ADVANCED INTELLIGENT SYSTEMS Open Access
www.advintellsyst.com

Multi-Disease Detection in Retinal Imaging Using VNet with Image Processing Methods for Data Generation

Samad Azimi Abriz, Mansoor Fateh, Fatemeh Jafarinejad, and Vahid Abolghasemi*

Deep learning faces challenges like limited data, vanishing gradients, high parameter counts, and long training times. This article addresses two key issues: 1) data scarcity in ophthalmology and 2) vanishing gradients in deep networks. To overcome data limitations, an image processing-based data generation method is proposed, expanding the dataset size by 12x. This approach enhances model training and prevents overfitting. For vanishing gradients, a deep neural network is introduced with optimized weight updates in initial layers, enabling the use of more and deeper layers. The proposed methods are validated using the retinal fundus multi-disease image database dataset, a limited and imbalanced ophthalmology dataset available on the Grand Challenge website. Results show a 10% improvement in model accuracy compared to the original dataset and a 5% improvement over the benchmark reported on the website.

1. Introduction

Currently, deep learning with image processing has made significant advancements across various domains,[1] including image scene classification,[2] assistant human,[3] medical field,[4] safe driving,[5] and computer gaming.[6] This progress has led to a reduced reliance on traditional machine-learning methods. Deep neural networks, in particular, have played a crucial role in this advancement by extracting superior features and patterns[7,8] through deeper and more complex architectures.[9] In the field of medicine, early diagnosis of eye diseases[10] with the help of artificial intelligence (AI) significantly helps doctors.

Many individuals worldwide suffer from ocular diseases which, if diagnosed early, could prevent disease progression and result in lower costs for both themselves and healthcare organizations. Extensive efforts have been made in this regard, and sophisticated medical equipment has been developed to provide clinicians with greater insights into patients' ocular conditions, enabling more thorough examinations and precise diagnoses of various diseases. With the advancement of technology, medical imaging modalities with high resolution and quality, AI, deep learning, and machine vision have garnered increased attention and they can assist clinicians in early disease detection.

One of the challenges in the field of artificial intelligence in medicine is the scarcity of data, primarily due to privacy concerns that healthcare institutions generally do not permit public access to. Data constitutes the most crucial component of deep learning; the greater and higher the quality of the training data, the more effectively the deep learning model can extract features, learn, and accurately diagnose diseases. For this challenge, most previous studies have employed transfer learning and pre-trained models using ImageNet images or conventional data augmentation techniques.[11] However, these approaches have not effectively achieved high accuracy, indicating the need for either collecting more data or generating new data from existing ones. Transfer learning can improve model performance to some extent by utilizing knowledge gained from general image datasets; however, medical data often presents unique characteristics that differ significantly from common visual data. Consequently, transfer learning alone is insufficient for achieving optimal performance in complex medical tasks. To overcome this challenge, effective data synthesis techniques are essential to enrich the training data. By generating realistic variations of existing data, synthesis methods can expand dataset diversity, improve model generalization, and enhance the robustness of deep learning models in medical imaging. Thus, data synthesis strategies is crucial to mitigating the impact of data scarcity in medical AI systems.

Another significant challenge in deep learning is that when neural networks become deeper and have more layers to extract better features, gradient vanishing may occur. This phenomenon refers to the gradients becoming very small or even disappearing entirely, preventing the weights of the early layers from being updated, thereby hindering the learning process of the model. this issue arises when gradients become extremely small during backpropagation, preventing effective weight updates in earlier layers. As a result, the network struggles to learn meaningful features from the input data, ultimately limiting its performance.

S. A. Abriz, M. Fateh, F. Jafarinejad
Faculty of Computer Engineering
Shahrood University of Technology
Shahrood 3619995161, Iran
E-mail: mansoor_fateh@shahroodut.ac.ir

V. Abolghasemi
School of Computer Science and Electronic Engineering
University of Essex
Colchester CO4 3SQ, UK

The ORCID identification number(s) for the author(s) of this article can be found under https://doi.org/10.1002/aisy.202401039.

DOI: 10.1002/aisy.202401039

Various strategies have been proposed to address this challenge. For instance, ResNet employs skip connections that allow gradients to flow directly through the network, effectively mitigating the vanishing gradient problem. Similarly, DenseNet introduces dense connections, where each layer is connected to every other layer in a feed-forward manner, promoting feature reuse and further improving gradient flow. While these architectures have demonstrated success in combating gradient vanishing, our proposed model incorporates a modified weight-updating mechanism that efficiently combines elements of ResNet and Inception blocks. This hybrid approach facilitates stable gradient propagation while enhancing feature extraction capabilities, ultimately improving model performance in complex medical image analysis tasks.

As stated, deep learning faces challenges such as the shortage of training data, gradient vanishing, large number of parameters, long training time, and the need for appropriate hardware. In this article, the two most important challenges have been addressed, the first challenge is the shortage of training data, especially in the field of ophthalmology, and the second challenge is the gradient vanishing. The key contributions of this study are as follows:

1.1. New Image Sample Generation

We propose an image processing method that expands the dataset size up to 12 times, from 1920 to 23 040 images. This method employs various image processing techniques such as noise removal, thresholding, morphology, and weighted image fusion. As a result, new images are generated from existing data, incorporating changes introduced by morphology on thresholded images, and then fusion with the original images. The noise-removal step enhances image quality by reducing artifacts and improving feature visibility. Thresholding is then applied to segment important regions, followed by morphological operations that refine the segmented regions by eliminating noise and enhancing object boundaries. The resulting processed images are subsequently fused with the original images using weighted image fusion, which combines the complementary features of both. This comprehensive approach not only increases the dataset size but also introduces diverse variations in the data, improving the model's robustness and generalization ability during training. These synthesized images effectively simulate potential changes observed in real-world retinal imaging scenarios. The intuition behind the data generation process lies in the goal of enhancing the diversity and robustness of the dataset through a combination of thresholding and morphological operations. From a single original image, the process begins by applying thresholding under four different conditions. These four thresholded images represent diverse segmentations of the original data, each capturing different aspects of the structure and features relevant to the model. After thresholding, the process moves to the morphological operations stage, where more sophisticated transformations are applied. Morphology is used here to modify the geometric structure of the segmented regions, altering the shapes and boundaries of the objects. These operations, such as dilation, erosion, opening, and closing, introduce changes in the angles of lines, as well as the addition or removal of certain points, refining the segmented structures. This stage results in eight new images, where two thresholded images without noise undergoes the morphological transformations to produce enhanced visual patterns, varying the geometric properties of the segmented features. These modifications simulate realistic changes that might occur in real-world imaging, such as variations in the positioning or angles of anatomical features due to factors like rotation or scaling.Thus, from a single original image, the process generates 12 new images: four from the different thresholding conditions and eight from the morphological transformations applied to the thresholded images. This combination of thresholding and morphological techniques enriches the dataset by producing diverse variations, which in turn helps improve the model's ability to generalize, better capturing the wide range of potential scenarios encountered in actual applications.

1.2. Fast Update of Model Weights

We introduce a deep neural network where the initial layers can swiftly update their weights. The model integrates two architectures: ResNet[12] with 3×3 filters and Inception[13] with 5×5 filters. The weights of Inception blocks are updated from ResNet blocks in two stages, while the weights of ResNet blocks are updated solely from the dense layers in one stage. This weight-updating mechanism enhances the model's training process, thereby achieving more accurate disease detection. This combined strategy leverages the strengths of both architectures, enhancing the model's capacity to learn complex patterns and improving its robustness against overfitting. Consequently, this hybrid weight-updating approach contributes to superior performance in identifying subtle retinal abnormalities associated with multiple diseases.

1.3. Conducting Extensive Experiments

Classification of eye diseases using the retinal fundus multi-disease image database (RFMiD)[14] dataset is presented on the Grand Challenge website.[15] The dataset is provided under the Creative Commons Attribution 4.0 International (CC BY 4.0) license. It offers a cash prize and competition to incentivize further research in this domain, thus highlighting its importance. The highest score recorded on this site so far is 88.5%, while the score achieved with our proposed data generation model and method is 93.63%, indicating a 5% improvement. This suggests that both proposed contributions have significantly enhanced accuracy and performance.

The rest of the article is organized as follows, Section 2 reviews related works in this context. Section 3 describes the proposed method for data generation using image processing techniques. Section 4 presents the proposed deep learning model and addresses the challenges along with solutions. Section 5 elaborates on the experiments and results of the proposed methods on the dataset with tables and graphs. Finally, the article is concluded in Section 6.

2. Related Works

Existing research in this field can be divided into deep learning and ensemble learning. Methods based on ensemble learning do not normally face challenges with parameters and data. However, generic deep learning methods face such challenges.

2.1. Ensemble Learning Methods

In ref. [16], ensemble learning and transfer learning methods were used. Five different convolutional neural network (CNN) architectures were trained to predict the presence of any pathology and classify 28 different pathologies. These models were trained with a modified form of binary cross-entropy to minimize asymmetric loss. Different CNN architectures such as SE-ResNeXt, DenseNet-121, Inception V3, EfficientNet-B4, and EfficientNet-B5 were used for image classification. All networks were set up with default pre-training in PyTorch on the ImageNet dataset. In ref. [17], bagging is used to enhance the efficiency and precision of machine-learning models, and binary logistic regression algorithms were employed. The deep learning architecture used in the article includes a combination of pre-trained models, transfer learning, and ensemble learning techniques to build a robust model for the detection and classification of retinal diseases. The models used in the architecture include DenseNet201 and EfficientNetB4 for detectors, and ResNet152, InceptionV3, and DenseNet201 for classifiers, all of which were pre-trained on the ImageNet dataset. In ref. [18], ensemble learning techniques, deep learning models, and binary cross-entropy are employed. Preprocessing and image augmentation techniques, along with transfer learning using EfficientNetB4 and EfficientNetV2S, are utilized. The authors combined all subsets of the RFMiD data and increased the number of images from 3200 to over 10 000 using image augmentation techniques. They experimented with various deep learning models such as ResNet50, EfficientNetB0, DenseNet, and InceptionResNetV2, ultimately selecting EfficientNetB4 as the best-performing model for further experimentation. In ref. [19], ensemble learning to combine the predictive capabilities of various deep convolutional neural network models is used. The authors trained several models based on the architectures of DenseNet-201 and EfficientNet-B4 for disease risk prediction, and architectures of ResNet152, InceptionV3, DenseNet201, and EfficientNetB4 for disease label classification. Ensemble learning strategies such as bagging and stacking were employed, and multiple individual trainings were conducted. Disease labels and risk prediction were trained using transfer learning on ImageNet.[20] aims to present a framework for the classification of multiple diseases. The framework consists of three stages: preprocessing, disease risk detection, and multi-disease classification. The preprocessing stage includes methods such as data augmentation, oversampling, resizing, and normalization. Disease risk detection can be accomplished using two convolutional neural network (CNN) architectures: DenseNet201 and EfficientNetB4. The multi-disease classification stage uses a hybrid approach incorporating three convolutional neural network architectures: DenseNet201, EfficientNetB4, and ResNet105.

2.2. Generic Deep Learning Methods

In ref. [21], a multi-label CNN (ML-CNN) model is proposed for diagnosing various ocular diseases (ODs) from color fundus images. The suggested system comprises three main stages: preprocessing, feature extraction, and multi-label classification (ML-C), followed by prediction. In the preprocessing stage, image resizing and data augmentation are applied to standardize the images and balance the dataset. The ML-CNN model architecture consists of three layers used to extract features and classify ocular diseases. In ref. [22], "Saliency-Guided Anomaly Awareness" (SatFormer) is a model with four stages. Each stage includes a saliency enhancement module (SEM) and sequential SatFormer blocks. The authors propose a SEM to extract more prominent indications of lesions and increase the activation of features related to small and dispersed areas of anomalies at each stage. The suggested model contains innovative components like the SEM and an anomaly-awareness attention mechanism, providing a more comprehensive understanding of diverse lesions and their interdependencies, thereby achieving improved performance in retinal disease classification. The greater part of the suggested methods in the articles are techniques that have utilized ensembling, transfer learning, stacking, and bagging. The main reason behind this is the shortage of training data, which significantly affects the model's accuracy and performance, preventing it from being well-trained and thereby achieving poor performance on test data. These techniques used in the articles somewhat improve the accuracy of disease diagnosis. However, to enhance the accuracy further, effective methods for data synthesis are required.

Ensembling, stacking, and bagging involve combining multiple models to enhance overall accuracy. While these techniques can improve the accuracy of disease diagnosis to some extent, they often do not fully compensate for the fundamental problem of data scarcity. Generic deep learning methods often rely on novel architectures or specialized techniques to achieve desired outcomes, but they may struggle with the complexities of deep learning, such as managing a large number of parameters, ensuring stable training, and acquiring sufficient data to avoid overfitting. To achieve a more significant enhancement in accuracy, effective data synthesis methods are needed. These methods can increase the variety and volume of training data, allowing models to learn more diverse features and improve their generalization capabilities. Thus, future works should focus on developing and refining data synthesis methods to complement existing methods, ultimately improving the reliability and performance of deep learning models in medical applications.

3. Proposed Method

3.1. Novel Data Generation

In deep learning, the scarcity of training data presents a significant challenge, exerting a substantial impact on the model's learning process. Adequate examples are also essential for effectively tuning the parameters of any deep learning model. A model trained on limited data may experience overfitting. Bad training can result in diminished accuracy and performance as the model cannot extract better features and patterns.

To address this issue for training of models, various methods such as data augmentation, transfer learning, and the use of simpler models are employed. Data augmentation is a widely used approach to mitigate this issue. It is used to expand the dataset by applying minor modifications such as rotation, shifting, scaling, cropping, reflection, and adjustments to color and brightness, to the training data (some examples provided in **Figure 1**). Although these modifications introduce some diversity to the original images, they may not sufficiently address the issue of limited training data.

In this study, we present a novel data synthesis method using image processing techniques that increases the number of training data up to 12 times. The image processing techniques used for data synthesis include thresholding, noise removal, morphology, and weighted image fusion. These are divided into four main stages as shown in **Figure 2**.

As shown in Figure 2, the input image is first converted to grayscale before applying thresholding in the initial stage. Thresholding is purposed to generate a binary image. We employ adaptive thresholding[23] where the threshold value is selected according to the local features of image blocks and hence provide a more accurate binary image.[24] It can be performed through mean[25] or Gaussian[26] features. This process is often performed both with and without noise removal, resulting in the generation of four images from a single image. In our proposed method, the two images that have undergone thresholding but have not had noise removal applied are further processed using weighted image fusion. This strategy is employed because, in the morphological stage, noise may introduce artifacts that could be mistaken for disease symptoms. The two remaining images, which have undergone both thresholding and noise removal, are then processed using morphological operations (to be explained next). **Figure 3** provides a graphical illustration of this process.

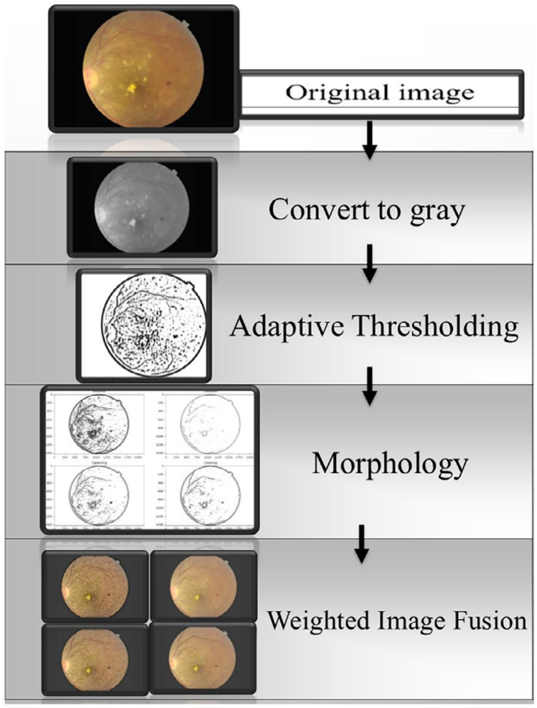

Figure 2. Four main stages of the proposed data generation method.

Morphology[27] is another crucial technique in image processing used to manipulate the shape and geometry of photos. These

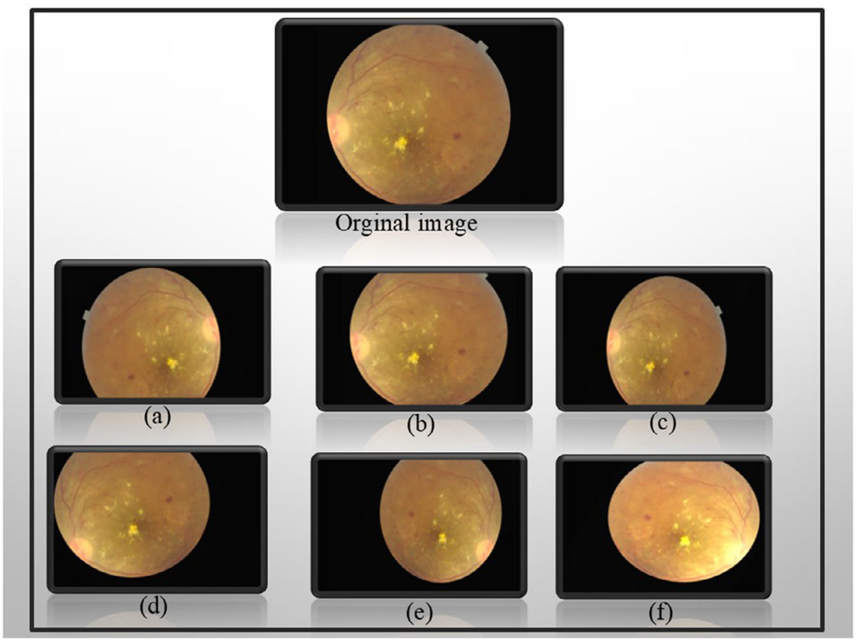

Figure 1. Six example images a–f) as a result of performing a random combination of operations including rotation, cropping, zooming, horizontal rotation, and brightness to the original image.

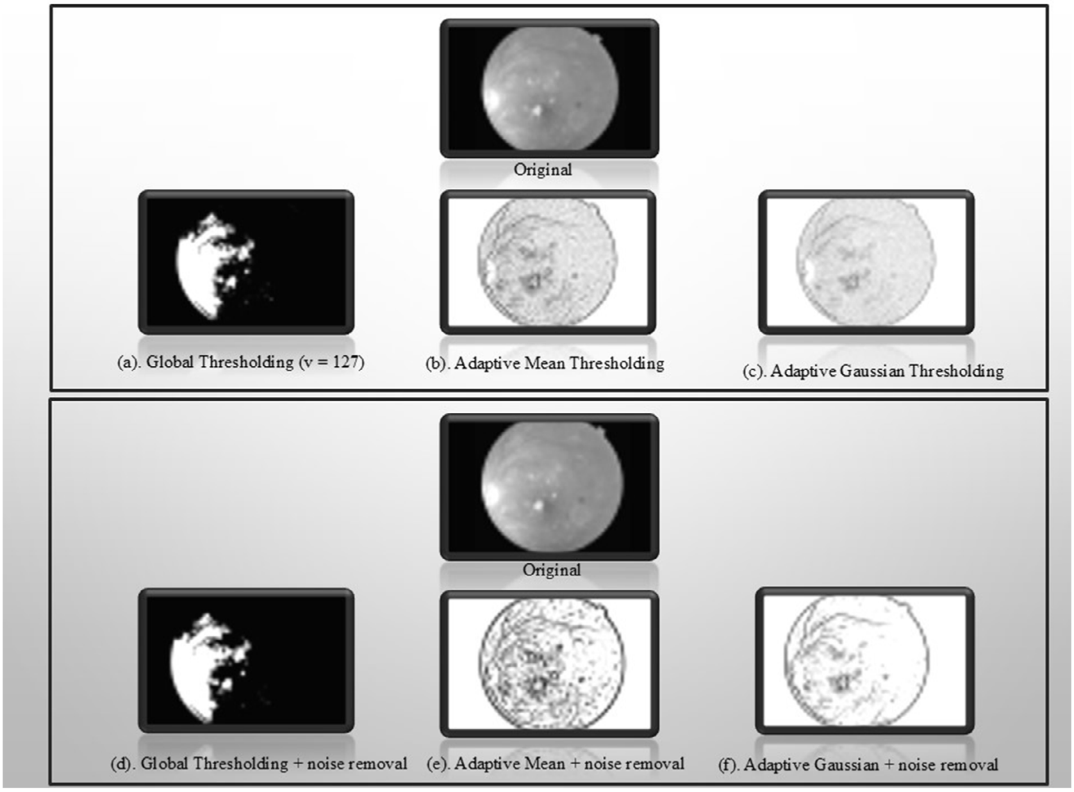

Figure 3. Stage 2 of the proposed data generation process. a–c): Results without noise removal—(a) Global thresholding, (b) Adaptive mean thresholding, (c) Adaptive Gaussian thresholding. d–f): Corresponding results with noise removal—(d) Global, (e) Adaptive mean, (f) Adaptive Gaussian thresholding. The global thresholding, i.e., (a) and (d), are ultimately skipped and only four images are generated.

manipulations involve operations such as resizing, removing specific points, and forming or enhancing edges. There are four fundamental morphological operators: dilation, erosion, closing, and opening. Dilation is utilized for geometric expansion, hole filling, and connecting lines; erosion is employed for reducing geometric size, eliminating noise, and enhancing edges; closing, which involves dilation followed by erosion; and opening, which involves erosion followed by dilation. For a visual representation of the effects of these operators, the results of applying them to two images from the previous step are shown in **Figure 4** and **5**, resulting in a total of 8 new images. Combined with the previous four images, a total of 12 new images are obtained per original image in the dataset. Consequently, using the proposed method, the amount of training data samples increases from 1920 to 23 040. The morphology operation generates a new image by adding and removing points, lines, and edges, and altering the geometric shape within the image. These four images are then forwarded to the next stage, which is weighted image fusion.

Considering that the original images are in color and consist of three channels, utilizing these color channels can lead to better feature extraction and pattern recognition. Therefore, the images obtained from thresholding and morphology are fused with the original images in a weighted manner,[28] as shown in **Figure 6**. This enables the model to extract more effective features and be well-trained. The key advantage of weighted image fusion is that it allows for the preservation of critical features from the original images while incorporating additional insights from the thresholded and morphologically transformed images. By fusing these various images, the resulting dataset provides a more comprehensive and nuanced representation of the underlying patterns. The equation for our proposed weighted image fusion is expressed as follows

$$I_{\text{fused}}(x,y) = \frac{\sum_{i=1}^{n} w_i \cdot I_i(x,y)}{\sum_{i=1}^{n} w_i} \tag{1}$$

$I_{\text{fused}}(x,y)$: The intensity of the final pixel at position (x,y) in the fused image.

$I_i(x,y)$: The intensity of the pixel at position (x,y) in the input image i.

w_i: The weight associated with the input image i.

n: The number of input images.

where $F(x,y)$ is the pixel value at position (x,y) in the final fused image, N represents the number of input images, $w_k(x,y)$ shows the weight assigned to the pixel at position (x,y) in the k-th input image, and $I_k(x,y)$ stands for the pixel value at position (x,y) in the k-th input image. Four thresholding stage images and eight morphological stage images are fused separately with the weighted original image, resulting in a total of 12 generated images. In the final stage, which involves the weighted image fusion, the final image quality may decrease, and certain features might not be extracted and could be

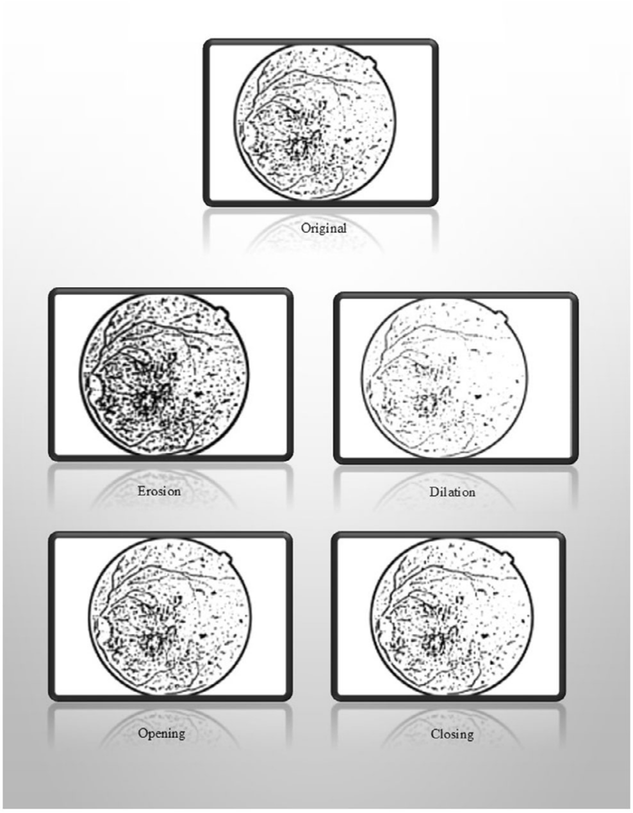

Figure 4. Morphological operations on the image that have been applied Adaptive mean thresholding and noise removal.

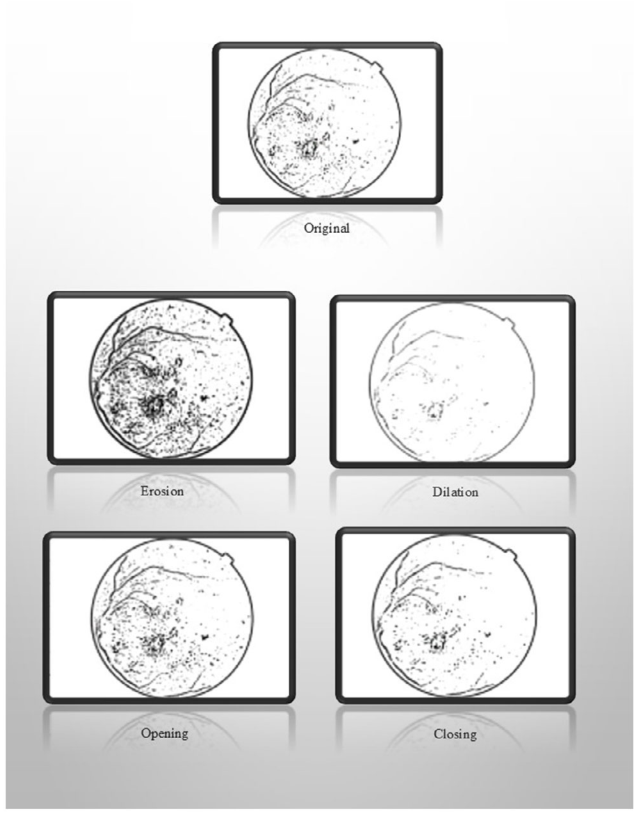

Figure 5. Morphological operations on the image that have been applied adaptive gaussian thresholding and noise removal.

overlooked during model training. Therefore, enhancing the quality of the images in the final stage is necessary.

Image quality plays a pivotal role in feature and pattern extraction,[29] as higher-quality images generally lead to better feature extraction. One method for enhancing image quality is adaptive histogram equalization (AHE),[30] which locally smooths different regions of the image. In this article, this technique is utilized to enhance image quality and extract new features, as shown in **Figure 7**.

By employing AHE, new sets of images are created, helping to mitigate overfitting by extracting new features. In this article, two distinct categories of data are generated: 1) The first category comprises images derived from the earlier stage where thresholded, morphologically processed, and original images are combined through a weighted fusion technique. This fusion process creates composite images with enhanced features and additional variations, providing a broader range of training examples. 2) The second category consists of images from the last step (weighted image fusion) for which AHE has been applied. By increasing contrast and emphasizing local details, these images offer new insights and help extract features that might not be visible in the original data. This category brings in a different form of data augmentation, enriching the dataset with higher contrast and potentially revealing subtle patterns.

It is worth mentioning that, for training deep learning models, one can use either category individually or a combination of both. Using the first category (weighted image fusion) allows for a diverse range of images with various features, while the second category (weighted image fusion with AHE) provides enhanced contrast and distinct details. Combining both categories can further increase the variety in the training dataset, thereby achieving more robust models that generalize better.

The choice of which category to use for training depends on the specific application and the desired outcomes. If overfitting is a concern, utilizing both categories can help create a more varied dataset, reducing the chances of the model memorizing the training data. On the contrary, using a single category might be beneficial if the goal is to focus on a specific type of feature extraction or image enhancement technique. Overall, employing AHE and weighted image fusion offers a flexible approach to augmenting training data, providing deep learning models with a richer and more diverse set of inputs. This approach can lead to improved performance and accuracy, especially in applications where detailed image analysis is required.

The AHE can be mathematically expressed as follows:

1) Histogram for each image block

$$H_b(i) = \sum_{(x,y)\in \text{block}} \delta(I(x,y) - i) \tag{2}$$

where:

$H_b(i)$: Histogram value for intensity level i in the block.

δ: Dirac delta function.

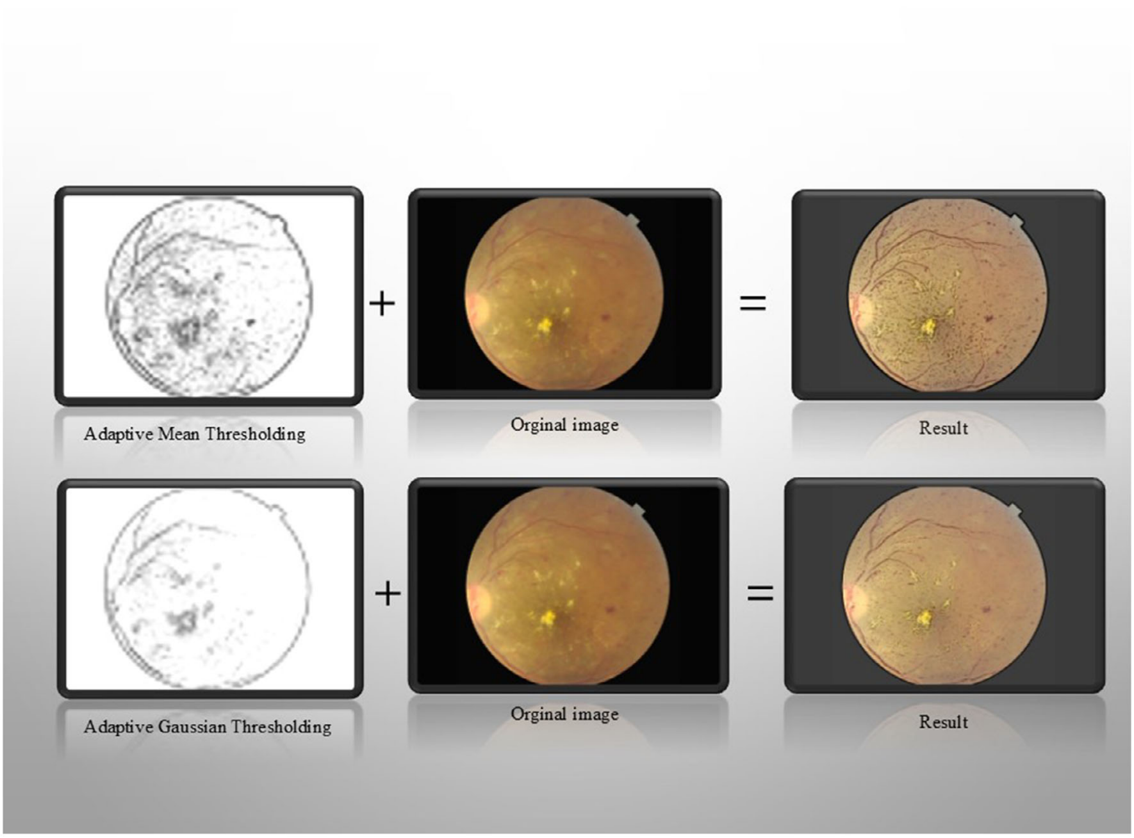

Figure 6. Stage 4: The weighted fusion of the adaptive mean thresholding and adaptive gaussian thresholding without noise removal with the original image.

$I(x, y)$: Intensity of the pixel at position (x, y).

2) Contrast enhancement

$$H_b^{\text{clipped}}(i) = \min(H_b(i), \text{ClipLimit}) \tag{3}$$

where:

$H_b^{\text{clipped}}(i)$: Clipped histogram value for intensity level i.

ClipLimit: Threshold value to limit the histogram.

3) Cumulative distribution function (CDF)

$$CDF_b(i) = \sum_{j=0}^{i} H_b^{\text{clipped}}(j) \tag{4}$$

where:

$CDF_b(i)$: Cumulative distribution function value for intensity level i in the block.

4) Intensity mapping

$$I'(x, y) = I_{\min} + \frac{CDF_b(I(x, y)) - CDF_b(I_{\min})}{\text{number of pixels in block}} \times (I_{\max} - I_{\min}) \tag{5}$$

where:

$I'(x, y)$: Enhanced intensity of the pixel at position (x, y).

$I_{\min}$: Minimum intensity value in the image.

$I_{\max}$: Maximum intensity value in the image.

number of pixels in block: Total number of pixels in the current block (block size $= 8 \times 8$).

The pseudocode for all stages of the proposed data generation method is provided in **Algorithm 1**.

3.2. Deep Learning Model

Although data synthesis is an essential step for the success of deep learning models, most deep neural networks suffer from a phenomenon known as "Vanishing Gradient". It especially occurs as the depth of the network increases, thus, presenting a challenge to the training process. In deep neural networks, the backpropagation algorithm[31] is commonly used to update weights during training. This algorithm calculates gradients by propagating errors from the output to the input layer. However, as the gradients are propagated backwards through multiple layers, they often diminish significantly, approaching zero. This diminishing gradient problem can hinder weight updates in earlier layers of the network, resulting in suboptimal training or even causing the training process to stall.

To address this issue, a deep neural network called "VNet", derived from the term "Vanishing" is proposed here. As shown in **Figure 8**, one side of this network (left) consists of blocks of convolutional layers, while on the opposite side (right), there are

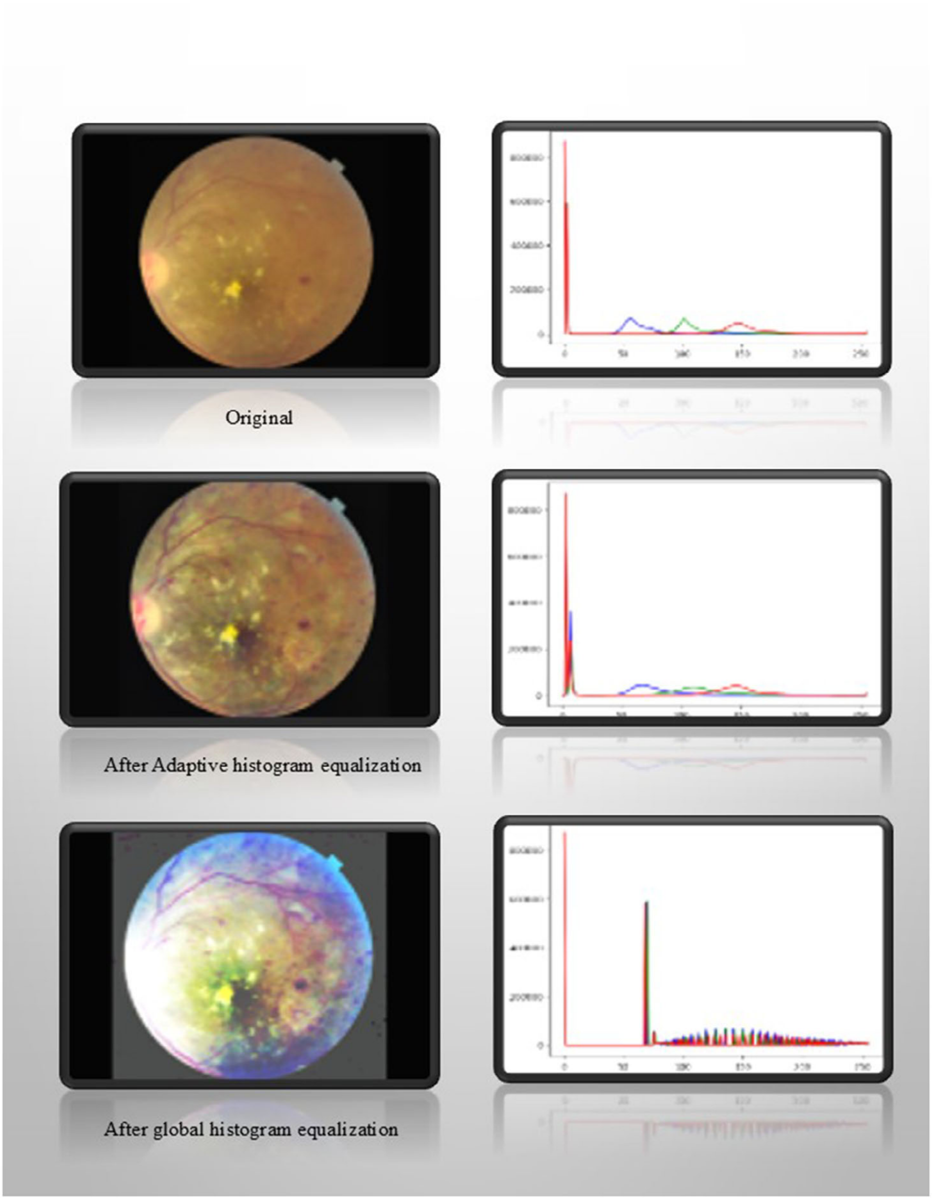

Figure 7. Implementation of global histogram and adaptive histogram on the original image.

dense layers that are connected in a star-shaped manner to all the convolutional blocks. In what follows, the proposed model is described in detail.

The side comprising convolutional layers utilizes two networks, namely "ResNet" and "Inception". The Inception network comprises three blocks, each containing two convolutional layers with filter sizes of 5×5 for feature extraction and 1×1 for updating the previous layer and block. These Inception blocks are designed to extract diverse features from the input data, with the larger 5×5 filters capturing broader patterns and the smaller 1×1 filters providing efficient updating within each block. The 5×5 filters are particularly useful for detecting high-level spatial features, while the 1×1 filters improve computational efficiency, serve as dimensionality reducers, and effectively combine information across channels. This combination of filter sizes allows for more flexible feature extraction and helps prevent information loss as the data moves through the network. The structure and input/output Equation (7)–(10) of the block layers are shown in **Figure 9**.

Layer (5×5) of Inception with input X, Output a_1

$$z_1 = Wx + b \tag{6}$$

$$a_1 = g(z_1) \tag{7}$$

Layer (1×2) of Inception with input X, Output a_2

$$z_2 = Wx + b \tag{8}$$

Algorithm 1. Enhanced data generation for training deep learning models.

Step 1: Convert to Grayscale
Input: Original color image
Output: Grayscale image
gray_image = convert_to_grayscale(*original_image*)
Step 2: Adaptive thresholding
Input: Grayscale image
Operations:
    Adaptive mean thresholding with and without noise removal
    Adaptive gaussian thresholding with and without noise removal
Output: Four thresholded images
thresholded_mean_noise = adaptive_mean_thresholding (*gray_image*, *noise_removal* = *True*)
thresholded_mean_no_noise = adaptive_mean_thresholding (*gray_image*, *noise_removal* = *False*)
thresholded_gaussian_noise = adaptive_gaussian_thresholding (*gray_image*, *noise_removal* = *True*)
thresholded_gaussian_no_noise = adaptive_gaussian_thresholding (*gray_image*, *noise_removal* = *False*)
Step 3: Morphological Operations
Input: Thresholded images with noise removal
Operations: Dilation, Erosion, Closing, Opening
Output: Eight new images per thresholded image
dilated_image = dilation(*thresholded_image*)
eroded_image = erosion(*thresholded_image*)
closed_image = closing(*thresholded_image*)
opened_image = opening(*thresholded_image*)
Element size: A 5 × 5 square with all values equal to one.
Step 4: Weighted Image Fusion and Quality Enhancement
Input: Thresholded and Morphologically Processed Images
Operations:
    Weighted fusion of images with the original image
    Image quality enhancement with Adaptive Histogram Equalization
Output: Final enhanced images
fused_image = weighted_image_fusion ([*original_image*, *thresholded_image*, *morphed_image*], *weights*)
enhanced_image = adaptive_histogram_equalization(*fused_image*)
End of Algorithm

$$a_2 = g(z_2) \tag{9}$$

Output of Inception with inputs (a_1,a_2, Output O(x)

$$O(x) = (a_1) + (a_2) \tag{10}$$

where g is activation function Relu, W is weight, and b is bias.

The first block in Figure 9 consists of 32 filters, the second block has 64, and the third block has 128 filters, all of which are connected to the blocks of the ResNet network. In addition, to ensure that weights are updated from the shortest path to prevent the occurrence of vanishing gradients, both networks collaborate in feature extraction.

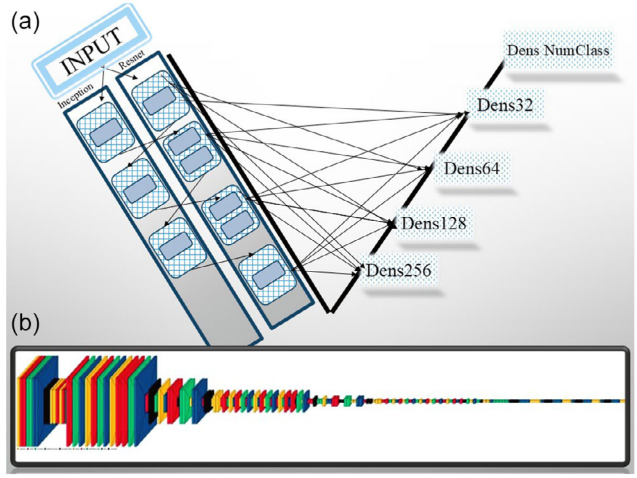

Figure 8. a) The general structure of the proposed model and b) visualization of the deep learning architecture.

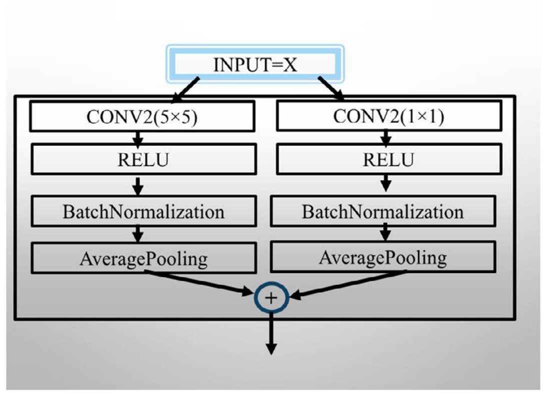

Figure 9. Blocks of inception network in the proposed model.

The ResNet network with 3 × 3 filters consists of four blocks, each of which can have two sub-blocks. The 3 × 3 filters are particularly effective for detecting fine-grained textures, edges, and detailed features within the images. Additionally, they contribute to increasing the receptive field while maintaining efficient parameter usage. The first block has 32 filters with one sub-block, the second block has 64 filters with two sub-blocks, the third block has 128 filters with two sub-blocks, and the fourth block has 256 filters with one sub-block. The structure and input and output Equation (12)–(16) of the block layers are shown in **Figure 10**.

Layer 1 of ResNet with input X, Output a_1

$$z_1 = Wx + b \tag{11}$$

$$a_1 = g(z_1) \tag{12}$$

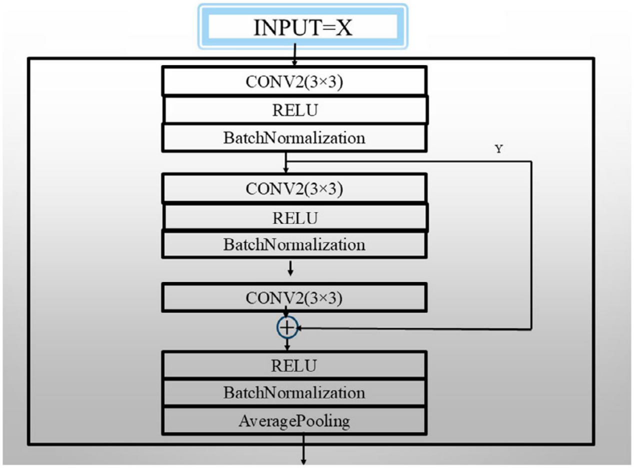

Figure 10. Sub-blocks of ResNet network in the proposed model that each block can have one or two sub-blocks.

Layer 2 of ResNet with input layer $1(a_1)$, Output a_2

$$z_2 = Wa_1 + b \tag{13}$$

$$a_2 = g(z_2) \tag{14}$$

Layer 3 of ResNet with input (layer $2(a_2)$,Y), Output a_3

$$z_3 = Wa_2 + b \tag{15}$$

$$a_3 = g(z_3 + Y) \tag{16}$$

Each sub-block of the ResNet network contains three convolutional layers, and each block has two outputs: one output is directed to the next Inception block to utilize features extracted by the 5×5 filters and also to facilitate rapid weight updates.

The second output is linked to a global average pooling layer, which in turn connects to all dense layers on the opposite side to ensure optimal, faster, and more efficient weight updates. In this approach, even the weights of the first convolutional layer can be updated from the last dense layer or other layers, thereby addressing the challenge of the vanishing gradient problem. As shown in **Figure 11** and **12**, the shapes of filters and feature maps in the third block of ResNet and the third block of inception vary, thereby achieving the extraction of distinct features.

In a model that uses two different networks with 3×3 and 5×5 filters, the filter values and their weights change when applied to the layers, which causes different feature maps to be created, resulting in diversity. In **Figure 13**, one can see the Grad-CAM[32] images for several sample images from the training dataset, where part (c), that is, the heat map, shows which regions of the image contributed most to disease prediction. In addition to the blood vessels, other regions such as damaged spots, areas with color or texture variations, and regions with abnormal features indicative of ocular diseases are also highlighted in the Grad-CAM images. This can enhance the understanding of the model's decision-making process and help identify critical features for disease diagnosis.

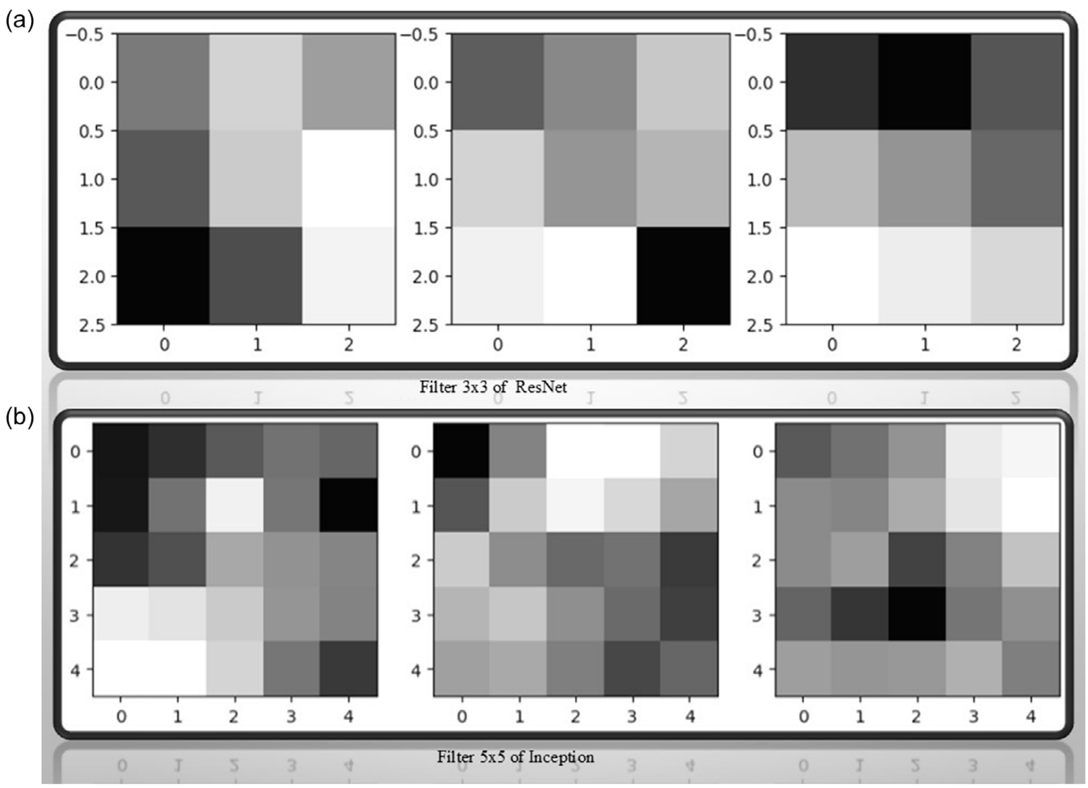

Figure 11. The shapes of the filters in block three of the proposed model: a) filter 3×3 of ResNet, b) filter 5×5 of inception.

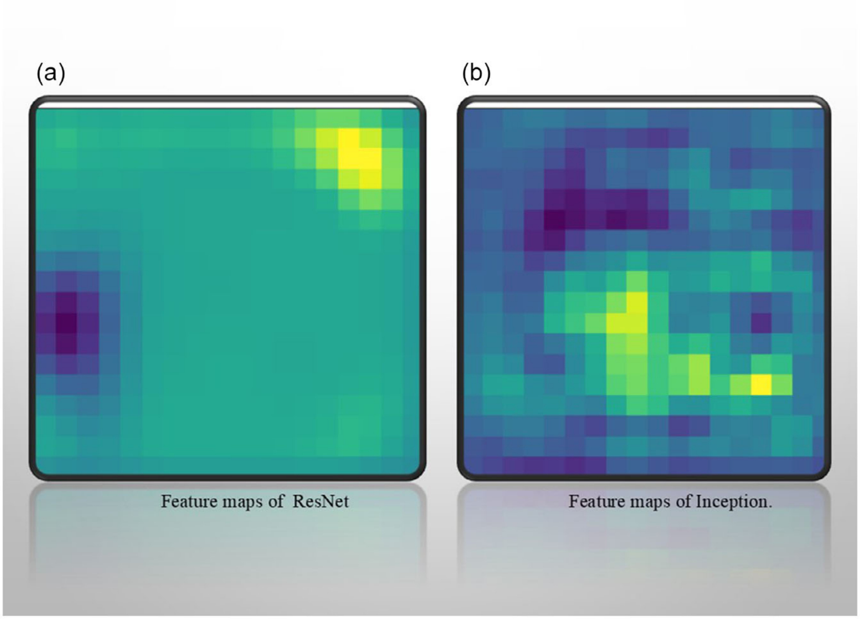

Figure 12. The shapes of the feature maps in block 3 of the proposed model: a) feature maps of ResNet, b) feature maps of inception.

As a result of feature maps and filter shapes in the network, the model is better trained and makes more accurate decisions. Employing diverse networks and combining them with different filters allows for various features to be extracted, facilitating improved training of the model and enhancing its accuracy and efficiency. Simultaneous utilization of both inception and ResNet networks with varying 3×3 and 5×5 filters enables the extraction of diverse and varied features, thus helping to prevent overfitting and effective network training. The network consists of 28 convolutional layers, seven blocks, and five dense layers, totalling approximately three million parameters. The learning rate was initially set to 0.001 and adjusted dynamically during training. If the loss reduction over two consecutive epochs was less than 0.004, the learning rate decreased by 30%; additionally, it spiked to ten times its current value at predefined epochs. In this network, ReLU,[33] Batch Normalization,[34] and Dropout[35] are used in all blocks. In each convolutional block, a dropout rate of 0.25 was applied, while a dropout rate of 0.5 was used after each dense layer. The model was implemented, trained, and tested on a Dell laptop equipped with 16 GB of RAM, an NVIDIA RTX 3060 GPU with 6 GB of memory, and an 11th-generation 7-core CPU.

In machine learning, particularly in deep learning, handling imbalanced data poses a significant challenge.[36] Imbalanced data occurs when there is a substantial disparity in the number of samples across different classes. This imbalance can lead to several issues, such as improper parameter adjustment, reduced accuracy, bias toward the majority class, and potential overfitting. When training data includes unequal samples for each class, the model may overfit to the majority class and underperform on the minority class, thereby compromising overall performance. Various methods have been suggested to address this issue, including adding samples to underrepresented classes or removing samples from overrepresented classes. However, these approaches may not be very effective, especially when the overall training data is limited. In this article, weighted binary cross entropy[37,38] is considered as a solution to this problem

$$L_w = -E[w_1 \cdot y_t \cdot \log(y_p) + w_0 \cdot (1 - y_t) \cdot \log(1 - y_p)] \tag{17}$$

where E represents the expectation or the average, w_1 is the weight for the positive class (class 1), w_0 is the weight for the negative class (class 0), y_t is the true label, which can be either 0 or 1, and y_p is the predicted probability for the positive class (class 1).

Equation (17) resembles the standard binary loss function, but with the distinction that it assigns varying weights to classes depending on the number of training samples they contain. This adjustment aims to mitigate the possibility of overfitting the training data. By applying this method, the model can better handle situations characterized by small and imbalanced training datasets, thereby enhancing both accuracy and efficiency. The pseudocode for deep learning model (VNet) is provided as a reference in **Algorithm 2**.

4. Experimental Results

4.1. Database

The RFMiD is a new public dataset designed to facilitate research in eye disease diagnosis and identification using retinal fundus images. The RFMiD dataset is available on the Grand Challenge platform, which serves as a dedicated space for medical imaging challenges and dataset sharing. This platform was utilized solely for accessing the dataset; no additional tools or benchmarking features from the platform were employed in this study. The RFMiD is unique in its composition, containing 3200 retinal fundus images taken with three different fundus cameras and

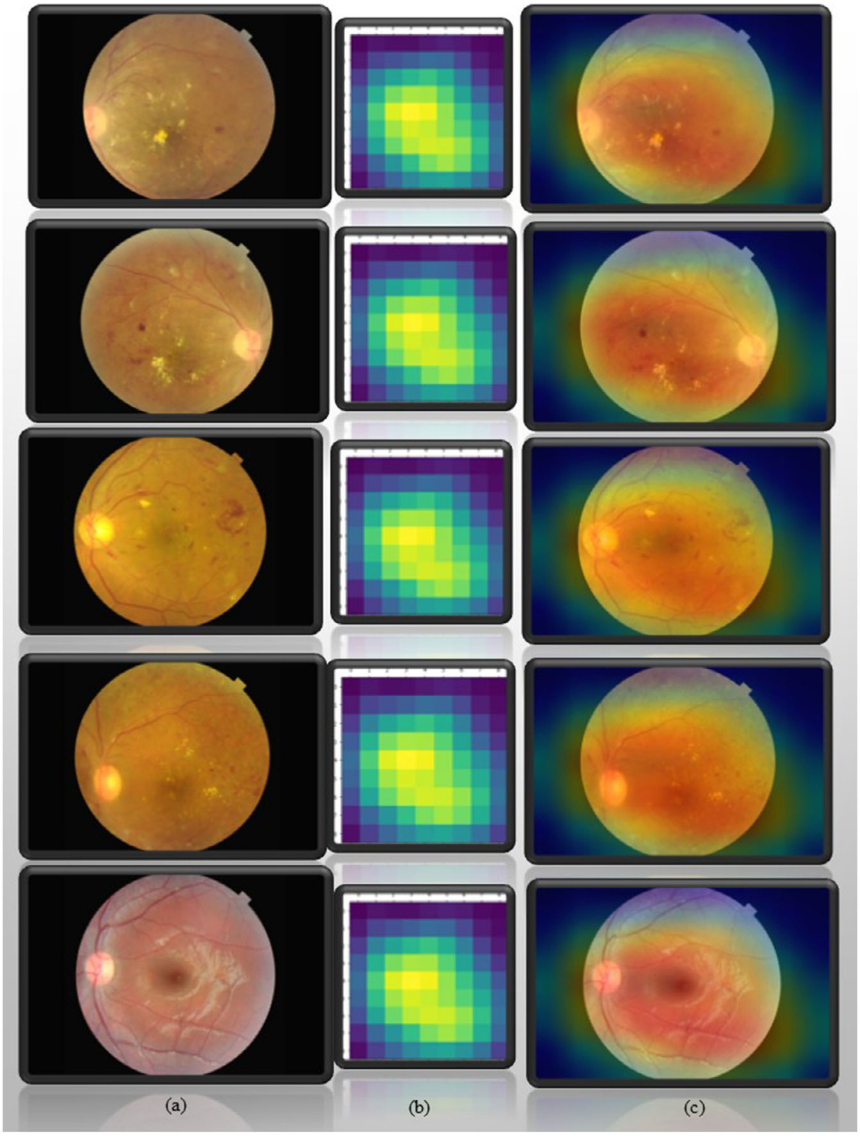

Figure 13. Grad-CAM images for several samples from the training dataset. a) original image, b) feature map, and c) heat map.

annotated for 46 different conditions, including both common and rare diseases. The dataset is divided into three sections: training, validation, and testing, which make up 60%, 20%, and 20% of the images, respectively. This dataset comprises a diverse array of retinal fundus images annotated for 46 different conditions, offering a wide range of diseases commonly observed in clinical settings. The images in this dataset were reviewed and validated by a project team lead. Each condition is described in detail, including its visual features and clinical significance. The model used for testing has been trained and tested exclusively on this dataset.

4.2. Evaluation Metrics

AUC is a common metric for evaluating multi-label classification models, where a higher AUC indicates better performance. The ROC curve shows the relationship between the false positive rate (FPR) and true positive rate (TPR); a curve closer to the upper-left corner signifies improved model performance. The PR (precision-recall) metric is useful for imbalanced datasets, measuring the trade-off between precision and recall. Higher PR curves indicate better performance. Accuracy is a straightforward measure that represents the proportion of correct predictions. High accuracy may indicate a good model, but in unbalanced data conditions, it may be misleading because the model may have high accuracy by focusing on the dominant class without performing well in predicting the minority class. This metric is not appropriate for this dataset. The accuracy metric in imbalanced data does not correctly reflect the model's performance because it simply measures the number of correct predictions relative to the total predictions. When the data is imbalanced, meaning the number of samples in one or more

Algorithm 2. Deep learning model (VNet).

Input: Deep neural network structure

Output: Deep learning model

1) **Addressing Vanishing Gradient Problem**
   - Use Backpropagation to update weights during training.
   - Identify gradient diminishment through multiple layers, which can stall training.
   - Introduce VNet architecture to mitigate vanishing gradient issues.
2) **Design VNet Structure**
   - Left side: Convolutional layers for feature extraction.
     – Use “Inception” and “ResNet” networks.
     – Inception: Apply 5 × 5 filters for broader patterns, 1 × 1 for updates.
     – ResNet: Apply 3 × 3 filters across 4 blocks to enhance feature diversity.
   - Right side: Dense layers connected to convolutional blocks to ensure efficient weight updates.
3) **Inception Block Configuration**
   - Block 1: 32 filters; Block 2: 64 filters; Block 3: 128 filters.
   - Capture varied features, preserving information through 5 × 5 and 1 × 1 filters.
4) **ResNet Block Configuration**
   - Block 1: 32 filters with one sub-block; Block 2: 64 filters with two sub-blocks; Block 3: 128 filters with two sub-blocks; Block 4: 256 filters with one sub-block.
   - Propagate features to next inception block and global average pooling layer to optimize updates.
   - Global average pooling layer connects to all dense layers
5) **Imbalanced Data Handling**
   - Apply weighted binary cross entropy loss to balance class representation: $L_w = -E[w_1 \cdot y_t \cdot \log(y_p) + w_0 \cdot (1 - y_t) \cdot \log(1 - y_p)]$
   - Adjust weights based on class frequency to enhance model performance on small and imbalanced datasets.

Output: deep learning model

classes is much higher than in other classes, accuracy can be misleading.

4.3. Results

In RFMiD dataset, the quantity of training samples for each class is severely low and imbalanced. The proposed method in Section 3.1 is used to mitigate this issue. Furthermore, the proposed VNet model in Section 3.2 is applied to address the vanishing gradient problem. The training process was conducted in two rounds: 1) The model was initially trained using the data from the first category, where local histogram equalizer was not applied. This initial training round lasted for 150 epochs, allowing the model to learn from a diverse range of generated samples. This step focuses on establishing a baseline and allowing the model to capture essential features from the augmented data. 2) After the first round, the model's weights were retained, and the same model was trained again with data from the second category, where the AHE was applied. This round also lasted for 150 epochs. The use of an AHE in the second category enhances image contrast and helps extract new features, providing additional learning opportunities for the model.

This two-round training approach serves several purposes:

Data Augmentation: By training initially with data from the first category, the model gains exposure to a broader set of training samples, reducing the risk of overfitting and allowing it to generalize better.

Gradient Stability: The proposed model's architecture is designed to address the vanishing gradient problem, ensuring that gradient flow is maintained throughout the training process. This stability is crucial for deep learning models, especially when dealing with imbalanced and low-sample datasets.

Feature Refinement: The second round of training with AHE provides an opportunity for further refinement. The improved contrast helps the model learn from additional features that might not have been evident in the first round, contributing to better accuracy and robustness. By following this training strategy, the proposed method and model work together to overcome the challenges of data scarcity, imbalance, and vanishing gradients, resulting in a more reliable and accurate deep learning model.

The model was trained for three different types of classification, each with varying complexity and number of classes. Here's a detailed explanation of each type, along with the corresponding AUC metrics for the training, validation, and test datasets:

Binary classification: Disease or Not Disease, meaning two classes. This represents the initial stage of the proposed system, as seen in **Figure 14**, in the article, where the AUC metric was 99.98 on the training and validation dataset and 96.34 on the test dataset.

Training on 27 classes plus an additional “other” class, as shown in **Table 1**, totalling 28 classes. This corresponds to the second stage of the proposed system, where the AUC metric was 99.96 on the training and validation dataset and 95.44 on the test dataset.

Training on 45 classes. In this stage, the proposed system achieved an AUC metric of 99.87 on the training and validation dataset and 94.53 on the test dataset.

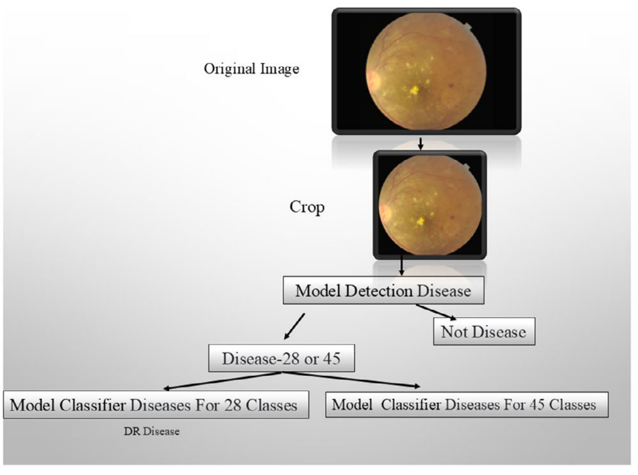

Figure 14. The framework of the proposed system.

Table 1. The second type consisted of 27 classes plus an additional "other" class. For the full list of abbreviations, please refer to Appendix.

k	Disease	*k*	Disease
1	DR	15	ARMD
2	MH	16	DN
3	MYA	17	BRVO
4	TSLN	18	ERM
5	LS	19	MS
6	CSR	20	ODC
7	TV	21	CRVO
8	AH	22	ODP
9	ODE	23	ST
10	AION	24	PT
11	RT	25	RS
12	CRS	26	EDN
13	RPEC	27	MHL
14	RP	28	OTHER

To validate the effectiveness of the proposed data generation method, we conducted an experiment where we trained the proposed model twice. The first training session involved using the original dataset for both training and testing. In the second session, we utilized the data generated by the proposed data generation method for both training and testing. The goal was to compare the model's performance between these two conditions. The training procedure for each scenario was as follows:

First training session (original data): The model was trained for 30 epochs using the original dataset for both training and testing. This provided a baseline for evaluating the model's performance without additional data augmentation.

Second training session (generated data): The model was trained again for 30 epochs, but this time using the data generated by the proposed data generation method for both training and testing. This allowed us to assess the impact of the data synthesis techniques on the model's performance.

As shown in **Figure 15** and **16**, the data generated by the proposed method demonstrates significantly better performance in terms of error rate, accuracy, and AUC score on the test set compared to the original data. Specifically, in terms of AUC score, the proposed data generation method shows a 13% improvement over the original data when tested on the model after the initial 30 epochs. Overall, these results highlight the effectiveness of the proposed data generation method in improving the model's performance across various metrics. The increased AUC score, lower error rate, and higher accuracy demonstrate the benefits of using data synthesis techniques to overcome data scarcity and enhance deep learning model training.

The training of the proposed model in all three classification types did not result in overfitting due to the larger amount of data. Additionally, the network architecture prevented vanishing gradients, thereby achieving improved model performance. The AUC curves for the 28 classes are depicted in **Figure 17**, offer a comprehensive view of the model's effectiveness in

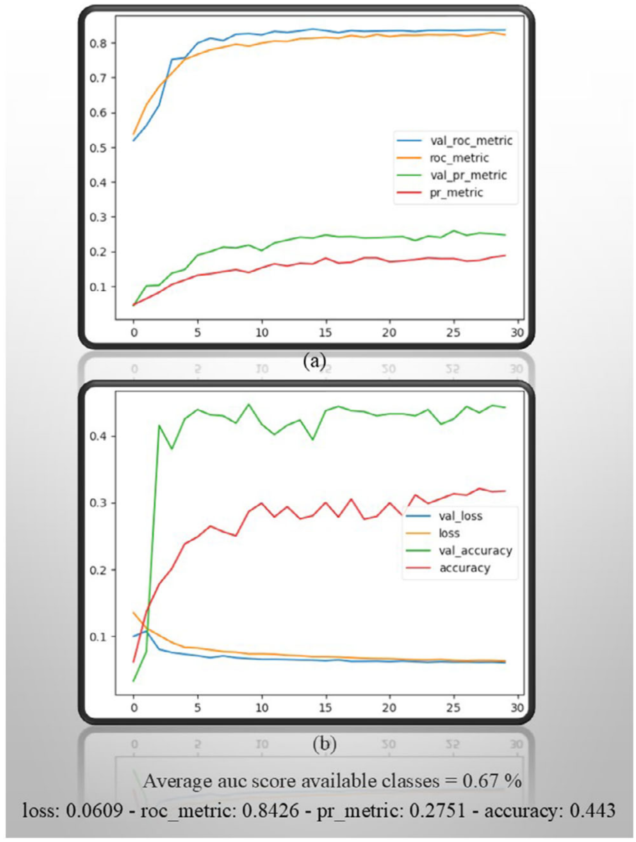

Figure 15. The result of original data for 30 epochs on the model. a) ROC and PR values during training, b) accuracy and error values during training.

distinguishing between classes, demonstrating its robustness and reliability in handling complex classification tasks.

In **Figure 18**, one can observe the weighted image fusion after morphological operations on a training sample without adaptive histogram equalization, while the application of adaptive histogram equalization is provided in **Figure 19**. These images represent different stages in the data generation and synthesis pipeline, contributing to a more diverse and robust training dataset for deep learning models. Adaptive histogram equalization is known for enhancing image contrast by adjusting the distribution of pixel intensities across smaller sections of the image. This technique improves the quality of the generated images, allowing the deep learning model to extract a wider range of features and patterns, ultimately thereby achieving improved performance.

Finally, when we trained the proposed model on 28 classes using the original training dataset, we acquired an AUC value of 85.17 on the test set. However, when the proposed model was trained with the same classification on our proposed generated dataset, we achieved an AUC of 95.44 on the test set, indicating an improvement of ≈10%. The proposed model did not suffer from overfitting during training due to the larger amount of data.

After successfully training the model with data generated using the proposed method, we applied 5-fold cross-validation on the model using the original data. **Table 2** presents the results for the detector (Disease, Not Disease) and the classifier (28 classes,

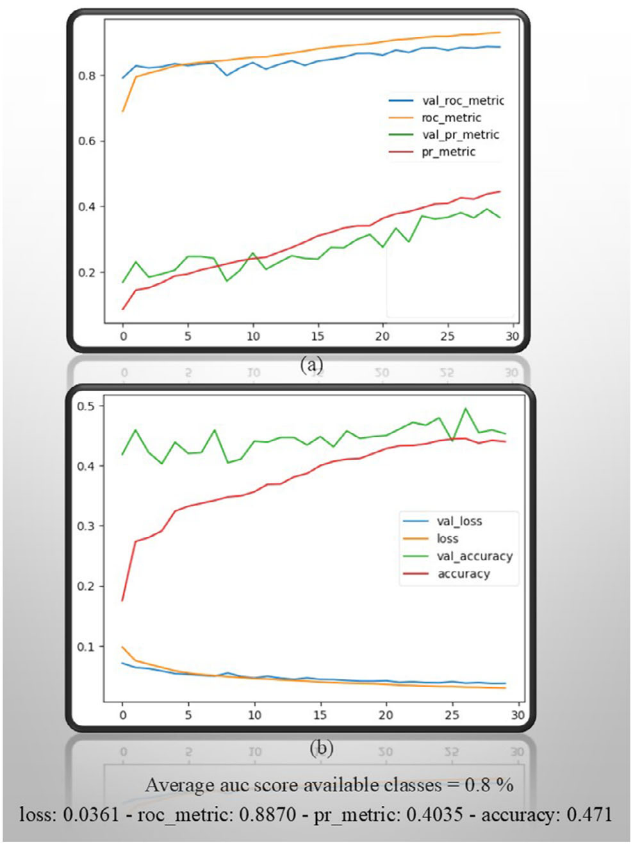

Figure 16. The results of the proposed data generation method for 30 epochs on the model. a) ROC and PR values during training, b) accuracy and error values during training.

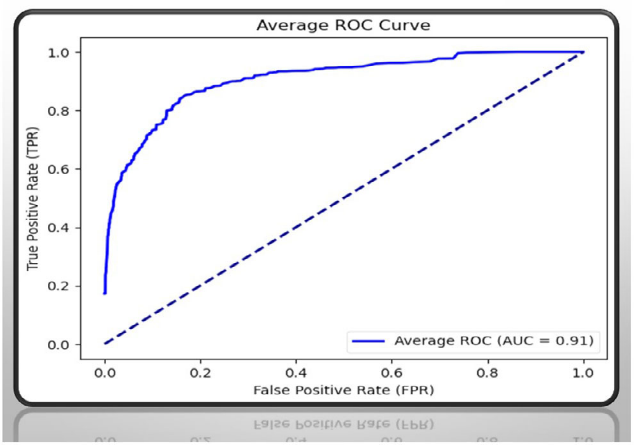

Figure 17. ROC curves for a multi-class classification with 28 classes, which involves creating a average ROC curve for all classes, considering each class as the positive class and the rest as the negative class.

as shown in Table 4C). According to **Table 3**, the proposed model outperforms other architectures based on the AUC metric.

The proposed model, along with the generated data, achieved a score of 93.5% (Average of 2 classes (disease or non-disease) and 28 classes) on the test data, improving the score recorded on the Grand Challenge website[15] by 5%, from 88.5% to 93.5%, as shown in **Figure 20**. This significant improvement highlights the crucial role of training data quantity in the model's training and accuracy. Regardless of how well a model is designed or how advanced transfer learning networks are, if the training data is insufficient, the accuracy will be suboptimal, and parameter adjustment will be challenging.

As seen in **Table 4** and **5**, ensembling and transfer learning models have high learning parameters, and they are trained on other datasets with only a few final layers trained on a limited amount of new data. There is no guarantee that they will learn the new data well or that the parameters will be properly adjusted. They may even overfit, especially if the data is imbalanced. On the contrary, if the model has appropriate parameters and sufficient data, it is possible to set the model's parameters effectively, thereby achieving good learning and accurate classification. The proposed model, with significantly fewer parameters compared to ensembling and transfer learning models, and trained with the data generated using the suggested method, achieved acceptable performance. It reduced both training and testing times and required less hardware. The proposed data generation method greatly contributed to model building, allowing for clear evaluation and correction of model performance as parameter and layer changes were made. Increasing the training data allows for more features to be extracted, improving the model's accuracy.

According to the presented findings, the DenseNet-201 model has demonstrated remarkable performance in disease classification across several studies, often achieving results comparable to ensemble models. Specifically, in Table 3 and 5, DenseNet-201 performed nearly as well as the ensemble model in disease classification. Furthermore, in Table 4, DenseNet-201 outperformed other models, including EfficientNet-B4 and ResNet-152, in classification tasks. These results highlight the effectiveness of DenseNet's densely connected architecture in improving information and gradient flow within deep networks.

Despite the positive results reported in previous studies, our proposed model achieved superior performance compared to DenseNet-201 and other state-of-the-art models. Notably, our model achieved the highest AUROC in disease classification (0.9752) and disease detection (0.9893). This improved performance is particularly significant given that our proposed model contains only three million parameters, substantially fewer than DenseNet-201's 20.2 million parameters. These results indicate that our model effectively extracts meaningful features for both classification and detection tasks through an optimized design with reduced computational complexity.

A key factor contributing to the enhanced performance of our proposed model was the use of image processing techniques in the data synthesis process. This synthesis strategy improved the model's robustness when exposed to diverse and challenging data variations. Moreover, data augmentation played a critical role in mitigating overfitting and improving parameter tuning, ultimately enhancing the model's overall performance.

In addressing the vanishing gradient problem, DenseNet's architecture effectively mitigates this issue by introducing direct connections between input and output layers, facilitating improved gradient flow. This method enhances training stability. In contrast, ResNet adopts skip connections to tackle the

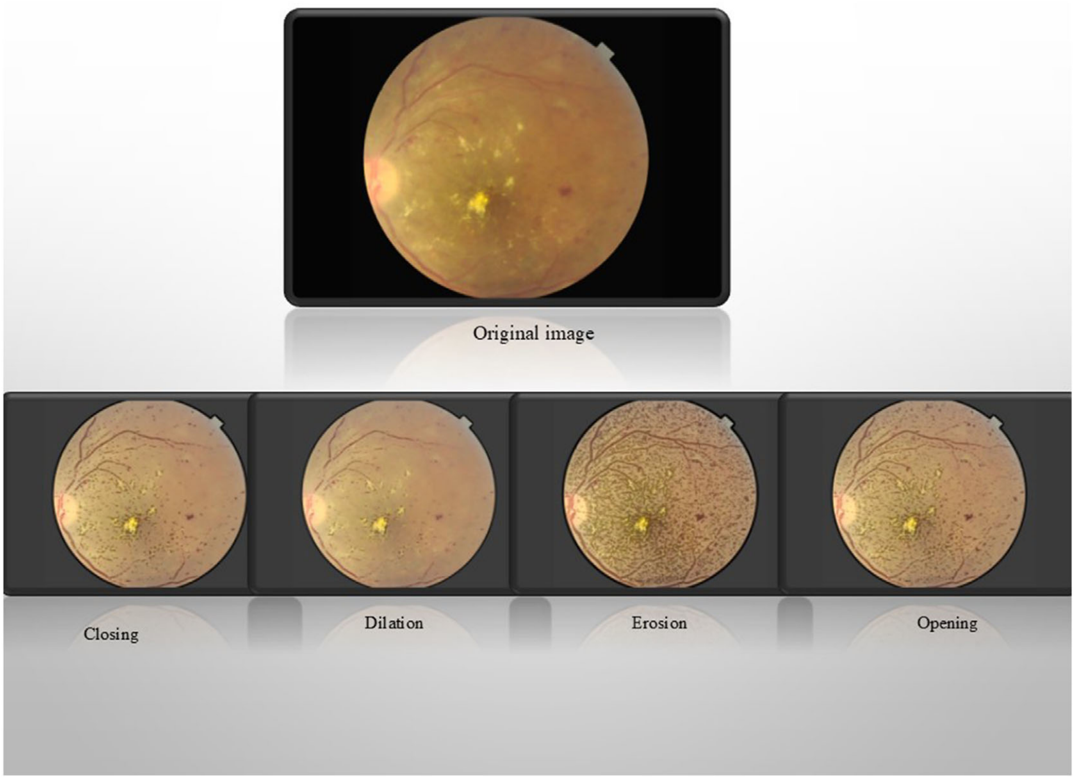

Figure 18. Weighted image fusion after morphological operations on a training sample without adaptive histogram equalization.

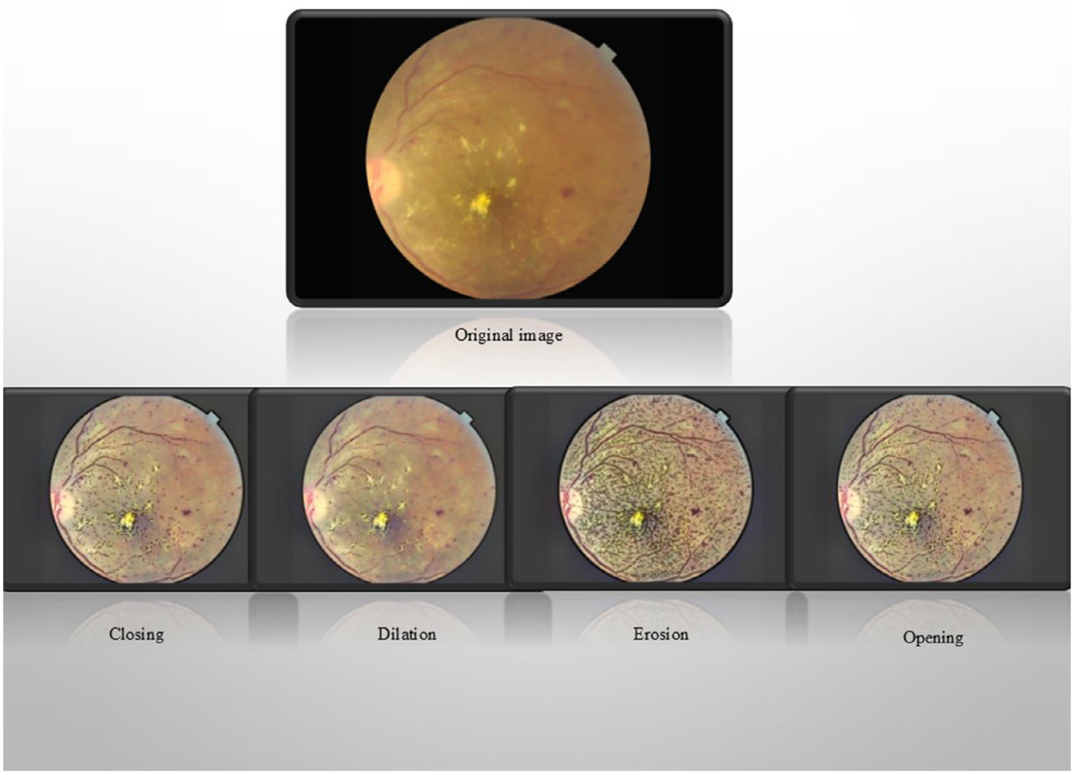

Figure 19. Weighted image fusion after morphological operations on a training sample with the application of adaptive histogram equalization.

Table 2. Result from 5-fold CV on the proposed model with original data.

k	Loss-classification (28)	Loss- detection	Classification (28) AUC	Detection AUC
1	0.0216	0.1936	0.9778	0.9849
2	0.0228	0.2263	0.9760	0.9843
3	0.0218	0.2058	0.9765	0.9913
4	0.02304	0.1186	0.9733	0.9962
5	0.0236	0.1880	0.9722	0.9895

Table 3. Performance comparison of the proposed model with other architectures based on the AUC metric.[17]

Model	Architecture	AUROC
Classification	DenseNet-201	0.9715
Classification	EfficientNet-B4	0.9666
Classification	ResNet-152	0.9697
Classification	Inception-V3	0.9215
Classification	Proposed model	0.9752
Detection	DenseNet-201	0.9685
Detection	EfficientNet-B4	0.9822
Detection	Proposed model	0.9893

vanishing gradient problem. While ResNet-152 has demonstrated reasonable performance in some studies, the results from our study and previous research indicate that DenseNet-201 outperforms ResNet-152 in both disease detection and classification tasks. This observation suggests that DenseNet's dense connections may offer superior gradient propagation and feature learning capabilities compared to ResNet's skip connections.

Table 4. AUROC scores for disease detection and classification.[16]

k	Model	Disease Detection AUROC	Disease Classification AUROC
1	Inception V3	0.9569	0.9091
2	SE-ResNeXt	0.9587	0.9066
3	DenseNet-121	0.9519	0.9298
4	EfficientNet-B4	0.9477	0.9030
5	EfficientNet-B5	0.9540	0.9163
6	Ensemble	0.9613	0.9295
7	Proposed model	0.9634	0.9452

This study emphasize that our proposed model, through its optimized architecture, effective utilization of image processing techniques for data augmentation, and improved parameter tuning, achieved superior performance while maintaining significantly fewer parameters than competing models. This highlights the importance of model structure optimization, appropriate image processing strategies, and data synthesis in enhancing the accuracy and stability of deep networks.

In this study, we proposed a data generation method that significantly enhances the training dataset by using image processing techniques. This approach increases the dataset size by 12 times through methods such as noise removal, thresholding, morphological operations, and weighted image fusion. To provide a comprehensive understanding, it is beneficial to compare this method with generative adversarial networks (GANs),[39] which are also widely used for data synthesis. One of the limitations of this study is the ability to generate a limited number of data. It is suggested that future research explore other image processing techniques, such as segmentation using different methods, to generate more data and further enhance model performance.

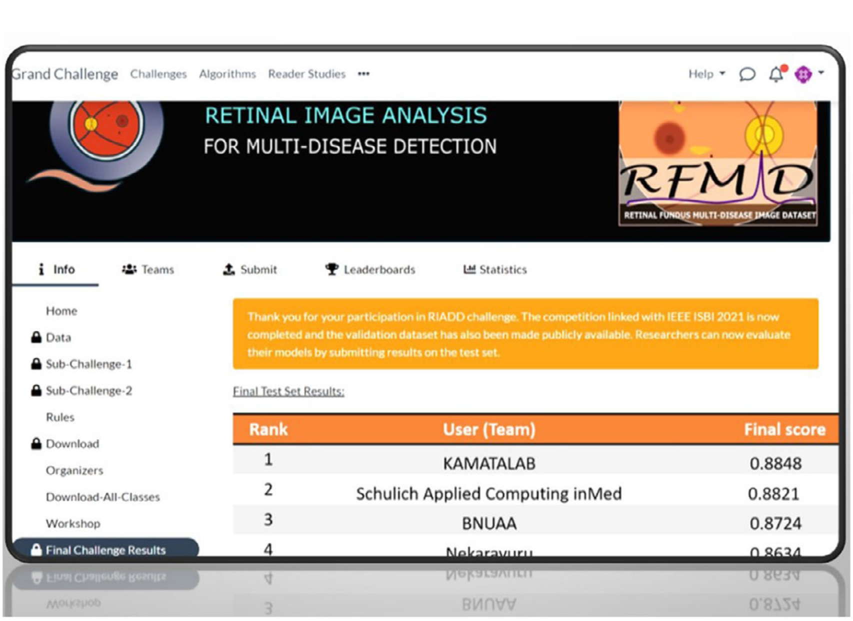

Figure 20. Final challenge results. Score recorded on the Grand Challenge website.[15]

Table 5. Parameters and AUC score in the articles.[18]

k	Authors	Model	Total param	AUC
1	Dominik Muller[19]	ResNet152	60.4 M	0.9700
		InceptionV3	23.9 M	0.9320
		DenseNet201	20.2 M	0.9730
		EfficientNetB4	19.5 M	0.9690
		Ensemble	–	0.9610
2	Amogh Jayant Dabholkar[17]	ResNet152	60.4 M	0.9697
		InceptionV3	23.9 M	0.9215
		DenseNet201	20.2 M	0.9715
		EfficientNetB4	19.5 M	0.9666
		Ensemble	–	0.9573
3	E. Sudheer Kumar[20]	Densenet201	20.2 M	0.9700
		EfficientNetB4	19.5 M	0.9600
		ResNet150	58 M	0.9700
4	Young-tack Oh[44]	EfficientNetB0	5.3 M	–
		EfficientNetB1	7.9 M	–
		EfficientNetB2	9.2 M	–
5	Omar Salman[18]	EfficientNetB4 variant 1	19.5 M	0.9491
		EfficientNetB4 variant 2	19.5 M	0.9631
		EfficientNetB4-V1V2	39 M	0.9644
		EfficientNetV2S variant 2	21.6 M	0.9417
		EfficientNetV2S variant 2	21.6 M	0.9585
		EfficientNetV2S-V1V2	43.2 M	0.9545
		FinalEnsemble	107 M	0.9730
6	–	Proposed model	3 M	0.9752

Python Codes are available in binary classes,[40] 28 classes,[41] 45 classes,[42] and Image processing.[43] These Python code repositories offer a practical resource for researchers and developers interested in implementing the proposed system or exploring its underlying concepts. By providing code for different types of classification, the system demonstrates its versatility in handling various scenarios, from simple binary tasks to complex multi-class problems.

5. Conclusion

In this study, we addressed data scarcity and gradient vanishing in deep learning for eye disease diagnosis. To combat data scarcity, we used image processing techniques for data augmentation, increasing samples twelvefold, which enhanced model performance and reduced overfitting. To prevent gradient vanishing, we employed ResNet and inception networks, optimizing weight updates for improved performance. Our method, compared to several transfer learning techniques, offers simplicity, computational efficiency, and precise control over image quality, making it suitable for resource-constrained environments. The study highlights the importance of data synthesis and neural network optimization in developing accurate models for early eye disease detection, suggesting further research into these areas for improved diagnostics in medical applications.

Appendix: Abbreviations

Table A1. Abbreviations and their definitions.

Abbreviation	Definition
DR	Diabetic Retinopathy
MH	Macular Hole
MYA	Myopic Atrophy
TSLN	Tessellated Fundus
LS	Laser Spots
CSR	Central Serous Retinopathy
TV	Toxoplasmosis
AH	Arterial Hypertension
ODE	Optic Disc Edema
AION	Anterior Ischemic Optic Neuropathy
RT	Retinal Tear
CRS	Central Retinal Artery Occlusion
RPEC	Retinal Pigment Epithelium Changes
RP	Retinitis Pigmentosa
ARMD	Age-Related Macular Degeneration
DN	Diabetic Neuropathy
BRVO	Branch Retinal Vein Occlusion
ERM	Epiretinal Membrane
MS	Macular Scar
ODC	Optic Disc Coloboma
CRVO	Central Retinal Vein Occlusion
ODP	Optic Disc Pit
ST	Stargardt Disease
PT	Papilledema
RS	Retinal Scar
EDN	Endophthalmitis
MHL	Macular Hole Large
OTHER	Other Conditions

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Samad Azimi Abriz: methodology (lead); validation (equal); visualization (lead); writing—original draft (lead). **Mansoor Fateh**: conceptualization (equal); supervision (lead); writing—review & editing (lead). **Fatemeh Jafarinejad**: supervision (supporting); writing—review & editing: (supporting). **Vahid Abolghasemi**: resources (lead); supervision (equal); writing—review & editing (lead).

Data Availability Statement

The data that support the findings of this study are available from the corresponding author upon reasonable request.

Keywords

deep learning, image processing, inception, ophthalmology, ResNet, vanishing gradient

Received: December 1, 2024
Revised: March 25, 2025
Published online:

[1] E. Parcham, M. Fateh, V. Abolghasemi, *PloS One* **2025**, *20*, e0314393.
[2] A. G. Sorkhi, H. Hassanpour, M. Fateh, *Multimed. Tools Appl.* **2020**, *79*, 18 033.
[3] A. Fateh, M. Fateh, V. Abolghasemi, *Inform. Sci.* **2021**, *581*, 479.
[4] P. Parhami, M. Fateh, M. Rezvani, H. Alinejad-Rokny, *J. Ambient Intell. Human. Comput.* **2022**, *14*, 10 883.
[5] S. Kaplan, M. A. Guvensan, A. G. Yavuz, Y. Karalurt, *IEEE Trans. Intell. Transport. Syst.* **2015**, *16*, 3017
[6] J. D. Funge, in *Artificial Intelligence For Computer Games*, Vol. *07*, Informa, London, England **2004**.
[7] M. M. Rahaman, C. Li, Y. Yao, F. Kulwa, X. Wu, X. Li, Q. Wang, *Comput. Biol. Med.* **2021**, *136*, 104649.
[8] M. Shehab, L. Abualigah, Q. Shambour, M. A. Abu-Hashem, M. K. Y. Shambour, A. I. Alsalibi, A. H. Gandomi, *Comput. Biol. Med.* **2022**, *145*, 105458.
[9] E. Parcham, M. Fateh, *Neural Networks* **2023**, *165*, 77.
[10] H. R. Khajeha, M. Fateh, V. Abolghasemi, *Eng. Rep.* **2024**, *6*, e12866
[11] P. Chlap, H. Min, N. Vandenberg, J. Dowling, L. Holloway, A. Haworth, *J. Med. Imag. Radiat. Oncol.* **2021**, *65*, 545.
[12] K. He, X. Zhang, S. Ren, J. Sun, in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA **2016**, pp. 770–778, http://openaccess.thecvf.com/content_cvpr_2016/html/He_Deep_Residual_Learning_CVPR_2016_paper.html (accessed: June 2016).
[13] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, Z. Wojna, in *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Las Vegas, NV, USA **2016**, pp. 2818–2826, https://www.cv-foundation.org/openaccess/content_cvpr_2016/html/Szegedy_Rethinking_the_Inception_CVPR_2016_paper.html (accessed: June 2016).
[14] S. Pachade, P. Porwal, D. Thulkar, M. Kokare, G. Deshmukh, V. Sahasrabuddhe, L. Giancardo, G. Quellec, F. Mériaudeau, *Data* **2021**, *6*, 14.
[15] Grand Challenge, https://grand-challenge.org/ (accessed: February 2021).
[16] E. Ho, E. Wang, S. Youn, A. Sivajohan, K. Lane, J. Chun, C. M. L. Hutnik, *Transl. Vis. Sci. Technol.* **2022**, *11*, 39.
[17] A. Dabholkar, A. Shah, P. Mehta, Deep Ensemble Model for Retinal Diseases Detection and Classification, https://amoghdabholkar.github.io/data/Stat_ML_Project_Report.pdf.
[18] O. Khan, R. Abbas, S. O. Gilani, A. Waris, *SSRN Electron. J.* **2022**, https://doi.org/10.2139/ssrn.4247846.
[19] D. Müller, I. Soto-Rey, F. Kramer, *Studies in Health Technology and Informatics*, Vol. *281* **2021**, pp. 115–119, https://ebooks.iospress.nl/volumearticle/57498 (accessed: September 2021).
[20] S. Kumar, *OSF* **2021**, *09*, https://doi.org/10.17605/OSF.IO/ZHA9C.
[21] O. Ouda, E. AbdelMaksoud, A. A. Abd El-Aziz, M. Elmogy, *Electronics* **1966**, *11*, 2022.
[22] Y. Jiang, K. Xu, X. Wang, Y. Li, H. Cui, Y. Tao, H. Lin, *Proc. 31st Int. Joint Conf. on Artificial Intelligence (IJCAI)*, Vienna, Austria, pp. 987–994, https://www.ijcai.org/proceedings/2022/0138.pdf (accessed: July 2022).
[23] P. Sahoo, S. Soltani, A. Wong, *Comput. Vis. Graph. Image Process.* **1988**, *41*, 233.
[24] E. P. Mandyartha, F. T. Anggraeny, F. Muttaqin, F. A. Akbar, *J. Phys. Conf. Ser.* **2020**, *1569*, 022054.
[25] P. Zhang, F. Li, *IEEE Signal Process. Lett.* **2014**, *21*, 1280.
[26] V. Vijaykumar, P. Vanathi, P. Kanagasabapathy, *IAENG Int. J. Comput. Sci.* **2010**, *37*, 78.
[27] P. Maragos, *Opt. Eng.* **1987**, *26*, 267623.
[28] L. Song, Y.-C. Lin, W. Feng, M. Zhao, in *2009 International Workshop on Intelligent Systems and Applications*, IEEE **2009**, pp. 1–4.
[29] B. Subramani, M. Veluchamy, A. K. Bhandari, *IEEE Trans. Emerg. Top. Comput. Intell.* **2023**, *8*, 992.
[30] S. M. Pizer, E. P. Amburn, J. D. Austin, R. Cromartie, A. Geselowitz, T. Greer, B. ter Haar Romeny, J. B. Zimmerman, K. Zuiderveld, *Comput. Vis. Graph. Image Process.* **1987**, *39*, 355.
[31] R. Rojas, in *Neural Networks: A Systematic Introduction*, Springer Science Business Media, New York **2013**.
[32] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, D. Batra, *Proc. of the IEEE Int. Conf. on Computer Vision (ICCV)* **2017**, pp. 618–626, http://openaccess.thecvf.com/content_iccv_2017/html/Selvaraju_Grad-CAM_Visual_Explanations_ICCV_2017_paper.html (accessed: October 2017).
[33] A. F. Agarap, arXiv preprint arXiv:1803.08375 **2018**, https://arxiv.org/abs/1803.08375 (accessed: March 2018).
[34] S. Santurkar, D. Tsipras, A. Ilyas, A. Madry, *Proc. of the 32nd Conf. on Neural Information Processing Systems (NeurIPS 2018)*, Montréal, Canada **2018**, pp. 2488–2498, https://proceedings.neurips.cc/paper/2018/hash/905056c1ac1dad141560467e0a99e1cf-Abstract.html (accessed: December 2018).
[35] S. Nitish, *J. Mach. Learn. Res.* **2014**, *15*, 1.
[36] A. Bria, C. Marrocco, F. Tortorella, *Comput. Biol. Med.* **2020**, *120*, 103735.
[37] S. Manna, *The Owl* **2024**, https://medium.com/the-owl/weighted-binary-cross-entropy-losses-in-keras-e3553e28b8db.
[38] M. R. Rezaei-Dastjerdehei, A. Mijani, E. Fatemizadeh, in *2020 27th National and 5th International Iranian Conference on Biomedical Engineering (ICBME)*, IEEE **2020**, pp. 333–338.
[39] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, Y. Bengio, *Neural Inform. Process. Syst.* **2014**, *27*, https://proceedings.neurips.cc/paper/5423-generative-adversarial-nets.
[40] Google Colab, Google.com **2024**, https://colab.research.google.com/drive/1nc3zNxJgCBjF01vp19HP0ZqbfbRunIxz?usp=sharing.
[41] Google Colab, Google.com **2024**, https://colab.research.google.com/drive/1KmC6gmHoUlbxSrUMD-ecngnDTgnwABTj?usp=sharing.
[42] Google Colab, Google.com **2024**, https://colab.research.google.com/drive/1RL2fhWZTk2YsKzvSIhb7dWxIlHoMvz58?usp=sharing.
[43] Google Colab, Google.com **2024**, https://colab.research.google.com/drive/1B2t932Pc7HqX-wSODkOp2XCAZfkBMn?usp=sharing.
[44] Y.-T. H. Oh, *Proc. of the Korean Institute of Information and Commucation Sciences Conf.* **2021**, pp. 614–616, https://koreascience.kr/article/CFKO202121751126220.page.