Scientia Iranica D (2023) 30(5), 1731–1742

Sharif University of Technology
Scientia Iranica
Transactions D: Computer Science & Engineering and Electrical Engineering
http://scientiairanica.sharif.edu

Deep learning-based convolutional neural network structured new image classification approach for eye disease identification

I. Topaloglu*

Department of Electrical Electronic Engineering, Faculty of Engineering, Cankiri Karatekin University, 18100, Cankiri, Turkey.

Received 5 April 2021; received in revised form 16 November 2021; accepted 24 January 2022

KEYWORDS
Deep learning;
Neural networks;
Python;
Image processing;
Eye disease;
Care model.

Abstract. This study implemented a new image classification method approach based on a convolutional artificial neural network using deep learning techniques. Sample application was carried out using the proposed method for the diagnosis of diabetic retinopathy. A problem that may arise here is how to collect information about the blood vessels and identify any abnormal patterns from the rest of the phonoscopic images, and then assess the degree of retinopathy. To solve this problem, the current research proposed a developed methodology and the algorithmic structure of this new approach. An approach called "care model" was utilized in this study, which differs from the classical Convolutional Neural Network (CNN) structure. The care approach is based on the idea of generating a better solution by incorporating new data obtained through rescaling the available data based on the total number of pixels before creating the average data pool. This allows for the continuation of CNN processes with enhanced accuracy and performance. In this approach, all data sets are multiplied by the number of elements by the number of epoch times eight tensors. The purposed care model included VGG19 image classification model and developed a mathematical model. The pre-trained model and all the image dataset were taken from the kaggle and keras to be implemented as the case study. The proposed model provided the training accuracy of 87%, testing accuracy of 88%, precision of 93%, and recall of 83%.

1. Introduction

Deep learning health practices constitute a multidisciplinary field of study. Although computer image classification is now inevitably included in our daily lives, it requires meticulous interpretation of software with health applications by healthcare professionals [1,2]. Today, with advances in technology, data processing speed, and data processing capacity [3,4], disease diagnosis with image processing has become a hot topic among the academic community [5] and private sector researchers [6,7].

1.1. Motivation

Diabetic retinopathy accounts for 12% of all new cases of blindness reported each year worldwide. For this reason, medical practitioners are constantly trying to diagnose this disease using several methods. It should be noted that this disease often develops without

*. *Tel.: +903762189532*
E-mail address: itopaloglu@karatekin.edu.tr (I. Topaloglu)

doi: 10.24200/sci.2022.58049.5537

showing any symptoms, with ruptured vessels and blood accumulation behind the eyes, even if patients with diabetes sleep at night.

Diabetic retinopathy can be detected by several methods such as visual acuity test, pupil dilation, ophthalmoscopy, fundus fluorescein angiography, retinal vessel analysis, and optical coherence tomography.

Each of these diseases is diagnosed using different methods, which involve analyzing the information obtained from patient follow-up and time-consuming studies by a doctor. In this study, a deep learning-based approach was developed where instant retina pictures of the patient's eye can be uploaded to diagnose the disease and assess the risk. The diagnosis can be made using only a few snapshots, without the need for extensive time and effort. The motivation behind this study has already been explained above.

1.2. Literature review

In the field of bio-medicine, image processing is typically used in the literature [8,9] for diagnosis of many serious diseases such as brain tumour [10,11], cancer [12,13], microorganism detection [14,15], eye diseases [16,17], MRI imaging [18,19], lung, liver, and other diseases. Deep learning focuses on the computer programs that simulate human brain functions. The history of deep learning dates back to 1943 when McCulloch and Pitts used mathematics and neural reasoning to mimic thinking. It primarily focuses on the creation of an algorithm-based computational model [20]. In 1958, a supervised algorithm for the detection of teaching patterns was developed based on a two-layer Rosenblatt neural computer network [21]. Ivakhnenko and Lapa employed some models with activation features of complex equations in 1965 to develop deep learning algorithms [22]. In 1988, Fukushima suggested neocognitron hierarchical and multi-layered nerve network to identify its composition and other patterns [23]. The Cresceptron process, which performs dimensional object recognition, was automated by Weng, Cohen, and Herniou in 1992 for three mixed scenes [24]. Later in 1995, Vapnik utilized Cortes and vector support networks to identify two related data categories [25]. In 1997, Hochreiter and Schmidhuber developed a paradigm for learning called Long Short-Term Memory (LSTM) to save knowledge for a long time with repeating back growth [26,27]. Deep learning, as one of the major research areas, has drawn considerable attention in the last decades. In a 2012 study conducted by Google's research team, 16,000 processors and over a billion connections were used. The algorithms demonstrated a level of efficiency in detecting artificial patterns that rivaled human performance [28]. By adding 120 million R-CNN parameters to automatically tag users in images, facial recognition utilized specific deep learning technologies called DeepFace in 2014 [29]. Deep learning is a type of neural artificial network functioning based on the biological nervous system information processing techniques that uses proven algorithms. Computers must then determine the meaning and model of each result.

Machine learning, along with Google Brain science team research, is one of the most popular approaches to deep learning and has been developed by engineers [30]. In addition, TensorFlow is one of the most popular tools for deep learning. TensorFlow, an open-source library for artificial intelligence and machine learning, uses data flow graphs to construct multi-layered and large-scale artificial neural networks. It is commonly used in applications such as in vision, discovery, classification, understanding, and prediction [31]. This experiment utilizes TensorFlow, particularly for deep learning.

With the rapid penetration of deep learning into our lives coupled with developing technology, processing of big data and development of its models have begun. These models generally consist of deep artificial neural networks, Convolutional Neural Networks (CNN), artificial neural networks, and other models. The open-source image processing models that utilize these networks to their advantage include GoogleNet, VGG16, VGG19, ResNet, and Inception [32,33]. However, a challenge frequently encountered in image processing studies. The fact that each of the images obtained in separate pixels, different drawing angles, the use of special lenses in some cases, and varying resolutions can all lead to a reduction in the quality of the work. The aforementioned situations can introduce noise in image datasets during analysis. Therefore, it is necessary to subject the image dataset to a series of processes before using them. These operations are carried out to ensure that image datasets are trainable with certain criteria, thereby directly increasing the quality of the work performed [34,35].

Deep learning and image processing models have been actively used in the diagnosis of eye diseases over the past ten years. Disease detection has become possible by using image datasets that can provide certain features with pre-trained models [36,37]. In this study, the sample application focuses on a specific eye disease. Image processing and classification models typically consist of multiple layers, including convolution layers, direct connected layers, maximum pool layers, and SoftMax layers in the case of CNN applications. To solve a classification problem, a loss function, pre-trained classification model, pre-trained label model, general classification model, and ready image datasets are used [38,39].

The weakness of classical CNN structures is that their data is often not pre-processed using effective methods, and many unrelated data points are involved in the processing, leading to longer processing times

[40,41]. However, in practice, processing is directly applied to the picture without any intermediate processing. In this case, convenience is ensured. In fact, the care model used in this research operates by prioritizing precision and accuracy in diagnosis and classification. This gives better results than most classical models in the literature.

1.3. Contribution

This study offers a new approach called the care model to overcome the limitations of the above classical structure with a new perspective. This study realized a new image classification approach for identifying eye diseases through a deep learning-based convolutional neural network structure. In practice, pre-trained models, configurations, weight functions, lost functions, ReLU functions, pre-trained model data, and pre-trained label data are initially defined in the Python environment. Subsequently, manual image data are read and image categorization is performed. The adopted CNN algorithm trains data, extracts information, and creates new data models. Unlike other applications in the literature, the picture pixels are retrained and pooled in this study, taking into account the total number of pixels by means of a model called care model in CNN structure. The best performing result is selected. The results are evaluated and classified by the decision-making function.

2. Methodology

The classical CNN method involves several steps to obtain the best solution through a single analysis, as shown in Figure 1, which depicts the CNN scheme for determining whether an eye is healthy or sick. In this study, the methodology is explained in an easy-to-understand manner. The input image dataset requires some pre-processing, after which the image data is trained for the neural network. Then, the decision function classifies the image as healthy or sick. The newly produced trained image data is compared with pre-trained image data, and the classification function selects the best solution for each analysis.

In this study, the input shape image data ($512 \times 512 \times 3$ in size) was processed by the VGG19 model. This image processing model creates 25,782,356 image parameters to train. Although the VGG19 model uses $224 \times 224 \times 3$ data input type, a data type of $512 \times 512 \times 3$ was selected in this study to increase the number of image parameters for training. The main purpose here is to enhance the quality of the input image as well as the prediction to be made after this training. The dimensions of the data shape were converted into $14 \times 14 \times 2048$ through the image processing model with batch process. In the classical CNN, image data are trained using convolutional and dropout operations through multiple layers. The Care model is applied to these newly trained image datasets. As a result, a $14 \times 14 \times 2048$ output data shape was converted into 1D 2048 and then, into 128. This model has a total of 25,782,356 parameters, of which 22,308,983 are trainable and 3,473,373 are non-trainable.

2.1. Image dataset

The data was collected from a Kaggle web site [42] and is an atypical collection from Kaggle. The mentioned data had already been cleaned in most Kaggle datasets, and very additional cleaning was achieved by the data scientist. However, this was not the case for the dataset used in this study. All the images were captured by cameras of varying sizes and taken by humans. This data set holds significant noises in the pre-processing stage; therefore, many pre-processing phases are required for all images to be in a useful format for model training. The training data contains 45,231 images that are expanded during pre-processing. The applied datasets are available in the GitHub folder [43]. The used image dataset, newly created trained dataset, and sample work were all uploaded to this folder.

Image dataset is a large collection of high-resolution retina photos taken in different imaging conditions, and there is a left or right field for each subject. Pictures are identified with a subject ID and a left or right arrow. The photographs in the collection were taken from different camera models and types, which in turn impact how the left and right sides visually appeared. Some photos depict the retina as it appears anatomically, while others show the retina as it appears through a microscope lens. The dataset contains both photos and labels that may contain noise. The photos may be out of focus, underexposed, or overexposed, among other possibilities.

2.2. Image pre-processing

In the present study, image pre-processing is vital to obtain the best solution. Pre-processing involves four steps in the proposed method. Initially, all images are resized into the 512×512 size for the input shape and then, a column of non-black images is created. Next, the column is tested to determine if the image is pitch black or not. Secondly, for both training and testing, all images are rotated by a specific degree and then converted into arrays. These arrays are then merged into a new array based on the image column matching the image file name. The process of cropping and resizing images provides extra detailed images for the network. Images with no color space or completely black images are finally removed from the dataset. Rotating pictures give additional trainable data that has previously been recognised to be ill.

Figure 2 clearly shows the effects of pre-processing

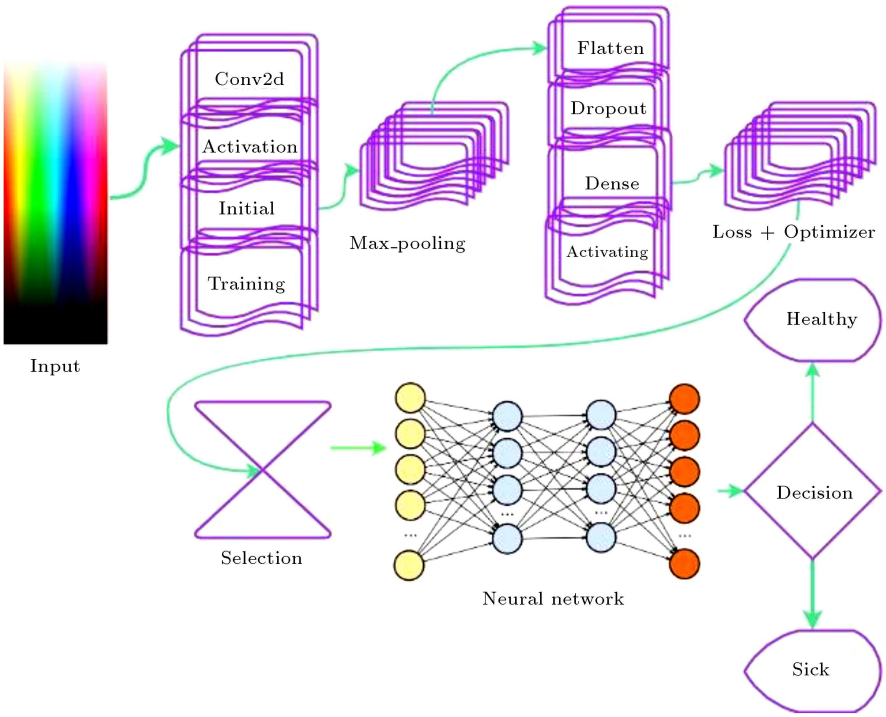


Figure 1. Convolutional neural network scheme for healthy or sick decision.

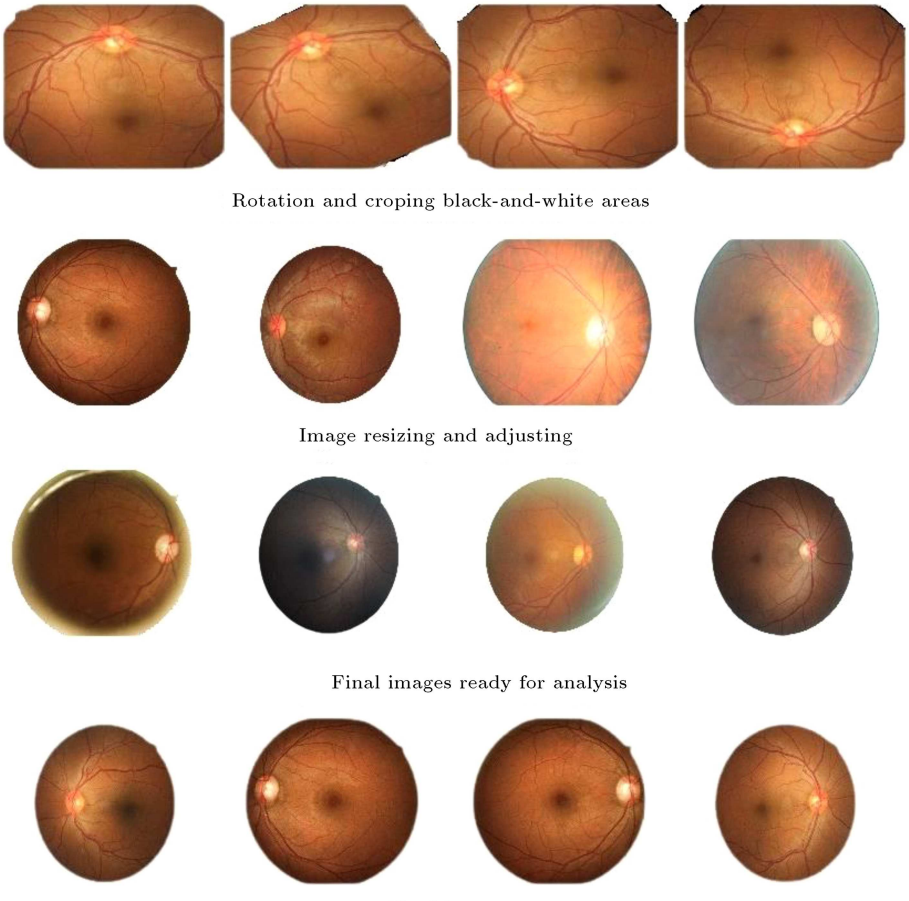


Figure 2. Image pre-processing steps.

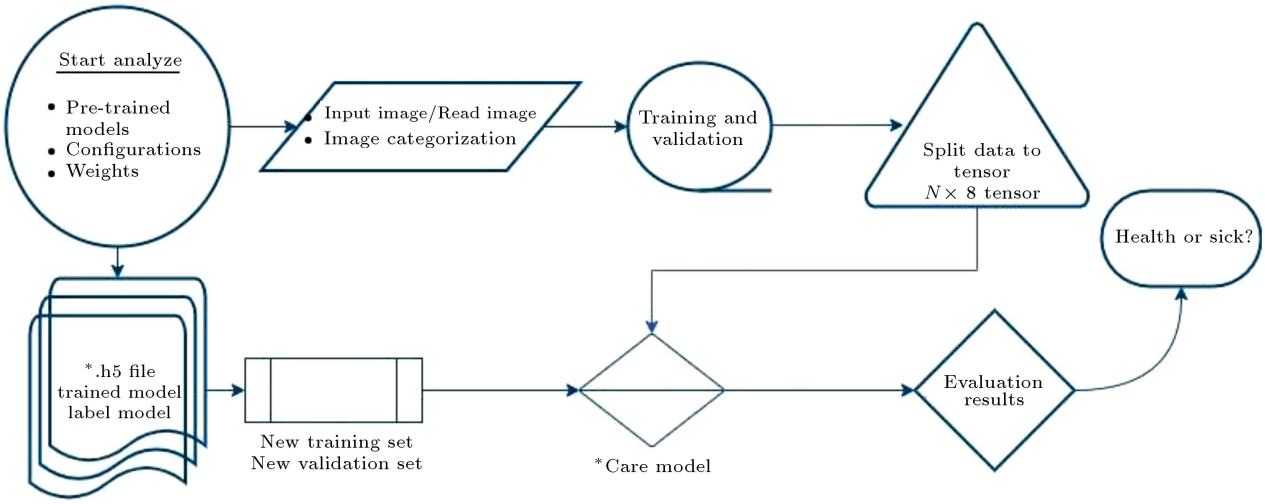


Figure 3. Deep learning model algorithm.

operations on images. The pre-processing steps include: 1) Rotating and cropping unnecessary black and white areas of the images and 2) resizing and adjusting images for analysis.

3. Deep learning model

To extract the best features from an input image in a deep learning model, we first take a 512 × 512 image and multiply it with a feature detector with a span of 1. Then, we apply the linear activation function to obtain the optimal features. To reduce the size of the image matrix, we use a 3 × 3 feature detector and multiply it with a large 512 × 512 input image matrix. Next, we apply max pooling by placing a 14 × 14 pixel box in the upper left corner of the image. We find the maximum value within that box, and then reduce the image resolution by that value. We then move the box to the right by two steps and repeat this process to obtain 32 pooled feature maps. To achieve high accuracy on the train and test sets, we added three convolutional layers. The first layer has 32 feature detectors with a 3 × 3 matrix. The second layer has 64 feature detectors with a 3 × 3 matrix, and the third layer has 128 feature detectors with a 3 × 3 matrix. We also applied max pooling to each convolutional layer. Next, we merge each feature map into a column by taking the numbers from the merged feature map row by row and placing them in a long column. This creates a large input vector for the artificial neural network.

We create two fully connected layers with dense embedding, each with an output dimension of 128 and a linear activation function. To calculate the loss, we use classification cross entropy and apply the *softmax* activation function to calculate the error in the output layer. We also use the *Adam* optimizer to propagate back through the network and adjust it to optimize performance. We train the *starter, pre-intermediate, intermediate, upper-intermediate*, and *serious level* datasets using a target size of 512×512 and a batch size of 10, and the test set using a target size of 512 × 512 and a batch size of 2. We train the network for 50 epochs and achieve a train accuracy of 87%, a test accuracy of 88%, a precision of 93%, and a recall of 83% on the test set. Figure 3 shows the deep learning model algorithm, and the main difference is the use of the Care model approach, which is explained in detail in the related caption.

3.1. Mathematical architecture of artificial neural network and Care model approach

The artificial neural network approach is generally expressed using Eqs. (1)–(3). In addition, this study elaborates on the care model approach by using Eqs. (4)–(8). As schematically seen in Figure 3, we have:

$$F(x) = f_n(f_{n-1}(...(f_1(x))), \tag{1}$$

$$h^k(x,y) = \sum_{u=-j}^{m} \sum_{v=-k}^{n} \sum_{w=-l}^{w} W_k \{(u,v,w),(x-u,y-v,z-w\}, \tag{2}$$

$$r(x) = \max(0, \max) \leftarrow \text{maximum pooling}. \tag{3}$$

In Eq. (2), x is the input image, n the number of hidden layers, f_i the ith activation function, and $F(x)$ the output of network in Eq. (1). In addition, x, y, z denote the locations of the pixels, h^k stands for the convolution filter, W_k shows the weight of the kth kernel, and m, n, and w represent the height, width, and depth, respectively. In Eq. (3), $r(x)$ is the ReLU function of maximum pooling.

$$weights = \begin{bmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{bmatrix}_{64\times 64}, \begin{bmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{bmatrix}_{32\times 32},$$

$$\begin{bmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{bmatrix}_{16\times16}, \begin{bmatrix} 1 & \cdots & 1 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 1 \end{bmatrix}_{8\times8}, [1]_{1\times1}, \tag{4}$$

$$M = weights * h^k(x, y), \tag{5}$$

$$G = GAP2D(M), \tag{6}$$

$$GAP_{new}(M) = G * weights, \tag{7}$$

$$\delta = \frac{\sum_0^n \mathrm{Img}[x_n]}{\sum_0^t \mathrm{Img}[x_t]} \leftrightarrow \forall * \{G, GAP_{new}\}. \tag{8}$$

Eq. (4) shows the weights, in which M is the map of newly produced filtered image data, G the global average pool newly created using M, GAP_{new} the newly extended global average pool, and δ the coefficient of new pixel size based on the total pixel size.

Unlike the classical CNN structure, the Care model approach is presented mathematically. This study discusses the sample application obtained from this model and its algorithm. In the Care model approach, trainable data is created by filtering the input image data with the filter defined in the CNN structure. This filtering is performed at the input stage, resulting in new trainable data that corresponds to the total number of pixels in the image.

3.2. Care model approach

The chart of the Care model approach is presented in Figure 4. A data repository is generated by training $512 \times 512 \times 3$ input image data in a five-layer structure, using both the initial and trained data sets in five iterations. Here, a tensor of $N \times 8$ for each element is created in the background. After filtering and training processes, image data pooling and retraining processes are implemented. A new trained image repository is created and filtered considering all the pixel values of the image data in the Care model approach discussed after this stage. Consequently, after finishing the predetermined loss function and optimization process, numerical estimation and classification process are initiated using the classification function at the output.

3.3. Eye Disease Identifying Software (EDIS)

Eye Disease Identifying Software (EDIS) is developed with PyQt5 and Qt designer in a Python environment. The main libraries used in the background are Keras, TensorFlow, Numpy, Pandas, and PyQt5 tools. EDIS is used to analyze images based on the CNN Care model approach. Although the designed program is developed to diagnose various eye diseases using deep learning, the sample application in the first stage focuses on the detection of diabetic retinopathy disease. Although the interface design may not be very efficient in the Python environment, the program works well as an open-source platform with TensorFlow as the backend and utilizes the functionalities provided by Keras to achieve great success. In the designed interface, under the menu item 'file', there are open, close, save, print, and exit parameters. The menu item, namely eye health, recommends visiting ophthalmologists in the user area based on a given map, if the disease is detected in the image analyzed and revealed to the user. A Care model-based deep learning analysis is performed in Analyse image. The analysis reports of the images are accessible in the "Results" section. For now, only two analysis reports are taken into consideration. These

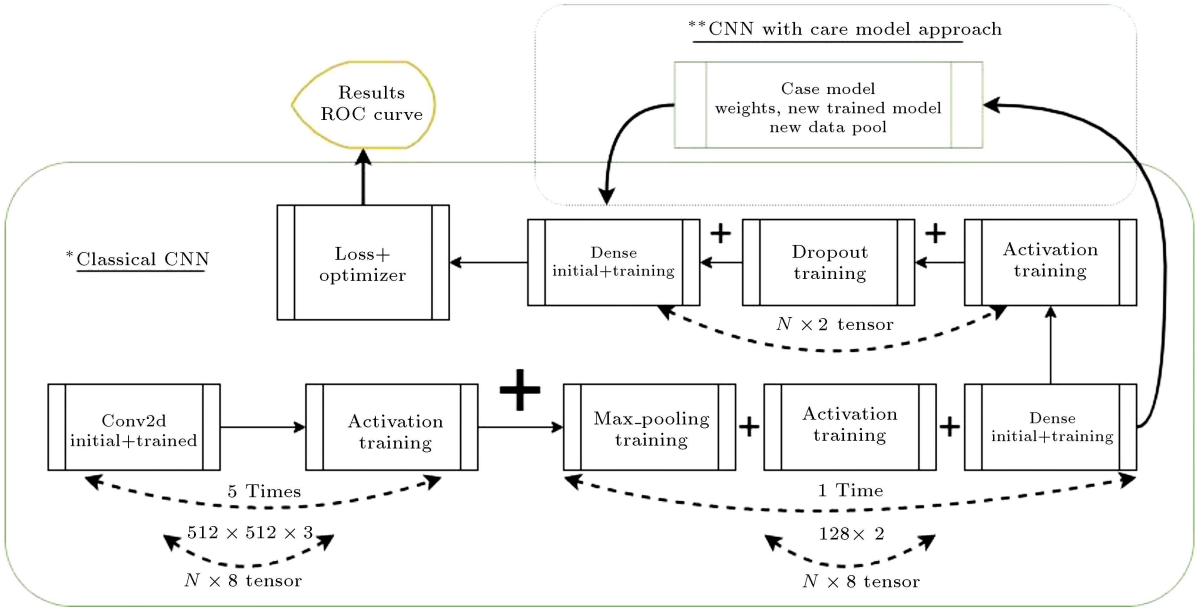
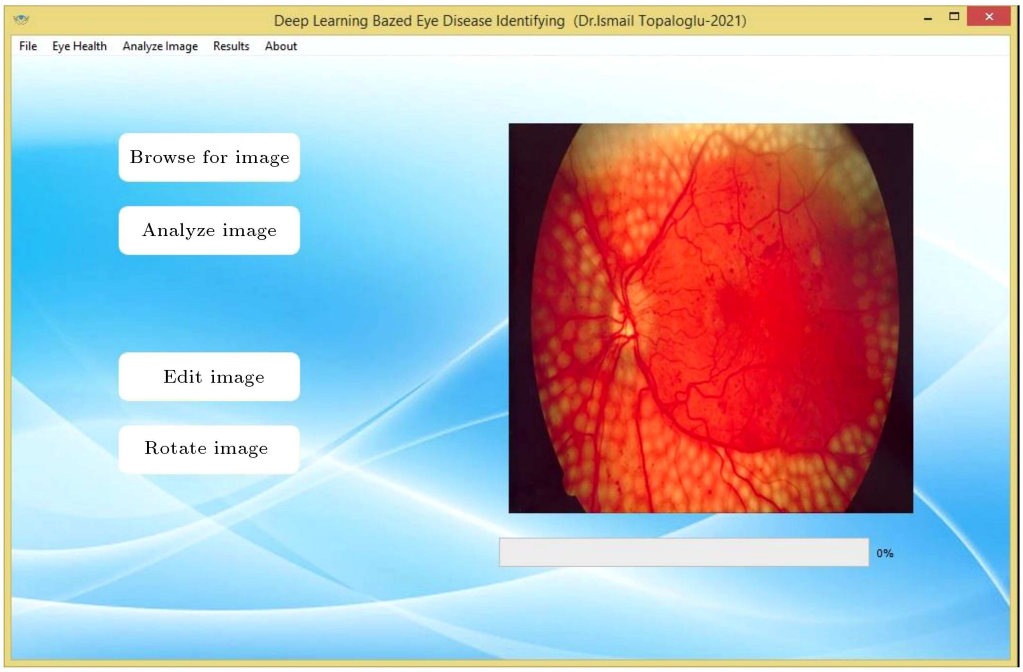
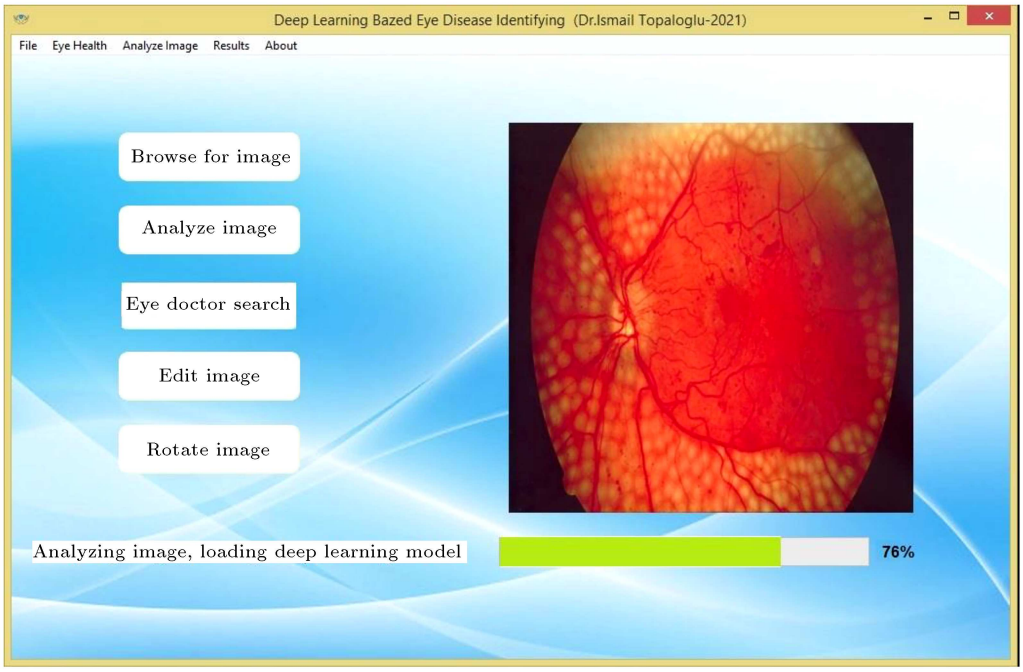


Figure 4. Care model approach scheme.



(a)



(b)

Figure 5. Developed eye disease identifying software: a) image loading and editing view and b) Image analyzing view.

are generated by a function that determines the disease state and disease prediction numerical ratio and disease level as a result of the analysis. The interface of the developed program can be seen in Figure 5. If the disease is detected during the analysis process, the software activates the ophthalmologist search button.

The analysis results are currently divided into two groups for the final report. The first group estimates whether there is a disease in the analyzed picture and determines the disease level. The second group, which is the subject of this study, presents the visualization of the pictures analyzed on the basis of the Care model. The first case can be seen in Figures 6 and 7, respectively.

The image analysis report of the initial state consists of image name, disease class, and analysis of

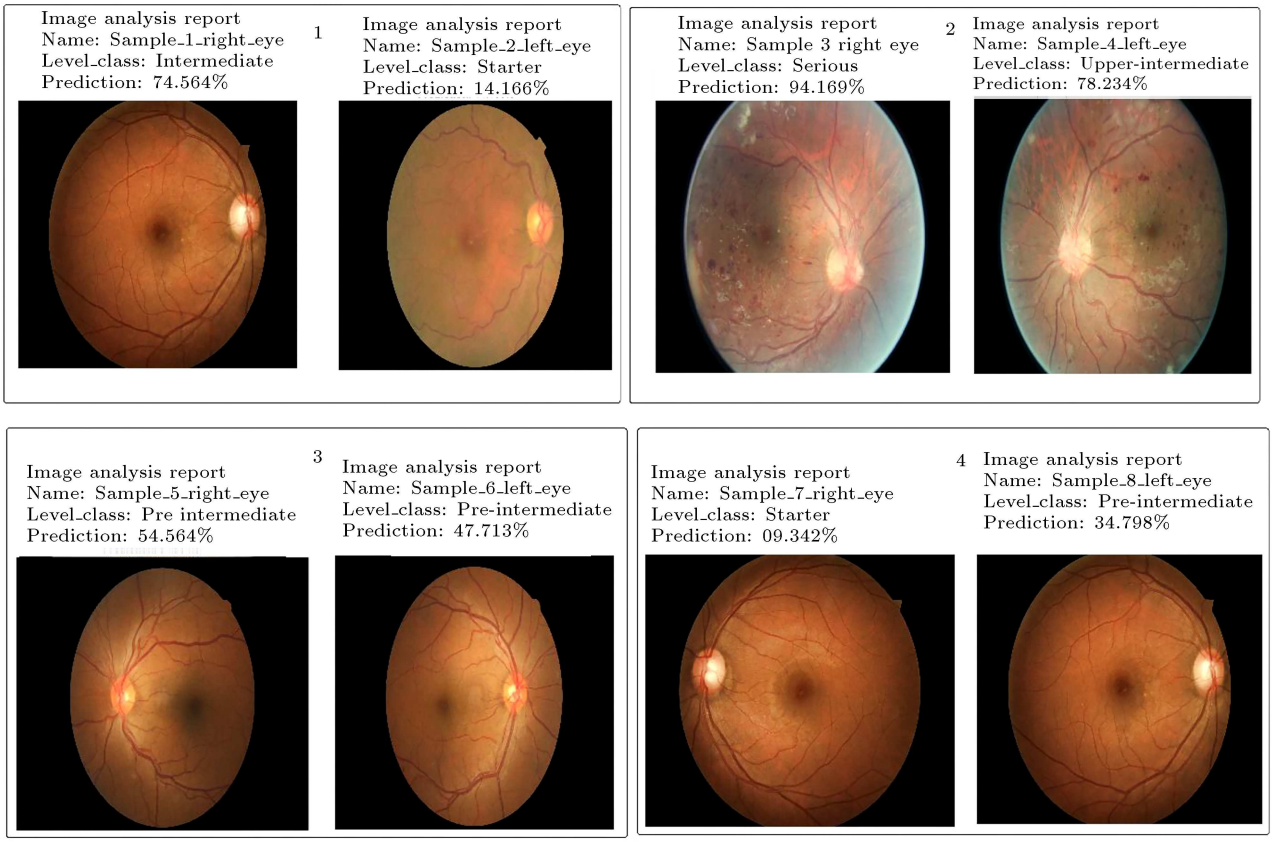


Figure 6. Analyzing results classification and analysis report.

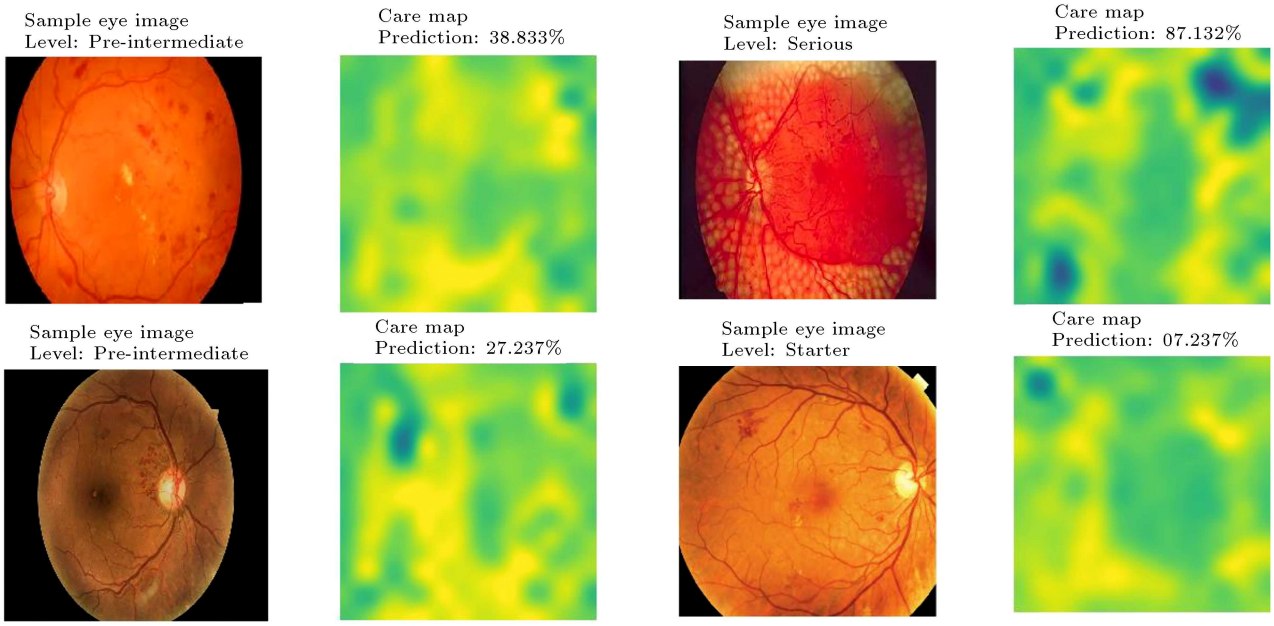


Figure 7. Care model implementation map.

estimated numerical data, as given in Figure 5. In the second case report, the analyzed image shows the disease class, numerical data of the probability of the disease, and Care model data. The most important characteristic of the Care model approach analysis is that it determines how the processes performed are viewed by the computer.

Expressing the algorithm that gives the solution of a problem or gives the solution of an applied method, not in a specific programming language but in almost

everyday speech is called pseudo code. In this study, the pseudo code is given in Figure 8. Of note, each subprocess is outlined in the code and its mathematical operations are not entered. In general, an attempt is made here to elaborate on the logic of the program functioning.

4. Results and discussion

The results are considered as the evaluation metrics, hardware, memory, and time, which are generally accepted and defined in the literature [19]. Evaluation metrics consist of three parts, namely the precision, recall, and accuracy. The model accuracy was investigated for both testing and training phases. Here, pre-processing is vital step in the neural network. In this way, black and colorless areas in the picture are cropped, which reduces the processing load and allows the network to focus on the relevant areas in the image. 13% of the total elements remained untrained. Figure 9 depicts the training curve of the proposed model. The total loss is less than 1, almost in every iteration. Figure 9 shows the validated and trained data loss in every epoch.

4.1. Evaluation metrics

The proposed model provides the train accuracy of 88%, test accuracy of 87%, precision of 93%, F_1 of 0.83, and recall 83%:

$$P = \frac{TP}{TP + FP}, \qquad R = \frac{TP}{TP + FN},$$

$$ACC = \frac{TP}{TP + FP + TN + FN},$$

```
Import Libraries
Model = VGG19 // image processing model
Backend = tensorflow // mathematical functions background
X= batch normalization X // to get 14x14x2048 data shape
X= get dropout and crop X// eye distribution and unnecessary data reduction
X=conv2d X // At least 5 times in every iteration-Split data into training and
validation
Care Model;
X=multiply X // Multiplying with new filtered pixels
X= Global average pooling X // Creating a new data pool with the data obtained by
multiplying the filtered pixels
X=Rescale X // Rescaling data
X=dropout and dense X // to avoid overfitting and necessary connections
While true
X_output = = output image with disease classification
Print output image
Print care model affect
Print output disease classification
Break
```

Figure 8. Pseudo-code of the proposed model.

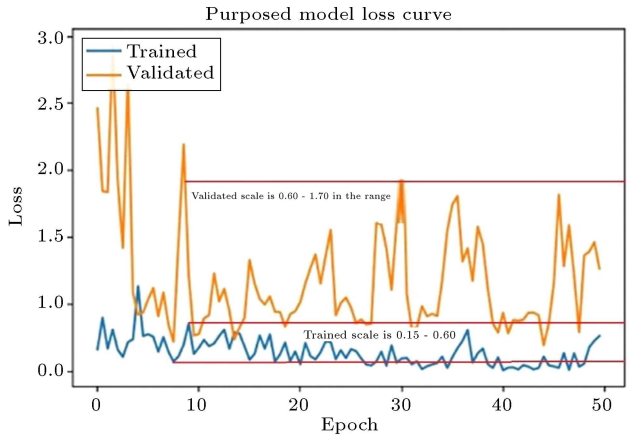


Figure 9. Training curve of the proposed model.

$$F_1 = 2 * \frac{P * R}{P + R}. \tag{9}$$

In the above equation, P stands for the precision, R the recall, ACC the accuracy, F_1 the harmonic mean of precision and recall values, TP the true positives, TN the true negatives, FP the false positives, and FN the false negatives.

Accuracy is a metric widely used to measure the success of a model; precision is a value that shows how many of the values that we estimate as positive are actually positive; and sensitivity (recall) is a metric that shows how many operations required for prediction are positive. The reason for using a harmonic average instead of a simple average is that it takes into account extreme cases and avoids ignoring them. If a simple average calculation was used, it could lead to a misleading F_1 score of 0.5 for a model with a precision value of 1 and a recall value of 0. This is why a harmonic average is used instead, to avoid ignoring extreme cases. The F_1 score value is important because it helps us avoid choosing an incorrect model in datasets that are not evenly distributed. It is also crucial to have a measurement metric that considers not only false negative or false positive but also all error costs, which is why F_1 score is highly valued.

Table 1 shows the evaluation metrics of the proposed method. In this table, the Care model yielded very good values at the output, i.e., four input layers and zero output layers. Figure 10 presents the confusion matrix of the proposed model.

4.2. Hardware

Nvidia Tesla K80 CUDA Cores Graphic Cards (GPU) were utilized in this research, which provided the required calculations and implementation of all processes with 4992 CUDA cores and 2× GK120 GPUs.

4.3. Memory

GPU has a memory of 24 GB, while the test platform has approximately 77 GB of free memory available for

Table 1. Evaluation metrics of all the models.

Class number/layer	Train accuracy	Test accuracy	Precision	F_1	Recall
0	88	87	93	0.83	83
1	56	44	36	0.27	39
2	27	31	29	0.23	29
3	22	17	16	0.13	23
4	1	1	1	0.01	1

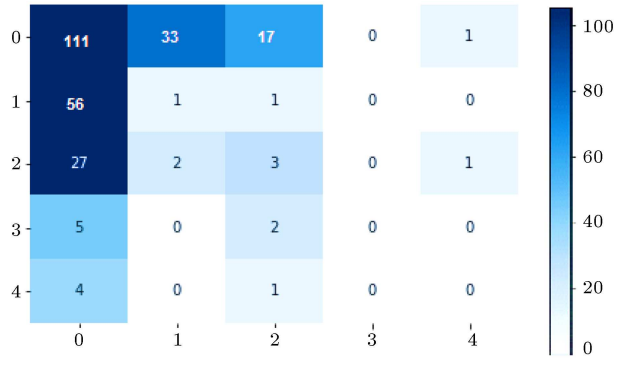


Figure 10. Confusion matrix of the proposed model.

implementation in the study. Clearly, more computational capacity and a larger free space are required.

4.4. Time

The total processing time lasted 2349.8 seconds, being equal to approximately 39.16 minutes. Some modifications can be done on both hardware and data side to shorten the processing time. On the data side, using datasets that contain only eye images with minimal noise and a stronger computation capacity on the hardware side certainly helps speed up the analysis.

5. Conclusion

The proposed approach in this study is a new image classification method that utilizes a convolutional artificial neural network with deep learning architecture to identify and diagnose eye diseases. The Care model approach was implemented using Eye Disease Identifying Software (EDIS) in Python. The developed model approach was explained mathematically and realized in the study, which focused on the case of diabetic retinopathy disease image. The image data for this disease was obtained from the Kaggle website. The newly developed model approach provided better results, which were discussed in depth and presented in the study. The proposed model achieved a training accuracy of 88%, testing accuracy of 87%, precision of 93%, F_1 score of 0.83, and recall of 83%.

5.1. Future outlook

The applied model can be developed by using high-resolution images in image datasets, improving the sampling process, pre-processing the data in pre-trained models, preventing data leakage between training and validation sets, and normalizing the age of the images.

References

1. Oksuz, I., Clough, J.R., Ruijsink, B., et al. "Deep learning-based detection and correction of cardiac MR motion artefacts during reconstruction for high-quality segmentation", *IEEE Transactions on Medical Imaging*, **39**(12), pp. 4001–4010 (2020).
2. Haan, K., Rivenson, Y., Wu, Y., et al. "Deep-learning-based image reconstruction and enhancement in optical microscopy", *Proceedings of the IEEE*, **108**(1), pp. 30–50 (2020).
3. Olefir, I., Tzoumas, S., Restivo, C., et al. "Deep learning-based spectral unmixing for optoacoustic imaging of tissue oxygen saturation", *IEEE Transactions on Medical Imaging*, **39**(11), pp. 3643–3654 (2020).
4. Liu, J., Pan, Y., Min, L., et al. "Applications of deep learning to MRI images: A survey", *Big Data Mining and Analytics*, **1**(1), pp. 1–18 (2018).
5. Li, L.F., Wang, X., Hu, W.J., et al. "Deep learning in skin disease image recognition: A review", *IEEE Access*, **8**, pp. 208264–208280 (2020).
6. Wang, J., Bai, Y., and Xia, B. "Simultaneous diagnosis of severity and features of diabetic retinopathy in fundus photography using deep learning", *IEEE Journal of Biomedical and Health Informatics*, **24**(12), pp. 3397–3407 (2020).
7. Seo, H., Bassenne, M., and Xing, L. "Closing the gap between deep neural network modeling and biomedical decision-making metrics in segmentation via adaptive loss functions", *IEEE Transactions on Medical Imaging*, **40**(2), pp. 585–593 (2021).
8. Schlemper, J., Caballero, J., Hajnal, J.V., et al. "A deep cascade of convolutional neural networks for dynamic MR image reconstruction", *IEEE Transactions on Medical Imaging*, **37**(2), pp. 491–503 (2018).
9. Xue, Y., Li, N., Wei, N., et al. "Deep learning-based earlier detection of esophageal cancer using improved empirical wavelet transform from endoscopic image", *IEEE Access*, **8**, pp. 123765–123772 (2020).
10. Noreen, N., Palaniappan, S., Qayyum, A., et al. "A deep learning model based on concatenation approach

for the diagnosis of brain tumor", *IEEE Access*, **8**, pp. 55135–55144 (2020).

11. Wei, L., Ding, K., and Hu, H. "Automatic skin cancer detection in dermoscopy images based on ensemble lightweight deep learning network", *IEEE Access*, **8**, pp. 99633–99647 (2020).
12. Qiao, L., Zhu, Y., and Zhou, H. "Diabetic retinopathy detection using prognosis of microaneurysm and early diagnosis system for non-proliferative diabetic retinopathy based on deep learning algorithms", *IEEE Access*, **8**, pp. 104292–104302 (2020).
13. Li, X., Hu, X., Yu, L., et al. "CANet: Cross-disease attention network for joint diabetic retinopathy and diabetic macular edema grading", *IEEE Transactions on Medical Imaging*, **39**(5), pp. 1483–1493 (2020).
14. Zhu, S., Liu, H., Du, R., et al. "Tortuosity of retinal main and branching arterioles, venules in patients with type 2 diabetes and diabetic retinopathy in China", *IEEE Access*, **8**, pp. 6201–6208 (2020).
15. Khansari, M.M. "Automated deformation-based analysis of 3D optical coherence tomography in diabetic retinopathy", *IEEE Transactions on Medical Imaging*, **39**(1), pp. 236–245 (2020).
16. Zeng, X., Chen, H., Luo, Y., et al. "Automated diabetic retinopathy detection based on binocular siamese-like convolutional neural network", *IEEE Access*, **7**, pp. 30744–30753 (2019).
17. Araújo, T., Aresta, G., Mendonça, L., et al. "Data augmentation for improving proliferative diabetic retinopathy detection in eye fundus images", *IEEE Access*, **8**, pp. 182462–182474 (2020).
18. Qummar, S., Khan, F.G., Shah, S., et al. "A deep learning ensemble approach for diabetic retinopathy detection", *IEEE Access*, **7**, pp. 150530–150539 (2019).
19. Sun, Y. "The neural network of one-dimensional convolution-an example of the diagnosis of diabetic retinopathy", *IEEE Access*, **7**, pp. 69657–69666 (2019).
20. McCulloch, W.S. and Pitts, W. "A logical calculus of the ideas immanent in nervous activity", *The Bulletin of Mathematical Biophysics*, **5**(4), pp. 115–133 (1943).
21. Haykin, S.S., *Neural Networks and Learning Machines-3*, Pearson Upper Saddle River (2009).
22. Ivakhnenko, A.G. and. Lapa, V.G. "Cybernetic predicting devices", *CCM Information Corporation* (1965).
23. Fukushima, K. "Neocognitron: A hierarchical neural network capable of visual pattern recognition", *Neural Networks*, **1**(2), pp. 119–130 (1988).
24. Weng, J., Cohen, P., and Herniou, M. "Camera calibration with distortion models and accuracy evaluation", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **8**(10), pp. 965–980 (1992).
25. Cortes, C. and Vapnik, V. "Support-vector networks", *Machine Learning*, **20**(3), pp. 273–297 (1995).
26. Hochreiter, S. and Schmidhuber, J. "Long short-term memory", *Neural Computation*, **9**(8), pp. 735–1780 (1997).
27. Hinton, G.E., Srivastava, N., Krizhevsky, A., et al. "Improving neural networks by preventing co-adaptation of feature detectors", *Arxiv.net*, pp. 1–18 (2016).
28. Lohr, S., *The Age of Big Data*, New York Times (2012).
29. Taigman, Y., Yang, M., Ranzato, M., et al. "Deepface: Closing the gap to human-level performance in face verification", *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1701–1708 (2014).
30. Lee, K. and Son, M. "Deepspotcloud: Leveraging cross-region GPU spot instances for deep learning", *2017 IEEE 10th International Conference on Cloud Computing (CLOUD)*, pp. 98–105 (2017).
31. Tokui, S., Oono, K., Hido, S., et al. "Chainer: A next-generation open source framework for deep learning, in Proceedings of workshop on machine learning systems", *The Twenty-Ninth Annual Conference on Neural Information Processing Systems*, **5**, pp. 1–6 (2015).
32. Dai, M., Xiao, G., Fiondella, L., et al. "Deep learning-enabled resolution-enhancement in mini- and regular microscopy for biomedical imaging", *Sens Actuators A Phys.*, **1**(331), 112928 (2021).
33. Yahia, S., Said, S., and Zaied, M. "Wavelet extreme learning machine and deep learning for data classification", *Neurocomputing*, **470**, pp. 280–289 (2021).
34. Valarmathi, S. and Vijayabhanu, R. "A survey on diabetic retinopathy disease detection and classification using deep learning techniques", *2021 Seventh International conference on Bio Signals, Images, and Instrumentation (ICBSII)*, Chennai, India, pp. 1–4 (2021).
35. Zou, J. and Zhang, Q. "EyeSay: Eye electrooculography decoding with deep learning", *2021 IEEE International Conference on Consumer Electronics*, USA, pp. 1–3 (2021).
36. Khan, I.A., Sajeeb, A., and Fattah, S.A. "An automatic ocular disease detection scheme from enhanced fundus images based on ensembling deep CNN networks", *11th International Conference on Electrical and Computer Engineering*, Dhaka, Bangladesh, pp. 491–494 (2020).
37. Haque, R.U., Pongos, A.L., Manzanares, C.M., et al. "Deep convolutional neural networks and transfer learning for measuring cognitive impairment using eye-tracking in a distributed tablet-based environment", *IEEE Transactions on Biomedical Engineering*, **68**(1), pp. 11–18 (2021).
38. Ştefan, A.M., Paraschiv, E.A., Ovreiu S., et al. "A review of Igaucoma detection from digital fundus

images using machine learning techniques", *International Conference on e-Health and Bioengineering*, Iasi, Romania, pp. 1–4 (2020).

39. Mehmood, T., Alfonso E., Gerevini, A.L., et al. "Combining multi-task learning with transfer learning for biomedical named entity recognition", *Procedia Computer Science*, **176**, pp. 848–857 (2020).
40. Wu, X., Chen, S., Huang, J., et al. "DDeep3M: Docker-powered deep learning for biomedical image segmentation", *Journal of Neuroscience Methods*, **342**, pp. 804–808 (2020).
41. Tian, Y. and Fu, S. "A descriptive framework for the field of deep learning applications in medical images", *Knowledge-Based Systems*, **210**, pp. 106–445 (2020).
42. Image dataset, "https://www.kaggle.com/datasets", [accessed:21.02.2021]
43. Pre-trained model and dockerfile, "https://github.com/itopaloglu/Artificial_Intelligence" [accessed:25.02.2021]

Biography

Ismail Topaloglu received the MSc and PhD degrees in Electrical Technology from the Institute of Natural and Applied Sciences, University of Gazi, Ankara, Turkey in 2009 and 2013, respectively. He joined the Device Modelling Group, School of Engineering, University of Glasgow, as a Research Fellow. Form March 2022 until going on he is a Research Fellow (Post Doctorate) in Electronics and Nanoscale Engineering and the member of the Device Modeling Group in the School of Engineering, University of Glasgow. He has been working for many years in computational electromagnetic. The author's research interests include the design, modelling, and analysis of power devices, optimization of linear/rotary systems and electromagnetic devices, electronic semiconductor design and modelling, as well as machine learning, image processing, C++ and artificial intelligence using Python.