International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Traitement du Signal
Vol. 40, No. 3, June, 2023, pp. 1163-1170
Journal homepage: http://iieta.org/journals/ts

Deep Learning Based Compression with Classification Model on CMOS Image Sensors

Vinayagam Palani[1], Tamilvizhi Thanarajan[2], Anand Krishnamurthy[3], Surendran Rajendran[4*]

[1] Department of Electronics and Communication Engineering, Saveetha Engineering College, Chennai 602105, India
[2] Department of Computer Science and Engineering, Panimalar Engineering College, Chennai 600123, India
[3] Department of Computer Science and Engineering, Rajalakshmi Engineering College, Chennai 602105, India
[4] Department of Computer Science and Engineering, Saveetha School of Engineering, Saveetha Institute of Medical and Technical Sciences, Chennai 602105, India

Corresponding Author Email: surendranr.sse@saveetha.com

https://doi.org/10.18280/ts.400332

Received: 30 December 2022
Accepted: 18 March 2023

Keywords:
image classification, CMOS image sensors, deep learning, MDL-CCIM

ABSTRACT

The complementary metal oxide semiconductor (CMOS) technique is widely used in modern manufacturing processes for the high compatibility. A novel metaheuristic with deep learning based compression with image classification model (MDL-CCIM) technique is developing to compress and classify the images captured by CMOS image sensors. The proposed MDL-CCIM technique follows two major processes, namely, butterfly compression and classification. Primarily, the BOA with LBG model is applied for image compression. Secondly, the DenseNet with softmax layer is employed for image classification. Finally, the hyper parameter tuning of the DenseNet model is optimally chosen by the Adam optimizer. A wide range of simulations was carried out to highlight the enhancement of the MDL-CCIM technique. The extensive comparative analysis reported the improved outcomes of the MDL-CCIM technique over the recent approaches. Hybrid DL models can be used for image classification purposes.

1. INTRODUCTION

The demand for lower-power sensing, higher-resolution devices with incorporated image processing capabilities, particularly compression capability, is growing. The complementary metal oxide semiconductor (CMOS) technique permits the incorporation of image processing and sensing, making it feasible to enhance the complete performance of the system. The CMOS is a semiconductor technology used in the manufacturing of integrated circuits (ICs). In a CMOS circuit, each transistor operates as either an on or off switch. When a transistor is on, it allows current to flow through it, while when it's off, no current flows. By using complementary pairs of transistors, CMOS circuits can efficiently switch between these two states, which makes them ideal for use in digital circuits. One of the main advantages of CMOS technology is its low power consumption. Since the transistors are either fully on or fully off, they consume very little power when they are not actively switching. This makes CMOS ideal for use in battery-powered devices, such as mobile phones and laptops. Another advantage of CMOS technology is its high noise immunity. The complementary pairs of transistors ensure that noise signals are canceled out, resulting in a high signal-to-noise ratio.

The CMOS is used in a wide range of applications, from digital logic circuits to analog signal processing circuits. The CMOS technique permits the incorporation of image processing and sensing, making the CMOS image sensor the optimal solution to enhance the performance of the overall system [1]. Figure 1 shows the block diagram of CMOS image sensor. Over the previous years, image sensors integrating distinct on-chip compression methods, namely, wavelet-based image processing, predictive coding, conditional replenishment, image processing, and SPIHT algorithm were introduced. The integral part of image compression lies in internet browsing, medical sciences, TV broadcasting, navigation applications, etc., with the development of science and techniques, the need for reduction in transmission time and the storage space requirement for digital images becomes an important concern. The broadcast range has been restricted which makes issue [2]. A proper and efficient image compression method is of major need to address the issue of constrained bandwidth.

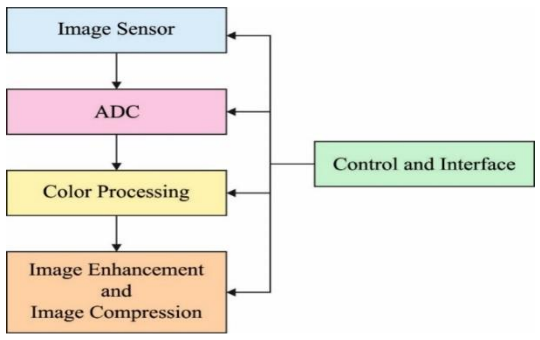


Figure 1. Block diagram of CMOS image sensor

Over the past decades, the research field has paid more towards the presented method for image compression. Vector Quantization (VQ) approach outperforms another method, namely, pulse code modulation (PCM), Adaptive DPCM, differential PCM (DPCM) that belongs to the group of scalar quantization methods [3]. VQ, the more common lossy image compression method is mainly a c-means clustering method commonly employed for pattern recognition and image compression, face detection, speech and image coding speech recognition due to the advantage that includes high compression rate and simple decoding architecture it offers with lower distortion [4]. Object detection and image classification are two applications of artificial intelligence (AI) commonly employed in the industry.

The research gap in the existing works is need to optimize the convolutional neural network (CNN) architecture and CMOS sensor to train the edge platform for larger data. The research motivation of this study is deep learning (DL) model plays an integral part in enhancing the performance and accuracy. High speed or resolution CMOS image sensor provides vital image quality to exact processes like CNN based object detection and image classification [5]. The CNN method that has been trained is extremely based on the training data. When the training data is contained in lower-quality images, the CNN cannot recognize the object under deliberation and is more possible to give poor accuracy for object detection and image classification methods.

The results of the proposed metaheuristics with deep learning based compression with image classification model (MDL-CCIM) method has been provided to identify and compress the images taken by CMOS image sensors. The butterfly compression and classification processes are the two main steps in the proposed MDL-CCIM approach. The BOA with LBG model is primarily used to compress images. Second, DenseNet with a softmax layer is used to classify images. The Adam optimizer then selects the DenseNet model's hyperparameter tweaking in the best possible way. The results are looked at from a variety of angles. A set of 300 images from each category are taken into consideration in this work. The compression result analysis of the MDL-CCIM methodology with new methods under various Codebook Size (CS) and pictures. The results showed that every CS and image had higher Peak signal-to-noise ratio (PSNR) values as a result of the MDL-CCIM technique.

Section 2 of the paper, which analyses the connected works involved in compression with image classification model, is how the other portions of the essay are structured. The suggested MDL-CCIM model is described in Section 3. The results are then analyzed and discussed in Section 4, along with a performance comparison with alternative approaches. Finally, the important findings of the suggested investigation are summarized in Section 5.

2. LITERATURE SURVEY

Gnanasambandam and Chan [6] presented a student-teacher learning system that permits to categorize the noisy raw information. In this work, they showed that using the presented method, we could accomplish image classification at a photon level of a single photon lower or for each pixel. The experiment result verifies the efficiency of the presented approach in comparison with the current solution. Ammah and Owusu [7] projected a discrete wavelet transform DWT method for compressing images and to maintain perceptual quality at a physically tolerant level. In the hybrid method, speckle and salt and pepper noises in ultrasound imagery are decreased considerably. When the image isn't ultrasound, the procedure has negligible effect, hover, edge is conserved. Then, the image is filtered through DWT. A threshold method is employed for generating the coefficient in effective manner. Later, the consequence is vector quantized. At last, the quantized coefficient is Huffman encoded.

Wang and Du [8] have developed a squirrel search algorithm (SSA) using LBG based image compression approach named SSA-LBG for UAV. The SSA is employed for constructing of the codebook for VQ and it uses LBG method as the beginning of SSA for VQ. Zou et al. [9] examined a technique, called Dynamic Predictive Block Adaptive VQ (DP-BAVQ), to decrease the capacity of downlinked information for staggered SAR. Firstly, it compresses the variance of prediction and raw data with the dynamic prediction block adaptive quantization (DP-BAQ). Next, a secondary compression is implemented by using. Xu et al. [10] proposed a higher-capacity RDHEI system-based VQ prediction (VQP) and adoptive block selection. VQ compression is a lossy compression technique. In this method, VQ compression is utilized for estimating the raw pixel with an empty room beforehand encryption. As the variance between the original image and the VQ decompressed image is smaller once the length of codebook is satisfactory, with the variance as a predictive error could attain and obtain space for embedding information.

This research introduces novel metaheuristics with deep learning based compression with image classification model (MDL-CCIM) on CMOS image sensors. The proposed MDL-CCIM technique aims to compress and classify the images captured by CMOS image sensors. The proposed MDL-CCIM technique follows two major processes, namely, butterfly compression and classification. At the initial stage, the butterfly optimization algorithm (BOA) with LBG model is applied for image compression. Next, in the second stage, the DenseNet with softmax layer is employed for image classification. Furthermore, the hyperparameter tuning of the DenseNet model is optimally chosen by the Adam optimizer. A wide range of simulations was carried out to highlight the enhancement of the MDL-CCIM technique.

3. PROPOSED METHODOLOGY

In this research study, a new MDL-CCIM technique has been developed to compress and classify the images captured by CMOS image sensors. The proposed MDL-CCIM technique follows two major processes, namely, butterfly compression and classification. Primarily, the BOA with LBG model is applied for image compression [11]. Secondly, the DenseNet with softmax layer is employed for image classification. Finally, the hyperparameter tuning of the DenseNet model is optimally chosen by the Adam optimizer.

3.1 Design of BOA-LBG based compression technique

The BOA-LBG technique is derived to compress images. An efficient image compression approach is named as BOA-LBG algorithm is established for compressing the image. The BOA-LBG approach creates to utilize of BOA technique with LBG algorithm. The VQ is a lossy data compression method in block coding [12]. The generation of codebook was detected as a vital process of VQ. Considering that size of new images $Y=\{y_{ij}\}$ be $M\times M$ pixel that separated into various blocks with size of $n\times n$ pixel. Especially, there are $N_b = \left[\frac{N}{n}\right] \times \left[\frac{N}{n}\right]$ block that is indicated as set of input vectors=(x_i, i=1, 2, …, N_b). Consider that L be $n\times n$. An input vector $\chi_i, \chi_i \in \Re^L$ where $\Re^L$

is L dimension Euclidean space. The codebook C has of N_c L dimensions code word, i.e., $= \{c_1, c_2, \ldots, c_{N_C}\}, c_j \in \Re^L, \forall j = 1,2, \ldots N_c$.

Every input vector was signified as row vector $x_i=(x_{i1}, x_{i2}, \ldots, x_{iL})$ and j^{th} codeword of codebook is illustrated as $c_j=(c_{j1}, c_{j2}, \ldots, c_{jL})$. The VQ methods assign every input vector for connecting codeword, and the codeword is changing the linked input vector finally to attain the resolve of compression. The optimizing of C interms of MSE was written as minimizing distortion function D. "Usually, the minimal value of D was optimal the quality of C in Eq. (1).":

$$D(C) = \frac{1}{N_b}\sum_{j=1}^{N_c}\sum_{i=1}^{N_b}\mu_{ij} \cdot \|x_i - c_j\|^2 \tag{1}$$

According to the succeeding constraint in Eq. (2):

$$\sum_{j=1}^{N_c}\mu_{ij} = 1, \forall i \in \{1,2, \ldots , N_b\} \tag{2}$$

$$\mu_{ij} = \begin{cases} 1 \; if x_i is \; in \; the \; jth \; cluster; \\ 0 \; otherwise \end{cases} \tag{3}$$

$$L_k \le c_{jk} \le U_k, k = 1,2, \ldots, L \tag{4}$$

where, L_k implies the minimal of k^{th} elements in all trained vectors, and U_k refers the higher of k^{th} elements in every input vector. The ‖x-c‖ indicates the Euclidean distance among the vector x as well as codeword c. The two important states that an optimum vector quantize in Eq. (3) and Eq. (4).

The partition R_j, j=1, …, N_c be needed to be done in Eq. (5):

$$R_j \supset \{x \in \chi: d(x, c_j) < d(x, c \; \forall k) \tag{5}$$

The codeword c_j was offered as the centroid of R_j in Eq. (6):

$$c_j = \frac{1}{N_j}\sum_{i=1}^{N_j} x_i \, , x_i \in R_j \tag{6}$$

where, N_j implies the whole count of vectors going to R_j.

3.1.1 LBG algorithm

This approach to a scalar quantizer was projected by Lloyd. This technique was named LBG or generalization Lloyd algorithm (GLA). It implements the 2 abovementioned states for the input vector for defining the codebook. For providing input vectors x_i, i=1, 2, …, N_b, distance function d, and first codewords $c_j(0)$, j=1, …, N_c. The LBG iteratively executes 2 states to produce a better codebook due to the succeeding approach:

(1) Partition the input vector as numerous groups employing the lesser distance rule. This outcome partition is stored from N_b×N_c binary indicator matrix U when the element is defined as subsequent in Eq. (7):

$$\mu_{ij} = \begin{cases} 1 \quad if \; d\left(x_i, c_j(k)\right) = \min\limits_{p} d\left(x_i, c_p(k)\right) \\ 0 \qquad\qquad otherwise \end{cases} \tag{7}$$

(2) Resolve the centroid of every partition. Interchange the old codeword with centroids Eq. (8):

$$c_j(k+1) = \frac{\sum_{i=1}^{N_b}\mu_{ij}\, x_i}{\sum_{i=1}^{N_b}\mu_{ij}}, j = 1, \ldots, N_c \tag{8}$$

(3) Repeat steps (1) & (2) until no c_j, j=1, …, N_c alters already.

3.1.2 Codebook construction process

For the optimal codebook construction process, the BOA is utilized. The BOA is derived from the characteristics of BFs at the time of matting and food source discovery. It employs a pair of navigation patterns for searching the area. During exploration ($r_1 \le p$), BFs goes in the direction of the optimal BF of the colony, whereas during exploitation ($r_1 > p$), the BFs carry out an arbitrary searching process within the searching area by searching the arbitrary BFs. They can be defined in a mathematically way as follows in Eq. (9) and Eq. (10).

If $r_1 \le p$, Global searching:

$$^{t+1}X_i = {}^{t}X_i + \left(r_2^2 \times g^* - {}^{t}X_i\right) \times \varphi_i \tag{9}$$

If $r_1 > p$, Local searching:

$$^{t+1}X_i = {}^{t}X_i + (r_3^2 \times {}^{t}X_j - {}^{t}X_k) \times \varphi_i \tag{10}$$

where, t and t+1 represent the present and update levels of the respective variables. Besides, the place of the optimal BF can be represented using g^*, and ${}^tX_i$ and ${}^tX_k$, indicates the places of two randomly chosen BF; r_1, r_2 and r_3 implies random scalars and φ_i indicates the fragrance factor which can be represented as follows in Eq. (11):

$$\varphi_i = cI^a \tag{11}$$

where, φ_i indicates the fragrance magnitude for *ith* BF; I and a denote the stimulus intensity and fluctuating absorption degree. I undergoes correlation to the objective function value, and for *ith* BF as $f(X_i)$, where f defines an objective function. The coefficients a and c are chosen from intervals of [0,1], p imply probability switching that computes the searching nature. The entire procedure contained in the BOA-LBG method was provided as follows.

During the initial phase, the parameter initialization occurs so that the codebook generated by LBG method is allocated to the initial solution (i.e., satin bowerbirds), however, the rest of the initial solutions are arbitrarily created. Every solution represents the codebook of NC codeword. In addition, the initialization of BOA occurs.

Secondary phase, the presented optimal solution is selected by computing the fitness of every place and the higher fitness place is detected as the optimal one.

During the tertiary stage, a novel solution is generated in the bowerbird mutation process.

The solution is ranked on the basic of FF in the subsequent step among that the optimal is chosen.

The secondary and tertiary steps are iteration until the termination state is met.

3.2 Design of image classification technique

The proposed image classification module incorporates three major processes, namely, DenseNet based feature extraction, Adam based hyperparameter tuning, and SM based classification.

3.2.1 DenseNet model

The DenseNet technique is a DL approach established on ResNet [13]. Recently, DenseNet is attaining optimum outcomes in the domain of image classification. The fundamental model of ResNet and DenseNet is similar; but, DenseNet introduces a dense connection amongst every preceding layer and the latter, and it recognizes feature reprocess with the connection of features to the channels. This feature creates DenseNet to obtain optimum efficiency than ResNet with some parameters and computational cost and alleviate gradient vanishing problems.

The DenseNet is mostly collected from DenseBlock and Transition layer. The DenseBlock implements a radical dense connection process; i.e., every layer is linked to everyone. Specially, all layers accept the resultant of each previous layer as their input. In DenseBlock, all layers have a similar size and all layers are concatenated with every previous layer in the channel dimensional. In order to network with L layer, DenseBlock has an entire of $L(L+1)/2$ connection. The input of layer L is as follows in Eq. (12):

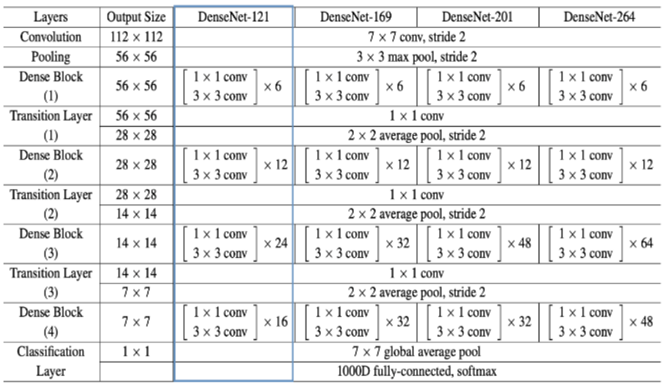
$$x_L = H_L\left([x_1, x_2, \dots, x_{L-2}, x_{L-1}]\right) \tag{12}$$

where, L implies the amount of layers. H_L stands for non-linear transformation that is, a group of Batch Normalization (BN), *ReLU*, Pooling, and Conv functions. The common DenseNet-B infrastructure was employed, and the bottleneck layer was utilized for reducing the count of computation; i.e., the infrastructure *BN*+*ReLU*+1×*lConv*+*BN*+*ReLU*+3×3 Conv was implemented. All layers from DenseBlock output k feature map were then convolutional, specifically, the number of convolutional kernels. When it can be set the channel count of input DenseBlock as k_0, afterward, the input channel count of L layer is $k_0+k(L-1)$. At this point, the last convolutional of all layers are k, and k is named as growth rate. In DenseBlock, with improvement in the number of layers, the number of input channels is superior. While the input size of the model passing to DenseBlock remains unvaried, the channel dimensional is remaining for increasing. So, dimensional decrease is essential for reducing computational complexity.

The Transition layer was mostly collected of 1×1 convolutional and 2×2 Avg_Pooling or Max_Pooling, and their infrastructure was *BN*+ReLU+1×1Conv+2×2 Avg_Pooling. It attaches 2 neighboring denseblock and decreases the dimensional resultant of the denseblock. In this case, DenseNet-121 model is utilized. Figure 2 shows the layered architecture of DenseNet-121 model. To adjust the hyperparameter values of the DenseNet model, the Adam optimizer is utilized in Eq. (13):

$$x_L = H_L\left([x_1, x_2, \dots, x_{L-2}, x_{L-1}]\right) \tag{13}$$

where, L implies the amount of layers. H_L stands for non-linear transformation that is, a group of Batch Normalization (BN), ReLU, Pooling, and Conv functions. The common DenseNet-B infrastructure was employed, and the bottleneck layer was utilized for reducing the count of computation; i.e., the infrastructure *BN*+*ReLU*+1×*lConv*+*BN*+*ReLU*+3×3 Conv was implemented. All layers from DenseBlock output k feature map were then convolutional, specifically, the number of convolutional kernels. When it can be set the channel count of input DenseBlock as k_0, afterward, the input channel count of L layer is $k_0+k(L-1)$. At this point, the last convolutional of all layers are k, and k is named as growth rate. In DenseBlock, with improvement in the number of layers, the number of input channels is superior. While the input size of the model then passing with DenseBlock remains unvaried, the channel dimensional is remaining for increasing. So, dimensional decrease is essential for reducing computational complexity.



Layers	Output Size	DenseNet-121	DenseNet-169	DenseNet-201	DenseNet-264
Convolution	112 × 112		7 × 7 conv, stride 2		
Pooling	56 × 56		3 × 3 max pool, stride 2		
Dense Block (1)	56 × 56	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 6$
Transition Layer (1)	56 × 56		1 × 1 conv		
	28 × 28		2 × 2 average pool, stride 2		
Dense Block (2)	28 × 28	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 12$
Transition Layer (2)	28 × 28		1 × 1 conv		
	14 × 14		2 × 2 average pool, stride 2		
Dense Block (3)	14 × 14	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 24$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 48$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 64$
Transition Layer (3)	14 × 14		1 × 1 conv		
	7 × 7		2 × 2 average pool, stride 2		
Dense Block (4)	7 × 7	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 16$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1\times1 \text{ conv} \\ 3\times3 \text{ conv} \end{bmatrix} \times 48$
Classification Layer	1 × 1		7 × 7 global average pool		
			1000D fully-connected, softmax		

Figure 2. DensNet 121 architecture

3.2.2 Softmax classifier

The SM layer is the forecast the label probability of input x_j with employing the feature learned from the 3rd hidden state representation $h_i^{(3)}$. The number of nodes existing in SM state was chosen as equivalent to the number of labels [14, 15].

The SM state is 5 nodes equivalent to grading groups in 1-5. But the classifiers like SVM are employed, softmax LR permits for optimizing the whole deep network to fine-tuned jointly. The succeeding main function was minimizing to a finetune the network to softmax state in Eq. (14) [16, 17].

$$\begin{aligned} &J_{SSAE-SMC}(W, b, x, \hat{z}) \\ &= \min_{W,b} J(x, \hat{z}) + \lambda^{smc} \|W^{smc}\|_2^2 \end{aligned} \tag{14}$$

where, W and b refers the weight and bias of entire deep network, $J(x, \hat{z})$ represents the LR cost amongst the classifier attained with input feature x and unsupervised outcome $\hat{z}$ and W^{smc} implies the weight and λ^{smc} stands for the weight decomposed parameter [18, 19]. By implementing finetuned, the weight and bias of SM are optimized together, and SM state was employed to the classifier. Assume that y_i refers the label of trained instances x_i. The probability of x_i refers to the k^{th} class was written in Eq. (15):

$$p\left(y_j = k \middle| x_j; W_{smc}, b_{smc}\right) = \frac{e^{W_{smc}^{(k)^T} x_i + b_{smc}^{(k)}},}{\sum_{j=1}^{N} e^{W_{smc}^{(j)^T}}, x_i + b_{smc}^{(j)}} \tag{15}$$

where, $W_{smc}^{(k)}$ and $b_{smc}^{(k)}$ implies the distribution of weight and bias from the k^{th} class. N signifies the entire amount of classes that relate to the 5 grade groups [20]. Due to the maximal probability, it can calculate the grade group of instances x_j utilizing the formula in Eq. (16):

$$Grade(x_i) = arg_k \max_{k=1\dots N} p\left(y_i = k \middle| x_i; W_{smc}, b_{smc}\right) \tag{16}$$

4. RESULTS AND DISCUSSION

The performance validation of the MDL-CCIM technique occurs using a benchmark dataset from Kaggle repository. It contains images under different classes namely Fresh Apples (FA), Fresh Bananas (FB), Fresh Oranges (FO), Rotten Apples (RO), Rotten Bananas (RB), and Rotten Oranges (RO). The results are examined in various aspects. In this work, a set of 300 images from each category are considered. Table 1 and Figure 3 illustrate the compression result analysis of the MDL-CCIM technique with recent methods under varying Codebook Size (CS) and images.

Table 1. PSNR analysis of MDL-CCIM technique under different CS

Codebook Size	Methods	Image-1	Image-2	Image-3
CS=8	IDELBG	26.42	21.41	27.39
	IPSOLBG	26.00	21.04	26.70
	BALBG	24.50	19.90	25.83
	FALBG	25.17	19.26	25.05
	MDL-CCIM	27.44	22.79	28.87
CS=16	IDELBG	27.85	21.20	28.51
	IPSOLBG	27.34	21.16	27.48
	BALBG	26.38	20.00	27.33
	FALBG	25.67	20.38	25.97
	MDL-CCIM	28.69	23.17	29.18
CS=32	IDELBG	28.61	21.57	29.81
	IPSOLBG	28.56	22.41	28.19
	BALBG	26.15	21.17	26.67
	FALBG	25.59	20.36	26.33
	MDL-CCIM	29.83	23.87	30.83
CS=64	IDELBG	30.01	23.84	29.52
	IPSOLBG	29.47	23.52	29.15
	BALBG	26.91	21.99	27.96
	FALBG	27.36	22.39	27.55
	MDL-CCIM	31.16	24.97	30.68
CS=128	IDELBG	30.73	23.89	30.48
	IPSOLBG	31.33	24.68	29.73
	BALBG	28.97	22.85	29.41
	FALBG	28.91	22.91	28.51
	MDL-CCIM	32.74	26.16	31.93
CS=256	IDELBG	32.16	25.42	31.78
	IPSOLBG	31.43	25.43	30.92
	BALBG	28.76	24.04	30.26
	FALBG	29.62	23.73	29.26
	MDL-CCIM	34.74	26.43	32.93

The results demonstrated that the MDL-CCIM technique has resulted in increased Peak signal-to-noise ratio (PSNR) values under every CS and image.

For instance, with CS of 8 and image 1, the MDL-CCIM technique has attained higher PSNR of 27.44dB, whereas the IDELBG, IPSOLBG, BALBG, and FALBG techniques have obtained lower PSNR of 26.42dB, 26dB, 24.50dB, and 25.17dB respectively. Likewise, with CS of 16 and image 1, the MDL-CCIM technique has achieved increased PSNR of 28.69dB, whereas the IDELBG, IPSOLBG, BALBG, and FALBG techniques have reached reduced PSNR of 27.85dB, 27.34dB, 27.34dB, 26.38dB, and 25.67dB respectively.

Similarly, with CS of 32 and image 1, the MDL-CCIM technique has gained improved PSNR of 29.83dB, whereas the IDELBG, IPSOLBG, BALBG, and FALBG techniques have attained decreased PSNR of 28.61dB, 28.56dB, 26.15dB, and 25.59dB respectively. Concurrently, with CS of 64 and image 1, the MDL-CCIM technique has depicted reasonable PSNR of 31.16dB, whereas the IDELBG, IPSOLBG, BALBG, and FALBG techniques have exhibited lower PSNR of 30.01dB, 29.47dB, 26.91dB, and 27.36dB respectively.

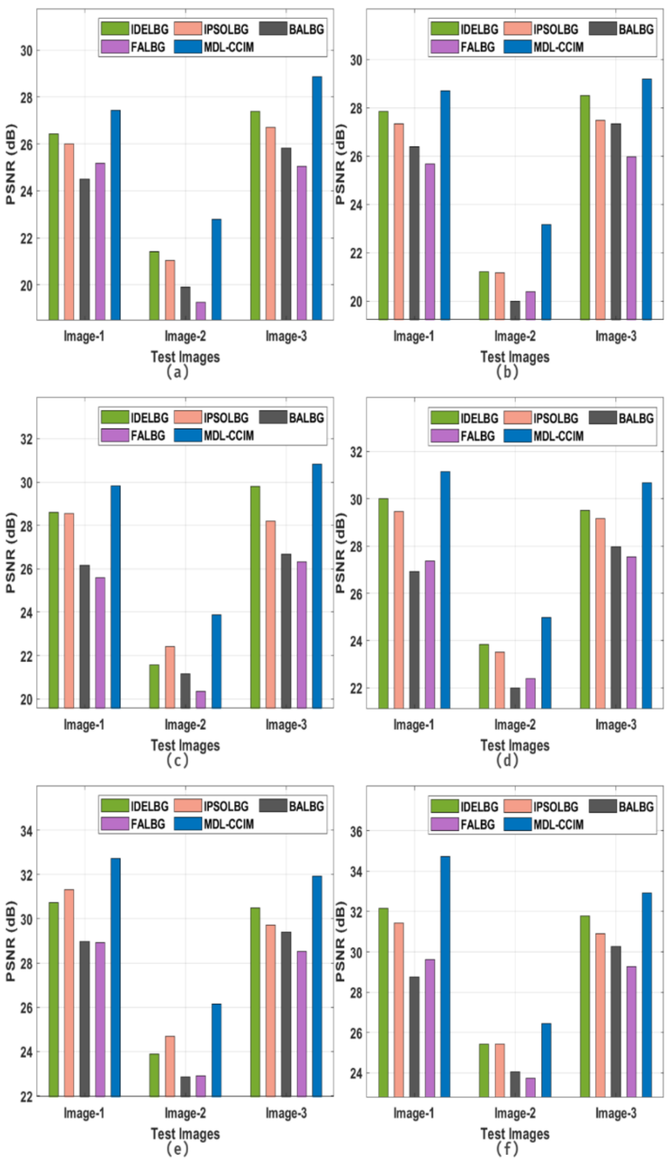


Figure 3. Comparative analysis of various methods in terms of PSNR (dB) a) CS=8 b) CS=16 c) CS=32 d) CS=64 e) CS=128 f) CS=256

Simultaneously, with CS of 128 and image 1, the MDL-CCIM technique has reached to maximum PSNR of 32.74dB, whereas the IDELBG, IPSOLBG, BALBG, and FALBG techniques have resulted in minimal PSNR of 2630.73dB, 31.33dB, 28.97dB, and 28.91dB respectively. At last, with CS of 256 and image 1, the MDL-CCIM technique has attained higher PSNR of 34.74dB, whereas the IDELBG, IPSOLBG, BALBG, and FALBG techniques have obtained lower PSNR of 32.16dB, 31.43dB, 28.76dB, and 29.62dB respectively.

Figure 4 highlights the confusion matrix offered by the MDL-CCIM technique. The figure reports that the MDL-CCIM technique has identified 284 images under class FA, 285 images under class FB, 291 images under class FO, 300 images under class RA, 292 images under class RB, and 285 images under class RO.

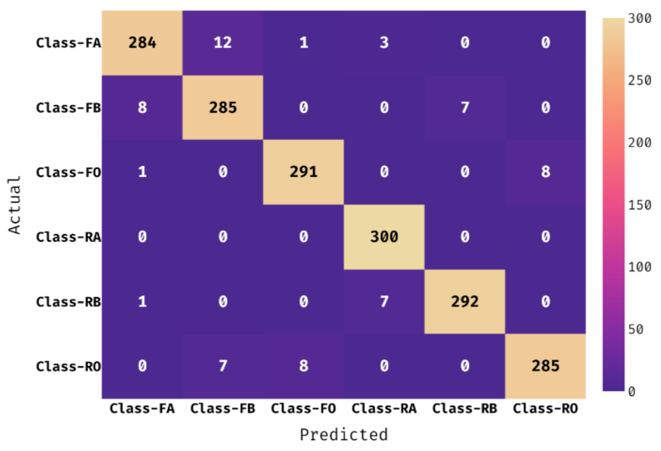


Figure 4. Confusion matrix of MDL-CCIM technique

Table 2 and Figure 5 and Figure 6 provide a brief classification results analysis of the MDL-CCIM technique under distinct classes. The results indicated that the MDL-CCIM technique has gained enhanced classifier results under every individual class. For instance, in FA class, the MDL-CCIM technique has attained $prec_n$, $reca_l$, $accu_y$, and F_{score} of 96.60%, 94.67%, 98.56%, and 95.62% respectively. Simultaneously, with FB class, the MDL-CCIM technique has attained $prec_n$, $reca_l$, $accu_y$, and F_{score} of 93.75%, 95%, 98.11%, and 94.37% respectively.

Concurrently, with FO class, the MDL-CCIM technique has attained $prec_n$, $reca_l$, $accu_y$, and F_{score} of 97%, 97%, 99%, and 97% respectively. At last, with DA class, the MDL-CCIM technique has attained $prec_n$, $reca_l$, $accu_y$, and F_{score} of 96.77%, 100%, 99.44%, and 98.36% respectively.

Table 2. Classification result analysis of MDL-CCIM technique

Methods	Precision	Recall	Accuracy	F-Score
Class-FA	96.60	94.67	98.56	95.62
Class-FB	93.75	95.00	98.11	94.37
Class-FO	97.00	97.00	99.00	97.00
Class-RA	96.77	100.00	99.44	98.36
Class-RB	97.66	97.33	99.17	97.50
Class-RO	97.27	95.00	98.72	96.12
Average	96.51	96.50	98.83	96.50

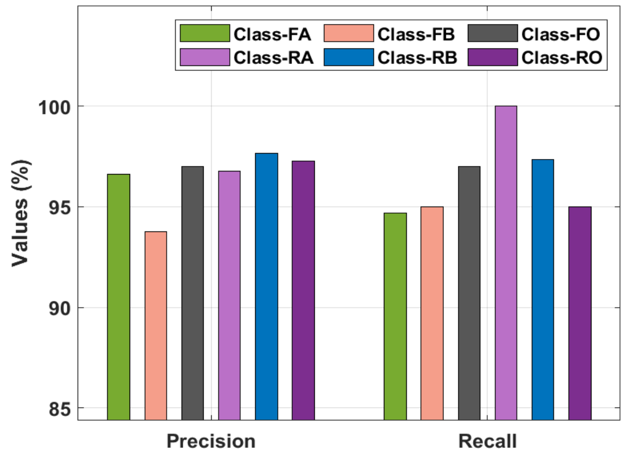


Figure 5. $prec_n$ and $reca_l$ analysis of MDL-CCIM technique

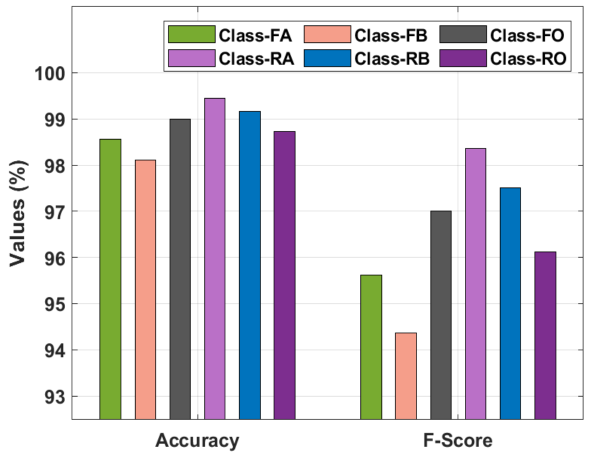


Figure 6. $accu_y$, and F_{score} analysis of MDL-CCIM technique

Table 3 reports the comparative results analysis of the MDL-CCIM technique with the existing DLFFCD technique. Figure 7 offers the comparison study of the MDL-CCIM technique with existing one in terms of $prec_n$. The figure reports that the MDL-CCIM technique has accomplished better values of $prec_n$ under all classes. For instance, with class FA, the MDL-CCIM technique has obtained increased $prec_n$ of 96.60% whereas the DLFFCD technique has attained decreased $prec_n$ of 91.04%.

Moreover, in class Fo, the MDL-CCIM technique has obtained increased $prec_n$ of 97% whereas the DLFFCD technique has attained decreased $prec_n$ of 96.64%. Furthermore, with class RO, the MDL-CCIM technique has obtained increased $prec_n$ of 97.27% whereas the DLFFCD technique has attained decreased $prec_n$ of 96.96%.

Table 3. Comparative result analysis of MDL-CCIM with recent models

Classes	Precision (%)		Recall (%)	
	DLFFCD	MDL-CCIM	DLFFCD	MDL-CCIM
Class-FA	91.04	96.60	93.72	94.67
Class-FB	92.21	93.75	94.5	95.00
Class-FO	96.65	97.00	94.58	97.00
Class-RA	95.61	96.77	95.17	100.00
Class-RB	97.05	97.66	96.43	97.33
Class-RO	96.96	97.27	94.53	95.00

Figure 8 offers the comparison study of the MDL-CCIM technique with existing one in terms of $reca_l$. The figure reports that the MDL-CCIM technique has accomplished better values of $reca_l$ under all classes. For instance, with class FA, the MDL-CCIM technique has obtained increase $reca_l$ of 94.67% whereas the DLFFCD technique has attained decreased $reca_l$ of 93.72%. Moreover, in class Fo, the MDL-CCIM technique has obtained increased $reca_l$ of 97% whereas the DLFFCD technique has attained decreased $reca_l$ of 94.58%. Furthermore, on class RO, the MDL-CCIM technique has obtained increased $reca_l$ of 95% whereas the DLFFCD technique has attained decreased $reca_l$ of 94.53%.

From these results and discussion, the improvement of the MDL-CCIM technique is verified. Thus, it can be utilized as an effective tool for CMOS image sensor compression and classification.

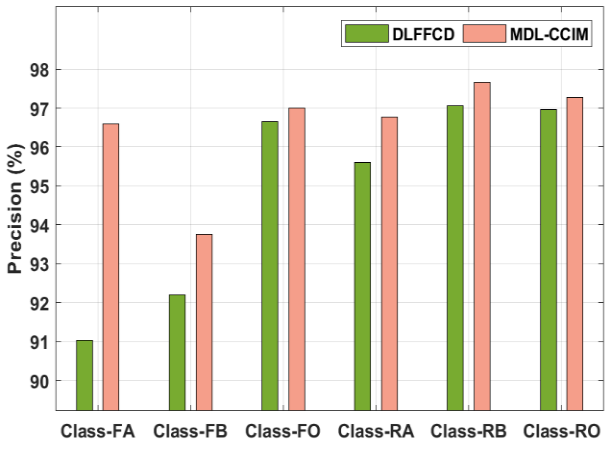


Figure 7. Comparative $prec_n$ analysis of MDL-CCIM with existing models

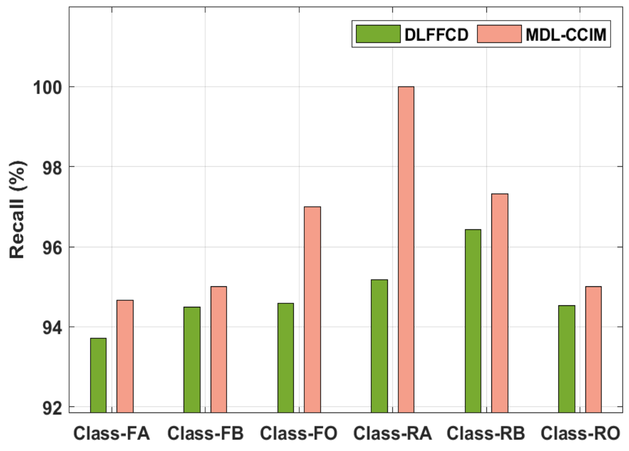


Figure 8. Comparative $reca_l$ analysis of MDL-CCIM with existing models

5. CONCLUSIONS

CMOS image sensors were used in this work to build a novel MDL-CCIM approach for compressing and classifying pictures. Butterfly compression and classification are the two main steps in the proposed MDL-CCIM method. The BOA with LBG model is mostly used for compressing images. Second, a DenseNet with a softmax layer is used to classify images. Finally, the Adam optimizer selects the best hyperparameter setting for the DenseNet model. The MDL-CCIM approach was improved using several simulations. Exhaustive comparison investigation found that the MDL-CCIM technique outperformed other techniques. The MDL-CCIM technique has accomplished better values of $reca_l$ under all classes. For instance, with class FA, the MDL-CCIM technique has obtained increase $reca_l$ of 94.67% whereas the DLFFCD technique has attained decreased $reca_l$ of 93.72%. Moreover, in class Fo, the MDL-CCIM technique has obtained increased $reca_l$ of 97% whereas the DLFFCD technique has attained decreased $reca_l$ of 94.58%. Furthermore, on class RO, the MDL-CCIM technique has obtained increased $reca_l$ of 95% whereas the DLFFCD technique has attained decreased $reca_l$ of 94.53%. In the future, hybrid DL models can be used for image classification purposes. Moreover, planning to evaluate the research with more distinct quality of service parameters.

REFERENCES

[1] Zhang, M., Bermak, A. (2010). CMOS image sensor with on-chip image compression: A review and performance analysis. Journal of Sensors, 2010: 920693. https://doi.org/10.1155/2010/920693

[2] Reyes-Cocoletzi, L., Olmos-Pineda, I., Olvera-López, J.A. (2022). Motion estimation in vehicular environments based on Bayesian dynamic networks. Journal of Intelligent & Fuzzy Systems, 42(5): 4673-4684. https://doi.org/10.3233/JIFS-219255

[3] Tamilvizhi, T., Surendran, R., Romero, C.A.T., Sendil, M.S. (2022). Privacy preserving reliable data transmission in cluster based vehicular adhoc networks. Intelligent Automation & Soft Computing, 34(2): 1265-1279. https://doi.org/10.32604/iasc.2022.026331

[4] Xie, S., Theuwissen, A.J. (2019). On-chip smart temperature sensors for dark current compensation in CMOS image sensors. IEEE Sensors Journal, 19(18): 7849-7860. https://doi.org/10.1109/JSEN.2019.2919655

[5] Shanmugam, G., Thanarajan, T., Rajendran, S., Murugaraj, S.S. (2023). Student Psychology based optimized routing algorithm for big data clustering in IoT with MapReduce framework. Journal of Intelligent & Fuzzy Systems, 1-13. https://doi.org/10.3233/JIFS-221391

[6] Gnanasambandam, A., Chan, S. (2020). One size fits all: Can we train one denoiser for all noise levels?. In International Conference on Machine Learning, pp. 3576-3586. https://doi.org/10.48550/arXiv.2005.09627

[7] Ammah, P.N.T., Owusu, E. (2019). Robust medical image compression based on wavelet transform and vector quantization. Informatics in Medicine Unlocked, 15: 100183. https://doi.org/10.1016/j.imu.2019.100183

[8] Wang, Y., Du, T. (2019). An improved squirrel search algorithm for global function optimization. Algorithms, 12(4): 80. https://doi.org/10.3390/a12040080

[9] Zou, H., Zhao, F., Jia, X., Zhang, H., Wang, W. (2021). Fusion of dynamic predictive block adaptive quantization and vector quantization for staggered SAR data compression. Remote Sensing Letters, 12(2): 199-208. https://doi.org/10.1080/2150704X.2020.1851796

[10] Xu, S., Horng, J.H., Chang, C.C. (2021). Reversible data hiding scheme based on VQ prediction and adaptive parametric binary tree labeling for encrypted images. IEEE Access, 9: 55191-55204. https://doi.org/10.1109/ACCESS.2021.3071819

[11] Varghese, P., Arockia Selva Saroja, G. (2022). An efficient hexagonal image framework using pseudo hexagonal pixel for computer vision applications. Journal of Intelligent & Fuzzy Systems, 42(4): 3879-3892. https://doi.org/10.3233/JIFS-212111

[12] Cappello, F., Di, S., Li, S., et al. (2019). Use cases of lossy compression for floating-point data in scientific data sets. The International Journal of High Performance Computing Applications, 33(6): 1201-1220. https://doi.org/10.1177/109434201985333

[13] Hasan, N., Bao, Y., Shawon, A., Huang, Y. (2021). DenseNet convolutional neural networks application for predicting COVID-19 using CT image. SN Computer Science, 2(5): 389. https://doi.org/10.1007/s42979-021-00782-7

[14] Wang, J. (2023). Classification and identification of garment images based on deep learning. Journal of

Intelligent & Fuzzy Systems, 44(3): 4223-4232. https://doi.org/10.3233/JIFS-220109

[15] Gao, F., Li, B., Chen, L., Shang, Z., Wei, X., He, C. (2021). A softmax classifier for high-precision classification of ultrasonic similar signals. Ultrasonics, 112: 106344. https://doi.org/10.1016/j.ultras.2020.106344

[16] Kadono, T., Hirose, R., Onaka-Masada, A., Kobayashi, K., Suzuki, A., Okuyama, R., Koga, Y., Fukuyama, A., Kurita, K. (2022). Reduction of white spot defects in CMOS image sensors fabricated using epitaxial silicon wafer with proximity gettering sinks by CH2P molecular ion implantation. Sensors, 22(21): 8258. https://doi.org/10.3390/s22218258

[17] Tamilvizhi, T., Surendran, R., Anbazhagan, K., Rajkumar, K. (2022). Quantum behaved particle swarm optimization-based deep transfer learning model for sugarcane leaf disease detection and classification. Mathematical Problems in Engineering, 2022: 3452413. https://doi.org/10.1155/2022/3452413

[18] Duraisamy, K., Thanarajan, T., Alharbi, M. (2022). Implementation of omar pigeon space-time (OPST) algorithm to mitigate the interference and peak-to-average power ratio (PAPR) using RPR mobile and HST-HM in the 5G. Traitement du Signal, 39(5): 1631-1638. https://doi.org/10.18280/ts.390520

[19] Park, S., Lee, C., Park, S., et al. (2022). A 64Mpixel CMOS Image Sensor with 0.50μm unit pixels separated by front deep-trench isolation. In 2022 IEEE International Solid-State Circuits Conference (ISSCC), pp. 1-3. https://doi.org/10.1109/ISSCC42614.2022.9731750

[20] Hussain, R., Alican Noyan, M., Woyessa, G., Retamal Marín, R.R., Antonio Martinez, P., Mahdi, F.M., Finazzi, V., Hazlehurst, T.A., Hunter, T.N., Coll, T., Stintz, M., Muller, F., Chalkias, G., Pruneri, V. (2020). An ultra-compact particle size analyser using a CMOS image sensor and machine learning. Light: Science & Applications, 9(1): 21. https://doi.org/10.1038/s41377-020-0255-6