A novel convolutional feature-based method for predicting limited mobility eye gaze direction

Amal Hameed Khaleel [a,1,*], Thekra H. Abbas [b,2], Abdul-Wahab Sami Ibrahim [b,3]

[a] Department of Computer Science, Basrah University, Basrah, Iraq
[b] Department of Computer Science, Mustansiriyah University, Baghdad, Iraq
[1] amal.khaleel@uobasrah.edu.iq; [2] thekra.abbas@uomustansiriyah.edu.iq; [3] dr.wahab.sa@uomustansiriyah.edu.iq
* corresponding author

ARTICLE INFO

Article history
Received October 11, 2023
Revised March 15, 2024
Accepted March 23, 2024
Available online May 31, 2024

Keywords
Convolutional neural networks
Computer vision
Eye gaze
Haar cascade
Iris

ABSTRACT

Eye gaze direction is a critical issue since several applications in computer vision technology rely on determining gaze direction, where individuals move their eyes to limited mobility locations for sensory information. Deep neural networks are considered one of the most essential and accurate image classification methods. Several methods of classification to determine the direction of the gaze employ convolutional neural network models, which are VGG, ResNet, Alex Net, etc. This research presents a new method of identifying human eye images and classifying eye gaze directions (left, right, up, down, straight) in addition to eye-closing discrimination. The proposed method (Di-eyeNET) stands out from the developed method (Split-HSV) for enhancing image lighting. It also reduces implementation time by utilizing only two blocks and employing dropout layers after each block to achieve fast response times and high accuracy. It focused on the characteristics of the human eye images, as it is small, so it cannot be greatly enlarged, and the eye's iris is in the middle of the image, so the edges are not important. The proposed method achieves excellent results compared to previous methods, classifying the five directions of eye gaze instead of the four directions. Both the global dataset and the built local dataset were utilized. Compared to previous methods, the suggested method's results demonstrate high accuracy (99%), minimal loss, and the lowest training time. The research benefits include an efficient method for classifying eye gaze directions, with faster implementation and improved image lighting.

1. Introduction

The eyes are responsible for one of the five senses humans possess. Essentially, each person has the main parts of their eye structure. There are three layers: the sclera, the iris, and the pupil [1]. Humans sense things from birth, and they retain what they identify until they happen to stumble into it anew. The process through which the eyes sense objects and interact with the mind to create images is known as human vision. Object identification is one of the majority of difficult computer vision challenges [2]. Computer vision enables computers now to sense their surroundings and distinguish things like the human eye.

The use of computers and digital cameras rather than human eyes to visually detect, track, and measure targets is known as computer vision. First, image processing is carried out. Then comes the process of visual object tracking [3]. The identification of human eyes has long been a crucial and difficult subject in computer vision and human-computer interaction [4].

The direction of the eye's gaze is crucial in numerous computer vision applications. It is a significant indicator that offers useful information about mental processes, social interactions, and user identification. Facial recognition can utilize gaze direction to deduce mental states. Gesture recognition studies have shown that the direction of someone's gaze can affect how long they appear to be looking at something. Specifically, when someone is looking directly at something, it is regarded as lasting longer compared to when they are looking away. In virtual reality, machine learning and deep learning models have been employed to accurately identify users by analyzing eye-gaze characteristics [5], [6]. In addition, it is used in the direction of eye gaze to help people with disabilities carry out their work by moving a wheelchair or car, in addition to writing via a virtual keyboard [7], [8].

Deep Learning (DL) allows the development of powerful AI systems in a variety of domains, including computer vision and gaze recognition. By adding numerous hidden layers to a neural network, DL may find hidden patterns, extract features, and learn them. Convolutional neural networks (CNN) are a form of neural network that trains and builds hidden layers of neurons that link input and output classes for classification. CNNs are helpful learning models that can manage the complexity of works requiring difficulties that are too big to be completely defined in any dataset [9]. CNN is an important branch of deep learning that has been used to analyze information with grid-based constructions, primarily images. The CNN is built using filters that convolution the input and share the weights to build filters. With each layer, a CNN takes up increasingly complicated kernels. Edges, corners, and simple, microscopic patterns are among the foundational elements given to the initial layers. To recognize object components, the intermediary layers gain filters [10]. The final layers, however, have a higher level of representation that can distinguish between complete objects of various shapes and sizes. Deep learning with CNN has lately provided considerable performance improvements to a variety of computer vision applications [11].

CNN is a mathematical model consisting of three kinds of layers: convolutional and pooling layers, which extract features, and a fully connected layer, which produces the final output. The convolutional layer is comprised of the convolution operation and activation function, and it serves as the central component of a CNN. Additionally, the convolution layer has filters that facilitate the processing of a specific portion of data at a time and is applied to all inputs. A pooling layer reduces the dimensionality of the feature maps from the previous layer. The feature map is the result of applying a filter to the layer before it. The last component in the convolutional neural network (CNN) architecture is a fully connected (FC) layer fed from the output of the preceding layers. The final output is produced by employing an activation function like sigmoid or SoftMax [12].

A brief overview of tracking methods that are directly linked to the proposed work, which uses CNN models, is presented below:

To address this problem, Wang et al. [13] suggested a gaze estimate approach using CNN and randomly generated forest regression as opposed to manually created features. The hybrid approach (deep features) for examining CNN-based image features was utilized to teach a random forest regressor how to map deep characteristics to gaze locations.

Another study [14] developed a hardware-friendly CNN-based model that uses a minimal computing demand for effective gaze estimates on low-consumption devices, while [15] presented a stages two-stage CNN method of learning for integrating head positions and various angles of view. When directly training the head position to eye-tracking models, the CNN architecture can decrease refit. Also, [16] used a Mask-RCNN approach for labeling the eyes, with three objects employed to track gaze (left, center, and right).

The images captured by the unaltered cameras are of poor quality and extremely sensitive to changes in brightness. Therefore, in research, [17] used an unaltered camera to follow a person's gaze. the state-of-the-art CNN network was utilized along with unmodified cameras routinely used with PCs. As a result, computing the precise gaze point was reduced to identifying the score as a member of one of 20 regions rather than treating it as a regression problem.

Another study [6] presented a system that can identify users by using only a few eye-gaze-based features without providing any tasks that are specifically tailored for identifying purposes. The study gathered gaze data from an educational virtual reality application and evaluated the system using two machine learning models, namely random forest and k-nearest-neighbors, as well as two deep learning models: convolutional neural networks as well as long short-term memory. Based on the results, machine learning and deep learning models were able to correctly identify users with over 98% accuracy using just six basic eye-gaze traits.

In this research, a new model was proposed for CNN to classify the four eye directions, in addition to eye gaze forward and eye closure, and then it was applied to a local dataset created with a Haar cascade method and a global dataset.

The following objectives have been established to accomplish the main goal:

- Cost reductions can be achieved by using a standard webcam instead of expensive hardware.
- To lower processing complexity, the local database is built to capture real-time images and enhance the lighting in limited mobility locations.
- To achieve high accuracy, the study focuses on the small size of the eye area and uses CNN models to determine gaze direction using the movement of the upper eyelid and iris eye.

2. Method

2.1. Background

Several methods of CNN were employed in the proposed approach to compare the results, as explained as follows.

2.1.1. VGGNet

The Visual Geometry Group at Oxford University created the CNN architecture known as the VGGNet [18]. It demonstrates the significance of network depth to improve the result by incorporating additional convolution, pooling, and fully connected layers as seen in Fig. 1.

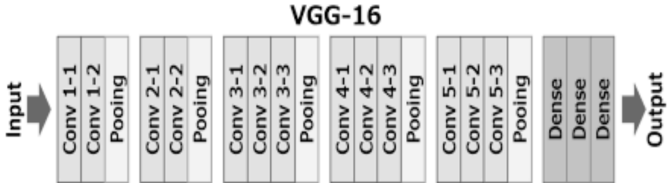


Fig. 1. VGG16 Architecture [19]

VGGNet is a large-scale image classification model with up to 16 or 19 layers that may yield more convolutional feature maps than shallow CNNs while using fewer processing resources than deeper CNN designs. VGG-Net, which is made up of many convolutional layers, allows you to describe the target from various angles with deep layers encoding semantic data for classification and thin layers providing localized features for recognition. On the other hand, VGGNet has significant limitations. Due to its "140 million" parameters, it requires a vast amount of memory and is expensive to assess exhausts [20].

In this research, the VGG-16 was used instead of the VGG-19, because too many layers lead to the fading of the image, and this negatively affects the results. The following procedures were put into practice to set up a VGG16 pre-trained model:

Step 1: For each entered fundus image with (224 x 224 x 3), the feature map extracted was (7 x 7 x 512).

Step 2: Added two FC layers (128 nodes in the first layer and six nodes in the output).

Step 3: A SoftMax activation function was added to the last layer for six classes of prediction. Both the convolutional and FC layers employed ReLU as a function of activation

2.1.2. ResNet

Microsoft Research Asia unveiled ResNet in 2015 [21]. ResNet consists of residual connections that allow ConvNet to learn from the input of the layer before it in addition to the output of the activation function (Fig. 2). Because of the operations that distribute the gradients, information is moved from one layer to the next using a simple gradient flow with backpropagation. ResNet revealed that architecture complexity is not required to produce the best results, but that more basic and deep designs may be modified to achieve outstanding results [22].

In principle, ResNet network efficiency is expected to improve as network depth increases. However, in real-world applications, residual networks can alleviate the problem when the CNN's depth reaches a certain point and layer-dependent network efficiency and speed of convergence start to fall. The residual block is the key innovation of residual learning, which may be characterized using Equation (1) as shown in Fig. 2.

$$H(X_i) = R(X_i) + X_i \tag{1}$$

Here $H(X_i)$ is the output, (X_i) denotes the residual component, and xi denotes the sample.

By stacking numerous residual blocks, ResNet successfully recovers features from input data, achieving excellent accuracy in many classification applications [23].

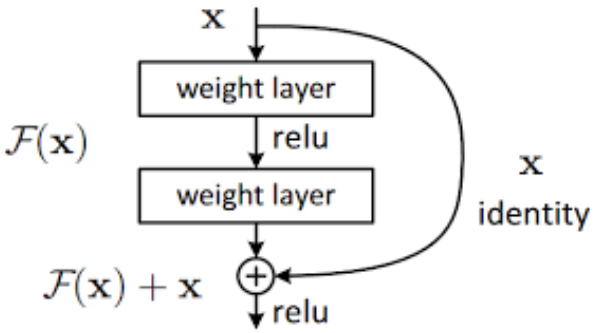


Fig. 2. Residual Blocks [24]

ResNet is a deep CNN network that is integrated with image, auto-encoding, and classification capabilities. The residual network, which has a total of 177 layers, was constructed from a residual network design with 50 layers. The weights of the network's first through 174th layers were frozen. The parameters of the frozen layers' set are not altered by the trained network. To greatly speed up network training and prevent over-fitting to the upcoming dataset, several early layer weights can be frozen, moreover, it's more suitable for images with a small size for the human eye.

To use the ResNet-50 pre-trained model in this paper, it performs the following steps:

Step 1: Using ResNet-50 convolutional part as a convolutional feature extractor.

Step 2: The FC Layer with node = 6 as the output layer.

Step 3: SoftMax was used for the prediction layer

2.1.3. Hear Cascade

Haar cascade is a protest detection method that is typically used for face detection [25]. It provides high-speed calculations that rely on the total number of pixels inside the rectangle feature rather than using the individual pixel values in the image. This method uses the integral image, the AdaBoost learning, the Cascade Classifier, and the Haar-like feature to recognize objects [26]. The heart of the

Haar cascade classifier for face identification is Haar features, which are used to find the presence of a feature in a given image. The instances of common Haar-like features are depicted in Fig. 3

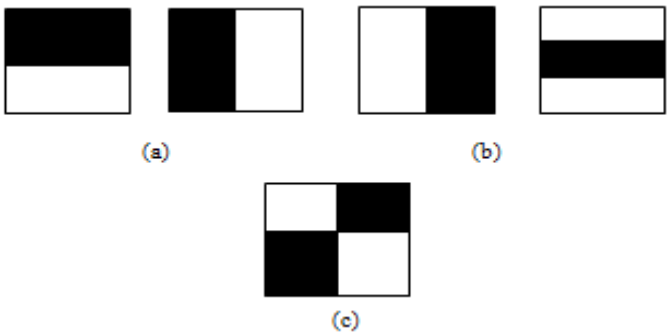


Fig. 3. Universal features of a Haar-like (a) Edge, (b) Line, (c) Four-Triangle feature

Each feature yields a single value, that is obtained by summing the pixels in the black rectangle. A rectangular characteristic known as the Haar-like feature gives a specific hint to an image to aid with face identification [27], [28].

The object detection value was calculated utilizing an integral image and the Haar-like feature value. It begins scanning an image from the top left corner to find faces in it, and it stops at the right bottom of the image. By creating a new image representation depending on the value of an already defined region utilizing a certain Haar-like feature, the integral image can reliably and quickly find values. The value at each location (xi, yi) is equal to the total of the area tables of all pixels located above and to the left of (xi, yi), as indicated in Eq. (2).

$$IM(xi, yi) = \sum_{\substack{xi \\ yi}} {}_{<}^{<} {}_{yi}^{xi} im(xi, yi) \tag{2}$$

Where $IM(xi, yi)$ is the pixel value at (xi, yi) and im(xi,yi) is the sum of the integral of pixel values. The value of the integral image, $IM(xi, yi)$ is derived by adding the previous index values from left to right. Furthermore, the summed-area table could be effectively calculated in just one motion over the image, as shown by the number in the summed-area at (xi, yi) in Eq. (3) [29].

$$IM(xi, yi) = im(xi, yi) + IM(xi, yi - 1) + IM(xi - 1) + IM(xi - 1, yi - 1) \tag{3}$$

After the summed area has been calculated, assessing the sum of intensities across every rectangular area regardless of area size takes exactly four array references. The total of $I(xi, yi)$ across the rectangle spanned by Ai, Bi, Ci, and Di is shown in Eq. (4)

$$\sum_{\substack{xio<xi<xi1 \\ yi0<yi<yi1}} im(xi, yi) = IM(Di) + IM(Ai) - IM(Bi) - IM(Ci) \tag{4}$$

The value acquired via the use of an integral picture is then contrasted with the threshold value set by AdaBoost for particular attributes. This is done to discover potential characteristics since not all features are relevant for specific item recognition. AdaBoost merges weak classifier features into a powerful classifier. Cascade classifiers are classified into two types: powerful classifiers and poor classifiers. A poor classifier predicts something that is less accurate or unimportant, whereas a strong classifier predicts something more accurate or relevant. AdaBoost's strong classifier can recognize objects in a cascade, level by level [30]. Since lighting and position have an impact on the accuracy of the Haar cascade classifier approach, consideration has been given to these two factors in this research to obtain good findings and an organized image dataset.

The experiences in this paper explain shows the Haar cascade predictor is unable to distinguish between face and eyes in persons who are not sitting straight and looking at the camera. Therefore, they must sit up straight to obtain an excellent result. Furthermore, the quality of images captured from the

webcam in real-time is affected by the lighting, so this paper proposes a new method to increase the lighting.

2.2. Proposed Methods and Architecture

The method of work is divided into three stages. The first stage is to create a real-time dataset. The second stage involves pre-processing the obtained dataset images by modifying the size and lighting and normalization of the image data values. Finally, the images are divided into six classes in the third step based on the proposed procedure.

2.2.1. Dataset

Two types of datasets were used (global and local), as follows:

- Global Dataset

  The SBVPI dataset (Sclera Blood Vessels, Periocular, and Iris) is divided into two components [31]. As seen in Fig. 4, the first segment provides periocular images, while the second contains sclera images.

  The first portion of the SBVPI utilized in this work has 1800 RGB images of 55 participants, sexuality (male or female), eye type (right or left), and labels for gaze/view direction (right, straight, left, up), with each gaze-direction having 450 images with sizes 1700 3000 px [32], [33]. Images for the dataset were taken with a Digital Single-Lens Reflex camera at the maximum quality and resolution level during a single recording session [34].

Fig. 4. Global dataset

- Local Dataset (Real-time)

  The suggested technique produced a local compilation of human eye imagery that included eye motion in four distinct orientations as well as the state of straight and eye closure. This study involves the use of 4500 images collected by six different individuals (young: male, female; child: male, female). Every image was taken under indoor illumination with different colors of skin for each person, as shown in the samples in Fig. 5.

  The images were captured with a web camera at 30 fps at 720p. The device's screen is perpendicular, and the distance between the camera and the eye is no more than 50 cm. The Haar cascade technique is used in this study to determine the eye in an image.

It's worth noting that the employed technique centered on the observation that the upward eye motion in typical cases is not significantly distinct from the forward eye motion, as the iris does not elevate considerably. Similarly, the downward eye motion closely resembles the closing motion. So, neural networks have a vital role in determining the direction of the human eye (upward, forward, or downward) by analyzing the characteristics of the upper eyelid and the extent of iris movement.

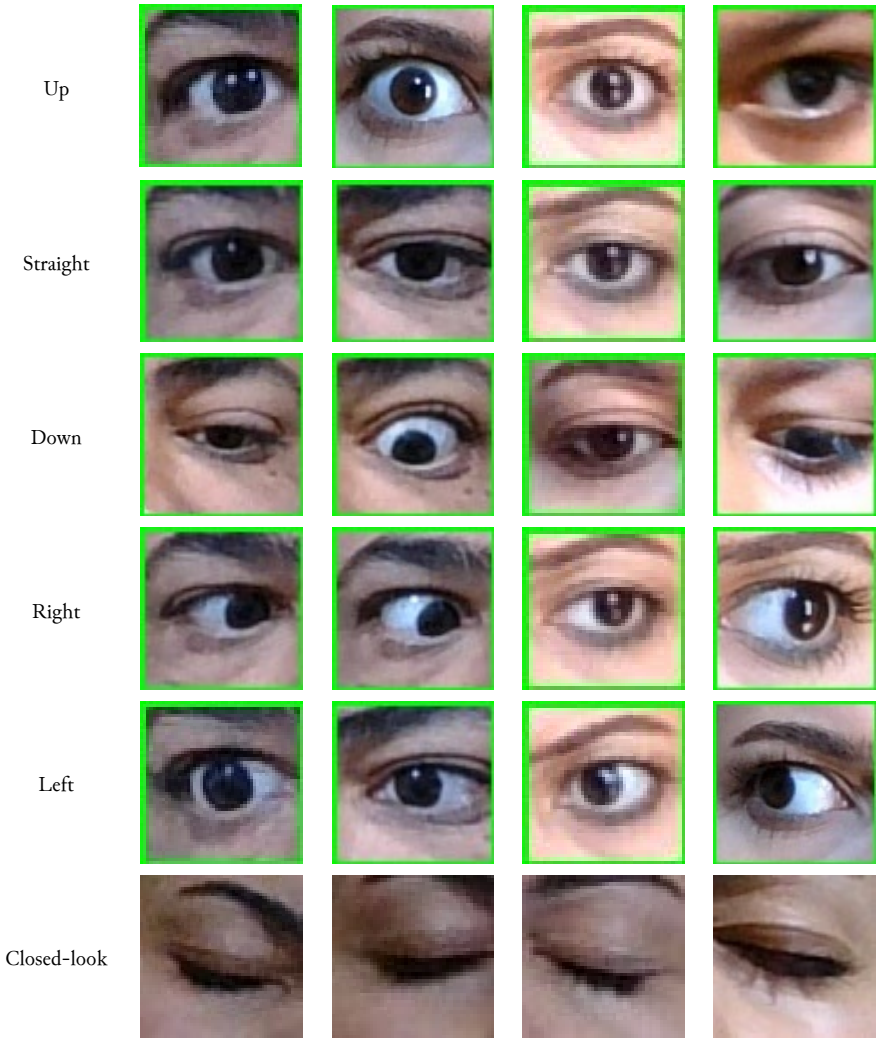


Fig. 5. Local dataset

2.2.2. Preprocessing of Eye Images

Preprocessing is always utilized in all stages of image processing to get the correct region for recognizing an object. In this study, the initial stage of preprocessing is resizing the image to 64×64 of the original size to recognize the region of an eye after defining both right and left eye areas using the Haar cascade algorithm. The Haar cascade technique has, however, various shortcomings, including the ability to detect a region that is not an eye. In this scenario, it enhances the lighting.

A novel self-regularized approach for lowlight image enhancement is suggested that is inspired by the HSV color, which has three channels: H for hue, S for saturation, and V for value. It keeps all of the colors (hue) while changing the saturation and brightness (value).

The method was developed to keep all of the colors while increasing the saturation and brightness. Color composition relies primarily on saturation and brightness, with 100% saturation signifying purity, 0% indicating grayscale, and 100% brightness indicating no black mixed with the color, which helps to correctly restore the colors hidden in the darkness and decreases complexity.

The following steps are taken by the suggested method (Split-HSV):

- Color space conversion from BGR to HSV
- Extract the channels for saturation (s) and value (v) and increase both of them to the required extent, as detailed below:

  value=50

  valadj = 255 - value

  v[v >valadj] = 255

  v[v <= valadj] += value

  saturation=1.5

  satadj=s*saturation

  s[s >satadj] = 255

  s[s <= satadj] += saturation

- Channels are re-merged. (HSV=H+S+V)
- Convert back to BGR

Through experiments, the values of brightness and saturation that were chosen were the best, but it is possible to increase their values if there is very little lighting or the webcam has a low resolution when taking images.

In the second step of preprocessing, each left and right eye region is transformed into a grayscale image to limit the degree of RGB colors. The third stage of preparation is averaging the blurring method is used to smoothen images and eliminate noise from images.

2.2.3. Dataset

Finally, the classification is performed by using the proposed model Di-eyeNET Architecture, and the output is six classes of images. The following is the work method, as shown in Fig. 6.

- Create dataset: To speed up the eye identification process, and because it doesn't need face identification, it used the Haar cascade approach to determine the eyes without defining the face. The eye area was found for colored images by enclosing the coordinates with a rectangle of the eye region estimated via the Haar cascade approach, where the image captured from the webcam was converted to a grey color to determine the eye area. A dataset of six classes (up, down, left, right, straight, and eye closed) was constructed, with each class including 750 images, resulting in a dataset of 4500 images. Where the left and right eye regions were processed independently.
- Image pre-processing through two methods
  - Enhancement of the lighting images using the suggested method (Split-HSV)
  - Resize the image to 64×64. Since the human eye is small, it enlarges as much as possible after performing many trials with a focus on the images after enlargement without distortion.
  - Image normalization to reduce the number of distributions in the input layer.
- Data Splitting: After preparing a dataset, it is divided into three main components: "training, validation, and testing". For the testing set, 10% of the data was separated. The remaining data (90%) were randomly assigned to 75:25 training groups and validation groups, with 3038, 1012, and 450 images, respectively, for training, validation, and testing. Furthermore, the number of classes has been limited to six.

- Create a model (Di-eyeNET): The Di-eyeNET structure of the model is based on current CNN model performance for multiple recognition tests and incorporates design considerations that have shown to be effective for a variety of computer vision applications.

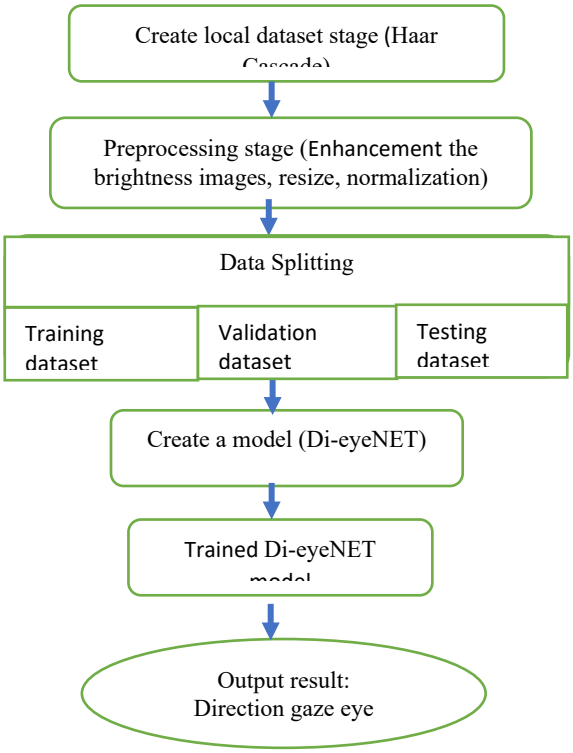


Fig. 6. Work method

The architecture is divided into two blocks, as shown in Fig. 7 and summarized in Table 1. To gather sufficient spatial information and reduce the dimensionality of the generated feature maps, the first block comprises two convolutional layers that use 128 big 5×5 filters having a stride of 2. The layer is followed by a max-pooling layer, which uses ReLU activations to reduce the size of dimensions along the feature map by a factor of 2×2.

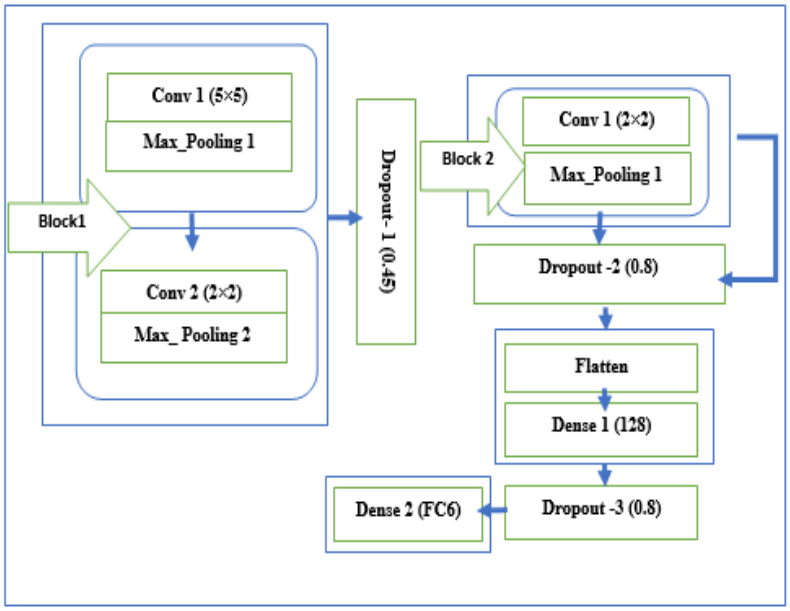


Fig. 7. Architecture of Di-eyeNET

Table 1. Summary of the (Di-eyeNET) Model

Layer (type)	Output Shape	Parameters
conv2d 1 (Conv2D)	(None, 60, 60, 64)	4864
max_pooling2d (MaxPooling2D)	(None, 12, 12, 64)	0
conv2d_2 (Conv2D)	(None, 11, 11, 64)	16448
max_pooling2d_1 (MaxPooling 2D)	(None, 5, 5, 64)	0
dropout 1 (Dropout)	(None, 5, 5, 64)	0
conv2d_3 (Conv2D)	(None, 4, 4, 32)	8224
max_pooling2d_2 (MaxPooling 2D)	(None, 2, 2, 32)	0
dropout_2 (Dropout)	(None, 2, 2, 32)	0
flatten (Flatten)	(None, 128)	0
dense 1 (Dense)	(None, 128)	16512
dropout_3 (Dropout)	(None, 128)	0
dense_2 (Dense)	(None, 6) in Local	774
	(None, 4) in global	516

Next, the second block is used, which includes a max-pooling layer with 2×2 filters and a convolutional layer. As a result of the max-pooling layers, the spatial dimensions of the feature maps are halved after block 1. To ensure that the feature maps have sufficient representational capacity, it increases the number of filters in the convolutional layers after each max-pooling operation. The last two blocks' outputs are sent to a flattened layer, which in turn is directed to a 128-dimensional fully connected (FC) layer. To keep performance high while limiting the number of trainable parameters, the proposed architecture aims to avoid having numerous FC layers. The FC layer is the final one and is coupled to one SoftMax layer that decides the six eye orientations for the MTL training phase.

- To speed up the training process while assuring accuracy and presenting near results in training and validation, the Relu activation function, Adam optimizer, and epochs of several 100 were utilized, as well as a batch size of 16.
- The two-stage procedure was utilized to extract only the relevant features while ignoring the others: block 1 of the first stage has two convolutional layers. In the second stage in block 2, in which the convolutional layers were reduced to half (i.e., just one convolutional layer was employed), since, after the first stage, it made the dropout of decreasing almost half of the variables (characteristics), dropout was employed after block 2 to reduce by around a quarter.
- The traits were finally categorized using a dense layer in the final step. Its prediction has six classes (left, right, top, down, straight, and closed-look).

3. Results and Discussion

This section describes the specifics of the suggested implementation and performs comparisons with the results of previous CNN architectures using local and global datasets.

3.1. Experimental Setup

The proposed model is implemented in Python 3.9 with an open CV2 environment. All the experiments are conducted on a PC with an Intel(R) core (TM) i7 (2.30 GHz) CPU, 16 GB RAM, an NVIDIA GTX GEFORCE graphics card, and a Windows 10 pro–operating system.

3.2. Types of Experiments

Experiments were carried out on two types of datasets: local (collected in real-time) and global (downloaded from "http://sclera.fri.uni-lj.si/").

These experiments were performed on the suggested model, and the outcomes were contrasted with those attained using datasets on the most important convolutional neural network models.

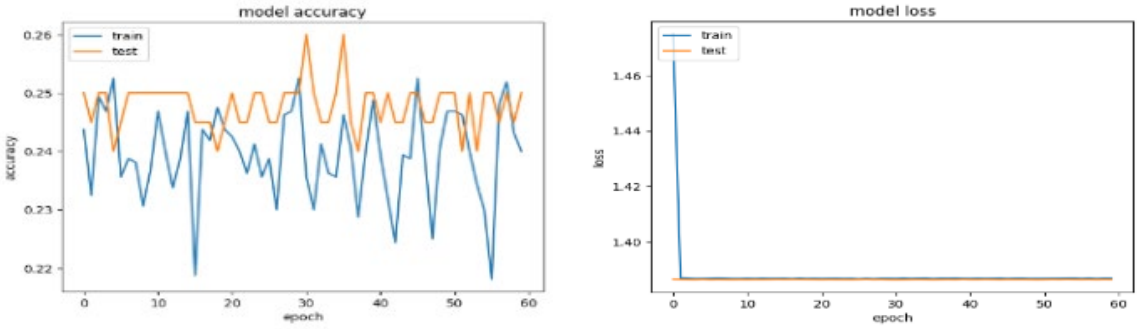
A VGG-16 model was chosen because it is considered one of the most accurate models due to its large number of layers, and the ResNet-50 model was also used because it reduced the number of layers to increase speed and increased accuracy due to its overlapping structure.

3.2.1. VGG-16 Experiments

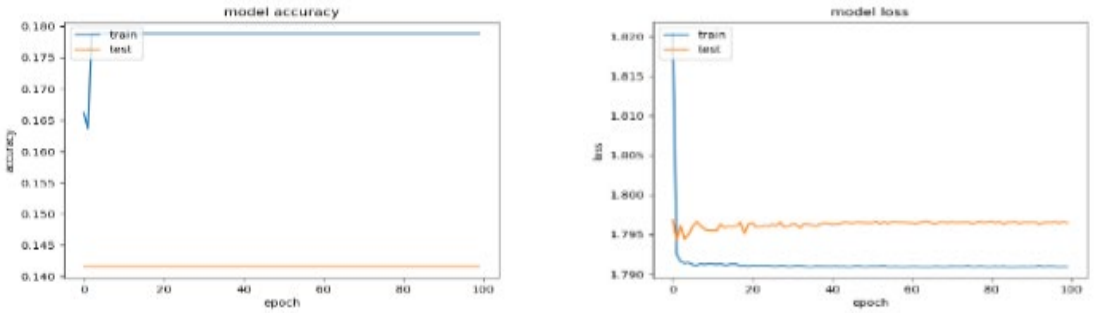
Model VGG-16 was utilized instead of Model VGG-19 to reduce the number of layers and speed up training. Table 2 contains a summary of the training procedure, whereas Fig. 8 depicts metrics on performance.

Table 2. Summary of the Training process of the VGG-16 model

Parameters	Global	Local
No. images in total	1806	4500
No. images of trains	1604	3600
No. images for testing	202	900
No. available class	4	6
No. parameters	134,276,932	134,285,126
No. trainable parameters	134,276,932	134,285,126
Untrainable- Parameters	0	0
Epoch	100	100
Size batch	32	32
validation loss	1.3863	1.7965
validation accuracy	0.2500	0.1416
EarlyTermination	After 60 epochs Val-loss not improved	After 50 epochs Val-loss not improved
Optimizer	Adam	Adam
Loss Function	Categorical cross-entropy	Categorical cross-entropy



a. Accuracy and Loss Plots for the VGG-16 in the Global Dataset



b. Accuracy and Loss Plots for the VGG-16 in the Local Dataset

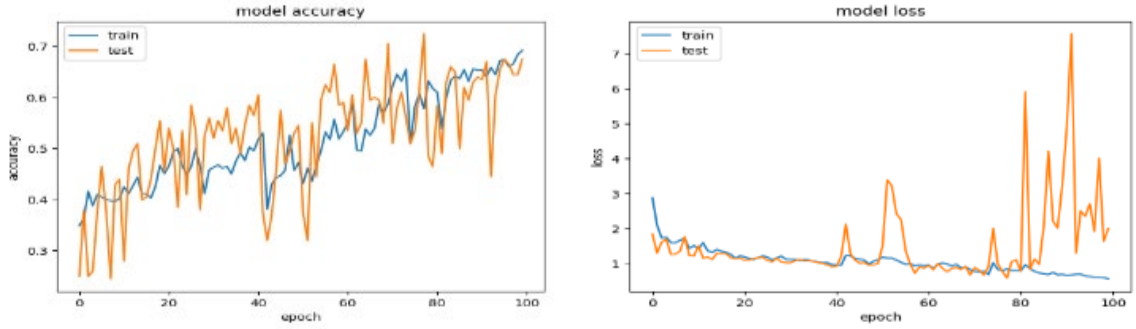
Fig. 8. Experimented with VGG-16 model performance during the training progress using training and validation datasets

3.2.2. ResNet-50 Experiments

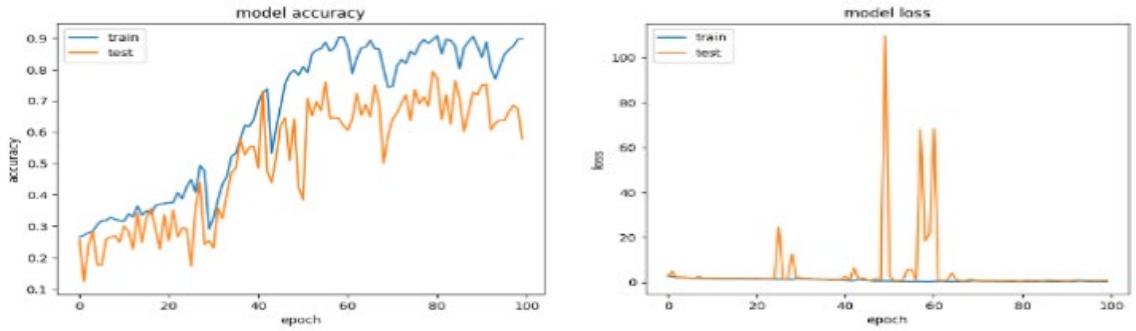
ResNet-50 was used instead of ResNet-100 or ResNet-101 in an attempt to lower the number of layers to improve training speed while also avoiding fading of the trained images due to their small size. Table 3 contains a summary of the training procedure of ResNet-50, whereas Fig. 9 depicts metrics on performance.

Table 3. Summary of the Training process of the ResNet-50 model

Parameters	Global	Local
No. images in total	1806	4500
No. images of trains	1604	3600
No. images for testing	202	900
No. available class	4	6
No. parameters	23,595,844	23,599,942
No. trainable parameters	23,542,724	23,546,822
Untrainable- Parameters	53,120	53,120
Epoch	100	100
Size batch	32	32
validation loss	0.5970	0.5532
validation accuracy	0.7250	0.7933
EarlyTermination	After 78 epochs Val-loss not improved	After 80 epochs Val-loss not improved
Optimizer	Adam	Adam
Loss Function	Categorical cross-entropy	Categorical cross-entropy



a. Accuracy and Loss Plots for the ResNet-50 in the Global Dataset



b. Accuracy and Loss Plots for the ResNet-50 in the Local Dataset

Fig. 9. Experimented ResNet-50 model performance during the training process using training and validation datasets

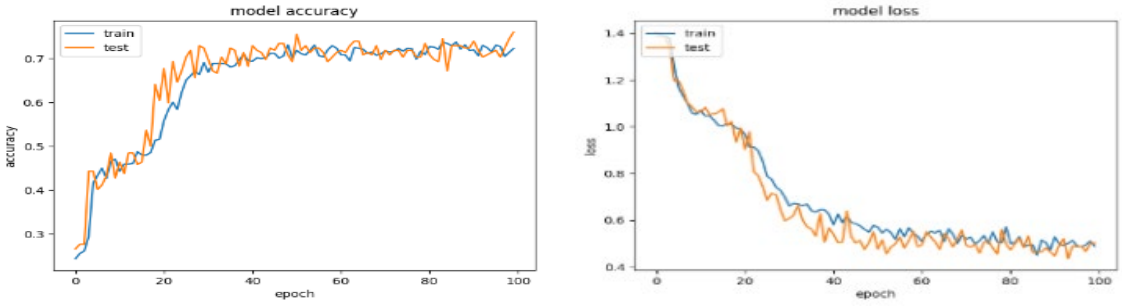
3.2.3. Di-eyeNET experiments

A novel suggested model was used in which the number of layers is low to improve training speed while also avoiding the fading of the learned images due to their small size.

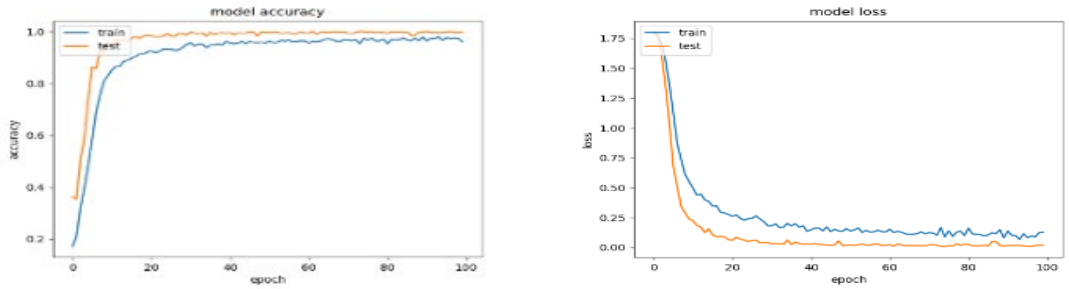
The dropout method was used twice to decrease the number of characteristics and use just the most significant ones. Table 4 contains a summary of the training procedure of Di-eyeNET, whereas Fig. 10 depicts metrics on performance.

Table 4. Summary of the Training process of the Di-eyeNET model

Parameters	Global	Local
No. images in total	1806	4500
No. images of trains	1604	3600
No. images for testing	202	900
No. available class	4	6
No. parameters	46,564	46,822
No. trainable parameters	46,564	46,822
Untrainable- Parameters	0	0
Epoch	100	100
Size batch	32	32
validation loss	0.0212	0.0183
validation accuracy	0.9918	0.9959
EarlyTermination	After 91 epochs Val-loss not improved	After 98 epochs Val-loss not improved
Optimizer	Adam	Adam
Loss Function	Categorical cross-entropy	Categorical cross-entropy



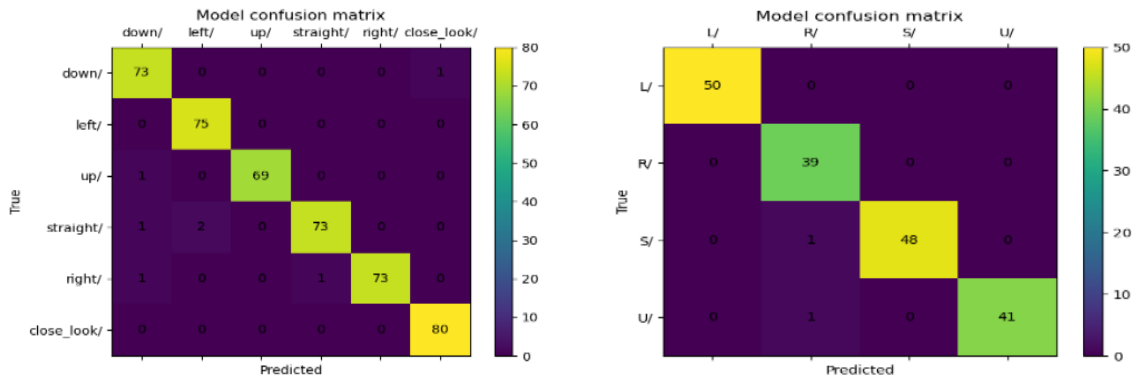
a. Accuracy and Loss Plots for the Di-eyeNET in the Global Dataset



b. Accuracy and Loss Plots for the Di-eyeNET in the Local Dataset

Fig. 10. Experimented Di-eyeNET model performance during the training process using training and validation datasets

The confusion matrix summarizes all necessary information for model evaluation. Numerous metrics may be computed using it [35]. True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) are the four types of metrics. To make the criterion computation clearer, Fig. 11 displays the confusion matrix for multi-class classification (six-class, four-class).



Six-Class Four-Class

Fig. 11. The Confusion Matrix

Assess the effectiveness of the suggested model utilizing typical performance metrics, which include (accuracy, recall, and the F1-score) are explained below. [36]–[39].

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{5}$$

$$Precission = \frac{TP}{TP+FP} \tag{6}$$

$$Recall = \frac{TP}{TP+FN} \tag{7}$$

$$F1 - Score = 2\frac{precission.recall}{precission+recall} \tag{8}$$

The performance metrics for the Four-Class global dataset and the Six-Class local dataset are shown in Table 5 and Table 6, respectively. In Table 5, the classes (straight, down) have lower accuracy than the classes (left, right, up, close_look) because straight and down directions are similar to close_look direction.

Table 5. The Performance Metrics with Six-Class of Local Dataset

Class	Precision	Recall	F1-score
1 (down)	0.96	0.99	0.97
2 (left)	0.97	1.00	0.99
3 (up)	1.00	0.99	0.99
4 (straight)	0.99	0.96	0.97
5 (right)	1.00	0.97	0.99
6 (close_look)	0.99	1.00	0.99

while in Table 6, only the right direction is low accuracy.

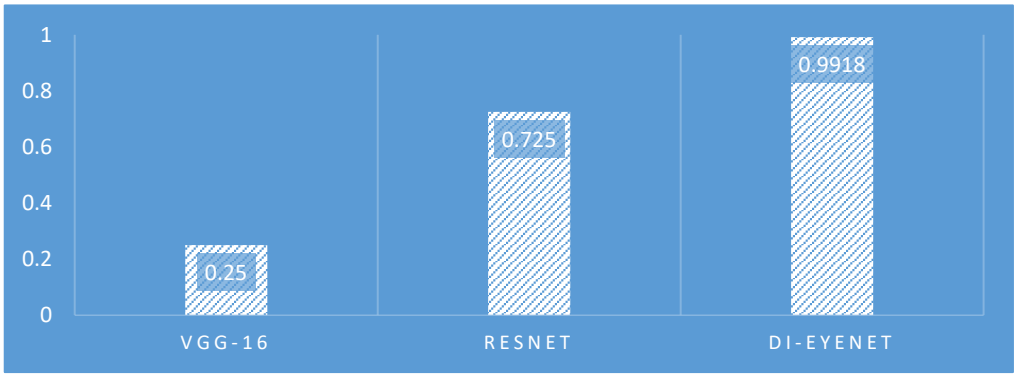
Table 6. The Performance Metrics with Four-Class of Global Dataset

Class	Precision	Recall	F1-score
1 (left)	1.00	1.00	1.00
2 (right)	0.95	1.00	0.97
3 (straight)	1.00	0.98	0.99
4 (up)	1.00	0.98	0.99

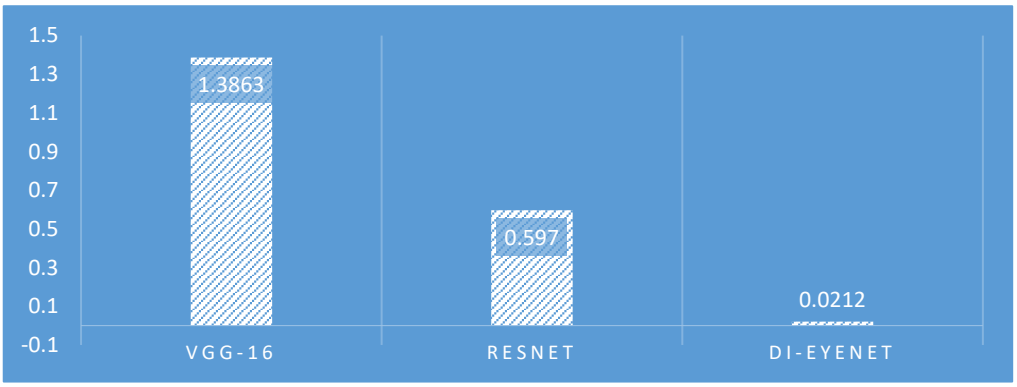
The final result of the suggested model is excellent in comparison to previous models, as shown in Table 7 and Fig. 12.

Table 7. Comparison of Proposed Model Result with the Two Strategies

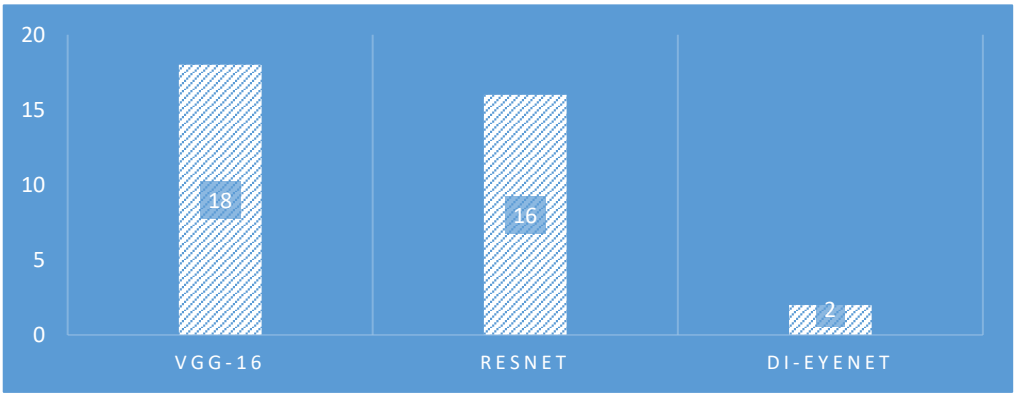
Models	Dataset	Val_Loss	Val_Accuracy	Parameters
VGG-16	Global	1.3863	0.2500	134,276,932
	Local	1.7965	0.1416	134,285,126
ResNET	Global	0.5970	0.7250	23,595,844
	Local	0.5532	0.7933	23,599,942
Di-eyeNET	Global	0.0212	0.9918	46,564
	Local	0.0183	0.9959	46,822



a. Accuracy



b. Loss



c. Training time

Fig. 12. Comparison of proposed model result with the VGG-16 & ResNet

3.3. Discussion

The VGG-16 model was implemented on two datasets (Local and Global). It was noticed, as seen in Table 2 and Fig. 8, that a comparable number of parameters were used, and there were none of the untrainable variables. when the same number of epochs and batch size were used, the global dataset's accuracy was somewhat higher than the local datasets, but the loss rate was higher, so the images faded and were not finished for the hundred epochs.

The ResNet-50 model was implemented on the two datasets; it was noticed that about the same total number of parameters was employed. Since the same number of epochs and batch size were used in both datasets, the accuracy was almost equal, even though the local dataset had more classifications than the global dataset did, by a significant margin. The proposed model's final result is pretty good when compared to other models, as seen in Table 3 and Fig. 9.

The Di-eyeNET model was implemented on both datasets, and it was noticed that virtually the same small number of parameters were needed. So, the dropout method was used to reduce and use just the most important features of the eye. Because the local dataset's images were captured with slightly more characteristics than the global dataset's images, the accuracy and loss results for the same number of epochs and batch size were marginally better, as seen in Table 4 and Fig. 10. Moreover, Fig. 11 explains that the class of eye closure that is noted in the confusion matrix is more accurate since the upper eyelid is closed, making determining its characteristics clearer. Also, because the movement of the upper eyelid in the top class matches that of the straight class, the classification results less accurately belong to the top class. The direction of the gaze in the global dataset is clear because the direction of the iris of the eye differs greatly between the directions of the different gazes of the eye. So, it is noticeable in Table 7 that all classes have almost the same degree of accuracy. According to Fig. 12, the suggested strategy was more accurate and had a lower loss than the prior strategies, and the suggested model's training time is the quickest because it uses fewer layers in its construction. Finally, in comparison to previous methods, the suggested method's results demonstrate high accuracy (99%), while the highest accuracy with the previous methods reached 98%, as in research [6].

4. Conclusion

One of the more noticeable characteristics of a human face is its eyes; the eyes and their movements are important in communicating a person's wants, thinking processes, and feelings. The importance of eye movements in a person's perception of and engagement with the world of visuals is implicitly accepted since it is the method by which it obtains the knowledge needed to navigate and identify the aspects of the visual world. Eye detection and tracking motions are crucial for progressing human-computer interaction and understanding human emotional states. One advantage of neural network training is that the images do not have to be high-resolution but with reasonable accuracy, which leads to cost reductions by employing a standard webcam rather than expensive hardware. Furthermore, CNNs can be trained on images with varying angles of view, decreasing the influence of head motions on the images during training. In this work, a novel approach to identifying eye gaze direction was proposed by using a new model of convolutional neural networks that are characterized by speed and accuracy in cases where determining the direction of gaze in a limited mobility location is necessary. The study focused on the small size of the eye area in the face and used the movement of the upper eyelid with the movement of the iris eye to determine the direction of gaze. Furthermore, a local database is built that captures images in real-time and enhances the lighting of images using a developed method (Split-HSV) in limited mobility locations using only a webcam. So, it is cheaper than previous methods because it does not use eye-tracking devices, and at the same time, it is relatively faster because its architecture consists of only two blocks and uses few features of the eye. Finally, the results of this study surpassed the most widely used CNN models, such as the VGG-16 model and the ResNet-50 model. The suggested model was considerably more accurate (99%) than earlier models, and it took a quarter less time to train. In future research, it intends to integrate the suggested technique with multiple classification algorithms to make more accurate decisions in a variety of situations.

Declarations

Author contribution. All authors contributed equally to the main contributor to this paper. All authors read and approved the final paper.
Funding statement. None of the authors have received any funding or grants from any institution or funding body for the research.
Conflict of interest. The authors declare no conflict of interest.
Additional information. No additional information is available for this paper.

References

[1] F. Utaminingrum, M. A. Fauzi, Y. A. Sari, R. Primaswara, and S. Adinugroho, "Eye Movement as Navigator for Disabled Person," in *Proceedings of the 2016 International Conference on Communication and Information Systems*, Dec. 2016, pp. 1–5, doi: 10.1145/3023924.3023926.

[2] R. Raj Bharath, M. E. Associate Professor, D. L. Kumar, and M. SundaramT, "Controlling Mouse and Virtual Keyboard using Eye-Tracking by Computer Vision," *J. Algebr. Stat.*, vol. 13, no. 3, pp. 3354–3368, Jul. 2022. [Online]. Available at: https://www.publishoa.com/index.php/journal/article/view/1115.

[3] M. Danelljan, G. Hager, F. S. Khan, and M. Felsberg, "Convolutional Features for Correlation Filter Based Visual Tracking," in *2015 IEEE International Conference on Computer Vision Workshop (ICCVW)*, Dec. 2015, vol. 2015-Febru, pp. 621–629, doi: 10.1109/ICCVW.2015.84.

[4] Z. Savaş, "Real-time detection and tracking of human eyes in video sequences," Middle East Technical University, p. 114, 2005. [Online]. Available at: https://open.metu.edu.tr/handle/11511/15424.

[5] A. Servais, C. Hurter, and E. J. Barbeau, "Gaze direction as a facial cue of memory retrieval state," *Front. Psychol.*, vol. 13, p. 1063228, Dec. 2022, doi: 10.3389/fpsyg.2022.1063228.

[6] S. M. Asish, A. K. Kulshreshth, and C. W. Borst, "User Identification Utilizing Minimal Eye-Gaze Features in Virtual Reality Applications," *Virtual Worlds*, vol. 1, no. 1, pp. 42–61, Sep. 2022, doi: 10.3390/virtualworlds1010004.

[7] A. H. Khaleel, T. H. Abbas, and A.-W. S. Ibrahim, "Enhancing Human-Computer Interaction: A Comprehensive Analysis of Assistive Virtual Keyboard Technologies," *Ingénierie des systèmes d Inf.*, vol. 28, no. 6, pp. 1709–1717, Dec. 2023, doi: 10.18280/isi.280630.

[8] M. Zhao *et al.*, "Gaze Speedup: Eye Gaze Assisted Gesture Typing in Virtual Reality," in *Proceedings of the 28th International Conference on Intelligent User Interfaces*, Mar. 2023, pp. 595–606, doi: 10.1145/3581641.3584072.

[9] Y. Tong, W. Lu, Y. Yu, and Y. Shen, "Application of machine learning in ophthalmic imaging modalities," *Eye Vis.*, vol. 7, no. 1, p. 22, Dec. 2020, doi: 10.1186/s40662-020-00183-6.

[10] N. M. Khassaf and S. H. Shaker, "Image Retrieval based Convolutional Neural Network," *Al-Mustansiriyah J. Sci.*, vol. 31, no. 4, pp. 43–54, Dec. 2020, doi: 10.23851/mjs.v31i4.897.

[11] A.-H. A. El-Shafie, M. Zaki, and S. E. D. Habib, "Fast CNN-Based Object Tracking Using Localization Layers and Deep Features Interpolation," in *2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC)*, Jun. 2019, pp. 1476–1481, doi: 10.1109/IWCMC.2019.8766466.

[12] M. Haqi Al-Tai, B. M. Nema, and A. Al-Sherbaz, "Deep Learning for Fake News Detection: Literature Review," *Al-Mustansiriyah J. Sci.*, vol. 34, no. 2, pp. 70–81, Jun. 2023, doi: 10.23851/mjs.v34i2.1292.

[13] Y. Wang, T. Shen, G. Yuan, J. Bian, and X. Fu, "Appearance-based gaze estimation using deep features and random forest regression," *Knowledge-Based Syst.*, vol. 110, pp. 293–301, Oct. 2016, doi: 10.1016/j.knosys.2016.07.038.

[14] J. Lemley, A. Kar, A. Drimbarean, and P. Corcoran, "Convolutional Neural Network Implementation for Eye-Gaze Estimation on Low-Quality Consumer Imaging Systems," *IEEE Trans. Consum. Electron.*, vol. 65, no. 2, pp. 179–187, May 2019, doi: 10.1109/TCE.2019.2899869.

[15] H. Huang, Y. Xu, X. Hua, W. Yan, and Y. Huang, "A crowdsourced system for robust eye tracking," *J. Vis. Commun. Image Represent.*, vol. 60, pp. 28–32, Apr. 2019, doi: 10.1016/j.jvcir.2019.01.007.

[16] I. Rakhmatulin and A. T. Duchowski, "Deep Neural Networks for Low-Cost Eye Tracking," *Procedia Comput. Sci.*, vol. 176, pp. 685–694, Jan. 2020, doi: 10.1016/j.procs.2020.09.041.

[17] M. F. Ansari, P. Kasprowski, and M. Obetkal, "Gaze Tracking Using an Unmodified Web Camera and Convolutional Neural Network," *Appl. Sci.*, vol. 11, no. 19, p. 9068, Sep. 2021, doi: 10.3390/app11199068.

[18] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, Sep. 2014, pp. 1–14. [Online]. Available at: https://arxiv.org/abs/1409.1556.

[19] G. A. Shadeed, M. A. Tawfeeq, and S. M. Mahmoud, "Automatic Medical Images Segmentation Based on Deep Learning Networks," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 870, no. 1, p. 012117, Jun. 2020, doi: 10.1088/1757-899X/870/1/012117.

[20] E. Mohamed, K. Sirlantzis, and G. Howells, "A review of visualisation-as-explanation techniques for convolutional neural networks and their evaluation," *Displays*, vol. 73, p. 102239, Jul. 2022, doi: 10.1016/j.displa.2022.102239.

[21] J. Peng *et al.*, "Residual convolutional neural network for predicting response of transarterial chemoembolization in hepatocellular carcinoma from CT imaging," *Eur. Radiol.*, vol. 30, no. 1, pp. 413–424, Jan. 2020, doi: 10.1007/s00330-019-06318-1.

[22] M. A. Wani, F. A. Bhat, S. Afzal, and A. I. Khan, *Advances in Deep Learning*, vol. 57. Singapore: Springer Singapore, p. 149, 2020, doi: 10.1007/978-981-13-6794-6.

[23] Z. Zhou, X. Yang, H. Ji, and Z. Zhu, "Improving the classification accuracy of fishes and invertebrates using residual convolutional neural networks," *ICES J. Mar. Sci.*, vol. 80, no. 5, pp. 1256–1266, Jun. 2023, doi: 10.1093/icesjms/fsad041.

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 770–778, doi: 10.1109/CVPR.2016.90.

[25] Ezenwobodo and S. Samuel, "An Assessment on the use of Mathematical Softwares in Teaching and Learning of Mathematics in Colleges of Education in South-Eastern Nigeria: A Case Study of Anambra and Enugu," *Int. J. Res. Publ. Rev.*, vol. 04, no. 01, pp. 1806–1812, 2022, doi: 10.55248/gengpi.2023.4149.

[26] S. Liu, D. Liu, G. Srivastava, D. Połap, and M. Woźniak, "Overview and methods of correlation filter algorithms in object tracking," *Complex Intell. Syst.*, vol. 7, no. 4, pp. 1895–1917, Aug. 2021, doi: 10.1007/s40747-020-00161-4.

[27] Y. Zhang, X. Tian, N. Jia, F. Wang, and L. Jiao, "Deep tracking using double-correlation filters by membership weighted decision," *Pattern Recognit. Lett.*, vol. 136, pp. 161–167, Aug. 2020, doi: 10.1016/J.PATREC.2020.06.004.

[28] R. M. Abdullah, S. A. H. Alazawi, and P. Ehkan, "SAS-HRM: Secure Authentication System for Human Resource Management," *Al-Mustansiriyah J. Sci.*, vol. 34, no. 3, pp. 64–71, Sep. 2023, doi: 10.23851/mjs.v34i3.1332.

[29] P. Viola and M. J. Jones, "Robust Real-Time Face Detection," *Int. J. Comput. Vis.*, vol. 57, no. 2, pp. 137–154, May 2004, doi: 10.1023/B:VISI.0000013087.49260.fb.

[30] N. Kamarudin *et al.*, "Implementation of Haar Cascade Classifier and Eye Aspect Ratio for Driver Drowsiness Detection Using Raspberry Pi," *Univers. J. Electr. Electron. Eng.*, vol. 6, no. 5B, pp. 67–75, Dec. 2019, doi: 10.13189/ujeee.2019.061609.

[31] M. Vitek, P. Rot, V. Štruc, and P. Peer, "A comprehensive investigation into sclera biometrics: a novel dataset and performance study," *Neural Comput. Appl.*, vol. 32, no. 24, pp. 17941–17955, Dec. 2020, doi: 10.1007/s00521-020-04782-1.

[32] P. Rot, M. Vitek, K. Grm, Ž. Emeršič, P. Peer, and V. Štruc, "Deep Sclera Segmentation and Recognition," in *Advances in Computer Vision and Pattern Recognition*, Springer, 2020, pp. 395–432, doi: 10.1007/978-3-030-27731-4_13.

[33] P. Rot, Z. Emersic, V. Struc, and P. Peer, "Deep Multi-class Eye Segmentation for Ocular Biometrics," in *2018 IEEE International Work Conference on Bioinspired Intelligence (IWOBI)*, Jul. 2018, pp. 1–8, doi: 10.1109/IWOBI.2018.8464133.

[34] Z. Ali, U. Park, J. Nang, J. S. Park, T. Hong, and S. Park, "Periocular Recognition Using uMLBP and Attribute Features," *KSII Trans. Internet Inf. Syst.*, vol. 11, no. 12, pp. 6133–6151, Dec. 2017, doi: 10.3837/tiis.2017.12.024.

[35] M. Grandini, E. Bagli, and G. Visani, "Metrics for Multi-Class Classification: an Overview," *arXiv*, pp. 1–17, Aug. 2020. [Online]. Available: https://arxiv.org/abs/2008.05756v1.

[36] A. Haque, I. Sutradhar, M. Rahman, M. Hasan, and M. Sarker, "Convolutional Nets for Diabetic Retinopathy Screening in Bangladeshi Patients," *arX*, pp. 1–8, Jul. 2021. [Online]. Available at: https://arxiv.org/abs/2108.04358v1.

[37] D. S. Kermany *et al.*, "Identifying Medical Diagnoses and Treatable Diseases by Image-Based Deep Learning," *Cell*, vol. 172, no. 5, pp. 1122-1131.e9, Feb. 2018, doi: 10.1016/j.cell.2018.02.010.

[38] V. Silaparasetty, *Deep Learning Projects Using TensorFlow 2*. Berkeley, CA: Apress, p. 421, 2020, doi: 10.1007/978-1-4842-5802-6.

[39] M. Hasan, L. Islam, I. Jahan, S. M. Meem, and R. M. Rahman, "Natural Language Processing and Sentiment Analysis on Bangla Social Media Comments on Russia–Ukraine War Using Transformers," *Vietnam J. Comput. Sci.*, vol. 10, no. 03, pp. 329–356, Aug. 2023, doi: 10.1142/S2196888823500021.