Hybrid deep convolutional neural models for iris image recognition

Jenkin Winston, J., Hemmanth, J., Angelopoulou, A. and Kapetanios, D.E.

This is an author's accepted manuscript of an article published in Multimedia Tools and Applications, DOI: 10.1007/s11042-021-11482-y, 2021.

The final publication is available at Springer via:

https://doi.org/10.1007/s11042-021-11482-y

Hybrid deep convolutional neural models for iris image recognition

J. Jenkin Winston, D. Jude Hemanth, A. Angelopoulou, E. Kapetanios

Karunya Institute of Technology and Science, Coimbatore, India

School of Computer Science and Engineering, University of Westminster, London, UK

Abstract

This paper briefly explains about the application of deep learning-based methods for biometric applications. This work attempts to solve the problem of limited availability of datasets which affects accuracy of the classifiers. This paper explores the iris recognition problem using a basic convolutional neural network model and hybrid deep learning models. The augmentations used to populate the dataset and their outputs are also shown in this study. An illustration of learned weights and the outputs of intermediary stages the network like convolution layer, normalization layer and activation layer are given to help better understanding of the process. The performance of the network is studied using accuracy and receiver operating characteristic curve. The empirical results of our experiments show that Adam based optimization is good at learning iris features using deep learning. Moreover, the hybrid deep learning network with SVM performs better in iris recognition with a maximum accuracy of 97.8%. These experiments have also revealed that not all hybrid networks will give better performance as the hybrid deep learning network with KNN has given lesser accuracy.

Keywords

Biometrics, Iris recognition, Convolutional neural networks, Deep learning

1. Introduction

In organizations, large territories, computer networks and even in personal space for varied applications, security has become a major concern. Hence, to ensure security authentication has become more important. Authentication is a method of proving our digital identity to a machine. Few traditional authentication methods are password, personal identification numbers and personal identity verification cards. All these methods either will cause a difficulty to remember or they are vulnerable to theft. An alternate solution which is widely used in recent years is the use of biometric features for authentication. The features may be biological or behavioural. These features characterize the individuals. They are more unique and does not change largely over their life span. Iris is one such biometric feature which is more stable and accurate for automated biometric recognition. It is made of interconnected web of tissues and muscle fibres. Mainly it contributes to tinted rings, crypts and furrows. This improves the textural patterns present in the iris region of each person. Hence, it is widely accepted and adapted in many automated authentication systems.

Several machine learning techniques which give the machine the ability to learn without being explicitly programmed has become more established among researchers over the recent years. The first automated iris recognition was presented by Daugman [1] in 1993. In this the iris region is encoded into a compact sequence of 256 bytes using multi-scale 2D Gabor wavelet coefficients. The confidence levels of a given iris were computed using Exclusive-OR comparisons. This proved to be a rapid and reliable authentication process that can be employed using iris biometrics. Pre-dominantly, classical iris recognition process includes pre-processing the input iris image, localization, normalization, feature extraction and classification. In literature, there are several contributions to each of these stages for iris recognition. The binarized iris code of the traditional method is too large. Hence, to identify the optimal iris code, the problem is formulated as an optimization problem in [2]. Two really simple objective functions were used to reduce the bits of the iris code which is of much lesser significance. However, the introduced optimization increases the computation involved in calculating the encoded features.

With many technology advancements, the sensors for capturing iris gets upgraded very often. Hence, it is not necessary that the sensor used in the training phase to be same as used in the validation or testing phase. In [3], J. K. Pillai et. al. have done extensive study on the influence of these cross sensors in iris recognition. They have developed a kernel learning based framework. A constraint-based optimization to adjust the performance limitations due to older sensor has been adapted through learning selected parameters. This improves the acceptance rate of the iris recognition. A cross spectral approach is tried by authors in [4]. This is done to extract more rich features from the iris region. A phase-only correlation and band-limited phase-only correlation method is introduced in the paper. As the phase information is more sensitive to artifacts present in the iris region, homomorphic filtering is done to alleviate this problem. However, due to non-linear intensity variation among cross spectrum, this method is not effective. Although, cross-spectral sensors provide ample information on iris feature, it was not successful. The problem of illumination variation in cross spectral sensor is addressed by Maulisa Oktiana et al. in [5]. It is done by Gradient faces-based normalization technique. Along with it, features were extracted using Difference of Gaussian filtering and Binary statistical image information. The scores of the classification from these three features are fused to get the final result. However, this method is also sensitive to specular reflections.

All the traditional machine learning algorithms used hand-crafted features for the learning the information about the given iris. Even the employment of cross-spectral sensors was not giving satisfactory results in unconstrained environment. Moreover, all these features are computationally expensive and are not robust. Lately, deep learning has become more popular and gives promising results across varied applications. And these deep learning models does not require hand crafted features. The development of dedicated Complimentary Metal-Oxide Semiconductor (CMOS) iris sensor and development of compact Graphical Processing Unit (GPU) has attracted several researchers to contribute towards iris recognition using deep learning. GPUs consume more power than CPUs. In [6], authors have investigated the energy efficiency of GPU with application towards iris recognition. The hardware configuration along-with improvement of algorithm has reduced the energy consumption. It enables parallel processing on multiple cores. As a result, energy consumption is reduced because of reduced execution time.

Few of the common deep learning architectures are Recurrent Neural Network, Convolutional Neural Network, Generative Adversarial Networks and Capsule Network. Pre-processing is the preliminary step in the learning process. It gives clear on the information on the subject to learn. Ming Liu et al. in [7] have suggested fuzzy based image enhancement. The use of triangular average filter and triangular median filter along with capsule network shows improvement in accuracy. On the contrary, other fuzzy membership functions for filtering is not explored to improve the accuracy further for practical implementation. Eduardo Ribeiro et al. in [8] have tried to bring super-resolution to make the iris region more informative. A comparative investigation on different methods to create a super-resolution from a single iris image. Compared to Generative Adversarial Network (GAN), CNNs produce better resolution. However, maintaining a balance between preserving the edges and smoothing is challenging to achieve photo-realistic super-resolution. In [9], authors have used the contours and edges as iris signatures. A Deep Belief Network (DBN) along with variable rate multi-layer feed forward neural network is used in the classification unit. In hidden layers the weight update occurs through contrastive divergence method. The edge features have improved the performance at low signal to noise ratios.

Capsule network and pre-trained network models are explored for iris recognition by authors in [10]. An adjusted dynamic routing algorithm is proposed to make the convergence during the learning process easier. This predicts the feature vector. Reconstruction of the images is done to show the versatility of the method. Pre-trained network along with dynamic routing displays better accuracy for iris recognition. The public datasets available for iris is usually limited. This causes challenges like overfitting and poor accuracy while validating deep learning architectures. The authors in [11], have proposed a special Generative Adversarial Network to augment available dataset. Image augmentation provides sufficient number of diversities among the training data. As the input image has periocular region during augmentation, it affects the performance of the classifier. Hence, the iris region was segmented and normalized and fed to the augmentation process. This enabled in providing better performance. Based on the depth information 3D models of iris were studied by authors in [12]. The depth information was computed using CNN model. The 3D rubber sheet maps of the iris produced better recognition than the 2D iris code. A discrete hashing method of generating iris code from the CNN features is used to reduce the bit size of the iris code in [13]. CNN along with the hashing method enhances the performance. A unified framework is worked upon by authors in [14] to develop a new deep learning-based iris recognition model. The iris region is segmented using masked R-CNN. The features are extracted from the normalized iris region using FeatNet. A triplet network and an extended loss function is used to learn the convolutional kernels of the FeatNet. With GPU, all the computations are faster. This greatly avoids the false accept rate. The deep features of the iris is taken using sparse filters on local patches and the whole picture in [15]. The likelihood of similarity is predicted by maximizing the likelihood. As they work on histogram from collaborative subspace of patch, the model may be inaccurate in different illumination conditions.

In forensics, during testing phase the iris may be captured from any image sensor. Hence, to have a better performance irrespective of the sensor, an identification model is used to find sensor used to capture the iris and appropriate enhancement method is used to before giving to the classifier. This method is extensively studied by authors in [16]. This method helps to ensure seamless interoperability for iris recognition. Transfer learning using pre-trained network models are investigated by authors in [17]. The CNN used in this model is able extract more complex features from the iris. Few shot learning can be more useful in iris biometrics as most of the public iris datasets have limited number of iris samples on each subject for training and testing. Among different pre-trained models, DenseNet provides better recognition results. A novel approach using filter bank was suggested by authors in [18]. The architecture is too complicated for training. But it shows promising results for recognition in cross sensor iris verification. Although, the training is time consuming, the testing can be extremely faster. To improvise on the limitation of softmax classification layer, a tight center loss function is used in the classification layer of the complex convolutional networks is suggested by authors in [19]. This helps the network learn more discriminant features. It reduces the false accept rate of the classifier. This method will be more beneficial for working on large iris datasets.

Very little work is done on optimization of the network design through optimization. The constrained optimization method helps to attain an optimal network architecture with lesser computation. This method is investigated by authors in [20]. It reduces the false acceptance rate of the classifier. It also reduces the memory required for the implementation of deep learning architecture. This network helps to extract both optimal features and an optimal network design that works well for iris recognition. A similar end to end trainable deep neural network is experimented by Quingqiao Hu et al. in [21]. Shervin Minaee et al. [22], have studied in detail about a pre-trained network and its influence of different layers on the recognition accuracy. Ranjeet Srivastva et al [35] has also tried with pre-trained deep networks like ResNet and Densenet through transfer learning for ECG based biometric recognition. With minimalistic effort transfer learning also seems to be a good option in deep learning solutions. As it is not built for the target application, it cannot be directly adapted because it may have negative effect without optimization.

The EEG data captured from the brain was also used with deep learning methods by authors in [36]. This work includes models like convolutional and recurrent networks for biometric identification.

It performed well above the traditional machine learning methods. A creative approach of converting 1D time series data into 2D image data was used by authors in [37] for biometric identification using ECG. This enables in better feature extraction from the ECG signal for authentication. A 3D hand pose in air signature was used as biometric by authors in [38]. In iris biometrics, a lightweight deep learning model was given by authors in [39]. This model is fast in feature extraction and has lesser error in recognition. A cancellable type iris biometric was discussed by authors in [40] for cloud and stand alone based platforms.

It is observed that the iris has more textural features and more robust to use in a biometric system. Several feature extraction methods along with machine learning methods have significantly contributed to the iris recognition. However, with large number of classes to recognize, the traditional machine learning has algorithmic constraints and solving them is more complex as the feature values are non-linear. With the advent of GPUs, the research in deep learning has proliferated and adds significant value to iris recognition. The success of deep learning has been in large number of layers chosen to form the learning network. However, more layers add up to the computation of the network weights. So, the challenge in deep learning is to find a suitable learning architecture with appropriate choice of hyperparameters for the relevant task it is assigned.

This paper suggests few deep learning models that improves the performance of the classifier, more specific towards iris recognition. Unlike traditional iris recognition process, this does not include localization and normalization. Instead, a significant portion of the iris is cropped for deep feature extraction. It also studies the influences of different optimizer on the classifier performance. The performance is also compared with state-of-the-art methods to show its robustness. A similar work on iris recognition using deep learning architectures is discussed in articles [23-28].

The structure of the rest of the paper is as follows. Section 2 gives an overview of the proposed work. The pre-processing done to support the input iris image for further processing in the deep learning architecture is discussed in section 3. In section 4, the proposed convolutional neural network is discussed in detail. The architecture of the hybrid convolutional neural network is detailed in section 5. The methodology of the iris recognition using deep learning model is given in section 6. The numerical results of the proposed methods is analysed in section 7. The conclusions are given in section 7.

2. Proposed work

Deep learning has been the state-of-the-art method used in various classification problem. The availability of a smaller number of samples per subject, has limited people from exploring the deep learning methods for iris recognition. Very lately, few articles are seen in deep learning for iris recognition. Since then, it has shown ample challenges to improve upon. This paper gives an elaborate detail on the deep learning architecture proposed for iris recognition application. The figure shows the block diagram of the proposed work.

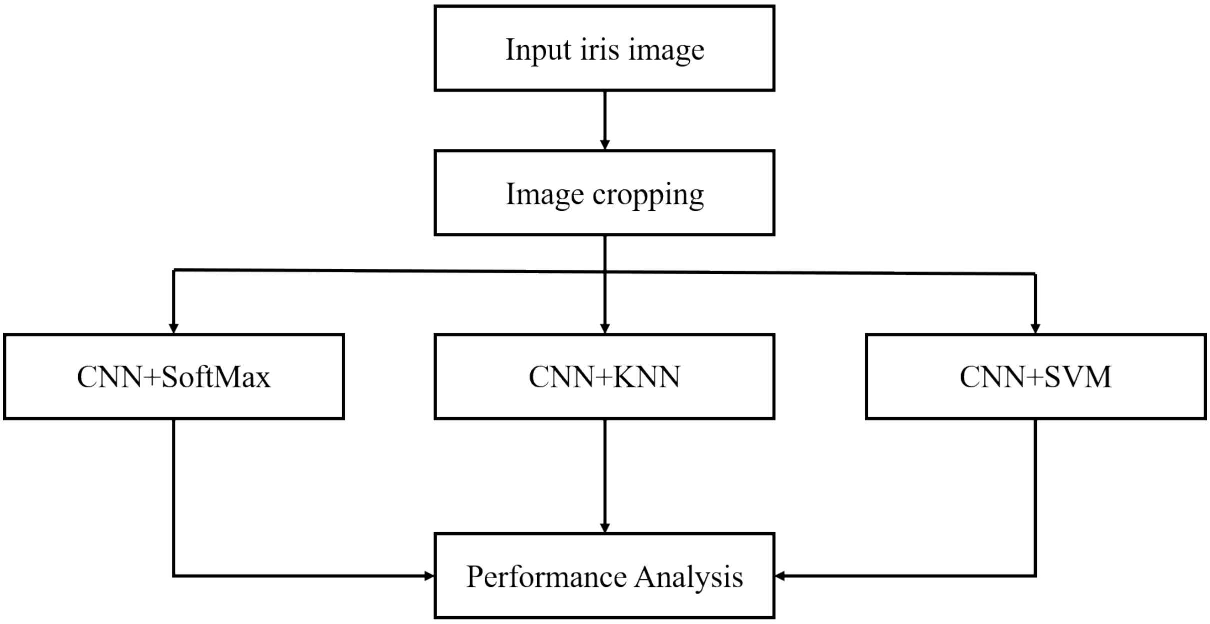

Figure 1. Block diagram of the proposed work

The iris images available in public databases are of varied dimensions. As this deep learning method involves convolution, it is expected to have images with dimensions having equal number of row and columns. Hence, the iris image cropped into a square image during pre-processing. Later it is recognized using three different deep learning architectures. The results of the classifier is compared using their performance metrics. Unlike other machine learning techniques, these deep learning methods involves several hyper-parameters that control the performance of the classifier. Hence, this work gives an clear idea on the steps involved in the convolution based deep learning architectures and their parameter selection to build a robust deep learning classifier.

3. Image cropping

In the image cropping step, the three-dimensional image is converted to two-dimensional image through gray scale conversion. Further the gray scale image is cropped into the size of square to be acceptable by the convolutional network. It also in a way locates the region of interest on the image from which further processing has to be done. Firstly, the 3-dimensional image of size MxNxZ is converted into 2 dimensional image of size MxNx1. Next, around the center of the images (M/2,N/2) the image is cropped to a size of RxC. Usually, the R is taken equal to C to support convolution operation of our deep learning network. The center of the iris can also be located using some object detection techniques also. For simplicity, we have assumed the center of the image as the center of the iris. The sequence of steps involved in the pre-processing is given in figure 2. The resultant image is a square such that it supports image convolution.

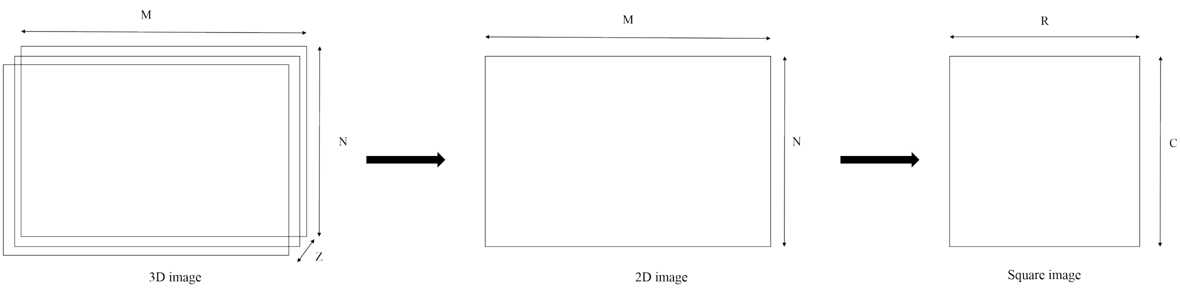

Figure 2. Steps in pre-processing

4. Convolutional Neural Network Framework

There are several deep learning neural network architecture discussed in the literature. However, the Convolutional neural networks are more popular in recognition applications because of its ability to learn features at different hierarchical levels. This gradient based learning of deep neural network was suggested by Yann LeCun et al. in [29]. Traditional classifiers has a separate feature extractor and a separate trainable classifier. This architecture particularly eliminates the need of handcrafted features in a recognition system. It usually has an integrated feature encoder and classifier. These CNNs are multi-layer networks which has the ability to encode high dimensional non-linear information from a huge dataset through gradient descent. It makes it more suitable for recognizing images with more complex iris patterns. The architecture of the proposed convolutional neural network is given in figure 3.

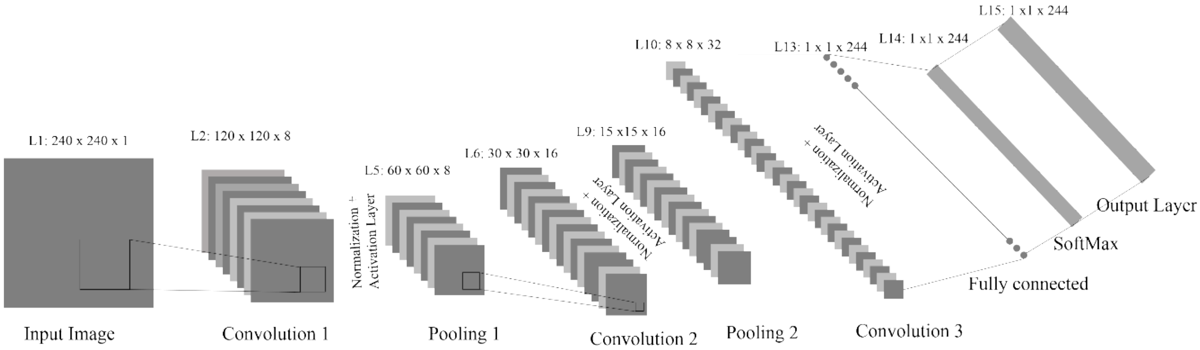

Figure 3. General Architecture of the Convolutional Neural Network

The architecture used in this paper has 15 layers. Figure shows the general architecture of the proposed CNN. It has three convolutional layers, batch normalization layers, and activation layer, two pooling layers and a one fully connected layer, softmax layer and output layer. The details of each layer in the network is given in table 1. All the learnable parameters in the network are adjusted using gradient descent approach such that the error is minimized. These networks are trained over a large number of iterations over a large dataset. The choice of each parameter can influence the parameter of the system. On passing an iris image into the architecture the corresponding neuron of the output layer will be excited. This helps the network to recognize who is that individual.

Table 1. Details of Enhanced Network configuration

Layer	No. of filter	Filter size	No. of Stride	Feature size	Learnable parameters	Total Learnable Parameters
Image Input	-	-	-	240x240x1	-	-
Convolution 1	8	3x3	2x2	120x120x8	Weights: 3x3x1x8 Bias: 1x1x8	80
Batch Normalization 1	-	-	-	120x120x8	Offset: 1x1x8 Scale: 1x1x8	16
ReLU 1	-	-	-	120x120x8	-	-
Max Pooling 1	-	2x2	2x2	60x60x8	-	-
Convolution 2	16	3x3	2x2	30x30x16	Weights: 3x3x8x16 Bias: 1x1x16	1168
Batch Normalization 2	-	-	-	30x30x16	Offset: 1x1x16 Scale: 1x1x16	32
ReLU 2	-	-	-	30x30x16	-	-
Max Pooling 2	-	2x2	2x2	15x15x16	-	-
Convolution 3	32	3x3	2x2	8x8x32	Weights: 3x3x16x32 Bias: 1x1x32	4640
Batch Normalization 3	-	-	-	8x8x32	Offset: 1x1x32 Scale: 1x1x32	64
ReLU 3	-	-	-	8x8x32	-	-
Fully Connected	-	-	-	1x1x224	Weights: 224x2048 Bias: 224x1	458976
SoftMax	-	-	-	1x1x224	-	-
Classification Output	-	-	-	-	-	-

The cascade of blocks in the proposed CNN architecture is given in figure 4.

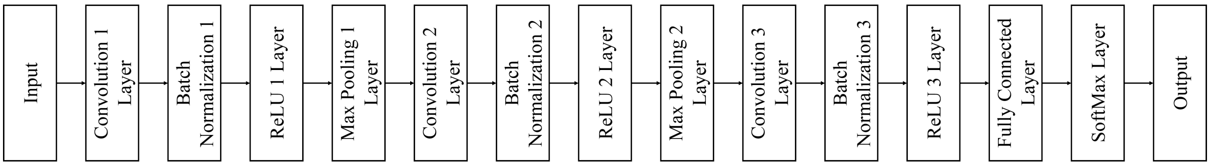

Figure 4. Block diagram of the proposed CNN architecture

The functionality of each these blocks we will see in the following sections.

4.1 Convolutional Layer

In this layer, 2 dimensional kernels are learned during the learning phase. The coefficients of these kernels are oriented such that it encodes most of the information from the input image. During the testing phase, these learned kernels are used to extract fine information from the validation dataset.

The important hyper-parameters associated with the CNN are the kernel size, number of kernels, stride and padding involved in the convolutional layers. These parameters greatly influence the performance of the system.

Convolution is the process of finding the sum of dot products between the image and the kernel. Later the kernel is moved by a finite step size over the image to find the next convolution sum. The output of the convolution will be another image of new dimension. In other words, convolution in 2 dimension is similar to spatial filtering. It is assumed that the kernel k is of size W x H, where W is the width and H is the height of the pixel, such that W = 2a+1 and H = 2b+1, a and b are positive integers. It is also assumed that the center of the kernel K(0,0) aligns over the pixel of the of image I at location (x,y). The convolution between an image I of size M x N and kernel K of size W x H is given as

$$C(x,y) = \sum_{u=-a}^{a}\sum_{v=-b}^{b} K(u,v)I(x+u, y+v) \tag{1}$$

The values of u and v are varied such that each pixel in K visits every pixel in I. The pixel by which we vary u and v is called the stride. During convolution operation the size of image is expected to have same number of rows and columns, i.e. M=N. Similarly, kernel is also expected to have same number of rows and columns, i.e. W=H.

Another technique which supports this convolution process is padding. It is introducing zeros around the border of the input image in order to maintain the aspect ratio of the resultant image after convolution. The size of the resultant image after convolution between image of size N x N x Z and kernel of size W x W x N_c as

$$[N, N, Z] * [W, W, N_c] = \left[\left\lfloor\frac{N + 2p - W}{s} + 1\right\rfloor, \left\lfloor\frac{N + 2p - W}{s} + 1\right\rfloor, N_f\right] \tag{2}$$

Where p is the number of padding, s is the stride length N_f is the number of kernel filters, N_c is the filter depth and Z is the image depth.

4.2 Batch Normalization Layer

The images from the convolutional layer is given as input to the normalization layer. Batch normalization is a transform applied to nullify the randomness or internal covariate shift in the convolutional weights. This helps to avoid vanishing or exploding gradient even at higher learning rate. It also helps to avoid overfitting. The introduction of batch normalization layer will improve the learning phase of the CNN. The statistical parameters like mean and variance obtained over a mini batch of the image dataset is used to normalize the images. After normalization the values are scaled by a factor γ and shifted by an offset of β.

4.3 Activation function Layer

The function of the activation layer is to perform mathematical operation on the incoming signal to identify the energy level of that node for that respective input. The images that come from the normalization layer is given as input to the activation layer. As there will be millions of for each sample data, the choice of activation function has to be done carefully. It helps to generalize the input images of a particular subject. The purpose of the activation layer is to introduce some non-linearity into the output so that the network get trained faster. In this architecture Rectified Linear Unit (ReLU) function is used as an activation function. The key benefit of this function it produces sparse outputs and avoids vanishing gradient. Because, gradient is more important in the learning process. If the gradient vanishes, the learning may not happen at the top layers. The output of the ReLU layer is given as

$$f(x) = \max(0,x) \tag{3}$$

There are also many other functions that belong to the ReLU family. The investigation on the performance of those function on CNN is beyond the scope of this paper.

4.4 Pooling Layer

Pooling layer is another essential block of the CNN. It helps to reduce sparsity and reduce the spatial size of the resultant images from previous layers. Since, we discard some information, it may badly affect the performance of the network. Hence, a proper pooling function has to be chosen to enhance the performance of the network. In this paper, max pooling is used. It highlights the key features present in the image and removes the low frequencies. The mask size and the stride determine the degree to which it reduces the spatial size of the image.

4.5 Fully connected layer

In fully connected layer all the neuron of the previous layer are connected and each neuron is connected with the neurons of the consecutive layer. Each node has a weight and bias associated with it. During the training phase, all these parameters are learned from the input image. It establishes the missing link between the feature extraction layers and the classifier layer. It represents the vital features of the corresponding input image. During the learning phase, these feature vectors are used to determine the loss function that will enable all other previous layers to train their respective weights.

4.6 SoftMax Layer

It is the final activation layer prior to classification layer. It is a normalized exponential function. This helps to enlarge the boundary between the respective classes into which the classifier needs to operate. Prior to this layer the outputs may be negative. After applying the softmax function the output gets mapped into the rage (0,1). Each component of the SoftMax layer will add up to one as it is normalizes into a probability distribution. The softmax function is given as

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{i=1}^{n} e^{z_i}} \tag{4}$$

Where n is maximum number of classes that the network is trained, z_i is the input value to the softmax classifier corresponding to the ith output.

4.7 Classification Layer

Classification layer computes the probability to which class does the input image belongs to. The widely used function in the multi-class classification task is cross-entropy. The cross entropy is calculated as

$$\text{Cross entropy of a class X} = -\, p(X)\log q(X) \tag{5}$$

Where p(X) is the probability of the class X in target and q(X) is the probability of the class X after prediction. If they are similar, it will result in more entropy.

4.8 Network Training

Training is an important phase in neural network-based classifiers. It will improve the recognition rate of the classifier. During the learning process the parameters of the individual layers are updated in unison with their respective class. It is all an automated process and does not need explicit programming. It usually takes in two phases, namely, the forward phase and the backward phase.

During the forward phase the input iris image is passed to move through the complete network. All the information gained during the forward phase is stored in the memory for further update during the backward phase. Each forward phase will be followed by a backward phase.

During the backward phase the cross-entropy loss is computed as gradient and backpropagated to update the respective weights. Each layer during the backward phase will receive a gradient of loss function with respect to its corresponding output and will return a loss function with respect to its corresponding input. In this paper two training methods are used to update the weights of the layers, namely stochastic gradient descent with momentum (SGDM) and adaptive moment estimation (ADAM).

4.8.1 SGDM

SGDM is an improved version of stochastic gradient descent. The momentum is the key success of this method to maintain it by incorporation it as a function of previous momentum and current gradient. The momentum helps to generate high velocity in the direction of consistent gradient. Local minima problem is overcome by using past momentum. This greatly avoids jitters. The weight update using SGDM is given as

$$v_n = \eta * v_n - \alpha \frac{d\sum_1^m L_m(k)}{dk} \tag{6a}$$

$$k_{n+1} = k_n + v_n \tag{6b}$$

Where $L_m(k)$ is the loss function of the mini batch, η is the momentum coefficient, v_n is the retained gradient and v_n is the velocity component. The parameter α is taken as 0.99 and initial value of η as 0.5.

4.8.2 ADAM

Adam is iterative approach to update the weights of the network proposed by Deiderik P. Kingma et. al in [31]. The key benefits of using this update method is that it is insensitive to noise on al large set of data, requires less memory and computationally efficient. It has a constant learning rate through the learning phase. The weight update using ADAM is given as

$$v_n = v_{n-1} * \beta_1 - (1 - \beta_1) * g_n \tag{7a}$$

$$s_n = s_{n-1} * \beta_2 - (1 - \beta_2) * g_n^2 \tag{7b}$$

$$k_{n+1} = k_n - \eta \frac{v_n}{\sqrt{s_n} + \varepsilon} * g_n \tag{7c}$$

Where η is the learning rate, v_n is the exponential average of gradients, s_n is the exponential average squares of gradient. g_n is the gradient at instance n. The values of parameters are taken as $\beta_1 = 0.9$, $\beta_2 = 0.99$ and $\varepsilon = 1^{-10}$ for better results.

5. Hybrid Convolutional Neural Network Framework

Hybrid classifier is another proposed network model to enhance the performance of the iris recognition. In hybrid convolutional neural network, the SoftMax and output layer is replaced by KNN and SVM classifier. The convolutional features drawn from the convolutional architectures is given as the feature vector to the KNN and SVM classifier. The blocks in the hybrid classifier is given in figure 5.

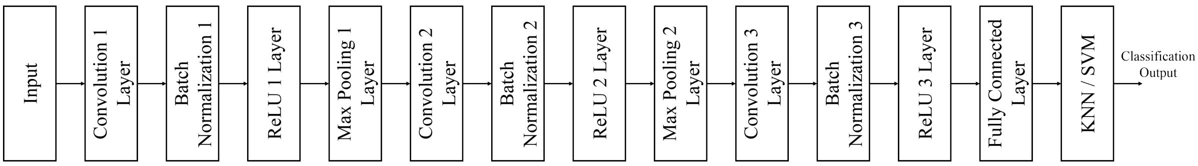

Figure 5. Framework of Hybrid Convolutional Neural Network

5.1 KNN Classifier

In this hybrid CNN classifier, the convolutional features are given as input to the KNN classifier. It is a supervised non-parametric machine learning algorithm. It does not have any training phase and uses all the data points for the classification process. The features of the validation dataset is measured for similarity measure with the training dataset. K is the number of nearest data points. The most frequent classes among the top K nearest points, is taken as the class of the test point data. Although it appears to be seamless and easy to implement, it requires huge computation and does not provide good results on large dataset.

5.2 SVM Classifier

SVM is a classical machine learning algorithm. It supports higher dimensional feature vector. The kernel trick, helps to handle non-linear input feature vector by transforming it into a higher dimensional space. It learns a maximum marginal hyperplane in multi-dimensional space iteratively to separate feature vectors among different classes. In this paper, linear kernel is used to convert the feature vectors linearly separable. One vs one method is used, where each class is treated as an individual set and a number of binary classifications are done to predict the output class. For a dataset having Q classes, the total number of binary classification that has to be done is given as Q x (Q-1)/2. After all the binary classification, the class with maximum number of positive predictions is chosen as the predicted class.

6. Methodology

The images from the IIT database are clipped such that it has equal number of rows and columns. The clipped images are also augmented to have big data to support deep learning. The augmented data set is split as training set and testing set. The training images are passed on to the deep learning model and the network parameters are changed using optimization methods like SGDM and ADAM over several epochs. After successful training, the testing images are passed on to the model and checked. The input iris images excite the corresponding output neuron in the output layer to which it belongs. From the outputs obtained, the results are calculated and performances are analysed.

7. Experiment Details

A series of CNN models is proposed and their results are discussed in this section to evaluate their performance. This deep learning architectures show an improved recognition results for iris recognition. The CNN architectures discussed in this paper are tested using MATLAB 2018a on iris dataset. All simulation are carried out on i5, 2.2 GHz CPU which has an integrated graphic processor intel HD 5500 and a dedicated 2GB NVIDIA GeForce 930M GPU.

7.1 Dataset

The iris images for the biometrics experiments were taken from one of the large-scale public database available in Biometrics Research Laboratory of IIT Delhi [34]. It provides iris images from captured using a private digital CMOS camera, JPC1000 manufactured by a Korean company called JIRIS. The iris images were captured under cooperative indoor conditions from around 224 subjects, including 176 males and 48 females. Table 2 shows the specification of the iris database.

Table 2. Specification of the IIT Delhi iris database

Number of subjects	Total number of images	Resolution	Spectrum	Format
224	1120	320x240	Near Infra-Red	.bmp

7.2 Image cropping output

The input image to the convolutional neural network, has to be a square image to support convolution operation. Hence the original image which of size 320x240 is cropped to a size of 240x240. The result of sample image is given in figure 6.

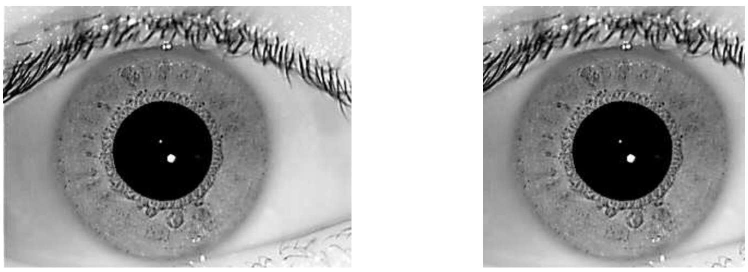

Figure 6. (a) original iris image of IIT Delhi dataset, (b) cropped image

7.3 Image Augmentation

The limited number of samples on each subject makes training of deep network more difficult. Either the number of iterations needed will be very huge, which will need a considerable amount of training time. The other issue is poor recognition performance because of over-fitting. Hence, it is important to have a large database. Image augmentation is process which can populate more data artificially from the original dataset. The augmentation done to improve the database size is given in table 3.

Table 3. Augmentation property

Name	**Augmentation factor**
Rotation	[-180 180]
X-translation	[-3 3]
Y-translation	[-3 3]

This creates multiple new datasets by performing operations like rotation and translation. It undergoes random rotations between -180 degree to 180 degree and a similar random translation of pixels along horizontal and vertical axis. Samples of augmented images is given in figure 7.

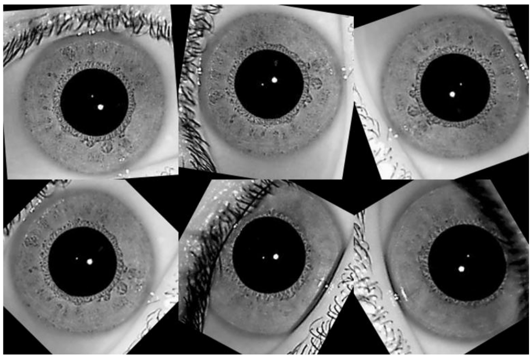

Figure 7. Preview of the augmented IIT Delhi iris database

7.4 Hyper-parameters

The configuration of hyper-parameters greatly influences the performance of the classifier. Each parameter has to be carefully chosen such that it enhances the classifier performance. The iris images from the training data set were trained for around 80 epochs. Each epoch has 14 iterations. Hence, a total of 1120 iterations were performed to train the weights in the convolutional layers. The mini batch size was set to 64 with a learning rate of 0.0003. Stochastic gradient descent with momentum and Adam optimizers were used to optimize the loss function during the training process. The network is trained and validated with same set of hyper-parameters. The hyper-parameters used in this CNN models is given in table 4.

Table 4. Hyper-parameters of the proposed architecture

<table>
<tr><th>Parameter</th><th colspan="3">Quantity</th></tr>
<tr><td>Epochs</td><td colspan="3">80</td></tr>
<tr><td>Iterations per epoch</td><td colspan="3">14</td></tr>
<tr><td>Total iterations</td><td colspan="3">1120</td></tr>
<tr><td>Mini Batch Size</td><td colspan="3">64</td></tr>
<tr><td>Optimizer</td><td colspan="3">SGDM / ADAM</td></tr>
<tr><td>Learning rate</td><td colspan="3">0.0003</td></tr>
<tr><td rowspan="2">Number of filters</td><td>Layer 1</td><td>Layer 2</td><td>Layer 3</td></tr>
<tr><td>8</td><td>16</td><td>32</td></tr>
</table>

Filter size	(a) 5x5	(b) 3x3
Stride	(a) 3x3	(b) 2x2

7.5 Kernel weights of the convolutional layers

During each iteration of the training process the kernel weights on the convolutional layer gets updated. These kernels help in extracting the features from the input iris image. The kernel in each layer has the capability of abstracting certain degree of features present in the image. Figure 8 shows the distribution of kernel weights of the first convolutional layer. It indicates that these kernels help in extracting the contour-based features from the iris image. The kernels towards the end of the network are shown in figure 9. It helps in extracting more complex information from the input image with the combination of kernels from previous layers.

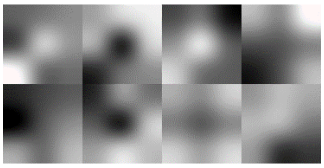

Figure 8. Learned kernel weights of the first convolutional layer

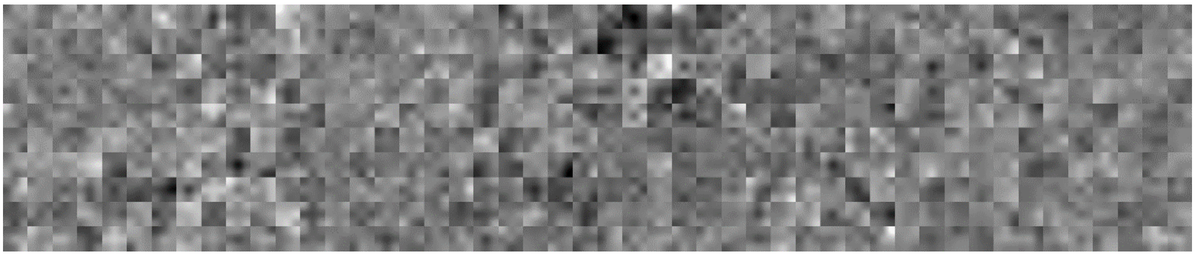

Figure 9. Learned kernel weights of the third convolutional layer

7.6 Learned features of intermediate layers

After the training process, the test image is passed on through the network and undergoes a series of operation. This helps is extracting the relevant information from test image. Figure 10 shows the images of output of first convolutional layer.

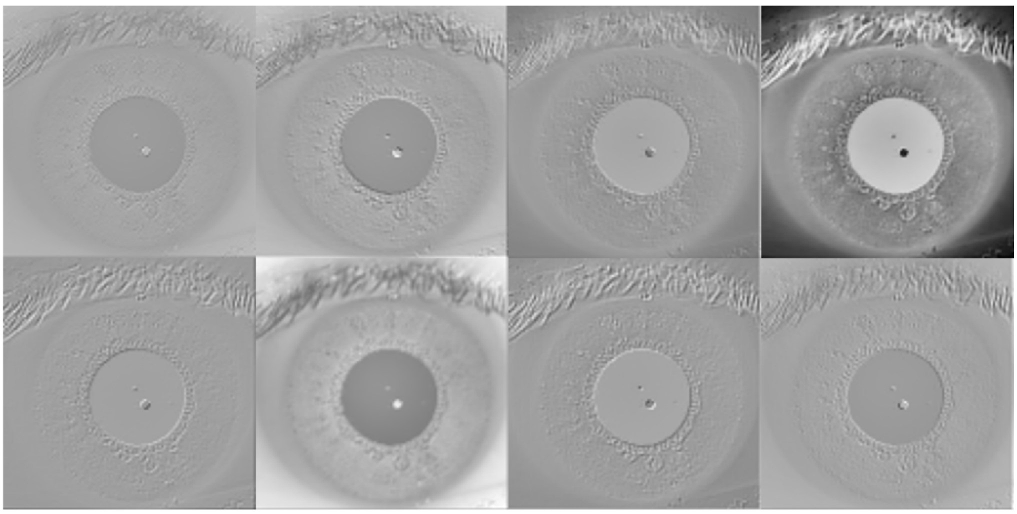

Figure 10. Convoluted features from the first convolutional layer

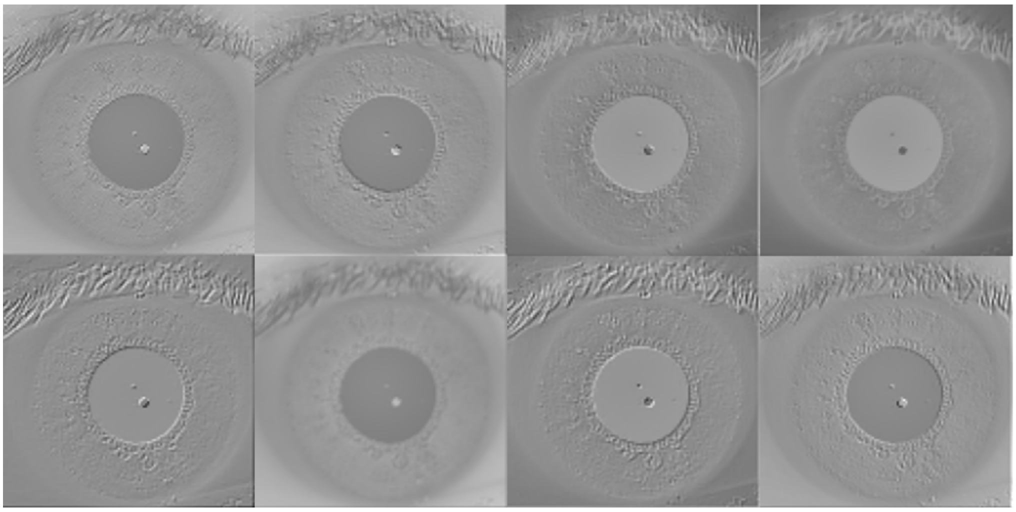

Figure 11. Batch normalization output of the first normalization layer

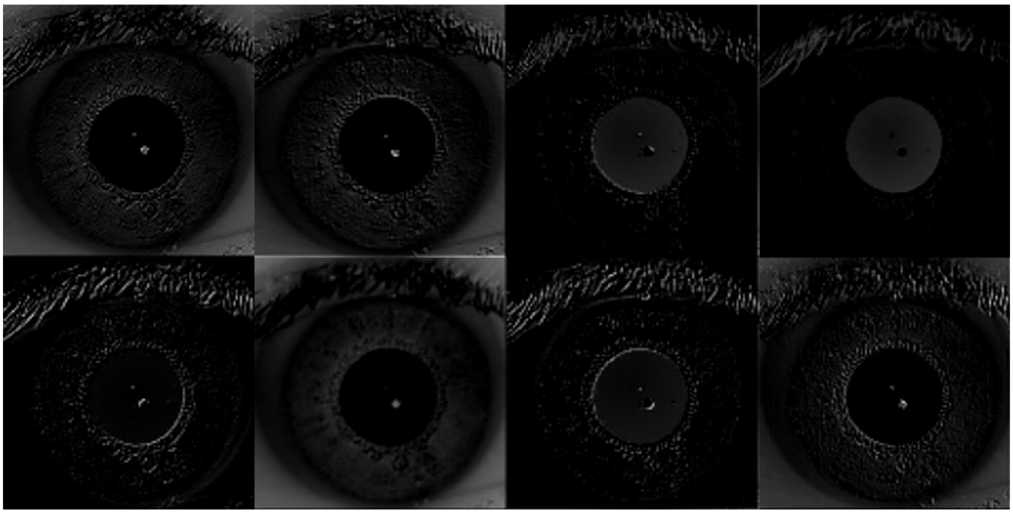

Figure 12. Activations layer output of first ReLU layer

After convolution the image batch normalized and the resultant figures of the first normalization layer is given in figure 11. The normalized images are given to the activation layer. The results of the first activation layer is given in figure 12. The activation layer results show the depth of information captured by the convolution layer, being highlighted by the ReLU activation function.

7.7 Performance Analysis of the CNN architecture

Firstly, the experiment results of CNN architecture is analysed in this section. The training is done using SGDM and then later by ADAM. The kernel size in the convolution layers is varied as 3x3 and 5x5 to extract the features from the input image.

The training accuracy of the different combination of kernel size and training method is given in figure. The models which appear to be trained better during the starting period does not show the same performance after the end of the training phase. Figure 13 shows the training performance of the different CNN models.

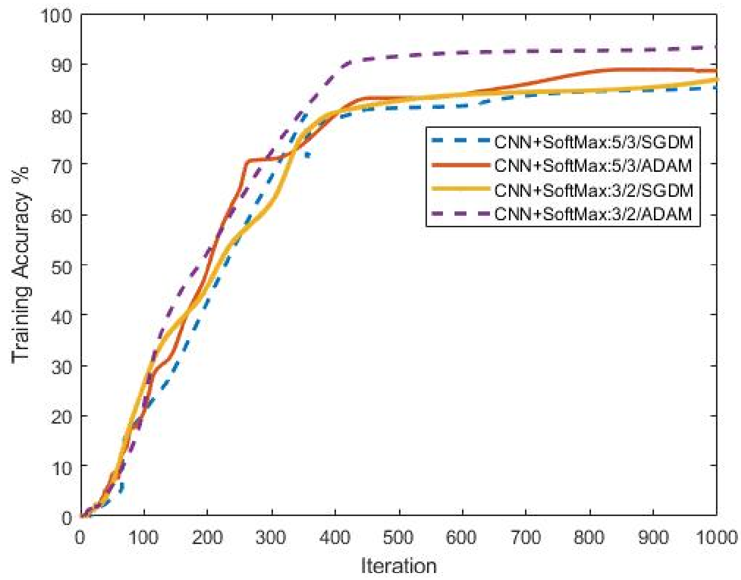

Figure 13. Training Accuracy of CNN+SoftMax models

The network is trained for around 1120 iterations. The performance of the network is tested during each iteration with the validation dataset also. Figure shows the validation accuracy of the CNN with SoftMax models. The network performance during each iteration improves because of the weight update. The larger weight update of the kernels happens during the first 32 epochs of the training phase.

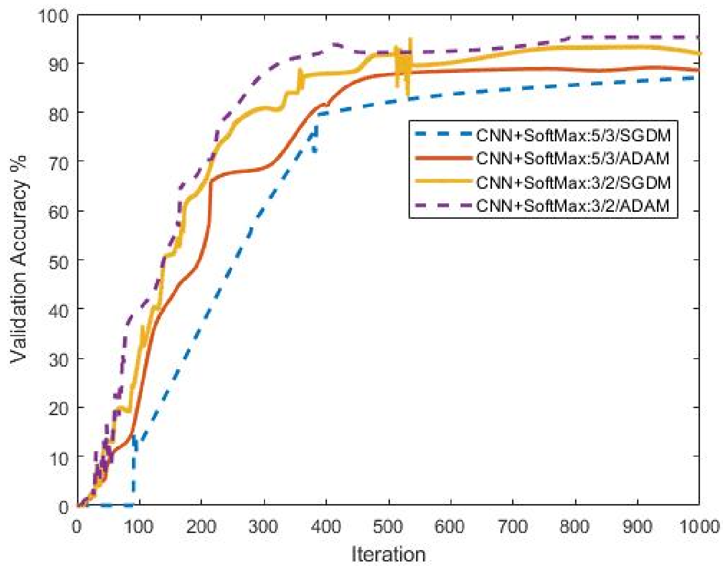

Figure 14. Validation Accuracy of the CNN+SoftMax models

The performance of the network on the validation dataset is shown in the figure 14. It can be observed that the model having kernel size 3x3, stride 2x2 and training method ADAM performs better than other methods. The reason for this is, that smaller kernels and smaller stride will capture large number of features from the iris image. Hence, we get more abstract information from the input iris image. A detailed performance metric is tabulated in table 5.

Table 5. Performance of the CNN+SoftMax models

	Training		Validation		
Model	**Accuracy (%)**	**Loss**	**Accuracy (%)**	**Loss**	**Training Method**
CNN+SoftMax (KS-5, S-3, P-0)	89.21	1.5922	87.56	1.7459	SGDM
CNN+SoftMax (KS-5, S-3, P-0)	93.75	0.9159	90.60	1.2621	ADAM
CNN+SoftMax (KS-3, S-2, P-0)	94.72	0.8058	91.02	1.0257	SGDM
CNN+SoftMax (KS-3, S-2, P-0)	96.43	0.1735	95.16	0.6602	ADAM

The ADAM optimizer is more versatile as it updates weight more efficiently based on adaptive moment estimation. It also requires very less memory and can speed up the learning process. Hence, it is more suited for large datasets. It is insensitive to noise and sparse gradients. Thus, it can be seen that ADAM optimizer provides better recognition accuracy.

Table 6. Accuracy improvement of the CNN+SoftMax models 1 and 2

Model	Parameters	Validation accuracy %	% Improvement
Model 1	KS – 5, S – 3, Training: SGDM	87.56	3.47
Model 2	KS – 5, S – 3, Training: ADAM	90.60	

An improvement in accuracy of 3.47 % by the model 2 of CNN with softmax classifier is noted from table 6. This again shows that tractability of the ADAM method in training the weights of the proposed CNN architecture.

Table 7. Accuracy improvement of the CNN+SoftMax models 3 and 4

Model	Parameters	Validation accuracy %	% Improvement
Model 3	KS – 3, S – 2, Training: SGDM	91.02	4.54
Model 4	KS – 3, S – 2, Training: ADAM	95.16	

An improvement in accuracy of 4.54 % by the model 4 of CNN with softmax classifier is noted from table 7. On comparison with the results from table 6 we can see the influence of the filter kernel size in the system performance. A smaller filter size of 3x3 and stride 2 browses through larger area and grabs more feature from the input image and helps in training the CNN architecture better. Hence the performance measures of model 4 is taken as reference for comparison with other hybrid structures.

7.8 Evaluation of Hybrid CNN architecture

The proposed architecture requires further improvement of accuracy. Hence a hybrid approach is used with the convolutional features. Here, the features extracted from the convolutional layers are given to classical statistical classifiers like KNN and SVM. The K value of the KNN classifier is chosen and 5 and SVM is a multi-class linear SVM. The receiver operating characteristic curve of the proposed models is shown in figure 15.

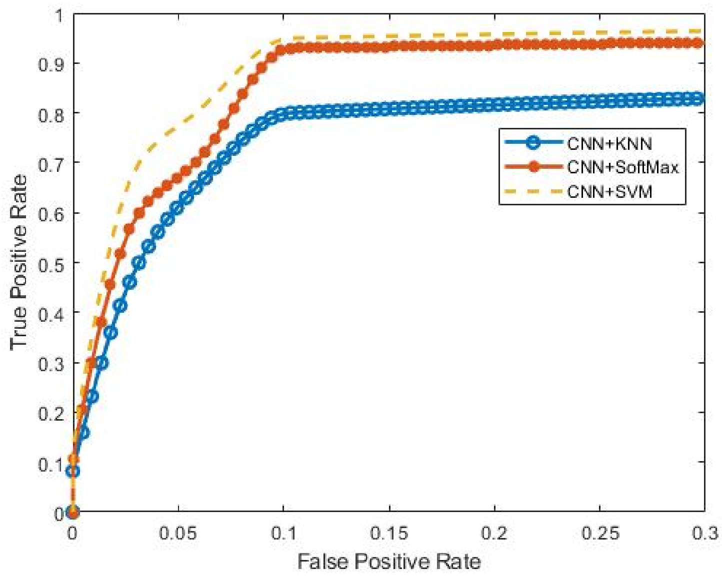

Figure 15. ROC curve of the proposed classifiers

The performance of the CNN with KNN is not satisfactory. This is because the KNN takes all the data in the samples space and classification becomes difficult for larger datasets. On the other hand, the hybrid classifier CNN with SVM performs well compared to the later. It also outperforms our proposed CNN architecture with softmax layer. The area under the curve and validation accuracy are tabulated in table 8.

Table 8. Performance analysis of proposed methods

Model	AUC	FAR	FRR	ERR	Validation Accuracy (%)
CNN+KNN	0.71	0.46	0.09	0.27	86
CNN+SoftMax (FS-3, S-2, P-0)	0.84	0.19	0.03	0.11	95.16
CNN+SVM	0.86	0.11	0.01	0.06	97.8

More the area enclosed by the ROC curve, better is the recognition accuracy of the model. It can be seen from the above table that the SVM based hybrid classifier provides an improved recognition accuracy.

7.9 Comparative Analysis

To validate the robustness of our algorithm, we have compared the results with other typical deep learning architectures used for iris recognition. From the table 9, it can be said that the proposed method gives better accuracies compared to other deep learning architectures.

Table 9. Comparison of the proposed methods

Model	**Validation Accuracy (%)**
Tiangming Zhao et al. [10], 12 Layer CNN	79.55
Ming Liu et al. [7], Fuzzy CNN	85.8
O. Oyedotun et al. [33], Deep Belief Network	93.67
CNN+KNN	86

CNN+SoftMax (FS-3, S-2, P-0)	95.16
CNN+SVM	97.8

The proposed CNN architecture performs better than other deep learning methods and can be adapted for iris recognition. Moreover, the proposed hybrid classifier CNN with SVM provides better performance than other methods. This is because of the classifying ability of the SVM in higher dimensional space. Although CNN with KNN is a hybrid classifier, the performance does not look appreciable. This is because of the inability of the KNN to handle large datasets and it is sensitive to noise introduced as a result of image augmentation.

8. Conclusion

The advancement of several computing methods and efficient multi-core processors has enabled iris recognition as a viable tool for authentication. The latest technology of deep learning which has a huge potential if properly used has attracted many people to adapt this technology. The data augmentation has helped in generating large datasets on individual subjects, which caused large difficulty implementing better deep learning models for iris recognition. In this work, deep convolutional network and a hybrid convolutional network has been proposed. In terms of filter size, the lowest order filter of size 3x3 is able to encode maximum possible key features from the region of interest. This is because, lower order filter will take many strides to complete the entire image span which enables getting a bigger feature space. The ADAM optimizer which makes use of adaptive momentum in generating gradient has enabled better learning process than the SGDM. However, compared to the raw CNN architecture, the hybrid CNN with SVM performs better with an accuracy of 97.8%. This is because of the ability of the SVM to handle features in multi-dimensional space. It has also avoided handcrafted segmentation process, as employed in several deep learning techniques and has still provided comparable performance. This work can be extended further, by exploring the other learning optimizers and by introducing more layers. Other hyper-parameter tuning can also be studied. It can provide chance for further improvement of the classifier performance. It can be noted that the hybrid CNN with SVM performs better than the traditional CNN because of the multidimensional adaptability of the SVM rather than the fully-connected layer. However, the deficiencies of this method is as follows. The performance measures of the proposed methods are limited to the IIT Delhi database and he performance of the network may fail for other iris databases. The other limitation is that the too much of computations involved with the hybrid structure. Even though the convolutional features are more distinct they consume more computation with deep structures and large data samples.

The authors declare no conflict of interest.

References

1. Daugman, J.G. High Confidence Visual Recognition of Persons by a Test of Statistical Independence. *IEE Trans. on Pattern Analysis and Machine Intelligence*. 1993, 15(11), 1148-1161. https://doi.org/10.1109/34.244676
2. Hu, Y.; Sirlantzis, K.; Howells, G. Optimal Generation of Iris Codes for Iris Recognition. *IEEE Trans.Inform.Forensic Secur*. **2017**, *12* (1), 157–171. https://doi.org/10.1109/TIFS.2016.2606083.
3. Pillai, J. K.; Puertas, M.; Chellappa, R. Cross-Sensor Iris Recognition through Kernel Learning. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *36* (1), 73–85. https://doi.org/10.1109/TPAMI.2013.98.
4. Oktiana, M.; Horiuchi, T.; Hirai, K.; Saddami, K.; Arnia, F.; Away, Y.; Munadi, K. Cross-Spectral Iris Recognition Using Phase-Based Matching and Homomorphic Filtering. *Heliyon* **2020**, *6* (2), e03407. https://doi.org/10.1016/j.heliyon.2020.e03407.

5. Oktiana, M.; Saddami, K.; Arnia, F.; Away, Y.; Hirai, K.; Horiuchi, T.; Munadi, K. Advances in Cross-Spectral Iris Recognition Using Integrated Gradientface-Based Normalization. *IEEE Access* **2019**, *7*, 130484–130494. https://doi.org/10.1109/ACCESS.2019.2939326.
6. Rakvic, R.; Broussard, R.; Ngo, H. Energy Efficient Iris Recognition With Graphics Processing Units. *IEEE Access* **2016**, *4*, 2831–2839. https://doi.org/10.1109/ACCESS.2016.2571747.
7. Liu, M.; Zhou, Z.; Shang, P.; Xu, D. Fuzzified Image Enhancement for Deep Learning in Iris Recognition. *IEEE Trans. Fuzzy Syst.* **2020**, *28* (1), 92–99. https://doi.org/10.1109/TFUZZ.2019.2912576.
8. Ribeiro, E.; Uhl, A.; Alonso-Fernandez, F. Iris Super-Resolution Using CNNs: Is Photo-Realism Important to Iris Recognition? *IET Biometrics* **2019**, *8* (1), 69–78. https://doi.org/10.1049/iet-bmt.2018.5146.
9. Baqar, M.; Ghani, A.; Aftab, A.; Arbab, S.; Yasin, S. Deep Belief Networks for Iris Recognition Based on Contour Detection. *International Conference on Open Source Systems and Technologies* **2016**, 6.
10. Zhao, T.; Liu, Y.; Huo, G.; Zhu, X. A Deep Learning Iris Recognition Method Based on Capsule Network Architecture. *IEEE Access* **2019**, *7*, 49691–49701. https://doi.org/10.1109/ACCESS.2019.2911056.
11. Lee, M. B.; Kim, Y. H.; Park, K. R. Conditional Generative Adversarial Network- Based Data Augmentation for Enhancement of Iris Recognition Accuracy. *IEEE Access* **2019**, *7*, 122134–122152. https://doi.org/10.1109/ACCESS.2019.2937809.
12. Benalcazar, D. P.; Zambrano, J. E.; Bastias, D.; Perez, C. A.; Bowyer, K. W. A 3D Iris Scanner From a Single Image Using Convolutional Neural Networks. *IEEE Access* **2020**, *8*, 98584–98599. https://doi.org/10.1109/ACCESS.2020.2996563.
13. Wang, K.; Kumar, A. Cross-Spectral Iris Recognition Using CNN and Supervised Discrete Hashing. *Pattern Recognition* **2019**, *86*, 85–98. https://doi.org/10.1016/j.patcog.2018.08.010.
14. Zhao, Z.; Kumar, A. A Deep Learning Based Unified Framework to Detect, Segment and Recognize Irises Using Spatially Corresponding Features. *Pattern Recognition* **2019**, *93*, 546–557. https://doi.org/10.1016/j.patcog.2019.04.010.
15. Raja, K. B.; Raghavendra, R.; Venkatesh, S.; Busch, C. Multi-Patch Deep Sparse Histograms for Iris Recognition in Visible Spectrum Using Collaborative Subspace for Robust Verification. *Pattern Recognition Letters* **2017**, *91*, 27–36. https://doi.org/10.1016/j.patrec.2016.12.025.
16. Marra, F.; Poggi, G.; Sansone, C.; Verdoliva, L. A Deep Learning Approach for Iris Sensor Model Identification. *Pattern Recognition Letters* **2018**, *113*, 46–53. https://doi.org/10.1016/j.patrec.2017.04.010.
17. Nguyen, K.; Fookes, C.; Ross, A.; Sridharan, S. Iris Recognition With Off-the-Shelf CNN Features: A Deep Learning Perspective. *IEEE Access* **2018**, *6*, 18848–18855. https://doi.org/10.1109/ACCESS.2017.2784352.
18. Liu, N.; Zhang, M.; Li, H.; Sun, Z.; Tan, T. DeepIris: Learning Pairwise Filter Bank for Heterogeneous Iris Verification. *Pattern Recognition Letters* **2016**, *82*, 154–161. https://doi.org/10.1016/j.patrec.2015.09.016.
19. Chen, Y.; Wu, C.; Wang, Y. *T* -Center: A Novel Feature Extraction Approach Towards Large-Scale Iris Recognition. *IEEE Access* **2020**, *8*, 32365–32375. https://doi.org/10.1109/ACCESS.2020.2973433.
20. Nguyen, K.; Fookes, C.; Sridharan, S. Constrained Design of Deep Iris Networks. *IEEE Trans. on Image Process.* **2020**, *29*, 7166–7175. https://doi.org/10.1109/TIP.2020.2999211.
21. Hu, Q.; Yin, S.; Ni, H.; Huang, Y. An End to End Deep Neural Network for Iris Recognition. *Procedia Computer Science* **2020**, *174*, 505–517. https://doi.org/10.1016/j.procs.2020.06.118.
22. Minaee, S.; Abdolrashidiy, A.; Wang, Y. An Experimental Study of Deep Convolutional Features for Iris Recognition. In *2016 IEEE Signal Processing in Medicine and Biology Symposium (SPMB)*; IEEE: Philadelphia, PA, USA, 2016; pp 1–6. https://doi.org/10.1109/SPMB.2016.7846859.
23. Minaee, S.; Abdolrashidi, A. DeepIris: Iris Recognition Using A Deep Learning Approach. *arXiv:1907.09380 [cs]* **2019**.

24. Al-Waisy, A. S.; Qahwaji, R.; Ipson, S.; Al-Fahdawi, S.; Nagem, T. A. M. A Multi-Biometric Iris Recognition System Based on a Deep Learning Approach. *Pattern Anal Applic* **2018**, *21* (3), 783–802. https://doi.org/10.1007/s10044-017-0656-1.
25. Umer, S.; Sardar, A.; Dhara, B. C.; Rout, R. K.; Pandey, H. M. Person Identification Using Fusion of Iris and Periocular Deep Features. *Neural Networks* **2020**, *122*, 407–419. https://doi.org/10.1016/j.neunet.2019.11.009.
26. Menotti, D.; Chiachia, G.; Pinto, A.; Robson Schwartz, W.; Pedrini, H.; Xavier Falcao, A.; Rocha, A. Deep Representations for Iris, Face, and Fingerprint Spoofing Detection. *IEEE Trans.Inform.Forensic Secur.* **2015**, *10* (4), 864–879. https://doi.org/10.1109/TIFS.2015.2398817.
27. Liu, X.; Bai, Y.; Luo, Y.; Yang, Z.; Liu, Y. Iris Recognition in Visible Spectrum Based on Multi-Layer Analogous Convolution and Collaborative Representation. *Pattern Recognition Letters* **2019**, *117*, 66–73. https://doi.org/10.1016/j.patrec.2018.12.003.
28. Wang, C.; Muhammad, J.; Wang, Y.; He, Z.; Sun, Z. Towards Complete and Accurate Iris Segmentation Using Deep Multi-Task Attention Network for Non-Cooperative Iris Recognition. *IEEE Trans.Inform.Forensic Secur.* **2020**, *15*, 2944–2959. https://doi.org/10.1109/TIFS.2020.2980791.
29. LeCun, Y.; Boytou, L.; Bengio, Y.; Haffner, P. Gradient Based Learning Applied to Document Recognition. *Proceedings of the IEEE.* **1998**, 86(11), 2278-2324. https://doi.org/10.1109/5.726791
30. Kavukcuoglu, K.; Sermanet, P.; Boureau, Y.; Gregor, K.; Mathieu, M.; Cun, Y. L. Learning Convolutional Feature Hierarchies for Visual Recognition. *NIPS'10: Proceedings of the 23rd International Conference on Neural Information Processing Systems.* ***2010,*** 1, 1090-1098. https://dl.acm.org/doi/10.5555/2997189.2997311
31. Qian, N. On the Momentum Term in Gradient Descent Learning Algorithms. *Neural Networks* **1999**, *12* (1), 145–151. https://doi.org/10.1016/S0893-6080(98)00116-6.
32. Kingma, D. P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 CORR,* **2015**. 1412.6980. https://arxiv.org/abs/1412.6980
33. Oyedotun, O.; Khashman, A. Iris nevus diagnosis: convolutional neural network and deep belief network. *Turkish Journal of Electrical Engineering & Computer Sciences.* **2017**, 25(2017), 1106-1115. doi:10.3906/elk-1507-190
34. IIT Delhi Iris Database version 1.0, UpToDate. Retrieved July 10, 2021 from http://www4.comp.polyu.edu.hk/~csajaykr/IITD/Database_Iris.htm
35. Srivastva, R.; Singh, A.; Singh, Y.N. PlexNet: A fast and robust ECG biometric system for human recognition. *Information Sciences*. **2021**, 558(2021), 208-228. https://doi.org/10.1016/j.ins.2021.01.001.
36. Maiorana, E Deep Learning for EEG-based Biometric Recognition. NeuroComputing. **2020**, 410, 374-386. https://doi.org/10.1016/j.neucom.2020.06.009
37. B. Ciocoiu; N. Cleju. Off-Person ECG Biometrics Using Spatial Representations and Convolutional Neural Networks. *IEEE Access*, **2020**, 8, 218966-218981. doi: 10.1109/ACCESS.2020.3042547
38. J. Malik; A. Elhayek; S. Guha; S. Ahmed: A. Gillani; D. Stricker. DeepAirSig: End-to-End Deep Learning Based in-Air Signature Verification. *IEEE Access*, **2020**, 8, 195832-195843. doi: 10.1109/ACCESS.2020.3033848
39. Narsi Reddy; Ajita Rattani; Reza Derakhshani. Generalizable deep features for ocular biometrics. *Image and Vision Computing*. **2020**, 103, 103996. https://doi.org/10.1016/j.imavis.2020.103996
40. T. Sudhakar; M. Gavrilova. Cancelable Biometrics Using Deep Learning as a Cloud Service. *IEEE Access*, **2020**, 8, 112932-112943. doi: 10.1109/ACCESS.2020.3003869