Todo list

Multimodal Biometric Identity Verification for Mobile Platforms

Project Report

Group 18gr842

Aalborg University
Vision, Graphics and Interactive Systems

This report is compiled in LaTeX. Additionally is Mathworks MATLAB, Adobe Illustrator, Lucidcharts.com, Inkscape, and Autodesk Eagle used to draw figures, schematics, and charts.

Vision, Graphics and Interactive Systems
Aalborg University
http://www.aau.dk

AALBORG UNIVERSITY
STUDENT REPORT

Title:
Multimodal Biometric Identity Verification for Mobile Platforms

Theme:
Computer Vision

Project Period:
Spring Semester 2018

Project Group:
Group 18gr842

Participants:
Marike Koch van den Broek
Shagen Djanian
Niclas Hjorth Stjernholm

Supervisor:
Kamal Nasrollahi

Number of Pages: 58

Date of Completion:
May 30, 2018

Abstract:

As computational power of mobile phones increases together with the capabilities of the built in camera new security measures are introduced to the consumer. These consist of biometric modalities such iris and face recognition separately. This project seeks to find a viable solution to combining the two modalities. By designing two networks, one for iris recognition and one for face recognition a fusion solution of the two networks are sought out. The iris recognition Convolutional Neural Network (CNN) manages to reach a precision of 99.7% and the face recognition 99.35% individuall[illegible]e fusion of the two networks...

Preface

Aalborg University, May 27, 2018

Marike Koch van den Broek
<mvanam13@student.aau.dk>

Shagen Djanian
<sdjani14@student.aau.dk>

Niclas Hjorth Stjernholm
<nstjer14@student.aau.dk>

Contents

Glossary

CDF Cumulative Distribution Function. 29

CNN Convolutional Neural Network. iii, 5, 6, 7, 13, 14, 35, 36, 37, 38, 40, 42, 43, 44, 45, 47, 51

CPU Central Processing Unit. 40

DBN Deep Belief Network. 13

DeepID Deep hidden identity features. 5, 6, 7

DeepID2 Deep IDentification-verification features. 6, 7

FC Fully Connected. 39, 40, 41

FCN Fully Convolutional Network. 14, 15

GPU Graphics Processing Unit. 40

HMM Hidden Markov Model. 5

KNN K-Nearest-Neighbours. 13, 16, 31, 32, 33

LDA Linear Discriminant Analysis. 5, 13, 31, 32, 33

LFW Labeled Faces in the Wild. 6, 7, 42, 45, 47, 51

MCS Multiple Classifier Systems. 16

NIR Near Infra-Red. 8, 11, 12, 15, 21, 23

PCA Principal Component Analysis. 5

ReLU Rectified Linear Unit. 37, 41

SPDNN Semi Parallel Deep Neural Networks. 15

SVM support vector machines. 5, 13, 16, 31, 32, 33, 35, 47

VL Visible Light. 8, 11, 12, 15, 21, 23, 47

Chapter 1

Introduction

The problem of protection of physical elements or information by limiting the access to only the people is well known. The central challenge to the problem of security is the challenge of verifying the identity of a person trying to acquire access. The emergence of biometric techniques has induced an increasing interest in biometric-based security rather than knowledge-based or token-based security. This is mainly due to the fact that the more traditional methods for security systems are easier breached or spoofed [Ross and Jain, 2003]. Through the last decades researchers have investigated identity verification based on different biometric modalities. In the last decade investigations have been conducted in combining several biometric modalities in one system with the purpose of creating a system that performs better than the ones only utilising a single one. Results shows that combining modalities performs better than any of the modalities seperately [Chen and Te Chu, 2005]. However, increased accuracy is not the only benefit of utilising multiple biometric traits. More modalities increases the universality of the system and decreases the influence of noisy measurements [Ross and Jain, 2003].

One of the fields where biometric-base security is increasingly applied is security for handheld devices such as smart phones. Although technology is advancing and mobile devices are equipped with still more advanced components, the computing power and the quality of the data from sensors, are both constrains of mobile devices. These limiting factors makes it more challenging ake successful biometric-based identity verification on mobile devices rather tha other applications. Though the use of machine learning for image processing is well known, it has only scarcely been applied on data obtained by cameras on mobile devices. Khan et al. [2017] presents different machine learning methods applied on iris images obtained by smartphone. Bazrafkan et al. [2017] applies deep learning for segmentation purposes on a database containing images acquired using a smart phones among others.

add source?

The work described in this report strives to make a system for identity verification based on multiple biometric traits for use on mobile devices. The biometric traits used are the iris and face. More specifically, this choice is made due to the arguments

d more challenges (n literature) before talking plan for solution

found i erature that iris is very distinctive, while face is non-invasive [Wang et al., 2009].

argue some more

Chapter 2

Research

To get an overview of which solutions already exist, research work has been carried out. This includes both face and iris recognition techniques but also existing solutions to network fusion, and is presented in this chapter, in the following sections.

2.1 Face Recognition

The first computer based face recognition was made in 1973. This was based on a feature approach, meaning the program identifies basic face features such as mouth, eye, and nose placement. From here three different types of approaches were made, namely a holistic, feature extraction and a hybrid approach.

The holistic approach encodes the entirety of a face and then identifies using template-matching, the feature extraction approach extracts a defined amount of features from the face, whereas the hybrid method uses both template-matching and feature extraction [Wechsler, 2007]. In 1990, Principal Component Analysis (PCA) was introduced for holistic face recognition. The PCA approach makes use of eigenfaces, each eigenface represents a component a face is encoded. But as Wechsler [2007] claims, Linear Discriminant Analysis (LDA) is a more effective suitable approach for face identification and authentication. Another holistic approach is using support vector machines (SVM) for face recognition [Wechsler, 2007].

The feature approach gave way for what is now known as recognition-by-parts, which uses the features and a global structure to link these features. A structure for linking 2D features is the Hidden Markov Model (HMM). PCA is also used in this approach, but is used to model shape or texture of the face.

In the late 2000s deep learning was introduced with representation-learning methods with multiple levels of representation. By feeding raw data and finding a emphasising on the important aspects of the data and suppressing the unimportant ones, higher level classification is possible [Lecun et al., 2015]. This is done with several *hidden layers*. The more layers there is the deeper a network is said to be. In the following some of the state of the art deep learning networks for face recognition are presented.

2.1.1 DeepID

Deep hidden identity features (DeepID) is a CNN which aims to use feature extraction for face identification and verification. It detects five facial landmarks; the two eye centres, the nose tip, and the two mouth corners. The network is made of four convolutional layers with max-pooling, which are used to extract features hierarchically. These are followed by the fully-connected DeepID layer and a softmax output layer to indicate identity classes. The feature extraction and recognition is done in two steps, where the first feature extraction is learned with the target of face identification [Sun et al., 2014a].

In the CNNs the neuron number of the last hidden layer in the network is much smaller than that of the output layer. This is done, to better classify faces [Sun et al., 2014a]. The network extracts low-level features in the bottom layers, where feature numbers decreases for each layer. In opposition, the high-level features are formed in the top layers.

The network is tested using the Labeled Faces in the Wild (LFW) database. This database is images of faces from different angles and scenarios consisting of 13.233 images [Huang et al., 2007]. It achieves 97.45% accuracy on this dataset, requiring weakly aligned faces [Sun et al., 2014a].

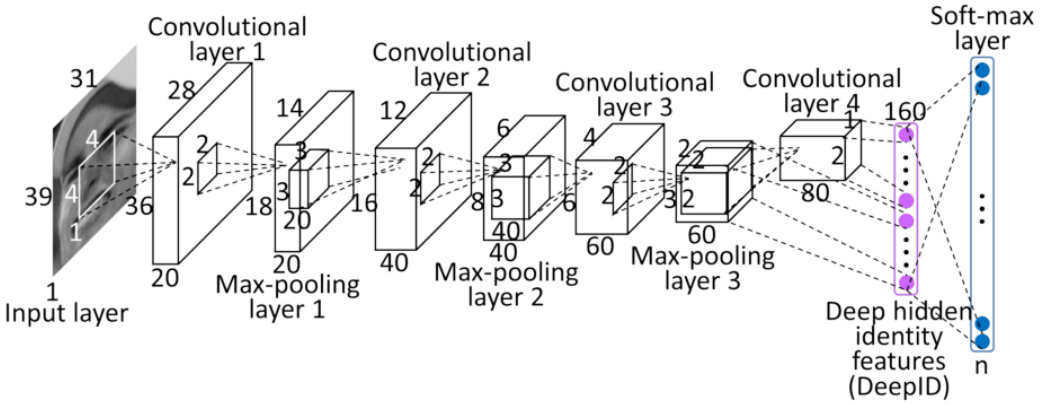


Figure 2.1: Structure of the CNN used in DeepID [Sun et al., 2014a]

2.1.2 DeepID2

Deep IDentification-verification features (DeepID2) is an expansion upon DeepID and is a deep Convolutional Neural Network used for face identification and verification. This is done by using feature extraction.

Just like DeepID, DeepID2 uses four convolutional layers but only the first three uses max-pooling. It uses 400 face patches instead of 60 [Sun et al., 2014a,b] and detects 21 landmarks of the face.

The DeepID2 layer is after the four convolutional layers. This layer is learned under two supervisory signals. The first is identification classifying the images into

identities. The second is face verification which manipulates the DeepID2 data to be similar to a matching identity should this be the same.

DeepID2 also uses the LFW database and achieves a 99.15% accuracy.

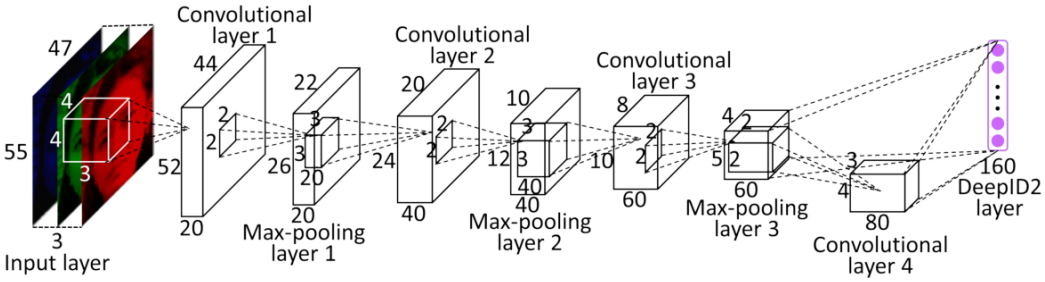


Figure 2.2: Structure of the CNN used in DeepID2 [Sun et al., 2014b]

2.1.3 DeepID3

DeepID3 is a further expansion of both DeepID and DeepID2 but is also drawing on some from elements from VGG net and GoogleNet [Sun et al., 2015]. The qualities from these two networks is the use of stacked convolution and inception layers. DeepID3 is in general a deeper network than DeepID2 and its expansion DeepID2+.

DeepID3 resembles DeepID2 in the use of adding supervisory signals to early layers.

Sun et al. [2015] proposes two different networks with DeepID3. One is using eight convolutional layers with max pooling after every other convolutional layer. The second network has four convolutional layers with max pooling after every other, following are five inception layers.

DeepID3 is also tested on the LFW dataset with an accuracy of 99.52% which is an increase in accuracy compared to DeepID2, but as stated in Sun et al. [2015] it is not an improvement of DeepID2+.

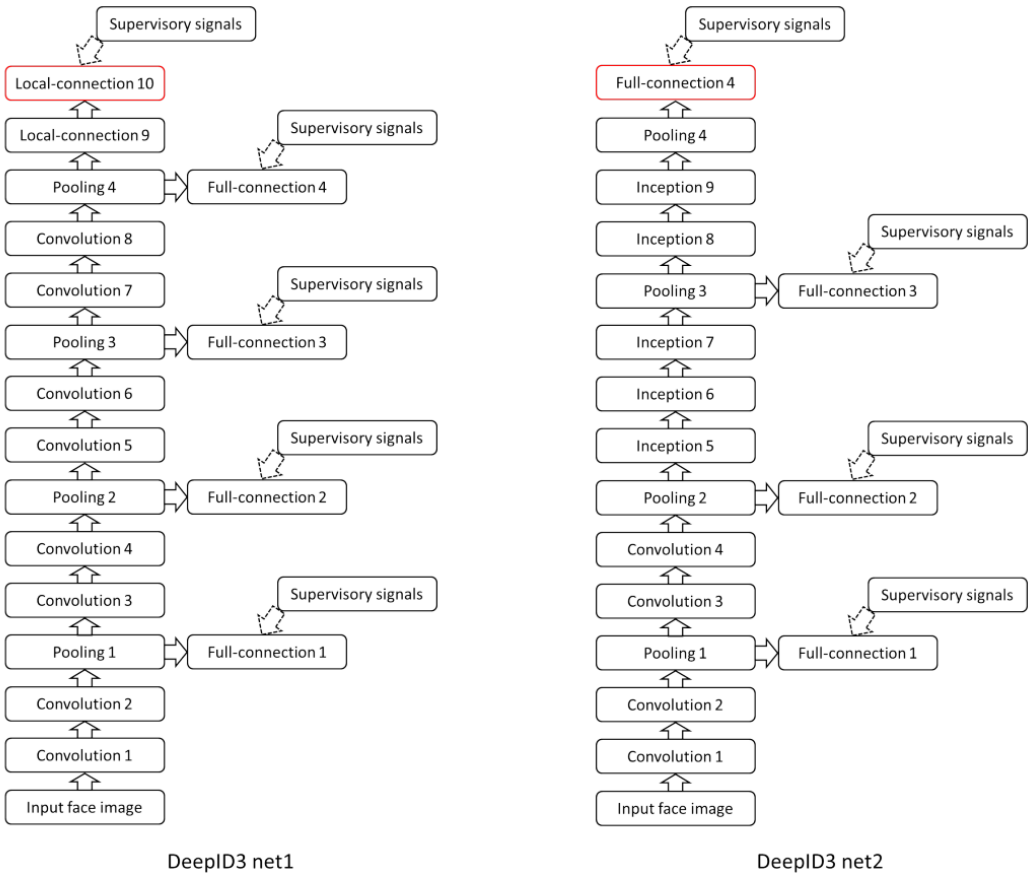


Figure 2.3: Structure of the CNNs used in DeepID3 [Sun et al., 2015]

2.2 Iris Recognition

Modern iris recognition was first introduced in an article by Daugman [1993] discussing the security of using iris for recognition. Here an outline of how to do recognition was laid out and even though the field has been extensively improved since Daugman [1993], the general methodology is more or less the same as Daugman proposed. A modern system is typically composed of image acquisition, segmentation and normalisation, feature extraction and matching.

2.2.1 Images

The images acquired are often taken in the Near Infra-Red (NIR) spectrum, which is ranging between 700 nm and 900 nm in wavelength. In this band, the melanin in the iris, which is the substance that gives the iris its colour e.g. brown or blue, is typically less prominent making the unique structure in the iris very distinct. To acquire usable NIR images, the user has to be in the millimetre range of the NIR

camera. In the visible light, the melanin is much more prominent and thus makes is harder to detect the structure. The band of visible light has many names in the literature, namely Visible Spectrum (VIS), Visible Wavelength (VW), and Visible Light (VL). VL will be used in this report. While NIR is beneficial in some cases, other useful features can be observed in the VL that cannot be seen with NIR. These can include the moles, freckles and conjunctival vasculature, which can help in making more accurate recognition systems. To make iris recognitions systems comparable with each other some publicly available databases are often used. Rifaee et al. [2017] give an outline of the some of the free databases that are used. A comprehensive table summarising the visual properties, statistics and type of subject used in various database are tabulated in Figure 2.4 and Figure 2.5. Most of the databases available are NIR images with CASIA being one of the most used database. For VL images, UBIRIS in the most commonly used. These contain more "real world" data as the images contain more noise in the form of eyelids obstruction, eyelash obstruction, glare, motion blur, out-of-focus or poor focused iris, partial iris and specular reflection [Rattani and Derakhshani, 2017]. The database has also been used in Noisy Iris Challenge Evaluation (NICE) I. The increasing usage of mobile devices also proposes an opportunity to integrate iris recognition as a biometric for verifying the identity of the user. For this purpose a competition, Mobile Iris CHallenge Evaluation (MICHE), is made to compare the state of the art mobile iris. They provide a the MICHE database that can be used, which they claim is a better database than UBRIS for mobile systems. The database contains noisy iris images taken with a Galaxy Samsung IV, iPhone 5, and Galaxy Tablet II. The noise includes noise from both artificial and natural light sources during acquisition, motion blur, occlusion due to eyelids, glasses, eyelashes, hair, or shadows, which can naturally occur when a user is trying to unlock a phone using their iris.

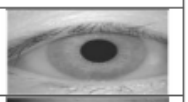
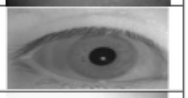
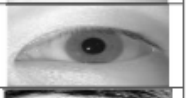
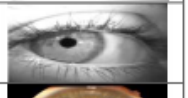
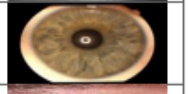
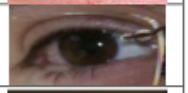
Database name	Database size	Light wave length	Varying distance	Camera	Sample image
CASIA v1	756	NIR	No	CASIA camera	
CASIA v2	2,255	NIR	No	CASIA camera	
CASIA v3	22,051	NIR	No	OKI iris-pass h	
CASIA v4	2,576	NIR	Yes	IKEMB-100 dual camera	
Bath	16,000	NIR	No	ISG LW 1.3 S 1394	
MMU 1	450	NIR	No	LG EOU 2200	
MMU 2	995	NIR	No	Panasonic BM ET 100 US	
ICE 1	2,900	NIR	No	LG EOU 2200	
ICE 2	75,000	NIR	No	LG EOU 2200	
WVU	3099	NIR	No	OKI iris-pass h	
UPOL	384	Visible	No	Sony DXC 950P 3CCD with TOPCON TRC501A	
UBIRIS v1	1877	Visible	No	NIKON E5700	
UBIRIS v2	11,357	Visible	Yes	Canon EOS 5D	
FRGC	50,000	Visible	Yes	Minolta Vivid 900/910	

Figure 2.4: A table depicting the contents of free iris image databases [Rifaee et al., 2017].

Database	Noise													
	Eyelashes	Eyelids	Specular Reflection	Light Reflection	Motion Blurred	Poor Focus	Gaze Deviated	Partially Occluded	Out of Iris	Over Distance	On The Move	Rotated	Glasses	Visible Wavelength
CASIA 1	√	√	-	-	-	-	-	-	-	-	-	-	-	-
CASIA 2	√	√	-	-	-	-	-	-	-	-	-	-	-	-
CASIA 3	√	√	-	-	-	-	-	-	-	-	-	-	-	-
CASIA 4	√	√	-	-	√	√	√	√	-	√	√	√	-	-
BATH	√	√	-	-	-	-	√	-	-	-	-	√	-	-
MMU 1	√	√	-	-	-	-	-	-	-	-	-	-	-	-
MMU 2	√	√	-	-	-	-	-	-	-	-	-	-	-	-
ICE 1	√	√	-	-	-	-	√	-	-	-	-	√	-	-
ICE 2	√	√	-	-	-	-	√	√	√	-	-	√	-	-
WVU	√	√	-	-	√	√	√	√	-	-	√	√	-	-
UPOL	-	-	-	-	-	-	-	-	-	-	-	-	-	√
UBIRIS.v1	√	√	√	√	√	√	√	√	√	-	-	√	√	√
UBIRIS.v2	√	√	√	√	√	√	√	√	√	√	√	√	√	√
FRGC	√	√	√	√	√	√	√	√	√	√	√	√	√	√

Figure 2.5: A table depicting the noise present in the databases [Rifaee et al., 2017].

2.2.2 Segmentation

Segmentation of an iris in iris recognition systems tries to detect the iris and find the pupillary boundary and the limbus boundary as well the eyelids and eyelashes that can cause noise on the image. The boundaries along with other parts of the iris can

be seen in Figure 2.6. The approach used for segmentation depends among other things on the wavelength of the image; NIR or VL. They both have some common challenges to them. Often the eyelids can cover a small part of the iris, causing the limbus boundary of the iris to not be circular or elliptical. Eyelashes can also cause a similar disturbance as they also can cover parts of the iris. Poor lighting can also make it extremely difficult to detect the boundaries. Specular reflections in the iris can also cause difficulties as they can lie on the iris boundary or close to. Most systems also require a great deal of user cooperation as an off angle iris, motion blur, or glasses or contact lenses can make it even more difficult to detect the boundaries. This can especially be the case for iris recognition in a phone as it cannot be expected of the user knows how to acquire a good iris image.

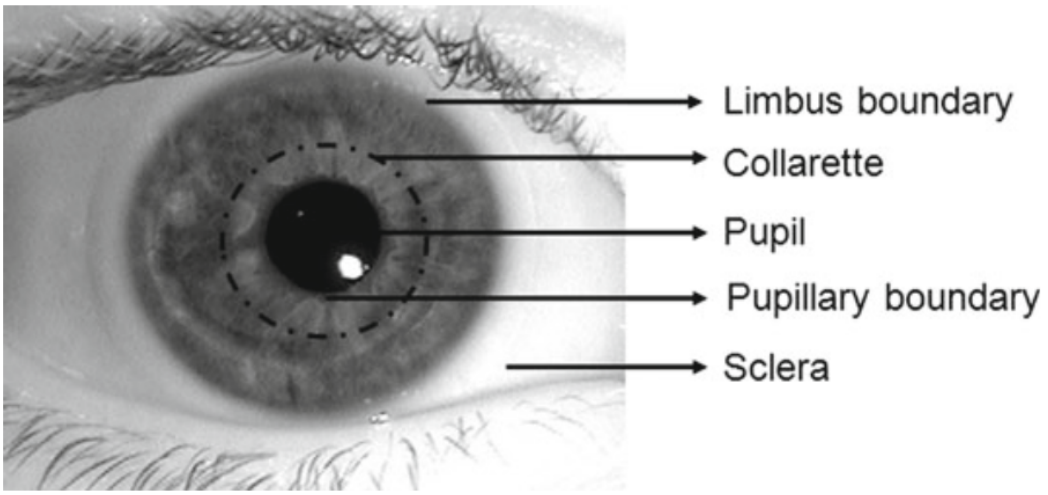


Figure 2.6: A close up NIR image of an iris depicting the different parts of the iris along with their names [Connaughton et al., 2016]

Two commonly observed approaches for segmentation in NIR band images are Daugman's approach [Daugman, 1993] , [Saha et al., 2017], [Rattani and Derakhshani, 2017], [Khan et al., 2017], and Hough Transform [Luhadiya and Khedkar, 2017], [Uka et al., 2017]. Daugman's approach consists of using a Gaussian filter on the image to attenuate the effect of noise and eliminate undesired weak edges like the boundaries within the iris while keeping the strong edges like iris boundaries and eyelid boundaries. An integro-differential operator is then used as a circular edge detector. It is then used iteratively to find the pupillary boundary and the limbus boundary. Hough Transform on the other hand is a histogram based model fitting approach. An edge map of the input map is generated using a gradient-based edge detector. Then a voting procedure is applied on the thresholded edge map to determine the parameters for a contour that best fits a circle. This operation gives an approximate edge map of the iris boundary. Lastly, the segmented iris is often normalised using Daugman's rubber sheet model that maps every point in the segmented region from

cartesian to polar coordinates. An open source MATLAB implementation based on updated version of the Daugman approach is a commonly used tool [Percy]. There exist other methods for different circumstances, but these are most commonly used. They can also be used on VL images if the image is converted from RGB to grey scale images [Bowyer et al., 2016].

2.2.3 Feature extraction and classification

There are multiple ways that the features can be extracted from the segmented iris. The most commonly used in the literature is a 2D Gabor filter which is a linear filter used for edge detection [Daugman, 1993]. The Hamming distance is then used as way to classify the iris. The Hamming distance is a measurement of how many bit flips a piece of data need to have to match another piece of data. The bits of the extracted features are then measured against the whole database and the pair with the lowest score is a match. This is called "1-to-N search". As the database gets larger and larger the computation time also grows as it will have to search through the whole database. That's why Kuehlkamp and Bowyer [2016] have suggested using a "1-to-first search" instead to improve the speed of the search. Here a threshold is chosen and as soon as a match has been found below the threshold it will stop the search. Other approaches to the categorisation have been proposed using machine learning. Khan et al. [2017] proposed using SVM, K-Nearest-Neighbours (KNN) and LDA with respective test accuracies of 97%, 95.1%, 94.28%. In comparisons the commercial systems ranged from 94.57% to 99.67% with VeriEye having the lowest and IriCore having the highest accuracy.

2.2.4 Deep Learning

According to Zhao and Kumar [2017] research in neural networks within iris recognition is still very new and not much has been done. Some of the work that has been done have used CNN and a Deep Belief Network (DBN). Al-Waisy et al. [2017b] used a common CNN to extract features and classify a segmented and normalised iris image. The segmentation was done using Circular Hough Transform (CHT) normalisation was done using Daugman's rubber sheet method. They named the network IrisConvNet and the architecture was inspired by the Spoofnet as it can be seen in Figure 2.7. This was the general architecture and they tried different numbers of maps and layers to find the best architecture. In general a 3x3 input kernal was used on a 64x64 or 128x128 pixel input image to create feature maps followed by 2x2 max pooling, 5x5 convolution, 2x2 max pooling 5x5 convolution, 2x2 max pooling, 5x5 convolution and finally a two fully connected layers to to a softmax regression classifier layer. A ReLU activation function is applied on the top of the convolutional and fully connected layers because it results in several times faster training without sacrificing accuracy. The AdaGrad algorithm was used for training the network. Three databases were used; SDUMLA-HMT, CASIA-Iris-V3 and IIT Dehli (IITD).

IrisConvNet scored an accuracy of 99.82% at categorising CASIA-Iris-V3 in 0.65 s. architecture

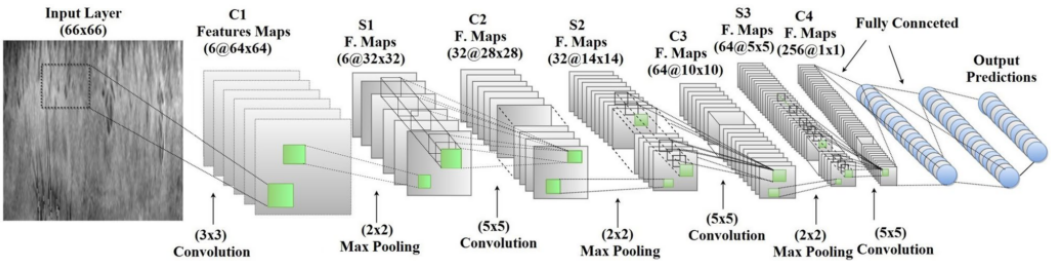


Figure 2.7: Example of a CNN architecture from Al-Waisy et al. [2017b].

Zhao and Kumar [2017] claim that the problem with traditional networks are that they are database specific and not very generalisable. They created a Fully Convolutional Network (FCN) using a loss function they created specifically for iris networks called Extended Triplet Loss (ETL). They claim this network is more generalisable than the previous networks and it shows superior results compared with IrisCode which is Daugman's segmentation and normalisation approach. A FCN differs from a CNN in that there are no fully connected layers, only convolutional layers, max pooling etc. They proposed an architecture called UniNet that can be seen in Figure 2.8. It consists of two FCNs; FeatNet and MaskNet. The network takes an iris that has been segmented and normalised using a recent approach [Zhao and Kumar, 2015]. The segmented iris has a resolution of 64x512 pixels. The image is then fed through FeatNet and MaskNet. FeatNet extracts the iris features while MaskNet masks the non-iris part of the image. e.g an eyelid that occludes. Three of these UniNet networks are then trained in parallel in Triplet-based network architecture that uses the ETL loss function. They used four databases to train and the networks; ND-IRIS-0405, CASIA V4, IITD and WVU Non-ideal Iris Database. It was then compared with a CNN network based on VGG-16 that used softmax, CNN with triplet, FeatNet only and DeepIrisNet which is a CNN that is proposed directly for iris recognition. The results can be seen in Figure 2.9 where UniNet outperforms the other nets and FeatNet is the by far worst performing network, which suggests that MaskNet is needed for the network to perform well.

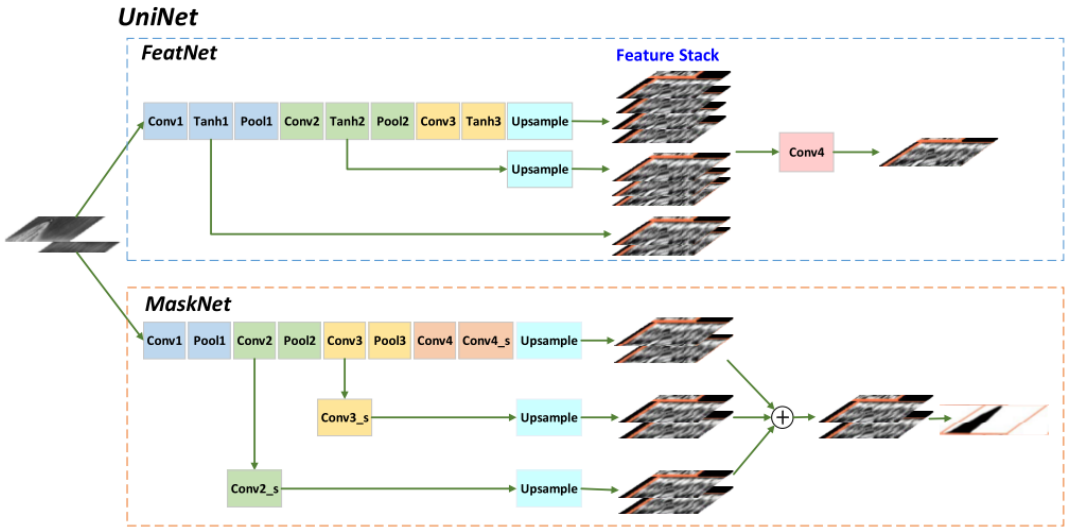


Figure 2.8: Example of a CNN architecture from **?**.

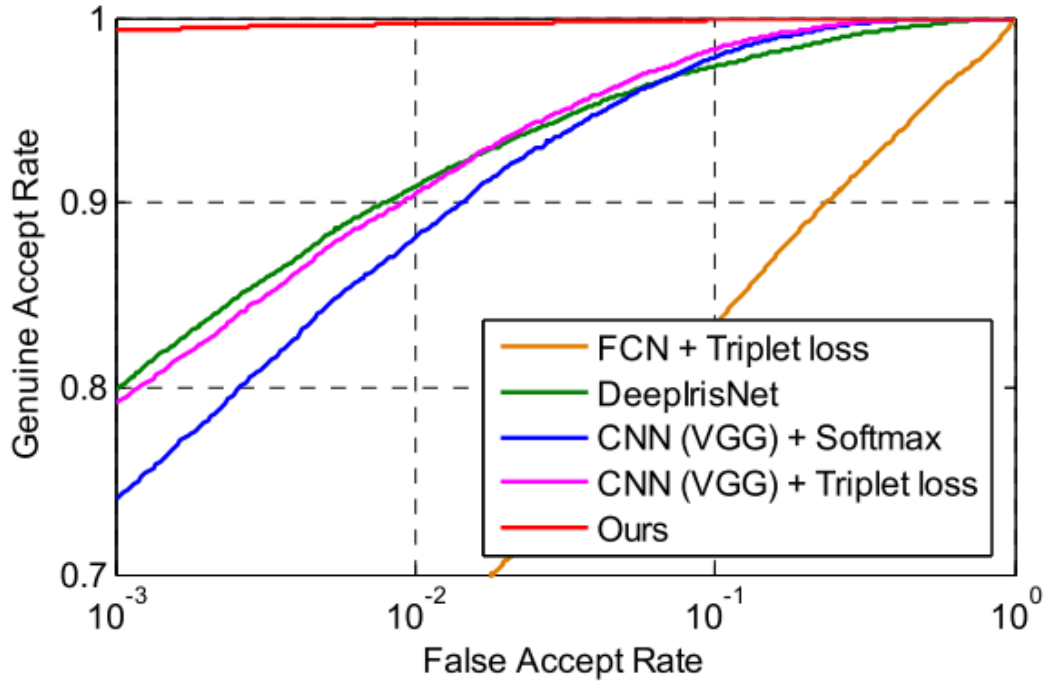


Figure 2.9: Results from the different networks tested on the ND-IRIS-0405 database **?**. "Ours" is UniNet and FCN + Tripletloss is FeatNet

Another promising approach by Bazrafkan et al. [2017] targets iris segmentation in low-quality consumer images obtained from smartphones is using Semi Parallel

Deep Neural Networks (SPDNN) for generating iris maps from low quality iris images. In SPDNN several deep networks are merged into a single model. This way it is possible to include different networks designs and combine their strengths. They combined four FCNs of different architectures that used a variaty of kernel sizes and layers. An example of one of the FCNs has 12 convolutional layers in this order; 3x3,5x5,7x7,9x9,11x11,13x13,15x15,13x13,11x11,9x9,7x7,5x5 and an output layer that is 3x3.It was trained by using the databases Bath800, CASIA Thousand, UBRIS and MobBio. They contained both NIR and VL images. Extensive noise was also added to the training images in the form of blur and lower contrast to generate degraded versions of the high quality images and utilise the databases better by creating more samples. Using the UBRIS database as a comparison with other state of the art segmentation techniques it achieved the highest accuracy with 99.30%. The technique with the lowest accuracy, of the ones it was compared to, had an accuracy of 98.10%.

2.3 Information Fus

A system that strives to utilise information from two or more different biometric traits in order to obtain one combined result is a called a multi-modal system which is a kind of multi-biometric system. Besides the multi-modal approach there are several other approaches, which results in information from two or more sources, therefore, methods for fusion of information from different sources into one system for classification purposes is a widely investigated area. [Connaughton et al., 2016]

Fusion of information can happen on different levels. It can happen on one of five levels, each with a higher level of preprocessing: signal level, feature level, score level, rank level, or decision level. The level on which the fusion should be done depends on the kinds of multi biometric data that has to be fused, and the purpose of the fusing. In general score-level and feature-level are the most popular fusing techniques [Connaughton et al., 2016]. Fusing at the lowest level, signal level, might be done in order to merge data from different sources to construct a more detailed or larger dataset or signal. At the feature level the fusing might happen through merging of extracted features from different sources into one feature vector [Ross and Jain, 2003]. At the score level the fusion can be done in order to determining the best sample to use for the processing based on which has the highest score and thus is the best match to the gallery samples. Rank level can be somewhat similar to the scores but depend on match rankings. At the decision level it can be making the decision based multiple classifiers, e.g. one for each modality[Fierrez et al., 2018].

In the literature a large variety of methods and algorithms have been utilised for fusing information on different levels. Some algorithms are fairly simple and basically makes decisions about which sample to use onwards for classification based on matching scores between samples and the gallery. Other methods are more advanced and well known for use in applications such as classification. These are methods such as SVM, KNN, decision trees, and bayesian methods [Ross and Jain, 2003].

The highest rank method and Borda count are both well known methods for combining information based on ranking. The highest rank method ranks possible classes based on the highest rank assigned to the class by a classifier across all classifiers.[Ho et al., 1994] The Borda count is a kind of majority voting which can be used in combination with Multiple Classifier Systems (MCS) [Connaughton et al., 2016; Ho et al., 1994]. An important aspect to consider is that whenever different kinds of information are fused together, the information should be represented in the same data space, otherwise unwanted weighing of certain information above other might occur.

In this report, the focus is to implement a multimodal system utilising the biometric traits face and iris. In literature different approaches for the fusion of these traits have been presented. Al-Waisy et al. [2017a] as one of the lates researches within the area presented a work utilising deep learning for feature extraction and matching of the biometric traits and tests different methods for score and rank level fusion of the modalities. The tested methods on score level include eg. sum, weighted sum, max etc while the rank level fusion uses the mentioned Borda count, Highest rank, or Logistic regression. Chen and Te Chu [2005] did processing of iris and face separately and only combined the two modalities at the decision-level by adding inferred probabilistic values in a linear combination.

2.4 Multi-Modal Databases

Even though recognition based on biometric traits is widely investigated, and research shows that multimodal systems perform better than the uni-modal systems based on the same data, the research in this area is limited and incomplete [Chen and Te Chu, 2005; Connaughton et al., 2016]. Because of the limited availability of multimodal datasets, such datasets are often synthetically constructed based on randomly combined data, eg. iris and face datasets [Chen and Te Chu, 2005]. Only a limited amount of studies utilise a multimodal dataset obtained from the same test subjects or maybe even with one sensor. However, a few multimodal datasets have been encountered in literature. The multimodal datasets varies in which modalities they include. The modalities can be different images of face, recordings of gait, hand geometry, handwriting, signature, fingerprint, finger vein, iris, speech etc. [Yin et al., 2011; Dessimoz et al., 2007; Ross and Jain, 2003; Ortega-Garcia et al., 2010]. Examples of multimodal datasets containing both iris and face are the database IV^2 [Petrovska-Delacrétaz et al., 2008], consisting of data obtained from 300 subjects, the MBioID based on 120 subjects [Dessimoz et al., 2007], The BiosecureID based on 400 subjects, The BioSec database based on 250 subjects, the BMDB DS2 based on 667 subjects [Ortega-Garcia et al., 2010], the SDUMLA-HTM database based on 106 subjects, the MobBio containing data from 105 subjects acquired through a mobile device [Sequeira et al., 2014],and the datasets provided for the The Multiple Biometric Grand Challenge (MBGC) [Connaughton et al., 2016]. The latter is available in two versions and can be obtained on request. Furthermore, it serves as a common

test set in order to compare performance. However, some multimodal datasets have been created by researchers during their work, which are not named nor generally available [Connaughton et al., 2016].

A concern often addressed in literature is that chimeric datasets comprising data from different databases might have a flaw compared to genuine multimodal datasets. This due to that fact that it assumes no correlation between the different biometric traits, which might not be true. In fact study shows that they are correlated and testing on chimeric datasets is thus not reflect the performance of a system in genuine settings[Al-Waisy et al., 2017a].

Database Name	Responsible	Number of Subjects	Modalities
IV^2	Petrovska-Delacrétaz et al. [2008]	300	VL Face 3D face laser scan 3D face stereoscopic IR Iris
MBioID	Dessimoz et al. [2007]	120	VL Face 3D face Iris Fingerprint Voice Online signature
BiosecureID	Fierrez et al. [2010]	400	Face Iris Fingerprint Voice Signature Hand Handwriting Keystroke
BioSec	Fierrez et al. [2007]	250	Face Iris Fingerprint Voice
BMDB DS2	Ortega-Garcia et al. [2010]	667	VL Face 3D face laser scan IR Iris Fingerprint Voice Signature Hand
SDUMLA-HTM	Yin et al. [2011]	106	VL Face NIR Iris Fingerprint Finger Vein Gait
MobBio	Sequeira et al. [2014]	105	VL Face VL Iris Voice
MBGC v1/v2	NIST	-	Face Iris
	Connaughton et al. [2016]	363	Face Iris

Table 2.1: Multi modal biometric databases containing both iris and face information.

Discuss this table

The table...

Based on the knowledge gained from the research done, a set of requirements regarding the development of a solution are constructed. This is done to measure the quality of the solution.

Chapter 3

Requirements

To be able to measure the quality of the identification method made, a set of requirements are presented. These are produced on the background of knowledge presented in chapter 2.

The context of the work carried out is identity verification on smart phones. Therefore, there are certain requirements to the system. Though some smartphones nowadays are equipped with Near Infra-Red (NIR) cameras not all are. It is also beneficial when successful identity verification is possible using the normal Visible Light (VL) front camera which most smartphones are equipped with, as this requires less sensors and thus is cheaper for the smart phone manufacturers. From these considerations two requirements for the system arises. The input images used for the identification has to be VL images, and furthermore, the images have to have a resolution which is low enough to represent the images that would be captured with a smart phone front camera.

Furthermore, the system has to identify using one of the biometric modalities face, or iris, or both, since those are the most reliable non-invasive biometric traits which can be obtained by a smart phone and especially with a VL camera.

The designed solutions must have an accuracy comparable to the ones of state of the art solutions presented in chapter 2. The accuracies presented are generally at 99% or above, which means the solution implemented in this project must have an accuracy equal to or above this accuracy as well.

For the solution for identification based on fusion of the two modalities to be accepted, the accuracy has to be higher than the accuracies of the system based on the individual modalities. If the accuracy is the same there is no gain in the increased computational complexity of a multimodal system.

Chapter 4

Implementation

4.1 Basic Methods

In order to get some basic understanding of the methods commonly used for iris classification the work presented in an article is implemented. The work implemented is the work of Khan et al. [2017] described in *Iris Recognition using Machine Learning from Smartphones Captured Images in Visible Light*. In the work the database Warsaw-BioBase containing VL images of the iris obtained by smartphones are processed and used to train different classifiers. The images of this database comply with the requirements set for this work mentioned in chapter 3. The processing consists of a sequence of steps. The steps included are

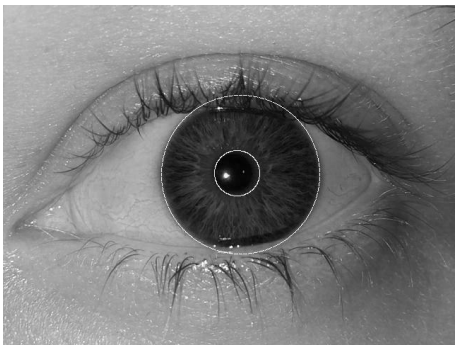
- Iris Location
- Eyelid Suppression
- Iris Normalisation
- Noise Removal
- Histogram Equalisation
- Feature Extraction
- Training and Classification

Figure 4.1a shows an example of an image from the database. Before the first step some simple preprocessing was done. The red channel is preserved and the other two other colour channels are neglected. This is done because this simulates the use of NIR light. The result is a greyscale image based on the red channel. In the following sections the implementation of each of the steps will be elaborated.

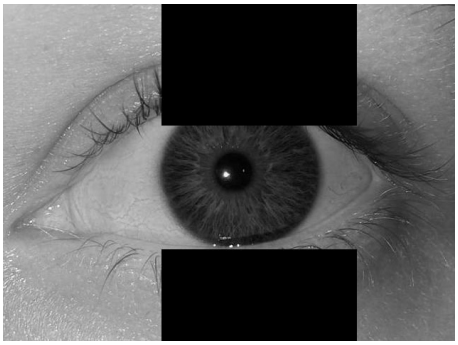
4.1.1 Iris Location

The first step of processing is locating the iris. For this purpose Daugman's Integro-differential operator was used. The operator identifies the circular contour, which has the greatest change in intensity, by varying the three parameters defining the circle

(a) Example of visible light image of an iris from the used database.



(b) The eye marked with the identified edges of the iris.



(c) The eye with the eyelid suppression applied.

Figure 4.1: The original image before and after different steps of processing.

(r, x_0, y_0), radius, and centre coordinates. The operator is defined by the formula in Equation 4.1.

$$\max(r, x_0, y_0)\left|G_\sigma(r) * \frac{\partial}{\partial_r}\oint_{\mathrm{r,x_0,y_0}} \frac{I(x,y)}{2\pi r} d_S\right| \tag{4.1}$$

Where $I(x, y)$ is the intensity or grey level of the image at the coordinates (x,y), S is the circle, $G_\sigma(r)$ is a gaussian smoothing function. The method is also used to find the edge between the pupil and the iris. The two identified endges are marked on the input image. Figure 4.1b shows an example of the identified edges of the iris.

4.1.2 Eyelid Suppression

The annulus identified by the iris location as the area of the iris might not only contain the iris. The subject might in some cases not open the eyelids fully or the angle of the camera will be such that parts of the iris will be covered by the eyelids. Therefore an algorithm is used which eliminates the eyelids. The algorithm is detecting applied on the image cutout of the original image which has the boundary box of the identified

iris as edges. The small image is divided into an upper and a lower part which are searched for the upper and the lower eyelid respectively. The search is done by using a method with some similarities to Daugman's Integro-differential operator shown in Equation 4.1, however it applies radon transform. Through the output of the radon transform the lines marking the two eyelids are found and the pixels above or below the two lines is eliminated. The resulting image is shown in Figure 4.1c.

4.1.3 Iris Normalisation

For the normalisation Daugman's rubber sheet model is used. The purpose is to get a rectangular image of the iris, which corresponds to taking the annulus covered by the iris region, cutting it open and unfolding or stretching it to a rectangular shape.

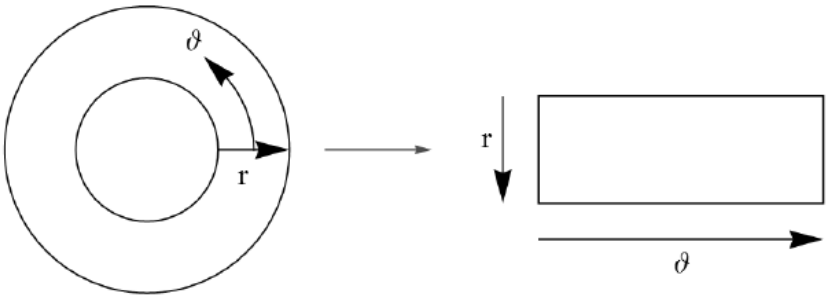


Figure 4.2: Daugman's Rubber Sheet Model. [Misztal et al., 2012]

The model does this by mapping the iris from the original image to a rectangular image. The mapping is done based on polar coordinates radius and angle, (r, ϑ), where $r \in [0, 1]$ and $\vartheta \in [0, 2\pi]$. In the rectangular image angle ϑ is on the x-axis and the radius r is on the y-axis of the image. Figure 4.2 shows the mapping of the method. This was implemented using a function in the library crated my Libor Masek [Libor Masek, 2003]. The resolution of the resulting image is dependent on arguments. The image of the normalised iris is shown in Figure 4.3.

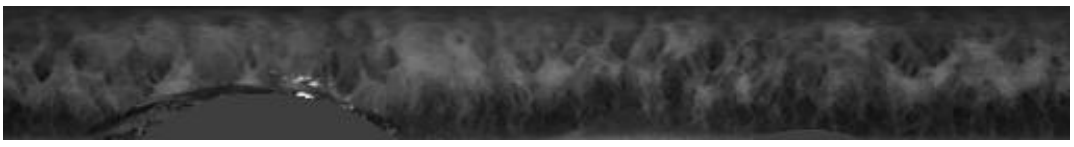


Figure 4.3: The normalised image of the iris before applying functions.

4.1.4 Noise Removal

Though the article uses several well known methods which are commonly used in processing of images of irises, their descriptions of the methods are quite inadequate. After obtaining the normalised iris, the next step applied is noise removal. The

purpose of the noise remover function is to remove noise occurring in the form of eyelashes covering parts of the iris. Usually the pixels showing the lashes will be among the darkest pixels. Since every image of the iris is different and how dark the iris is also varies, a threshold has to be identified adaptively. The article does not describe in depth how this is implemented, it simply states that some histogram analysis is done in order to obtain the lowest pixel values. Figure 4.4 shows the histogram of the normalised image before any noise removal.

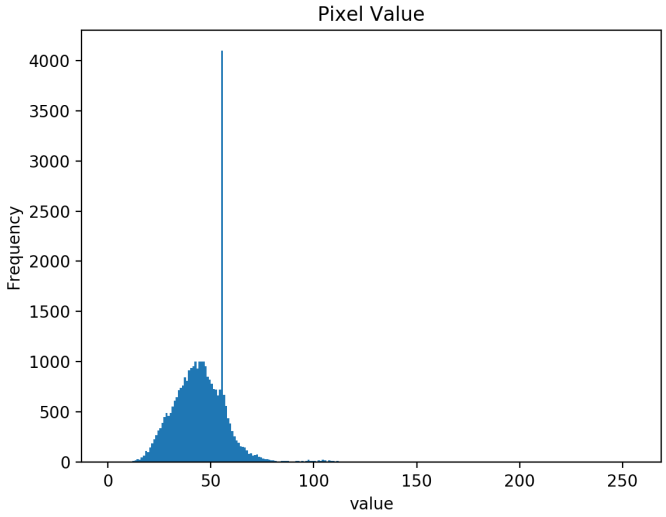


Figure 4.4: The histogram before applying the functions.

As the information about the exact approach used in the article is inadequate, an adaptive algorithm is created. The algorithm implemented identifies a threshold value based on the histogram. This is done by first identifying the highest and lowest bin-value, which has a frequency of more than a specified "recognition value", which was set to 10 in this project. The recognition value is introduced to make sure outliers are not defining for the threshold. Afterwards the threshold is calculated by the formula in Equation 4.2, where *Fraction* is a parameter set manually defining how large a part of the identified pixel value range has to me thresholded. During the processing in this project "Fraction" was set to be equal 0.1.

$$\text{Threshold} = \text{Low Value} + \text{Fraction} \cdot (\text{High Value} - \text{Low Value}) \tag{4.2}$$

After the threshold has been found it is applied to all pixels in the image. The pixels lower than the threshold are eliminated and have to be reconstructed from neighbouring pixels. Also here the article provides very limited information about the algorithm applied. Therefore an algorithm was implemented, which restores pixels from non-occluded neighbour regions. A part of the algorithm identifies the pixels with values lower than the threshold and saves the pixel coordinates of the pixels.

The saved pixels are reconstructed iteratively from neighbouring pixels following the 4-connectivity principle. The pixels are reconstructed when there are at least 2 neighbour pixels they can be reconstructed from. They are only reconstructed from pixels above the threshold, this can be pixels that initially were above the threshold, or it can be pixels that have already been reconstructed. The pixels are reconstructed by assigning the average of the neighbours with values above the threshold as the new pixel value. Once all thresholded pixels have been reconstructed the reconstructed image is returned.

Because the eliminated pixels are reconstructed only from pixels with a value higher than the threshold there are certain traits that can be expected in the histogram of the reconstructed image. One of the traits is that there is a flatline from 0 to the threshold value. A second trait is that a peak close to the threshold is likely to occur because the eliminated values are reconstructed from neighbours which are likely to be close to the value of the eliminated pixels. However, the plotted histogram after reconstruction is as shown in Figure 4.5.

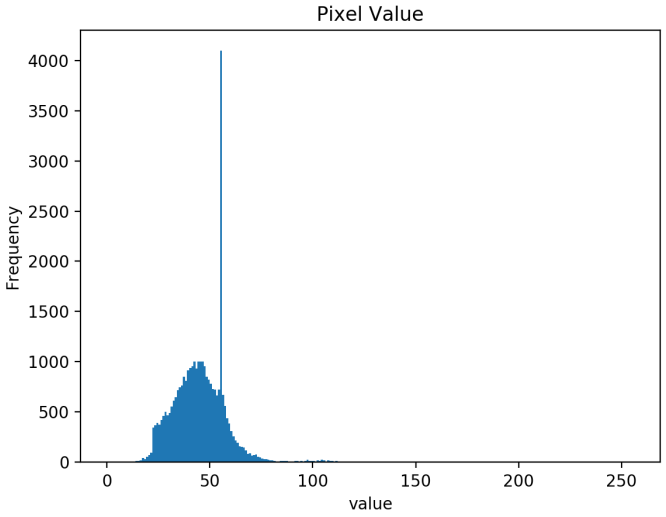


Figure 4.5: The histogram of the image after reconstruction of pixels.

As can be seen there is a small peak as expected, however, there seem to be a "spill over" across the threshold to the lower values. By closer examination of the code it was discovered that this was caused due to a programming mistake. The mistake was that the values used for reconstruction were obtained from the original image and not from the reconstructed image. Once this mistake was corrected the histogram was as expected as shown in Figure 4.6.

In relation to the reconstruction of pixels it should be noted that it could have been done with 8-connectivity, and the iterative process could be split up to more steps, such that pixels are always constructed from as many neighbours as possible. This may give a better reconstruction, however, this has not been investigated.

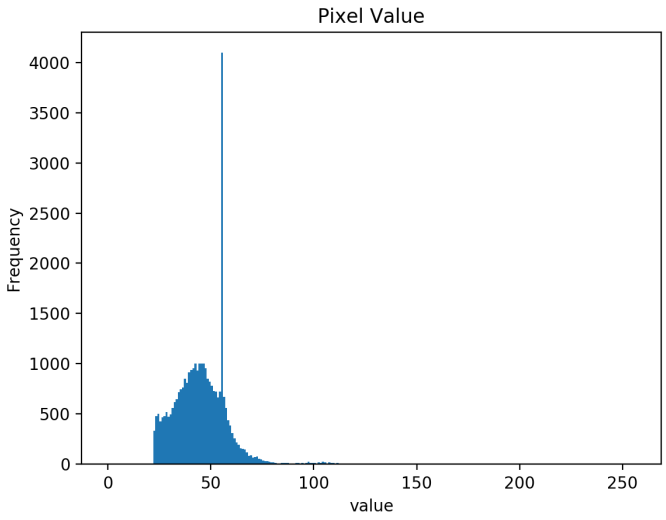


Figure 4.6: The histogram of the image after applying the corrected noise remover function.

Furthermore, small tests showed that the adaptive threshold is difficult to define in an optimal way. If the threshold is too low the eyelash might be reconstructed from edge pixels of the eyelash creating just a lighter eyelash, which is still darker than the iris in the background. If the threshold is higher, it might eliminate the eyelashes, but also be destructive to darker parts of the iris. Maybe this could also be solved if the reconstruction happened based on the neighbourhood and not just the most adjacent neighbour pixels.

During implementation the histograms were frequently inspected in order to ensure the results were as expected. Figure 4.7 shows a histogram with some eliminated pixel values. It turns out that the function used for generating histograms in python by default takes the range of the pixel values and splits that into as many bins as specified. As a result some of the bins cover a range of values that is entirely between two integer values and thus do not count any instances of pixel values as they are always integers between 0 and 225. This was solved by passing a specific range (0:256) as an argument and then the histogram was as described in Figure 4.6.

4.1.5 Histogram Equalisation

The histogram equalisation is applied to increase contrast. This is necessary to enhance the structures in the iris. This was implemented manually. Just as in the noise removal the lowest and highest bin values with more instances than a specified value are found. Based on the values found the histogram is stretched using the formula in Equation 4.3.

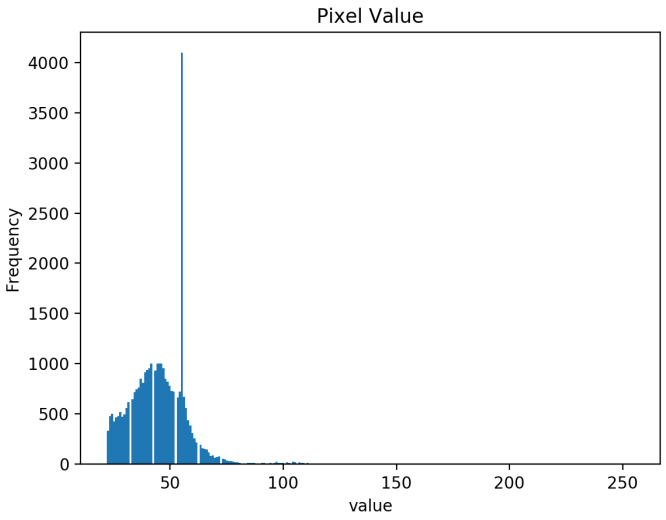


Figure 4.7: The histogram of the reconstructed image with eliminated values.

$$\text{New Pixel Value} = (\text{Pixel Value} - \text{Low Value}) \cdot \frac{255}{\text{High Value} - \text{Low Value}} \tag{4.3}$$

The pixels, which have values above 255 or below 0 after the formula is applied, are set to 255 or 0 respectively. The resulting image is shown in Figure 4.8, while Figure 4.9 shows the histogram after applying the formula to each pixel. However,

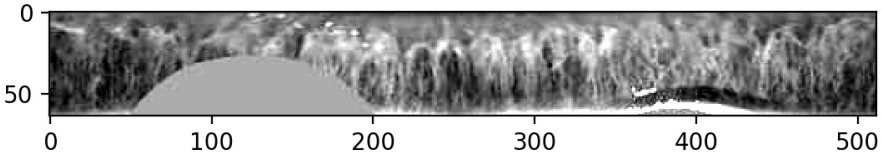


Figure 4.8: The image of the iris after applying the initial "equalise histogram" function.

it was discovered that the implemented method was actually histogram stretching which was mistakenly taken as the same as histogram equalisation. Though the two methods have somewhat the same effects histogram equalisation is better at ensuring a uniform spreading of the pixels across the histogram. This is done by identifying a transform of the bin values which causes the Cumulative Distribution Function (CDF) of the histogram to be as linear as possible, and apply it to the grey levels or bin values. The result of the histogram equalisation is showed in Figure 4.10 and Figure 4.11. When comparing the two images of the iris after applying the two different contrast adjustment methods it can be concluded that the histogram

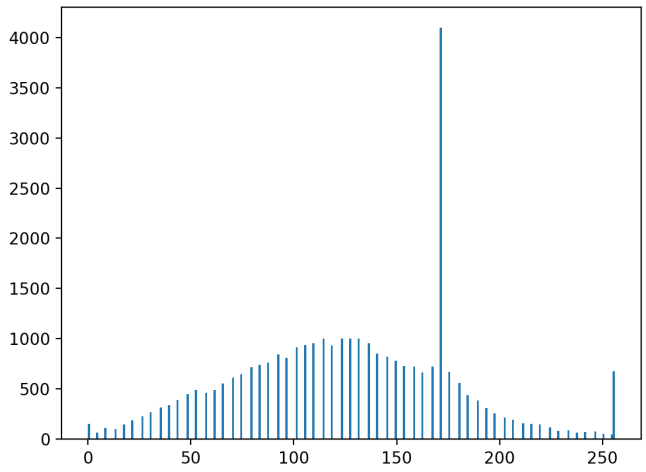


Figure 4.9: The histogram of the image after applying the initial "equalise histogram" function.

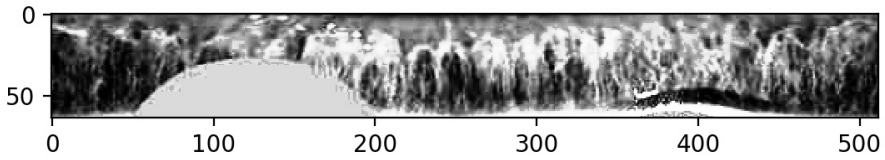


Figure 4.10: The image of the iris after applying the real equalisation of the histogram.

equalisation does indeed result in better contrast and a more enhanced and out spoken appearance of the structures in the iris than the histogram stretching does. Testing also showed that using the equalisation rather than the stretching increased the accuracy, this is further addressed in section 4.1.7.

4.1.6 Feature extraction

As it is difficult to extract all the complex structures in the iris the work presented uses the iris image as input. However, because the image in full will result in too large amounts of data to handle during classification feature extraction is done. Feature vectors are extracted using Haar wavelets in a wavelet decomposition. Haar wavelets are a fast and simple way to obtain wavelets. In Haar wavelets the decomposition is done by calculating the average of neighbouring pixels as well as half the difference between neighbouring pixels. Due to the way the method works the image is lowpass or hi-pass filtered two times in each of the possible combinations. The image of the greatest interest in this case is the one that has been lowpass filtered in both horizontal and vertical direction, this is referred to at LL. This image will be a

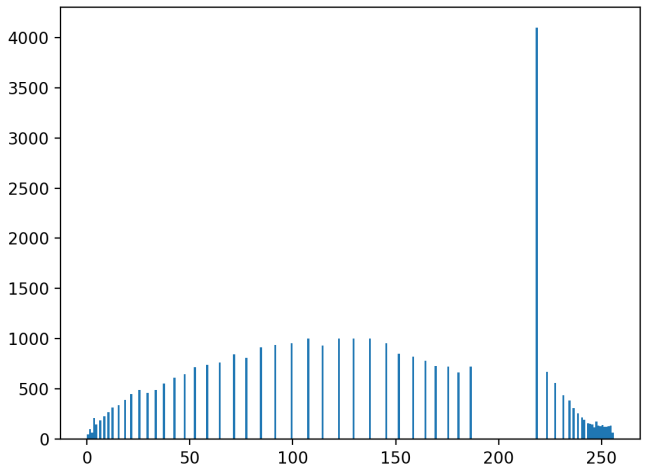


Figure 4.11: The histogram of the image after applying the real equalisation of the histogram.

compressed version of the original image, showing an approximation or the tendency of the original image. The other images created in one stage are HH, HL, LH all of which contain detail coefficients. HH is representing the diagonal detail, while HL presents horizontal detail and LH presents vertical detail. The LL will naturally appear in the upper left corner of the output matrix.

In order to compress the image sufficiently three stages or passes of Haar wavelet decomposition was carried out. The result is showed Figure 4.12. The resulting

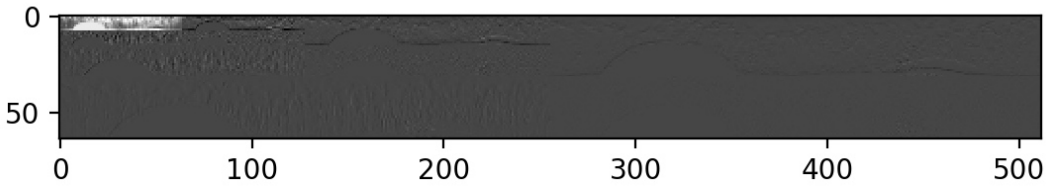


Figure 4.12: The result of the three level Haar wavelet decomposition. Note that the small light image in the upper left corner is the LL3, which is used onwards.

matrix of LL3 coefficients is then put into a feature vector, which is used for training the classifiers described in section 4.1.7.

4.1.7 Training and Classification

For classification different classifiers were tested. The best of the investigated classifiers are the K-Nearest-Neighbours (KNN), Linear Discriminant Analysis (LDA),

as well as a support vector machines (SVM) with linear or quadratic kernel. In the following sections the theory behind the different classifiers will briefly be described

K-Nearest Neighbour

KNN is one of the most simple supervised classification methods. It is a non-parametric method. This means that the train data is used "raw" and there are no parameters that are deducted from the data in order to form a model.

KNN utilises Bayes theorem in order to determine the probability of a sample belonging to a certain class. The probability of the sample belonging to different classes is found by looking at the density of datapoint belonging to the different classes among the k nearest neighbours. The probability of a class given the sample can be estimated by the formula in Equation 4.4.

$$p(C_k|\mathrm{x}) = \frac{K_k}{K} \tag{4.4}$$

Where C_k is a given class, x is the sample, K is the number of neighbours considered, and K_k is the amount of data points among the K neighbours belonging to the class C_k.

Finally, a class is assigned to the sample based on which class has the highest conditional probability.

The value K greatly influences the way the classifier performs. Therefore it is important to determine an optimal value of K. This may depend on the performance but also the amount of data that has to be compared with. In this project was found by determining which value of K for $K \in [1, 30]$ results in the best performance. This is done by training on a test set of five images and evaluating the train error. Figure 4.13 shows the misclassification error dependent on the value of K. As can be seen

Linear Discriminant Analysis

LDA is a parametric metho ich is trained by supervised learning. LDA is a classifier in which linear decision boundaries are learned based on training data. It is part of a more general group of linear discriminant functions. The decision boundary is found such that interclass variance is maximised while intra class variance is minimised. There are a range of possibilities for identifying the equation for the linear decision border eg. by using SVD or EVD.

Support Vector Machines

SVM identifies the optimal linear hyperplane as the decision boundary. The decision borders are found for linearly separable data. A separating hyperplane is not necessarily unique, however, the optimal hyperplane should be unique. The way the optimal linear decision boundary is found is by identifying the hyperplane, which has the maximal margin around it not intersecting with any data points is the training

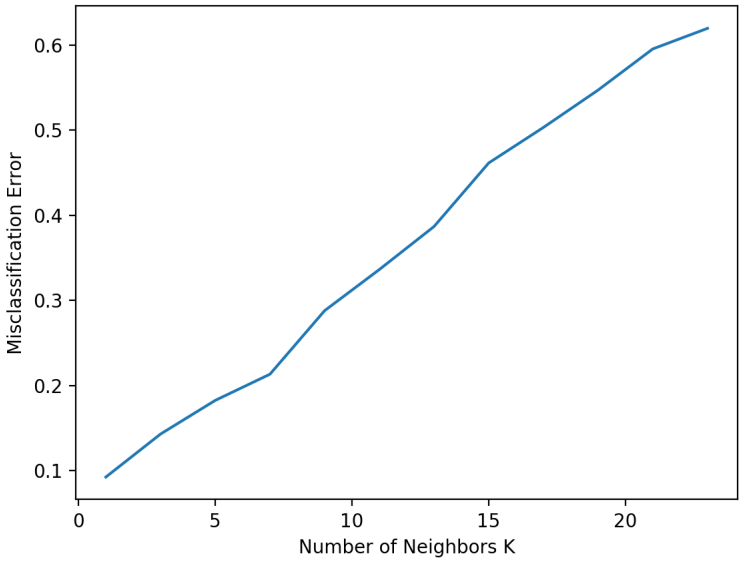


Figure 4.13: The misclassification error for the KNN as a result of the number of neighbours K used for the classifier.

set. Pin practice the hyperplane is found based on a subset of data points that lie close to the separating plane, those are called the support vectors.

However, in some cases the data is not linearly separable. For those cases Kernel Functions can be applied before identifying the the optimal hyperplane. The kernel function maps the original data from into a new space where it can be linearly separated.

Cross Validation

In this project the above mentioned classification methods were implemented. Just as in the work described in the article cross validation was carried out in order to test the performance of the classifiers. Five images per class were used for the cross validation. The cross validation done was a five fold cross validation. However, since some classes had too few images those classes were neglected. As mentioned in section 4.1.5 changing from histogram stretching to histogram equalisation affects the performance of the classifiers. In Table 4.1 the accuracies as well as the precision of the classifiers in both the case with histogram stretching and histogram equalisation is showed.

Considering the results for the KNN shown in Table 4.1 there seem to be a significant increase in the accuracy of the classifier. However, since it is based on such a limited amount of measurements it is probably more representative and valuable to consider the general trends. When comparing the two accuracies across all of the

	Histogram Stretching		Histogram Equalisation	
Classifier	MA (%)	2σ (%)	MA (%)	2σ (%)
KNN (k=1)	79	±6	91	±4
LDA	94	±2	95	±3
SVM (Linear)	85	±8	92	±2
SVM (quadratic)	77	±8	92	±4

Table 4.1: Performance of the different classifiers evaluated by five-fold cross validation. MA is for Mean Accuracy, and σ is the standard deviation

classifiers there seem to be evidence for an increase in accuracy when the histogram equalisation is used rather than the histogram stretching. Therefore the histogram equalisation is used from this point on. In Table 4.2 the accuracies achieved by the

	Cross Validation Accuracy (%)		
	Khan et al. [2017]	Replicated	
KNN (k=1)	95.1	91	91
LDA	94.28	94	**95**
SVM (Linear)	96.46	91	92
SVM (quadratic)	**97**	92	92

Table 4.2: Comparison of the performance of the different classifiers achieved by Khan et al. [2017] and the replicating work in this project. The first column under "Replicated" are accuracies based on data from 55 classes, while the second column is based on data from 91 classes.

work of Khan et al. [2017] are compared with the accuracies achieved in the replication of the work when conducting five-fold cross-validation on a set of 5 images from each class. The comparison reveals that the replication does not achieve the same accuracies. In general the original work seem to acquire higher accuracies, however, the original work does not present the variance of the mean accuracy calculated in relation to the cross validation, therefore it is impossible to verify how precise the classifiers were. Another thing to note is the fact that the quadratic SVM is the one that performs best according to the original work, while the replicated work suggests that the LDA classifier gives the highest accuracy. There are several factors that might contribute to the difference between the results presented in the original work and the result obtained through replication. One factor might be the amount of data. Though the amount of data samples used per class is the same, the amount of classes is not the same. Due to some issues with processing of the database, which are described in section 4.1.8, only a part of the database was available for testing. First only 55 classes were available and after a while 91 classes of the total 140 classes in the database. Some tests were done with both 55 and 91 classes and the results show that the amount of classes does have somewhat influence on the accuracies as is shown in Table 4.2. Another factor that might cause different results is the fact

that some steps of the processing where described quite briefly, thus the replication of the work might not be entirely aligned with the original work.

	Classification accuracy (%)		
Classifier	Train	Test	Validation
KNN (k=1)	100	96	97
LDA	100	98	97
SVM (Linear)	100	97	98
SVM (quadratic)	100	98	98

Table 4.3: The different accuracies achieved by the classifiers when trained on 70% of the data and tested and validated respectively on 15% of the data.

In order to examine the performance of the systems implemented in this work when trained on more data a test was conducted. During the test the classifiers were trained on 70% of the data and were tested on 15% of the data. Finally they were validated by testing on the remaining 15% of the data. The results are shown in Table 4.3. As the table shows the train accuracies for all the classifiers are 100%, which means the models are fitted perfectly to the train data. As can be expected the test and validation accuracies are lower than the train accuracy. It is worth noting that the accuracies are higher than the accuracies obtained in the cross validation based on only a smaller amount of data. Furthermore the training om more data results in accuracies higher than the ones obtained in cross validation in the original work. However, this test seemingly confirms the statement of the original work that the quadratic SVM performs best.

Though the achieved accuracies are quite high, the performance of these machine learning methods still cannot compete with the state of the art accuracies within iris classification and general biometric identification methods as described in chapter 3. Therefore, another method, Convolutional Neural Network (CNN), is investigated for iris classification in order to explore whether this method can classify with accuracies closer to the state of the art performances and comply with the requirement of an accuracy higher than 99%. The work on the implementation of a CNN for iris classification is described in section 4.3.

4.1.8 Challenges

As mentioned previously several issues were encountered during the implementation of the work described in this chapter. This section elaborates on a few of the issues.

Algorithm Description

Though overall steps of the processing algorithm are presented in the article by Khan et al. [2017], there are a lot of unclear details when considering the descriptions of the individual steps of the processing. This makes it difficult to replicate the work conducted in the article, and it results in a parts of the code being implemented

based on the best knowledge supplemented with ideas, while other things based on the indications from tests.

Processing Issues

Once the algorithm was implemented it had to be applied in order to process the entire database consisting of 3192 images of irises obtained from 70 subjects. The handling and processing of this amount of data proved quite troublesome. First of all the processing of one image using the implemented MATLAB libraries takes quite some time. The time of cause depends on the resources of the computer, however, it took a few minutes per. image. Also the process sometimes reported that it was too demanding for the laptops used. Therefore, the processing was moved to be carried out by a stationary computer with more power. This was a shared computer borrowed through an internet connection using TeamViewer, which meant that it was only accessible sometimes when it was free.

Another issue was the variation in the images being processed. Though the algorithm is implemented in a way that is somewhat flexible, which automatically identifies the areas of interest, it does sometimes fail. In some cases the algorithm is unable to identify the wanted area or it falsely identifies the wrong area as the area of interest. In those cases the algorithm would break or get stuck, which required manual restart or continuation of the processing.

The combination of the algorithm not running continuously and the fact that accessing the running processing was only possible sometimes made it difficult to ensure that the algorithm was running, and the process would waste large amounts of time being stuck on a single image. Therefore, processing of the full database was not successful, however, 2086 images in 91 classes of the full 140 classes were processed and used for the development and testing in this section.

4.2 Convolutional Nerual Networks

This section describes what a CNN is and what structure is used for creating the iris recognition CNN. The section is based mainly on Karpathy [2016] and Nielsen [2015].

4.2.1 Convolution

A CNN is type of Nerual Network that is especially good for image recognition. Instead of the input being fully connected it operates by using a kernel instead. A kernel can be seen as a small window which moves through the input image. It performs an operation known as a convolution where the coefficients, also known as weights, of the kernel are multiplied with the pixel it is covering and summing the value. A 1D example of this is shown in Figure 4.14 where a kernel of size 3 with weights $\begin{bmatrix} 1 & 0 & -1 \end{bmatrix}$ are convolved through bottom rows which produces the top

rows. Two other concepts are also shown in the example; stride and zero padding. Stride is the amount the kernel moves each time. The left example has a stride of 1 and the right example has a stride of 3. Zero padding is when pixels with value 0 are added at the edge of the image. This is done to ensure that the convolution can actually be made on all pixels, otherwise only one convolution would be possible in the example with a stride of 3. It also has the benefit of keeping the dimensions of the input image and output image the same when the stride is 1, as convolution otherwise shrinks the image.

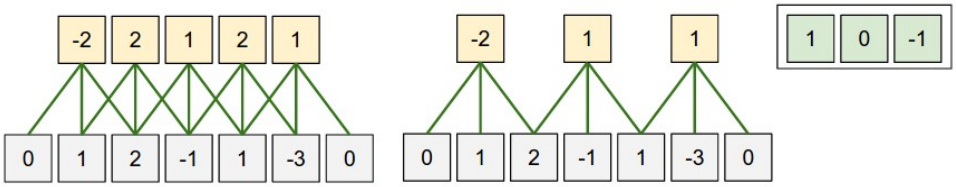


Figure 4.14: Example of a kernel in a CNN [Karpathy, 2016]

The output of the convolution creates a new image. This image becomes the input of an activation function. An activation can be thought of as how much should the neuron fire, e.g how active is it. A common activation function for CNN is the Rectified Linear Unit (ReLU), shown in Equation 4.5, which gives an output of x if the x is positive and 0 otherwise. This activation function has proved to be consistently better and faster for deep neural networks than the previously popular sigmoid activation function.

$$f(x) = max(0, x) \tag{4.5}$$

4.2.2 Activation function/feature maps

The output of the convolution is the hidden layer and is called a feature map. An example is shown in Figure 4.15 where a kernel of 5×5 is used on a input image of 28×28 to produce a feature map of 24×24. The dimensions can be calculated by using Equation 4.6 where W_2 is the width of the feature map, W_1 is the width of the input image, F is the size of the kernel and P is the amount of padding.

$$W_2 = \frac{(W_1 - F + 2P)}{S} + 1 \tag{4.6}$$

The weights of the kernel is what is being learned by the network. The kernel has what is known as shared weights and biases, as its the same weights that convolve through the whole image. This is shown in Equation 4.7 which is how a single neuron of the feature map in Figure 4.15 is computed. σ is an activation function, b is the shared bias, w are the weights of the kernel with l, m being their position and a is

the input pixel at the j, kth position .

$$\sigma(b + \sum_{l=0}^{4} \sum_{m=0}^{4} w_{l,m} * a_{j+l,k+m}) \tag{4.7}$$

In other words the kernel learns to detect a feature that can be found in the image, e.g. straight lines. As a single feature is not enough for recognition, multiple kernels are used to make multiple feature maps. These new feature maps become the input of another convolution layer that can learn higher level features, e.g. combining the vertical and horizontal features to create a kernel that detects edges. Each convolution adds a new level of abstraction.

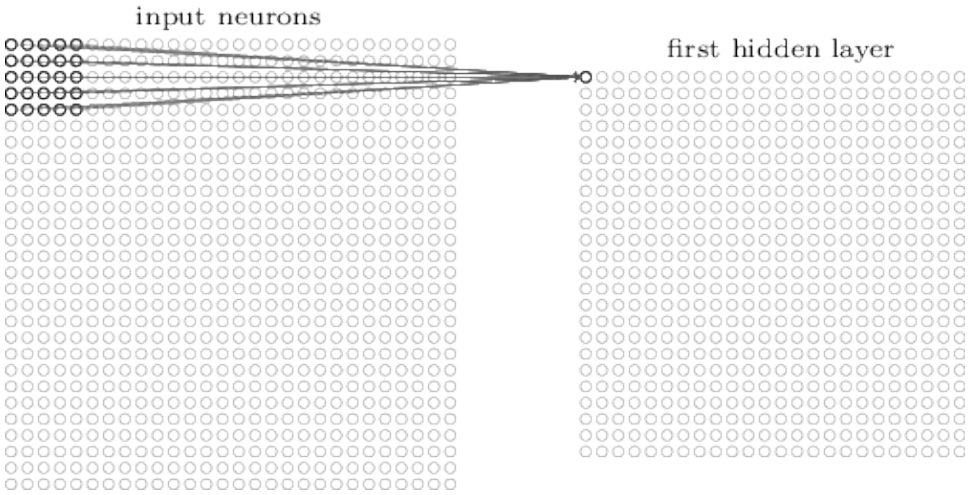


Figure 4.15: Example of a kernel in a CNN [Nielsen, 2015]

4.2.3 Pooling

Pooling is an operation often added after a convolutional layer. Like a convolution, a small windows moves through the image, except it does not convolve but simply looks at the pixels in its current location and outputs a single pixel depending of the type of pooling used. Maxpooling is a common type of pooling used in CNNs. An example of maxpooling with a window of size 2×2 and stride 2 is shown in Figure 4.16. This has the benefit of downscaling the spatial size of the image for faster computation and reducing overfitting.

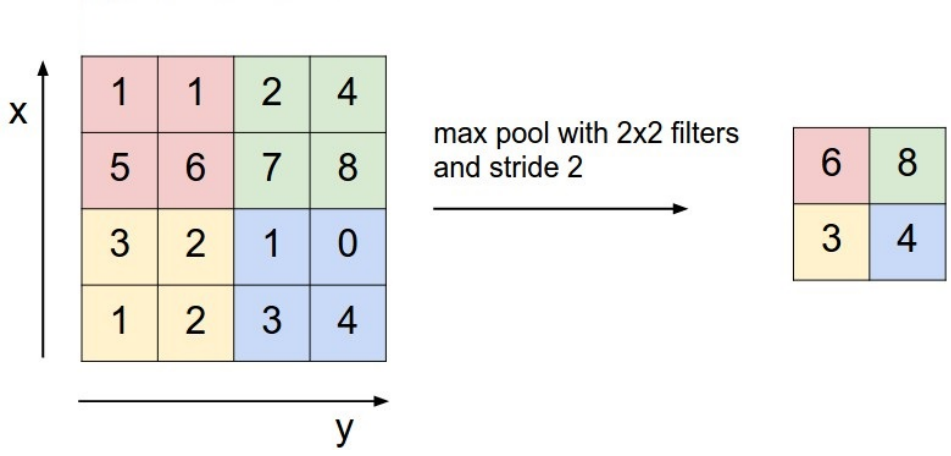


Figure 4.16: Example of maxpooling with size $2x2$ and stride 2 [Karpathy, 2016]

4.2.4 Classification

To use the features extracted from the convolutional layer one or more Fully Connected (FC) layers are added at the end of a network. In a FC layer all the neurons from the previous layer are connected to all of the neurons in current layer. This relationship is shown in Equation 4.8. The output of a single neuron is the sum of all the previous weights w_i and their bias b put through some activation function f.

$$f\left(\sum_i w_i x_i + b\right) \tag{4.8}$$

To classify N number of classes the last FC layer usually has the same amount of neurons as classes. A common activation function for multi class categorisation is the softmax function. It is a normalised exponential function that calculates the probability of each class over all possible classes. The outputs are in a range of $0-1$ and the sum of all class probabilities is 1. The function is described in Equation 4.9 where Z is a vector of weights from the previous layer, j is the output class and K is the total amount of classes.

$$\sigma(Z)_j = \frac{e^{Z_j}}{\sum_{k=1}^{K} e^{Z_k}} \tag{4.9}$$

A common loss function that is used to optimise the softmax function is a cross-entropy loss function. Cross-entropy is defined as Equation 4.10 where q is the estimated distribution and p is the actual distribution. In other words it's a measurement of how far the estimated classes are from the true classes.

$$H(p,q) = -\sum_x p(x) log(q(x)) \tag{4.10}$$

The estimated classes are softmax function, in other words $q = \sigma(Z)_j$. The loss function that is optimised is then Equation 4.11.

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=1}^{K} 1\{y_i = j\} log\left(\frac{e^{\theta_j}}{\sum_{k=1}^{K} e^{\theta_k}}\right)\right] + \frac{\lambda}{2}\sum_{i=1}^{K}\sum_{j=0}^{n}\theta_{ij}^2 \tag{4.11}$$

θ is the weights that are randomly set and are learned through back propagation. K is the number of classes and n is the number of labelled training samples. y_i is the label and $1\cdot$ is a function that is 1 when the input is true and 0 when it is not. The second term is weight decay regularization term that reduces the magnitudes of weights and prevents it from overfitting.

4.2.5 Network Construction

To construct and train the networks, Keras is used with Tensorflow as a backend. Keras is an API written in Python, able to run on three different backends. With a high level of modularity by having separate models added on top of each other construction and overview of a network is simplified. By using Tensorflow with Keras it is possible to utilise a Conda compatible Graphics Processing Unit (GPU) which decreases the training greatly from using only a Central Processing Unit (CPU). Keras has several example programs from which inspiration and techniques can be gathered, but also contains well known specific models to implement and configure in a new network. This includes the VGG16 network including weights trained on the ImageNet database with 2000 different classes.

4.3 Iris Recognition

The structure for the iris CNN is based on work done by Al-Waisy et al. [2017b]. They made a structure named IrisConvNet that uses a segmented and normalised iris image, described in section 4.1 to classify. For training they tried three iris databases; SDUMLA-HMT, CASIA-Iris-V3 and IITD. The CNNs were only trained and tested on one database at a time. The normalised images were originally 256×70 and 256×135. The images were resized to be squares the size of the shortest dimension. To produce more samples data augmentation was performed. Five $64x64$ images were generated by cropping the rescaled corresponding to the four corners and center. These images were also flipped horisontally, totalling in 10 images augmented from a single image. Different amounts of feature maps were tested to get the best classifier. The general structure was the same. They tried CNNs with 3-5 convolutional layers with 2x2 maxpooling in between. The first kernel was 3x3 and the rest were 5x5. At the end there are two FC layers and a prediction layer. The input sizes 64×64 or 128×128. They varied the amount of feature maps with the first layer having 6 feature maps and the following layer having gradual increases each layer. The best performing configuration for the 64×64 images achieved between 99.62% and

100% accuracy between the databases. It had four convolutional layers with 6, 32, 64 and 256, respectively, feature maps. For input images of size 128×128 the best configuration achieved between 99.41% and 100% accuracy. It had five layers with 6, 16, 32, 64 and 256, respectively, feature maps.

4.3.1 Parameters and Settings

They had various parameters and settings for their network that will be listed for ease of replication. They deemed 500 epochs to an appropriate training time before it started overfitting. They used AdaGrad optimiser with a learning rate of 10^{-2} with a weight decay of 0.0005. The ReLU activation function was used for convolutions and the two FC layers, while the classifying layer used softmax and cross-entropy as the loss function. They also had a dropout layer of 0.5 in the two FC. This means that during each iteration each node has a 50% probability of being ignored which helps prevent interdependencies among the nodes to appear and reduces overfitting. Zero padding of 1 pixel on the input image which changes it's dimensions from 64x64 to 66x66. The number of neurons in the FC layers were not specified except for the classifying layer having the same amount of neurons as classes.

4.3.2 Project Solution

As previously stated the solution for the network is heavily based on the structure used by Al-Waisy et al. [2017b], but with alterations. An overview of the design is shown in Figure 4.17. The learning rate was set to 10^{-3} as 10^{-2} turned out to be to big. The original normalised iris were 64×512 so they were resized to 64×64. The data augmented images then became 58×58. The two FC layers were set to 1024 as they had not been specified by Al-Waisy et al. [2017b]. The same optimizer and hyper parameters were used.

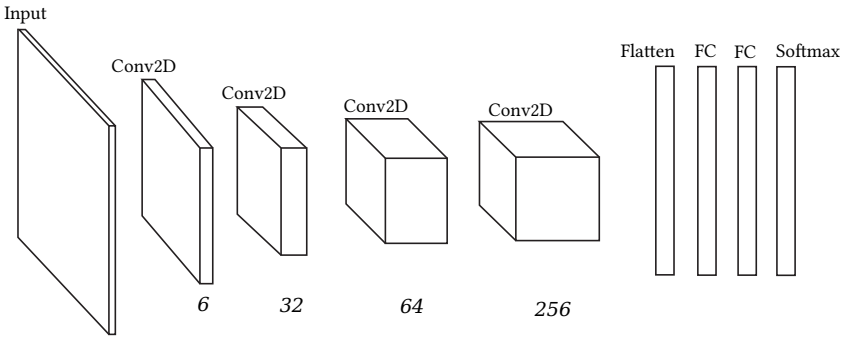


Figure 4.17: Overview of the iris recognition network design

A more detailed description of the network is shown in Table 4.4.

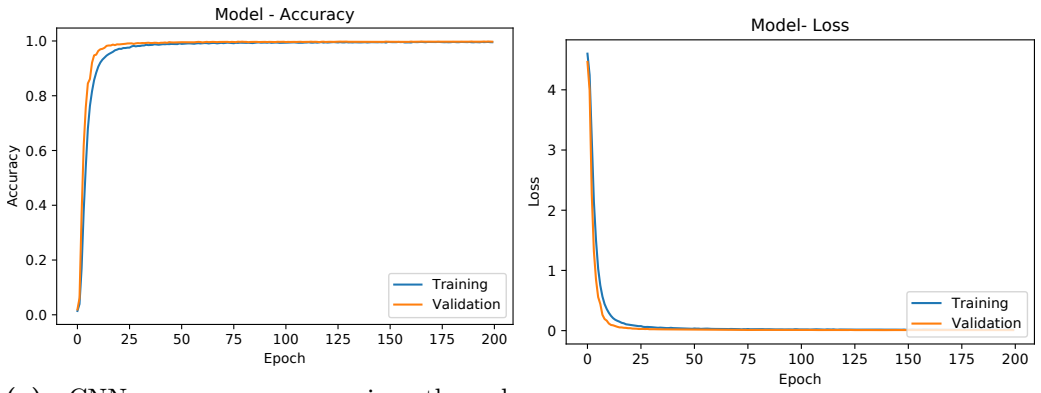
Table 4.4: Detailed description of network design and settings

Layer Type	Feature Map Size	Kernel/Pool Size	Activation	Other
Conv2D	6	3×3	ReLU	
MaxPooling2D		2×2		
Conv2D	32	5×5	ReLU	
MaxPooling2D		2×2		
Conv2D	64	5×5	ReLU	
MaxPooling2D		2×2		
Conv2D	256	5×5	ReLU	
Flatten				
Dense	1024		ReLU	
Dropout				0.5
Dense	1024		ReLU	
Dropout				0.5
Dense	Amount of Classes		Softmax	

To train and evaluate the model, the dataset is split into three sets; training, validation and testing. The model is trained on the training set while being evaluated on the validation set. The models parameters are than adjusted on the basis of the validation set. Finally it is evaluated on the testing set. The split is 70%, 15%, 15%. The accuracy is defined simply by:

$$Accuracy = \frac{CorrectCategorisation}{Totalnumber} * 100 \tag{4.12}$$

The results achieved with this network is an accuracy of 99.7%. This is achieved with a batch size of 128 and 200 epochs. In Figure 4.18 the accuracy and loss progression is shown.



(a) CNN accuracy progression through epochs

(b) CNN loss progression through epochs

Figure 4.18: Accuracy and loss progression for the iris recognition CNN.

4.4 Face Recognition

To make a face recognition CNN an already well known model is used. The network is based on the VGG16 model, including the weights pre-trained on the ImageNet database with 2000 classes. The model is split into five blocks. The input size of the images are $64 \times 64 \times 3$ and are from the Labeled Faces in the Wild (LFW) database. The network is made and well known for image classification, and with the pre-trained weights from ImageNet, it is a suitable solution for face recognition [Simonyan and Zisserman, 2015]. An overview of the network is shown Figure 4.19 and a more detailed setup description is shown in Table 4.5. The network gets an accuracy of 99.35% on the LFW database. Figure 4.20 shows the accuracy and loss progression.

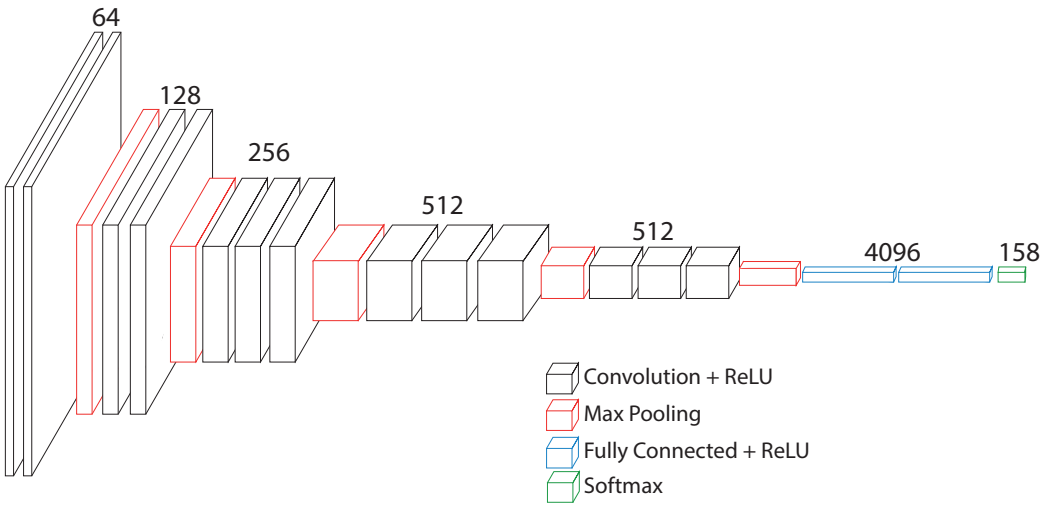
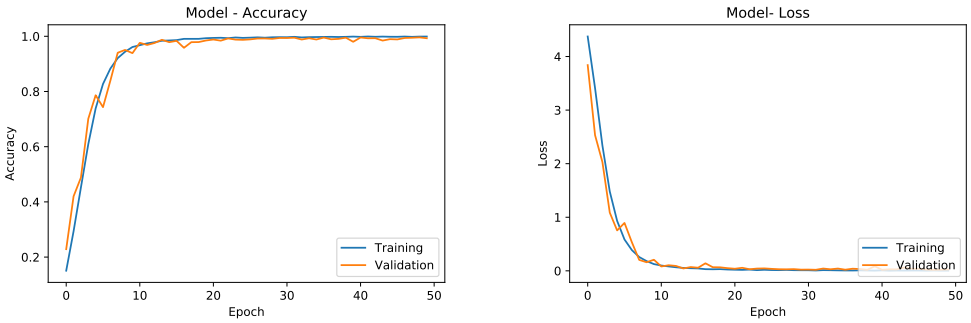


Figure 4.19: Overview of the VGG16 CNN used for face recognition. Only the depth of the layers is known and noted. The input is excluded from the figure.

Table 4.5: Detailed description of VGG16 design. The input is $64 \times 64 \times 3$ images.

Layer Type	Feature Map Size	Kernel/Pool Size	Activation	Other
Block 1				
Conv2D	64	3×3	ReLU	
Conv2D	64	3×3	ReLU	
MaxPooling2D		2×2		
Block 2				
Conv2D	128	3×3	ReLU	
Conv2D	128	3×3	ReLU	
MaxPooling2D	2×2	5×5	ReLU	
Block 3				
Conv2D	256	3×3	ReLU	
Conv2D	256	3×3	ReLU	
Conv2D	256	3×3	ReLU	
MaxPooling2D	2×2			
Block 4				
Conv2D	512	3×3	ReLU	
Conv2D	512	3×3	ReLU	
Conv2D	512	3×3	ReLU	
MaxPooling2D	2×2			
Block 5				
Conv2D	512	3×3	ReLU	
Conv2D	512	3×3	ReLU	
Conv2D	512	3×3	ReLU	
MaxPooling2D	2×2	3×3	ReLU	
Classification				
Flatten				
Dense	4096		ReLu	
Dropout				0.5
Dense	4096		ReLu	
Dropout				0.5
Dense	Amount of Classes		Softmax	



(a) Face recognition CNN accuracy progression through epochs

(b) Face recognition CNN loss progression through epochs

Figure 4.20: Accuracy and loss progression for the face recognition VGG16 CNN.

4.5 Network Fusion

To be able to combine iris and face verification, a combination of the results of the two networks must be made. Because of time constraints it was not possible to implement this functionality. Instead, the theory of network fusion is presented in this section.

To make this two-stream CNN the networks are first training individually and can then be connected using a dense fully connected layers connecting the streams from the two networks. After the fully connected layer a classification is necessary. This can be done using a softmax layer after the dense layer [Eitel et al., 2015]. This should in theory increase the accuracy in ID verification in a potential instance where verification is required. The idea of this design is shown in **??**.

To be able to reuse as much as possible, the two databases used with the networks separately are also used in the fusion of the networks instead of using a multimodal database as presented in chapter 2. This means that a combination of the two databases is necessary as the data have not been collected by the same groups or from the same people. The databases does not have the same amount of classes, therefore the larger database is reduced also reducing the entire amount of usable data.

4.5.1 Multimodal Database

A fusion net should be applied on a multimodal biometric database. Ideally the database used in this project should consist of face and iris images obtained from the same subjects captured with the same camera complying with the requirements set in chapter 3.

However, even though literature suggests that chimeric databases are less adequate than genuine multimodal biometric databases, the multimodal database used during the work with information fusion is synthetically created. This because a

few reasons. First of all, as section 2.4 mentions there are only a limited amount of multimodal biometric databases available containing both iris and face data, and to the extend of the knowledge gain from the research presented in section 2.3 there is only one of the databases which is obtained using mobile devices namely the MobBio database. For this database the Asus Transformer Pad TF 300T is used, which has a camera of 8MP, thus comparable to the iPhone 5s used for the caption of the Warsaw-BioBase used for the iris identification methods presented in section 4.1 and section 4.3 [Sequeira et al., 2014]. However, despite several attempts of contacting the responsible for the creation of the database it was not possible to establish communication or gain access to the database. Therefore the available way to obtain a multimodal database with mobile device images is to create it synthetically. Secondly, because the goal is to compare the performance of the fused CNNs with the individual CNNs on the face and iris data respectively, it is desirable to test on the same data. Therefore a databased was created by combining the LFW and the Warsaw-BioBase.

The database was created by combining iris classes arbitrarily with an face class for as many classes as there were classes available from both the iris and the face dataset. Before combining the datasets the classes with ten or less samples were neglected. The samples inside the new classes was made by combining an arbitrary iris image with an face image for as many samples there are samples available of both the face and iris.

Marike: Shagen, lyder dette rigtigt?

Chapter 5

Evaluation

The evaluation is done in regards to the final networks created and their fulfilment of the requirements. As stated in chapter 3, the networks must have a precision of 99% or higher to be comparable to the state of the art networks being used now.

In order to comply with the demands for the input data used for training the systems mentioned in chapter 3, the database Warsaw BioBase created by Trokielewicz [2016], which contains VL images of irises obtained with a smart phone is used. The smart phone used was an iPhone 5s with an 8 megapixel camera, which is comparable to the resolution of the front facing cameras in the newest phones on the market.

The first attempt of obtaining an acceptable accuracy was a unimodal iris classification system using regular machine learning. With this approach it was not possible to reach as high an accuracy as of the state of the art. The best performance achieved was when using a polynomial kernel for the SVM, which resulted in an accuracy of 98%. However, the model had a training accuracy of 100%, thus maybe the model is over fitting. A way to improve this could be training on more data in order to get a more generalised model. Another thing that could be investigated is the polynomial kernel applied, which in this case was defined as a 3^{rd} degree polynomial. However, it is possible that a polynomial of a different degree would map the data to a space where it was better separable by linear hyperplanes.

Nonetheless, because of this and the objective of the project, deep learning was taken into consideration as a solution. A CNN consisting of 14 layers in total was implemented for iris classification. The CNN was trained on the same database used for the machine ing method implemented and managed to reach an accuracy of 99.7%, which sa s the requirement. The design of this is shown in chapter 4.

tjek at denne er tilsvarende for hvad vi har opnaaet

The face recognition CNN is based on the VGG16 network and uses the LFW database for testing. By using the pre-trained weights from ImageNet trained on 2000 classes it is po to reach an accuracy of 99.35%. The design of this is shown in chapter 4.

Chapter 6

Conclusion

Chapter 7

Future Work

This chapter cast lights on some of the considerations and potential improvements that could be considered during prospective work.

Though fusion of the two modalities was accomplished there are still possibilities to investigate. In chapter 1 different benefits of multimodal biometric systems are mentioned, some of them being increased safety, universality and reduced noise sensitivity. The latter two being important points in regard to usability of the system. In some cases people might not be able to provide certain data, or the data provided might be noisy because of imperfect data capture.

In the current implementation of the fused CNN the left and right irises of the same subject are considered separately as two different classes. This was done because of the fact that the two irises are structurally different even thought they might have the same colour. In future work a fusion net could be designed in such a way that it fuses three inputs and thus fuses face data with data from both irises of the same person. This might be a way to obtain more information from one subject with the means that are already used. This could make the system more secure, but also increase the chances of classification when the inputs are noisy.

Another point that might be worth considering for future work is the database used. The dataset used for the fusion was mentioned a synthetic database comprising the two databases LFW and Warsaw-Biobase. However, in literature it is suggested that this is not representative of how a real multimodal biometric database obtained from the same subject because of the natural correlation between the biometric traits recorded. Therefore the performance implemented system might be most realistically represented when trained on a genuine multimodal biometric dataset.

Furthermore, as CNNs are best trained on large data set it might be interesting to train on a larger dataset in order to investigate whether this improves the results.

Bibliography

Al-Waisy, A. S., Qahwaji, R., Ipson, S., and Al-Fahdawi, S., 2017. A multimodal biometrie system for personal identification based on deep learning approaches. *2017 Seventh International Conference on Emerging Security Technologies (EST)*, pp. 163–168. doi: 10.1109/EST.2017.8090417.

Al-Waisy, A. S., Qahwaji, R., Ipson, S., Al-Fahdawi, S., and Nagem, T. A., 2017. A multi-biometric iris recognition system based on a deep learning approach. *Pattern Analysis and Applications*, (0123456789), pp. 1–20. ISSN 14337541. doi: 10.1007/s10044-017-0656-1. Available at: <https://doi.org/10.1007/s10044-017-0656-1>.

Bazrafkan, S., Thavalengal, S., and Corcoran, P., 2017. An End to End Deep Neural Network for Iris Segmentation in Unconstraint Scenarios. dec. Available at: <http://arxiv.org/abs/1712.02877>.

Bowyer, K. W., Hollingsworth, K. P., and Flynn, P. J. Chapter 2 A Survey of Iris Biometrics Research: 2008–2010. In *Handbook of Iris Recognition*, pp. 23–61. 2016. ISBN 978-1-4471-6782-2. doi: 10.1007/978-1-4471-6784-6. Available at: <http://link.springer.com/10.1007/978-1-4471-6784-6>.

Chen, C.-H. and Te Chu, C., 2005. Fusion of Face and Iris Features for Multimodal Biometrics. pp. 571–580. ISSN 03029743. doi: 10.1007/11608288_76. Available at: <http://link.springer.com/10.1007/11608288{_}76>.

Connaughton, R., Bowyer, K. W., and Flynn, P. J. Chapter 17 Fusion of Face and Iris Biometrics. In *Handbook of Iris Recognition*, pp. 397–415. 2016. ISBN 978-1-4471-6782-2. doi: 10.1007/978-1-4471-6784-6. Available at: <http://link.springer.com/10.1007/978-1-4471-6784-6>.

Daugman, J., 1993. High confidence visual recognition of persons by a test of statistical independence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15 (11), pp. 1148–1161. ISSN 01628828. doi: 10.1109/34.244676. Available at: <http://ieeexplore.ieee.org/document/244676/>.

Dessimoz, D., Richiardi, J., Champod, C., and Drygajlo, A., 2007. Multimodal biometrics for identity documents ({A figure is presented}). *Forensic Science International*, 167(2-3), pp. 154–159. ISSN 03790738. doi: 10.1016/j.forsciint.2006.06.037.

Eitel, A., Springenberg, J. T., Spinello, L., Riedmiller, M., and Burgard, W. Multimodal deep learning for robust RGB-D object recognition. In *IEEE International Conference on Intelligent Robots and Systems*, volume 2015-Decem, pp. 681–687, jul 2015. ISBN 9781479999941. doi: 10.1109/IROS.2015.7353446. Available at: <http://arxiv.org/abs/1507.06821>.

Fierrez, J., Galbally, J., Ortega-Garcia, J., Freire, M. R., Alonso-Fernandez, F., Ramos, D., Toledano, D. T., Gonzalez-Rodriguez, J., Siguenza, J. A., Garrido-Salas, J., Anguiano, E., Gonzalez-de Rivera, G., Ribalda, R., Faundez-Zanuy, M., Ortega, J. A., Cardeñoso-Payo, V., Viloria, A., Vivaracho, C. E., Moro, Q. I., Igarza, J. J., Sanchez, J., Hernaez, I., Orrite-Uruñuela, C., Martinez-Contreras, F., and Gracia-Roche, J. J., 2010. BiosecurID: A multimodal biometric database. *Pattern Analysis and Applications*, 13(2), pp. 235–246. ISSN 14337541. doi: 10.1007/s10044-009-0151-4.

Fierrez, J., Ortega-Garcia, J., Torre Toledano, D., and Gonzalez-Rodriguez, J., 2007. Biosec baseline corpus: A multimodal biometric database. *Pattern Recognition*, 40 (4), pp. 1389–1392. ISSN 00313203. doi: 10.1016/j.patcog.2006.10.014.

Fierrez, J., Morales, A., Vera-Rodriguez, R., and Camacho, D., 2018. Multiple classifiers in biometrics. part 1: Fundamentals and review. *Information Fusion*, 44 (November 2017), pp. 57–64. ISSN 15662535. doi: 10.1016/j.inffus.2017.12.003. Available at: <https://doi.org/10.1016/j.inffus.2017.12.003>.

Ho, T. K., Hull, J. J., and Srihari, S. N., 1994. Decision Combination in Multiple Classifier Systems. *IEEE Trans. Pattern Anal. Mach. Intell.*, 16(1), pp. 66–75. ISSN 0162-8828. doi: http://dx.doi.org/10.1109/34.273716.

Huang, G. B., Ramesh, M., Berg, T., and Learned-Miller, E., 2007. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. *University of Massachusetts Amherst Technical Report*, 1, pp. 07–49. ISSN 1996756X. doi: 10.1.1.122.8268. Available at: <http://vis-www.cs.umass.edu/lfw/lfw.pdf>.

Karpathy, A., 2016. Convolutional Neural Networks for Visual Recognition. *Stanford University*, pp. 1–21. Available at: <http://cs231n.github.io/classification/>.

Khan, F. F., Akif, A., and Haque, M. A., 2017. Iris Recognition using Machine Learning from Smartphone Captured Images in Visible Light. pp. 26–28.

Kuehlkamp, A. and Bowyer, K. W., 2016. An analysis of 1-to-first matching in iris recognition. *2016 IEEE Winter Conference on Applications of Computer Vision, WACV 2016*. doi: 10.1109/WACV.2016.7477687.

Lecun, Y., Bengio, Y., and Hinton, G., 2015. Deep learning. *Nature*, may, 521(7553), pp. 436–444. ISSN 14764687. doi: 10.1038/nature14539. Available at: <http://www.nature.com/articles/nature14539>.

Libor Masek, P. K., 2003. *MATLAB Source Code for a Biometric Identification System Based on Iris Patterns*, The School of Computer Science and Software Engineering, The University of Western Australia. Available at: <http://www.peterkovesi.com/studentprojects/libor/sourcecode.html>.

Luhadiya, R. and Khedkar, A., 2017. Iris detection for person identification using multiclass SVM. *2016 IEEE International Conference on Advances in Electronics, Communication and Computer Technology, ICAECCT 2016*, pp. 387–392. doi: 10.1109/ICAECCT.2016.7942619.

Misztal, K., Tabor, J., and Saeed, K. A new algorithm for rotation detection in iris pattern recognition. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 7564 LNCS, pp. 135–145, 2012. ISBN 9783642332593. doi: 10.1007/978-3-642-33260-9-11.

Nielsen, M., 2015. Neural Networks and Deep Learning. *Determination Press*. Available at: <http://neuralnetworksanddeeplearning.com/index.html>.

Ortega-Garcia, J., Fierrez, J., Alonso-Fernandez, F., Galbally, J., Freire, M. R., Gonzalez-Rodriguez, J., Garcia-Mateo, C., Alba-Castro, J. L., Gonzalez-Agulla, E., Otero-Muras, E., Garcia-Salicetti, S., Allano, L., Ly-Van, B., Dorizzi, B., Kittler, J., Bourlai, T., Poh, N., Deravi, F., Ng, M. N., Fairhurst, M., Hennebert, J., Humm, A., Tistarelli, M., Brodo, L., Richiardi, J., Drygajlo, A., Ganster, H., Sukno, F. M., Pavani, S. K., Frangi, A., Akarun, L., and Savran, A., 2010. The multiscenario multienvironment biosecure multimodal database (BMDB). *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(6), pp. 1097–1111. ISSN 01628828. doi: 10.1109/TPAMI.2009.76.

Percy, O. Iris localization using Daugman ' s algorithm. *Blekinge Institute of Technology*, pp. 1–48. Available at: <http://www.diva-portal.org/smash/get/diva2:831173/FULLTEXT01.pdf>.

Petrovska-Delacrétaz, D., Lelandais, S., Colineau, J., Chen, L., Dorizzi, B., Ardabilian, M., Krichen, E., Mellakh, M. A., Chaari, A., Guerfi, S., D'Hose, J., and Amor, B. B., 2008. The IV2 multimodal biometric database (including Iris, 2D, 3D, stereoscopic, and talking face data), and the IV2-2007 evaluation campaign. *BTAS 2008 - IEEE 2nd International Conference on Biometrics: Theory, Applications and Systems*, 00, pp. 3–9. doi: 10.1109/BTAS.2008.4699323.

Rattani, A. and Derakhshani, R., 2017. Ocular biometrics in the visible spectrum: A survey. *Image and Vision Computing*, 59, pp. 1–16. ISSN 02628856. doi: 10.1016/j.imavis.2016.11.019. Available at: <http://dx.doi.org/10.1016/j.imavis.2016.11.019>.

Rifaee, M., Abdallah, M., and Okosh, B., 2017. A Short Survey for Iris Images Databases. *international journal of multimedia & its applications*, (April). Available at: <https://www.researchgate.net/publication/316093004{_}A{_}Short{_}Survey{_}for{_}Iris{_}Images{_}Databases>.

Ross, A. and Jain, A., 2003. Information Fusion in Biometrics. 24(13), pp. 2115–2125.

Saha, R., Kundu, M., Dutta, M., Majumder, R., Mukherjee, D., Pramanik, S., Thakur, U. N., and Mukherjee, C., 2017. A Brief Study on Evolution of Iris Recognition System. *Information Technology, Electronics and Mobile Communication Conference (IEMCON), 2017 8th IEEE Annual*, pp. 685–688. Available at: <http://ieeexplore.ieee.org.ezproxy.psu.edu.sa/stamp/stamp.jsp?arnumber=8117234>.

Sequeira, A. F., Monteiro, J. C., Rebelo, A., and Oliveira, H. P., 2014. MobBIO: A Multimodal Database Captured with a Portable Handheld Device. *9th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, (c), pp. 1–14.

Simonyan, K. and Zisserman, A., 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. *International Conference on Learning Representations (ICRL)*, sep, pp. 1–14. ISSN 09505849. doi: 10.1016/j.infsof.2008.09.005. Available at: <http://arxiv.org/abs/1409.1556>.

Sun, Y., Wang, X., and Tang, X. Deep learning face representation from predicting 10,000 classes. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1891–1898, 2014a. ISBN 9781479951178. doi: 10.1109/CVPR.2014.244. Available at: <http://mmlab.ie.cuhk.edu.hk/pdf/YiSun{_}CVPR14.pdf>.

Sun, Y., Wang, X., and Tang, X. Deep Learning Face Representation by Joint Identification-Verification. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1891–1898, 2014b. ISBN 9781479951178. doi: 10.1109/CVPR.2014.244. Available at: <https://arxiv.org/pdf/1406.4773.pdfhttp://arxiv.org/abs/1406.4773>.

Sun, Y., Liang, D., Wang, X., and Tang, X., 2015. DeepID3: Face Recognition with Very Deep Neural Networks. Available at: <https://arxiv.org/pdf/1502.00873.pdfhttp://arxiv.org/abs/1502.00873>.

Trokielewicz, M. Iris recognition with a database of iris images obtained in visible light using smartphone camera. In *2016 IEEE International Conference on Identity, Security and Behavior Analysis (ISBA)*, pp. 1–6. IEEE, feb 2016. ISBN 978-1-4673-9727-8. doi: 10.1109/ISBA.2016.7477233. Available at: <http://ieeexplore.ieee.org/document/7477233/>.

Uka, A., Roçi, A., and Koç, O., 2017. Improved Segmentation Algorithm and Further Optimization for Iris Recognition. *IEEE EUROCON 2017 -17th International Conference on Smart Technologies*, (July), pp. 6–8.

Wang, Z., Han, Q., Niu, X., and Busch, C., 2009. Feature-level fusion of Iris and face for personal identification. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 5553 LNCS(PART 3), pp. 356–364. ISSN 03029743. doi: 10.1007/978-3-642-01513-7_38.

Wechsler, H. *Reliable face recognition methods: System design, implementation and evaluation*. Springer US, Boston, MA, 2007. ISBN 038722372X. doi: 10.1007/978-0-387-38464-1. Available at: <http://link.springer.com/10.1007/978-0-387-38464-1https://link-springer-com.zorac.aub.aau.dk/book/10.1007{%}2F978-0-387-38464-1{#}about>.

Yin, Y., Liu, L., and Sun, X. SDUMLA-HMT: A multimodal biometric database. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 7098 LNCS, pp. 260–268, 2011. ISBN 9783642254482. doi: 10.1007/978-3-642-25449-9_33.

Zhao, Z. and Kumar, A. An accurate iris segmentation framework under relaxed imaging constraints using total variation model. In *Proceedings of the IEEE International Conference on Computer Vision*, volume 2015 Inter, pp. 3828–3836. IEEE, dec 2015. ISBN 9781467383912. doi: 10.1109/ICCV.2015.436. Available at: <http://ieeexplore.ieee.org/document/7410793/>.

Zhao, Z. and Kumar, A., 2017. Towards More Accurate Iris Recognition Using Deeply Learned Spatially Corresponding Features. *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 3829–3838. doi: 10.1109/ICCV.2017.411. Available at: <http://ieeexplore.ieee.org/document/8237673/>.