will produce lowest hardware cost with quantization error does not exceed maximum error specification. To perform wordlength optimization, LDA algorithm will be divided into groups. For each group, integer length will be determined first by performing range analysis for all variables. Fraction length then will be determined by performing multi-objective GA with Pareto ranking system for the GA selection process. The best solution is selected from the Pareto front with lowest hardware cost and error power satisfy maximum error specification. The proposed method gives lowest hardware cost compared with sequential search, single (combined) objective GA, and Matlab fixed-point toolbox. This method is also more convergent in searching best hardware cost with constrained error compared with single-objective GA.

***Keywords*–** fixed-point, LDA classifier, wordlength optimization, multi-objective GA, Pareto ranking

I. Introduction

A. Background

Wood recognition system is a special purpose system which is enabled to determine wood species based on properties measured from wood samples. Automated wood recognition systems have been implemented by many researchers [1]–[4] in PC based systems. Wood recognition system implemented by Khairuddin et.al [4] gets 95% accuracy for 52 species. The system consists of training and testing stage. Training stage selects most discriminative features on training data. Testing stage runs the trained system by applying input data and produce classified output data. Linear Discrimination Analysis (LDA) is used as classifier in both training and testing stage, which is the most time and resource consuming part.

Embedded system implementation of large matrix processing algorithm will require fixed-point arithmetic to save chip area, operation speed, and power consumption, because of complexity and hardware cost of floating point arithmetic. Fixed-point implementation of digital signal processing needs long wordlengths to guarantee high precision, especially for large two-dimensional data. Whereas, hardware implementation of algorithm need wordlengths as short as possible to reduce hardware costs. With only trial-and-error technique, determination of optimum wordlength can be time consuming. Common algorithm such as FFT [5] will not need special operations to convert its variables floating-point to fixed-point and apply wordlength optimization, but complicated ones will need. In general, wordlength optimization can be performed by solving the algorithm as equation analitycally [6]–[13] or running algorithm in simulation and make measurement in its variables [14]–[17]. For complicated system, which involve large matrices computations, system solving by simulation is preferred, since the difficulty of modelling such system analytically. Methods for wordlength determination can be classified in two kinds. First kind of methods is gradient based or greedy method which determines optimum set of wordlengths by performing sequential search from arbitrary initial wordlength and search the target wordlength in up direction or down [14], [16]. The other method is based on random search by performing genetic algorithm (GA) [10], [11]. All methods developed in wordlength determination researches are general purpose method with implementation examples are in small scale algorithms or computations such as small matrix computation, filter and mathematical formula.

LDA classifier in wood recognition system is considered as large scale algorithm. It consists of several computation units with each of them involves in large matrices operation. The objective of the research conducted in this paper is devising a method to implement LDA classifier algorithm for wood recognition in fixed-point number. Hence, this paper will present the method and its effectiveness for wordlength determination of the LDA classifier.

This paper is organized as follows. Next two subsections in the introduction will explain briefly about LDA computation, fixed-point number format, and multi-objective optimization. In section 2,several previous works for floating-point to fixed-point conversion are presented. Section 3 gives detailed de-

This work was supported by Ministry of Education Malaysia and Universiti Teknologi Malaysia through a research grant (04H40) titled "Dimension Reduction and Data Clustering for High Dimensional and Large Datasets".

E. D. Kusuma is with Malaysia-Japan International Institute of Technology, Universiti Teknologi Malaysia and the Department of Electrical Engineering and Information Technology, Faculty of Engineering, Gadjah Mada University, Indonesia (e-mail: enas.dhuhri@gmail.com, enas@ugm.ac.id)

R. Yusof is with Malaysia-Japan International Institute of Technology and Center of Artificial Intelligence and Robotics, Universiti Technology Malaysia (corresponding author to provide phone: +603-26913710; fax: +603-26970815; e-mail: rubiyah@ic.utm.my)

M.F. Othman is with Electrical Engineering Faculty and Center of Artificial Intelligence and Robotics, Universiti Technology Malaysia (e-mail: fauzi@fke.utm.my).

scription about LDA algorithm and its breaking down into computation units. Section 4 presents the method for implementing the algorithm in fixed-point. In section 5, simulation results will be discussed, section 6 is the conclusion, and section 7 for possible future works.

B. LDA Computation

Input of feature selection unit is combination of basic gray level aura matrices (BGLAM) feature and statistical pores pattern distribution (SPPD) of preprocessed wood cross section image developed by Cordova [3] and Khairuddin et.al [4]. After pass several conditioning procedure and normalized, feature data matrix will be saved in database to be processed in classifier. In training process, training data matrix will be masked with several bit string to select which feature will be processed in the training stage. The selection is column wise, then only selected columns will be processed in stage. The final result of the computation is the selected column combination which produces highest fitness. Thus, if the number of total features is 157 and the training matrix size is m x n, number of column being processed in LDA (n) will be various depends on the bit string with $n \leq 157$.

In its implementation as classifier [4], [18], derived LDA computation can be shown in equation (1).

$$\begin{aligned} f_k(\boldsymbol{x}) &= \log_2 p_k + \log_2 |R| \\ &\quad - \frac{1}{2}[(\boldsymbol{x}-\boldsymbol{\mu}_k)^T (R^T R)^{-1} (\boldsymbol{x}-\boldsymbol{\mu}_k)] \end{aligned} \tag{1}$$

Function $f_k(\boldsymbol{x})$ represents how proper sample data vector $\boldsymbol{x}$ can be classified as k_{th} class member. Sample data vector $\boldsymbol{x}$ will be classified as k_{th} class member if $f_k(\boldsymbol{x}) > f_l(\boldsymbol{x})$, $\forall l \neq k$. $\boldsymbol{\mu}_k$ is a vector which contains average value of training data in group k. Vector $\boldsymbol{\mu}_k$ and $\boldsymbol{x}$ have the same length with number of features. $\log_2 p_k$ is base-2 logarithm of prior probability of k_{th} class. Prior probabilities for species k is equal number of species k in training database divided by number of total species in training database. R matrix is representation of normalized training data matrix, which is produced by QR decomposition of normalized training data matrix A, shown in equation(2).

$$\mathbf{A = Q\ R} \tag{2}$$

QR decomposition of covariance matrix is used as data reduction method in LDA based classifier. QRD can represent covariance matrix with much smaller data [19]. QRD factorizes m x n sized input matrix A with m > n into m x n orthogonal Q matrix and n x n upper triangular R matrix, while Q matrix is unused.

Hence, overall computation of the classifier can be broken down to several computation modules as follows:

1) QRD of normalized training data from database, produce R matrix.
2) Inverse of R matrix.
3) Matrix multiplication modules to compute $(\boldsymbol{x}-\boldsymbol{\mu}_k)^T (R^T R)^{-1} (\boldsymbol{x}-\boldsymbol{\mu}_k)$
4) Determinant of R matrix. Because of its upper triangular characteristic, $\log_2 |R|$ is the sum of the logarithm of R's diagonal components absolute value.
5) Computation of each sample's rank. This computation is also extensive because involves few matrices multiplications with the size are as large as the size of sample matrix.

C. Fixed-point number

Fixed-point number is preferrable for embedded system implementation since its architectures, operators, and registers need less area compared to floating point. Fixed-point (FXP) number has finite dynamic range and quantization [16], [17]. That makes one FXP format usually can only be implemented to one variable in an algorithm. Unsuitable FXP format implementation can result in signal distortion if its length is insufficient or result in redundancy if its length is more than requirement. Fixed point number consists of sign bit, integer part, and fractional part. Notation (WL,FL, 0 unsigned/1 signed) will be used to represents the number with WL is total bits and FL is number of fractional bits. the integer length (IL) is the number of bits used in the integer part, and the fractional length (FL) is assigned to the fraction. Hence, total bits or total word-length (WL) of a two's complement number is IL + FL + 1. The IL will depends on the number's range ($\boldsymbol{r}$) and the FL will determine quantization size (Δ) of the number, which is described as $-2^{IL} \leq \boldsymbol{r} < 2^{IL}$ and $\Delta = 2^{-FL}$.

D. Multi-objective optimization

In the optimization cases, sometimes there are several objectives that have to be simultaneously satisfied. Those objectives sometimes are contradicting and cannot reach optimum at the same time. Moreover, sometimes improving the value of one objective means getting worst values for remaining objectives. One of techniques to obtain several dominant values among the solution spaces is Pareto method [20]–[22]. By performing Pareto method, the user will obtain a set of non-dominated solutions. From non-dominated set, one solution which is best suits the requirements and needs can be selected.

If it is assumed that there are k objective functions to be minimized, definitions related to optimality are as follows [20]

Definition 1: Dominating: Vector $\boldsymbol{v}$ is considered as dominating vector $\boldsymbol{u}$, if:

$$\begin{aligned} &\forall i \in 1,2...k : f_i(\boldsymbol{v}) \leq f_i(\boldsymbol{u}) \text{ and} \\ &\exists j \in 1,2...k : f_j(\boldsymbol{v}) < f_j(\boldsymbol{u}) \end{aligned} \tag{3}$$

Thus for all fitness function, v's output is not greater than u's and for at least one fitness function, v's output is giving smaller value than u's

Definition 2: Pareto optimality: Vector $\boldsymbol{x} \in S$ is a Pareto-optimal solution, if and only if:

$$\begin{aligned} &\text{there is no vector } \boldsymbol{y} \in S, \\ &\text{for which } f(\boldsymbol{y}) = (f_1(\boldsymbol{y})...f_k(\boldsymbol{y})) \\ &\text{dominates } f(\boldsymbol{x}) = (f_1(\boldsymbol{x})...f_k(\boldsymbol{x})) \end{aligned} \tag{4}$$

Thus a point can be included in Pareto front, if that point is less than all remaining points on solution space in at least one objective function.

II. Related Works

Rocher, et.al [6] developed automatic hardware description language (HDL) generator for digital filter, which involved wordlength optimization on its process. This research is considered to be most suitable for fixed-point conversion of digital filters but quite unsuitable for more complicated algorithm. Optimization method used in the research is algorithm grouping. A group contains operations executed in same operator and those operations will have the same WL corresponding to operator WL. This method then applied gradient search for finding optimum WL in every group.

Analytical method for implementing mathematical function in fixed-point system has been developed by Lee, et.al [7] and Pradhan, et.al [13]. With Pradhan, et.al create only error models, Lee, et.al create models of basic non-linear functions for limited intervals, which cover error models and cost models. Those researches give contributions for modelling error and cost of several mathematic functions in fixed-points, with certain word length. They also devised a method to convert floating-point to fixed-point system for more complex system. Still using analytical method, they implemented affine arithmetic technique to create error model of the system. However, this method requires that algorithm being converted can be represented in analytical form, therefore it is unsuitable to be implemented in complex system.

Han, et.al [14], [15] emphasized their work on wordlength optimization method in simulation approach. Their work covers sensitivity informations, consist of root mean squared error and hardware cost, which is useful in the optimization and their combination. In wordlength implementation, they utilized sequential search (hill-climbing) to get the optimum point. Sequential search makes the optimization process run fast and suitable with simulation approach for system with a few variables and low complexity such as digital filter. For system with many variables and high complexity, sequential methods have not been tested. In its implementation on more complicated system, a drawback which possible to be happened to this method is possibilty to be trapped in local optimum, since sequential search is only local search. Sequential search based method strongly depends on initial word selection which determines where the next point is.

Roy, et.al [16] and Banerjee, et.al [17] introduced 3 steps floating to fixed-point conversion which cover range determination, coarse optimization, and fine optimization. Range determination finds the proper integer bit length for every variable, after integer length being determined, the process is continued in finding fractional bit length. Coarse optimization determines uniform fraction length before being fine optimized to find multiple fractional length . Their work used simulation based analysis and found the optimum worldlenght through sequential search. The other approach in this research is about using scaled version of original system to find optimum wordlength, then determined wordlength will be applied in the original system. However, the research only used one sensitivity parameter which was error measurement between floating and fixed-point representation. Similar research worked on error measurement for sensitivity parameter is from Rodellar, et.al [23]. Different from two previous researches which proposed general purposed method, that research had been implemented on specific case in adaptive noise canceller. Those researches also only tested the method in small-scaled system, therefore effectiveness of the method has not been proven in large system.

Constantinides, et.al [8], [9] implemented WL optimization for linear DSP. The method in determining wordlength is slightly similar with another works mentioned. Nevertheless, this paper contains a good models for calculating hardware cost estimation. In this paper, the researchers provided several hardware cost estimation for several basic operations such as adder and multiplier. Hardware cost models proposed by Constantinides, et.al are more detail compared with another cost models proposed by Zhang, et.al [12] and Ahmadi, et.al [10] which only used linear relationship for adder, and second order for multiplier. Division hardware model with respect to wordlength, had been proposed by Lam [24]. The researcher made a simple division model whose complexity is as simple as complexity of addition operator.

Ahmadi, et.al [10] and Sulaiman, et.al [11] had proposed genetic algorithm (GA) for optimizing WL with target device is digital signal processor. They used analytic method to get sensitivity informations. Although it was mentioned they used signal-to-noise ratio and hardware cost, they combined sensitivity informations into one objective function by performing weighting sum. Hence, it is mainly a single or combined objective GA. In the research [10], target algorithm is divided into several functional units, where every variable in the same unit has the same WL. Consequently, if certain functional unit has the same WL for all variables, range dynamics inside the functional is quite narrow. Meanwhile, LDA computation has possibility to have wide range variables in its several functional units, especially those which have division and large matrix multiplication.

Another randomized method to determine wordlength had been proposed by Chen, et.al [25]. To determine wordlength of variables and parameters in the communication related module, they used Monte Carlo simulation method. With simulation by giving certain amounts of randomized input to the module, the range and precision of variables and output of the module can be decided. From those ranges knowledge, wordlength and fraction length of variables and parameters can be determined. However, Monte Carlo method, due to its requirement of numerous simulations, is not very suitable to be implemented in LDA case, which needs long computation time.

This paper will discuss about fixed-point wordlength optimization of LDA classifier, which is a large scale DSP algorithm. Method developed is multi-stage, consists of functional grouping and for every group, range and fractional optimization will be performed. Functional grouping to LDA is required since the complexity of its computation. Two-objective GA will be utilized to perform fractional width optimization.

III. LDA Algorithm

A. Algorithm Examination

Detailed LDA algorithm can be described in Figure 1. LDA algorithm has 3 main inputs, which are training data, sample data being tested, and training group means. All inputs are saved in matrix form in storage.

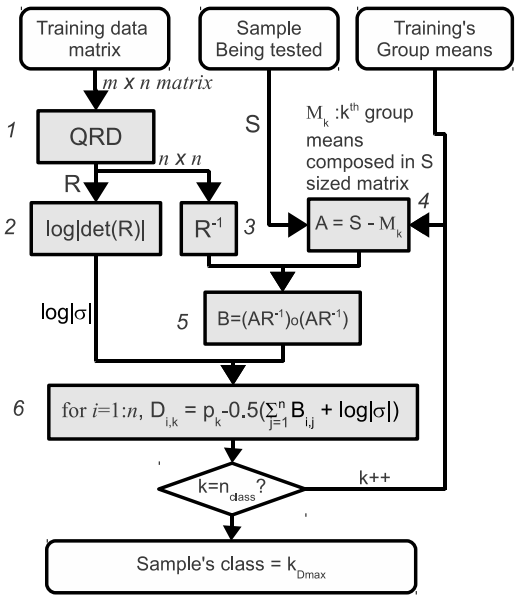

Fig. 1. LDA simplified flow diagram

Functionally, LDA can be divided into 5 major blocks as follows,

1) QRD block, produces R matrix,
2) logarithm of R matrix determinant ($log|R|$),
3) matrix inverse block, produces inverse of R matrix (R^{-1}),
4) sample and group means matrix subtraction block, produces A matrix,
5) multiplication of A and R^{-1}, then perform inner square of its result ($(AR^{-1}) \circ (AR^{-1})$) and LDA main loop computation.

Function blocks 1,2,3 are executed once in overall process, while blocks 4 and 5 are executed *k* times, with *k* represents number of group in classification. Since R matrix as result of QRD is an upper triangular, algorithm for computing inverse can be modified to make a lighter computation and shorter processing time. Even though it will not be discussed in this paper, the final product of the research is embedded system in System on Programmable Chip (SoPC) platform. In SoPC, application will be implemented in both custom hardware and software executed by embedded processor. In this research, the system will be pre-examined to decide which block will be prioritized to be implemented as fixed-point in hardware and which one will be run in software as floating-point. Pre-examination is performed by simply making measurements in computation time for every computation block in Matlab. Even though computation is in Matlab, it is not performed by utilizing the existing toolbox, instead the algorithm is created by combining basic functions. Result for time measurement in every block is presented in Table I. Measurement was done for one-time computation and not total computation time in whole loop.

TABLE I
TIME MEASUREMENT RESULT FOR EACH FUNCTIONAL BLOCK

block number	execution time (s)
1	20.267
2	0.013
3	0.389
4	0.011
5	3.5

According to Table I, execution time for functional blocks 1,3,and 5 are very dominant compared to total execution time. Thus, blocks 1,3,and 5 are the functional blocks will be converted to fixed-point and wordlength of their variables will be optimized. Meanwhile, we let another blocks be computed in software level with floating-point variables. The next discussion in the LDA algorithm section will only cover parts which need algorithm modifications. Those parts are QRD and triangular matrix inversion.

B. Householder QRD

There are many methods of computing QRD, such as Modified Gram-Schmidt, Givens Rotation and Householder [26]. In this paper, Housholder method will be utilized. Householder method's advantage among others is the capability to simultaneously eliminate matrix elements below diagonals. Algorithm for Householder QRD is described in Algorithm 1 (HQR). Householder method processes input matrix in column wise. In every column processed, it will produce Householder vector **v**. The vector will be multiplied to processed matrix to get all elements below diagonals zeroed. Problem will come if norm of column being processed (shown by $\boldsymbol{d_3}$ in Algorithm 1) is near zero. Thus, in pseudocode, arbitrary reflection will be performed if zero column. If algorithm get zero column, Householder vector will be replaced by identity vector, with all zero elements but first element is one.

Algorithm 1 also produces output which is used for updating rightside blocks of the processed block. Block update process is simply matrix addition and multiplication, shown in equation (5)

$$M = A + Y * W' * A; \tag{5}$$

with **A** is matrix being updated, Y and W are HQR results used for update, and M is updated matrix result. To apply QRD in large data size, blocked QRD algorithm will be applied since the research orientation is to implement computation in FPGA.

C. Blocked Householder QRD Algorithm

In blocked algorithm, input matrix is partitioned into blocks and QRD is applied to every block with certain technique. There are several techniques in blocked QRD computation, which are tall-skinny QRD [27] and tiled QRD [28]. Tiled QRD is prefered for implementation to resource limited device, since it works in smaller block with adjustable size. In

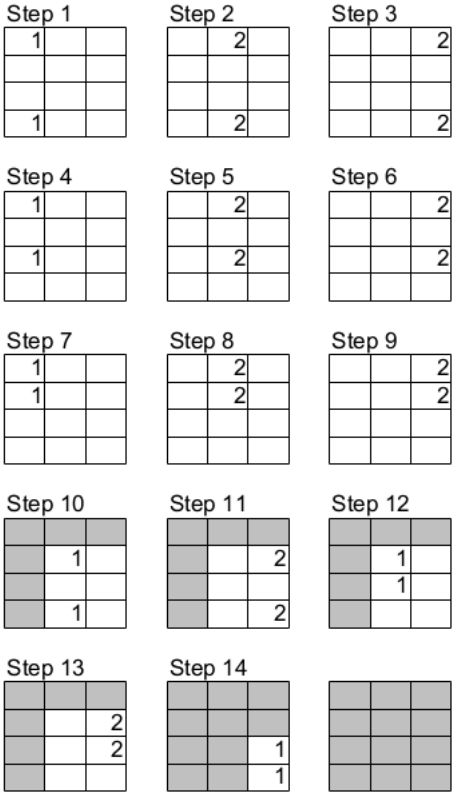

Fig. 2. Bottom-up Method of Tiled QRD

tiled QRD, input matrix is divided into several square blocks. Individual QRD are implemented to diagonal block and all blocks below it sequentially. In one operation, individual QRD process 2 blocks, one block is diagonal block as pivot, and the other is block below diagonal.

Two blocks being processed by individual QRD is shown in Figure 1. The upper block of individiual QRD is pivot block, a block which will be triangularized. Lower block is called zeroed block because all of its contents will be eliminated. Right side blocks of diagonal blocks will be modified in update processes. Tiled QRD algorithm discussed in this paper will be bottom-up variant, which is used in Fibonacci and Greedy methods [28]. In bottom-up method, shown in Figure 2, computation is done in lower blocks first while diagonal block is still used as pivot block. Shown in Figure 2, two blocks labelled 1 are blocks being processed by individual QRD and label 2 means two blocks are being updated.

Algorithm 1 : HQR

A = input matrix
R = output matrix, result of QRD
W and Y = output matrix, used in blocked QRD to update the next block

1: $[m, n] = size(A)$
2: $Y = zeros(size(A))$
3: $W = zeros(size(A))$
4: $vzero = 2^{-16}$
5: **for** $k = 1 : n$ **do**
6: $x = A(k : m, k)$;
7: $d1 = x' * x$;
8: $d2 = \sqrt{d1}$;
9: $v = x$;
10: $v(k) = x(k) + sign(x(k)) * d2$;
11: $d3 = \sqrt{v' * v}$;
12: **if** $d3 < vzero$ **then**
13: $v = zeros(size(x))$;
14: $v(k) = 1$;
15: **else**
16: $v = v/d3$;
17: **end if**
18: **for** $j = k : n$ **do**
19: $y = A(k : m, j)$;
20: $d4 = v' * y$;
21: $y = -2 * v * d4 + y$;
22: $A(k : m, j) = y$;
23: **end for**
24: **Y and W generation**
25: **if** $k == 1$ **then**
26: $Y(:, k) = v$;
27: $W(:, k) = -2 * v$;
28: **else**
29: $z = -2 * v$;
30: **for** $i = 1 : k - 1$ **do**
31: $yv = Y(:, i)' * v$;
32: $z = z - 2 * W(:, i) * yv$;
33: **end for**
34: $W(:, k) = z$;
35: $Y(:, k) = v$;
36: **end if**
37: **end for**
38: $R = A$

D. Triangular Matrix Inversion

Triangular matrix, upper or lower, is easier and requires less number of operations. Input of this module is upper triangular matrix *R* generated from HQR module. While full $n \times n$ matrix requires n^3 operations to compute its inverse, triangular system only requires $\frac{1}{2}n^2$ operations to solve [29]. The algorithm used for computing the inversion of upper triangular matrix is given in Algorithm 2. Critical point of the algorithm is division operation $X(j, j) = \frac{1}{R(j,j)}$, which will be unstable if $R(j, j)$ near zero. Thus to anticipate that, the value of $R(j, j)$ will be replaced by the smallest value in the diagonal which still greater than zero (Algorithm 2 line 7).

Algorithm 2 : InverseTriu

R = input matrix, upper triangular, result of QRD
Ri = output matrix, inverse of R

1: $n = size(R, 1)$;
2: $X = zeros(n)$;
3: $dR = |diag(R)|$;
4: $md = 2^{-4} min(dR(dR > 0))$;
5: **for** $j = n : -1 : 1$ **do**
6: **if** $|R(j, j))| < md$ **then**

```
        R(j,j) = md;
    end if
    X(j,j) = 1/R(j,j)
    for k = j+1 : n do
        for i = j+1 : n do
            X2 = X(j,i)R(i,k);
            X(j,k) = X(j,k) + X2;
        end for
    end for
    for k = j+1 : n do
        Y = -X(j,j) * X(j,k)
        X(j,k) = Y;
    end for
end for
Ri = X;
```

E. LDA Main Loop

The purpose of using LDA is to compute the fitness of specific sample. Based on equation (1), LDA main loop computation is shown in Algorithm 3. The algorithm classified all given samples to specific class, then compared classification results to the reference. Input of this module is R^{-1} matrix from INV module, precomputed *logDetSigma* which is logarithm of *R*'s determinant, sample data being classified (*Sample*), and group means of training data (*gmeans*). Quantity and size of R^{-1} matrix depends on training data, input of HQR module. Training data fed to the HQR algorithm depends on the masking word applied to whole training data. If classification output from algorithm is the same as the one from reference, then a counter variable is incremented, until all samples are examined. Fitness of certain training data combination is determined by the final amount of the counter.

Algorithm 3 : LDAMain
input: R^{-1} matrix, logDetSigma, sample, gmeans
output: D matrix, can be used to determine sample's class

```
A = zeros(size(sample));
m = NumberOfSamples;
for k = 1 : NGroups do
    for i = 1 : m do
        A(i,:) = sample(i,:) - gmeans(k,:);
    end for
    A = A * Ri;
    A2 = A. * A;
    D(:,k) = 1/NGroups - .5 * (sum(A2,2) + logDetSigma);
end for
```

Data obtained from database contains training data itself (*Training*), sample data for testing, which is not included in training (*sample*) and training data group average (*gmeans*). *Gmeans* was average values computed within every group in training data with *NGroups* was number of groups defined in the system, which was 52 groups. Value of classification result was in logarithmic scale to make determinant computation easier. Algorithm state $D(:,k) = \frac{1}{NGroups} - .5 * (sum(A2,2) + logDetSigma)$ was the representation of equation (1).

IV. Fixed-point Implementation of Functional Units

After being broken down into several functional units, algortihm of selected functional units will be converted to fixed-point. Therefore, conversion will be performed locally for every functional unit. For every functional unit being converted, conversion steps will be performed as follows.

1) Range determination of every variables involved, which also determine integer length(IL), and
2) Fractional length(FL) determination.

Range or IL determination is performed by analyzing all processing variables during the runtime. Variables being analyzed are variables whose types are floating-point and being processed by arithmetics operation during runtime. Thus, helping variables such as counter in the loop or status variables are not considered as variables being analyzed. Meanwhile, FL is related with quantization error caused by conversion from floating-point to fixed-point. Longer FL makes quantization error getting smaller, but also increases the hardware cost. Optimization technique is performed to determine FL in order to minimize hardware cost, with quantization error should not above predefined maximum quantization error. The optimization problem can be stated as

$$min(C(\boldsymbol{f})) \text{ subject to } \epsilon(\boldsymbol{f}) \geq \epsilon_{max} \quad (6)$$

$C(\boldsymbol{f})$ is representation of total hardware cost measured in the system as a function of fraction-length vector $\boldsymbol{f}$. Vector $\boldsymbol{f}$ consists of fraction lengths of all analyzed variables in the algorithm. An algorithm with *m* fixed-point variable being analyzed has a fraction-length vector with length *m*. In this paper, that problem will be defined as two-objective problem and will be solved by multi-objective GA.

A. Integer Length Determination

Determination of variables range will result in integer length suitable for the variables and sign bit assigned. The main idea of this process is to find out the maximum and minimum value achieved by every variable in the algorithm during the computation. Maximum absolute value obtained by a variable will determine its integer length. As an algorithm, range determination can be stated in Algorithm 3 as follows.

Algorithm 4: RangeDet
Compute integer length of all variables
*Output: **il**(integer length vector), **s**(sign vector)*

```
initialization
for i = 1 : NumberOfVar do
    RangeMinMax_i = [1, 0]
    s_i = 0
end for
while AnalyzedAlgorithm running do
    for i = 1 : NumberOfVar do
        if var_i < RangeMinMax_i(1) then
            RangeMinMax_i(1) = var_i;
        else if var_i > RangeMinMax_i(2) then
            RangeMinMax_i(2) = var_i;
```

```
12:        end if
13:        if  var_i < 0 then
14:            s_i = 1;
15:        end if
16:    end for
17:    for i = 1 : NumberOfVar do
18:        MaxAbs_i = Max(Abs(RangeMinMax_i));
19:        il_i = log_2(MaxAbs_i)
20:    end for
21: end while
```

In algorithm 3, sign bit is determined by observing variable condition. In this research, 0 for positive sign bit and 1 for negative will be used. Once a variable get less then zero condition, the sign bit will be negative, which is shown in Algorithm 4 line 13. Range of variable determination is performed by computing maximum and minimum value during the runtime. It is shown in line 8-12 Algorithm 4. At the end of analyzed algorithm runtime, variable range will be obtained in the format $[min, max]$. Signed variable will have negative number in its min value. To obtain integer length, absolute value of the range is first computed, then the greater of two value will be selected as $MaxAbs_i$. Finally, integer length of i_{th} variable will be calculated by performing log_2 of $MaxAbs_i$.

B. Objective functions

Optimum fraction length should be obtained in order to satisfy equation (6). There are two objectives should be achieved, minimum quantization error and minimum hardware cost. Therefore, two objective functions are defined, one computes quantization error, and the other computes hardware cost. This research computes objectives by simulation, which means calculation of error and cost are inserted in the algorithm being analyzed as instrumentation code. The computation method of error and cost representation will be described as follows.

1) Quantization error representation: Quantization error is obtained after both fixed-point and floating-point algorithm execution. From fixed-point algorithm output ($output_{fixed}(\boldsymbol{f})$) and floating-point output ($output_{float}$) error difference is computed as follows

$$e(\boldsymbol{f}) = output_{fixed}(\boldsymbol{f}) - output_{float} \tag{7}$$

To represent error in optimization purpose, ratio between norm-2 of error difference and floating-point output is used.

$$\epsilon(\boldsymbol{f}) = \|e(\boldsymbol{f})\| / \|output_{float}\| \tag{8}$$

In the next discussion, $\epsilon(\boldsymbol{f})$ will be mentioned as error power.

2) Hardware cost representation: Cost models of every operator are required in hardware cost measurement. In this research, hardware cost representations from Constantinides,et.al [8], [9] for register, adder and multiplier then Lam [24] for division are used. Model for subtraction and division are the same with addition. For square-root operation, fixed-point sqrt algorithm by Turkowski [30] are used. Meanwhile, cost model sqrt realization from its algorithm utilizes previous hardware cost models for add, mul and div. Equation (9), (10), and (11) show hardware cost estimation function formulated by Constantinides, et.al [8], [9]. Measurement unit for hardware cost is in Logic Cells (LCs) Subscripts are *o*=output, *a*=first operand, and *b*=second operand. Meanwhile, variables are *c*=cost, *w*=WL, *i*=IL, and *f*=FL.

Hardware cost estimation for register is,

$$c = w + 1 \tag{9}$$

hardware cost estimation for adder,

$$c = \begin{cases} m_{av} + 2, & \text{if } (w_o + m_v) > (m_a + 1) \\ (w_o + 1) & \\ \quad + 0.5(m_{av} - w_o + 1) & \text{otherwise} \end{cases} \tag{10}$$

with

$f_v = (f_a - f_b)$,
$m_a = max(w_a - f_v, w_b)$,
$m_v = max(i_a, i_b) - i_o$, and
$m_{av} = (m_a - m_v)$

and hardware cost estimation for multiplier,

$$c = 0.6 w_b (w_a + 1) + 0.85(w_a + w_b - w_o) \tag{11}$$

C. Fraction length determination

1) Multi-objective GA as wordlength search engine: In this research, we utilize multi-objective GA as search engine for fraction length determination, with two objectives. The two objective functions are error power and hardware cost. Multi-objective GA is used to find a set of fraction length vectors which provide a set of minimum errors and costs.

As shown in Figure 3, GA operation for wordlength optimization is described as follows.

1) Initial population is generated randomly. Size of a chromosome, n is equal to the number of variables being converted to fixed-point in analyzed algorithm. A chromosome will be a n-byte string consists of integer from 1(minimum length) to 32(maximum length), as shown in Figure 4. This byte string will act as fraction length of fixed-point variables used in the algorithm. For the purpose of FL determination, we set the population size for the chromosome at 90 and the number of generation at 150.
2) Chromosome with high fitness value will be selected as parents for the next generations population. Concept used is elitism and roulette wheel selection.
3) Crossover and mutation are operation of GA. In crossover, child or new chromosome is produced by combining 2 parent chromosomes at a random crossover point. Mutation is a process of changing the value of allele byte in a chromosome. Mutation process is controlled by predefined mutation rate.
4) GA will continue until either one of two stopping criteria is met. The first criterion is when the number of generation set has been reached and another criterion is when all population becomes Pareto-optimal.

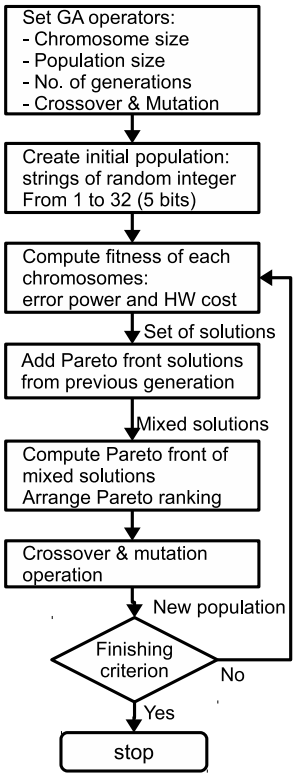

Fig. 3. GA flowchart for finding optimized wordlength

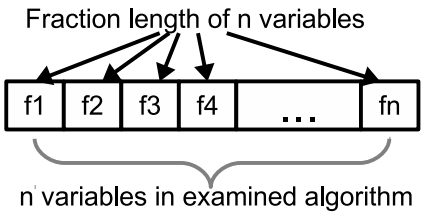

Fig. 4. GA flowchart for finding optimized wordlength

2) Pareto ranking method for parents selection: In multi-objective GA, it is quite difficult to combine the objectives linearly or nonlinearly to obtain the fitness of chromosome [31]. Whereas, those fitnesses are required in making chromosome rank to perform parents selection. Hence, this paper will adopt Pareto multi-objective ranking approach described in equation (3) and (4) by Popov [20], Nezhad, et.al [21] and Konak, et.al [22], which is called non-dominated sorting method.

For example, individual l_i at generation t is dominated by $n(t)$ individuals in the current population. Individual xi will be assigned a rank in its current population at generation t in following equation.

$$rank(l_i, t) = 1 + n(t) \tag{12}$$

All non-dominated individuals previously were assigned Rank 1. To assign fitness from the rank, ranks are recalculated, so the fitness of the individuals on the Pareto-optimum position gets maximum, and roullette wheel method can be implemented.

3) Best Individual Selection from Pareto Front: With Pareto front gives us several points to be selected as best point, then a convention should be defined to select which the best point is. Popov [20], in his Matlab toolbox select the best point from its Euclidean distance to reference point. However, according to the wordlength optimization rule in equation (6), in this research the best point is the point with minimum cost and its error power does not exceed defined maximum error power. Then, the rule is defined that in every generation n, minimum cost is decided as follows,

$$C(n)_{min} = min(\boldsymbol{C}(n)) \text{ with } \boldsymbol{\epsilon}(n) \leq \epsilon_{max} \tag{13}$$

V. Results

In this section, we present wordlength optimization results of every parts in LDA algorithm. We compare multi-objective GA (MOGA) with Pareto ranking method used in this research to several existing method, such as:

1) Sequential search [14]–[17], which is gradient based method and basically single objective. In this method, two objectives are combined using weighting constant (α), shown in equation (14)

$$obj = (1 - \alpha)\epsilon + \alpha \cdot cost \tag{14}$$

   Weighting constant are selected to get equivalent order between error power and hardware cost, which are very much different. For HQR unit, $\alpha = 10^{-4}$, for INV and LDA main loop unit, $\alpha = 10^{-6}$
2) Combined-objective which is essentially single-objective GA (SOGA) [10], [11], which is using same fashion as above to obtain single objective from two objectives.
3) Conversion by Matlab fixed-point toolbox which is automated conversion from floating-point to fixed point for any given function.

Performance of observed methods is examined mainly on their error power and hardware cost results. Beside that, MOGA and SOGA convergences comparison in searching optimum wordlength is also presented, in order to show the effectiveness comparison of two types of GA. Finally, according to the main objective of this research, which is LDA classifier implementation in fixed-point, results of classifier accuracy using fixed-point are also presented.

Data being used in this section are obtained from wood features database, which are saved in Basic Grey Level Aura Matrix (BGLAM) format [4]. Experiment data input covers:

1) Normalized training data
2) Group means of training data. It contains averages of every group in training data, after it was being divided into 52 groups.
3) Sample data to test training process

A. HQR and InverseTriu wordlength optimization

We decide to separate discussion about HQR and Cholesky inversion implementation and about LDA main loop. It is because LDA main loop is directly related with system output and considered to be discussed in different approach.

TABLE II
ERROR POWER AND HARDWARE COST OF HQR AND CHOLESKY TRIANGULAR INVERSE WITH SEVERAL METHODS AND ϵ_{max}

Method	ϵ_{max}	HQR		Triu Inv	
		ϵ	cost	ϵ	cost
MOGA	0.01	$6.9 \cdot 10^{-4}$	$4.93 \cdot 10^{4}$	$2.3 \cdot 10^{-3}$	$1.92 \cdot 10^{6}$
	0.005	$6.3 \cdot 10^{-4}$	$4.93 \cdot 10^{4}$	$2.3 \cdot 10^{-3}$	$1.92 \cdot 10^{6}$
	0.001	$6 \cdot 10^{-4}$	$4.96 \cdot 10^{4}$	$8.4 \cdot 10^{-4}$	$2.25 \cdot 10^{6}$
SS	0.01	$5.6 \cdot 10^{-4}$	$4.96 \cdot 10^{4}$	$4.1 \cdot 10^{-3}$	$2.49 \cdot 10^{6}$
	0.005	$2.7 \cdot 10^{-4}$	$5.19 \cdot 10^{4}$	$4.2 \cdot 10^{-3}$	$2.85 \cdot 10^{6}$
	0.001	$8 \cdot 10^{-4}$	$5.41 \cdot 10^{4}$	$8 \cdot 10^{-4}$	$2.87 \cdot 10^{6}$
SOGA	-	$1.1 \cdot 10^{-3}$	$6.4 \cdot 10^{4}$	$1.3 \cdot 10^{-3}$	$2.82 \cdot 10^{6}$
FXTB	-	$4 \cdot 10^{-2}$	$9.61 \cdot 10^{4}$	$1.3 \cdot 10^{-3}$	$5.7 \cdot 10^{6}$

MOGA =multi-objective GA, SOGA=single-objective GA, SS=sequential search, FXTB=Matlab Fixed Point Toolbox

Inputs for HQR and InverseTriu algorithm are normalized training data and R matrix output from HQR respectively. Table II provides error powers (ϵ) and hardware cost as measurements of wordlength optimization process. Hardware cost measurement unit is in Logic Cells (LCs) unit. Beside being compared with several optimization methods previously mentioned, optimization is performed for a few maximum error powers limit (ϵ_{max}) if possible, such as in SS and MOGA methods.

Due to extensive matrices operations and the algorithms are not modified, hardware cost quantities used in this research are very huge. For those two algorithms, blocked QRD and Cholesky inversion of triangular matrix (InverseTriu), hardware costs are in orders of 10^4 and 10^6 respectively. According to Table II, multiobjective GA performs better than another method in terms of hardware cost. With the same ϵ_{max}, MOGA produces less hardware cost than sequential search (SS). Even though error power of some results are still greater than SS, those conditions still satisfy the requirement that error power should less than maximum error power shown in equation (13).

B. Result Evolution

Not only produces lower costs, MOGA also results in more stable cost for various setting of ϵ_{max}. With no option to set maximum error power available in single-objective GA and Matlab fixed-point toolbox automated conversion, results of both of them are still below MOGA's result.

In its evolution process for solving wordlength optimization, MOGA is also more convergent than SOGA. In this case, evolution process of HQR wordlength optimization using MOGA and SOGA are compared. Hardware cost and error power evolution in optimizing HQR wordlength are shown in Figure 5 and Figure 6 respectively. According to Figure 5, proposed MOGA with individual selection from Pareto front has better trend, which produces better individual in next several generations. Otherwise, SOGA produces unstable evolution process for solving wordlength optimization for HQR, hence in the next discussion, SOGA will not be included. In some points, next generations in SOGA give worse result than its previous. From Figure 6, it is shown that in its final generations, MOGA can satisfy below maximum error power requirement, which is 0.001 in this case, even though SOGA can achieve lower error power.

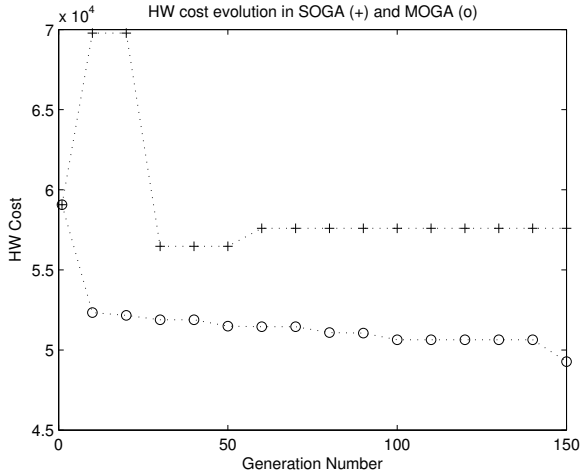

Fig. 5. Evolution of HW cost in HQR wordlength optimization with MOGA and SOGA

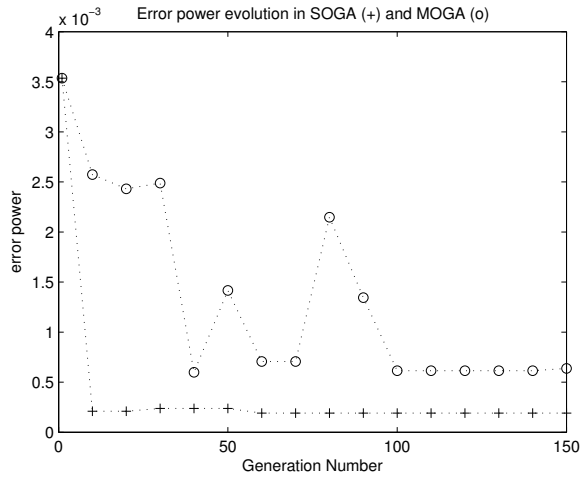

Fig. 6. Evolution of error power in HQR wordlength optimization with MOGA and SOGA

Pareto front visualizations of HQR wordlength optimization process are shown in Figure 7. Figure 7 shows Pareto front for 50th(a), 100th(b), and last generation(c). With the number of population is kept constant, the number of solutions belong to Pareto front(non-dominated) will tend to increase following the number of generations. Hence, following the generation increment, non-dominated solution will be dominating entire population. From quality of results point of view, non-dominated solutions in the 100th generation is better than the 50th and the 150th better than the 100th, seen from lower HW cost and error power produced.

C. Wordlength optimization result of LDA main loop

Input for this module are taken from outputs of triangular matrix inversion parts, normalized sample for testing, and group means of training data. Logarithm of R matrix determinant, which is devised to be computed in software, also becomes input for LDA main loop. Output of this algorithm is D matrix, which determines class estimation of the given

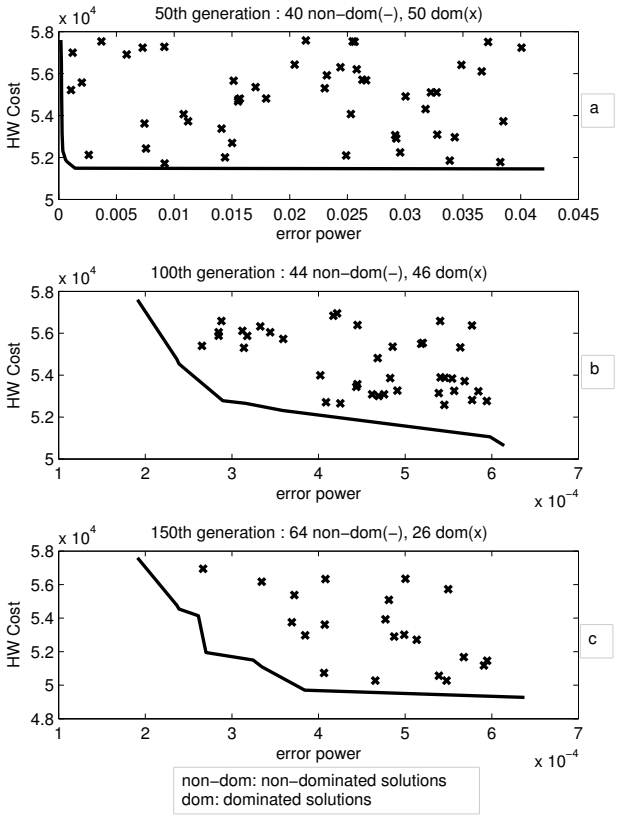

Fig. 7. Pareto front visualization of the 50th(a), 100th(b), and 150th generation(c).

samples. Error power is measured from D matrix. However, in this paper, classification accuracy of entire fixed-point system is also presented because this module directly connected to system's main output. To obtain the achieved accuracy of entire system, we tested system by giving selected training data to HQR module. In wood recognition system software working scheme, training data was selected by several masking words [4]. Every masking word produced different classification accuracies and after searching process in wood recognition training, an optimum masking word was obtained to get best accuracy.

In this experiment, we use five combinations of masking word. One masking word is the optimum word, which gave 95 % accuracy in existing system [4] and the remaining masking words are non optimum. The objectives of this experiment is to compare classification accuracies obtained by proposed method with another existing methods. Produced hardware costs, as a consequence of high computation accuracy are also presented. The expectation of this experiment is to get the best accuracy in a minimum hardware cost. Result of worlength optimization for LDA main loop and its classification accuracies is presented in Table III.

Data being processed as input for LDA main loop module is output from triangular matrix inverse module with $\epsilon_{max} = 0.001$. According to Table III, for lower error power specification, MOGA can perform better, which produces less

TABLE III
CLASSIFICATION ACCURACY FOR SEVERAL DATA COMBINATIONS (FOUR NON-OPTIMUM AND ONE OPTIMUM), IMPLEMENTED WITH SEVERAL METHODS AND ϵ_{max} IN LDA LOOP STAGE

Method	ϵ_{max}	cost ($\times 10^6$)	acc_1	acc_2	acc_3	acc_4	acc_{opt}
FLP	-	-	81.07	84.46	87.32	81.96	95
MOGA	0.005	4.91	79.21	81.76	84.28	79.64	31.61
	0.001	5.36	81.07	84.46	87.14	81.43	55.35
	10^{-7}	11.6	81.07	84.46	87.32	81.96	93.57
	10^{-8}	13.5	81.07	84.46	87.32	81.96	95
SS	0.005	4.79	76.61	72.68	75.54	79.64	13.75
	0.001	5.3	81.07	84.29	81.25	81.25	50.89
	10^{-7}	12.1	81.07	84.46	87.32	81.96	91.43
	10^{-8}	14.9	81.07	84.46	87.32	81.96	95
FXTB	-	32.67	81.07	84.46	87.32	81.96	94.8

Inputs are taken from INV output with $\epsilon_{max} = 0.001$.
ϵ_{max} was measured from error power of variable D in Algorithm 3.
Cost was measured from LDA main loop only.
acc_n is accuracy achieved with training data masked by n_{th} mask word
acc_{opt} is accuracy achieved with training data masked by optimum mask
FLP is accuracy achieved by floating point system as reference

hardware cost than another method with same classification accuracy. However, in optimizing wordlength for LDA main loop, very low error power specification is required to obtain classification accuracy. From last column in Table III, it is shown that for training data being masked by optimum word [4], classification accuracy (acc_{opt}) achieved by fixed-point system cannot reach expected accuracy (95 %) if error power specification is below 10^{-8}. For training data being masked by non-optimum words, shown in column 4-7 Table III, expected accuracy can be obtained in $\epsilon_{max} = 0.001$. Therefore, to ensure the accuracy of fixed-point LDA classifier, ϵ_{max} specification for LDA main loop module should be at least 10^{-8} or above, while ϵ_{max} for previous module (HQR and inverse) can be kept at 0.001. With ϵ_{max} for HQR and inverse are in 0.001 and LDA main loop module in 10^{-8} total cost of entire system obtained by wordlength optimization are shown in Table IV. However, MOGA-based wordlength optimization still produce lowest hardware cost. Table V shows wordlength and fraction length results for all modules implemented in fixed-point. That results also use ϵ_{max} mentioned above. In wordlength results as shown in Table V, all obtained wordlengths do not exceed 32 bit, which means devised system in the future can be integrated with 32 bit embedded processor.

TABLE IV
TOTAL COST

Method	HQR ($\times 10^4$) $\epsilon_{max} = 10^{-3}$	INV ($\times 10^6$) $\epsilon_{max} = 10^{-3}$	LDA loop ($\times 10^6$) $\epsilon_{max} = 10^{-8}$	total ($\times 10^6$)
MOGA	4.96	2.25	13.5	15.8
SS	5.41	2.49	14.9	17.4
FXTB	9.61	5.7	3.27	38.5

VI. CONCLUSIONS

In this paper, a simulation approach to finding optimum wordlength combinations for fixed-point LDA classifier in wood recognition system was developed. By dividing entire LDA into computation modules, then performing range determination before determine the fraction length, optimization

TABLE V
OPTIMIZED WL AND FL IN ALL MODULES WITH MOGA METHOD

HQR			Inv			LDA loop		
Var	FL	WL	Var	FL	WL	Var	FL	WL
W	18	20	R	16	20	A	14	22
Y	27	30	X	5	27	A2	14	27
X	17	18	X2	8	25	D	9	23
d1	27	27	Y	6	27	Gmeans	14	16
d2	16	16	Ri	8	32	Ri	13	23
v	16	2				Sample	14	17
d3	17	17				lds	13	21
y	17	18				lprior	12	24
d4	16	17						
z	26	29						
yv	24	27						
R	25	32						

process can be simplified. Computation modules required to be implemented in fixed-point are QR decomposition, triangular matrix inverse (INV), and main loop of LDA.

By using optimization rules, finding minimum hardware cost with error power must not exceed maximum specification, wordlength optimization (WLO) can be formulated to two-objective problem. Proposed method had been compared with another method such as sequential search(SS), single-objective GA(SOGA), and automated conversion by Matlab fixed-point toolbox (FXTB). The two-objective GA used Pareto ranking to select non-dominated solutions in entire population, then among non-dominated solutions, a best solution which has lowest hardware cost and satisfies maximum error specification will be selected. The method has more convergence than single-objective GA with combined objective between error and cost. This method also performs better than another methods (SS,SOGA, and FXTB) in finding wordlength with lowest hardware cost and constrained error power. This WLO method also result proper set of wordlengths, thus it can be integrated to 32 bit system.

LDA main loop module needs more restricted error power specification than QRD and INV, especially for optimized training matrix. Error power specification of 0.001 is sufficient for QRD and INV, but LDA needs at least 10^{-8}. Combination of modules with different error power specification was possible and had produced same classification accuracy as floating point system.

VII. FUTURE WORKS

Based on the results of this work, there are many ways in which the work presented in this paper could be expanded in designing large scale fixed-point system, especially LDA classifier. In this section some of the possibilities will be expanded upon.

Involving large matrix computation, a module will need huge hardware cost for its registers and operations without redesigning its algorithm. Further studies are required to develop algorithm of the modules which can reduce the number of its operations with acceptable time constraints and latencies.

An matrix inversion method called generalized inverse, can perform pseudo-inversion for rank-deficient matrix, which cannot be handled by ordinary inversion. Required study in this point is about to find generalized inverse algorithm which is feasible to be implemented in fixed-point system.

Genetic algorithm is basically time consuming, especially for this research, which used 150 generations and 90 populations with 5 bit integer chromosome. A method should be implemented to reduce the number of population and generation without reduce the quality of algorithm results. The example method for that purpose is performing search for minimum and maximum fraction length within particular constraint before running GA. Thus, GA searching area will be bounded and require less number of population.

ACKNOWLEDGMENT

The authors would like to thank Ministry of Education Malaysia and Universiti Teknologi Malaysia for funding this research project through a research grant (04H40) titled "Dimension Reduction and Data Clustering for High Dimensional and Large Datasets".

REFERENCES

[1] Khalid, M., Lew, Y., Yusof, R., and Nadaraj, M. Design of An Intelligent Wood Species Recognition System. *IJSSST*, 2008.

[2] Rosli, N. R., Khalid, M. and Yusof, R. Wood Species Recognition based on Gabor Filter Image Processing Technique. *Proceedings of 2008 Student Conference on Research and Development*, 2008.

[3] Cordova, F. *Wood Recognition*. Technical report. Centre for Artificial Intelligence and Robotics (CAIRO), Universiti Teknologi Malaysia. 2009.

[4] Khairuddin, U., Yusof, R., Khalid, M. and Cordova, F. Optimized Feature Selection for Improved Tropical Wood Species Recognition System. *ICIC Express letters, Part B: Applications, An International Journal of Research and Surveys, 441-446*, 2011.

[5] Saeed, A., Elbably, M., Abdelfadeel, G. and Eladawy, M. I. Efficient FPGA implementation of FFT/IFFT Processor. *NAUN International Journal Of Circuits, Systems And Signal Processing*, 2009. 3(3): 103–111.

[6] Rocher, R., Menard, D., Herve, N. and Sentieys, O. Fixed-point configurable hardware components. *EURASIP Journal on Embedded Systems, no. 1*, 2006.

[7] Lee, D.-U., Gaffar, A. A., Cheung, R. C., Mencer, O., Luk, W. and Constantinides, G. A. Accuracy-guaranteed bit-width optimization. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 25, no. 10 : 1990-2000*, 2006.

[8] Constantinides, G. A., Cheung, P. Y. and Luk, W. Wordlength optimization for linear digital signal processing . *IEEE Trans. Computer-Aided Design , vol. 22, no. 10, pp. 1432-1442, Oct*, 2003.

[9] Constantinides, G. A., Cheung, P. Y. and Luk, W. Optimum wordlength allocation. *in Proc. of the 10th Annual IEEE Sym. on Field-Programmable Custom Computing Machines pp. 219 - 228*, 2002.

[10] Ahmadi, A. and Zwolinski, M. Word-length oriented multiobjective optimization of area and power consumption in dsp algorithm implementation. *In Microelectronics, 2006 25th International Conference on, pp. 614-617. IEEE*, 2006.

[11] Sulaiman, N. and Arslan, T. A multi-objective genetic algorithm for on-chip real-time optimisation of word length and power consumption in a pipelined FFT processor targeting a MC-CDMA receiver. *In Evolvable Hardware, 2005. Proceedings. 2005 NASA/DoD Conference on, pp. 154-159. IEEE*, 2005.

[12] Zhang, L., Zhang, Y. and Zhou, W. Floating-point to fixed-point transformation using extreme value theory. *In Computer and Information Science, 2009. ICIS 2009. Eighth IEEE/ACIS International Conference on, pp. 271-276*, 2009.

[13] Pradhan, T., Kabi, B., Routray, A. and Anirudh, G. Fixed-Point Hestenes SVD Algorithm for Computing Eigen Faces. *NAUN International Journal Of Circuits, Systems And Signal Processing*, 2013. 7(6): 312–322.

[14] Han, K., Eo, I., Kim, K. and Cho, H. Numerical word-length optimization for CDMA demodulator. *In Circuits and Systems, ISCAS 2001. The 2001 IEEE International Symposium on, vol. 4, pp. 290-293*, 2001.

[15] Han, K. and Evans, B. L. Optimum wordlength search using sensitivity information. *EURASIP Journal on Applied Signal Processing*, 2006.
[16] Roy, S. and Banerjee, P. An algorithm for converting floating-point computations to fixed-point in MATLAB based FPGA design. *Proceedings of the 41st annual Design Automation Conference. ACM,*, 2004.
[17] Banerjee, P., Bagchi, D., Haldar, M., Nayak, A., Kim, V. and Uribe, R. Automatic conversion of floating point MATLAB programs into fixed point FPGA based hardware design. *In Field-Programmable Custom Computing Machines, FCCM 2003. 11th Annual IEEE Symposium on*, 2003.
[18] Bohling, G. *Classical Normal-Based Discriminant Analysis.* Technical report. Kansas Geological Survey. 2006.
[19] Ye, J. and Li, Q. A Two-Stage Linear Discriminant Analysis via QR-Decomposition. *IEEE Transactions On Pattern Analysis And Machine Intelligence, 27: 929 - 942*, 2005.
[20] Popov, A. *Genetic Algorithms For Optimization*. Technical report. Programs for MATLAB Version 1.0 User Manual. 2005.
[21] Nezhad and Sanati, A. Multi objective optimization of part orientation in stereolithography. In: Rudas, I.; Mastorakis, N. (ed.). *WSEAS International Conference. Proceedings. Mathematics and Computers in Science and Engineering. WSEAS*, 2009.
[22] Konak, A., Coit, D. W. and Smith, A. E. Multi-objective optimization using genetic algorithms: A tutorial. *Reliability Engineering and System Safety 91, no. 9, 992-1007.*, 2006.
[23] Rodellar, A., Muoz, V., lvarez, A., Martinez, E., Gonzlez, C., Gmez, P. and de MontegancedoBoadilla de Monte, C. Word-length optimization of an Adaptive Noise Canceller. *In Proc., 5th WSEAS International Conference on Signal Processing*, 2006.
[24] Ming, L. Y. *An Optimization Framework for Fixed-point Digital Signal Processing*. Ph.D. Thesis. The Chinese University of Hong Kong. 2003.
[25] Chen, J., Zhang, Y. and Wang, X. A New Finite Word-length Optimization Method Design for LDPC Decoder. *WSEAS Transactions on Communications 12, no. 4,*, 2013.
[26] Trefethen and Lloyd. Householder triangularization of a quasimatrix. *IMA journal of numerical analysis 30.4 : 887-897.*, 2010.
[27] Rafique, A., Kapre, N. and Constantinides, G. A. Enhancing performance of Tall-Skinny QR factorization using FPGAs. *Field Programmable Logic and Applications (FPL), 2012 22nd International Conference on. IEEE*, 2012.
[28] Bouwmeester, H., Jacquelin, M., Langou, J. and Robert, Y. Tiled QR factorization algorithms. *In High Performance Computing, Networking, Storage and Analysis (SC), 2011 International Conference for, pp. 1-11. IEEE*, 2011.
[29] Burian, A., Takala, J. and Ylinen, M. A fixed-point implementation of matrix inversion using Cholesky decomposition. *Circuits and Systems, 2003 IEEE 46th Midwest Symposium on*, 2003.
[30] Turkowski, K. *Fixed Point Square Root*. Technical report. Apple Technical Report No. 96. 1994.
[31] Camelo, M., Donoso, Y. and Castro, H. MAGS An Approach Using Multi-Objective Evolutionary Algorithms for Grid Task Scheduling. *NAUN International Journal Of Applied Mathematics And Informatics*, 2011. 5(2): 117–127.

Enas Dhuhri Kusuma is Currently pursuing Ph.D in Malaysia-Japan International Institute of Technology, UTM. Obtained B.Eng. (Sarjana Teknik) Electrical from Universitas Gadjah Mada (UGM), Indonesia in 2005, and M.Eng Electrical from UGM 2010. Possess 3 years experience in IT industry as embedded system engineer.

Rubiyah Yusof is a Professor at the Centre of Artificial Intelligence and Robotics (CAIRO), Universiti Teknologi Malaysia, Malaysia (UTM). She is currently the Dean of the Malaysia-Japan International Institute of Technology, UTM. Her interests include adaptive control, system identification, biometric systems and ICT. She also is a co-author of the book entitled Neuro Control and its Applications published by Springer, United Kingdom.

Mohd Fauzi Othman is currently a senior lecturer at the Faculty of Electrical Engineering, Universiti Teknologi Malaysia (UTM). He obtained his Bachelor of Mathematic (B.Math) in Applied Math from University of Wollongong, Australia in 1990 and Master of Electrical Engineering in Electrical Power System from UTM in 1996. He received his PhD in Control System from the University of Sheffield in 2004 . Dr Mohd Fauzi is a co-author of a book chapter entitled "Fault Diagnosis in Power Distribution Network Using Adaptive Neuro-Fuzzy Inference System (ANFIS)" of the book "Fuzzy Inference System - Theory and Applications" published by InTech. He is a member of Institute of Electrical and Electronics Engineers Malaysia (IEEE) and Society of Industrial and Applied Mathematics (SIAM).