Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2017) 25: 1106 – 1115
© TÜBİTAK
doi:10.3906/elk-1507-190

Iris nevus diagnosis: convolutional neural network and deep belief network

Oyebade OYEDOTUN[1,2,*], **Adnan KHASHMAN**[2]
[1]Department of Electrical & Electronic Engineering, Faculty of Engineering, Near East University, Lefkoşa, Northern Cyprus
[2]European Centre of Research and Academic Affairs, Lefkoşa, Northern Cyprus

Received: 22.07.2015 • **Accepted/Published Online:** 24.03.2016 • **Final Version:** 10.04.2017

Abstract: This work presents the diagnosis of iris nevus using a convolutional neural network (CNN) and deep belief network (DBN). Iris nevus is a pigmented growth (tumor) found in the front of the eye or around the pupil. It is seen that racial and environmental factors affect the iris color (e.g., blue, hazel, brown) of patients; hence, pigmented growths may be masked in the eye background or iris. In this work, some image processing techniques are applied to images to reinforce areas of interests in them, after which the considered classifiers are trained. We describe the automated diagnosis of iris nevus using neural network-based systems for the classification of eye images as "nevus affected" and "unaffected". Recognition rates of 93.35% and 93.67% were achieved for the CNN and DBN, respectively. Hence, the systems described in this work can be used satisfactorily for diagnosis or to reinforce the confidence in manual-visual diagnosis by medical experts.

Key words: Iris nevus, diagnosis, convolutional neural networks, deep belief networks

1. Introduction

Iris nevus is a tumor found on the iris or areas around the pupil. It often requires little or no attention, but medical studies have shown that such benign growths have the capability to transform into iris melanoma, which is malignant and therefore requires very critical medical intervention. By Kaplan–Meier estimates, iris nevus transformation into melanoma occurs in $<1\%$, 3%, 4%, 8%, and 11% of cases at 1, 5, 10, 15, and 20 years, respectively [1].

The diagnosis of iris nevus is achieved by taking eye images of patients, which a medical expert examines for growths and pigmentations. Generally, medical experts visually scan the images for signs of nevi. A medical expert may recommend the archiving of the eye images so that the history of suspected cases can be followed for possible transformations into malignancy. It has also been found that there exists the chance of iris nevus transforming into secondary glaucoma, which is another serious medical condition requiring crucial medical attention. It is noteworthy that one of the most effective approaches to fighting iris melanoma or secondary glaucoma is having an efficient diagnosis system for iris nevus, a phase that often precedes malignancy. Once iris nevus is diagnosed, effective medical intervention can be carried out. The diagnosis of iris nevus is made difficult in that patients often have different colors of irises; hence, the pigmented growths may be concealed within the iris when manual examination of medical eye images is carried out [2]. Furthermore, we cannot rule out that such a delicate examination process may be affected by situations such as eye conditions of medical experts

[*]Correspondence: oyebade.oyedotun.k@ieee.org

(e.g., color appreciation deficiency) and fatigue. Hence, this work describes the automatic detection of iris nevus in patients, achieved using machine intelligence (based on neural networks). Such systems do not suffer image processing and appreciation deficiency as found in humans; also, errors due to fatigue are completely absent.

This work is a sequel to an earlier published work on the diagnosis of iris nevus. The capability of the networks considered in this paper for the enhanced classification of iris nevus is presented and discussed in detail in the following sections. Furthermore, the results presented in this paper are advancements on a previously published work for the same task using support vector machines, backpropagation networks, and a radial basis function network.

The capability of convolutional neural networks (CNNs) in the computer vision field is quite remarkable [3–5]. This can be attributed to the architectural build of these networks. They enjoy a structure that closely simulates biological visual processing, and many advances and modifications to the initial architecture of these networks have been subject to significant research breakthroughs in medicine, enabling better understanding of how humans or animals achieve similar perception tasks almost effortlessly.

The standard LeNet-5 architecture has been used for the validation of the considered application, using eye images of patients collected from a publicly available database. The raw images were processed and used as data with which networks were trained and tested in this research.

2. Related work

In a recent work, Oyedotun et. al proposed the use of intelligent decision support systems to model the diagnosis of iris nevus. The systems considered in this work include support vector machines, backpropagation neural networks, and radial basis function networks. In the same work, eye images of patients from a publicly available database were collected. These images were processed in order to extract and reinforce features and regions of interest, respectively. The processed images were used to train the designed networks. The best performance obtained from the models was obtained from the support vector machine implemented with quadratic kernel, giving a recognition rate of 90.8%, while a recognition rate of 72.5% was obtained from the other support vector machine model that was implemented with a linear kernel. Furthermore, the backpropagation neural network and radial basis function network achieved recognition rates of 87.3% and 88.6%, respectively [2].

3. Neural network

3.1. Convolutional neural network preliminaries

The architecture of CNNs generates key ideas for guiding its learning, which are local receptive fields, weights sharing, and pooling operations [6]. CNNs are basically alternating layers of convolution and pooling or subsampling layers, followed by a conventional classifier as the last layer in the network [7].

Convolutional networks are very apt in pattern recognition problems; unfortunately, these networks can be somewhat hard to train. The big questions have almost always been what parameters or operations can be manipulated for a better response of these networks. Some parameters that significantly contribute to the performance of CNNs are the size of the local receptive field or kernels used for convolution operations, number of feature maps in each convolution layer, type of pooling operation implemented in the subsampling layer, build of convolution maps, depth of the network, and activation functions.

In this work, the standard LeNet-5 architecture has been used; it has two convolution layers and two pooling layers (Figure 1).

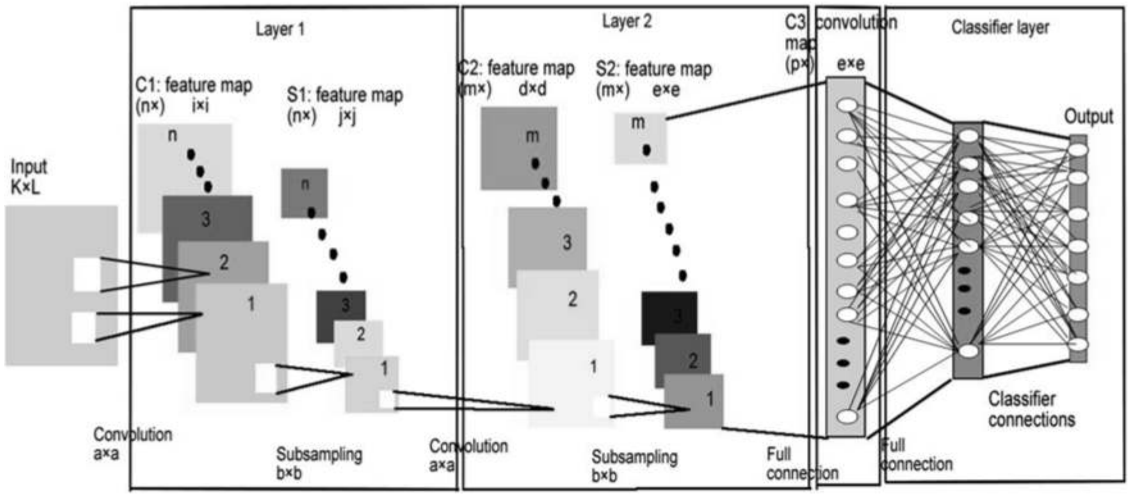

Figure 1. Convolutional neural network.

From Figure 1, the convolution layer feature maps can be seen as C1 and C2 for the first and second layer, respectively; the subsampling feature maps S1 and S2 for the first and second layer, respectively, are also shown. In order to fully appreciate the presented ideas in this research, a brief discussion of the basic CNN build is given below.

3.2. Kernels, convolution, and subsampling

The convolution operation is achieved by using kernels or filters to convolve the input to the convolution layer. Generally, the number of distinct kernels used for this operation is the number of the feature maps in a convolution layer. The kernels extract local features from the input captured by the receptive field. The receptive field is the size of the kernel used. Each feature map represents distinct extracted features from the input, e.g., one may represent vertical edges and another may represent points [8].

The units in each feature map are orientated in 2D, and they share the same set of weights including the bias. Each convolution feature map is succeeded by its corresponding subsampling feature map; hence, the number of subsampling feature maps is the same as the convolution feature maps. Eq. (1), describing the convolution operation, is given below.

$$x_j^l = f(\sum_{i \in M_j} x_i^{l-1} * k_{ij}^l + b_j^l) \tag{1}$$

Here j is the particular convolution feature map, M_j is a selection of input maps, k_{ij} is the convolution kernel, b_j is the bias of each feature map, l is the layer in the network, and f is the activation function [9].

The subsampling layer pools features extracted in the convolution layer; each subsampling feature map aggregates extracted features in its corresponding convolution feature map, usually preceding it [10,11].

A suitable subsampling mask size and type of pooling operation are first determined and then implemented at the subsampling layer. In contrast to convolution operation where the overlap of kernels is allowed, there is usually no overlap of subsampling masks in pooling operations. The subsampling mask captures the regions of the input considered for pooling operations. The two commonly used pooling operations are average pooling and max pooling [12].

The pooling scheme performs an operation on the pixel values captured by the subsampling mask, multiplies it by a trainable coefficient, and then adds a trainable bias [13]. Eq. (2) describes the pooling operation.

$$x_j^l = f(\beta_j^l pool(x_j^{l-1}) + b_j^l) \tag{2}$$

Here x_j^l is the value or outcome of the pooling operation carried out on the jth region in the input, x_j^{l-1} is the jth region of interest captured by the subsampling mask in the preceding layer, 'pool' is the particular operation performed on the region (average or max), β_j^l is a trainable coefficient, b_j^l is a trainable bias, and f is an activation function [7]. The max pooling operation takes the maximum value from the pixel values captured by the pooling window set for the subsampling operation. For average pooling, the average of the pixel values captured by the pooling window is obtained during subsampling operations. It has been established in several research works that the type of pooling operation implemented in the subsampling layer affects network performance, especially for pattern invariance learning [14].

3.3. Deep belief network

Deep belief networks (DBNs) are generative networks used to learn feature representation of data through probabilistic modeling [15]. These networks can be considered as directed, acyclic, and graphical models.

In these networks, all the hidden layers have interconnections that are directed, except the two topmost layers, which are undirected. For the sake of simplicity, a DBN is sometimes considered as a stack of restricted Boltzmann machines (RBMs). Although this is not true in every sense, the approximate conception works and greatly simplifies computation and other intractability issues that do arise. Since the RBM is the backbone of DBNs, it is briefly discussed below.

RBMs have only two layers, the visible (input) layer and the hidden (stochastic) layer. The visible layer is where the input variables are supplied to the network, while feature learning by stochastic units is achieved in the hidden layer. In contrast to Boltzmann machines where there are interconnections between units (neurons) of the same layer, RBMs do not have interconnections for units of the same layer; it can thus be inferred that for RBMs units are conditionally independent.

The RBM training scheme is aimed at maximizing the likelihood of input vectors at a layer below given a configuration of a hidden layer that is directly on top of it [15]. RBMs are used to compute the joint distribution of visible and hidden units, given some model specific parameters. In DBNs, RBMs are stacked on top of one another, and training is achieved through a scheme known as "greedy layer-wise training". In greedy layer-wise training, the weights of each layer are trained one at a time. A DBN with 2 hidden layers, as used in this work, is shown below.

The training of the network is achieved by learning the weights of the whole network in a layer-wise fashion. From Figure 2, the weights between the input and hidden 1, $W_h^{(h1)}$, are first learned and then fixed. The activation values of hidden 1 at the end of learning $W_h^{(h1)}$ are now used as inputs to hidden 2, and the training algorithm is run again in order to learn weights $W_h^{(h2)}$. To learn weights $W_h^{(h3)}$, the activation values obtained at the end of learning $W_h^{(h2)}$ for hidden 2 are used as input for hidden 3, and the training algorithm is run again to learn $W_h^{(h3)}$. The greedy layer-wise training is such that the weight connections between the layers of networks are learned one at a time, in contrast to typical training algorithms where the weights of the whole network are updated each time. The training algorithm used for the layer-wise weights learning in DBNs is known as “contrastive divergence”, proposed by Hinton et al. [16]. The equation used to update network weights is given as Eq. (3).

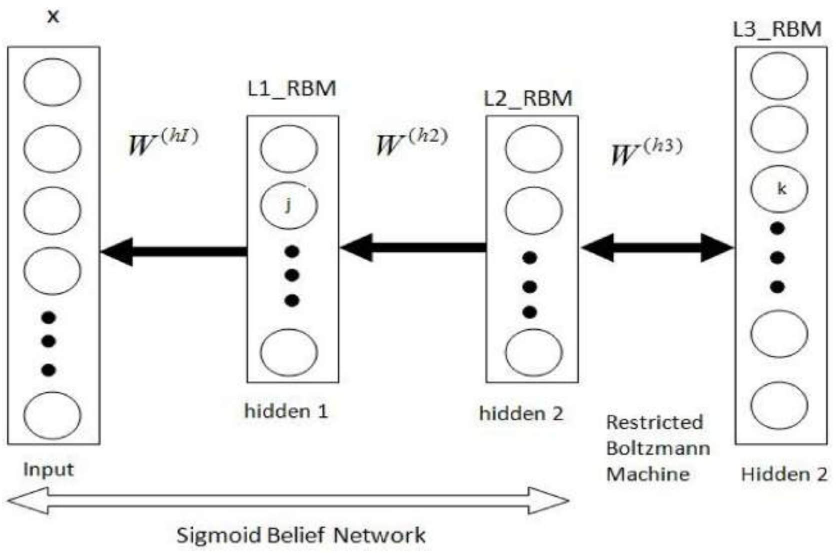

Figure 2. Deep belief network.

$$\Delta W_{ij} = E_{data}(x_i h_j) - E_{\bmod el}(x_i h_j) \tag{3}$$

Here, E$_{data}$ is the actual expectation when h$_j$ is sampled from x, given the training set, and E$_{model}$ is the expectation of h$_j$ sampled from x$_i$ considering the distribution defined by the model [17].

By implementing such a pretraining scheme as discussed above, features that are data-dependent can be extracted through probabilistic samplings. The stacking of several RBMs allows abstract features extracted in a lower layer to be combined into more defined features in the upper layers progressively, i.e. hierarchical learning. Finally, for classification tasks, the DBN (stack of trained RBMs) is fine-tuned (made discriminative) using a supervised learning algorithm, the backpropagation weights update algorithm.

4. Data analysis

The data used in this research are eye images collected from a public database (The Eye Cancer Foundation, Eye Cancer Network, http://www.eyecancer.com/research/image-gallery/12/iris-tumors), containing nevus-affected eyes and nevus-unaffected eyes. These images were processed to extract regions of interest and relevant features for learning.

Through thorough study, it was found that the region affected in almost all cases is the iris; hence, using the whole images would only lead to redundancy in information for the task. Therefore, irises of eye images were considered as the region of interest and then cropped out, so that further image processing could be applied to them. Sample images of unprocessed irises are shown below; both nevus-unaffected (healthy) and nevus-affected (unhealthy) irises are shown in Figures 3a and 3b, respectively.

It will be seen in Figure 3a that the pupil is surrounded by a uniform and clear background (more formally called the iris). In contrast, Figure 3b shows the pupil surrounded by a pigmented and nonuniform background. The image processing techniques applied were hence aimed at obtaining end images in which the irises (pupil background) were reinforced, i.e. for clear and uniform irises, the obtained images have more uniformity, while for pigmented and nonuniform irises, the pigmentation and nonuniformity becomes broader. The image processing operations are described briefly below.

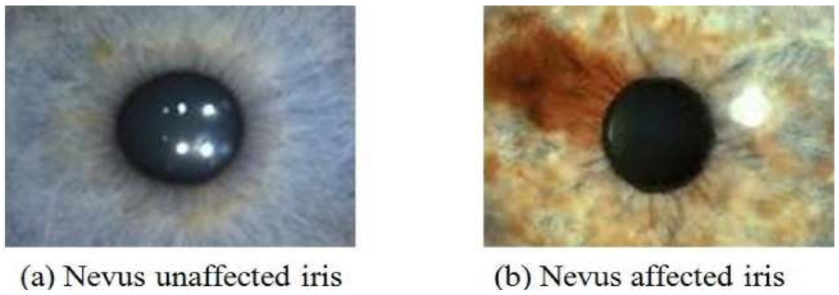

Figure 3. Unprocessed irises.

The images were rotated through an angular step of 15°, as this allows for rotational invariance to be built into the designed systems during training and creation of a larger training database.

The images were converted from color (RGB) to grayscale, after which the images were filtered using a median filter of size 10 × 20; the filtering was used to roughly achieve the reinforcement of uniformity and nonuniformity in the irises. It also helped to reduce camera glares found in some of the collected images.

The images were then downsized to 30 × 30 pixels to reduce computational requirements and training time. Furthermore, the pixel values of the images were normalized to the range 0 to 1, as obtained in neural networks input.

In Figure 4, samples of unprocessed and processed iris images are shown. Subfigures a1 and b1 show unprocessed images for nevus-affected irises, while a2 and b2 show the processed images, respectively. Subfigures c1 and d1 show unprocessed images for nevus-affected irises, while c2 and d2 show the processed images, respectively.

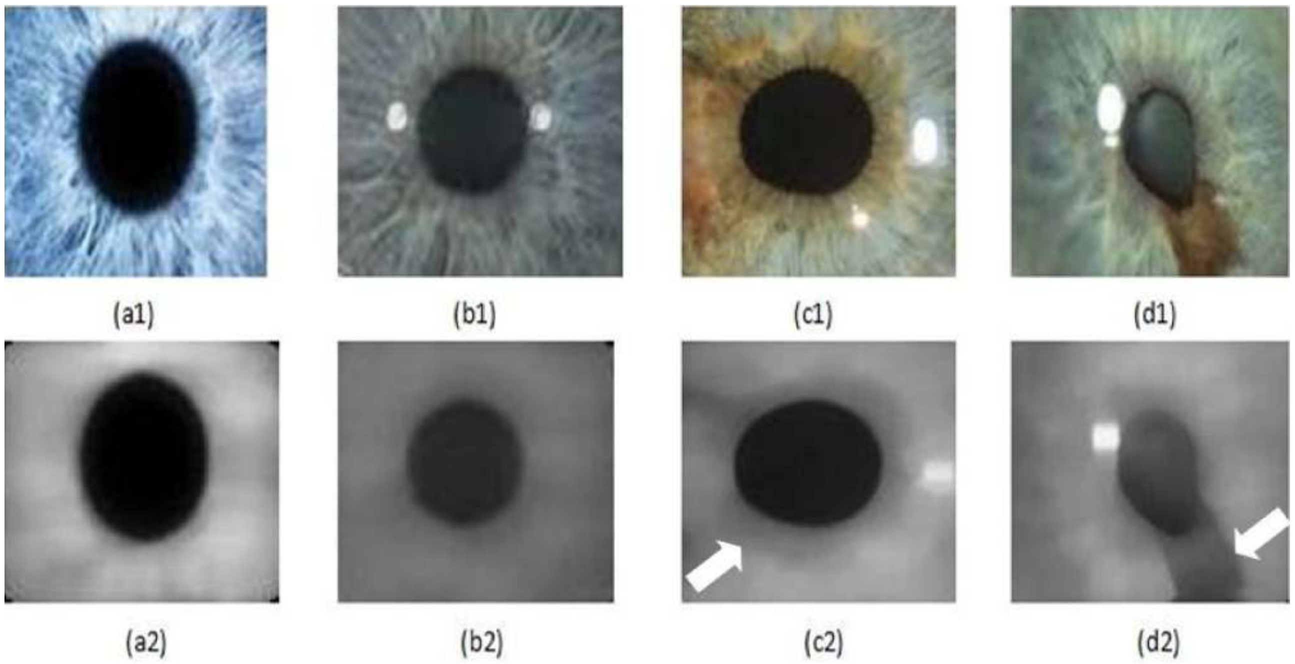

Figure 4. Nevi-unaffected and nevi-affected irises (white arrows show affected regions) [2].

5. Training of networks

The considered networks were trained using the processed images of the irises as discussed in Section 4. The designed CNN and DBN were trained on 500 samples containing both nevus-affected and unaffected iris images; the input images were of size 30 × 30 pixels (900 pixels). The training data contain 299 nevus-affected and 201 nevus-unaffected samples. The initial weights of the networks were initialized randomly, while final training parameters were heuristically obtained during training. Note that for all the networks in this work, a 10-fold cross-validation criterion is used to stop training in order to reduce the chance of overfitting. The output coding of the classification task (output neurons) has been achieved as described below.

- Nevus-unaffected iris: [0 1]
- Nevus-affected iris: [1 0]

5.1. Convolutional neural network

A convolution kernel of size 5 × 5 was used to convolve the input images, and 6 feature maps were generated. A subsampling mask size of 2 × 2 was used for pooling features extracted in the first convolutional layer. Furthermore, a convolution kernel of size 5 × 5 was used to convolve features aggregated in the preceding (subsampling) layer, and 12 feature maps were generated. Lastly, the features extracted in the second convolution layer were pooled using a subsampling mask of size 2 × 2. The classifier layer is a single hidden layer neural network with 300 hidden neurons. During training, several training experiments were performed in order to determine optimal parameters for the learning rate, convolution kernel sizes, and subsampling mask sizes. A step size of 0.06 was used as the learning rate. Table 1 shows the different networks trained with respective learning parameters; furthermore, it can be seen that CNN-2 achieved the lowest mean squared error (MSE) after training and hence optimum learning.

Table 1. Training parameters for CNN.

Networks	CNN-1	CNN-2	CNN-3
Number of training samples	500	500	500
Activation function	Log-sigmoid	Log-sigmoid	Log-sigmoid
Learning rate	0.02	0.06	0.24
Maximum iterations	6500	6500	6500
Iterations reached on training	5300	4000	3140
Training time (s)	126	102	98
Mean squared error (MSE)	0.007	0.003	0.005

The output layer of the CNN has 2 neurons to accommodate the two categories of image classification. The MSE function was used as the cost function for learning and a batch size of 10 for the training samples was used to achieve stochastic gradient descent update of network weights. The maximum number of iterations is set as 6500. The learning curve for the CNN with optimum learning, CNN-2, is shown in Figure 5.

5.2. Deep belief network

The designed DBN was also trained with the same samples used for the CNN; two hidden layers were implemented in the DBN. Training parameters are shown below in Table 2. Note that using the aforementioned cross-validation training scheme, the maximum number of iterations for the DBNs is set as 300. Several networks were trained, and it can be seen that the network that achieved the lowest MSE after training is DBN-1, compared to DBN-2 and DBN-3.

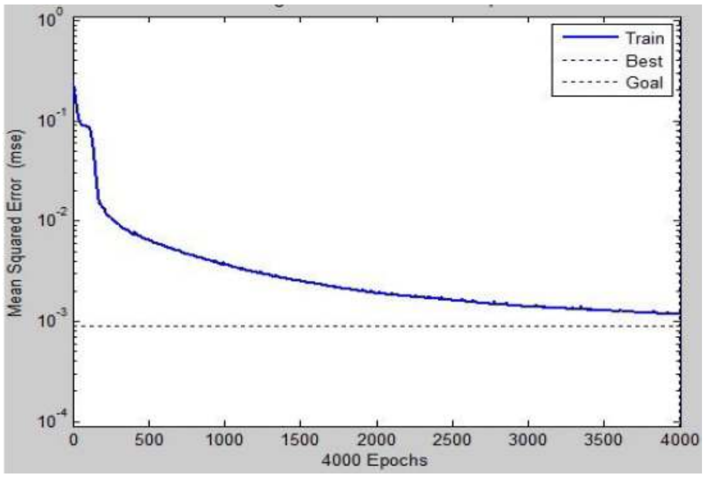

Figure 5. Learning curve for CNN-2.

Table 2. Training parameters for DBN.

Networks	DBN-1	DBN-2	DBN-3
Number of training samples	500	500	500
Hidden layer 1- neurons	25	30	35
Hidden layer 2- neurons	15	20	15
Activation function	Log-sigmoid	Log-sigmoid	Log-sigmoid
Learning rate for pretraining	0.2	0.1	0.3
Learning rate for fine-tuning	0.2	0.2	0.3
Iterations for pretraining	10	15	10
Max. iterations for fine-tuning	300	300	300
Iterations for fine-tuning	100	80	120
Training time (s)	54	52	59
Mean squared error (MSE)	0.0016	0.0042	0.0029

6. Testing of networks

The two trained networks were tested with 316 sample images that were not part of the training data to obtain the performance of the networks in classifying new cases. There are 181 nevus-affected irises and 135 nevus-unaffected irises in the test data. Eq. (4) is used to obtain the recognition rates (r.r) of the networks.

$$r.r = \frac{\text{number of samples classified correctly}}{\text{Total number of test samples}} \tag{4}$$

Table 3 shows the recognition rates achieved by the trained networks.

Table 3. Testing of networks.

Network	CNN	DBN
Number of test samples	316	316
Correctly classified samples	295	296
Recognition rate	93.35%	93.67%

6.1. Discussion and conclusion

From the table above, it will be seen that both networks achieved roughly the same recognition rates on the test data. However, it is noteworthy that the DBN learned the classification task in a shorter time compared to the CNN. Furthermore, the DBN has a simpler architecture than the CNN, in which network parameters that have significant effects on the performance of the network have to be decided; such parameters include the sizes of the convolution kernels and subsampling masks, number of features maps to be implemented in the convolution, and pooling layers.

In an earlier work on the same task as described in this work, a backpropagation neural network (BPNN), radial basis function network (RBFN), and support vector machine (SVM) were used for the classification of iris images; Table 4 shows the results [2].

Table 4. Recognition rates for other networks in earlier work [2].

Network	BPNN	RBFN	SVM
Number of test samples	316	316	316
Correctly classified samples	276	280	287
Recognition rate	87.30%	88.6%	90.8%

In this research, we described a CNN and DBN, which are two more biologically inspired neural networks for learning the classification of processed eye images into nevus-affected and unaffected categories. During the training phase of the networks, we were able to establish that the designed DBN took less time and fewer iterations for training compared to the CNN. It was observed that the CNN and DBN achieved roughly the same recognition rates when tested.

Furthermore, it can be seen that both the CNN and DBN described in this work outperform other networks that were used in an earlier work for the same classification task; hence, there should be higher confidence in the accuracy of the CNN and DBN when deployed for this task as compared to the backpropagation neural network, radial basis function network, and support vector machine.

In conclusion, this work is significant in that the correct detection of iris nevus greatly reduces the risk of nevus transformation into melanoma and secondary glaucoma since medical experts can now promptly follow diagnosed patients with necessary medical interventions.

References

[1] Shields CL, Kaliki S, Hutchinson A, Nickerson S, Patel J, Kancherla S, Peshtani A, Nakhoda S, Kocher K, Kolbus E et al. Iris nevus growth into melanoma: analysis of 1611 consecutive eyes. Ophthalmology 2013; 120: 766-772.

[2] Oyedotun OK, Olaniyi EO, Helwan A, Khashman A. Decision support models for iris nevus diagnosis considering potential malignancy. International Journal of Scientific & Engineering Research 2014; 5: 419-426.

[3] Li W, Jia F, Hu Q. Automatic segmentation of liver tumor in CT images with deep convolutional neural networks. Journal of Computer and Communications 2015; 3: 146-151.

[4] Syafeeza AR, Khalil-Hani M, Liew SS, Bakhteri R. Convolutional neural networks with fused layers applied to face recognition. International Journal of Computational Intelligence and Applications 2015; 14: 1550014.

[5] Dieleman S, Willett KW, Dambre J. Rotation-invariant convolutional neural networks for galaxy morphology prediction. Mon Not R Astron Soc 2015; 450: 1441-1459.

[6] Lawrence S, Giles CL, Tsoi AC, Back AD. Face recognition: a convolutional neural network approach. IEEE T Neural Networ 1997: 98-113.

[7] Doorn J. Analysis of deep convolutional neural network architectures. In: 21st Twente Student Conference on IT; 23 June 2014; Enschede, the Netherlands. pp. 3-5.

[8] Nagi J, Ducatelle F, Di Caro G, Cireşan D, Meier U, Giusti A, Nagi F, Schmidhuber J, Gambardella LM. Max-pooling convolutional neural networks for vision based hand gesture recognition. In: IEEE International Conference on Signal and Image Processing Applications; 16–18 November 2011; Kuala Lumpur, Malaysia. New York, NY, USA: IEEE. pp. 343-349.

[9] Bouvrie J. Notes on Convolutional Neural Networks. Cambridge, MA, USA: Center for Biological and Computational Learning of Massachusetts Institute of Technology, 2006.

[10] Masci J, Meier U, Cireşan DC, Schmidhuber J. Stacked convolutional auto-encoders for hierarchical feature extraction. Lect Notes Comp Sci 2011; 6791: 52-59.

[11] Cireşan DC, Meier U, Masci U, Gambardella LM, Schmidhuber J. Flexible, high performance convolutional neural networks for image classification. In: Proceedings of the 22nd International Joint Conference on Artificial Intelligence; 16–22 July 2011; Barcelona, Spain. pp. 1238-1243.

[12] Scherer D, Muller A, Behnke S. Evaluation of pooling operations in convolutional architectures for object recognition. In: 20th International Conference on Artificial Neural Networks; 15–18 September 2010; Thessaloniki, Greece. pp. 1-6.

[13] LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. P IEEE 1998; 8: 2278-2324.

[14] Yu D, Wang H, Chen P, Wei Z. Mixed pooling for convolutional neural networks. In: 9th International Conference on Rough Sets and Knowledge Technology; 24–26 October 2014; Shanghai, China. pp. 364-375.

[15] Oyedotun OK, Olaniyi EO, Khashman A. Deep learning in character recognition considering pattern invariance constraints. International Journal of Intelligent Systems and Applications 2015; 7: 1-10.

[16] Hinton GE, Osindero S, Teh YW. A fast learning algorithm for deep belief nets. Neural Comput 2006; 18: 1527-1554.

[17] Deng L. An overview of deep-structured learning for information processing. In: Asia-Pacific Signal and Information Processing Association Annual Summit and Conference; 9–12 December 2014; Chiang Mai, Thailand: pp. 2-4.