Applied Soft Computing 9 (2009) 541–552

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

No-reference image quality assessment using modified extreme learning machine classifier

S. Suresh [a], R. Venkatesh Babu [b,*], H.J. Kim [c]

[a] School of Electrical and Electronics Engineering, Nanyang Technological University, Singapore
[b] Department of Electrical Engineering, Indian Institute of Science, Bangalore, India
[c] CIST, Korea University, Seoul, Republic of Korea

ARTICLE INFO

Article history:
Received 19 May 2007
Received in revised form 18 December 2007
Accepted 27 July 2008
Available online 14 August 2008

Keywords:
Image quality assessment
No-reference metric
Blockiness measurement
Neural network
JPEG
Evolutionary algorithms
Extreme learning machine

ABSTRACT

In this paper, we present a machine learning approach to measure the visual quality of JPEG-coded images. The features for predicting the perceived image quality are extracted by considering key human visual sensitivity (HVS) factors such as edge amplitude, edge length, background activity and background luminance. Image quality assessment involves estimating the functional relationship between HVS features and subjective test scores. The quality of the compressed images are obtained without referring to their original images ('No Reference' metric). Here, the problem of quality estimation is transformed to a classification problem and solved using extreme learning machine (ELM) algorithm. In ELM, the input weights and the bias values are randomly chosen and the output weights are analytically calculated. The generalization performance of the ELM algorithm for classification problems with imbalance in the number of samples per quality class depends critically on the input weights and the bias values. Hence, we propose two schemes, namely the k-fold selection scheme (KS-ELM) and the real-coded genetic algorithm (RCGA-ELM) to select the input weights and the bias values such that the generalization performance of the classifier is a maximum. Results indicate that the proposed schemes significantly improve the performance of ELM classifier under imbalance condition for image quality assessment. The experimental results prove that the estimated visual quality of the proposed RCGA-ELM emulates the mean opinion score very well. The experimental results are compared with the existing JPEG no-reference image quality metric and full-reference structural similarity image quality metric.

1. Introduction

The main objective of image/video quality assessment metrics is to evaluate the visual quality of a compressed image/video with/without referring to their original form. It is imperative that these measures exhibit good correlation with perception by the human visual system (HVS). The most widely used objective image quality metrics, namely, the mean square error (MSE) and the peak signal-to-noise ratio (PSNR), as widely observed, do not correlate well with human perception [1] besides requiring the original reference image to compute distortion. Most images on the Internet and in multimedia databases are only available in compressed form, and hence inaccessibility of the original reference image makes it difficult to measure the image quality. Therefore, there is an unquestionable need to develop metrics that closely correlate with human perception without needing the reference image.

Considerable volume of research has gone into developing objective image/video quality metrics that incorporate the perceived quality measurement with due consideration for HVS characteristics. However, most of the proposed metrics based on HVS characteristics require the original image as a reference [2–5]. Though it is easy to assess the image quality without any reference by manual observations, developing a no-reference (NR) quality metric is a difficult task. To develop NR metrics, it is essential to have a priori knowledge about the nature of artifacts. Currently, NR quality metrics are the subject of considerable attention by the research community, visibly so, with the emergence of video quality experts group (VQEG) [6] which is in the process of standardizing NR and reduced-reference (RR) video quality assessment methods.

The most popular and widely used image format in the Internet as well as in digital cameras happens to be JPEG [7]. Since JPEG uses block-based DCT transform for coding to achieve compression, the

[*] Corresponding author.
E-mail addresses: sundaram.suresh@hotmail.com (S. Suresh), venkatesh.babu@gmail.com (R. Venkatesh Babu), khj-@korea.ac.kr (H.J. Kim).

1568-4946/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.asoc.2008.07.005

major artifact that JPEG-compressed images suffer is blockiness. The compression rate (bit-rate) and image quality are mainly determined by the degree of quantization of these DCT coefficients. The undesirable consequences of quantization manifest as blockiness, ringing and blurring artifacts in the JPEG-coded image. It turns out that the subjective data for all these artifacts are highly correlated [8]. Hence, measuring the blockiness with reference to HVS criteria in turn indicates the image quality. Since, the image quality is a subjective phenomenon, the manual inspection plays an important role. The subjective test is concerned with how an image is perceived by a viewer and designates his/her opinion on a particular image (opinion score). The mean opinion score (MOS) provides average opinion score over all subjects. Here, the objective is to find the functional relationship between the extracted HVS features and MOS for quantifying the quality of the image.

Existing algorithms to measure the blockiness have used a variety of methods to do so. Wang and Bovik proposed an algorithm based on computing the FFT along the rows and columns to estimate the strength of the block edges of the image [9]. Further, they proposed a nonlinear-model for NR quality assessment of JPEG images, where the parameters of the model were determined with subjective test data [10]. Vlachos used cross-correlation of subsampled images to compute a blockiness metric [11]. Wu and Yuen proposed a metric based on computing gradients along block boundaries while tempering the result with a weighting function based on the HVS [3]. Here, the block edge strength for each frame was computed. Similar ideas about the HVS were utilized by Suthaharan [4] and Gao et al. [5]. The general idea behind these metrics was to temper the block edge gradient with the masking activity measured around it. These approaches utilize the fact that the gradient at a block edge can be masked by more spatially active areas around it, or in regions of extremities in illumination (very dark or bright regions) [12]. Jung et al. [13] proposed an NR metric for emulating the full-reference metric proposed by Karunasekera and Kingsbury [2] using neural network. Here they have used general image features for training the neural network and the results are not compared against the subjective test scores. On the other hand, recently, Gastaldo et al. [14,15] proposed a circular back propagation (CBP)-based image quality evaluation method using the general pixel-based image features such as higher order moments without considering the HVS. In all these above mentioned approaches, extracting large number of general image features are computationally quite complex for real-time implementation. Also, the functional relation between the HVS features and the MOS are nonlinear and is difficult to mathematically model. Under these circumstances, neural networks are best suited for solving such problems.

In the last few decades, extensive research has been carried out in developing the theory and the application of artificial neural networks (ANNs). ANNs possess an inherent structure suitable for mapping complex characteristics, learning and optimization have emerged to be a powerful mathematical tool for solving various practical problems like pattern classification and recognition, medical imaging, speech recognition and control [16–19]. Furthermore, from a practical perspective, the massive parallelism and fast adaptability of neural network implementations provide more incentives for further investigation in problems involving complex mapping with uncertainties. Of the many neural network architectures proposed, single layer feedforward network (SLFN) with sigmoidal or radial basis function are found to be effective for solving a number of real world problems. The free parameters of the network are learned from given training samples using gradient descent algorithms. The gradient descent algorithms are relatively slow and have many issues in error convergence.

Recently, it is shown that the SLFN network with randomly chosen input weights and hidden bias values can approximate any continuous function to any desirable accuracy [20]. Here, the output weights are analytically calculated using the Moore-Penrose generalized pseudo-inverse [21]. Since, the learning algorithm is faster and has a good generalization ability, it is called 'extreme learning machine' (ELM). The ELM algorithm overcomes many issues in traditional gradient algorithms such as stopping criterion, learning rate, number of epochs and local minima. In fact, the performance of the ELM algorithm on many real-world problems have been compared with the other neural network approaches [22] and its performance has been found to be better.

In this paper, we present image quality estimation using ELM algorithm. In general, the quality estimation problem is the process of finding the functional relationship between the MOS values and the feature inputs. But, the MOS values depend on the number of opinion scores per image. If the number of opinions available for a given image is low and statistically different, then it will affect performance of the image quality estimator. Hence, in this study, the problem is circumvented by converting the MOS values to the quality class. The functional relationship between the HVS features and the quality class label is approximated using the ELM classifier. The image quality metric is calculated using the predicted class label and the posterior probability.

Here, the quality classification problem has fewer training samples per class and high imbalance in number of samples per class. In such cases, the generalization performance of the ELM algorithm depends on the proper selection of the input weights and hidden bias values (fixed parameters). Also, the number of hidden neurons affects the generalization performances. Hence, in this paper, we present k-fold cross-validation (KS-ELM) and real-coded genetic algorithm (RCGA-ELM) approaches to select appropriate values for the free parameters in extreme learning machine classifier. In the RCGA-ELM approach, the minimal number of hidden neurons, its corresponding input weights and the bias values are selected automatically, whereas the KS-ELM approach requires an exhaustive search to determine the number of hidden neurons. The proposed RCGA-ELM is different from the existing 'evolutionary ELM' (E-ELM) algorithm [23]. In E-ELM, the genetic algorithm searches only for the best input weights and the bias values for a given number of hidden neurons such that the network has better generalization performance. In the E-ELM, the optimal number of hidden neurons are obtained using exhaustive search. Whereas in the RCGA-ELM, new genetic operators are defined to find the minimal number of hidden neurons and their corresponding input weights and the bias values. First, we evaluate the performances of KS-ELM, RCGA-ELM and ELM algorithms using classification problems from UCI machine learning repository [24] to validate the proposed schemes. The results clearly indicate that the proposed KS-ELM and RCGA-ELM provide better generalization performance over conventional ELM algorithm for the classification problems.

For our image quality estimation, experiments are carried out using two disjoint set of original images with its compressed version from the JPEG LIVE image quality database [25]. Out of 29 original images, 20 original images and its compressed version are used for image quality model development. The remaining nine original images and its compressed version are used for evaluating the performance. Finally, the performance of proposed image quality estimators are compared with the available NR image quality metric [10] and full-reference (FR) structural similarity image quality metric (SSIM) [26] techniques.

The organization of this paper is as follows: Section 2 describes the HVS based feature extraction technique. In Section 3, we briefly present the recently developed ELM algorithm and issues related to the ELM algorithm for classification problems with high imbalance in the samples. Section 4 present *k*-fold ELM and the RCGA-ELM classifier to handle high imbalance in the number of samples per class. Performance evaluation of the proposed classifiers for three benchmark multi-category problems and image quality estimation are presented in Section 5. Section 6 summarizes the main conclusions from this study.

2. HVS-based Feature Extraction

It is easily deducible that most of the distortion in image/video is due to the block DCT-based compression. The most popular and widely used image format, on Internet and digital cameras happens to be JPEG [7]. Since JPEG uses the block-based DCT transform for coding, to achieve compression, the major artifact that JPEG-compressed images suffer, is blockiness. In the JPEG coding, non-overlapping 8 × 8 pixel blocks are coded independently using DCT transform. The compression ratio and the image quality are mainly determined by the degree of quantization of these DCT coefficients. The undesirable consequences of quantization manifest as blockiness, ringing and blurring artifacts in the JPEG-coded image. It turns out that the subjective data for all these artifacts are highly correlated [8]. Hence, measuring the blockiness in turn indicates the overall image quality.

The proposed image quality metric is designed to take into consideration the various human visual criteria while quantifying the blocking artifact. These blocking artifacts would appear as horizontal and vertical edge distortions at the boundaries of 8 × 8blocks. The visual sensitivity to these edges is affected by the following four parameters [2]: (i) edge amplitude, (ii) edge length, (iii) background activity, and (iv) background luminance.

- *Edge amplitude*: Edge amplitude quantifies the strength of edge along the borders of 8 × 8blocks. It is meant to distinguish between the true image edges and those that apparently arise due to compression. Since the JPEG uses the DCT-based transform coding for every 8 × 8 image block, the effect of quantization appears as the blocking artifacts at the edges of each 8 × 8 block. Edge amplitude at the 8 × 8 block boundaries is proportional to the amount of compression the image is subjected to. In other words, the edge-amplitude increases with decreasing bit-rates.
- *Edge length*: Edge length quantifies the length of continuous block edges. Similar to edge amplitude, edge length, in general, also increases with compression.
- *Background activity*: Background activity measures the amount of high frequency texture content around the block edges. The blocking artifacts will be masked by the activity in the background. For instance, an edge that happens to lie in a textured region is less visible to an observer compared to the edge against a plain background.
- *Background luminance*: This attribute measures the amount of brightness around the block edges. The visibility of the edge is often affected by the surrounding regions. For example, an edge lying in a darker region is less visible compared to the edge in the brighter region.

The objective of the proposed metric is to integrate the aforementioned human visual factors to measure the quality of the JPEG-compressed images. The algorithmic steps for computing the feature vector are shown in Fig. 1. First, we obtain the edges along horizontal and vertical directions using the corresponding 'Prewitt' edge operators. Activity along, as well as, on either sides of the horizontal and vertical edges is captured by the high-pass filtering. The final binary activity mask is obtained by thresholding the activity measure. This mask only permits regions with lower activity to be considered for blockiness measurements. The background luminance weights are obtained based on the model proposed by Karunasekera and Kingsbury [2]. Here darker regions (0–127) are given less weight and brighter regions (128–255) are given higher weights. Each pixel of the edges that belong to the activity mask is multiplied by the corresponding luminance weight in order to obtain the final horizontal and vertical edge maps. The horizontal and vertical edge profiles are computed from these weighted edge maps. These profiles indicate the edge strength along each row and column of the weighted edge map. Since the effect of blockiness is seen only at block boundaries, every eighth location of the horizontal and vertical profiles is considered for measuring blockiness. The measure of deviation at every eighth location from the average value of the neighborhood of both (horizontal and vertical) profiles is used for extracting the features.

2.1. Implementation details

Consider a gray scale image of the size $M \times N$ (rows × columns). The intensity of the image at any pixel location (i, j) is given by $I(i, j)$, which lies in the range of 0–255.

The algorithm is explained for computing the difference profile along horizontal direction. A similar approach is used for the vertical direction as well.

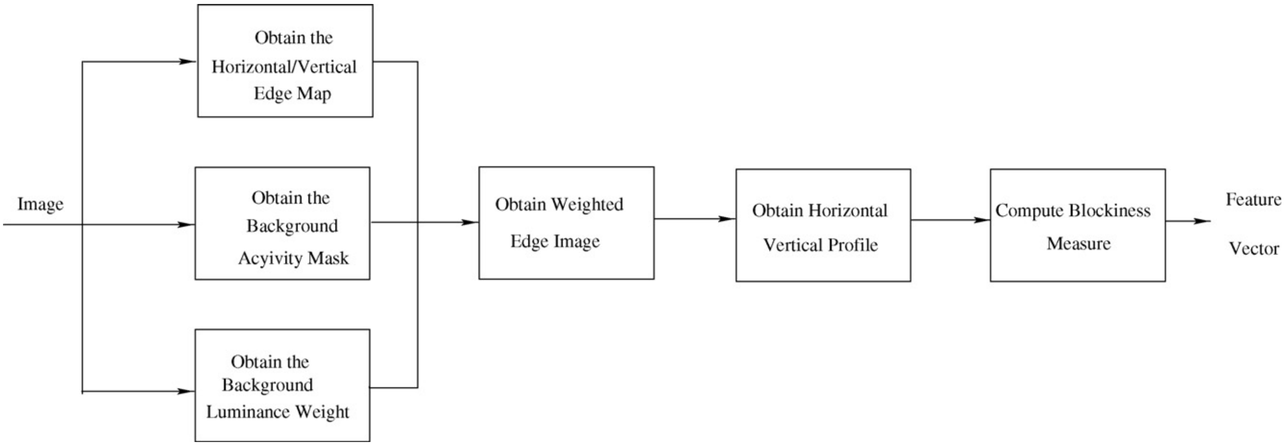
Image
Obtain the Horizontal/Vertical Edge Map
Obtain the Background Acyivity Mask
Obtain the Background Luminance Weight
Obtain Weighted Edge Image
Obtain Horizontal Vertical Profile
Compute Blockiness Measure
Feature Vector

Fig. 1. Overview of feature extraction phase.

(1) Obtain the horizontal edge map using the horizontal 'Prewitt' filter.

$$\begin{aligned}\hat{E}_{\mathrm{h}} &= I * P_{\mathrm{h}},\\ E_{\mathrm{h}}(i,j) &= \begin{cases}\hat{E}_{\mathrm{h}}(i,j) & \text{if } \hat{E}_{\mathrm{h}}(i,j) < \tau_{\mathrm{e}}\\ 0 & \text{otherwise}\end{cases}\end{aligned} \tag{1}$$

where

$$P_{\mathrm{h}} = \frac{1}{3}\begin{bmatrix}1 & 1 & 1\\ 0 & 0 & 0\\ -1 & -1 & -1\end{bmatrix}$$

is the 'Prewitt' horizontal filter and '*' indicates convolution operation. Since the edges caused by blocking artifact have relatively lesser magnitude compared to a true image edge, a pre-designed threshold is used to capture the edges due to blockiness and to avoid the true (strong) image edges. It has been observed experimentally that the maximum edge magnitude due to blockiness is below 40 for wide range of compressions. Choosing a lower threshold might miss the real blocky edges resulting from compression. Thus, the typical value of the edge threshold (τ_{e}) used in all our experiments is set at 35.

(2) Measure the background activity along the horizontal direction (A_{h}).

$$A_{\mathrm{h}} = I * F_{\mathrm{ah}}, \tag{2}$$

where

$$F_{\mathrm{ah}} = \frac{1}{8}\begin{bmatrix}1 & -1 & 1 & -1 & 1 & -1 & 1 & -1\\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0\\ -1 & 1 & -1 & 1 & -1 & 1 & -1 & 1\end{bmatrix}$$

The filter F_{ah} captures the background activity along the horizontal edges. The activity values of the entire image are normalized to the range [0,1].

(3) Mask the edges in the active background region by a pre-defined threshold τ_{a}. The mask M_{h} is given by

$$M_{\mathrm{h}}(i,j) = \begin{cases}1 & \text{if } A_{\mathrm{h}}(i,j) < \tau_{\mathrm{a}}\\ 0 & \text{otherwise}\end{cases} \tag{3}$$

The typical value of the activity threshold (τ_{a}) used in our experiments is 0.15. It has been observed experimentally that the effect of blockiness is masked by the activity when it is more than this threshold (τ_{a}).

(4) The background luminance (W_{l}) of every pixel is obtained using a low-pass filter and the weight of each pixel is assigned as follows:

$$W_{\mathrm{l}}(i,j) = \begin{cases}\sqrt{\dfrac{I_{\mathrm{l}}(i,j)}{128}} & \text{if } 0 \le I(i,j) \le 128\\ 1 & \text{otherwise}\end{cases}$$

where

$$I_{\mathrm{l}} = I * f_{\mathrm{lp}} \tag{4}$$

where

$$f_{\mathrm{lp}} = \frac{1}{4}\begin{bmatrix}1 & 0 & 1\\ 0 & 0 & 0\\ 1 & 0 & 1\end{bmatrix}$$

is the low-pass filter used in order to avoid horizontal and vertical artifacts.

(5) The final weighted edge image (E_{H}) for the horizontal direction is obtained as:

$$E_{\mathrm{H}}(i,j) = E_{\mathrm{h}}(i,j) \times M_{\mathrm{h}}(i,j) \times W_{\mathrm{l}}(i,j). \tag{5}$$

(6) Obtain the vertical profile of E_{H} (projection of the rows of E_{H} on a vertical axis).

$$P_{\upsilon}(i) = \sum_{j=1}^{N} E_{\mathrm{H}}(i,j). \tag{6}$$

(7) Obtain the difference profile along horizontal direction (D_{H})

$$D_{\mathrm{H}} = \frac{1}{M}\left(P'_{\upsilon} - \bar{P}'_{\upsilon}\right), \tag{7}$$

where

$$P'_{\upsilon} = P_{\upsilon}(8n), \quad \bar{P}'_{\upsilon}(n) = \bar{P}_{\upsilon}(8n), \quad \bar{P}_{\upsilon} = \mathrm{median}(P_{\upsilon}).$$

Similar steps are used to obtain the difference profile along the vertical direction (D_{υ}). The final difference profile (D_{F}) is obtained by combining the horizontal (D_{H}) and vertical (D_{υ}) difference profiles (i.e., $D_{\mathrm{F}} = \{D_{\mathrm{H}}, D_{\upsilon}\}$). The final feature vector (U) is obtained by quantizing the combined difference profile D_{F} into 14 bins (14 features). Since most of the difference profile values fall in the range of −1 to 2.5, the bins in this range are finely quantized. The following bin-intervals are used to obtain the final feature vector: $[\mathrm{Min}, -1)$, $[-1, -0.6)$, $[-0.6, -0.3)$, $[-0.3, -0.1)$, $[-0.1, 0.1]$, $(0.1, 0.3]$, $(0.3, 0.6]$, $(0.6, 1]$, $(1, 1.5]$, $(1.5, 2]$, $(2, 2.5]$, $(2.5, 3.5]$, $(3.5, 5]$, $(5, \mathrm{Max}]$. The procedure to obtain the feature vector for a given image is summarized below:

- Obtain the horizontal (D_{H}) and vertical (D_{υ}) difference profile.
- Combine the horizontal and vertical profiles to form a final profile as $D_{\mathrm{F}} = [D_{\mathrm{H}} \ D_{\upsilon}]$.
- Obtain the final feature vector U by constructing histogram of D_{F} with the specified bin intervals. Let function $b : \mathfrak{R} \rightarrow \{1, 2, \ldots, 14\}$ associates the ith value of final difference profile (D_{F}^{i}) with the bin index. The 14-dimensional feature vector U is obtained as

$$U_{\mathrm{h}} = \sum_{i=1}^{n} \delta[b(D_{\mathrm{F}}^{i} - h)] \quad h = 1, 2, \ldots, 14. \tag{8}$$

  where δ is the Kronecker delta function and n be the number of elements in the final difference profile.

Since image quality is a subjective phenomenon, manual inspection plays a major role in testing any given image quality metric. The subjective test is concerned with how an image is perceived by a viewer and designates his/her opinion on a particular image (opinion score). The mean opinion score (MOS) provides the average of the opinion scores over all subjects. The aim of any quality metric is to predict the quality as close to the MOS as possible. Hence, the objective here is to find the functional relationship between the extracted HSV features and the MOS for quantifying the quality of the image. The functional relationship between the HVS features are nonlinear and difficult to model mathematically. In recent years, neural networks have emerged as powerful mathematical tools for solving problems as diverse as pattern classification/recognition, medical imaging, speech recognition, etc. The increasing popularity of neural networks is due to their ability to construct good approximation of functional relationship between the known set of input and output data. Hence, in this study, we explore the possibilities of using neural networks for image quality estimation.

2.2. Database

The LIVE image quality assessment database [25] is used in our study. Here, 29 JPEG images are used to generate a database of 204 JPEG images with different compression rates. Including the original images, we have 233 JPEG images for image quality estimation. The study was conducted in two sessions: first session with 116 images with 20 subjects and the second session with 117 images with 13 subjects. Each subject was shown the images randomly and asked to mark the perception quality. The quality class 1 indicates the MOS range 1–10, quality class 2 indicates the range 10–20, and so on. Here, class C_1 symbolizes the worst quality and class C_{10} stands for the best quality. Finally, the perception quality was linearly transformed to 1–10 range. The resulting MOS values are used for comparison.

To develop the quality model, we have selected two disjoint sets of images for training and testing. The training set images and its compressed versions are not used in the testing set. Out of 29 source images, 20 images are used for training and the remaining nine are used for testing, i.e., 154 (20 original and their 134 compressed versions) images are used to develop the quality model and 79 (9 original and their 70 compressed versions) images are used to estimate the generalization performance. Some of the example images from the database are shown in Fig. 2 for training and testing.

The distribution of training and testing images over quality class are listed in Table 1. From the table, we can see that class C_1 has 4 samples for the model development, and 9 samples for testing the performance. The class C_3 has 17 samples for the model development, but only 3 samples for testing the performance. Since there are no samples representing the class C_{10}, it is neglected in the study. From the table, it can be observed that the image quality data set have high imbalance in the number of samples per class. Similar observation can be made from most of the practical classification problems. The objective here is to find a method which can handle the imbalance and provide better generalization performance.

Table 1
Distribution of images over quality class

Data set	Quality class									
	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}
Training	4	22	17	10	15	13	14	39	20	0
Testing	9	9	3	7	6	7	7	23	8	0

3. Extreme learning machine

In this section, we present a brief overview of the extreme learning machine (ELM) algorithm [22]. ELM is a single hidden layer feedforward network, where the input weights are chosen randomly and the output weights are calculated analytically. For hidden neurons, many activation functions such as sigmoidal, sine, Gaussian and hard-limiting function can be used, and the output neurons have linear activation function. ELM uses the non-differentiable or even discontinuous functions as an activation function.

In general, a multi-category classification problem can be stated in the following manner. Suppose we have N observation samples $\{X_i, Y_i\}$, where $X_i = [x_{i1}, \ldots, x_{in}] \in \Re^n$ is an n-dimensional features of sample i and $Y_i = [y_{i1}, y_{i2}, \ldots, y_{iC}] \in \Re^C$ is its coded class label. If the sample X_i is assigned the class label c_k, then the k th element of Y_i is one ($y_{ik} = 1$) and other elements are –1. Here, we assume that the samples belong to the C distinct classes. The function describing the useful information on probability of predicting the class label with the desired accuracy is called as classifier function and is defined as

$$Y = F(X) \tag{9}$$

The objective of the classification problem is to estimate the functional relationship between the random samples and its class label from the known set of samples.

Using universal approximation property, one can say that the single layer feedforward network with sufficient number of hidden neurons H can approximate any function to any arbitrary level of accuracy [22]. It implies that for bounded inputs to the network,

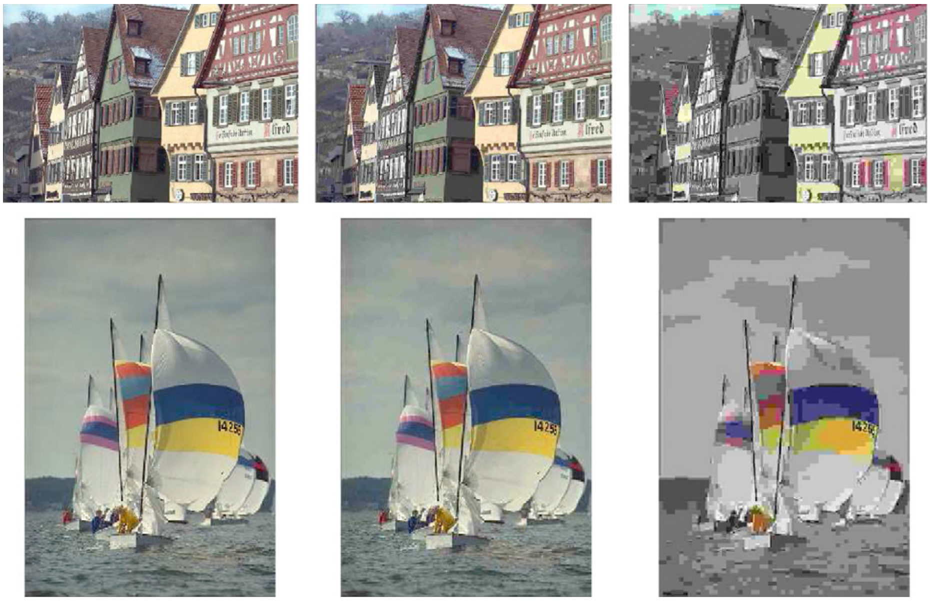

Fig. 2. Example figures from the database. The first row shows one of the training images used at different compression rates. Similarly, second row shows one of the test images used. The first column images are uncompressed.

there exist optimal weights (not necessarily unique) to approximate the function. Let W_{i} be $H \times n$ input weights, B be $H \times 1$ bias of hidden neurons and W_{o} be $C \times H$ output weights. The output ($\hat{Y}$) of the ELM network with H hidden neurons has the following form:

$$\hat{F}(X_i) = \sum_{j=1}^{H} W_{\mathrm{o}_{kj}} G_j(W_{\mathrm{i}}, B, X_i), \quad k = 1, 2, \ldots, C \tag{10}$$

where $G_j(\cdot)$ is the output of the jth hidden neuron, and $G(\cdot)$ is the activation function.

For the sigmoidal hidden neurons, the output of the jth hidden neuron $G_j(\cdot)$ is defined as

$$G_j = G\left(\sum_{k=1}^{n} W_{\mathrm{i}_{jk}} x_{ik} + b_j\right), \quad j = 1, 2, \ldots, H, \tag{11}$$

In case of the radial basis function (RBF), the output of the jth Gaussian neuron $G_j(\cdot)$ is defined as

$$G_j = G(b_j ||X - W_{\mathrm{i}}||), \quad j = 1, 2, \ldots, H, \tag{12}$$

where W_{i} and b_j ($b_j \in \Re^+$) are the center and width of the RBF neuron, respectively.

Eq. (10) can be written in matrix form as

$$\hat{Y} = W_{\mathrm{o}} Y_{\mathrm{H}}, \tag{13}$$

where

$$Y_{\mathrm{H}} = \begin{bmatrix} G_1(W_{\mathrm{i}}, b_1, X_1) & \cdots & G_1(W_{\mathrm{i}}, b_1, X_N) \\ \vdots & \vdots & \vdots \\ G_H(W_{\mathrm{i}}, b_H, X_1) & \cdots & G_H(W_{\mathrm{i}}, b_H, X_N) \end{bmatrix}.$$

Here, Y_H (dimension $H \times N$) is called the hidden layer output matrix of the neural network; the ith row of Y_H is the ith hidden neuron outputs for the entire training input X. For most of the practical problems, it is assumed that the number of hidden neurons are always less than that of training samples.

In the ELM algorithm, for a given number of hidden neurons, it is assumed that the input weights W_{i} and the bias B of hidden neurons are selected randomly. By assuming the predicted output $\hat{Y}$ is equal to the coded labels Y, the output weights are estimated analytically as

$$\hat{W}_{\mathrm{o}} = Y Y_H^{\dagger}, \tag{14}$$

where $Y_H^{\dagger}$ is the Moore-Penrose generalized pseudo-inverse of Y_H [21].

In summary, the following are the steps involved in the ELM algorithm:

- For a given training samples (X_i, Y_i), select the appropriate activation function $G(\cdot)$ and the number of hidden neurons;
- Generate the input weights W_{i} and the bias values B randomly.
- Calculate the output weights W_{o} analytically: $W_{\mathrm{o}} = Y Y_H^{\dagger}$.

Issues in ELM: A multilayer perceptron network trained using the back propagation algorithm searches for optimal W_{i}, B and W_{o} for a given number of hidden neurons such that the approximation error is minimum, i.e.,

$$\min_{\{W_{\mathrm{i}}, B, W_{\mathrm{o}}\}} ||W_{\mathrm{o}} Y_H - Y|| \tag{15}$$

which is equivalent to minimizing the mean squared error $E = \sum_{i=1}^{N} (\hat{Y}_i - Y_i)^2$. Here, a gradient descent algorithm is used iteratively to adjust the free parameters.

In the ELM algorithm, the output weights are found analytically by minimizing the errors for randomly selected input weights, hidden bias values for a given number of hidden neurons, i.e.,

$$\min_{\{W_{\mathrm{i}}, B\}} \left\{ \min_{W_{\mathrm{o}}} ||W_{\mathrm{o}} Y_H(W_{\mathrm{i}}, B, X) - Y|| \right\}. \tag{16}$$

Here, one has to find the optimal number of hidden neurons and the corresponding W_{i} and B such that the analytically calculated output weights ensure better generalization ability of the ELM network. Now, we pose this problem as an optimization problem.

Given: The input–output data and activation function

Find: Optimal number of hidden nodes (H). The corresponding input weights (W_{i}) and the bias values (B) such that the ELM network with analytically found output weights has better generalization performance.

We know that for randomly fixed input weights and the bias values, one can directly use Eq. (14) to find the optimal output weights in the least-square sense. However, the generalization performance of the ELM algorithm depends on the proper selection of the fixed parameters (the number of hidden neurons, input weights and the bias values). Particularly, the selection of these parameters affects the generalization performance considerably in classification problems where fewer number of training samples and a high imbalance in the number training samples per class are present. For our study, we consider an ELM network with 70 hidden neurons. We call the ELM algorithm 500 times for the same training/testing data and network configuration and find the mean and the variance values of the training and the testing accuracies. Each time the ELM algorithm is called, the fixed parameters (the input weights and the bias values of the hidden neurons) are initialized randomly from a uniform distribution. The input data are normalized between −1 to 1, and the weights and the bias values are initialized between ±1. In this study, we use an unipolar sigmoidal activation function ($1/(1 + \mathrm{e}^{-\lambda u})$) for the hidden neurons. The slope (λ) of the sigmoidal function is selected as 0.1 (approximately equal to the inverse of the number of features or the number of input neurons). The mean and the standard deviation of training efficiency are 81.76 and 2.39, respectively. Similarly, the mean and the standard deviation of testing efficiency are 50.5 and 4.3, respectively. The variation of training and testing efficiency for different runs are illustrated in Fig. 3. From the figure, we can say that the random selection of fixed parameters affects the generalization performance of the ELM classifier significantly for classification problems with high imbalance in the number of samples per class.

Also, the behavior of the ELM classifier with respect to initial parameters changes considerably with the number of hidden neurons. To illustrate the behavior, we conducted an experimental study by varying the number of hidden neurons from 20 to 100 in steps of 10. The variation of the training and the testing accuracies are given in Figs. 4 and 5. From Fig. 4, we can see that the training accuracy increases with increase in number of hidden neurons and reaches a maximum between 90 and 100 hidden neurons. Also, the variation in accuracy for different initial parameters decreases with increase in number of hidden neurons. From Fig. 5, we can observe that the testing/generalization accuracy decreases with the increase in number of hidden neurons. Also, the testing accuracy reaches a maximum when the number of hidden neurons are between 40 and 60. When the number of hidden neurons reaches 40, the variation in performance with respect to the initial parameters also increases considerably.

Since there are many possible peaks in the generalization efficiency surface, it is difficult to find the best parameters (H, W_{i} and B) such that the training and generalization efficiency is

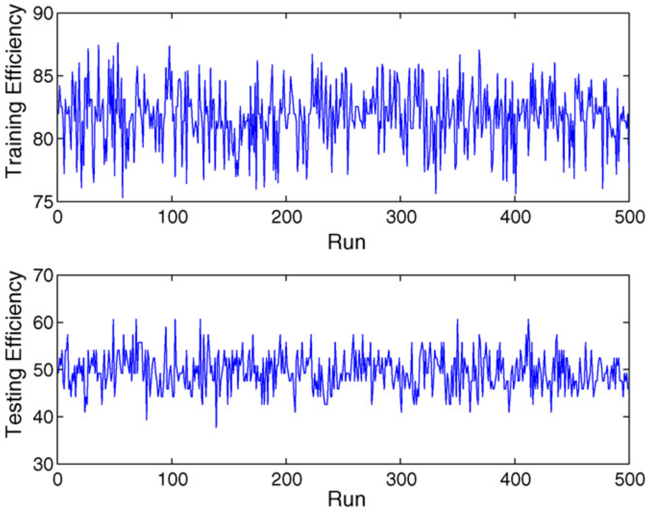

Fig. 3. Effects of the initial parameter selection on training and testing performance.

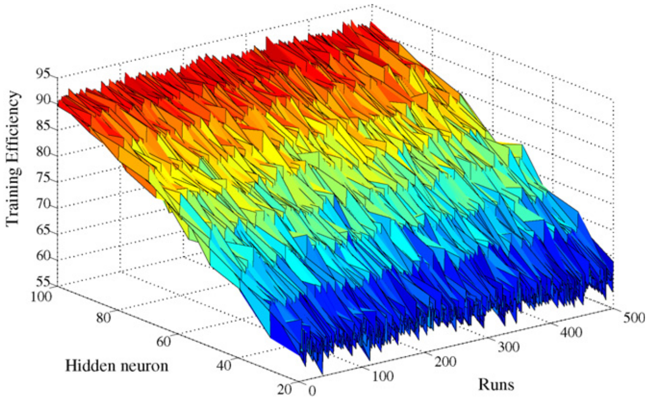

Fig. 4. Training accuracy variation with respect to hidden neurons and initial parameters.

maximum for a given classification problem. Finding the best parameters for the ELM classifier is also a complex problem. Here, one has to search for the optimal number of hidden neurons, input weights and the bias values such that the generalization efficiency is maximum. In this paper, we present two schemes to select appropriate ELM parameters for image quality assessment. The ELM parameters are selected using (i) *k*-fold cross-validation and (ii) real-coded genetic algorithm. These approaches are explained in the following subsections.

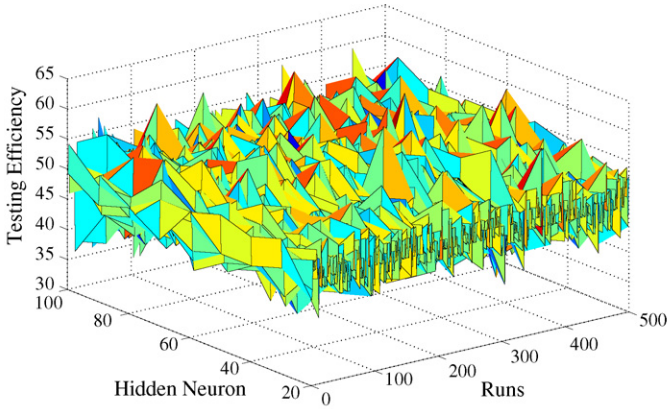

Fig. 5. Testing accuracy variation with respect to hidden neurons and initial parameters.

3.1. *k*-fold cross-validation approach

Cross-validation (CV) and bootstrapping are both methods for estimating generalization error based on 'resampling' [27]. The resulting estimates of generalization error are often used for choosing among various models, different network architectures and parameters. In this paper, CV approach is used to select the input weight and the bias values such that the estimated generalization error is small. In *k*-fold cross-validation, the training data set is divided into *k* equal subsets (of approximately equal size). ELM algorithm is called *k* times, each time leaving one of the subsets from training, and the generalization accuracy is calculated based on the omitted subset. Here, the value of *k* is 5. In order to avoid the bias/variability in estimating the generalization error, 10 random 5-way splits of data are generated using the training samples. After, selecting the optimal input weights and the bias values, the classifier model is developed using the complete training set. This produces better training and generalization accuracy than the earlier approach. We call this algorithm as '*k*-fold Sparse Extreme Learning Machine (KS-ELM)'.

4. Real-coded genetic algorithm approach

The real-coded genetic algorithm (RCGA) is perhaps the most well-known of all evolution based search techniques [28]. Genetic algorithms were developed in an attempt to explain the adaptive processes of natural systems and to design artificial systems based upon these natural systems. Genetic algorithms are widely used to solve complex optimization problems where the number of parameters and constraints are large and the analytical solutions are difficult to obtain. In recent years, many schemes for combining genetic algorithms and neural networks have been proposed and tested. The complete survey on evolving neural networks using genetic algorithms can be found in [29]. In this paper, we use the hybrid real-coded genetic algorithm approach for the hidden neuron selection, its corresponding input weights and the bias values. In this approach, two different genetic operators are used. The first network based operator controls the number of hidden neurons and the next weight based operator evolves the input weights and the bias values. Since, the input weights and the bias values are in continuous domain (real value), real numbers are used to represent them as strings. The weights and the bias values are represented in the string as a two-dimensional real array. The size of the array depends on the number of hidden neurons (rows) and the number of input features (columns). For example, let us consider an ELM classifier with three input features and three hidden nodes. The input weights and the bias values are coded in the string (S) as

$$S = \begin{bmatrix} W_{i_{11}} & W_{i_{12}} & W_{i_{13}} & b_1 \\ W_{i_{21}} & W_{i_{22}} & W_{i_{23}} & b_2 \\ W_{i_{31}} & W_{i_{32}} & W_{i_{33}} & b_3 \end{bmatrix} \tag{17}$$

In this string representation, the number of rows are different for different strings in the population. The genetic operators are defined such that it can handle the stings with different sizes. Using the aforementioned string representation we can uniquely represent all combinations of the hidden neurons.

4.1. Selection function

In genetic algorithm, the selection of a search node from the existing search nodes (population), to produce new search nodes for the next generations plays an important role. A probabilistic selection is performed based upon the fitness of search nodes, such that the better search nodes have a better chance of being selected for producing new search nodes using genetic operators. It is possible that a search node in the population can be selected more than once for producing new search nodes. Several schemes such as roulette wheel selection and its extensions, scaling techniques, tournament and ranking methods are presented for the selection process [28]. In our studies, normalized geometric ranking method [28] is used for the selection process. In normalized geometric ranking method, the search nodes are arranged in descending order of their fitness value. Let q be the selection probability for selecting best search node and r_j be the rank of the jth search node in the partially ordered set. The probability of search node j being selected using normalized geometric ranking method is

$$s_j = q'(1-q)^{r_j - 1} \tag{18}$$

where $q' = (q/(1-(1-q)^N))$ and N is the population size.

4.2. Genetic operators

Genetic operators provide the basic search mechanism of the genetic algorithm. The operators are used to create new search nodes based on existing search nodes in the population. New search nodes are obtained by combining or rearranging parts of the old search nodes, and a new search node obtained may be a better solution to the optimization problem. These genetic operators are analogous to those which occur in the natural world: reproduction (i.e., crossover or recombination) and mutation. The probability of these operators affects the efficacy of the genetic algorithm. The real-coded genetic operators used in our study are described below.

4.2.1. Crossover operator

Crossover operator is a primary operator in genetic algorithm. The role of crossover operator is to recombine information from the two selected search nodes to produce two new search nodes. The crossover operator improves the diversity of the solution. In this paper, we present the weight connection and the network architecture based crossover operators. The operators which act on individual weight connections of the search nodes are called the ‘weight based crossover operators’ and operators which act on network architecture are called the ‘network based crossover operators’. Let C and B are the two search nodes selected for crossover operations.

$$B = \begin{bmatrix} W_{i_{11}} & W_{i_{12}} & W_{i_{13}} & b_1 \\ W_{i_{21}} & W_{i_{22}} & W_{i_{23}} & b_2 \\ W_{i_{31}} & W_{i_{32}} & W_{i_{33}} & b_3 \\ W_{i_{41}} & W_{i_{42}} & W_{i_{43}} & b_4 \end{bmatrix} \tag{19}$$

$$C = \begin{bmatrix} W_{i_{11}} & W_{i_{12}} & W_{i_{13}} & b_1 \\ W_{i_{21}} & W_{i_{22}} & W_{i_{23}} & b_2 \\ W_{i_{31}} & W_{i_{32}} & W_{i_{33}} & b_3 \end{bmatrix} \tag{20}$$

Now, we present the detailed description of these operators.

4.3. Weight based operator

In case of weight connection based crossover, L weight values are randomly selected in the parents. This operator uses averaging operation to generate the values of the selected connections in the children. Let $P_1 = [W_{i_{11}}, W_{i_{22}}, W_{i_{33}}]$ are the three weights selected from the parent C and $P_2 = [W_{i_{21}}, W_{i_{12}}, W_{i_{32}}]$ be the three weights selected from the parent B. The new values of weights in the children are generated as

$$H_1 = P_1 + \beta(P_1 - P_2), \tag{21}$$

$$H_2 = P_2 + \beta(P_2 - P_1), \tag{22}$$

where β is a scalar value in the range of $(0 \le \beta \le 1)$. In our simulation studies, β is set to 0.2.

4.4. Network based operator

In case of network based operator, we randomly select the weights of L hidden neurons from the parents. This operator uses heuristic operation to generate the weights of the L hidden neurons. Let hidden neurons 1 and 4 (the weights connected to the neurons 1 and 4, i.e., rows 1 and 4) be selected from the parent B and hidden neurons 2 and 3 (the weights connected to the neurons 2 and 3, i.e., rows 2 and 3) be selected from the parent C. The weights values of hidden neurons in the child are generated as

$$H_1 = P_1 \pm \gamma w_m \frac{P_1 - P_2}{\|P_1 - P_2\|}, \tag{23}$$

$$H_2 = P_2 \pm \gamma w_m \frac{P_2 - P_1}{\|P_2 - P_1\|}, \tag{24}$$

where w_m is the range of the weight vectors and γ is a positive constant. In our experiment, w_m and γ are set to 2 and 0.2, respectively.

4.4.1. Mutation operator

The mutation operator alters one solution to produce a new solution. The mutation operator is needed to ensure diversity in the population, and to overcome the premature convergence and the local minima problems. Similarly, here also we have weight and network based mutation operators. Now, we describe different mutation operators used in this study. Let us assume that B is the parent selected for the mutation operation.

4.5. Weight based operator

This operator randomly select M weight values for mutation operations. If this operator is applied in a generation t, and let G be the maximum number of generations, then

$$H_{12} = W_{i_{12}} + \Delta(t, w_{\min} - W_{i_{12}}), \quad \text{if} \quad \gamma = 0, \tag{25}$$

$$H_{12} = W_{i_{12}} + \Delta(t, w_{\max} - W_{i_{12}}), \quad \text{if} \quad \gamma = 1, \tag{26}$$

where $\gamma \in [0, 1]$ and

$$\Delta(t, y) = y(1 - r^{1-(t^b/G)}), \tag{27}$$

where r is the random number between the interval [0,1] and b is the parameter that determines the degree of dependency. This function gives a value in the range $[0, y]$ such that the probability of returning a number close to zero increases as the algorithm advances [28].

4.6. Network based operator

The network based mutation operator adds or deletes a hidden neuron in the selected parent. In case of adding a hidden neuron, random numbers in the range w_m are assigned to the weights connecting the hidden neuron with input neurons. In case of deleting a hidden neuron, zero values are assigned to the weights connecting the hidden neuron with the input neurons.

In genetic algorithm, the evolution process continues until a termination criterion is satisfied. The maximum number of generations which is the most widely used termination criterion is used in our simulation studies. We call real-coded genetic algorithm based ELM algorithm as 'real-coded genetic algorithm for extreme learning machine (RCGA-ELM)'. The approach presented here is different from the E-ELM described in [23]. In E-ELM, the genetic algorithm searches only for the best input weights and the bias values for a given number of hidden neurons. The optimal number of hidden neurons are obtained using exhaustive search. In this paper, we present new genetic algorithm approach to search for the number of hidden neurons and their corresponding input weights and the bias values.

4.7. Fitness function

The objective of the ELM classifier is to develop a better model such that the generalization accuracy is very high. In this approach, we use fivefold cross-validation approach to estimate the testing accuracy. Hence, for a given string, the fitness value is equal to the estimated cross-validation accuracy. i.e., first, we find the output weights using analytical Eq. (14) with 4 equal data set and evaluate the generalization accuracy of the classifier using the leave-out set.

The steps involved in the genetic algorithm approach is described below:

1. Randomly create initial population of search nodes.
2. Calculate the fitness of each search node.
3. Select the parents for genetic operations.
4. Generate new population of search nodes using genetic operators.
5. If termination criterion is satisfied, then stop. Otherwise, go to step 2.

We have successfully implemented and tested the real-coded hybrid genetic algorithm approach for ELM classifier in MATLAB on a Pentium-IV machine. The parameters used in the genetic algorithm are: the population size, the selection probability (S_c), the crossover probability (p_c) and the mutation probability (p_m). The following are the GA parameters used in our simulation: P_m is 0.05; P_c is 0.6; maximum number of generation (M) is 500; S_c is 0.08; and population size is 30. These values are selected by trail-and-error.

5. Experiments and discussions

In this section, we present the performance comparison of proposed KS-ELM, RCGA-ELM and ELM classifiers on benchmark classification data sets first. Next, we present the results for the image quality estimation problem.

5.1. Benchmark classification problem

Here, we present results on three benchmark classification problems from UCI machine repository, viz.: Image segmentation, Vehicle and Glass [24]. The detailed specification on the number of inputs, the number of classes and the number of training/testing samples are given in Table 2. The image segmentation problem is balanced and has equal number of samples per class. The 'Glass' and the 'Vehicle' classification problems have fewer samples for classifier development and have strong overlap between the classes. In case of the Vehicle problem, the number of training samples in each classes are: 119, 118, 98 and 89. In case of the Glass problem, the number of training samples in each classes are 35, 38, 9, 7 5 and 15. For the Glass problem, the presence of fewer samples in class C_3, C_4 and C_5 affects the performance considerably. For all three problems, the classifiers are developed using the bi-polar sigmoidal activation function.

Table 2
Specification of UCI classification data set

Data set	Features	Classes	Samples	
			Training	Testing
Image segmentation	19	7	210	2100
Vehicle	18	4	424	422
Glass	9	6 [a]	109	105

[a] Actual number of classes are 7 but there are no samples available for one class.

Recently in [30,31], support vector machine (SVM) based classifiers are developed using unequal cost of misclassification to handle high imbalance in number of samples for each class. The selected value for cost of misclassification influences the performance of the classifier considerably. Hence, one should fix the cost of misclassification properly to achieve better performance. For all benchmark problems, the cost of misclassification is fixed proportional to the number of samples available in the class. The performance of the classifiers evaluated using the test samples and are reported in Table 3. In SVM classification, we use cross-validation scheme to optimize the cost and width of Gaussian function. From the results, we can see that the KS-ELM, RCGA-ELM classifiers and SVM with unbalanced cost performs better than ELM classifier. But, the proposed RCGA-ELM classifier outperforms the both KS-ELM and SVM with unbalanced cost of misclassification. The main issue in the SVM with unbalanced cost if fixing the cost a priori. For multi-category classification problems with strong overlap between classes and high imbalance in samples per class, fixing the above cost a prior is nearly impossible. The above issue affects performance of SVM classifier.

From the table, we can see that the training/testing efficiency of the proposed KS-ELM and the RCGA-ELM classifiers are same as the normal ELM classifier. But, the number of neurons are small for the proposed classifiers. In case of 'Glass' and 'Vehicle' classification problems, the proposed RCGA-ELM classifier testing efficiency is approximately 10% more than the normal ELM classifier. At the same time, the RCGA-ELM classifier finds a minimal network architecture to approximate the functional relationship between the input features and class labels.

The image segmentation classification problem has balanced samples per class in training and testing data set, whereas the 'Glass' and the 'Vehicle' classification problems have fewer training samples and high imbalance in the data set. As discussed in previous section, the random selection of the weights and the bias

Table 3
Performance comparison using benchmark classification examples

Data set	Algorithm	H	Training efficiency in %	Testing efficiency in %
Image segmentation	ELM	100	100	90.67
	KS-ELM	75	100	91.00
	RCGA-ELM	50	99.52	91.00
	SVM	90 [a]	99.0	90.62
Vehicle	ELM	300	90.57	67.62
	KS-ELM	225	88.21	68.74
	RCGA-ELM	75	89.15	74.18
	SVM	234	90.12	68.72
Glass	ELM	60	89.91	68.95
	KS-ELM	60	88.07	73.71
	RCGA-ELM	60	88.99	78.09
	SVM	102	90.07	64.23

[a] Number of support vectors.

values used in normal the ELM algorithm affects the generalization performance considerably for classification problems with fewer samples and high imbalance. For such cases, the proposed classifiers outperforms the conventional ELM classifier by proper selections of the input weights and the bias values. Hence, from the above study, we can say that the proposed KS-ELM and RCGA-ELM classifiers can find the best classifier model even under high imbalance condition.

5.2. Image quality model

In this section, first we compare the proposed KS-ELM and RCGA-ELM classifiers with the conventional ELM algorithm. Next, we estimate the MOS value using class label and the posterior probability. To develop the KS-ELM, RCGA-ELM and ELM image quality models, we have selected two disjoint sets of images for training and testing. The training set images and its compressed versions are not used in testing set. Out of 29 source images, 20 images were used for training and the remaining 9 source images were used for testing. Totally 154 images were used for training (20 original and their 134 compressed versions) and 79 images for testing (9 original and their 70 compressed versions). In our studies, we use both the sigmoidal and the radial basis function as an activation functions for the hidden neurons. The training and testing accuracies for ELM, RCGA-ELM, KS-ELM and SVM with unbalanced cost are given in Table 4. From the table, we can see that the KS-ELM, RCGA-ELM (both sigmoidal and Gaussian hidden neurons) and SVM outperforms the ELM algorithm. The RCGA-ELM outperforms SVM with unbalanced cost and KS-ELM. Further, the RCGA-ELM finds minimal hidden neurons to approximate the decision surface and produce better generalization efficiency than KS-ELM.

To study the image independency of the quality class model developed using various classifiers, we conducted 50 runs with different training images and testing images, i.e., for each run we select 20 different original images and its compressed version for training and remaining 9 original image and its compressed version for testing. The mean and standard deviation of training and testing efficiencies of different classifiers are reported in Table 4. From the mean and standard deviation, we can see that the proposed RCGA-ELM has smaller variance (1.5) when compared to other methods. Hence, we can say that the proposed RCGA-ELM approach handles the high imbalance in the class and overlap between the classes effectively.

It has been proved in literature that the neural network based classifier approximate the posterior probability very well [32]. In our formulation, the class labels are coded between ±1. Hence, the posterior probability of image X_i belongs to class c_j is obtained as

$$\hat{p}(X_i|c_j) = \frac{\max(\min(\hat{Y}_j, 1), -1) + 1}{2} \tag{28}$$

From our result, we observed that the misclassified images are very close to true label, i.e., the image belongs to class the k is classified into the class $k-1$ or $k+1$. Most of these images belong to the boundary between any two neighboring classes, and is easily seen from their respective MOS values. For example, let us consider an image classified as class 9 instead of class 8. The predicted posterior probability for the classes 7, 8, and 9 are: 0.0043, 0.4425 and 0.5678 and zero for other classes. The true MOS value is 7.95. From these probability values, we can see that the posterior probability of the classes 8 and 9 are almost same. This is due to the fact that the MOS of this image is very close to the boundary region of the classes 8 and 9. Hence, we present a method to calculate the image quality value for a given image using the estimated posterior probability and its predicted class label. The image quality (Q) is given as

$$Q = \frac{\sum_j \bar{c}_j \hat{q}_j}{\sum_j \hat{q}_j} \tag{29}$$

where $\bar{c}_j$ indicates the mean MOS of the class j.

Now, we present the image quality model using RCGA-ELM with sigmoidal activation function and other approaches used in the literature. The correlation between MOS and RCGA-ELM based image quality metrics are shown in Figs. 6(a) and 7(a) for training and test images, respectively. Similar study is carried out using Wang's NR quality metric (see Figs. 6(b) and 7(b)) [10] and FR based SSIM index (Figs. 6 and 7(c)) [26]. Here the SSIM index is an improved version of the universal image quality index [33] which is a full-reference metric (FR).

From the figures, we can see that the proposed RCGA-ELM based image quality model emulates the MOS better than the other conventional methods. This can also be deduced from the quantitative performance analysis. The root mean square error (RMSE) deviation from the MOS for image quality metric using different methods and the R-square estimates are given in Table 5. From the table, it can be inferred that RMSE error of the proposed model is less than the Wang's NR model, but it is slightly better than the FR model. Similar observations are made for the KS-ELM based image quality metric.

Furthermore, the R-square statistical estimate provide clear picture on the effectiveness of the proposed image quality model over the Wang's JPEG image quality index and the SSIM [26]. If the R-square value is close to one, then the predicted quality metric emulates the MOS very well. From Table 5, we can see that the RCGA-ELM and FR metrics are close to one and hence, they emulate the MOS very well. Since the R-square estimate for the NR metric is negative, we can say that the proposed RCGA-ELM is better than NR image quality metric.

From the results, we can say that the assessment of image quality using the proposed RCGA-ELM is comparable with the FR approach. But, the FR scheme requires original reference image for

Table 4
Performance of KS-ELM, RCGA-ELM, ELM and SVM algorithms

Type	Algorithm	H	Training efficiency in %			Testing efficiency in %		
			Best	Mean	S.D.	Best	Mean	S.D.
SIG	ELM	110	90.11	88.00	2.5	47.54	42.50	5.1
	KS-ELM	80	84.47	82.1	2.1	55.64	51.11	4.5
	RCGA-ELM	64	88.81	87.4	1.4	81.11	80.10	1.3
RBF	ELM	100	87.20	84.89	2.9	50.82	46.89	4.1
	KS-ELM	70	84.41	81.89	2.6	58.22	55.45	2.8
	RCGA-ELM	62	88.67	87.12	1.4	74.67	72.82	1.7
RBF	SVM	100 [a]	92.05	89.67	2.4	62.02	60.41	2.3

[a] Number of support vectors.

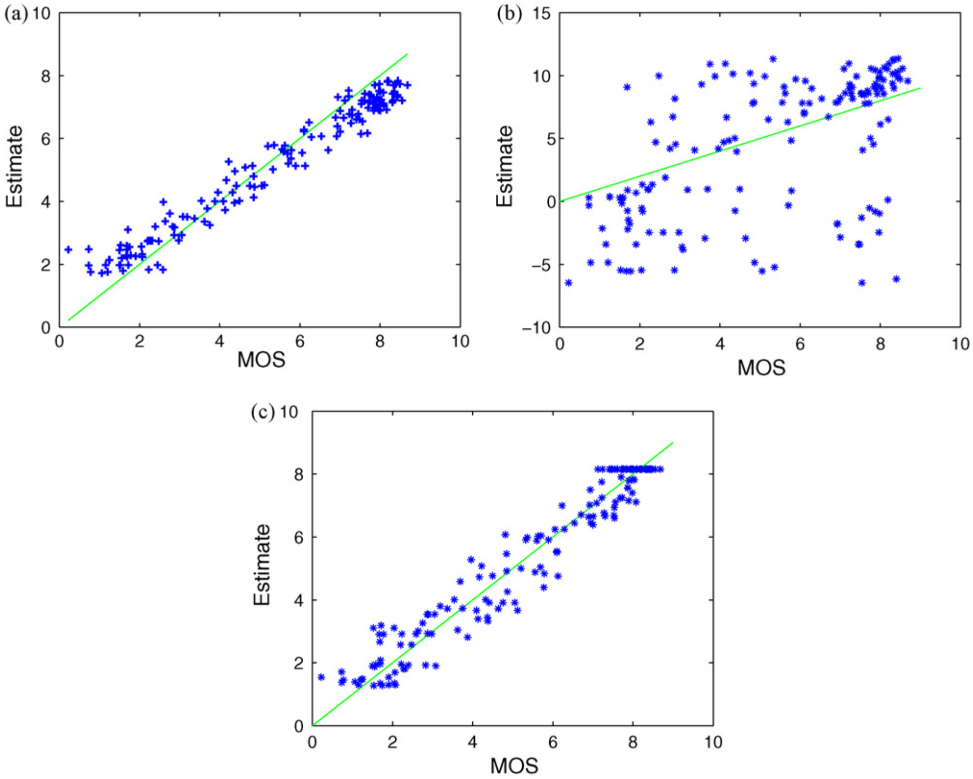

Fig. 6. Image quality prediction by (a) proposed method, (b) Wang-Bovik method and (c) SSIM score for 154 training images.

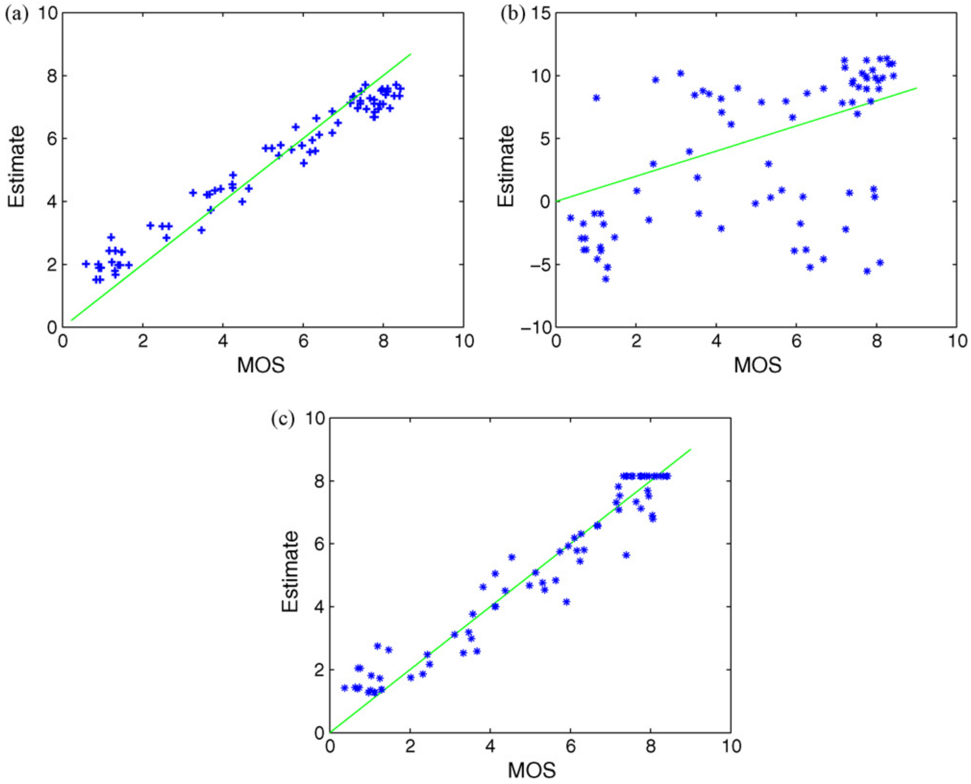

Fig. 7. Image quality prediction by (a) proposed S-ELM method, (b) Wang-Bovik method and (c) SSIM score for 79 test images.

Table 5
RMSE and statistical measures for performance comparison

Method	Training		Testing	
	RMSE	*R*-Square	RMSE	*R*-Square
RCGA-ELM	0.69	0.901	0.70	0.923
NR	4.69	−2.39	5.09	−2.59
FR (SSIM)	0.63	0.907	0.65	0.927

predicting the image quality which is often unavailable. Hence, the proposed no-reference RCGA-ELM approach is more suitable for practical application.

6. Conclusions

In this paper, we have presented a system for predicting image quality using extreme learning machine algorithm, considering various human visual characteristics. The functional relationship between the extracted HVS features and the MOS is modeled by the ELM algorithm. The random selection of input weights and the bias values considerably affects the generalization performance of the ELM algorithm for classification problems with high imbalance in training data set. For this purpose, we presented two schemes, namely, the *k*-fold validation based ELM (KS-ELM) and the real-coded genetic algorithm based ELM (RCGA-ELM). The generalization ability of the proposed KS-ELM and RCGA-ELM classifiers outperform the conventional ELM classifier. We also compared the performance of the proposed schemes with the well-known support vector machines for high imbalance classification problems. The result indicate that the proposed RCGA-ELM outperforms the other methods. The performance of the proposed ELM classifier based image quality metric is found to be better than other previously reported no-reference image quality metrics and achieves performance closer to the full-reference metric. This metric can be easily extended to measure the quality of MPEG/-H.26x compressed videos which also use similar block DCT-based compression.

Acknowledgments

This work was in part supported by the ITRC, Korea University, Korea, under the auspices of the Ministry of Information and Communication. The authors would also like to thank Prof. Bovik and his lab members for providing the JPEG image quality assessment database to test our metric.

References

[1] Z. Wang, H.R. Sheikh, A.C. Bovik, No-reference perceptual quality assessment of JPEG compressed images, in: Proceedings of the ICIP'02, vol. 1, 2002, pp. 477–480.

[2] S.A. Karunasekera, N.G. Kingsbury, A distortion measure for blocking artifacts in images based on human visual sensitivity, IEEE Transactions on Image Processing 4 (6) (1995) 713–724.

[3] H.R. Wu, M. Yuen, A generalized block-edge impairment metric for video coding, IEEE Signal Processing Letters 70 (3) (1998) 247–278.

[4] S. Suthaharan, A perceptually significant block-edge impairment metric for digital video coding, in: Proceedings of the ICASSP'2003, vol. 3, Hong Kong, (2003), pp. 681–684.

[5] W. Gao, C. Mermer, Y. Kim, A de-blocking algorithm and a blockiness metric for highly compressed images, IEEE Transactions on Circuits and Systems for Video Technology 12 (12) (2002) 1150–1159.

[6] Video Quality Experts Group (VQEG), website: http://www.vqeg.org.

[7] JPEG official site, http://www.jpeg.org/.

[8] L. Meesters, J.-B. Martens, A single-ended blockiness measure for JPEG-coded images, Signal Processing 82 (3) (2002) 369–387.

[9] Z. Wang, A.C. Bovik, B.L. Evans, Blind measurement of blocking artifacts in images, in: Proceedings of the ICIP'00, vol. 3, 2000, pp. 981–984.

[10] Z. Wang, A.C. Bovik, L. Lu, Why is image quality assessment so difficult? in: Proceedings of the ICASSP'02, vol. 4, 2002, pp. IV-3313–IV-3316.

[11] T. Vlachos, Detection of blocking artifacts in compressed video 36 (13) (2000) 1106–1108.

[12] M. Yuen, H.R. Wu, A survey of hybrid MC/DPCM/DCT video coding distortions, Signal Processing 4 (11) (1997) 317–320.

[13] M. Jung, D. Léger, M. Gazalet, Univariant assessment of the quality of images, Journal of Electronic Imaging 11 (3) (2002) 354–364.

[14] P. Gastaldo, R. Zunino, I. Heynderickx, E. Vicario, Objective quality assessment of displayed images by using neural networks, Signal Processing: Image Communication 20 (2005) 643–661.

[15] P. Gastaldo, R. Zunino, Neural networks for the no-reference assessment of perceived quality, Journal of Electronic Imaging 14 (3) (2005), 033004:1–033004:11.

[16] Y.L. Murphey, Y. Luo, Feature extraction for a multiple pattern classification neural network system, Pattern Recognition Proceedings, 2, 2002, pp. 220–223.

[17] M. Voultsidou, S. Dodel, J.M. Herrmann, Neural networks approach to clustering of activity in fMRI data, IEEE Transactions on Medical Imaging 24 (8) (2005) 987–996.

[18] E. Trentin, M. Gori, Robust combination of neural networks and hidden markov models for speech recognition, IEEE Transactions on Neural Networks 14 (6) (2003) 1519–1531.

[19] F.N. Chowdhury, P. Wahi, R. Raina, S. Kaminedi, A survey of neural networks applications in automatic control, in: Southeastern Symposium on System Theory, 2001, 349–353.

[20] G.-B. Huang, Learning capability and storage capacity of two-hidden layer feedforward networks, IEEE Transactions on Neural Networks 14 (2) (2003) 274–281.

[21] R. Penrose, A generalized inverse for matrices, Cambridge Philosophy Society 51 (1955) 406–413.

[22] G.-B. Huang, Q.Y. Zhu, C.K. Siew, Extreme learning machine: Theory and applications, Neurocomputing 70 (2006) 489–501.

[23] Q.Y. Zhu, A.K. Qin, P.N. Suganthan, G.-B. Huang, Evolutionary extreme learning machine, Pattern Recognition 38 (10) (2005) 1759–1763.

[24] C. Blake, C. Merz, Uci machine learning database, http://www.ics.uci.edu/mlearn/MLRepository.html.

[25] H. R. Sheikh, Z. Wang, L. Cormack, A. C. Bovik, Live image quality assessment database, http://www.live.ece.utexas.edu/research/quality.

[26] Z. Wang, A.C. Bovik, H.R. Sheikh, E.P. Simoncelli, Image quality assessment: From error visibility to structural similarity, IEEE Transactions on Image Processing 3 (4) (2004) 600–612.

[27] L. Breiman, Heuristics of instability and stabilization in model selection, Annals of Statistics 24 (1996) 2350–2383.

[28] Z. Michalewicz, Genetic Algorithm + Data Structures = Evolution Programs, Springer-Verlag, 1994.

[29] J.D. Schaffer, D. Whitley, L.J. Eshelman, Combinations of genetic algorithms and neural networks: A survey of the state of the art, in: Proceedings of the International Workshop Combinations Genetic Algorithms Neural Networks 1 (1), 1992, pp. 1–37.

[30] Y. Lin, Y. Lee, G. Wahba, Support vector machines for classification in nonstandard situations, Machine Learning 46 (1-3) (2002) 191–202.

[31] M.P.S. Brown, W.N. Grundy, D. Lin, N. Cristianini, C.W. Sugnet, T.S. Furey, M. Ares, D. Haussler, Knowledge-based analysis of microarray gene expression data by using support vector machines, Proceedings of the National Academy of Sciences 97(1) (2000) 262–267.

[32] R. Rojas, A short proof of the posterior probability property of classifier neural networks, Neural Computation 8 (2) (1996) 41–43.

[33] Z. Wang, A.C. Bovik, E.P. Simoncelli, A universal image quality index, IEEE Signal Processing Letters 9 (3) (2002) 81–84.