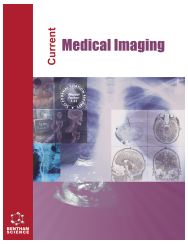
BENTHAM SCIENCE

CrossMark

Current Medical Imaging

Content list available at: https://benthamscience.com/journals/cmir



RESEARCH ARTICLE

Computational Model for the Detection of Diabetic Retinopathy in 2-D Color Fundus Retina Scan

Akshit Aggarwal[1], Shruti Jain[2,*] and Himanshu Jindal[3,*]

[1]*Research Labs, Department of CSE, Indian Institute of Technology, Guwahati, India*
[2]*Department of ECE, Jaypee University of Information Technology, Solan, Himachal Pradesh, India*
[3]*Amity University Punjab, Mohali, India*

Abstract:

Background:

Diabetic Retinopathy *(DR)* is a growing problem in Asian countries. *DR* accounts for 5% to 7% of all blindness in the entire area. In India, the record of *DR*-affected patients will reach around 79.4 million by 2030.

Aims:

The main objective of the investigation is to utilize 2-D colored fundus retina scans to determine if an individual possesses *DR* or not. In this regard, Engineering-based techniques such as deep learning and neural networks play a methodical role in fighting against this fatal disease.

Methods:

In this research work, a Computational Model for detecting *DR* using Convolutional Neural Network (DRCNN) is proposed. This method contrasts the fundus retina scans of the *DR*-afflicted eye with the usual human eyes. Using CNN and layers like Conv2D, Pooling, Dense, Flatten, and Dropout, the model aids in comprehending the scan's curve and color-based features. For training and error reduction, the Visual Geometry Group (VGG-16) model and Adaptive Moment Estimation Optimizer are utilized.

Results:

The variations in a dataset like 50%, 60%, 70%, 80%, and 90% images are reserved for the training phase, and the rest images are reserved for the testing phase. In the proposed model, *the VGG-16* model comprises 138M parameters. The accuracy is achieved maximum rate of 90% when the training dataset is reserved at 80%. The model was validated using other datasets.

Conclusion:

The suggested contribution to research determines conclusively whether the provided OCT scan utilizes an effective method for detecting *DR*-affected individuals within just a few moments.

Keywords: Retina, 2-D fundus, Convolutional neural network, Diabetic retinopathy, Hyperglycemia, NPDR.

Article History	Received: February 08, 2023	Revised: July 28, 2023	Accepted: August 25, 2023

1. INTRODUCTION

DR is a diabetic illness of the eyes and is a problem that occurs in many people who are suffering from diabetes. The first sign of *DR* is often blurred vision. *DR* occurs during small blood vessels blood leaks and retinal damage cause blindness. The main reason behind *DR* is when blood sugar levels are too high for a long period as it has the potential to harm the small blood vessels that provide blood to the retina. It is a disorder in which the blood vessels within the retina are damaged as a result of continuous contact with elevated amounts of hyperglycemia. It happens due to the inability of the body to correctly manage glucose levels. Individuals may notice blind patches in their range of vision as the condition advances.

* Address correspondence to this author at the Amity University Punjab, Mohali, India; E-mail: himanshu19j@gmail.com

Eventually, they may lose vision completely. In India, the rate of DR-affected patients will reach around 79.4 million by 2030 [1, 2]. *DR* is broadly classified into two categories, which are Non-Proliferative *DR* (*NPDR*) and Proliferative *DR* (*PDR*) [3, 4]. In *NPDR,* there is no formation of fresh blood cells, therefore is regarded as the initial phase of *DR*. *NPDR* is easily manageable. In *PDR*, fresh blood vessels are generated in the retina, which is regarded as a challenging stage to manage. DR has at first no signs or symptoms, however, it eventually damages people's eyes. The most common symptoms of DR include floating spots in the vision, fuzzy vision, or a black or empty space in the retina [5]. If a person is suffering from this kind of symptom, then the physician will advise the *OCT* test. The physician checks the scan report and on behalf of this scan, the physician will predict whether the person is suffering from *DR* or not.

1.1. Related Work

This part presents some of the most comprehensive studies in the area of diagnosing DR using neural network approaches in deep learning. This research includes data collection of images, studying the features of Images, normalizing the features, and finding accuracy and efficiency. The efficiency reflects the model's development process, the speed of execution, and the accuracy. Various techniques are presented for efficiencies such as model selection, and dataset of images. Among them, the model selection and accordance with the number of images play a vital role and are discussed.

Gulshan *et al*. [6] created and validated a deep-learning system for detecting DR retinal Fundus. The author took into account the dataset's high worth by achieving an elevated level of responsiveness and specificity to identify attributed DR. The problem with this architecture is that the employed method might not work well for images with minor discoveries that are missed by the vast majority of ophthalmic surgeons. To enhance it, the author, primarily Doshi*et al*., developed DR detection utilizing the Deep CNN model on five stages of the illness depending on severity [7]. The simplest modeled quadratic weighted kappa measurement accuracy in this study is 38.6%, which is rather poor. Few of the authors have extended Doshi's approach by taking fundus photograph images based on the severity [8]. The limitation of Harry *et al's* study is the achieved accuracy is 75% which is quite low and validation images take 188 seconds to run on the *CNN* network which is quite high. Prentasic and Loncaric examined the identification of exudates in fundus pictures using neural networks with deep learning and anatomic landmark recognition fusion to improve it [9]. The investigation achieves accuracy as low as 78%. Bhatia *et al.* customized DR diagnostics utilizing machine learning categorization [10]. The discovery addresses the belief in the presence of diseases after using an ML on feature-extracted retina images. The key disadvantage of this research is that the data input is not in the form of images. Somasundaram and Alli created a Machine Learning Ensemble Classifier (ML-BEC) for early estimation of the DR method [11]. The downside of this approach is that, while it predicts with high accuracy, it doesn't employ raw data. To improve, a number of the writers proposed detecting DR in eye visuals using transferred learning [12, 13]. The authors forecast DR using the Inception V3 network and obtain excellent effectiveness with the greatest accuracy of 48.2%. The key constraint of the proposed study is its low precision. Furthermore, some writers have expanded on Sarfaraz's research by employing an artificial intelligence method for disease staging, especially deep learning for enhanced DR staging [14]. The method employs the GoogleNet model for DR classification and achieves an accuracy of 81%. The biggest shortcoming of this method is that the GoogleNets model requires an excessive amount of time to train for the given data set. Ting [15]. model for DR classification is trained using a large dataset of pictures, and its accuracy is approximately equivalent to that of Takahashi *et al*. [14]. The fundamental shortcoming of the proposed approach is that the visual results are inadequate. Gupta and Chhikara have described the *DR*: Present and Past [16]. The author has used two steps for detecting *DR i.e.,* feature extraction and classification. The main limitation of the description is that there is no proper feature extraction technique and no proper specificity of the designed model. To its improvement, some authors have postulated retinal disease detection using machine learning techniques [17]. The screening system was created by the authors by classifying DR using three distinct classifiers. The model reaches 82.85% accuracy, however, the fundamental limitation of this research is that the data is in raw part form rather than image form. To improve, a neural network model for automated identification of DR has been presented [18]. The researchers employed GoogleNet, AlexNet, and other ImageNet models to increase the model's efficiency, however, this model yields 74.5%, 68.8%, and 57.2% for GoogleNet, AlexNet, and ImageNet models, respectively. The analyzed model's downside is its poor degree of accuracy. Few scholars have built on Carson's work by using deep-learning fundus image analysis for assessing DR and macular degeneration [19]. They created a deep learning-based system that determines if the accessible scan corresponds to the DR class or not. The biggest disadvantage of the intended study is the employed dataset, which comprises non-dilated pupil photographs, which physicians do not suggest. Several writers have improved the work by considering identifying the seriousness of DR using the five-level PIRC scale, however, several elements are lacking in this model [20]. Furthermore, the most current advancement in the detection method has been investigated for the diagnosis of DR [21]. The created method is based on intelligence-based computing expertise and image processing via pixel. The model does not investigate feature learning at the layer level. A few writers have expanded upon the previous work by employing Deep CNN to calculate computer-assisted diagnostic for DR based on images of the fundus [22, 23]. The preciseness of the developed prototype is 86.17%. The fundamental disadvantage of this methodology is that the divided dataset lessons do not contain an equal image. The many causes and clinical consequences of loss to follow-up in PDR individuals have been developed [24]. The investigation assesses the PDR and whether or not the model is adequate for clinical considerations. The key shortcoming of this technique is that they only assessed PDR rather than NPDR. In addition, a few writers have developed a deep learning model based on PCA-Firefly for early identification of DR [25]. The model does dimensionality reduction using the

firefly technique, however, its fundamental shortcoming is that it uses the raw dataset rather than the image dataset. Few writers have contributed to its advancement by developing exudate detection for DR utilizing pre-trained CNN [26]. The method uses the Visual Geometry Group (VGG-19) model to determine whether or not the OCT-based scan is from a DR patient. The approach has an accuracy of around 95%, which is fairly good, but the primary drawback of the approach is that the VGG-19 model adds layers to the approach, resulting in the difficulty of calculating and identifying scan-based fundus retinal.

1.2. Problem Formulation, Motivation, and Objectives

The limitations of most strategies for detecting DR using neural network-based models are excessive complexity, no feature learning for every layer, low accuracy, unbalanced dataset, and no adequate DR stage-wise detection. As a consequence, there is a need to build an efficient approach with suitable feature learning that computes the result in the shortest amount of time while maintaining the reliability of performance metrics. This research paper proposes a computational model for the detection of DR using Convolutional Neural Network (*DRCNN*) in 2D color fundus retina scans. The proposed technique helps in detecting whether the given fundus retina scan is of DR-affected patients or not within a few seconds. *DRCNN* plays an important role in estimating *DR* by a patient when they have their own 2-D fundus retinal scan of the eye while the physician may not have enough time to check the scan. The novelty of this paper lies in the feature extraction technique as no manual work has to be done. The main contribution/objectives are:

- To design a fully automated model for the detection of DR.
- Evaluation of complexity, and other performance parameters.
- Validation of proposed model on another dataset.

Section 2 explains the proposed methodology; Section 3 explains the results and discussion which is followed by a Conclusion and future work, described in Section 4.

2. MATERIALS AND METHODOLOGY

2.1. Materials

In this paper, images are collected from the Messidor dataset [27]. For training, images are randomly selected which are dilated pupil images. These images consist of an equal number of Diabetic Retinas and Normal retinas. The selected images for this research work are in the form of *.jpg* format and all the images of the 2-D color fundus scan are in normal condition having no blurriness or fog. The dataset comprises images with different age categories and ethnicity, and with a wide range of illumination intensities in the eye's retina. The unnecessary variation in classification levels and images with variable pixel levels of intensity leads to the normalization of the data/images. Data normalization makes data analysis and visualization of information in any database more efficient as it removes data anomalies that hamper or complicate the process. Aside from making data analysis more efficient, it can reduce disk space and improve your database's performance. Although various normalization algorithms with different image types have been used in the literature [28 - 30], they have not been used in this work because they may result in the disappearance of images information. To overcome this problem, authors have used Color Normalization (CN). CN is the procedure where mean color transformation is done from one image to another. There are different algorithms for the color normalization of the image like the Structure-Preserving Color Normalization (SPCN), Reinhard method, histogram specification, complete CN, and Stain Color Descriptor (SCD).

2.2. Proposed Methodology

The key goal of this research article is to propose a technique that computes the result in the shortest amount of time and classifies the fundus imaging based on the appropriate DR level of the stage with a high value of performance parameters. This research paper proposes a computational model *DRCNN* that can be used for 2D color fundus retina scans. The proposed model for this research article is depicted in Fig. (**1**).

The resolution of the images is high and has a noteworthy capacity for memory. The data set scaled to 256 × 256 pixels keeps the detailed characteristics and augments the dataset so that algorithms based on deep learning may be utilized. Data Augmentation is a strategy for increasing the quantity of data available by including modified copies of current data or freshly produced artificially generated information from existing databases. When trying with different machine learning models, this technique works as a regularizer and helps in reducing overfitting. Data Augmentation is most popular in the computer vision field. Data Augmentation can be done by padding, vertical & horizontal flipping, the image moved along the x, y, direction, random rotation, zooming, changing contrast, random erasing, darkness & brightness, or color modification, re-scaling, cropping, etc. It is known that deep networks are data-hungry and different augmentation techniques have been used in the literature to solve various problems with deep networks [28 - 30] and to improve the robustness of the methods. In this work, an efficient augmentation method has been used to obtain high performance from the proposed model followed by removal of noise. Noise reduction algorithms are crucial for improving signal/ image processing systems as they help to filter out unwanted background noise. Several techniques in signal processing can be used to reduce noise in an image. Some of the most common techniques include (a) *Image averaging* where multiple images of the same scene are considered and averaged together and the Weighted Average filter smooths stripe data by computing a new location for each pixel. More weight and importance are given to the center value. This can effectively reduce random noise, but it may not work as well for structured noise. (b) *Median filtering* involves replacing each pixel in the image with the median value of the pixels in a neighborhood around it. This can effectively remove salt-and-pepper noise and other types of impulse noise. (c) *Gaussian blurring* involves convolving the image with a Gaussian kernel, which effectively smooths out noise while preserving

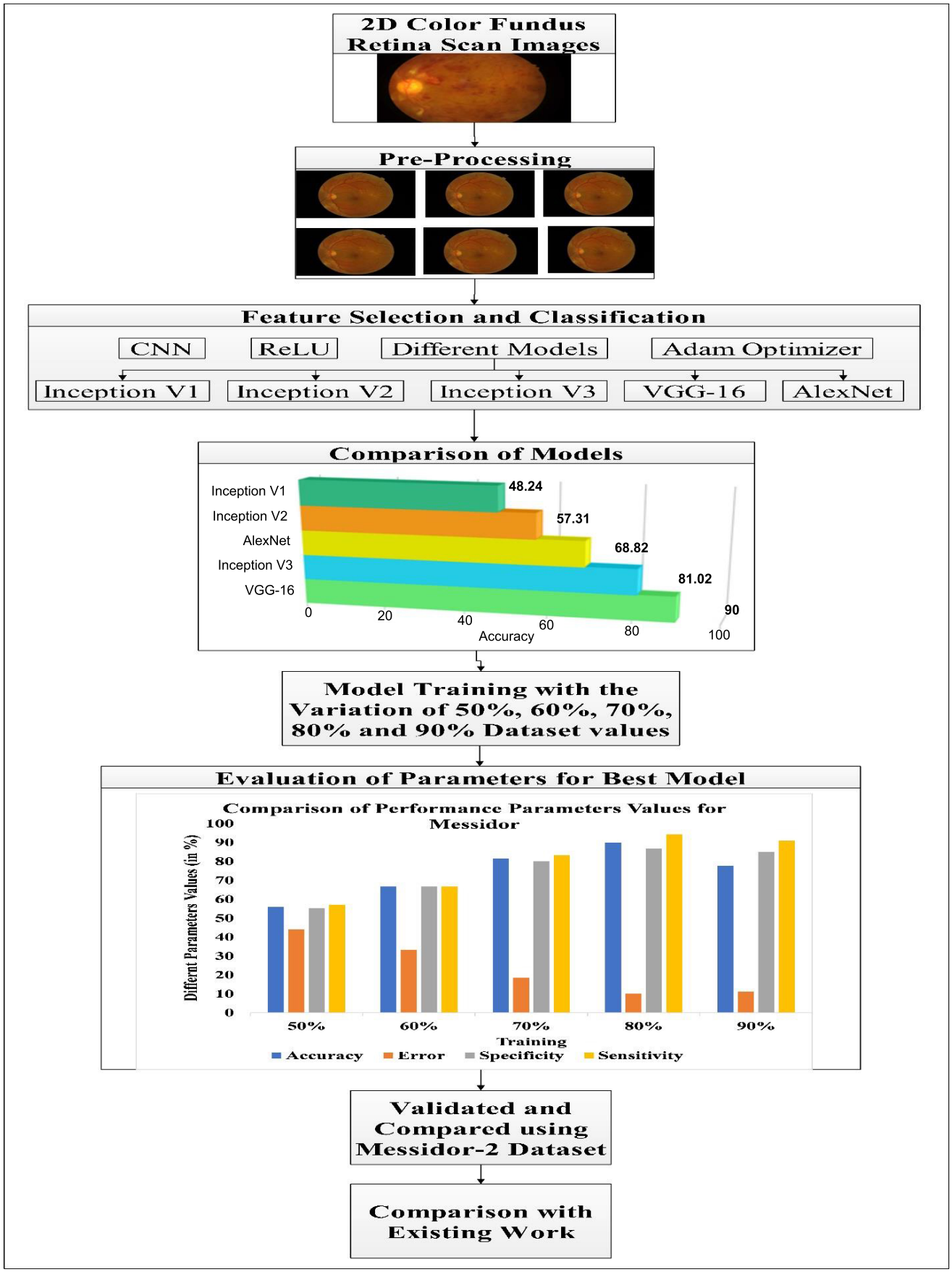


Fig. (1). Proposed computational DRCNN model.

edges. (d) *Non-local means* is analogous to median filtering but uses a weighted average of the pixels in a neighborhood around the target pixel to reduce noise. (e) *Wavelet denoising* decomposes an image into different frequency subbands and then applies thresholding to reduce noise. (f) *Gaussian Filter* is a standard low-pass filter. It replaces every element of the input signal with a weighted average of its neighborhood. The choice of technique will depend on the specific type of noise present in the image and the desired level of noise reduction. In this paper, the authors have used a Weighted average filter and Gaussian filter and are applied to remove noise. The blurred images are not recognized by the physician as well as the *DRCNN* base system. So, in this proposed work non-blurry images are considered for training and testing purposes. Once the dataset of images is gathered, it is converted into a form of 1-D array or in the form of scaling the Pixel [27]. The feature selection is the second phase, where the features of various kinds of images are used with the help of different algorithms like *CNN* and *ReLu*. The formal algorithm is used for the features of images [31]. The latter algorithm helps to convert the image into two color forms [32]. To transform the image into a one-color image and investigate its characteristics, *CNN* and ReLu algorithms are used. Further, to reduce the errors in training the developed model *VGG-16* model is employed [33, 34]. Adam optimizer was used to optimize the feature extraction.

The CNN method is useful for identifying a 2-D color fundus-based scanned image. It helps to study the features of the image. It generates the feature map with the help of its layer (Fig. **2**). The working of *CNN* is described in Algorithm 1.

Algorithm 1 takes all horizontal rows of the input image (a_x) and tries to extract the feature pixel-wise with a kernel filter (γ_x and e). Afterward, each element value is convolute with pixel value with the addition of a temporary accumulator which is initially zero. After the convolutional operation, it generates the feature map ($e \times p_i$). The output image is converted into the 1-D array and helps to extract the feature. The operations for generating a feature map are shown in Fig. (**3**).

The whole process is described in various subsections of *CNN*.

The layers of *CNN* (as shown in Fig. (**4**) is helpful for generating the feature map firstly, the conv2d layer helps produce the tensor of outputs, pooling layer reduces the dimensionality of the input image matrix [35]. Further, the Dropout layer drops some neurons to prevent the *DRCNN* technique from overfitting. The flattened layer flattens the output and generates the feature map. Afterward, the dense layer preserves the robust network that connects the neurons of every single layer to the neurons of the next layer.

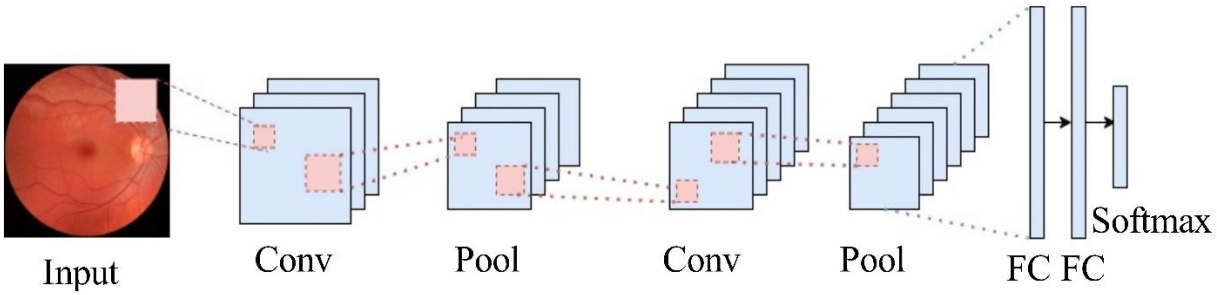


Fig. (2). Flow graph showing different layers of cnn model.

Algorithm 1 *CNN* Algorithm
Initialize the input image a_i {i=1,2,3,…..,n} considering *ac* to be the accumulator, and *e* to be the element value.
1: for each a_x in a_n: // a_x is the input image 2: **for each** p_i **in**p_x: // extracting features pixel-wise 3: **set***ac*=0 // accumulator set to zero 4: **for each** γ_x in γ // kernel filter 5: **for each** e **in** the γ_x 6: **if** e → P_i: // finding each filter in pixel 7: **return***e* * p_i and add the result to ac //generate feature map 8: **end if** 9: **end for** 10: **end for** 11: **end for** 12: **end for** 13: **set**O_i //extracted image in 1-D array

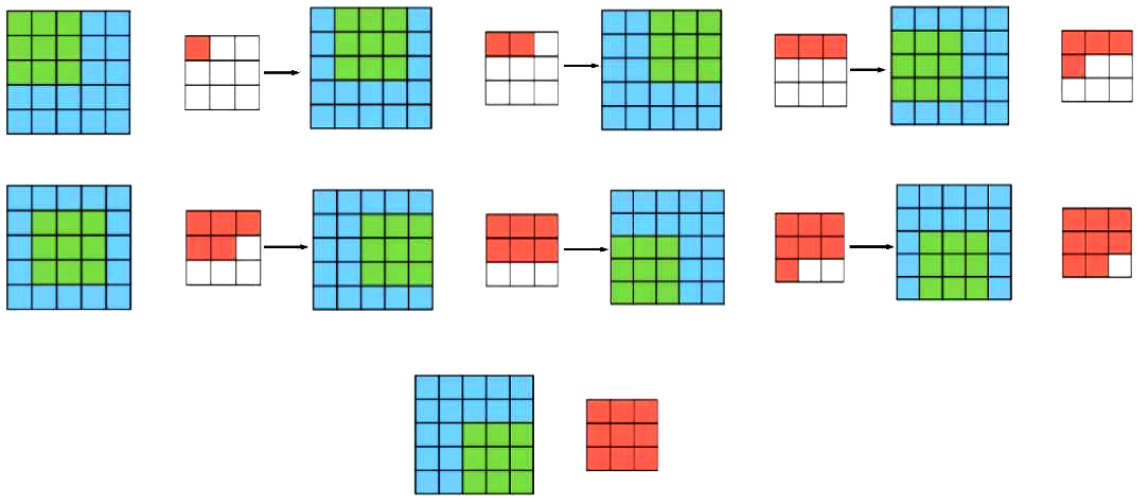


Fig. (3). Operations for generating feature map with the help of convolutional operation.

Fig. (4). Layers in CNN model.

Convolutional 2-D (Conv2D): This layer plays a vital role in generating the convolution kernel, which coils with levels and aids in the production of a stack of outcomes [36]. It helps in transferring the data in the form of the array which is known as tensors and Pseudo-code is shown in Algorithm 2.

Algorithm 2 initially computes the dot result of the kernel and the given input image tensor $a_n = \overrightarrow{a_x} . \overrightarrow{\gamma_x}$ resulting in an output as a single integer. Later, the filter will glide across another suitable field, resulting in the formation of a matrix a_n, stack of outcomes. The output that is fed as an input image in tensor form is depicted in Fig. (**5**).

Algorithm 2 Convolutional -2D Layer **Consider $\overrightarrow{a_x}$, and $\overrightarrow{\gamma_x}$ be the input image vector and kernel filter vector.**
1: for each a_x in a_n: // a_x is the image vector 2: **for each** γ_x in γ: // γ is the kernel vector 3: **set** $a_n = \overrightarrow{a_x} . \overrightarrow{\gamma_x}$ // dot product of receptive fields 4: **repeat** 1 to 3 5: **end for** 6: **end for** 7: **return** a_n //tensor of outputs or stack of outcomes

1	2	1	5	1	5
6	7	0	3	3	3
2	2	0	0	0	4
6	4	5	4	1	2
3	2	1	3	3	2
0	2	1	2	4	4

Fig. (5). Tensor form representation of input image.

Pooling Layer: The CNN's additional structures slice is done using *the pooling layer*. This layer is useful for enlarging or reducing the dimensions of a picture without affecting its pattern or feature [37]. The computation takes average pooling in the layers which is used to take the average value of each sample. Algorithm 3 discusses the pseudo-code for the pooling layer.

Initially, the window size ($w = a_x / \gamma_x$) was calculated and the average of each grid $\left[\left(\frac{a_n - w}{s}\right)+1\right]$ was computed. The vector in dimensionality decrease form is generated by taking the average of each grid and is shown in Fig. (**6**).

Dropout Layer: The next layer, *i.e., the dropout layer,* automatically resets the input units to zero at a given frequency and training rate. The main purpose of this layer is to prevent the *DRCNN*-based technique from overfitting [38]. For this purpose, some of the neurons are dropped. In the proposed work, the 2-D fundus retina scan consists of a black color at the boundary level which is not used for prediction. The black color can cause the model to over-fit. To remove them, the dropout layer is used. Algorithm 4 describes the pseudo-code of the dropout layer.

In Algorithm 4, initially, some neurons were dropped with pre-trained random learning ($a^{(d+1)}$) and later the generated activation function correlates to the output layer, which aids in predicting the scan category. ($f(a^{(d+1)})$).

Flatten layer: It converts the combined trait plot into a single column that is assigned to the completely linked sheet [39]. This layer flattens the convolutional layer's output to produce a single and long characteristic vector, known as a feature map [40]. The feature map is used for pattern recognition for an Image. The pseudo-code for the flattening layer is shown in Algorithm 5.

Algorithm 3 Pooling Layer
Initialize the input image a_i {i=1,2,3,…..,n} considering w be the pooling window size and s be the stride. Set s=1 (initially)
1: for each γ_x in γ: // γ_x is the image vector 2: **for each** γ_x in γ is the kernel vector 3: **set** w = (a_x/γ) // compute window size 4: **perform** $\lfloor\frac{a_n - w}{s}\rfloor + 1$ //average of each grid 5: **end for** 6: **end for**

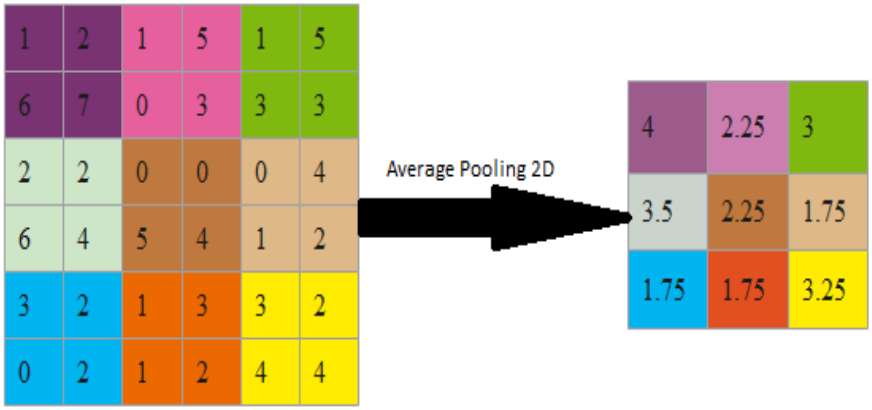


Fig. (6). Average pooling of tensor form image.

Algorithm 4 Dropout Layer
Suppose a neural network has H hidden layers $d \in$ {1,…….,H}.$\vec{d}$ be the vector of input, O^d be the weights and biases, respectively, at layer, d; and $f(a)$ be the activation function
1: for each H: //hidden layers 2: **for each** m: //weights 3: **for each**O: //vector of outputs 4: **set** $a^{(d+1)} = m^{(d+1)}O^H + B^{(d+1)}$ // random learning for neurons finding and dropping 5: **set**$O^{(d+1)} = f(a^{(d+1)})$ //predicting category 6: **end for** 7: **end for** 8: **end for**

Algorithm 5 Flatten Layer
Suppose a_x, a_y be the dimension of the input image, and γ be the kernel filter.
1: for each $\boldsymbol{a_x}$, $\boldsymbol{a_y}$ in $\boldsymbol{a_n}$: //input image matrix 2: **set** (a * γ) [a_x, a_y] //complete image matrix with kernel filter 3: **for** in 1 to n: 4: **for j** in 1 to n: 5: **set** $\sum_i\sum_j\gamma[i, j]a[a_x>-i, a_y-j]$ //feature map 6: **end for** 7: **end for** 8: **end for**

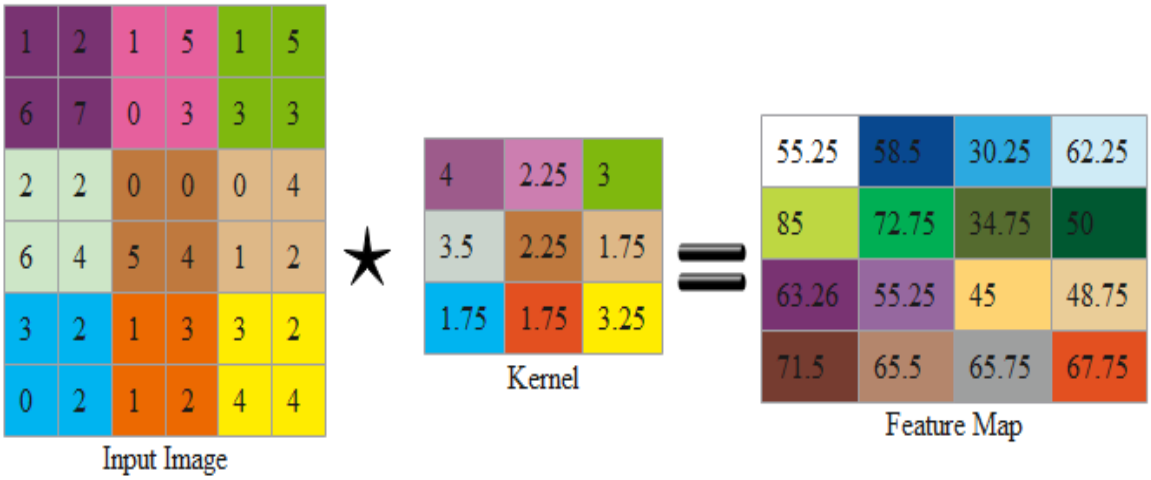


Fig. (7). Generated feature map.

In Algorithm 5, with a kernel filter, it is possible to traverse the input picture matrix in both row and column form [(a x y)(a_x, a_y)]. Following that, the convolutional operation on the input matrix and kernel filter produces another matrix known as a feature map ($\Sigma_i\Sigma_j\gamma[i, j]a[(a_x-i), (a_y-j)]$), and the generated feature map is shown in Fig. (**7**).

Dense Layer: This layer is used after the pooling layer for the mapping of features in a row or a column. The *dense layer* benefits from generating the relation between the convolution networks and the desired category [41]. Each neuron from the previous state is fully coupled to a neuron from the succeeding state in an extensive dense layer. The dense layer is the set of nodes that creates a connection between the layers. The Keras library handles the connection between the layers, automatically. Algorithm 6 discusses the pseudo-code for the dense layer.

Algorithm 6 keeps a strong network between neurons in each layer and neurons in the following layer ($tmp + m[i][j] \times B$[j]) and updates the weight matrix corresponding to each neuron (O[i] = tmp).

Rectified Linear Unit (*ReLu*): This layer is used after every CNN layer [42]. The aim of this model is to initiate nonlinearity into neural network models that are computed as linear. *ReLu* model works only in a single color either black or gray. For gray color, it gives a positive value and zero for black color(as shown in Fig. **8**).

Algorithm 6 Dense Layer
Suppose m is the weight, B is bias, x, and y is the row and column of the feature matrix, and O is the output. Initialize tmp=0
1: for each i in 1 to x: //row of feature matrix 2: **set** tmp=0 //initialize neuron value 3: **for each** j in 1 to y: 4: tmp = tmp+ m[i][j] x B[j] //maintain network of neurons 5: tmp = tmp x B[j] 6: **end for** 7: O[i] = tmp //update weight matrix of each neuron 8: **end for**

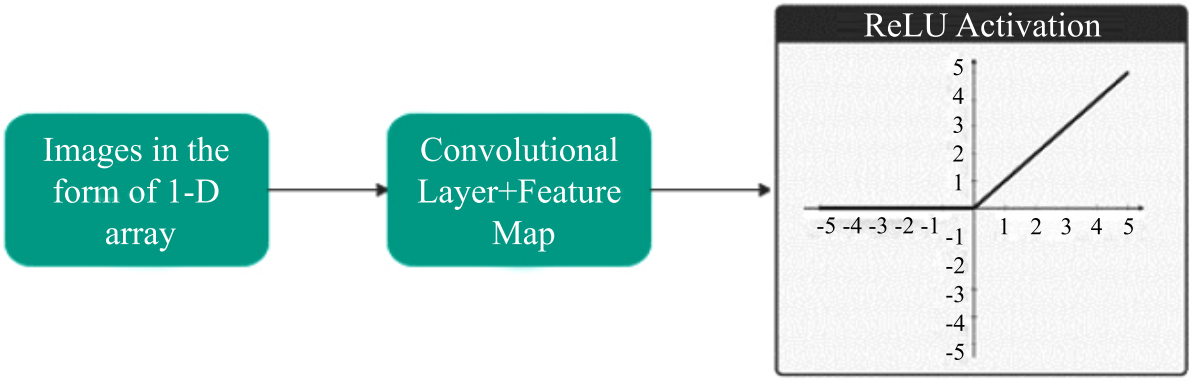


Fig. (8). Architecture of *ReLU* model.

Table 1. *VGG-16* model with training parameters.

Padding and Stride	Input Dimensions (3 × 224 ×224)	Activation Shape	Parameters #
$z = 1, s = 1$	Conv3-64	224 × 224 × 64	(3 × 3 × 3 + 1) 64 + (3 × 3 × 64 + 1) 64
$s = 2$	Pooling-1	112×112×64	0
$z = 1, s = 1$	Conv3-128	112×112×128	(3 × 3 × 64 + 1) × 128 + (3 × 3 × 128 + 1) × 128
$s = 2$	Pooling-2	56×56×128	0
$z = 1, s = 1$	Conv-256	56×56×256	(3 × 3 × 128 + 1) × 256 + (3 × 3 × 256 + 1) × 256 + (3 × 3 × 256 + 1) × 256
$s = 2$	Pooling-3	28×28×256	0
$z = 1, s = 1$	Conv3-512	28 × 28 × 512	(3 × 3 × 256 + 1) × 512 + (3 × 3 × 512 + 1) × 512 + (3 × 3 × 512 + 1) × 512
$s = 2$	Pooling-4	14×14×512	0
$z = 1, s = 1$	Conv3-512	14×14×512	(3 × 3 × 512 + 1) ×512 + (3 × 3 × 512 + 1) × 512 + (3 × 3 × 512 + 1) × 512
$s = 2$	Pooling-5	7×7×512	0
-	FC-4096	(4096,1)	4096 × (7 × 7 × 512) + 4096
-	FC-4096	(4096,1)	4096 × 4096 + 4096
-	FC-4096	(4096,1)	1000 × 4096 + 1000
Total			approximately **138M**

This approach reduced the features of the image. The mathematical approach is expressed in Eq.(**1**).

$$f(a)=\begin{cases}0 & a<0\\ a & a\geq 0\end{cases} \quad \textbf{(1)}$$

Visual Geometry Group-16 Model (VGG-16) is based on Imagenet architecture and increases the number of features that expand the depth of the network [43 - 46]. The *VGG -16* model consists of 3 × 3 meaningful small kernel size and the VGG-*16* network comprises 138 million parameters (as shown in Table **1**).

Table **1** uses the five successive convolutional layers and 3 fully connected layers. The input image is of an *RGB* channel having 224 × 224 dimensions. Table **1** calculates the number of parameters at each layer with the help of Eq. (**2**).

$$parameters = \left[\left(n \times r \times a\right)+1\right] \times F \quad \textbf{(2)}$$

In Eq. (**2**), a is the input feature map of the image and F is the output feature map, n, and r is the width and height of filters respectively. The mathematical approach helps in calculating several parameters. After calculating the number of parameters it shows that the total result is 138 million. For a small number of images, *the VGG-16* model plays an efficient role. Here, a three-channel based *RGB* image of size 224 × 224 passes through the *DRCNN*-based technique (as shown in Fig. **9**). The succession of the 3 × 3 convolutional layer still introduced the non-linearity which leads to the greater discrimination power to the network.

Adaptive Moment Estimation *(Adam)* optimizer was used to minimize the error rate of the proposed model [47]. Adam optimizer is a combination of Root Mean Square *(RMS)* prop and AdaGrad algorithm, so that it can preserve the property of both algorithms [48, 49]. This optimizer supports the power of adaptive learning rate. This algorithm can also run in a noisy environment and gives the best accuracy. Adam optimizer reduces the error and updates the model. The pseudo-code of the Adam optimizer is shown in Algorithm 7.

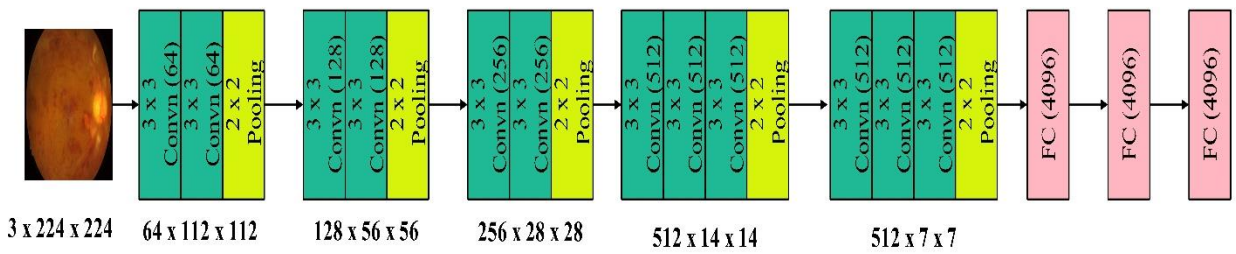


Fig. (9). Different filter sizes of various layers for VGG-16 model.

Algorithm 7 ***Adam Optimizer Reducing Error***

Let α be the step size and β_1, β_2 ε [0,1) be the exponential rates for moment estimates. $f(\theta)$ is the stochastic objective function with parameters θ. θ_0 be the initial parameter vector.
$Ø_0 \leftarrow 0$ (Initialize 1st moment vector)
$v_0 \leftarrow 0$ (Initialize 2nd moment vector)
t ← 0 (Initialize timestamp)

1: while θ_t not converged do //moment vector
2: t ←t + 1
3: $g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ // get gradients
4: $Ø_t \leftarrow \beta_1.Ø_{t-1} + (1-\beta_1).g_t$ //update biased first-moment estimation
5: $v_t \leftarrow \beta_2.v_{t-1} + (1-\beta_2).g_t$ //update biased second-moment estimation
6: $\widehat{\phi_t} \leftarrow \phi_t/(1-\beta_1{}^t)$ // compute bias corrected first-moment estimation
7: $\widehat{v_t} \leftarrow v_t/(1-\beta_2{}^t)$ // compute bias corrected second-moment estimation
8: $\theta_t \leftarrow \theta_{t-1} - \alpha.\widehat{\phi_t}/((sqrt(\widehat{v_t}) + \epsilon)$ //updating parameters
9: **end while**
10: **return** θ_t //Resulting parameters

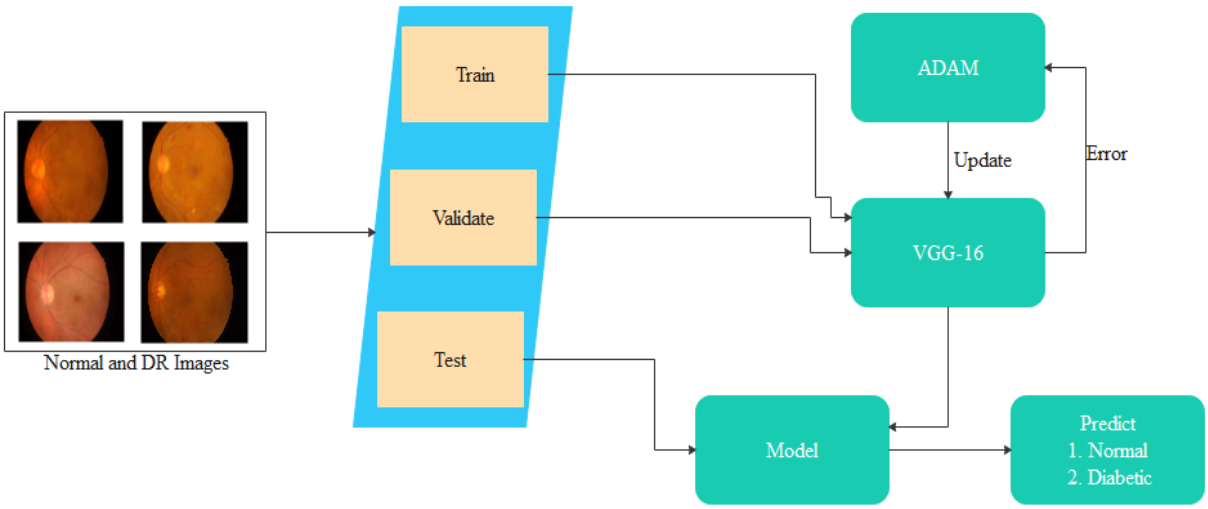


Fig. (10). Working of *adam* optimizer for error optimization.

This algorithm helps in reducing the error rate of the proposed DRCNN technique. The single learning rate was maintained by gradient descent for the updation of weight and there was no alteration in the learning rate during training [50]. The working of Adam is shown in Fig. (**10**).

The DRCNN technique works through the aforementioned layers for feature scanning and extraction. Further, for reducing errors and training, *the VGG-16* model and Adaptive Moment Estimation Optimizer are employed. The outcomes of the proposed technique are described with its results and a discussion on the same is provided.

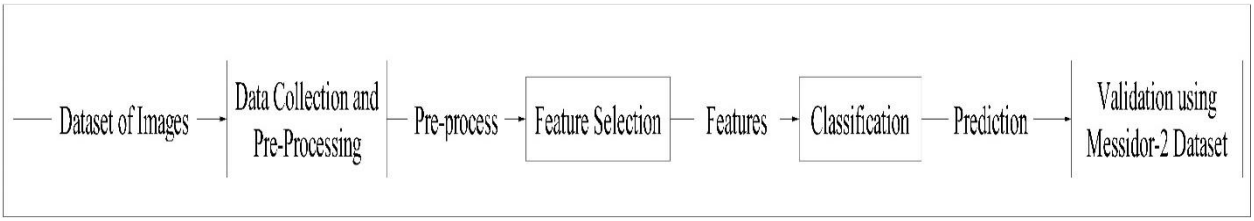


Fig. (11). Overview of DRCNN.

3. RESULTS AND DISCUSSION

DRCNN comprises four phases viz. collection of data & pre-processing, feature selection, classification, and model validation as shown in Fig. (**11**). The images were considered from the Messidor dataset where CNN was considered as feature selection technique that was applied to the *VGG-16* model. The Adam optimizer was used for classification where the model was validated using the Messidor-2 dataset [27, 51].

Experiments are carried out using a 64-bit Windows operating system with an Intel(R) Core (TM) i5-7200U CPU @ 250GHZ Processor. The DRCNN approach is developed by training the model with dataset values varying by 50%, 60%, 70%, 80%, and 90%. The investigation has taken place on a Google Colab, that offers free GPU resources. The suggested model detects the DR using CNN-VGG-16, and the model's efficiency is demonstrated for different training dataset values. Table **2** shows the training setup used to train the model.

Table 2. Different attributes used for training.

Attributes	Values
Number of Epochs	10
Size of Batch	25
Learning Rate	0.001
Multi-processing	False
Shuffling	False
Workers	1
Images Distribution	80% Training and 20% Testing
Models for Pre-Processing	Weighted Filter, Gaussian Filter
Models for Detection	VGG-16

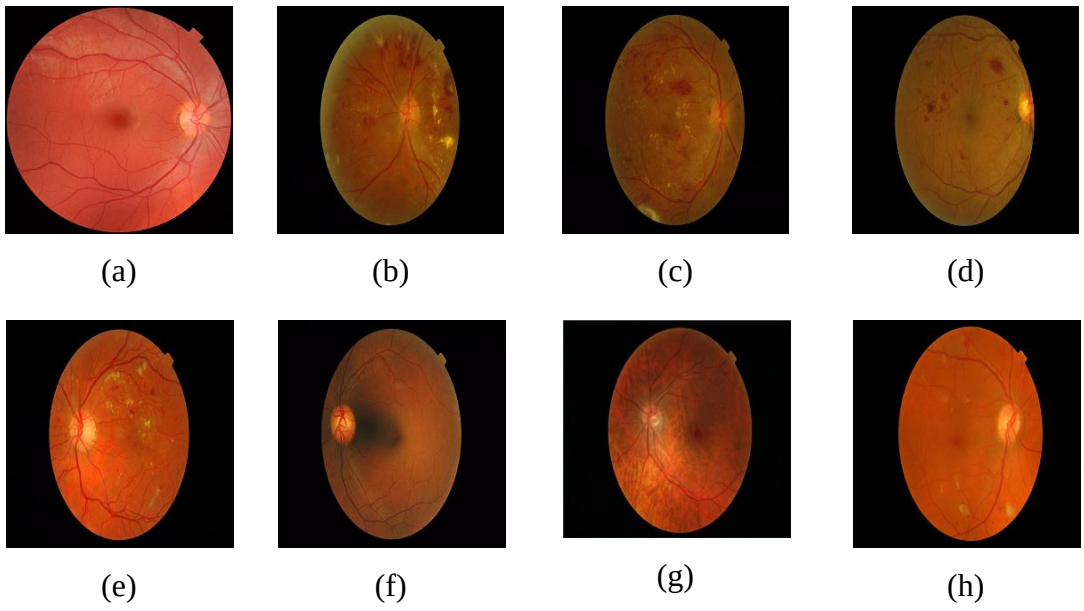


Fig. (12). Input Images of Size 256 × 256 (**a-d**) Images from Messidor Dataset (**e-h**) Images from Messidor-2 Dataset

The DR data is raw data of colored images from the Messidor and Messidor-2 datasets, respectively. To demonstrate and assess the suggested approach, a few images are considered and resized into 256 × 256. These images are shown in Fig. (**12**).

For extracting features of the 2-D fundus scans, the *CNN-VGG-16* algorithm is applied. The important features like color, curve, and edge of the image are studied with the help of *CNN-VGG-16*. The *CNN* algorithm comprises an input layer, hidden layers, and an output layer on which forward as well as backward propagation is applied to maintain the strong network between the layers. The convolutional layers help to create the feature map. The conv-2d layer creates the tensor of the output matrix. Essential library Tensorflow 2.0 is installed on the machine. The tensor of the output matrix is reduced with the help of Average Pooling which takes the average of each grid [52 - 54]. Afterward, the tensor matrix and pooling matrix perform the convolution to generate the feature matrix. Further, the *Relu* model helps introduce nonlinearity as well as converts the image into one color image. The entire dataset of 2-D fundus images is trained with some learning rate. The learning rate helps with weights' updation during training time, which is further referred to as step size.

In this proposed research study, the *DRCNN* technique is trained with different learning rates. The optimal learning rate is considered for this research study. Consequently, the trained fundus dataset is processed through different variations of the Batch normalization technique using the Weighted average filter and Gaussian Filter. This technique helps in increasing the speed of the *DRCNN* technique. Further, with optimal Batch size number of the epoch are calculated which estimates the number of iteration that a learning algorithm will work through the entire training dataset [55].

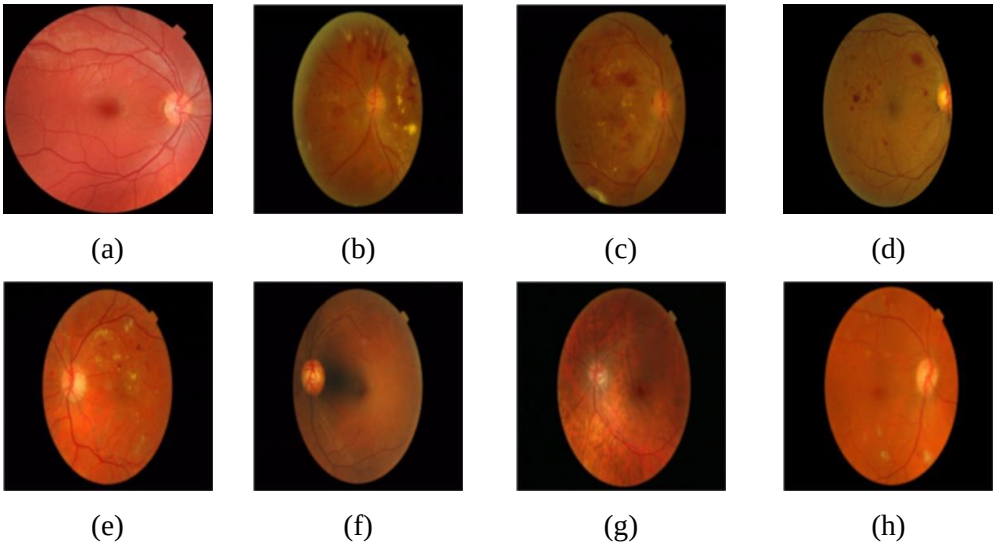


Fig. (13). Output Images after pre-processing using weighted average filter (**a-d**) images from messidor dataset (**e-h**) images from messidor-2 dataset.

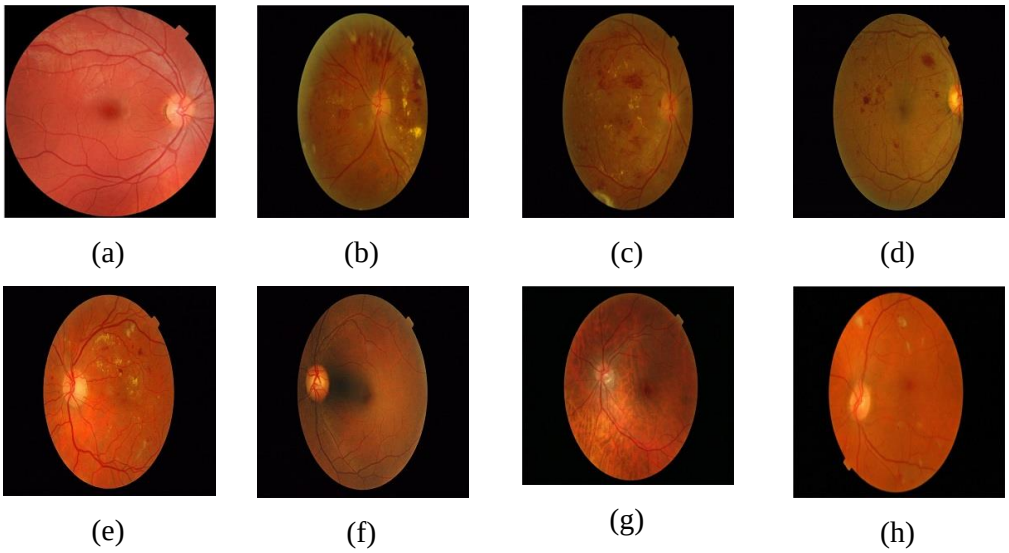


Fig. (14). Output images after pre-processing using gaussian filter (**a-d**) images from messidor dataset (**e-h**) images from messidor-2 dataset.

The dataset contains images with varied levels of lighting. The computation cost for such images is significant, and these images display some blurriness near the edges and maximum data which is not a region of interest; therefore, these images are pre-processed using a weighted average filter and Gaussian Filter. Thus, results generated from pre-processing are shown by each of the implemented methods for weighted average and Gaussian filter are shown in Figs. (**13** and **14**), respectively.

After applying batch normalization i.e. batch size at 25, the model was tested for accuracy on the test dataset. Further, the comparison was done based on test and training accuracies.

If the test accuracy < the training accuracy; fundus images are optimized test accuracy > the training accuracy; the value of batch size, learning rate, and epochs are assigned to the model and work till the model becomes optimized.

The computation time corresponding to training rate estimates is shown in Table **3**.

Since different learning is given to the proposed technique, it is found that, 0.001 is the learning rate which gives the minimum computation time of 94 sec. Therefore, 0.001 is accepted for this study. Afterward, the estimation of the learning rate of the Batch (evaluated using **a** number of samples / Batch Size) is tabulated in Table **4**).

From Table **3**, it can be interpreted that the computation time is observed for different batch sizes. The best results are with a batch size of 25 resulting in 301 sec computation time. The batch size of 25 is fixed for the proposed *DRCNN* technique as it gives minimum computation time. Several epochs are calculated on behalf of batch size. Different pre-trained models like Inception V1, Inception V2, Alex Net, Inception V3, and VGG-16 are used for the evaluation of the accuracy of the dataset. The comparison of accuracy with other existing models is shown in Fig. (**15**).

The achieved accuracy for Inception V1, Inception V2, Alex net, Inception V3, and VGG16 is 48.24%, 57.31%, 68.82%, 81.02%, and 90.00% respectively. Due to the simple network, the *VGG-16* model shows the highest accuracy value which is adopted for the proposed *DRCNN* system. Further, the estimation of the learning rate of the Batch is calculated against different pre-trained models like Inception V1, Inception V2, AlexNet, Inception V3, and VGG-16 keeping the batch size at 25 and the learning rate at 0.001 [50]. The computation is shown in Table **5**.

Table 3. Evaluation of computation time at various learning rates.

S. No.	Learning Rate (α)	Computation Time (in Seconds)
1.	0.1	389
2.	0.01	193
3.	0.001	94

Table 4. Evaluation of computation time at various batch and batch size.

S. No.	Batch Size	Batch	Computation (in sec.)
1.	25	20	301
2.	50	10	481
3.	100	5	602
4.	200	3	999
5.	250	2	1091

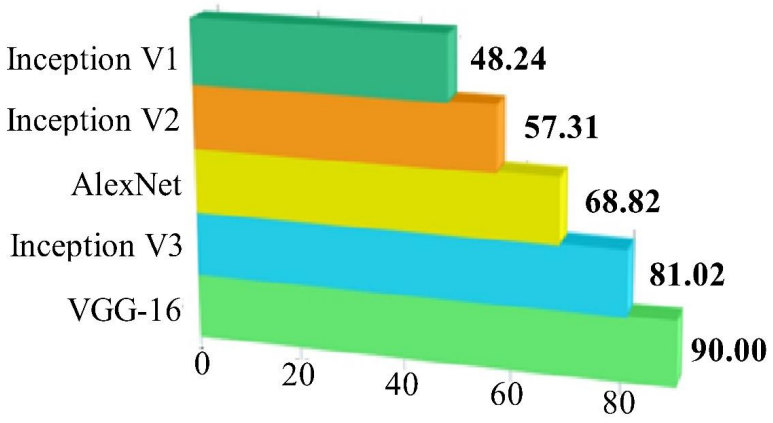


Fig. (15). Evaluation of accuracy for different pre-trained models.

Table 5. Evaluation of computation time for different pre-trained models.

S. No.	Pre-Trained Models	Computation (in sec.)
1.	VGG-16	301
2.	Inception V1	354
3.	Inception V2	498
4.	AlexNet	555
5.	Inception V3	687

Table 6. Comparison of evaluation parameters on the messidor dataset.

Training	Confusion Matrix	Accuracy (in %)	Error (in %)	Specificity (in %)	Sensitivity (in %)
(50%)	Actual / Predicted: 12, 13, 9, 16	56.00	44.00	55.17	57.14
(60%)	Actual / Predicted: 16, 12, 8, 24	66.67	33.33	66.67	66.67
(70%)	Actual / Predicted: 25, 5, 8, 32	81.42	18.57	80.00	83.33
(80%)	Actual / Predicted: 33, 6, 2, 39	**90.00**	**10.00**	**86.67**	**94.28**
(90%)	Actual / Predicted: 30, 8, 12, 40	77.78	11.11	85.10	90.90

In Table **5**, it is shown that concerning computation time, *VGG-16* takes less time and helps in detecting *DR*. As the *VGG-16* model achieves the highest accuracy prediction, the value of other performance parameters like error rate, specificity, and sensitivity is calculated on the *VGG-16* model. Therefore, *VGG-16* is evaluated for its performance on various parameters on the dataset. The suggested model is trained using Messidor dataset values of 50%, 60%, 70%, 80%, and 90%. After forecasting accuracy, the developed model's error rate, specificity, and sensitivity were assessed. Table **6** tabulates the parameters of the accepted model.

Fig. (**16**) compares the results value of performance parameters which is a graphical representation of the simulation of VGG-16 at 50%, 60%, 70%, 80%, and 90% dataset values of Messidor.

For the 80% training set, the maximum accuracy is 90%. Over 90%, 70%, and 60% training sets, respectively, accuracy improves by 13.5%, 9.5%, and 25.9%. Specificity and Sensitivity have been estimated to be 86.67% and 94.28%, respectively. The predicted output for the Messidor dataset is shown in Fig. (**17**).

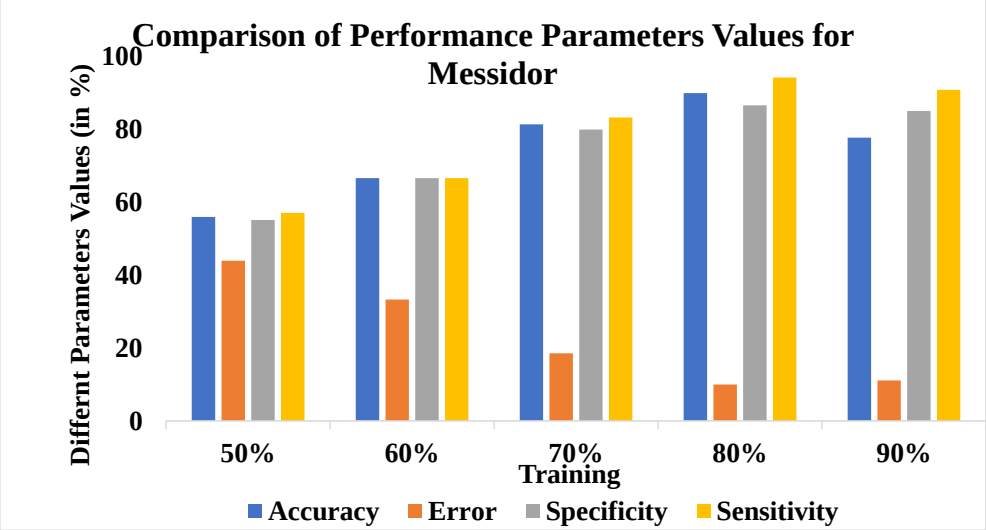


Fig. (16). Comparison of performance parameters values for accuracy, error, specificity, and sensitivity.

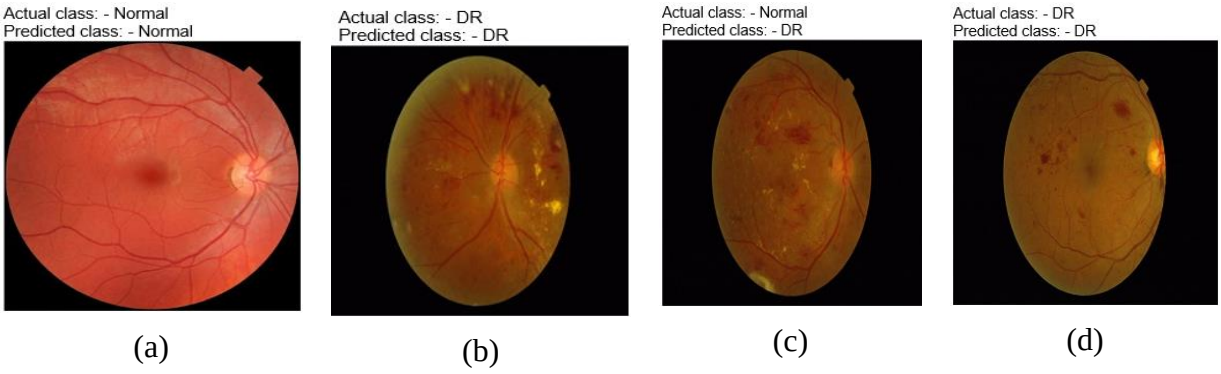


Fig. (17). Actual and predicted class images using VGG-16 from messidor dataset.

3.1. Extension for Feasibility Study

To supplement the experimental results, the suggested model is trained with dataset values varying by 50%, 60%, 70%, 80%, and 90%. of Messidor-2 [51]. Messidor-2 dataset consists of a total of 1748 fundus images of human eyes (both left and right eye) that consist of normal as well as DR-affected scans. The fundus images belong to various ethnicity and age groups [27, 51, 53]. For ease of comparison with the Messidor-2 dataset, images are randomly selected for training and testing perspectives [34, 35, 40]. The randomly selected images consist of an equal number of affected as well as an equal number of normal fundus scans. After predicting accuracy, the error rate, specificity, and sensitivity were calculated on the developed *DRCNN* system. Table **7** tabulates the parameters of the accepted model.

Table **7** compares the results value of performance parameters which is a graphical representation of the simulation of VGG-16 at 50%, 60%, 70%, 80%, and 90% dataset values of the Messidor-2 dataset. Fig. (**18**) compares the results value of performance parameters which is a graphical representation of the simulation of VGG-16 at 50%, 60%, 70%, 80%, and 90% dataset values of Messidor-2.

Table 7. Comparison of evaluation parameters on messidor 2 dataset for VGG-16 model.

Training	Confusion Matrix	Accuracy (in %)	Error (in %)	Specificity (in %)	Sensitivity (in %)
(50%)	Actual / Predicted: 10, 18, 7, 15	50.00	50.00	45.45	58.82

Training	Confusion Matrix	Accuracy (in %)	Error (in %)	Specificity (in %)	Sensitivity (in %)
(60%)	Actual / Predicted: 14, 15, 8, 23	61.66	38.34	60.52	63.63
(70%)	Actual / Predicted: 24, 10, 6, 30	77.14	22.86	75.00	80.00
(80%)	Actual / Predicted: 31, 7, 4, 38	**86.25**	**13.75**	**84.44**	**88.57**
(90%)	Actual / Predicted: 29, 13, 9, 39	75.55	24.45	75.00	76.31

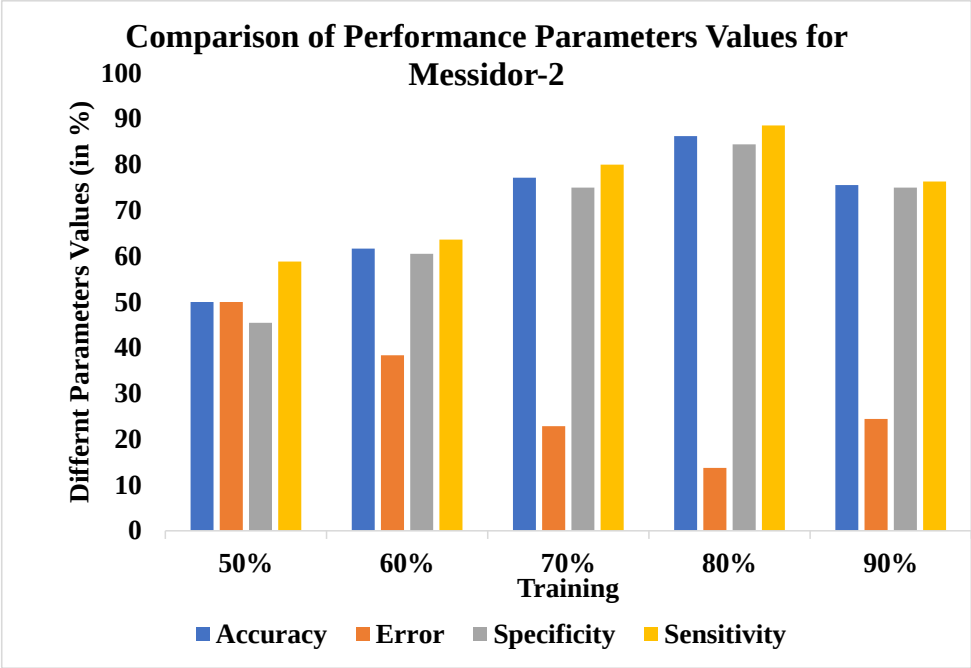


Fig. (18). Comparison of performance parameters values for accuracy, error, specificity, and sensitivity.

For the 80% training set, the maximum accuracy is 86.25%. Over 90%, 70%, and 60% training sets, respectively, accuracy improves by 14.2%, 10.1%, and 29.9%. Specificity and Sensitivity are also reported to be 84.44% and 88.57%, respectively. The predicted output for the Messidor-2 dataset is shown in Fig. (**19**).

Furthermore, the Adam optimizer is utilized to lower the error rate, which iteratively minimizes the rate of error and modifies the model. Thus, Table **8** describes the best simulation parameters of the proposed framework.

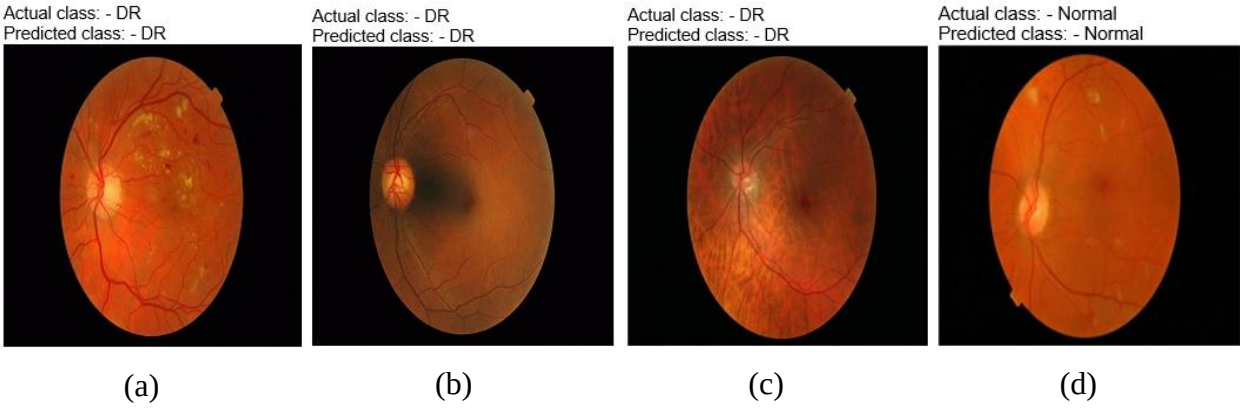


Fig. (19). Actual and predicted class images using VGG-16 from messidor-2 dataset.

Table 8. Simulation parameters of the proposed framework.

S. No	Attributes	Values	Specification
1.	Model	VGG-16	*VGG-16* is a 16-layer architecture that is used to train the *DRCNN* algorithm.
2.	Optimizer	Adam	The Adam optimizer was used to reduce error rates, and it can also function in a noisy setting.
3.	Learning rate	0.001	This rate helps in updating the weights at the time of training.
4.	Batch Size	25	It permits computational speedups from the parallelism GPUs.
5.	Epochs	2	It helps in training the NN for one cycle with all training data.
6.	Accuracy	90.00% and 86.25%	Calculated as: $\frac{TruePositive + TrueNegative}{TotalNumberofSamples}$
7.	Error	10.00% and 13.75%	Calculated as: $\frac{FalsePositive + FalseNegative}{TotalNumberofSamples}$
8.	Specificity	86.67% and 84.44%	Calculated as: $\frac{TrueNegative}{TrueNegative + FalsePositive}$
9.	Sensitivity	94.28% and 88.57%	Calculated as: $\frac{TruePositive}{TruePositive + FalseNegative}$

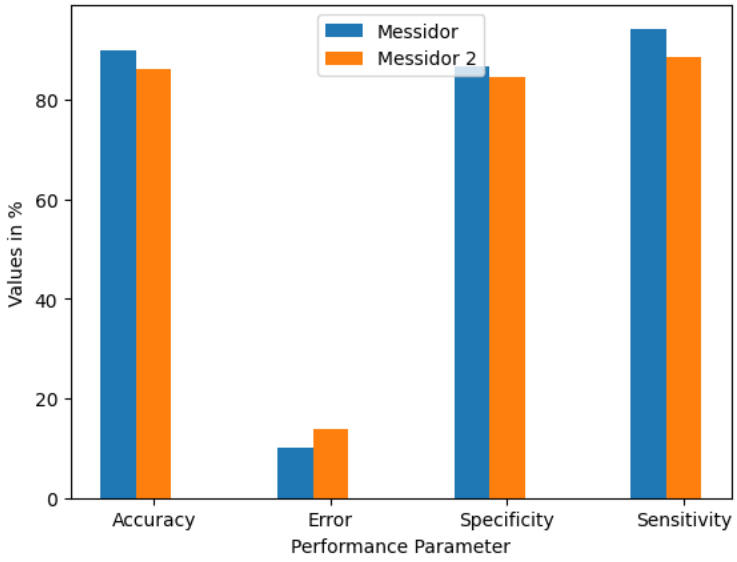


Fig. (20). Comparison of performance parameter for various datasets.

3.2. Comparison of Performance Parameters on Different Datasets

The proposed model is compared for its performance parameters for Messidor and Messidor-2 datasets' values observed for Accuracy, Error, Sensitivity, and Specificity at an 80% training rate and is shown in Fig. (**20**).

It is observed from Fig. (**20**), that VGG-16 outperforms well for the Messidor dataset and is best suited for the detection of DR.

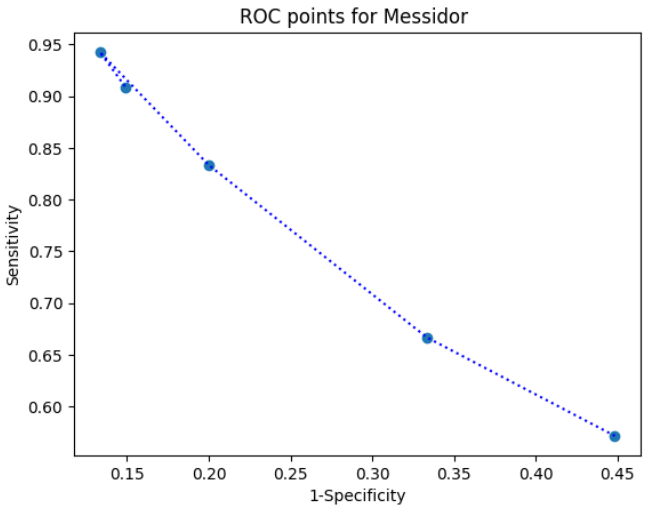


Fig. (21). *ROC* curve for messidor dataset.

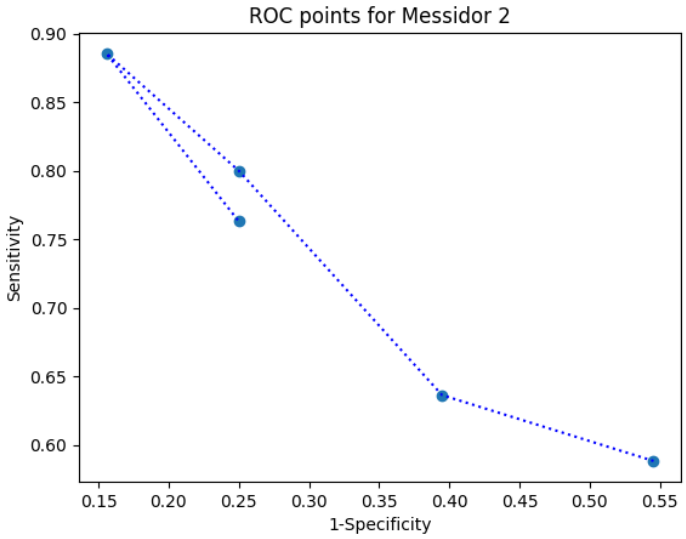


Fig. (22). *ROC* curve for messidor-2 dataset.

Table 9. Comparison with existing work.

Models	Accuracy
Proposed DRCNN with Training (80%)	**90%**
[9]	75%
[10]	78%
[18]	74.5%
[28 - 30] (96.25% for Skin)	92%
[52]	86%
[54]	83%

For performance measurement for *DR* and normal retina images, an *ROC* Curve is plotted for the Messidor and Messidor-2 dataset and is shown in Figs. (**21** and **22**), respectively.

The curve is plotted between the true positive rate which is known as Sensitivity, and the false positive rate which is known as 1- Specificity. The *ROC* curve is plotted for the testing purpose of the usefulness of test cases. The area under the *ROC* curve is high which is very near to 1 which shows that the model can predict the images very efficiently and the model is an excellent classifier.

3.3. Comparison with Existing Work

The suggested model is compared to previous work, which is reported in Table **9**.

The suggested DRCNN model achieves 90% accuracy, whereas [9, 10, 18, 28 - 30, 52, 54] achieve 75%, 78%, 74.5%, 92%, 86%, and 83% accuracy, respectively. 16.6%, 13.3%, and 16.8% accuracy improvement is obtained over [9, 10], and [18], respectively. A 1.2% accuracy decrease is observed as compared with [28 - 30]. 4.2% and 7.9% accuracy improvement are obtained over [52], and [54], respectively. As a result, it is concluded that DRCNN identifies and predicts DR effectively.

CONCLUSION AND FUTURE WORK

The developed *DRCNN* model helps to detect the 2-D color fundus retina-based scan, in*DR*-affected patients. *DRCNN* technique is developed which detects whether a person is suffering from *DR* or not. With the help of *CNN* and *ReLU*, the model helps to understand the feature of the scan which is very significant for prediction. The achieved accuracy for Inception V1, Inception V2, Alex net, Inception V3, and VGG16 is 48.24%, 57.31%, 68.82%, 81.02%, and 90.00% respectively. The maximum classification accuracy is obtained using VGG16 which results in a 9.9% accuracy improvement in comparison to Inception V3. The suggested model is trained with dataset values varying by 50%, 60%, 70%, 80%, and 90%. The maximum accuracy, error, specificity, and sensitivity of *DRCNN* techniques are 90.00%, 10.00%, 86.67%, and 94.28% respectively when 80% of the data is reserved for the training phase and the remaining 20% data is reserved for the testing phase considering VGG-16 model. Over 90%, 70%, and 60% of training sets, respectively, accuracy improvements of 13.5%, 9.5%, and 25.9% were attained, for the Messidor dataset. Further to validate the model, it is implemented for the Messidor-2 dataset. The maximum accuracy of 86.25% is obtained for the 80% training set for Messidor-2. 14.2%, 10.1%, and 29.9% accuracy improvement is obtained over the 90% 70%, and 60% training set, respectively. Thus, if a person has a fundus scan then it can be detected in a few seconds using the proposed model. In the future, to extend this proposed model which can also detect other eyes-related diseases, and convert this proposed model into platform-independent model.

LIST OF SYMBOLS

$\mathbf{a_n}$ = Input Image

$\mathbf{m_n}$ = Weight Corresponds to each layer

I = Input Layer

H = Hidden Layer

f(a) = Activation Function

B = Biasing Index

$\mathbf{O_n}$ = Output Corresponds to Input Image

a^d = Vector on Input Layer

m^d = Vector Corresponds to Weight

B^d = Biasing Vector

O^d = Output Vector

w = Polling Window Size

s = Stride

$\mathbf{a_x}$ = Dimension of Input Image along Horizontal Direction

$\mathbf{a_y}$ = Dimension of Input Image along Vertical Direction

γ = Kernel

$\mathbf{p_i}$ = Input pixel

$\mathbf{o_i}$ = Output image pixel

$\mathbf{p_x}$ = Pixel row

γ_x = Kernel row

z = Padding

n = Width of Convolutional filter

r = Height of Convolutional filter

LIST OF ABBREVIATIONS

Adam	=	Adaptive Moment Estimation
RGB	=	Red Green Blue
CN	=	Color Normalization
CNN	=	Convolutional Neural Network
Conv2d	=	Convolutional 2-Dimesional
DR	=	Diabetic Retinopathy
DRuNN	=	Diabetic Retinopathy Using Neural Network
GUI	=	Graphic User Interface
NPDR	=	Non-Proliferative Diabetic Retinopathy
OCT	=	Optical Coherence Tomography
PDR	=	Proliferative Diabetic Retinopathy
ReLu	=	Rectified Linear Unit
RMS	=	Root Mean Square
ROC	=	Receiver Operating Characteristic
VGG	=	Visual Geometry Group

ETHICS APPROVAL AND CONSENT TO PARTICIPATE

Not applicable.

HUMAN AND ANIMAL RIGHTS

Not applicable.

CONSENT FOR PUBLICATION

Not applicable.

AVAILABILITY OF DATA AND MATERIALS

Not applicable.

FUNDING

None

CONFLICT OF INTEREST

The authors declare no conflict of interest financial or otherwise.

ACKNOWLEDGEMENTS

Declared none.

REFERENCES

[1] Wild S, Roglic G, Green A, Sicree R, King H. Global prevalence of diabetes: estimates for the year 2000 and projections for 2030. Diabetes Care 2004; 27(5): 1047-53. [http://dx.doi.org/10.2337/diacare.27.5.1047] [PMID: 15111519]

[2] Maureen A. Eye health: Anatomy of the eye", [Person's Designation:] M.S., CVRT 2022. Available from: https://network.aphconnectcenter.org/wp-content/uploads/2019/11/3484.jpg

[3] Bhardwaj C, Jain S, Sood M. Transfer learning based robust automatic detection system for diabetic retinopathy grading. Neural Comput Appl 2021; 33(20): 13999-4019. [http://dx.doi.org/10.1007/s00521-021-06042-2]

[4] Bhardwaj C, Jain S, Sood M. Deep learning–based diabetic retinopathy severity grading system employing quadrant ensemble model. J Digit Imaging 2021; 34(2): 440-57. [http://dx.doi.org/10.1007/s10278-021-00418-5] [PMID: 33686525]

[5] Bhardwaj C, Jain S, Sood M. Hierarchical severity grade classification of non-proliferative diabetic retinopathy. J Ambient Intell Humaniz Comput 2021; 12(2): 2649-70. [http://dx.doi.org/10.1007/s12652-020-02426-9]

[6] Gulshan V, Peng L, Coram M, *et al.* Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. JAMA 2016; 316(22): 2402-10. [http://dx.doi.org/10.1001/jama.2016.17216] [PMID: 27898976]

[7] Doshi D, Shenoy A, Sidhpura D, Gharpure P. Diabetic retinopathy detection using deep convolutional neural networks 2016 International Conference on Computing, Analytics and Security Trends (CAST). 261-6. [http://dx.doi.org/10.1109/CAST.2016.7914977]

[8] Pratt H, Coenen F, Broadbent DM, Harding SP, Zheng Y. Convolutional neural networks for diabetic retinopathy. Procedia Comput Sci 2016; 90: 200-5. [http://dx.doi.org/10.1016/j.procs.2016.07.014]

[9] Prentašić P, Lončarić S. Detection of exudates in fundus photographs using deep neural networks and anatomical landmark detection fusion. Comput Methods Programs Biomed 2016; 137: 281-92. [http://dx.doi.org/10.1016/j.cmpb.2016.09.018] [PMID: 28110732]

[10] Bhatia K, Arora S, Tomar R. Diagnosis of diabetic retinopathy using machine learning classification algorithm 2016 2nd International Conference on Next Generation Computing Technologies (NGCT), IEEE. 347-51. [http://dx.doi.org/10.1109/NGCT.2016.7877439]

[11] S K S, P A. A machine learning ensemble classifier for early prediction of diabetic retinopathy. J Med Syst 2017; 41(12): 201. [http://dx.doi.org/10.1007/s10916-017-0853-x] [PMID: 29124453]

[12] Masood S, Luthra T, Sundriyal H, Ahmed M. Identifcation of diabetic retinopathy in eye im-ages using transfer learning 2017 International Conference on Computing, Communication and Automation (ICCCA). 1183-7. [http://dx.doi.org/10.1109/CCAA.2017.8229977]

[13] Garg S, Jindal H. Evaluation of time series forecasting models for estimation of PM2. 5 Levels in Air. Proceedings of 6th International Conference for Convergence in Technology (I2CT). 1-8.

[14] Takahashi H, Tampo H, Arai Y, Inoue Y, Kawashima H. Applying artificial intelligence to disease staging: Deep learning for improved staging of diabetic retinopathy. PLoS One 2017; 12(6): e0179790. [http://dx.doi.org/10.1371/journal.pone.0179790] [PMID: 28640840]

[15] Ting DSW, Cheung CYL, Lim G, *et al.* Development and validation of a deep learning system for diabetic retinopathy and related eye diseases using retinal images from multiethnic populations with diabetes. JAMA 2017; 318(22): 2211-23. [http://dx.doi.org/10.1001/jama.2017.18152] [PMID: 29234807]

[16] Gupta A, Chhikara R. Diabetic retinopathy: Present and past. Procedia Comput Sci 2018; 132: 1432-40. [http://dx.doi.org/10.1016/j.procs.2018.05.074]

[17] Pawar PM, Agrawal AJ. Retinal disease detection using machine learning techniques. HELIX 2018; 8: 3932-7. [http://dx.doi.org/10.29042/2018-3932-3937]

[18] Lam C, Yi D, Guo M, Lindsey T. Automated detection of diabetic retinopathy using deep learning. AMIA Jt Summits Transl Sci Proc 2018; 2017: 147-55. [PMID: 29888061]

[19] Sahlsten J, Jaskari J, Kivinen J, *et al.* Deep learning fundus image analysis for diabetic retinopathy and macular edema grading. Sci Rep 2019; 9(1): 10750. [http://dx.doi.org/10.1038/s41598-019-47181-w]

[20] Ruamviboonsuk P, Krause J, Chotcomwongse P, *et al.* Author Correction: Deep learning *versus* human graders for classifying diabetic retinopathy severity in a nationwide screening program. NPJ Digit Med 2019; 2(1): 68. [http://dx.doi.org/10.1038/s41746-019-0146-5] [PMID: 31341955]

[21] Qureshi I, Ma J, Abbas Q. Recent development on detection methods for the diagnosis of diabetic retinopathy. Symmetry (Basel) 2019; 11(6): 749. [http://dx.doi.org/10.3390/sym11060749]

[22] Li YH, Yeh NN, Chen SJ, Chung YC. Computer-assisted diagnosis for diabetic retinopathy based on fundus images using deep convolutional neural network. Mob Inf Syst 2019; 2019: 1-14. [http://dx.doi.org/10.1155/2019/6142839]

[23] Kaur M, Bharti M. Fog computing providing data security: A review. Int J Comput Sci Eng 2014; 4(6)

[24] Abdelmotaal H, Ibrahim W, Sharaf M, Abdelazeem K. Causes and clinical impact of loss to follow-up in patients with proliferative diabetic retinopathy. J Ophthalmol 2020; 2020: 1-8. [http://dx.doi.org/10.1155/2020/7691724] [PMID: 32089871]

[25] Gadekallu TR, Khare N, Bhattacharya S, *et al.* Early detection of diabetic retinopathy using pca-_rey based deep learning model. Electronics (Basel) 2020; 9(2): 274. [http://dx.doi.org/10.3390/electronics9020274]

[26] Mateen M, Wen J, Nasrullah N, Sun S, Hayat S. Exudate detection for diabetic retinopathy using pretrained convolutional neural networks. Complexity 2020; 2020: 1-11. [http://dx.doi.org/10.1155/2020/5801870]

[27] Decencière E, Zhang X, Cazuguel G, *et al.* Feedback on a publicly distributed image database: The messidor database. Image Anal Stereol 2014; 33(3): 231-4. [http://dx.doi.org/10.5566/ias.1155]

[28] Goceri E. Classification of skin cancer using adjustable and fully convolutional capsule layers. Biomed Signal Process Control 2023; 85: 104949. [http://dx.doi.org/10.1016/j.bspc.2023.104949]

[29] Goceri E. Capsule neural networks in classification of skin lesions. Proceeding of International Conference on Computer Graphics, Visualization, Computer Vision and Image Processing. 29-36.

[30] Goceri E. Analysis of capsule networks for image classification. Proceeding of International Conference on Computer Graphics, Visualization, Computer Vision and Image Processing. 53-60.

[31] Higuchi T, Yoshifuji N, Sakai T, *et al.* Clpy: A numpy compatible library accelerated with opencl 2019 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW). 933-40. [http://dx.doi.org/10.1109/IPDPSW.2019.00159]

[32] Chauhan R, Ghanshala KK, Joshi R. Convolutional neural network (cnn) for image detection and recognition 2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC). 278-82. [http://dx.doi.org/10.1109/ICSCCC.2018.8703316]

[33] Bharti M, Jindal H. Automatic rumour detection model on social media. Proceedings of Sixth International Conference on Parallel, Distributed and Grid Computing (PDGC). 367-71. [http://dx.doi.org/10.1109/PDGC50313.2020.9315738]

[34] Jindal H, Bharti M, Kasana SS, Saxena S. An ensemble mosaicing and ridgelet based fusion technique for underwater panoramic image reconstruction and its refinement. Multimedia Tools Appl 2023; 82(22): 33719-71.

[http://dx.doi.org/10.1007/s11042-023-14594-9]

[35] Singh PK, Sood S, Kumar Y, Paprzycki M, Pljonkin A, Hong W-C. Futuristic trends in networks and computing technologies. Proceedings of Second International Conference, FTNCT 2019. Chandigarh, India. Springer Nature 2020.November 22–23, 2019; 1206 Revised Selected Papers

[36] Chang J, Sha J. An efficient implementation of 2d convolution in cnn. IEICE Electron Express 2016; 13: 20161134.

[37] Ahmadi M, Vakili S, Langlois JP, Gross W. Power reduction in cnn pooling layers with a preliminary partial computation strategy 2018 16th IEEE International New Circuits and Systems Conference (NEWCAS), IEEE125-9.
[http://dx.doi.org/10.1109/NEWCAS.2018.8585433]

[38] Ko B, Kim H-G, Oh K-J, Choi H-J. Controlled dropout: A different approach to using dropout on deep neural network 2017 IEEE International Conference on Big Data and Smart Computing (BigComp). 358-62.

[39] Li M, Wang H, Yang J. Flattening and preferential attachment in the internet evolution 2012 14th Asia-Pacific Network Operations and Management Symposium (APNOMS), IEEE. 1-8.

[40] Bharti M, Jindal H. Optimized clustering-based discovery framework on Internet of Things. J Supercomput 2021; 77(2): 1739-78.
[http://dx.doi.org/10.1007/s11227-020-03315-w]

[41] Huang G, Liu Z, Van Der Maaten L, Weinberger KQ. Densely connected convolutional networks Proceedings of the IEEE conference on computer vision and pattern recognition. 4700-8.

[42] Douglas SC, Yu J. Whyrelu units sometimes die: Analysis of single-unit error backpropagation in neural networks 2018 52nd Asilomar Conference on Signals, Systems, and Computers, IEEE. 864-8.

[43] Wuth SN, Coetzee R, Levitt SP. Creating a python gui for a c++ image processing library. 2004 IEEE Africon 7th Africon Conference in Africa (IEEE Cat No 04CH37590). IEEE; 2004; pp. 2: 1203-6.

[44] Qassim H, Verma A, Feinzimer D. Compressed residual-vgg16 cnn model for big data places image recognition 2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC). 169.
[http://dx.doi.org/10.1109/CCWC.2018.8301729]

[45] Deng J, Dong W, Socher R, Li L-J, Li K, Fei-Fei L. Imagenet: A large-scale hierarchical image database In: 2009 IEEE conference on computer vision and pattern recognition, Ieee. 2009; pp. 248-55.
[http://dx.doi.org/10.1109/CVPR.2009.5206848]

[46] Arya S, Singh R. A comparative study of cnn and alexnet for detection of disease in potato and mango leaf 2019 International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT). volume 1: 1-6.
[http://dx.doi.org/10.1109/ICICT46931.2019.8977648]

[47] Zhang Z. Improved adam optimizer for deep neural networks 2018 IEEE/ACM 26th International Symposium on Quality of Service (IWQoS). 1-2.
[http://dx.doi.org/10.1109/IWQoS.2018.8624183]

[48] Dogo E, Afolabi O, Nwulu N, Twala B, Aigbavboa C. A comparative analysis of gradient descent based optimization algorithms on convolutional neural networks 2018 International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS). 92-9.
[http://dx.doi.org/10.1109/CTEMS.2018.8769211]

[49] Jindal H, Saxena S. Kasana Sewage Water Quality Monitoring Framework Using Multi-parametric Sensors, Wireless Personal Communications, Springer 2017; 97(1): 881-913.

[50] Jindal H, Saxena S, Kasana SS. Triangular pyramidal topology to measure temporal and spatial variations in shallow river water using Ad-hoc sensors network'. Ad Hoc Sens Wirel Netw 2017; 39(1-4): 1-35.

[51] Abràmoff MD, Folk JC, Han DP, *et al.* Automated analysis of retinal images for detection of referable diabetic retinopathy. JAMA Ophthalmol 2013; 131(3): 351-7.
[http://dx.doi.org/10.1001/jamaophthalmol.2013.1743] [PMID: 23494039]

[52] Wang L, Shoulin Y, Alyami H, *et al.* A novel deep learning□based single shot multibox detector model for object detection in optical remote sensing images. Geosci Data J 2022; gdj3.162.
[http://dx.doi.org/10.1002/gdj3.162]

[53] Karim S, Qadir A, Farooq U, Shakir M, Laghari AA. Hyperspectral imaging: A review and trends towards medical imaging. Curr Med Imaging Rev 2022; 19(5): 417-27.
[http://dx.doi.org/10.2174/1573405618666220519144358] [PMID: 35598236]

[54] Laghari AA, He H, Shafiq M, Khan A. Assessment of quality of experience (QoE) of image compression in social cloud computing. Multiagent and Grid Systems 2018; 14(2): 125-43.
[http://dx.doi.org/10.3233/MGS-180284]

[55] Bharti M, Saxena S, Kumar R. A middleware approach for reliable resource selection on Internet□of□Things. Int J Commun Syst 2020; 33(5): e4278.
[http://dx.doi.org/10.1002/dac.4278]