

A Convolutional Neural Network for Iris Recognition

Wael Alnahari

College of Computing and Information Technology, Department of Information Systems, University of Bisha, Bisha, Asir, Saudi Arabia

Corresponding author

Wael Alnahari, College of Computing and Information Technology, Department of Information Systems, University of Bisha, Asir, Saudi Arabia.

Submitted: 25 Aug 2022; Accepted: 30 Aug 2022; Published: 24 Sep 2022

Citation: Wael Alnahari. (2022). A convolutional neural network for iris recognition. *J Robot Auto Res*, 3(3),254-257.

Abstract

In this paper, I proposed an iris recognition system by using deep learning via convolutional neural networks (CNN). Although CNN is used for machine learning, the recognition is achieved by building a non-trained CNN network with multiple layers. The main objective of the code is recognizing test pictures' category (aka person name) with high accuracy rate after having extracted enough features from training pictures of the same category which are obtained from a dataset that I added to the code. I used IITD iris dataset which included 10 iris pictures for 223 people. The pictures were divided into 3 pictures for testing and 7 pictures for training. The categories are from 001 to 223. Since the number of pictures for training is low, in order to enhance the recognition accuracy of the network, I used five sets of layers and altered nine parameters of sgdm training option. The code concluded an accuracy rate of 97.46% and the time elapsed was 10 minutes and 30 seconds.

Keywords: Biometrics, Biometric Systems, Machine Learning, Deep Learning, Recognition, IRIS

Introduction

Introduction to CNN

Convolutional neural networks (CNN) is a deep learning algorithm used to help machines categorize objects. CNN is used as a computer vision detection on images. CNN is designed to mimic human brain hence the name neural. CNN is used to treat visual inputs and determine what the input represents based on the inputs features. CNN uses a set of multiple layers to analyze the visual inputs and to determine what are the inputs' distinguishing features based on probability of repetition. CNN is useful for Artificial Intelligence (AI) such as self-driving cars to help the cars determine what the objects appearing in their cameras represent. In general, CNN is two parts: convolution layer and fully connected layer. The convolution layer is a single layer used to extract a specific feature from the visual input which can be repeated as many times as desired. The fully connected layer connects all convolution layers to one layer or representation to combine the features [1].

CNN Architecture

In order to create a CNN system, one needs a dataset to be used for training and another dataset to be used for testing. The training dataset is categorized according to the desirable categories. The set of data for training is usually very large compared to the test data but it is not a requirement; however, the bigger the number of data for training the better the quality of the result. The machine uses the training images or train dataset to find distinguishing features of each category. The features are extracted with the help of layers.

A layer in this case represents a specific process which transforms the image into another shape, size, color, or look which is done in some pixels that are extremely smaller than the actual image. Many layers are applied for images of the same category which are then stored in the network of the CNN. The stored features are based on the probability of repetition of a number of features among a category from the training dataset. Afterward, when an image is tested, the same layers will be applied to it and then are processed to determine which category is most similar to its features.

Implementation of Cnn for Biometric Recognition

Data Set

Description of the Iitd Iris Image Database Version 1.0

This iris image database mainly consists of the iris images collected from the students and staff at IIT Delhi, India. This database has been acquired in the Biometrics Research Laboratory during January - July 2007 using JIRIS, JPC1000, digital CMOS camera. The acquired images were saved in bitmap format. The database of 2240 images is acquired from 224 different users and made available freely to the researchers. All the subjects in the database are in the age group 14-55 years comprising of 176 males and 48 females. The resolution of these images is 320 x 240 pixels and all these images were acquired in the indoor environment. All the images in the database were acquired from the volunteers who were not paid or provided any honorarium. The images were acquired using an automated program that requires users to present their

eyes in a sequence until ten images are registered. Organization of Database.

The acquired database is saved in 224 folders, each corresponding to 224 subjects. Majority of images were acquired from the left eyes while the rest images were acquired from right eye. Now the database has a label 'L' or 'R' which designates left or right eye. There are 1288 images from 224 subject that are from left eyes while the rest images from 211 subjects are from right eyes. Except folders 1-13, 27, 55 and 65 all other folders have five left and 5 right eye images. (**appended on 20-04- 2016**). Usage of Database. This database is only available for research and non commercial purposes. Commercial distribution or any act related to commercial use of this database is strictly prohibited. Kindly acknowledge all the publicly available publications/work employing this database with the following

Acknowledgment

“Portions of the work tested on the IITD Iris Database version 1.0” A citation to “IIT Delhi Iris Database version 1.0, http://web.iitd.ac.in/~biometrics/Database_Iris.htm Related Publication:

- A. Sample of the dataset pictures
- B. Layers
- C. CNN is composed of different types of layers as follows: [1]

Convolutional layer: in this layer the computer scans the visual input few pixels at a time to determine a feature for each few pixels then continue to the next few pixels until the whole visual input is finalized.

Pooling layer (down sampling): in this layer the features of the previous layer is down sampled after having stored the most important features.

Fully connected layer: in this layer all the previous layers are connected together to build a conclusion of all important features.

The used layers of our CNN are: layers = [image Input Layer ([x y z]); convolution 2d Layer (3,8,'Padding','same') batch Normalization Layer relu Layer (); Max Pooling 2d Layer (5,'Stride',2) convolution 2d Layer (3,16,'Padding','same') batch Normalization Layer Relu Layer (); average Pooling 2d Layer (5,'Stride',2) convolution 2d Layer (3,16,'Padding','same') batch Normalization Layer relu Layer ();max Pooling 2d Layer (5,'Stride',2)convolution 2d Layer (3,32,'Padding','same') batch Normalization Layer relu Layer (); average Pooling 2d Layer (5,'Stride',2) convolution 2d Layer (3,32,'Padding','same') batch Normalization Layer relu Layer (); fully Connected Layer (223,'BiasLearnRateFactor',2); soft max Layer classification Layer];

Training Options

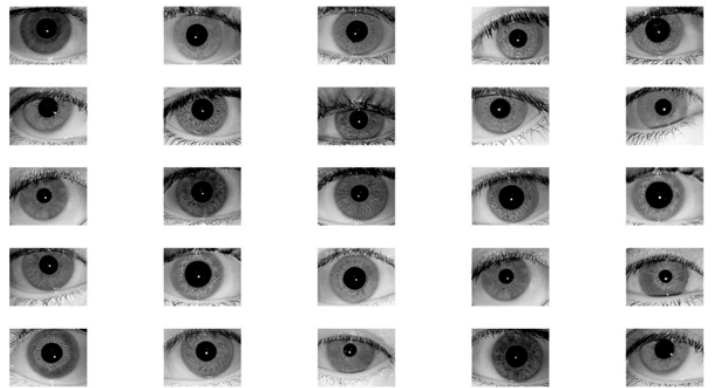
options = training Options ('sgdm', ...'Initial Learn Rate', 0.0001, ...'Validation Data',imsd Validation, ... 'ValidationFrequency',30, ... 'Shuffle',' every-epoch', ... 'Max Epochs', 10,...'Mini Batch Size', 8,... 'Validation Frequency',50,...'Learn Rate Schedule', 'piecewise',... 'LearnRateDropFactor',0.05,... 'LearnRateDropPeriod',60, ...'Execution Environment',' parallel',... 'Verbose', true,

'Plots',' training-progress');

Results

A. Training Trails

B. Got an accuracy of 54 56% using the following specifications for layers and options: ? Define Network Architecture layers = [image Input Layer ([x y z]); convolution 2d Layer (3,8,'Padding','same') Batch Normalization Layer relu Layer max Pooling 2d Layer (2,'Stride',2) convolution 2d Layer (3,16,'Padding','same') batch Normalization Layer relu Layer max Pooling 2d Layer (2,'Stride',2) convolution 2d Layer(3,32,'Padding','same')batch Normalization Layer relu Layer fully Connected Layer (size(categories,2),' Bias Learn Rate Factor',2); soft max Layer classification Layer]; %% Specify Training Options options = training Options ('sgdm', ... 'Initial Learn Rate', 0.01, ... 'Validation Data',imsd Validation, ... 'ValidationFrequency',35, ...



Sample of Dataset Images

Figure 1.

'Shuffle', 'every-epoch', ...'Max Epochs', 15, ...'Mini Batch Size', 8, ... 'Learn Rate Schedule',' piecewise', ... 'LearnRateDropFactor',0.05,...'LearnRateDropPeriod',60, ... 'Execution Environment',' parallel', ... 'Verbose', true, 'Plots',' training-progress'); When I adjusted 'Initial Learn Rate' to 0.001 instead of 0.01 I got an accuracy rate of 92.83% When I used the same training options as above but the max Epoch to 6 in order to compare the results with the previous trial and changed the layers. I managed to reduce the elapsed time to 00:10:41 with 96.41% accuracy? Define Network Architecture layers = [image Input Layer ([x y z]); convolution 2d Layer (3, 8,'Padding','same') batch Normalization Layer relu Layer ();Max Pooling 2d Layer (5,'Stride',2)convolution 2d Layer (3,16,'Padding','same') batch Normalization Layer relu Layer ();average Pooling 2d Layer (5,'Stride',2)convolution 2d Layer(3,16,'Padding','same') batch Normalization Layer relu Layer (); max Pooling 2d Layer (5,'Stride',2)convolution 2d Layer (3,32,'Padding','same') batch Normalization Layer relu Layer(); average Pooling 2d Layer (5,'Stride',2) convolution 2d Layer (3,32,'Padding','same') batch Normalization Layer relu Layer (); fully Connected Layer (size(categories,2),'Bias Learn Rate Factor',2); Soft max Layer classification Layer]; %% Specify Training Options options = training Options ('sgdm', ... 'Initial

Learn Rate', 0.001, ... 'Validation Data', imds Validation, ... 'Validation Frequency', 35, ... 'Shuffle', 'every epoch', ... 'MaxEpochs', 6, ... 'Mini Batch Size', 8, ... 'Learn Rate Schedule', 'piecewise', ... 'Learn Rate Drop Factor', 0.05, ... 'Learn Rate Drop Period', 60, ... 'Execution Environment', 'parallel', ... 'Verbose', true, 'Plots', 'training-progress'); When I changed 'Initial Learn Rate' to 0.0001 and 'Validation Frequency' to 35 I ended up with a lower accuracy 86.70% When I changed 'InitialLearnRate' to 0.00001 I ended up with an extremely low accuracy of 15.55%

Final Accuracy

Using the following layer and training options: layers = [image Input Layer ([x y z]); convolution 2d Layer(3,8,'Padding','same') batch Normalization Layer relu Layer(); Max Pooling 2d Layer (5,'Stride',2) convolution 2d Layer (3,16,'Padding','same') batch Normalization Layer relu Layer(); average Pooling 2d Layer (5,'Stride',2) convolution 2d Layer (3,16,'Padding','same') batch Normalization Layer relu Layer (); maxPooling2dLayer (5,'Stride',2) convolution 2d Layer (3,32,'Padding','same') batch Normalization Layer relu Layer(); Average Pooling 2d Layer (5,'Stride',2) convolution 2d Layer (3,32,'Padding','same') batch Normalization Layer relu Layer(); fully Connected Layer (size(categories,2),'Bias Learn Rate Factor',2); soft max Layer classification Layer]; %% Specify Training Options options = training Options('sgdm', ... 'Initial Learn Rate', 0.001, ... 'Validation Data', imds Validation, ... 'Validation Frequency', 50, ... 'Shuffle', 'every-epoch', ... 'Max Epochs', 10, ... 'Mini Batch Size', 8, ... 'Learn Rate Schedule', 'piecewise', ... 'Learn Rate Drop Factor', 0.5, ... 'Learn Rate Drop Period', 50, ... 'Execution Environment', 'parallel', ... 'Verbose', true, 'Plots', 'training-progress');

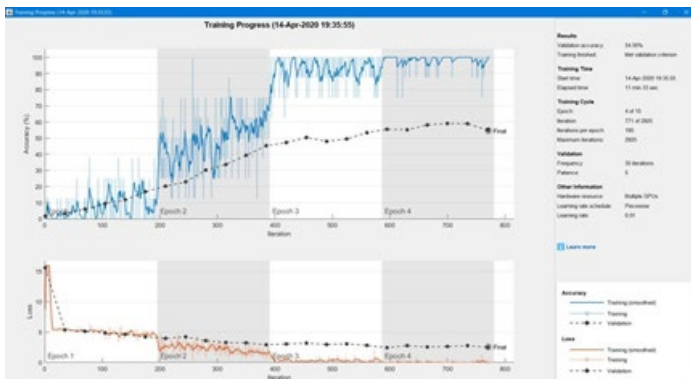


Figure 2.

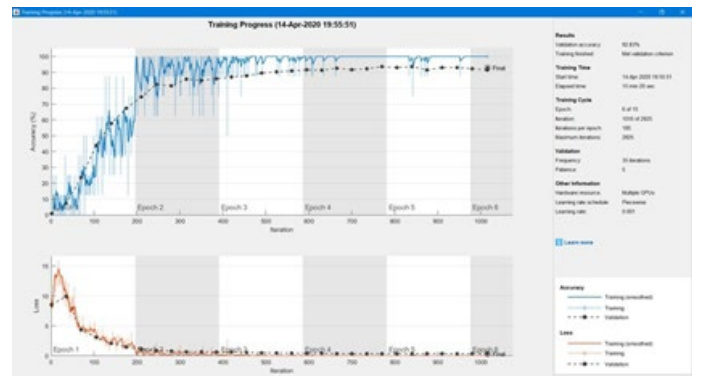


Figure 3.

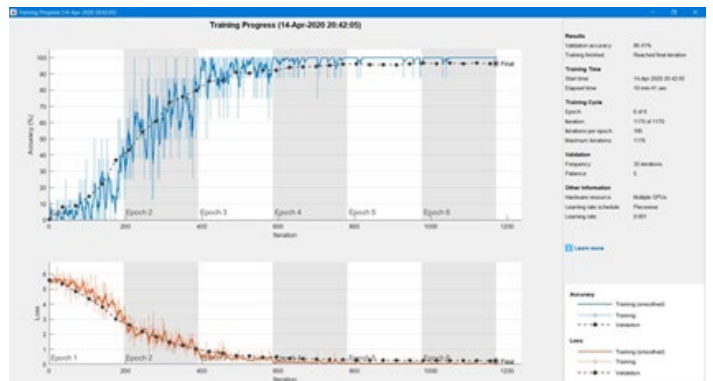


Figure 4.

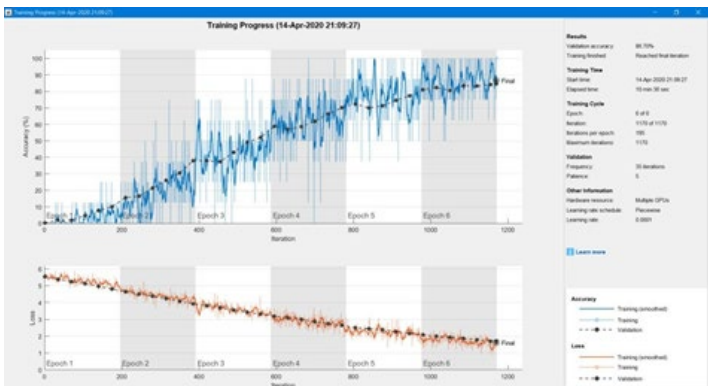


Figure 5

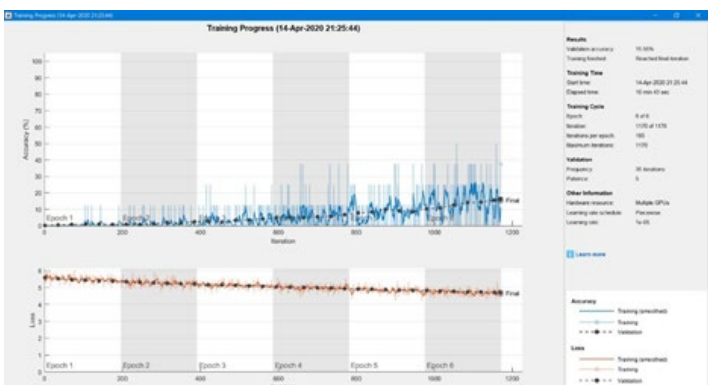


Figure 6.

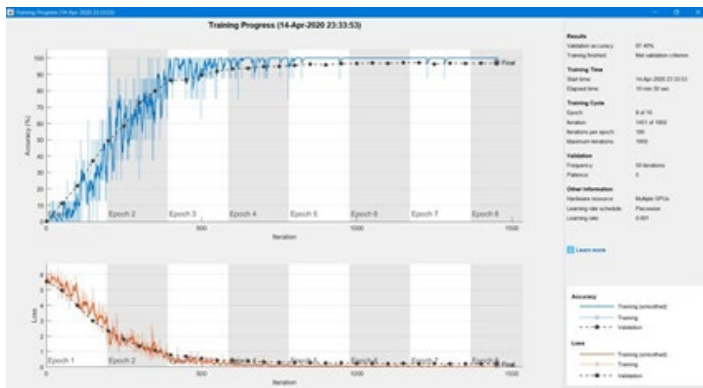


Figure 7.

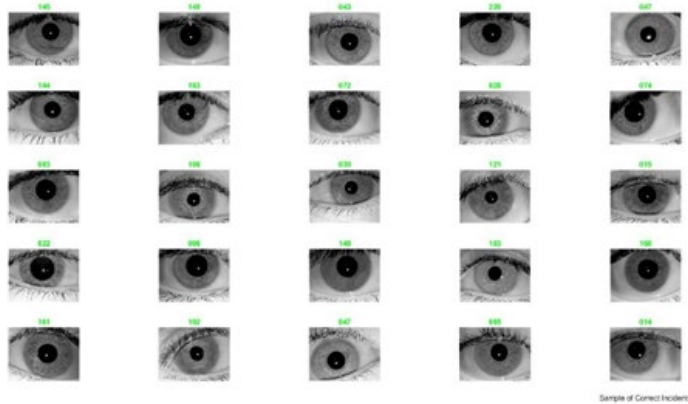
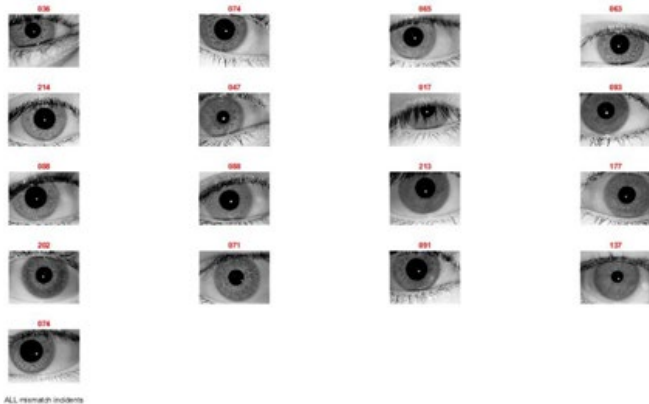


Figure 8.

Samples of Identification Results

The for loop for samples was adjusted to display all mismatches then a sample of successful matches were displayed with the same numbers of mismatches as follows:

Acknowledgment

Wish to thank my parents for their support and encouragement throughout my studies.

Reference

1. Kumar, A., & Passi, A. (2008). Comparison and combination of iris matchers for reliable personal authentication. Pattern recognition, 43(3), 1016-1026.

Copyright: ©2022 Wael Alnahari. This is an open-access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.