Deep Learning for 3D Hand Biometric Systems

Doctoral Dissertation submitted to the
Faculty of Informatics of the Università della Svizzera Italiana
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

presented by
Jan Svoboda

under the supervision of
Prof. Michael M. Bronstein and Dr. Jonathan Masci

July 2020

Dissertation Committee

Prof. Kai Hormann	USI Lugano, Switzerland
Prof. Jürgen Schmidhuber	USI Lugano, Switzerland
Prof. Mark Nixon	University of Southampton, United Kingdom
Dr. Andrew Fitzgibbon	Microsoft Research Cambridge, United Kingdom

Dissertation accepted on 17 July 2020

Research Advisor	Co-Advisor
Prof. Michael M. Bronstein	**Dr. Jonathan Masci**

PhD Program Director

Prof. Walter Binder

I certify that except where due acknowledgement has been given, the work presented in this thesis is that of the author alone; the work has not been submitted previously, in whole or in part, to qualify for any other academic award; and the content of the thesis is the result of work which has been carried out since the official commencement date of the approved research program.

Jan Svoboda
Lugano, 17 July 2020

To all who ever believed in me.

I have no special talents. I am only passionately curious.

Albert Einstein

Abstract

Hands are an indispensable part of human bodies used in our everyday life to express ourselves and manipulate the surrounding world. Moreover, a hand contains highly-unique characteristics that allow for distinguishing among different individuals. Though fingerprints are widely-known for this, the hand also has a unique geometric shape, palmprint, and vein structure. The shapes of a hand and its parts have been studied as biometric identifiers in the past, yielding state-of-the-art approaches in the field of Biometrics; however, these are limited in their practicality. This thesis aims at overcoming the limitations and improving the performance of hand-shape-based biometric systems, taking advantage of the latest developments in Deep Learning and 3D Sensing. In particular, such systems can be improved either by means of biometric fusion or by working directly with the hand shape.

Biometric fusion allows for improvement in hand-shape-based biometric systems by simultaneously utilizing other modalities such as palmprints or fingerprints. To this end, we propose novel deep-learning based approaches to contact-free palmprint and fingerprint recognition. These improvements achieve state-of-the-art results on multiple standard benchmarks in both palmprint recognition and in processing latent fingerprint impressions.

Additionally, current 3D-sensing technologies provide low-cost sensors that allow us to capture the surface of the hand as a 3D point cloud. This type of data is often noisy but can be dealt with by employing recent deep-learning architectures designed for point clouds. We base our solution on state-of-the-art geometric deep learning architectures, extending them with a novel clustering layer. We show how to train an optimal representation of a noisy 3D point cloud of a human hand purely from synthetic data. For evaluation, we collect a brand-new dataset of human hand videos in RGB-D, named NNHand RGB-D. Extensive evaluation of our approach on our dataset unveils the viable potential of low-precision, hand-shape-based biometric systems.

Acknowledgements

If I would want to mention one thing I have learnt, be it that my future is composed of the opportunities I will not be afraid of taking. These, however, do not come by themselves. Such are given to us by the people we meet. I am grateful to have met so many amazing people that supported me and allowed me to be where I am right now.

Let me start by expressing my gratitude to my parents, **Miroslava Svobodová** and **Jiří Svoboda**, for their endless support in my education and personal development despite the difficult situation our family has been in. Even though my father is not with us anymore, I wish this would make him proud.

Next, I would like to thank my advisor **Michael Bronstein** for giving me the chance to conduct this research in his group, for passing the best of his academic and professional experience, always bringing new insights into my work and believing in what I am doing. I would also like to thank Michael for his incredible financial support, which allowed me to gather knowledge and meet fellow researchers at summer schools and conferences all over the world.

Probably the most influential person who has shaped my personality as a researcher is **Andrew Fitzgibbon**, who maybe does not even know he prevented me from having a professional burnout. It is hard to express in words how thankful I am for meeting Andrew and having him give me the chance to spend amazing 4 months as an intern at Microsoft Research in Cambridge, UK working with him and his colleague **Tom Cashman**.

This dissertation might not have ever been finished without the support of my current employer **NNAISENSE SA**, who gave me the resources and space while writing up this text. Special thanks to co-founders **Jonathan Masci** and **Jan Koutník** for their exceptional availability.

Besides that, I have to thank my co-advisor **Jonathan Masci** for becoming my machine learning mentor and always driving my ideas the right way. Most of the work in this dissertation would not happen without Jonathan's insights. Moreover, I feel privileged to work with someone who understands me so well as a person.

My gratitude goes also to **Davide Eynard**, who has been the infinite source of positive energy in our research group. Besides valuable discussions and teaching me a lot about doing research, Davide's personality has become a true inspiration for me, both in professional and private life.

Throughout my PhD, I had the opportunity to work along with some extraordinary scientists who have always helped to stimulate ideas and gave the workplace a very pleasant spirit - those are **Artiom Kovnatsky**, **Davide Boscaini**, **Federico Monti**, **Emanuele Rodolà** and **Fabrizio Frasca**.

I would also like to thank all other people I have been directly collaborating with on research and I haven't mentioned yet, such as **Christian Osendorfer**, **Asha Anoosheh**, **Timon Willi** and **Pietro Astolfi**.

There was a long road that led to me starting a PhD. I would like to thank **Martin Drahanský** for allowing me to join his research lab already during my undergraduate studies at the Brno University of Technology. Only this experience permitted me later on to join the company **TBS Biometrics AG**, where I could grow professionally while working with many first-class researchers and engineers. It would be hard to list all names here, but let me at least mention the ones I worked closely with - **Radim Dvořák**, **Jan Váňa**, **Tomáš Novotný** and the CTO **Torsten Meister**. I will be forever thankful for their hospitality and for stimulating my professional growth, which allowed me to start my PhD in Lugano in Switzerland later on.

Even though I am often seen as rather a loner, having good friends is essential for my well being. Outside of research, I have been incredibly lucky to have always found endless support from my many amazing friends who have never stopped believing in me. Those were, first of all, my friends from Czech republic that I knew for ages - **Tomáš Burian**, **Lenka Burianová**, **Adam Morkus**, **Michaela Matějková**, **Vendula Kadlecová**, **Vít Kadlec**, **Lucie Zerzánková** and **Veronika Zerzánková**. But also others that I have met along the way and who somehow made my life in Switzerland better - to name a few, these are **Odd Rune Strømmen Lykkebø**, **René Schuler**, **Cornelia Becker**, **Tomáš Horyl**, **Boryana Stratieva**, **Ezekiel Barnett**, **Oscar Gonzalez**, **Lara Ponzio**, **Giacomo Vassalli**, **Lea Bertini**, **Salvatore Ingala**, **Radim Janalík**, **Kail Frank**, and many many others who I hope won't get offended not being mentioned explicitly.

Positive energy is what makes me who I truly am, it reinforces my inner self. To this end, I am forever thankful to all the special people I have met throughout these years sharing my passion for ski touring, rock climbing, kitesurfing and many other activities.

Let me also spend a line or two to thank all the volunteers who contributed to the novel video sequence RGB-D dataset that is presented in this text.

Contents

Figures

Tables

Chapter 1

Introduction

Hands are an indispensable part of human bodies used in our everyday life to express ourselves and manipulate the surrounding world. Moreover, a hand contains highly-unique characteristics that allow for distinguishing among different individuals. The shapes of a hand and its parts have been studied as biometric identifiers in the past, yielding state-of-the-art approaches in the field of Biometrics; however, these are limited in their practicality. This thesis aims at overcoming the limitations and improving the performance of hand-shape-based biometric systems, taking advantage of the latest developments in Deep Learning and 3D Sensing. In particular, such systems can be improved either by means of biometric fusion or by working directly with the hand shape.

The human hand has been extensively studied as a biometric identifier in the last 30 years, yielding state-of-the-art approaches in fingerprint recognition [Maltoni et al., 2009; Jain et al., 2010b; Wang et al., 2014c; Cao and Jain, 2015] and palmprint recognition [Jalali et al., 2015; Minaee and Wang, 2016; Fei et al., 2019a]. Besides this, it has been shown that hand veins [Ding et al., 2005; Wang et al., 2006a; Huang et al., 2014; Mirmohamadsadeghi and Drygajlo, 2011] and hand geometry [Zunkel, 1996; Ross et al., 1999; Sanchez-Reillo et al., 2000; Kumar et al., 2003; Kanhangad et al., 2009, 2011; de Santos-Sierra et al., 2011] form patterns that are, to a certain degree, distinctive between individuals.

The focus of this work is the development of novel deep-learning-based approaches with applications to hand-geometry recognition and its potential fusion with palmprint and fingerprint biometrics.

1.1 Motivation

Unlike two-dimensional hand geometry recognition, which is receiving attention recently [Afifi, 2019], the current state-of-the-art in three-dimensional hand geometry recognition [Kanhangad et al., 2009, 2011] is aging, as the last notable work dates back to 2011. The proposed methods are designed for highly controlled environments and do not perform well in challenging scenarios, such as in the presence of noise, variable illumination or non-rigid deformations of the hand. Several works have tried to propose improvements, which however do not manage to remove the aforementioned constraints Wang et al. [2014a]; Svoboda et al. [2015] or are mainly focused on the geometry of the palmprint area and not the whole hand [Genovese et al., 2014].

Since then the field of computer vision has been revolutionized with the renaissance of deep learning [Masci et al., 2011; Krizhevsky et al., 2012], setting the new state-of-the-art in image processing and analysis. For biometrics, in particular, deep learning has been applied to the task of facial recognition [Yi et al., 2014; Taigman et al., 2014; Hu et al., 2015; Sun et al., 2014], demonstrating machines can do better than humans at recognizing people by their faces. Convolutional neural networks [Fukushima, 1980; LeCun, 1998a] have also been successfully applied in other areas of biometrics such as fingerprinting [Wang et al., 2014c; Cao and Jain, 2015; Lin and Kumar, 2018; Su et al., 2017; Darlow and Rosman, 2017; Tang et al., 2017b; Nguyen et al., 2018] and gait recognition [Wan et al., 2018]. In 2015, it has been brought to attention that the convolutional neural networks, so powerful on image processing tasks, can be easily fooled using so-called adversarial attacks [Goodfellow et al., 2015], which poses a threat, especially for security-oriented applications. Taking into consideration the issue of adversarial examples, hand-geometry recognition could benefit from deep learning as well.

Besides deep learning, 3D-sensing technology has greatly advanced over the last decade, bringing many affordable off-the-shelf 3D sensors (e.g. Microsoft Kinect, Intel RealSense, etc.), suitable for industrial deployment thanks to their favorable form factor, resolution, and price.

Last but not least, the new 3D sensors have brought much more attention to hand pose estimation and to the development of novel algorithms [Tang et al., 2015; Zhou et al., 2016; Chen et al., 2017; Zimmermann and Brox, 2017; Romero et al., 2017; Kulon et al., 2019; Bouritsas et al., 2019; Kulon et al., 2020], which work well on noisy and low precision data, in challenging settings such as partial occlusions or highly non-rigid deformations. These algorithms could ease the semantic segmentation of a 3D hand point cloud and potentially allow for non-

rigid 3D hand geometry recognition.

We believe that the recent advances in computer vision and machine learning allow for a "rebirth" in the field of hand geometry recognition. The current state-of-the-art could be improved by either performing fusion of several different biometric modalities [Ross et al., 2011] (allowing one to compensate for another, which is possibly corrupted in some difficult setting) or robust hand annotation (taking advantage of recent progress in hand pose estimation).

1.2 Contributions

The main goal of this work is to leverage successes of the latest trends in computer vision and machine learning and bring the same paradigm shift to 3D hand-biometric systems. So far, 3D hand-biometric systems have heavily relied on the ability to fuse several different modalities, typically shape and palmprint [Kanhangad et al., 2011]. Adding to this, we introduce the following improvements to the field of hand-based biometric systems:

- First we propose a novel solution to palmprint recognition. Instead of designing new features by hand, we follow the recent trends and employ convolutional neural networks to learn the most suitable features automatically. We define our problem by means of metric learning, aiming at the separation of the genuine and impostor score distributions, and design a novel objective based on the discriminative index [Daugman, 2000a] called *d-prime*. This novel loss function shows better generalization and outperforms the traditional Siamese loss [Bromley et al., 1993; Chopra et al., 2005] used for metric learning in deep neural networks. Moreover, *d-prime* loss presents a general principle that has been shown useful in many applications beyond biometrics [Vijay Kumar et al., 2016; Wang et al., 2018a; Gainza et al., 2019].

- Next, we hypothesize that, by using a high-resolution RGB camera, one can potentially see at least partial fingerprints on the human hand, as well. This scenario shares similarities with latent-fingerprint extraction and matching, where partial fingerprints are typically lifted from a surface and photographed with a high-resolution camera for further processing. It is appealing to approach the problem using convolutional autoencoders; however, it yields the problem of scarce training data. Inspired by the work of Schuch et al. [2016], we decide to solve the task at hand by training an autoencoder model on a synthetically-generated dataset of latent

fingerprints. We evaluate this approach on publicly-available benchmarks showing state-of-the-art performance. The achieved results support the notion that a lack of training data for deep learning models in biometrics can be overcome by means of transfer learning by generating a training set synthetically.

- Both of the above contributions employ convolutional neural networks in order to improve biometric systems. In 2015, Goodfellow et al. [2015] has shown that such models are very vulnerable to noise, which makes them easy to attack adversarially. Over the years, many attack mechanisms have been proposed [Yuan et al., 2019; Chakraborty et al., 2018] making this a real concern. The situation becomes even more critical in biometric applications, where security and reliability are priority number one. We, therefore, devoted some time to work on the stability of neural networks. This resulted in a novel family of architectures called *PeerNets*, wherein alternate classical Euclidean convolutions with graph convolutions to harness information from a graph of peer samples. *PeerNets* have shown to be more stable yielding state-of-the-art performance on different types of attacks. Beyond adversarial attacks, the concept of PeerNets is the first attempt to bring together traditional convolutional neural networks with graph neural networks, resulting in a general principle with far more applications. We have recently demonstrated this by applying PeerNets to arbitrary image style transfer [Svoboda et al., 2020a].

- Ultimately, we would like to not depend on the fusion of several different modalities and perform recognition based solely on the 3D-shape information. This ideal becomes more accessible with the recent development of 3D sensing, which has brought some affordable off-the-shelf RGB-D sensors to the market. Data acquired by such sensors will, indeed, be of lower quality compared to that from high-end 3D scanners. Nevertheless, the dawn of geometric deep learning [Bronstein et al., 2017a] (GDL) has introduced many approaches to more reliable processing of low-precision range scans and point cloud data. To make research in this direction possible, we present a new dataset of RGB-D sequences of hands captured with the Intel RealSense RS-300 camera, together with several baseline methods based on GDL. The new dataset is an attempt to revive the field of 3D hand biometrics. Furthermore, it opens the door to designing and evaluating methods for non-rigid 3D hand recognition, as well as hand-shape recognition from dynamic RGB-D sequences.

1.3 Structure

The remaining text is structured as follows. Chapter 2 gives an overview of the development and the current state-of-the-art in hand biometrics. Chapter 3 summarizes the relevant advances in computer vision. Chapters 4 and 5 present two algorithms designed to improve the fusion-based techniques. In Chapter 4, a novel loss function for training deep palmprint recognition models is described. Chapter 5 presents an approach to improve low-quality fingerprints based on convolutional autoencoders. Chapter 6 proposes a novel family of architectures that improves the stability and security of deep learning models. Diverting towards pure 3D geometry-based hand recognition, Chapter 7 presents our new dataset and first approach to less constrained 3D-hand geometry recognition based on RGB-D input from a low-cost 3D sensor. The work is concluded in Chapter 8, where eventual future steps are discussed.

Chapter 2

Background in hand biometrics

Hand biometrics is the set of biometric traits found on a human hand. In particular, such traits are of fingerprints, palmprint, hand shape, and veins. These are among some of the oldest biometric identifiers throughout history; for example, fingerprints have been used as means of person authentication since ancient times. This chapter first gives an overview of what a biometric system is and how to evaluate its performance. Afterwards, it discusses the most significant hand traits: hand shape, fingerprints and palmprint (see Figure 2.1).

2.1 Biometric systems

A biometric system is an information system that first acquires biometric trait from a subject. It follows by extracting some features from the data and compares the extracted features against a database of feature templates (feature vectors from enrolled users). Finally, based on the comparison score the system makes a decision and executes appropriate action. In general, a biometric system consists of the following modules [Jain et al., 2010a]:

- *Data acquisition module:* A specific device that is used to acquire the biometric data from a user. The performance of the biometric system is highly dependent on the performance of the acquisition module. Poor quality of the acquired data has a direct impact on the decision of the system and often strongly degrades its reliability.

- *Feature extraction module:* An algorithm that is designed to extract a set of salient discriminative features from the pre-processed input data (i.e. biometric trait). It is often preceded by input data quality assessment in order to avoid processing poor input samples and rather reject them at an

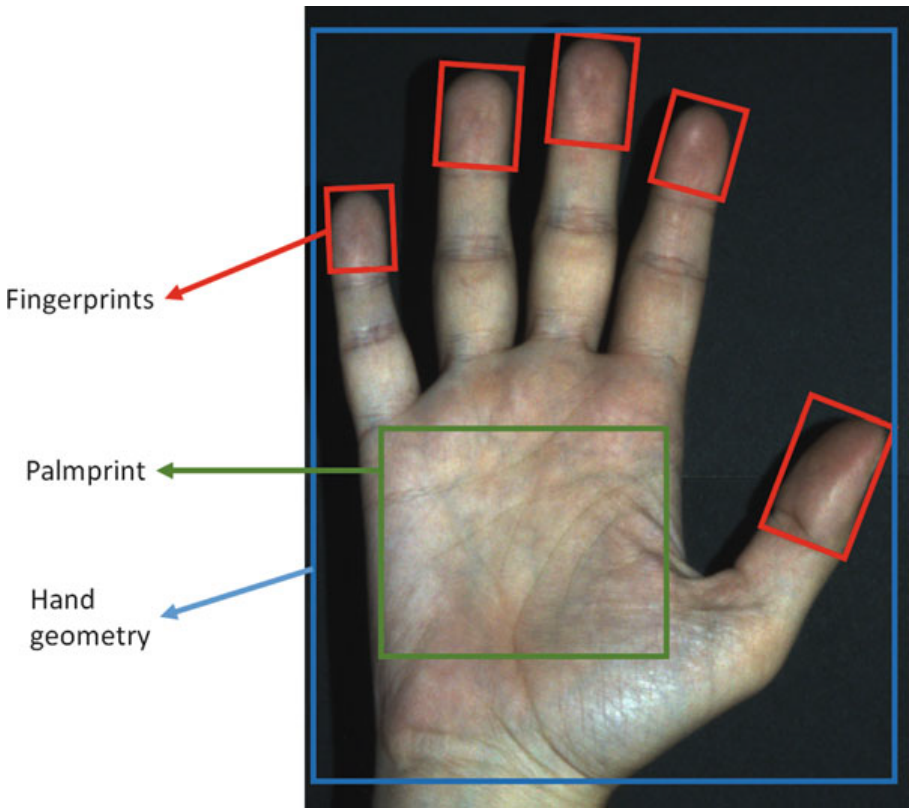


Figure 2.1. The main biometric traits on the palm side of human hand (Genovese et al. [2014]);

early stage. The extracted feature vector is passed to the matching module or, in case of enrollment, stored as a template in the database.

- *Matching module:* Performs a comparison of the extracted feature vectors against the template feature vectors stored in the database in order to generate match scores. It contains a decision-making submodule, which, based on the match scores, either validates a claimed identity (i.e. verification) or provides a ranking of the enrolled identities in order to identify an individual (i.e. identification).

- *Database module:* Stores the biometric information of individuals that were enrolled in the system. During the enrollment process, several samples are acquired for the new identity and processed to extract feature vectors that are stored in the database as so-called *templates*. Depending on the demands on the enrollment process, it might or might not be supervised by a human operator.

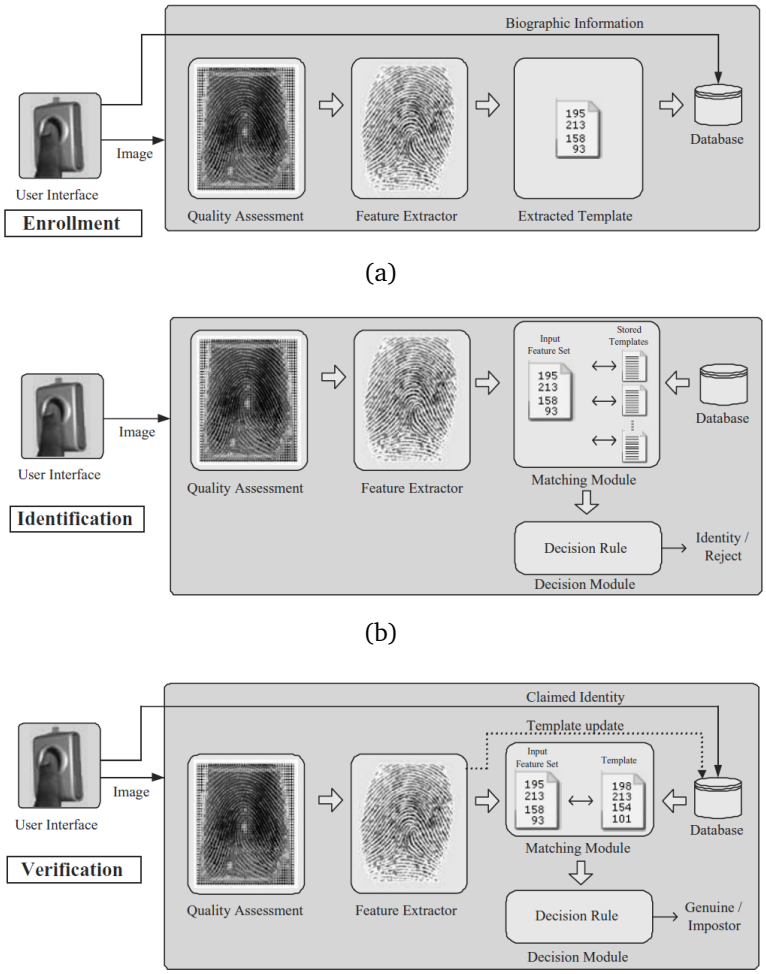


Figure 2.2. Enrollment, identification and verification modes of a biometric system. [Jain et al., 2010a].

2.1.1 Modes of operation

Depending on the desired application, a biometric system typically has two modes of operation: identification and verification [Jain et al., 2010a] (see also Figure 2.2):

- *Identification:* In this mode, the biometric system attempts to recognize an individual by comparing it to the templates of all the users in the database looking for a matching identity. The system is thus performing one-to-many comparison and answers to which identity the presented biometric

trait belongs or fails in case it does not find any match. Such systems are often applied in criminal investigations or monitoring of people.

- *Verification:* An individual who desires to be recognized by the system first claims his identity (via PIN, smart card or user name). The system then acquires the individual's biometric data and compares the extracted feature vectors to the template stored in the database for the claimed identity.

2.2 Evaluation of biometric systems

This section introduces the basic terminology and scores that are used for the evaluation of a biometric system. As opposed to for example card reader based systems, where we search the database for an exact match with the hash stored on the card, in biometrics it is very rare to encounter an exact match. This is due to e.g. sensor noise, nature of human behavior, environmental conditions, etc. As a result, two feature vectors coming from two different acquisitions of the same biometric trait of one user will very rarely be perfectly identical. This yields the following terminology [Jain et al., 2010a]:

- *Intra-class variability:* The variability observed between biometric feature vectors originating from the same individual.

- *Inter-class variability:* Is the variability between feature vectors originating from two different individuals.

- *Biometric entropy:* The measure of how much information does a particular biometric trait contain. The higher the biometric entropy, the more discriminative power a biometric trait has.

In practice, it is desired to extract features which have small *intra-class variability* and large *inter-class variability*, so that they allow to distinguish well between different individuals while staying very similar for the same individual.

Similarity between two biometric feature vectors is expressed by so-called *similarity score* [Jain et al., 2010a]:

- *Genuine score:* Similarity score from matching two feature vectors from the same individual.

- *Impostor score:* Similarity score obtained while matching two feature vectors originating from two different individuals.

The decision of the biometric system is typically defined by *decision threshold T*, which says what is a sufficient matching score in order to yield a positive result of matching. There are the following possible outcomes:

- *True Accept:* Genuine score which is greater than T.
- *True Reject:* Impostor score which is smaller than T.
- *False Accept:* Impostor score which is greater than T.
- *False Reject:* Genuine score which is smaller than T.

False Accept and False Reject are also sometimes called False Match and False Non-Match respectively in the literature.

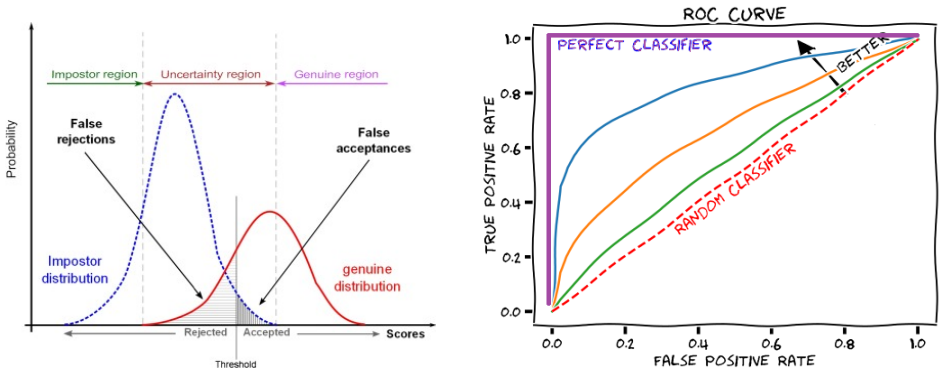
2.2.1 Error visualization and metrics

With the following terminology in hand, we can introduce some measures typically used in biometric systems [Jain et al., 2010a]:

- *False Accept Rate (FAR):* The fraction of impostor scores that exceed the threshold T (see Figure 2.3(a)).
- *False Reject Rate (FRR):* The fraction of genuine scores that fall below the threshold T (see Figure 2.3(a)).
- *Genuine Accept Rate (GAR):* It is defined as GAR $= 1 -$ FRR, which is the fraction of genuine scores that exceed the threshold T.

The decision threshold T allows us to control the FAR and FRR of a biometric system. However, as we decrease FAR, the FRR increases and the other way around. It is not possible to decrease both of them at the same time. The behavior of FAR and FRR for different values of T can be summarized by *Detection Error Tradeoff (DET)* curve [Martin et al., 1997] that plots the dependency of FRR on FAR for different values of T on a normal deviate scale. As an alternative, one can use *Receiver Operating Characteristic (ROC)* curve [Egan, 1975], which instead plots the dependency of GAR on FAR (see Figure 2.3(b)) typically on linear, logarithmic or semi-logarithmic scale.

As opposed to curves, a single value measure of performance of a biometric system is the so-called *Equal Error Rate (EER)*. The EER is defined as the point on the DET curve where FAR $=$ FRR, or equally the point on ROC curve, where GAR $=$ FAR.



(a) (b)

Figure 2.3. (a) An example of the genuine and impostor score distributions [Fakhar et al., 2016]; (b) An example of the ROC curve.

An alternative single value measure is the *discriminative index (d-prime)* [Daugman, 2000a], which measures the separation between the means of genuine and impostor distributions in standard deviation units under the assumption that both distributions can be modelled as normal distributions. It is defined as:

$$d' = \frac{\sqrt{2}|\mu_{gen} - \mu_{imp}|}{\sqrt{\sigma_{gen}^2 + \sigma_{imp}^2}}, \tag{2.1}$$

where μ_{gen}, σ_{gen} and μ_{imp}, σ_{imp} are the means and standard deviations of the genuine and the impostor distributions respectively. The higher the value of d-prime is, the better the performance of the biometric system.

2.3 Hand shape

The first research on 3D hand shape recognition appeared in 1988 when apparatus for 3D hand profile identification [Sidlauskas, 1988] was invented. This initial attempt served as a starting point for the development of this new field, which in the following years attracted many researchers, producing devices such as Biomet Digi-2 (discontinued by now) or the still commercially used Hand-Key II [Allegion, 2017]. Such devices, however, simulate the 3D information by obtaining several 2D images from different viewpoints and processing them separately. The vast majority of works focusing on data from similar devices has been summarized in Duta [2009].

The first approach working with full 3D information dates back to the early 2000's. The rapid development in 3D sensing allowed researchers to either purchase or design their own 3D scanner, making it possible to capture the previously unavailable 3D structure of the human body. Such technologies have been proven valuable in face recognition applications and started making their way into different areas of biometrics since then.

The pioneering work of Woodard and Flynn [2005] focused on using 3D range images of the dorsal side of a hand, extracting finger profiles and using them for biometric identification. This effort was followed a year later, in 2006, by Malassiotis et al. [2006], who presented follow-up work on 3D finger geometry.

Other researchers, in contrast to these initial efforts, have focused mainly on the palm side of the hand, as it supposedly contains much more useful information for recognition. Kanhangad et al. [2009] were the first ones to exploit the full shape of the hand. Using a precise 3D scanner, they collected a dataset of hand 3D range scans that they have made publicly available to other researchers, offering the first standard benchmark for 3D hand biometric recognition. Their efforts were followed by others [Genovese et al., 2014; Wang et al., 2014a; Svoboda et al., 2015], further shaping the field.

The following sections provide a summary of steps that are usually taken in 3D hand shape biometric approaches and present the main ideas employed in the past.

2.3.1 Three-dimensional data acquisition

Over the last decade, leveraging 3D information in computer vision tasks has attracted particular interest, because it provides more information about the shape of an object and therefore makes tasks such as object recognition, scene segmentation or automated quality control easier. This has set a fast pace to new developments in 3D sensing, desired by researchers and industry in order to capture the newly popular 3D information.

Numerous methods have been proposed to address the problem of 3D reconstruction, such as passive stereo vision, photometric stereo, structured-light, time-of-flight and several others (e.g. ultrasound, computed tomography, magnetic resonance imaging, light field cameras, etc.). The particular method has to be carefully chosen based on the use-case, as each imposes some constraints in order for it to provide correct measurements. Moreover, one has to choose a compromise between accuracy and hardware cost. The most relevant methods with respect to the rest of this text are described in the following paragraphs.

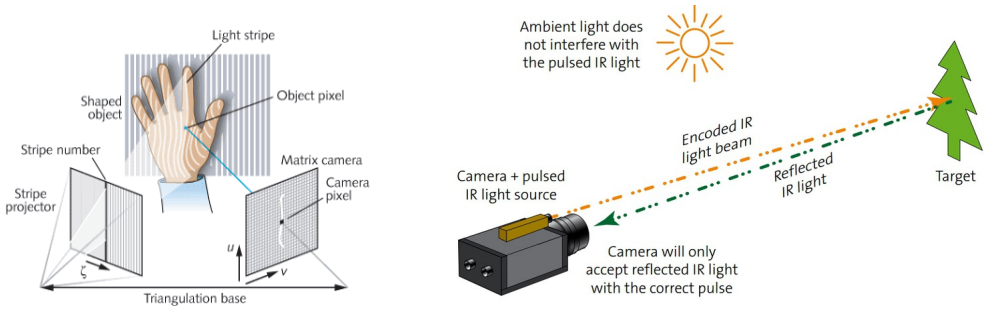
Passive stereo vision Based on principles closely related to human vision [Klette et al., 1998], it is probably the most common technology. Human vision is essentially a binocular process taking two images obtained from different viewpoints and using the parallax between them in order to estimate the depth information. Their main advantage is the simplicity of the hardware setup, which, in the simplest case, consists only of two RGB cameras.

The system consists of two (or more) cameras that capture the scene at the same time. The reconstruction process is based on principles from epipolar geometry [Klette et al., 1998], which is used to produce the so-called disparity map. Such a disparity map serves as input for the 3D coordinate estimation. The detailed description of such a pipeline can be found in Drahanský [2018].

Structured-light methods The structured-light principle can be viewed as a modification of the static binocular stereo [Klette et al., 1998], replacing one of the cameras by a light source, which projects a light pattern into the scene. Depth information is then triangulated by intersecting the camera projection ray and the light ray (or sometimes plane) that is generated by the light source. Various light patterns are available. They can be either statically projected (stripe patterns, dots), or dynamically changing based on temporal coding of several patterns (binary encoding, phase shifting), see Klette et al. [1998]. Latest developments in structured light methods have brought e.g. time-multiplexed structure light approach by Zabatani et al. [2019]. An example of a structured-light method is depicted in Figure 2.4(a).

Time-of-flight approaches A laser range finder is used in order to capture the surface of an object based on the speed of light. It sends multiple pulses of light toward the object and calculates the elapsed time for the pulses to bounce on the surface of the object and come back to the sensor, which is illustrated in Figure 2.4(b). Time-of-flight approaches are gaining popularity in the last years, which is well demonstrated by the recent announcement of micro lidar technologies by Apple Inc. and Intel.

Plenoptic cameras Sometimes also called light-field cameras, capture information about the light field in a scene. In contrast to a classical camera, which records the light intensity, plenoptic camera records the intensity of light in a scene, but also the direction that the light rays are traveling in space. This enables new possibilities, such as refocusing the acquired image to different depths or viewing the same scene from slightly different perspectives.



(a) (b)

Figure 2.4. (a) Multiple parallel stripes structured-light pattern [Klette et al., 1998]; (b) An illustration of the Time-Of-Flight imaging principle [Stemmer-Imaging, 2020].

2.3.2 Acquisition constraints

The 3D hand data acquisition for biometric purposes can be categorized into two main groups:

- *Constrained acquisition:* Imposes constraints on the data acquisition process (e.g. user behavior constraints, environmental constraints).
- *Unconstrained acquisition:* Data is captured without imposing any specific constraints on the user, environment or hardware.
- *Less-constrained acquisition:* The compromise between constrained and unconstrained. Some of the constraints are dropped in favor of convenience.

Most solutions nowadays use constrained acquisition. For a 2D setup, that would mean restricting the hand placement by so-called distance pegs [Jain et al., 1999]. In the case of 3D, the acquisition is kept contact-free and the constraints are typically imposed in terms of environmental light, constant background, etc.

The constrained setup is unfortunately very impractical and researchers, therefore, strove towards introducing unconstrained solutions. As this has been very challenging, a third group, so-called less-constrained acquisition, has been introduced. An interested reader can find more details in Drahanský [2018].

2.3.3 Preprocessing

Input smoothing Range data acquired by low-cost depth cameras, in comparison to high-precision 3D scanners, contain inherently much more noise due to

the lower quality optics hardware and usually less-precise reconstruction methods that are used. Input range scans, therefore, have to be further smoothed in order to get a better model suitable for the recognition. We seek a compromise between model smoothness and loss in detail. A possible solution could be the use of mesh denoising methods [Sun et al., 2007; Lu et al., 2016], which have shown promising results being applied to 3D face recognition by Mráček et al. [2015] in the past.

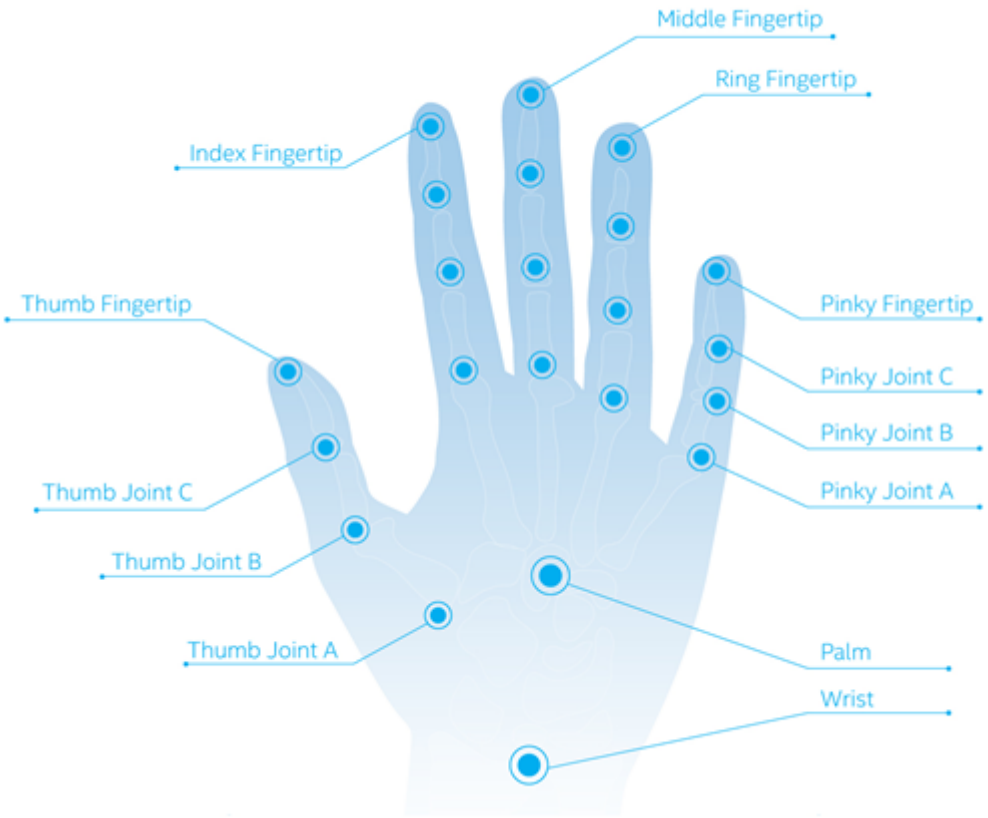


Figure 2.5. The points of interest on a human hand [Intel, 2015];

Segmentation and annotation Detecting the object of interest is a crucial step in any camera-based biometric system. Range scans provide very valuable information for this task and make the segmentation process much simpler compared to any purely RGB-based method. The vast majority of 3D cameras nowadays provide a one-to-one mapping between RGB data and range map pixels, which

makes the segmentation performed in one channel easily transferable into the other. Strangely, there are only a few methods that have employed range information in the past. All, however, assume that certain positioning constraints are met. The work by Svoboda et al. [2015] assumes the hand to be the object closest to the camera. To further improve the separation of the hand from the rest of the arm, one can consider the approach of Wang et al. [2014a]. A more robust solution, requiring fewer constraints, would be to perform statistical modeling of the hand in 3D space as presented in Malassiotis et al. [2006].

Once we have the object of interest segmented, we need to understand it as a human hand. In particular, we are interested in knowing the position of fingertips, finger valleys, the center of the palm and possibly wrist (see Figure 2.5), which are needed by a majority of the existing feature extraction methods. The center of the palm is often detected by means of distance transform, while annotation of fingertips and finger valleys is typically done using curvature analysis of the hand contour. A more detailed explanation is provided in Drahanský [2018].

Pose normalization Our hands are naturally massively non-rigid objects. Most of the real-world scenarios will, therefore, yield samples with out-of-plane rotations or non-rigid deformations of the hands. Many state-of-the-art approaches are not robust to those. There are a few [Kanhangad et al., 2011; Wang et al., 2014a; Svoboda et al., 2015] that try to either reject incorrectly posed hands or compensate for out-of-plane rotations by, e.g., fitting a plane to the previously located center of the palm. The rotation angle is then computed using the plane normal and this can be used to de-rotate the hand. To the best of our knowledge, there are however no methods that could deal with non-rigid deformations of the hand surface.

2.3.4 Feature extraction

Given the data is already annotated and normalized, the next step is to extract features that will be matched against reference templates in order to perform the recognition. As of now, the majority of the existing methods describe the hand surface by means of distances, normal vectors and surface curvatures.

Surface normal estimation Normal vectors are essential for performing curvature analysis of a 3D surface. Methods for their estimation differ based on what representation of the 3D surface we are given. Typically we work on range data, and the neighborhood structure is represented by the range map implicitly. We

can therefore obtain an estimate of the surface normal $\mathbf{n} = (n_x, n_y, n_z)$ at point $\mathbf{p}_{u,v} = (p_x^{u,v}, p_y^{u,v}, p_z^{u,v})$ using finite difference method expressed by

$$\mathbf{n}' = (p_z^{u+1,v} - p_z^{u-1,v}, p_z^{u,v+1} - p_z^{u,v-1}, 2.0), \tag{2.2}$$

$$\mathbf{n} = \frac{\mathbf{n}'}{||\mathbf{n}'||_2}, \tag{2.3}$$

where u, v are the range map grid coordinates. Approaches for computing 3D surface normals for different types of representations are further discussed in Drahanský [2018].

Curvature analysis One of the traditional, yet still widely used, approaches to the 3D hand geometry analysis. The curvature of a surface S at point p is typically computed analytically by fitting a polynomial to a set of neighboring points S_p. Various types of polynomials have been employed in the past, e.g. bicubic Monge surface [Woodard and Flynn, 2005]. Such polynomial can be further used to compute principal curvatures of the surface κ_{min} and κ_{max}, from which we can compute so-called Shape Index (SI) [Dorai and Jain, 1997], which can be used for classification and is expressed as

$$SI = \frac{1}{2} - \frac{1}{\pi}\arctan\left(\frac{\kappa_{max} + \kappa_{min}}{\kappa_{max} - \kappa_{min}}\right), \tag{2.4}$$

where $\kappa_{max} \geq \kappa_{min}$ and $SI \in [0, 1]$. Visual meaning of SI is depicted in Figure 2.6.

An alternative to this approach would be for example fitting 2D polynomials to cross-sectional line segments [Kanhangad et al., 2009] and other methods discussed in Drahanský [2018]. Besides analytic curvature computation, Wang et al. [2014a] proposed an approach performing discrete curvature analysis on a three-dimensional hand contour, which is summarized in more detail by Drahanský [2018].

Geodesic distance computation Utilizing the one-to-one correspondence between range map points and point cloud vertices, a 2D annotation algorithm can be used to find the positions of the fingertips and finger valleys, which are used to compute an axis of each finger. Consequently, a finger geodesic length is computed by walking on the finger surface along the direction of the finger axis using a fast marching algorithm, as proposed by Svoboda et al. [2015].

Computation of the finger axes is done by first separating each finger from the hand blob, employing the algorithm from de Santos-Sierra et al. [2011].

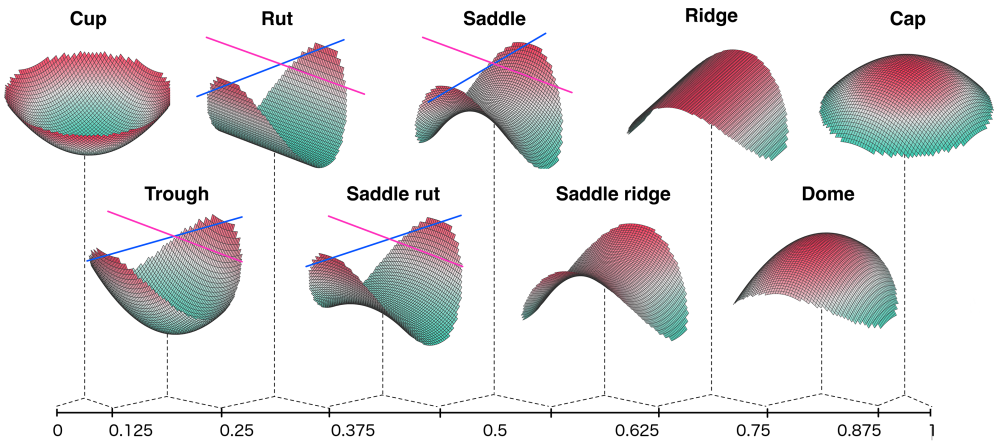


Figure 2.6. Geometrical meaning of different values of Shape Index (SI). Thakku et al. [2015].

Subsequently, for each finger blob, the image moments [de Santos-Sierra et al., 2011] are computed as

$$m_{ij} = \sum_{x,y} I_{x,y} x^j y^i, \tag{2.5}$$

where $I_{x,y}$ is the pixel value at a position (x, y). It is then straightforward to express the center of mass

$$\bar{x} = \frac{m_{10}}{m_{00}}, \bar{y} = \frac{m_{01}}{m_{00}}, \tag{2.6}$$

which is further used to express the central moments

$$\mu_{ij} = \sum_{x,y} I_{x,y} (x - \bar{x})^j (y - \bar{y})^i. \tag{2.7}$$

The second-order central moments form a 2D covariance matrix as

$$cov(I_{x,y}) = \begin{pmatrix} \mu'_{20} & \mu'_{11} \\ \mu'_{11} & \mu'_{02} \end{pmatrix}, \tag{2.8}$$

with coefficients expressed as

$$\mu'_{20} = \frac{\mu_{20}}{\mu_{00}}, \mu'_{02} = \frac{\mu_{02}}{\mu_{00}}, \mu'_{11} = \frac{\mu_{11}}{\mu_{00}}. \tag{2.9}$$

Performing an eigendecomposition of this matrix gives the orientation of the principal axis as the angle between the eigenvector associated with the largest

eigenvalue and the axis which is closest to this eigenvector

$$\Theta = \frac{1}{2} arctan\left(\frac{2\mu'_{11}}{\mu'_{20} - \mu'_{02}}\right), \tag{2.10}$$

where the special case of $\mu'_{20} - \mu'_{02} = 0$ is assumed to be handled explicitly.

2.3.5 Matching

Depending on the features extracted from the hand 3D surface, different matching algorithms have been developed in the past, giving the *matching score*, which is used to determine the identification outcome. Let us list the main ones in the following paragraphs.

Shape Index matching As the name suggests, this matching method proposed by Woodard and Flynn [2005] is used together with the SI features that are described earlier in this section. The matching score is computed as the normalized correlation coefficient of two SI features, giving a value in range $[-1, 1]$, where higher means better. It is given by the following formula:

$$eCC(SI_Q, SI_T) = \frac{\sum_{(i,j)\in valid}(SI_Q(i,j) - \overline{SI}_Q)(SI_T(i,j) - \overline{SI}_T))}{\sqrt{\sum_{(i,j)\in valid}(SI_Q(i,j) - \overline{SI}_Q)^2 \cdot \sum_{(i,j)\in valid}(SI_T(i,j) - \overline{SI}_T))^2}}, \tag{2.11}$$

where $SI_Q(i,j)$, $SI_T(i,j)$ are shape index values, which are implicitly assumed to be valid. $\overline{SI}_Q$, $\overline{SI}_T$ are the sample mean shape index values of the query and the template images respectively. The SI matching score is typically computed for each finger separately followed by a score-level fusion.

Matching curvatures and normals Efficient methods to match curvature and normal features were designed by Kanhangad et al. [2009]. Such features are usually extracted from fingers of the hand and the final score is afterwards obtained by score-level fusion, which should compensate for small off-the-plane rotations of the hand. Given features extracted from N_s cross-sectional segments on i-th finger for both template T_i and query Q_i samples, they propose to match curvature features using the cosine similarity metric defined as

$$s_i^c = \frac{1}{N_s} \sum_{j=1}^{N_s} \frac{\Phi(T_i^j, l_T^j, l_Q^j)\Phi(Q_i^j, l_Q^j, l_T^j))}{|\Phi(T_i^j, l_T^j, l_Q^j)||\Phi(Q_i^j, l_Q^j, l_T^j))|}, \tag{2.12}$$

where Φ is a function ensuring both T_i and Q_i are the same lengths. For its definition, we refer a reader to Drahanský [2018]. The cosine similarity distance cannot be used to match the normal vectors directly and the authors, therefore, propose another metric to match the normals, based on the angle between two vectors, expressed by

$$s_i^n = \frac{1}{N_s}\sum_{j=1}^{N_s} cos^{-1}(\Phi(T_i^j, l_T^j, l_Q^j)\Phi(Q_i^j, l_Q^j, l_T^j)). \tag{2.13}$$

The resulting matching score is then computed as a linear combination of the curvature score s_i^c and normal score s_i^n

$$S = w\frac{1}{4}\sum_{i=1}^{4} s_i^c + (1-w)\frac{1}{4}\sum_{i=1}^{4} s_i^n, \tag{2.14}$$

where w is an empirical weight expressing the importance of the normal features compared to the curvature features.

Metric learning approach Wang et al. [2014a] and Svoboda et al. [2015] propose a learning-based approach for matching k-dimensional vectors of distances computed over the hand surface. Under the assumption that some of the computed distances are more important for telling apart different individuals, it is desired to introduce some weighting, which will give more importance to the more prominent features. Such weighting can be introduced by using the Mahalanobis distance

$$d_M(\mathbf{x},\mathbf{y}) = \sqrt{(\mathbf{x}-\mathbf{y})^T\mathbf{M}(\mathbf{x}-\mathbf{y})}, \tag{2.15}$$

where $\mathbf{x}$ and $\mathbf{y}$ are two feature vectors and $\mathbf{M}$ defines a metric space that is better suitable for comparing $\mathbf{x}$ and $\mathbf{y}$. The matrix $\mathbf{M}$ can be obtained, e.g., by metric learning, as described in Drahanský [2018].

2.4 Fingerprints

Fingerprints are, both historically and nowadays, the most widespread means of biometric person authentication. As stated in Moenssens [1971]; Lee and Gaensslen [2001], human fingerprints date back to the ancient times as they have been discovered on many archeological items. However, the first more scientific work examining fingerprints dates back to the 19th century [Galton, 1892]. By the beginning of the 20th century, fingerprints were already well understood and

fingerprint recognition was formally accepted as a personal identification method and became a standard procedure in forensics. Since then, various methods for processing different types of fingerprints were introduced. Fingerprints remain one of the most popular biometric traits until nowadays since they are very stable and unique. Compared to other biometric traits, they suffer less from aging and develop at a very early age.

2.4.1 Data acquisition

With regard to how the fingerprints are collected, they can be classified as off-line or online (live-scan). An off-line fingerprint is obtained by applying ink on the fingertip and then placing or rolling the fingertip on a piece of paper. The inked impression is then typically digitalized by means of an optical scanner. A particular type of off-line fingerprints are the *latent* fingerprints. The greasy nature of human skin can leave a fingerprint impression on surfaces we touch. Using special chemicals, latent fingerprints can be lifted from the surface and stored. Such greasy impressions can be usually found at crime scenes. On the other hand, an online fingerprint is captured directly by placing the fingertip on a sensor that digitalizes the impression upon contact. Some examples of fingerprints acquired using different acquisition devices are shown in Figure 2.7. Based on the acquisition method, fingerprints can be further categorized as follows

- *inked* - rolled (nail-to-nail), flat (single finger) or slap (four-finger flat);
- *sensor-scan* - different technologies yielding various acquisition qualities, e.g. optical, capacitive, etc.;
- *latent* -impressions of the papillary lines that are unintentionally left by a subject at e.g. crime scenes.

2.4.2 Feature extraction

The selection of the fingerprint representation has far-reaching implications on the matching modules [Maltoni et al., 2009]. One needs to be particularly careful to select features which are suitable for discriminating between identities and remain invariant for a given individual over time.

Fingerprint pattern can be analyzed at different scales at which it exhibits different types of features:

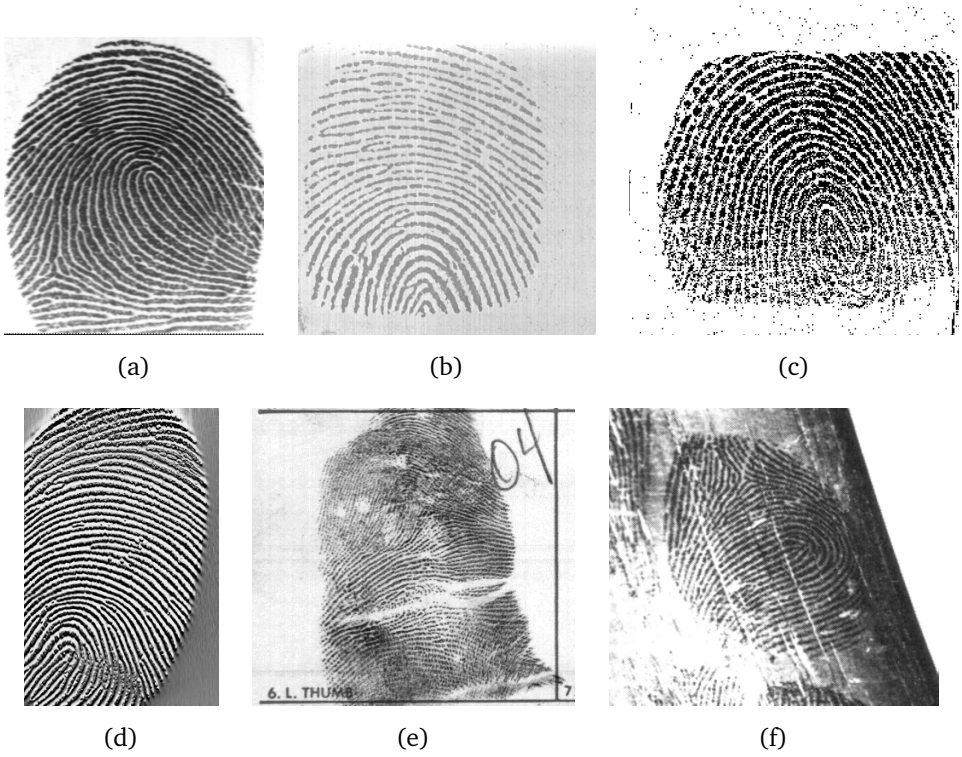


Figure 2.7. Fingerprint images from different sensors: (a) FTIR-based optical scanner; (b) Capacitive scanner; (c) Piezoeletric scanner; (d) Thermal scanner; (e) Inked impression; (f) Latent fingerprint. [Maltoni et al., 2009]

- *Global-level* at which the ridgeline flow forms specific patterns (see Figure 2.8). These are typically called *singular points*(loop or delta) and define control points surrounded by the ridgelines. Besides that, at a global level, additional features as shape, orientation image and frequency image can be computed.

- *Local-level* for which Moenssens [1971] identified a total of about 150 different ridge characteristics called *minutiae* details. Their visibility heavily depends on the quality of fingerprints impression. Two most prominent minutiae are *ridge endings* and *ridge bifurcations* (see Figure 2.9(a)).

- *Very-fine-level* allows detecting intra-ridge details, such as width, shape, curvature, edge contours of ridges, etc. Supposedly the most important of which are finger sweat pores (see Figure 2.9(b)), whose geometry is considered highly distinctive. Extracting the fine details however requires

very high-resolution sensing and is often neither practical nor feasible.

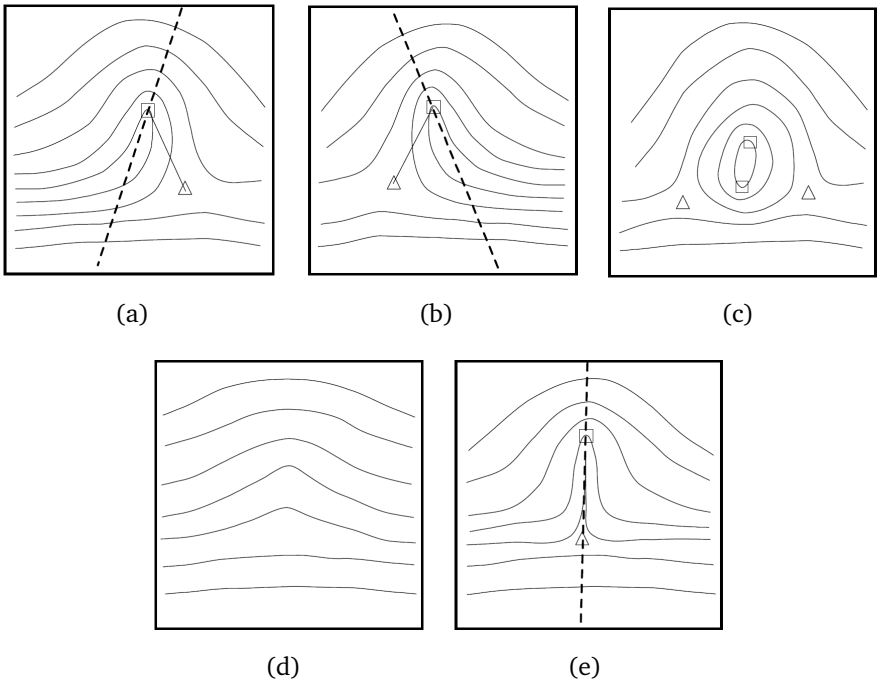


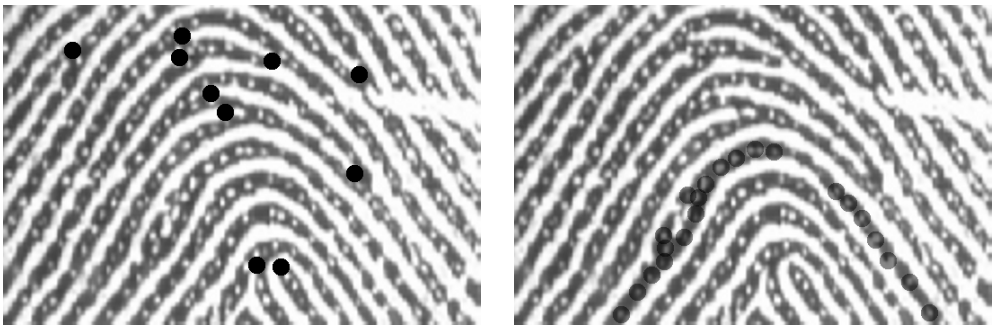
Figure 2.8. Coarse level fingerprint patterns: (a) Left loop; (b) Right loop; (c) Whorl; (d) Arch; (e) Tended arch; Squares and triangles denote loop-type and delta-type singular points respectively. [Maltoni et al., 2009]

2.4.3 Fingerprint matching

Fingerprint matching is an extremely difficult problem due to the large variations in different impressions of the same finger cause by displacement, rotation, partial overlap, non-linear distortion, variable pressure, etc. Human fingerprint examiners consider several factors during the manual fingerprint matching process, which are listed in Maltoni et al. [2009].

Automated fingerprint matching is usually inspired by the approach of human experts and according to Maltoni et al. [2009] it can be categorized into three classes as follows:

- *Correlation-based matching* takes two superimposed fingerprint images and computes the correlation between corresponding pixels for different alignments.



(a) (b)

Figure 2.9. Fine-level fingerprint patterns: (a) Black circles denote fingerprint minutiae; (b) Transparent circles denote sweat pores on a single ridgeline. (Maltoni et al. [2009])

- *Minutiae-based matching* starts by extracting minutiae from two fingerprints and storing them as sets of points in two-dimensional space. Matching is then equivalent to finding the alignment between two sets of minutiae that results in the maximum number of minutiae pairings.

- *Feature-based matching* is applied for low-quality fingerprints, where extracting minutiae would be too difficult or impossible. Instead, it is more reliable to extract less informative texture information, local orientation and frequency, etc.

Finding the optimal matching mechanism always yields a trade-off between whether one should do more engineering in order to obtain better quality data or whether to spend more time designing a very robust matching algorithm that will be less sensitive to different qualities of the input samples.

The first two categories have been thoroughly researched in the past yielding state-of-the-art traditional methods like Lee et al. [2001]; Komarinski [2004]; Maltoni et al. [2009]; Cappelli et al. [2010]; Jain et al. [2000, 2001]; Feng [2008]; Jain et al. [2010b]; Tico and Kuosmanen [2003]; Jiang and Yau [2000] as well as methods based on deep learning [Wang et al., 2014c; Cao and Jain, 2015; Lin and Kumar, 2018; Su et al., 2017; Darlow and Rosman, 2017; Tang et al., 2017b; Nguyen et al., 2018]. On the contrary, latent fingerprints are typically partial, blurred and noisy which yields poor ridge quality. They can be lifted from the surface by means of sophisticated chemical procedures or photographed using a high-resolution camera for further processing. The latter type of acquisi-

tion makes this scenario very similar to extracting fingerprints from RGB stream of a depth camera that provides photographs of fingerprints, if available, of very poor quality. There have been several works [Jain and Feng, 2011; Yoon et al., 2010, 2011; Sankaran et al., 2011; Feng et al., 2013; Cao et al., 2014; Tang et al., 2017] trying to improve latent fingerprint matching. Many of them are however only semi-automatic and therefore not suitable for the use-case targeted in this text.

2.5 Palmprint

Palmprint is probably the most prominent texture-based characteristic of the human hand viewed from the palm side. Compared to fingerprints, palmprints appear to be more user friendly, as their capture can be performed easily in a contactless manner with a regular CCD camera sensor. Palmprints are similar to fingerprints in their structure, which allowed for the transfer of knowledge from fingerprint technologies and has boosted the development in the last 20 years.

Similarly to fingerprints, the palmprint consists of the following characteristics (see Figure 2.10):

- *flexion creases* - the three principal lines on the palm;
- *secondary creases* - several additional well-visible wrinkles;
- *series of ridges* - similar to ridges that fingerprints are composed of, creating features such as singular points and minutiae.

2.5.1 Data acquisition

The type of features that can be captured from a palmprint strongly depends on the acquisition device. There are very precise sensors that capture also the fine ridges, thus allowing for high-precision palmprint recognition. Such sensors are usually less user-friendly. Nevertheless, decent palmprint information allowing for high-accuracy palmprint recognition, while maintaining the advantage of user-friendliness, can be captured even by an ordinary CCD camera. According to Genovese et al. [2014] the acquisition methods can be grouped into

- *Touch-based two-dimensional methods* are further divided into offline and online. Offline methods include scanning of inked palmprints and lifting of latent impressions. Online methods, instead, use an optical device or

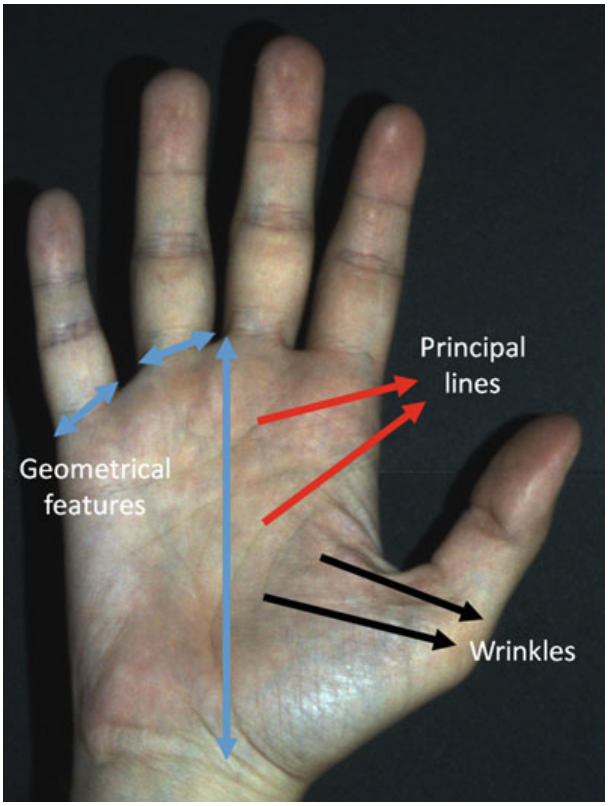


Figure 2.10. Characteristics of palmprint [Genovese et al., 2014];

CCD camera, which captures palmprint of the user's hand placed on a fixed support.

- *Touch-based three-dimensional methods* again require the user to place his hand on fixed support in order to create a 3D reconstruction of the hand palmprint using structured light illumination techniques.

- *Touchless two-dimensional methods* are mostly based on CCD cameras, smartphones, etc. Such methods are typically used to design a low-cost and more user-friendly palmprint recognition systems.

- *Touchless three-dimensional methods* make use of costly three-dimensional laser scanners to enable accurate and less constrained palmprint recognition.

2.5.2 Palmprint features

Depending on the level of detail of the input samples, Genovese et al. [2014] categorized features that can be extracted from palmprint images as follows:

- *Principal line features* represent the flexion creases on a human hand palm (see Figure 2.10 and 2.11). Such features are very stable and thanks to their prominence, they are easy to capture even with low-resolution sensors. Unfortunately, they can be easily faked and are not very distinctive.

- *Wrinkle features* are the secondary creases (see Figure 2.10), which are very distinctive due to their high irregularity. They are still well-collectable, even though they require a higher resolution sensor compared to the principal line features. Moreover, they are less permanent.

- *Delta point features* are similar to the delta points extracted from fingerprints. Their collectability, unfortunately, depends on a high-resolution sensing device.

- *Minutiae features* represent the same minutiae features that are typically extracted from fingerprints (e.g. bifurcations, etc.). They are very distinctive and permanent but require acquisition using a high-resolution sensor.

- *Level 3 features* such as pores, incipient ridges or scars analogically to the features extracted from fingerprints. These features are very distinctive but require highly precise acquisition devices.

In practice, input resolutions ranging from 150dpi up to 400dpi or more [Kong et al., 2009] are used. High-resolution images, allowing for the extraction of minutiae, are typically used only for forensic applications [Duta et al., 2002; Shu and Zhang, 1998]. On the other hand, the vast majority of the current research and state-of-the-art methods focus on lower-resolution images that can be employed in civil and commercial applications.

2.5.3 Feature extraction and matching

Different approaches for palmprint recognition have been proposed based on the acquisition method and algorithms used to extract and match palmprint features. In a broader sense, we can divide palmprint recognition techniques into five different categories [Genovese et al., 2014], which are shortly described next:

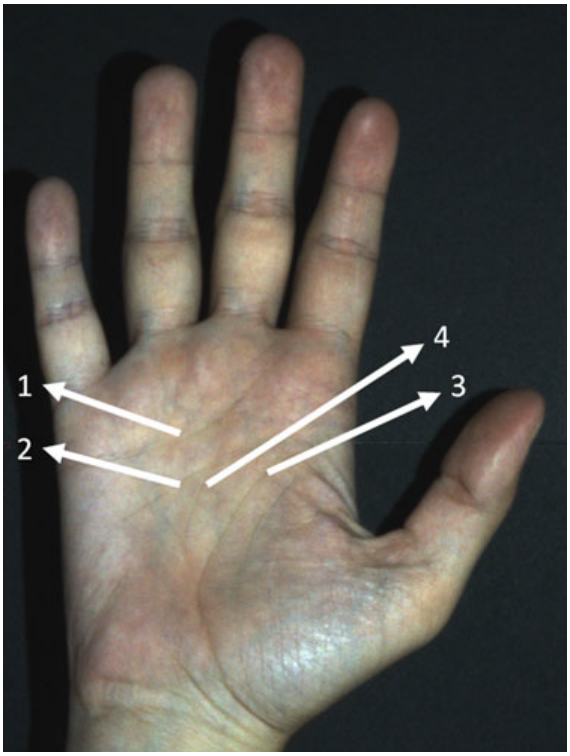


Figure 2.11. Palmprint principal line features: (1) the heart line; (2) the head line; (3) the life line; (4) the fate line. [Genovese et al., 2014];

- *Ridge-based* methods are analyzing the ridge pattern, the position of the delta points and the location of the minutiae points in order to perform recognition [Liu et al., 2013; Laadjel et al., 2010]. They are usually inspired by technologies for processing fingerprints.
- *Line-based* approaches focus on processing of the flexion and secondary creases [Nibouche and Jiang, 2013; Wu et al., 2004]. The processing is typically based on computer vision methods for edge detection and description.
- *Subspace-based* methods leverage feature extraction based on PCA, LDA or ICA [Huang et al., 2008; Ribaric and Marcetic, 2012; Wang and Ruan, 2006b]. The input images can be additionally preprocessed by means of e.g. Fourier transform, Gabor filtering, etc.
- *Statistical approaches* are either based on local or global statistics of the image. Local statistics (e.g. mean, variance) are computed on subregions

of the input image. Global statistics, instead, are computed on the entire image. In particular, e.g. Zernike moments [Yang and Wang, 2010], Binarized Statistical Image Features (BSIF) [Raghavendra and Busch, 2014], etc. have been applied in the past.

- *Texture-based* or sometimes also *coding-based* approaches are first preprocessing the input by means of different filters and then create an encoding by quantization of the magnitude or phase of the response. The Hamming distance is used to compute the similarity between the templates. Typical representatives of such methods are PalmCode [Kumar and Shen, 2004], Competitive Code [Kong and Zhang, 2004], Ordinal Code [Sun et al., 2005], etc. Most coding based schemes are using the Hamming distance with several different offsets for matching.

- *Hybrid approaches* Some researchers have worked on a combination of several of the above categories in order to improve performance. This results in so-called hybrid approaches [Lu et al., 2009; Wang et al., 2012, 2013].

In recent years, the field of deep learning has become popular, yielding a vast amount of applications, including biometrics. In palmprint recognition, the most notable works are Ariyanto et al. [2018]; Jalali et al. [2015]; Minaee and Wang [2016]; Zhong et al. [2019]; Izadpanahkakhk et al. [2018], however only a few have evaluated their method on contactless palmprint recognition. A broader summary of palmprint recognition methods can be found in Fei et al. [2019a]; Ungureanu et al. [2020].

2.6 Biometric fusion

In less-constrained settings, such as contactless hand recognition, it is often difficult to obtain high-quality data. The system should, therefore, be able to deal with sensor noise, which may be imposed by a sudden environmental change (e.g. sunlight, rain, etc.), incorrect use of the device as well as a spoofing attempt. As suggested by Ross et al. [2006], one of the straightforward ways of dealing with sensor noise is to introduce a fusion of multiple biometric modalities, which is further discussed in Ross et al. [2011]; Fierrez et al. [2018]. In such case, may there be a noisy or invalid input from one sensor, the other sensors could potentially still provide useful information and system could function transparently. Besides, using a multimodal biometric system can significantly improve population coverage and provide additional protection against spoofing attacks.

There are however several considerations to be taken into account in the design of a multimodal biometric system. One has to make a trade-off between additional cost and performance improvements [Ross et al., 2011]. The acquisition process has to be defined carefully as multiple biometric traits are being captured. Last but not least, an appropriate fusion scheme has to be chosen.

As we have already shown in the previous section, human hands not only allow us to measure their geometry, but also possess some of the strongest biometric modalities, such as fingerprints and palmprints. They are therefore naturally good candidate for performing biometric fusion.

2.6.1 Levels of fusion

The critical thing to be decided while designing a multimodal biometric system is at which stage the information should be combined by the fusion algorithm. Biometric information from different modalities can be fused at different levels, either prior or after matching. Clearly, the earlier we decide to merge several different modalities, the more information we have available for the fusion process. For example, a raw input data can have several megabytes of information, while the fusion of the decision scores has to work with only a few bytes. A brief description of the most typical fusion types is described next.

Sensor-level fusion operates on raw data captured by multiple sources before they are fed to the feature extraction module, also-called image-level fusion. In biometrics, a typical example would be merging multiple images made by a sweep-sensors [Xia and O'Gorman, 2003] in order to create a composite fingerprint image in a process often regarded to as mosaicing. Stitching several images together can be also applied in for example face recognition, observing face from multiple different viewpoints and merging them into one resulting face image. A similar approach can be also taken advantage of in order to reconstruct an approximate 3D model of one's face.

Feature-level fusion aims at consolidating information from two biometric feature sets of the same identity. The two feature sets might not necessarily have to originate from the same biometric modality and feature extraction algorithm. In the case of the same algorithms and modalities, the situation is quite simple. One can either update or improve the existing template. For two different modalities, however, the situation is rather problematic. In order to be able to combine features from two different modalities, one has to perform feature normalization

(many schemes have been proposed e.g. min-max, median, etc.) eventually followed by feature selection and transformation (e.g. dimensionality reduction). Besides, a novel approach based on feature-level rank fusion [Guo. et al., 2019] has been proposed recently. An example of such process designed by Ross and Govindarajan [2005] is depicted in Figure 2.12.

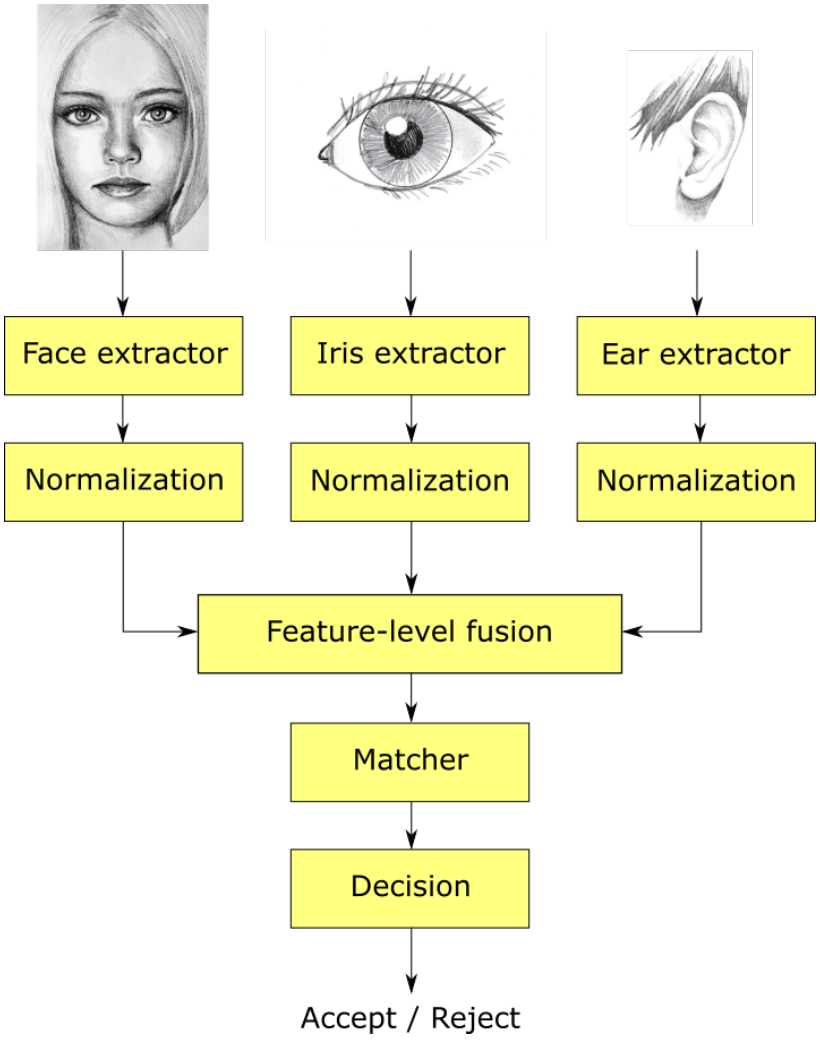
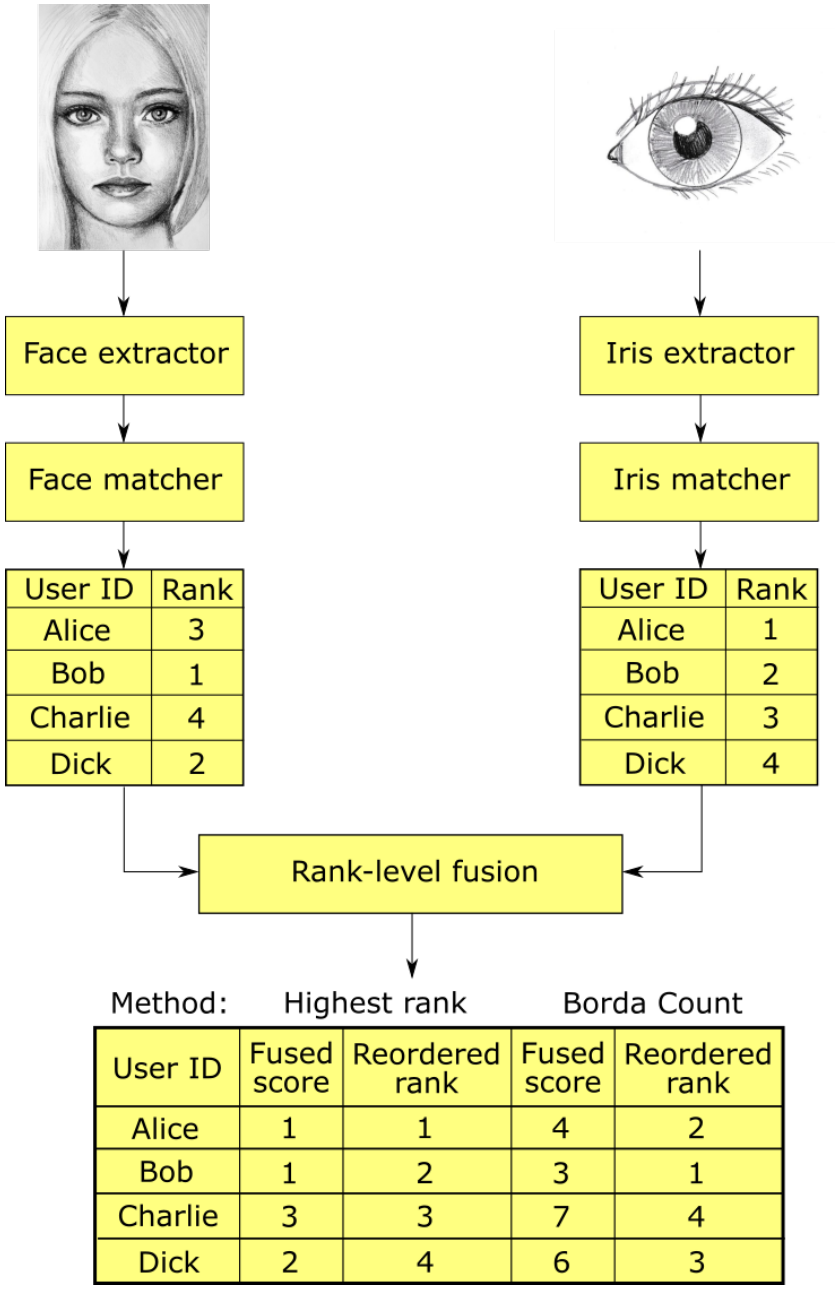


Figure 2.12. Feature-level fusion procedure. [Ross et al., 2011]

Rank-level fusion is typically used during identification, where the system output can be viewed as a ranking of the identities in the database. The fusion algorithm then merges the rankings from different biometric modalities in or-

der to make the final decision. There are different methods for combining the ranks, such as highest rank method, Borda count method, logistic regression method, etc. [Ross et al., 2011]. An example of rank-level fusion is depicted in Figure 2.13.

Decision-level fusion is employed mainly in systems where only the final decision outputs of the individual matching modules are available. Numerous schemes for decision-level fusion have been proposed in the past including AND/OR rules [Daugman, 2000b], majority voting [Lam and Suen, 1997], weighted majority voting [Kuncheva, 2004], Bayesian decision fusion [Xu et al., 1992], the Dempster-Shafer theory of evidence [Xu et al., 1992] and behavior knowledge space [Huang and Suen, 1995]. More details about the specific schemes can be found in Ross et al. [2011]. See Figure 2.14 for an example of a decision-level biometric fusion.



User ID	Rank
Alice	3
Bob	1
Charlie	4
Dick	2

User ID	Rank
Alice	1
Bob	2
Charlie	3
Dick	4

User ID	Fused score	Reordered rank	Fused score	Reordered rank
Alice	1	1	4	2
Bob	1	2	3	1
Charlie	3	3	7	4
Dick	2	4	6	3

Figure 2.13. An illustration of rank-level fusion for two different fusion schemes. [Ross et al., 2011]

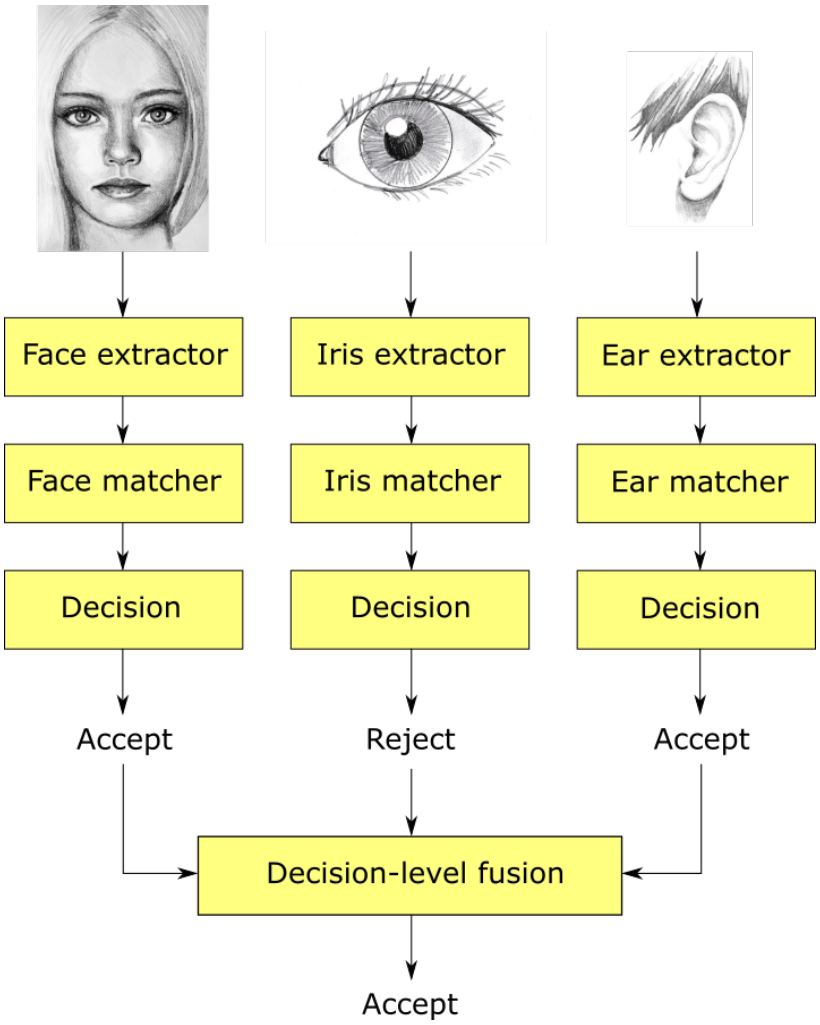


Figure 2.14. Combining decisions of multiple matching modules based on the majority voting scheme. *ID* and *Ver* denote the identification and verification modes of operation respectively. [Ross et al., 2011];

Chapter 3

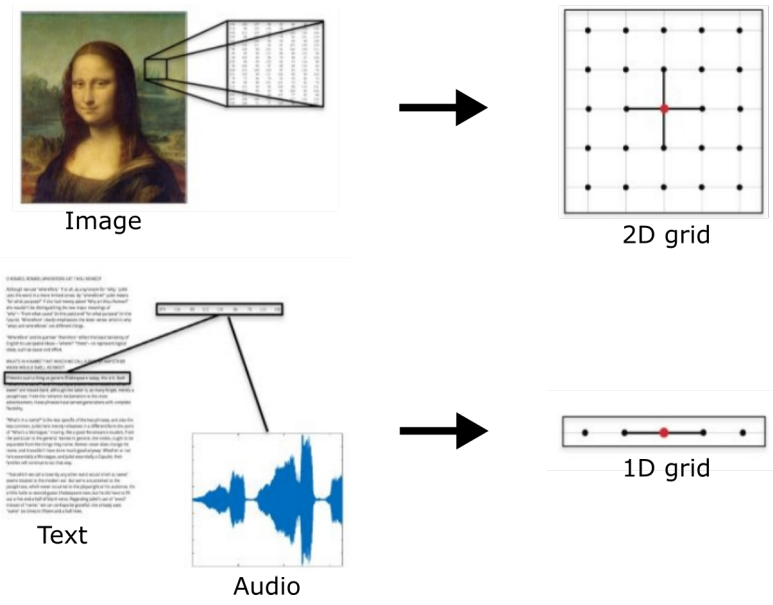
Related advances in deep learning

Biometric traits are often captured by cameras, leading to overlap in the fields of biometrics and computer vision. Both fields share the same principles for extraction of interesting features from the input. This chapter discusses the advancements in deep learning for computer vision which are closely related to applications in this work.

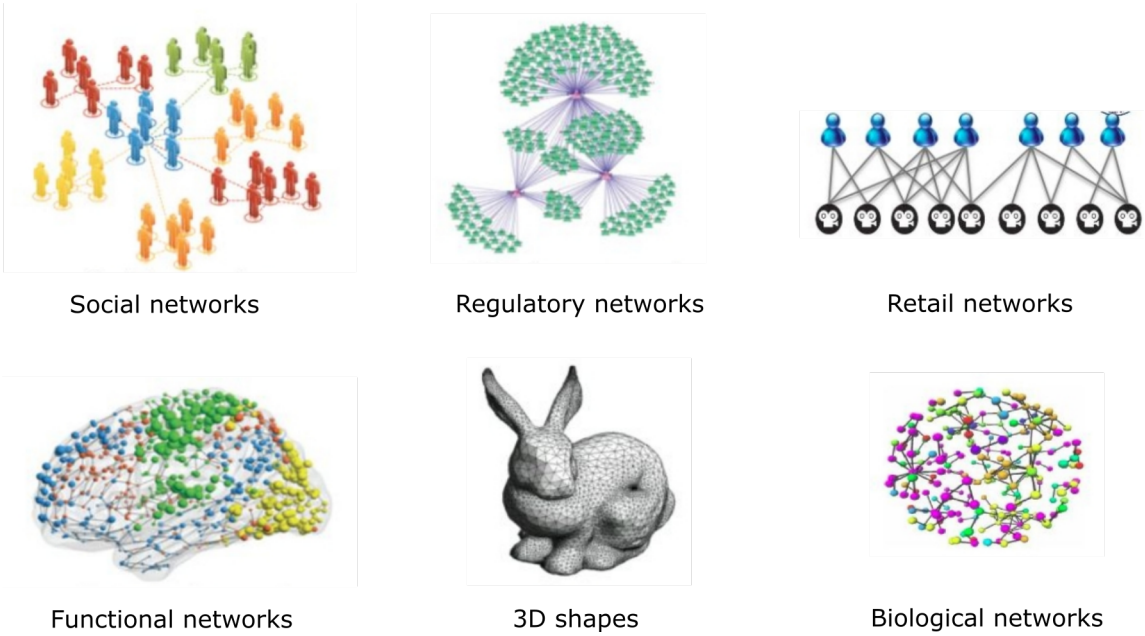
Deep learning is a powerful machine learning framework for both supervised and unsupervised learning tasks. Increasing depth of networks allows representing functions of higher complexity. In recent years, deep learning has revolutionized many fields (see Figure 3.1(a)), such as image processing [Ciresan et al., 2011b,a; Krizhevsky et al., 2012; Zhu et al., 2017], natural language processing [Young et al., 2017], and many others, achieving performances nobody could imagine a decade ago. Only very recently, deep learning has made its way into fields working on *non-Euclidean data* (see Figure 3.1(b)) such as point clouds or social graphs. This was with the introduction of Geometric Deep Learning (GDL) [Bronstein et al., 2017a], which focuses on designing deep-learning approaches for data without conveniently-organized Euclidean structure.

Surprisingly, despite their success, these methods are only slowly making their way into some fields of biometrics. They have been heavily exploited for fingerprinting applications [Wang et al., 2014c; Cao and Jain, 2015; Schuch et al., 2016, 2017], face recognition [Yi et al., 2014; Taigman et al., 2014; Hu et al., 2015; Sun et al., 2014] and palmprint [Jalali et al., 2015; Minaee and Wang, 2016; Ariyanto et al., 2018; Genovese et al., 2019; Fei et al., 2019a; Ungureanu et al., 2020], showing the looming potential. However, the hand geometry and has been left quite untouched by the latest trends. The aforementioned works suggest that re-developing three-dimensional hand geometry biometrics based on the current computer vision trends could lead to previously unprecedented

results.



(a) Euclidean domains



(b) Non-Euclidean domains

Figure 3.1. Deep learning on different domains - examples of the data structures that can come as an input. [Bronstein et al., 2017b]

3.1 Multilayer perceptron (MLP)

MLPs [Rosenblatt, 1958], in machine learning slang sometimes also-called "universal approximators" [Cybenko, 1989; Hornik et al., 1989], are the essential elements of deep learning models [Ivakhnenko and Lapa, 1965; Ivakhnenko, 1971] nowadays. In simple words, continuous function f can be approximated with a feedforward network, which defines a mapping $y = f(x, \theta)$. The network is representing a classifier $y = f(x)$, which maps an input sample x to a class y, and θ are the learnable parameters of the network, which are optimized to provide the best approximation of the function f. Feedforward neural networks can be further extended with feedback connections, which results in so-called recurrent neural networks [McCulloch and Pitts, 1943; Kleene, 1956].

Such models are called networks because they are typically composed of many different functions, which are connected together in a directed acyclic graph describing the relations between different functions. Such functions can be connected in a chain, where the first function would be called the first layer, second the second layer, and so on. The number of layers is called the depth of the neural network. Deeper networks were found to provide better performance, which yields the name of the field deep learning. Each hidden layer has its dimensionality, called width, defined. The last layer of the model is called the output layer and all layers between the first and last are referred to as hidden layers because their outputs are not directly visible. For an example of such feed-forward network, the reader is referred to the book of Goodfellow et al. [2016].

3.2 Building blocks

In the following paragraph, we will explain the relevant building blocks of deep learning models such as *Fully connected layer*, *Convolutional layer*, *Pooling layer* and *Activation function*. Please note that this is not an exhaustive overview. For more details, we refer the reader to Schmidhuber [2015]; Goodfellow et al. [2016].

Fully connected layer is a basic building block of multilayer perceptrons mentioned above. It computes a nonlinear transformation of its input $\mathbf{x}_i$ and outputs new representation $\mathbf{y}_i$. It is represented by an affine transformation $\mathbf{t}_i$ which is parametrized by weights $\mathbf{W}_i$ and bias $\mathbf{b}_i$, typically followed by a non-linear

activation function σ. The above can be summarized as:

$$\mathbf{x}_i = \mathbf{y}_{i-1}, \tag{3.1}$$

$$\mathbf{t}_i = \mathbf{W}_i \mathbf{x}_i + \mathbf{b}_i, \tag{3.2}$$

$$\mathbf{y}_i = \sigma(\mathbf{t}_i), \tag{3.3}$$

where $i \in \{1, \ldots, N\}$ is the index of a layer for a multilayer perceptron with N layers. Output of the i-th layer is the input to the $i+1$-st layer. The input to the first layer is then the input sample fed to the multilayer perceptron, while output of the last layer is the final output of the model.

Convolutional layers were pioneered by Fukushima [1980]; Waibel [1987]; Weng et al. [1993]; LeCun et al. [1998b] for data with regular spatial structure, such as audio or images. Compared to fully connected layers, they can also capture the spatial relationships in the data. Combination of such layers gives raise to so-called Convolutional Neural Networks (CNNs). A convolutional layer operating on an N-dimensional input $f(\mathbf{x}) = \{f_1(\mathbf{x}), \ldots, f_n(\mathbf{x})\}$ with a set of filters $H = \{h_{ij} | i = 1 \ldots m, j = 1 \ldots n\}$ and a nonlinearity σ can be defined as:

$$g_n(\mathbf{x}) = \sigma\left(\sum_{j=1}^{m} (f_j \star h_{ij})(\mathbf{x})\right), \tag{3.4}$$

where g is a m-dimensional output $g(\mathbf{x}) = \{g_1(\mathbf{x}), \ldots g_m(\mathbf{x})\}$, so-called *feature maps*. The operator $\star$ denotes standard convolution. CNNs are thoroughly described by Schmidhuber [2015]; Goodfellow et al. [2016].

Pooling layers Convolutional layers are often combined with pooling layers which perform some sort of data sub-sampling in order to reduce the spatial dimensionality of the data. The most common types of pooling are the *average* and *max* pooling, which take the *mean* and *maximum* respectively, and are applied over a fixed-size receptive field of the inputs.

Activation functions are chosen depending on a problem at hand. If the model is being trained to predict unbounded real-valued outputs, activation is left as the identity function. Should one desire to bound the output of the model in a certain range, a variety of activation functions have been proposed, such as sigmoid, hyperbolic tangent, Rectified Linear Unit [Nair and Hinton, 2010], Exponential Linear Unit [Clevert et al., 2015] and others. The outputs of the model can also represent a categorical probability distribution over a set of labels, which is realized by using the softmax activation function.

3.3 Training neural networks

Training of a neural network is the process of adjusting the parameters θ in order to find the best approximation of the desired function f on a given dataset. The function we desire to approximate is defined by so - called *objective function*, which computes the scalar loss we are trying to minimize during learning. The most common choices of objective functions are Binary Cross-Entropy loss (BCE), Siamese [Bromley et al., 1993; Chopra et al., 2005; Hadsell et al., 2006] or Triplet loss [Wang et al., 2014b; Hoffer and Ailon, 2015], Mean Squared Error (MSE), etc.

In order to prevent the phenomenon of overfitting [Caruana et al., 2000], one needs to employ different regularization techniques, such as weight decay, dropout [Srivastava et al., 2014], early stopping [Holmström, 1989; Caruana et al., 2000] or batch normalization [Ioffe and Szegedy, 2015].

The parameters of a neural network are typically optimized using gradient descent, which however requires the computation of derivatives of the objective function. The most common method to efficiently compute derivatives of the neural network outputs with respect to the inputs is called *backpropagation* [Linnainmaa, 1976; Werbos, 1982]. It is essentially a backward pass through the neural network (passing through the layers in the reverse order of the forward pass using the chain rule of derivatives). Backpropagation computes derivatives for each layer's activations and parameters using activations stored during the forward pass and the intermediate derivatives computed so far, which equals the complexity of the forward pass and makes the backward propagation efficient. The backpropagation is described in detail by Schmidhuber [2015]; Goodfellow et al. [2016].

3.4 Learning paradigms

Based on the problem at hand and the available data one has to choose the best-suited learning paradigm. These can be divided into three main groups: supervised learning, unsupervised learning and semi-supervised learning. The following sections focus on the first two groups, while for the description of the last group, we refer a reader to Goodfellow et al. [2016].

3.4.1 Supervised learning

In the supervised setting, the learning algorithm is provided with labelled data, which allows it to accurately measure the performance at each step. Supervised learning is mainly useful in the following areas:

Classification Given a set of labels $\mathbb{Y}$ and some input data, the algorithm is asked to predict a label $y \in \mathbb{Y}$ for each input sample. Classification models are commonly trained using the Cross-Entropy (CE) objective function, defined as:

$$\text{CE}(\mathbf{y}, \hat{\mathbf{y}}) = \sum_{(x,y)} -\log p(\hat{\mathbf{y}} = \mathbf{y} | \mathbf{x}, \Theta), \tag{3.5}$$

where $\mathbf{y}$ is a one-hot vector encoding the target label y, $\hat{\mathbf{y}}$ is the output vector of predictions, $\mathbf{x}$ is input sample and Θ are model parameters.

Regression In regression problems, we look at continuous data. Given a particular input sample $\mathbf{x}$, we are trying to estimate the expected value of a variable $\mathbf{y}$. The hypothesis, therefore, is that there is a correlation between variables $\mathbf{x}$ and $\mathbf{y}$. Regression models can be trained for example with Mean Squared Error (MSE) objective:

$$\text{MSE}(\mathbf{y}, \hat{\mathbf{y}}) = \frac{1}{n} \sum_{i=1}^{n} (\mathbf{y}_i - \hat{\mathbf{y}}_i)^2. \tag{3.6}$$

Metric learning Aims to learn a distance metric that well establishes the notion of similarity/dissimilarity between groups of objects. On one hand, it tries to decrease the distance between similar objects, and on the other hand, it aims to increase the distance between dissimilar objects. The optimization is usually realized using the *Triplet loss*:

$$L = \frac{1}{2} ||\mathbf{x} - \mathbf{x}_{pos}||^2 + \max\{0, M - ||\mathbf{x} - \mathbf{x}_{neg}||^2\}, \tag{3.7}$$

where $(\mathbf{x}, \mathbf{x}_{pos}, \mathbf{x}_{neg})$ are a *triplet*. The M is the separating margin that is being enforced between similar and dissimilar samples.

3.4.2 Unsupervised learning

In unsupervised learning, the model is provided data without any specific annotations or guidelines on correct behavior. As there is no desired outcome predefined by the dataset, the model is handed the raw data and attempts to discover

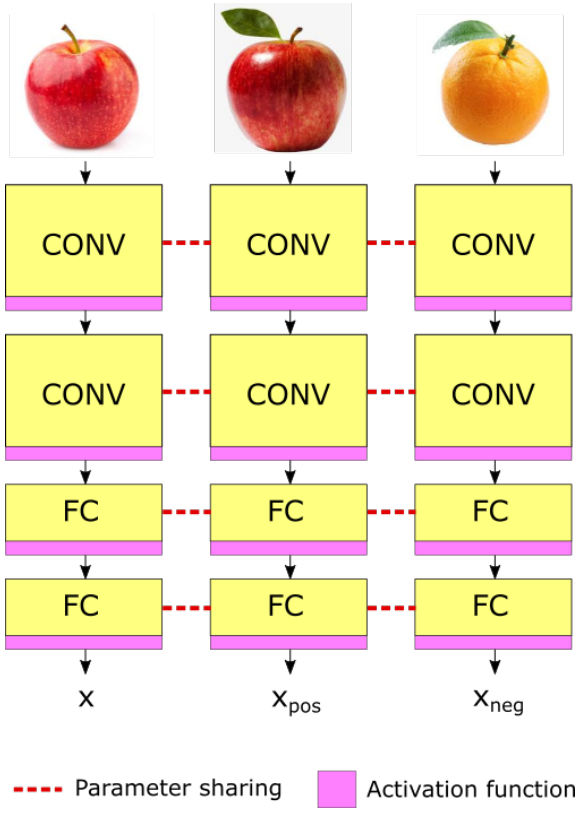


Figure 3.2. Illustration of Siamese/Triplet learning, where the same network is run multiple times for triplets of inputs producing three outputs x, x_{pos} and x_{neg} having x and x_{pos} coming from the same category and x and x_{neg} being two different categories..

the main factors of variation that explain the data at hand. It is typically used for data clustering, anomaly detection or learning embeddings of the data, which are shortly described next.

Clustering In clustering, the deep learning model tries to learn a measure of similarity between the training data samples and group them together accordingly. Approaches to deep clustering in different domains have been proposed by Tian et al. [2014]; Xie et al. [2015]; Hershey et al. [2016]; Caron et al. [2018].

Self-supervised methods Self-supervised learning aims at getting supervision from the unlabeled data itself by exploiting the intrinsic structure of the domain to generate target signals. This can be done by e.g. withholding some part of

the data and train the network to predict the missing part. It is widely used in natural language processing [Mikolov et al., 2013; Devlin et al., 2019], where it can be formulated as predicting next word in a sentence given the previous sequence of words. Only recently, successful applications in computer vision and image processing started appearing as well [Hjelm et al., 2018; van den Oord et al., 2018].

Autoencoder Autoencoders are amongst the most common architectures employed in unsupervised learning tasks. They are neural network architectures trained to output an identical reconstruction of its input (see Figure 3.3). Such architectures typically consist of two parts, the *encoder* function $\mathbf{z} = f(\mathbf{x})$ which maps the input $\mathbf{x}$ to an internal code called latent representation and a *decoder* function $\hat{\mathbf{x}} = g(\mathbf{z})$ that aims to reconstruct the original input $\mathbf{x}$ given its latent representation $\mathbf{z}$. Convolutional autoencoders were introduced by Masci et al. [2011] and have established state-of-the-art results in many fields of computer vision over the last years. A broader overview of autoencoder architectures is provided in Schmidhuber [2015]; Goodfellow et al. [2016].

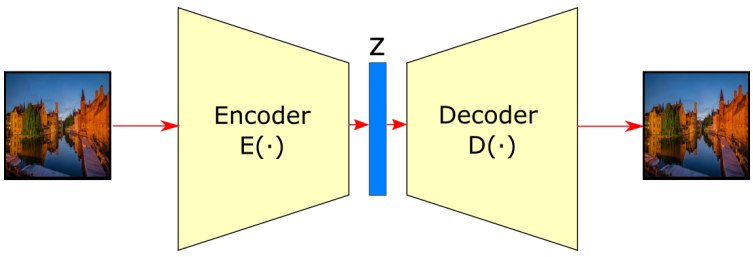


Figure 3.3. Typical autoencoder neural network architecture consists of an encoder E that produces a latent representation z, and a decoder D which takes the latent representation and reconstructs back the original input.

3.5 Geometric deep learning

Many fields of science focus on data with non-Euclidean underlying structure (e.g. social networks, brain imaging, meshed surfaces, etc.). Witnessing the success of deep learning models in one- and two-dimensional signal processing, there has been an increasing interest in generalization of the powerful deep learning to non-Euclidean structured data such as graphs and manifolds. This gave rise to a new field referred to as *geometric deep learning* in Bronstein et al. [2017a]. The

first attempts of learning on graph structures date back to the works of Goller and Küchler [1995]; Küchler and Goller [1996]; Sperduti and Starita [1997]; Frasconi et al. [1998]; Gori et al. [2005]; Scarselli et al. [2009], where the authors considered steady-state of learnable diffusion process (recent works of Li et al. [2016] and Gilmer et al. [2017] improved this approach using modern deep learning schemes). Since then, much work has been done introducing novel approaches, which can be divided into two main computation paradigms presented below.

3.5.1 Spectral domain graph CNNs

They were developed by Bruna et al. [2014] and Henaff et al. [2015] by formulating convolution-like operations in the spectral domain, defined by eigenvectors of the graph Laplacian. The notable drawback of this architecture is $\mathcal{O}(n^2)$ computational complexity yielded by the cost of computing the forward and inverse graph Fourier transform. Furthermore, such architectures have $\mathcal{O}(n)$ parameters per layer, and no guarantee of spatial localization of the filters.

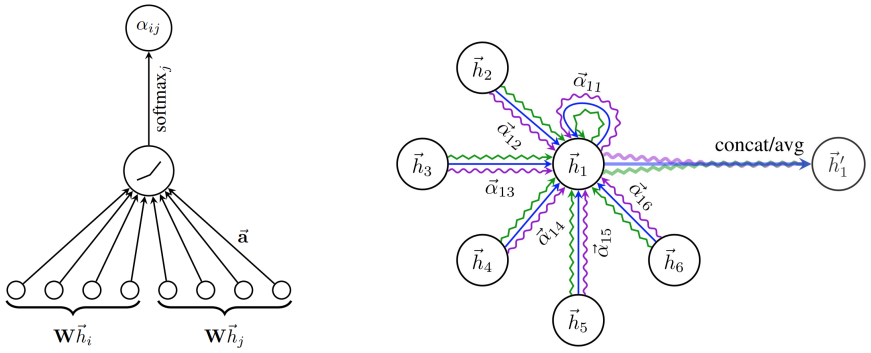
In order to mitigate the drawbacks of previous works, Defferrard et al. [2016]; Kipf and Welling [2017a] and others have proposed spectral filters that can be expressed in terms of simple operations (additions, scalar and matrix multiplications) w.r.t. the Laplacian. In particular, Defferrard et al. [2016] considered polynomial filters of degree p, which incur only p times multiplications by the Laplacian matrix (having cost $\mathcal{O}(|\epsilon|)$ in general or $\mathcal{O}(n)$ if the graph is sparsely connected), also guaranteeing filters that are supported in p-hop neighborhoods. Kipf and Welling [2017a] further simplified this idea, proposing Graph Convolutional Network (GCN) reducing to only 1st order approximation. This work was later extended with attention mechanism by Veličković et al. [2018] in Graph Attention Network (GAT). Levie et al. [2018] proposed rational filter functions including additional inversions of the Laplacian, which were carried out approximately using an iterative method. Monti et al. [2018] used multivariate polynomials w.r.t. multiple Laplacians defined by graph motifs; Monti et al. [2017b] used Laplacians defined on products of graphs in the context of matrix completion problems. Approaches that are the most relevant for us are shortly described below.

Graph Convolutional Network (GCN) Proposes a way of reducing the computational complexity of spectral graph CNNs, which require explicit computation of the Laplacian eigenvectors, a very costly operation, especially for a larger graph. Kipf and Welling [2017a] proposed to circumvent the problem by using a

1st order approximation of the spectral filter. For a signal $\mathbf{X} \in \mathbb{R}^{N \times C}$ with C input channels, the convolution with F filters can be written as:

$$\mathbf{Z} = \tilde{\mathbf{D}}^{-\frac{1}{2}} \tilde{\mathbf{A}} \tilde{\mathbf{D}}^{-\frac{1}{2}} \mathbf{X} \Theta, \tag{3.8}$$

where $\Theta \in \mathbb{R}^{C \times F}$ is matrix of filter parameters and $\mathbf{Z} \in \mathbb{R}^{N \times F}$ is the output signal matrix. $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}_N$, where $\mathbf{A}$ is the adjacency matrix and $\mathbf{I}_N$ an identity, and $\tilde{\mathbf{D}}_{ii} = \sum_j \tilde{\mathbf{A}}_{ij}$.



(a) (b)

Figure 3.4. GAT graphically: (a) The attention mechanism employed by the GAT model. (b) Illustration of multi-head attention of the node h_1 on its neighborhood (different line colors correspond to different attention heads). [Veličković et al., 2018]

Graph Attention Network (GAT) Introduces an attention mechanism as a substitute to the statically normalized "convolution" used in GCN. Veličković et al. [2018] defined the convolution operation performed by GAT as:

$$\mathbf{h}'_i = \sigma\left(\sum_{j \in N_i} \alpha_{ij} \mathbf{W}\mathbf{h}_j\right), \tag{3.9}$$

where $\mathbf{h}_i, \mathbf{h}_j \in \mathbb{R}^F$ are the node features for nodes i and j respectively, N_i is the neighborhood of node i in the graph, $\mathbf{W} \in \mathbb{R}^{F' \times F}$ is the weight matrix and α_{ij} is the attention coefficient between nodes i and j which are computed as follows:

$$\alpha_{ij} = \frac{\exp\left(\text{LeakyReLU}\left(\mathbf{a}^T \left[\mathbf{W}\mathbf{h}_i || \mathbf{W}\mathbf{h}_j\right]\right)\right)}{\sum_{k \in N_i} \exp\left(\text{LeakyReLU}\left(\mathbf{a}^T \left[\mathbf{W}\mathbf{h}_i || \mathbf{W}\mathbf{h}_k\right]\right)\right)}, \tag{3.10}$$

where $\mathbf{a} \in \mathbb{R}^{2F'}$ is the weight vector parametrizing the attention mechanism and $||$ is the concatenation operation. Complexity of the approach is reduced by performing so-called masked attention, which computes the attention mechanism only for nodes $j \in N_i$. The main principles of GAT are depicted graphically in Figure 3.4.

3.5.2 Spatial domain graph CNNs

Spatial domain graph CNNs operate on local neighborhoods on the graph [Duvenaud et al., 2015; Atwood and Towsley, 2016; Hamilton et al., 2017; Wang et al., 2019; Kipf and Welling, 2017b]. Monti et al. [2017a] proposed the Mixture Model networks (MoNet), generalizing the notion of image patches to graphs. The construction is based on a system of local pseudo-coordinates $\mathbf{u}_{ij} \in \mathbb{R}^d$ assigned to a neighbor j of each vertex i. The convolution operation is in this case redefined as a Gaussian mixture in these coordinates.

In the last years, special deep neural networks operating directly on point cloud data, which are designed to handle the irregularity of point clouds, have been designed. The pioneering approach in this area is PointNet [Qi et al., 2016], where the invariance to permutation of points is achieved by processing each point independently and accumulating features by applying a symmetric function. There have been various extensions to PointNet, which allow exploiting local features by considering neighborhoods of points rather than operating on each point independently [Qi et al., 2017; Shen et al., 2017]. They, however, treat points in local neighborhood independently to allow permutation invariance, which neglects the geometric relationships among points. This limitation has been mitigated by Wang et al. [2019], who treats the local neighborhood by constructing graph over it, which describes the geometric relationships on the local level. The idea of learning the underlying graph structure of the data has been extended by introducing a fully differentiable graph module (DGM) by Kazi et al. [2020], which allows learning a function that dynamically predicts the edge probabilities in the graph relevant for the task at hand. Approaches that are the most relevant for this work are shortly described below.

PointNet++ Successor of PointNet, which learns a spatial encoding of individual points and aggregates the pointwise features into a global feature vector describing the whole point cloud, which however does not capture the local structure induced by the metrics. PointNet++ applies its predecessor PointNet hierarchically at differently subsampled versions of the input point cloud. This

results in so-called *Set Abstraction layer* (see Figure 3.5), which combines the following operations into a single step:

- **Sampling:** Selects a set of points which will be the centroids of the new local regions. The sampling is typically realized using Farthest Point Sampling (FPS).
- **Grouping:** Assigns points to local regions by finding neighbors around the region centroids. This is usually done using either *ball of radius r* or *k-Nearest Neighbors* search.
- **PointNet:** Realizes the PointNet operation on the local regions in order to encode them into feature vectors.

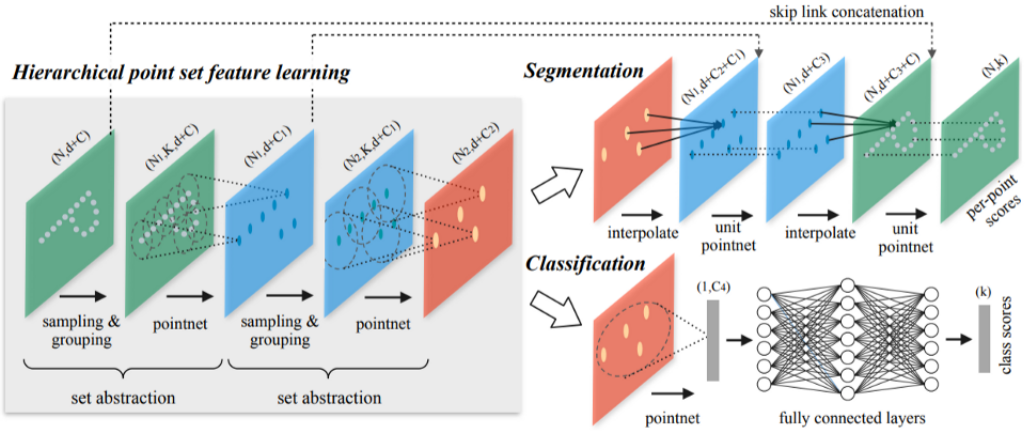


Figure 3.5. PointNet++ architecture comprising of multiple Set Abstraction Layers and an additional sub-networks which have different structure depending on the application: segmentation or classification. [Qi et al., 2017]

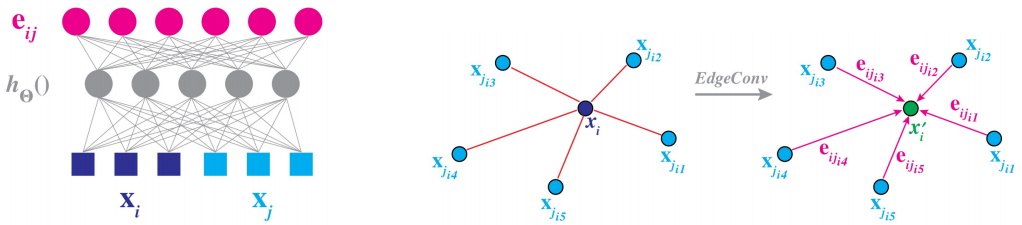
Dynamic Graph CNN (DGCNN) Introduces a new neural network module called *EdgeConv*, which exploits the local geometric structure by constructing a local neighborhood graph and applying operations on edges connecting neighboring pairs of points. Such an approach yields the properties of translation invariance and non-locality. The neighborhood graph is not fixed, but it is dynamically updated after each layer of the network using the *k-Nearest Neighbors*

search, which allows capturing changes in the neighborhood structure from one layer to another based on the current embedding.

The *EdgeConv* operation aggregates the edge features associated with edges from each connecting vertex. The *edge features* are defined as $\mathbf{e}_{ij} = h_\Theta(\mathbf{x}_i, \mathbf{x}_j)$, where $h : \mathbb{R}^F \times \mathbb{R}^F \to \mathbb{R}^{F'}$ is some parametric non-linear function with a set of learnable parameters Θ and $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^F$ are two vertices i and j respectively. The operation is illustrated in Figure 3.6 and the output of *EdgeConv* for a vertex $\mathbf{x}_i$ is defined as:

$$\mathbf{x}'_i = \sum_{j \in N_i} h_\Theta(\mathbf{x}_i, \mathbf{x}_j), \tag{3.11}$$

where N_i is the set of vertices neighboring vertex i. The *EdgeConv* operation can be applied multiple times interleaved with other operations, such as pooling or spatial transformer, depending on the task at hand.



(a) (b)

Figure 3.6. DGCNN operations: (a) Computation of an edge feature e_{ij} from a pair of vertices x_i, x_j. Here, the h_Θ is realized as a fully connected layer with learnable parameters Θ; (b) Explanation of the *EdgeConv* operation, whose output is calculated by aggregating the edge features associated with the i-th vertex neighborhood. [Wang et al., 2019]

3.6 The curse of adversarial examples

It has been discovered by Szegedy et al. [2014] that the deep neural networks, achieving state-of-the-art results in many computer vision tasks, tend to learn very discontinuous mappings from input to the output, which can result in misclassification of an image while influenced by just a barely perceptible adversarial perturbation. Many methods for finding such perturbations have been proposed to date [Moosavi-Dezfooli et al., 2016, 2017; Su et al., 2019].

Discovery of this phenomena has potentially very dramatic implications on the security of deep learning, undermining its suitability for security applications. As a consequence, adversarial attacks and defenses against them have become a very active topic of research [Rauber et al., 2017; Kurakin et al., 2018].

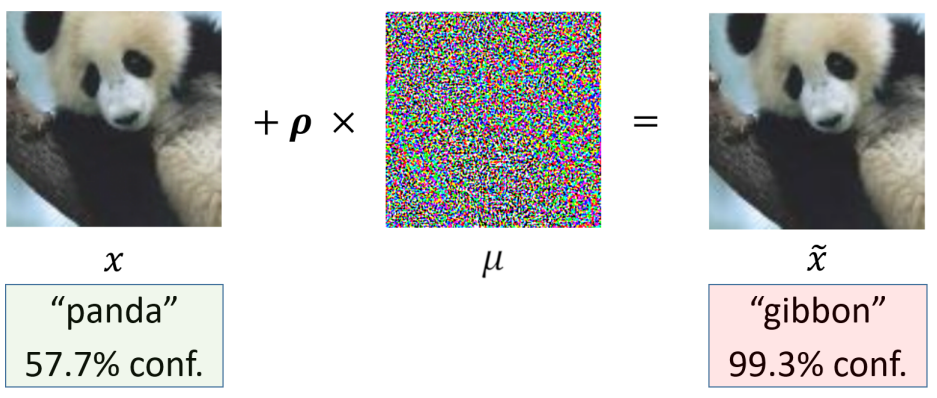


Figure 3.7. An input image of panda was recognized as panda with almost 60% confidence. After adding sufficient amount of adversarial noise μ controlled by parameter ρ, the network will wrongly recognize a gibbon with nearly 100% confidence. [Goodfellow et al., 2015]

3.6.1 Adversarial noise

Let us first explain what adversarial noise is. Szegedy et al. [2014] has explained the vulnerability of neural networks to subtle changes in the input using a simple linear model as:

$$\mathbf{w}^T\tilde{\mathbf{x}} = \mathbf{w}^T\mathbf{x} + \mathbf{w}^T\mu, \tag{3.12}$$

where $\mathbf{w}$ are the network weights, $\mathbf{x}$ is the input feature vector, μ is an adversarial noise vector and $\tilde{\mathbf{x}}$ is the input feature vector corrupted by the noise. Adversarial noise is structured noise (see examples in Figure 3.8) which is generated based on the desired outcome. The maximum norm of μ will not grow with increasing dimensionality. However, as the problem size increases, many infinitesimal changes in $\mathbf{x}$ may accumulate into one significant change in $\tilde{\mathbf{x}}$. This is dangerous as the linear model is forced to attend exclusively the signal which aligns the most with its weights, despite the fact that there would be some alternative, even better, choices. It implies that if a linear model has sufficiently high dimensionality, it can have adversarial examples. Neural networks we encounter in machine

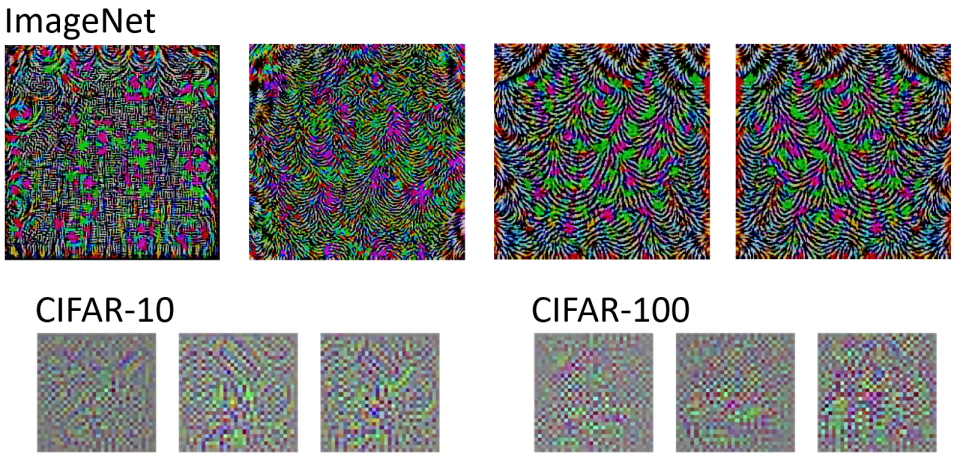


Figure 3.8. Examples of universal adversarial noise introduced by Moosavi-Dezfooli et al. [2017] generated for images from various datasets.

learning are highly non-linear models, which are however designed to behave in a very linear way. As Szegedy et al. [2014] explains, this suggests that a simple perturbation of a linear model should damage non-linear models as well (see an example in Figure 3.7).

3.6.2 Adversarial attacks

Adversarial attacks are methods designed to foster adversarial examples. An adversarial attack is typically an optimization process, which aims at generating adversarial noise that will result in a shift of the decision boundary of the neural network model classifier. Their performance is measured in terms of so-called *fooling rate*, which is the number of images for which the resulting label changes under the effect of the generated adversarial noise.

They can be categorized in several different ways which will be described next. First, we can divide adversarial attacks into two groups based on the *desired outcome* as:

- *Non-targeted:* Given an input sample $\mathbf{x}$ and conditional probability over the labels $p(\mathbf{x}|x)$, the goal is to generate an adversarial example $\tilde{\mathbf{x}}$ such that $p(\tilde{\mathbf{x}}|\tilde{y})$ is highest for any $\tilde{y} \neq y$, while keeping the strength of the perturbation small enough, i.e. $||\tilde{\mathbf{x}} - \mathbf{x}|| \leq \epsilon$.
- *Targeted:* Given an input sample x and conditional probability over the labels $p(\mathbf{x}|x)$, the goal is to generate an adversarial example $\tilde{\mathbf{x}}$ such that

$p(\tilde{\mathbf{x}}|\tilde{y})$ is highest for some class $\tilde{y} = A$ where $\tilde{y} \neq y$, while keeping the strength of the perturbation small enough, i.e. $||\tilde{\mathbf{x}} - \mathbf{x}|| \leq \epsilon$.

Another way of categorizing adversarial attacks is by the amount of knowledge about the model they work with:

- *White-box:* Attacks that have perfect knowledge of the model (e.g. they have access to the architecture, model parameters, optimization process, etc.).
- *Black-box:* Attacks that generally view the model as a black-box. They have limited or no knowledge of the model.

Now that the categorization has been introduced, we will spend a few paragraphs on briefly introducing some of the most common adversarial attack methods.

Gradient-descent attack generates an adversarial example by solving a constrained optimization problem using gradient descent algorithm. It can be defined by the following formula:

$$\tilde{\mathbf{x}} \leftarrow \tilde{\mathbf{x}} + \epsilon \cdot \nabla \log p(\tilde{y}|\tilde{\mathbf{x}}),$$

where ten ϵ values are scanned in ascending order, until the algorithm either succeeds or fails.

Fast Gradient Sign Method (FGSM) proposed by Goodfellow et al. [2015] generates an adversarial example $\tilde{\mathbf{x}}$ for an input $\mathbf{x}$ with label y as follows:

$$\tilde{\mathbf{x}} \leftarrow \mathbf{x} + \epsilon \operatorname{sign}(\nabla_{\mathbf{x}} J(\theta, \mathbf{x}, y)),$$

where $J(\theta, \mathbf{x}, y)$ is the loss function used during training (e.g. cross-entropy as above), and ϵ controls the magnitude of the perturbation.

Projected Gradient Descent (PGD) represents a much stronger adversary as it lifts any constraints on how much time the adversary has to find the best attack. It is often referred to as a multi-step FGSM^k method [Madry et al., 2017]. Formally, given an input $\mathbf{x}$ with label y, a single iteration generating adversarial sample $\mathbf{x}^{t+1}$ is defined:

$$\mathbf{x}^{t+1} \leftarrow \prod_{\mathbf{x}+S} (\mathbf{x}^t + \epsilon \operatorname{sign}(\nabla_{\mathbf{x}} J(\theta, \mathbf{x}, y))),$$

where $J(\theta, \mathbf{x}, y)$ is the training loss and ϵ controls the magnitude of the perturbation. Operator $\prod$ defines the projection on S-ball around $\mathbf{x}$.

Universal Adversarial Perturbation Moosavi-Dezfooli et al. [2017] has designed an attack that is able to generate a universal adversarial perturbation $\mathbf{v}$, which works across different network architectures and datasets. The authors have provided a publicly available implementation in their toolbox *DeepFool*. The strength of the perturbation was bound by $||\mathbf{v}||_2 \leq \rho \mathbb{E}||\mathbf{x}||_2$, where the expectation is taken over the set of training images used to generate the attack and the parameter ρ controls the strength.

3.6.3 Defense against adversarial attacks

Many have recently tried to theoretically explain the phenomena [Goodfellow et al., 2015; Fawzi et al., 2018] and propose strategies to defend against adversarial perturbations. The efforts have focused on network architecture [Mahdizadehaghdam et al., 2018; Dhillon et al., 2018], training procedure [Tramèr et al., 2018] and data pre-processing [Meng and Chen, 2017; Zantedeschi et al., 2017].

Lately, Athalye et al. [2018] have pointed out the problem of so-called *gradient obfuscation*, which has to be taken into consideration while designing a new defense method. The authors have proven that many state-of-the-art methods, in fact, obfuscate the gradient and that causes them to, seemingly, work. We propose a novel defense mechanism which addresses the aforementioned problem later in Chapter 6.

Chapter 4

Exploiting palmprint images using discriminative index learning

This chapter is based on Svoboda et al. [2016]. It begins by going over one of the first approaches to efficiently apply CNNs for palmprint recognition. In the majority of cases, 3D scanners provide RGB-D streams, which contain not only the depth information but an intensity image as well. Besides fusing different types of 3D features, one can think of fusing the 2D information, obtained from an intensity image, together with 3D information, which is provided by the range map. This type of fusion has been shown to be powerful Kanhangad et al. [2011]. What types of features one can extract from the intensity image heavily depends on its resolution. Off-the-shelf, low-cost RGB-D sensors often provide considerably-noisy depth maps; therefore, it is advantageous to fuse the geometry with analysis of the intensity image. Even though intensity images from RGB-D sensors are not of relatively high quality, even just extraction of the principal lines together with some more prominent wrinkles gives valuable information when combined with the depth.

To this end, we propose the use of a convolutional architecture while training the model with a novel loss function closely related to the *d*-prime discrimination index from biometrics, which allows achieving a better genuine/impostor separation of the score distributions. As opposed to previous learning-based approaches working on the computed scores directly (e.g. the Siamese loss [Bromley et al., 1993; Chopra et al., 2005]), the genuine/impostor score distributions are approximated as normal distributions, and the maximum separation between the distributions is enforced by means of pushing the means apart while maintaining small standard deviations among both distributions.

We evaluate our method on standard benchmarks in palmprint recognition

and show competitive performance with respect to the current state-of-the-art. By considering the entire distributions, rather than individual scores themselves, this approach learns a more general representation of palmprints and generalizes better to new, previously unseen, subjects.

4.1 Discriminative Index Optimization

The key goal of biometric recognition system based on metric learning approaches is achieving a separation of the genuine and impostor score distributions. As explained in Section 2.2.1, a popular criterion in literature is *d-prime* (also referred to as *sensitivity-* or *discriminative index*) [Daugman, 2000a], modeling the genuine and impostor score distributions as normal distributions $\mathcal{N}(\mu_{gen}, \sigma_{gen})$ and $\mathcal{N}(\mu_{imp}, \sigma_{imp})$ respectively, where μ_{gen}, μ_{imp} are the means and $\sigma_{gen}, \sigma_{imp}$ the standard deviations. The separation of the normal distributions can be measured as

$$d' = \frac{\mu_{imp} - \mu_{gen}}{\sqrt{\frac{1}{2}(\sigma_{imp}^2 + \sigma_{gen}^2)}}. \tag{4.1}$$

Well-separated distributions in the *d-prime* sense should thus have distant means and small variances. Based on the discriminative index, this work proposes a novel similar criterion of separation referred to as *d-prime* loss,

$$l = \sigma_{gen} + \sigma_{imp} + \mu_{gen} + \max\{0, M - \mu_{imp}\}, \tag{4.2}$$

where the last term is a standard hinge loss trying to pull the genuine and impostor means at least M apart.

Assuming that sample distances can be approximated by a normal distribution, *d*-prime loss provides better generalization and therefore allows to train successfully having only little training data available.

In order to be able to use a gradient-descent optimization algorithm, the gradients of the *d*-prime loss for a batch of samples have to be derived, as follows:

$$\frac{\partial l}{\partial d_{gen}^i} = \frac{1}{N}\Big[2(d_{gen}^i - \mu_{gen}) + 1\Big] \tag{4.3}$$

$$\frac{\partial l}{\partial d_{imp}^i} = \frac{1}{N}\Big[2(d_{imp}^i - \mu_{imp}) - \mathbf{J}(\mu_{imp} < M)\Big], \tag{4.4}$$

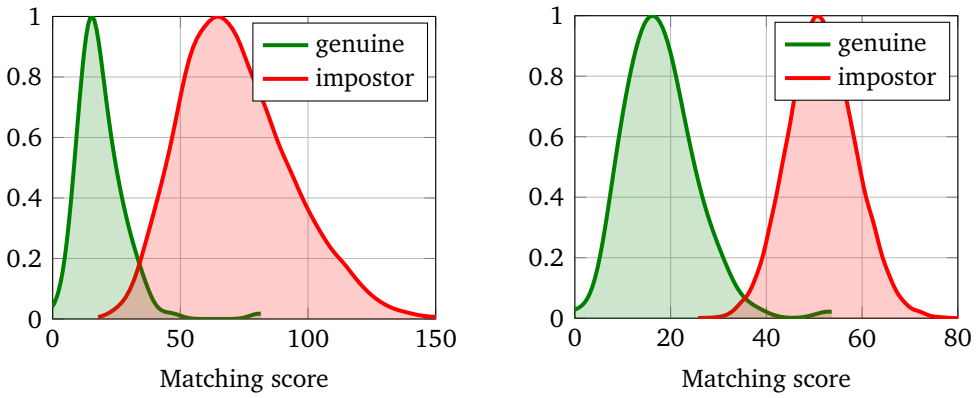
where

$$\mu_{gen} = \frac{1}{N}\sum_{i=1}^{N} d^i_{gen}, \qquad \mu_{imp} = \frac{1}{N}\sum_{i=1}^{N} d^i_{imp},$$
$$\sigma^2_{gen} = \frac{1}{N}\sum_{i=1}^{N} (d^i_{gen} - \mu_{gen})^2, \qquad \sigma^2_{imp} = \frac{1}{N}\sum_{i=1}^{N} (d^i_{imp} - \mu_{imp})^2$$

are the means and standard deviations of the normal distributions and

$$d^i_{gen} = ||\mathbf{x}^i - \mathbf{x}^i_+||, \qquad d^i_{imp} = ||\mathbf{x}^i - \mathbf{x}^i_-||$$

are the genuine and impostor distances respectively. N is the number of samples in a training batch, $\mathbf{J}$ is full unit matrix and $\mathbf{x}_+$ is a feature vector belonging to the same class as $\mathbf{x}$, while and $\mathbf{x}_-$ belongs to a class different from $\mathbf{x}$.



(a) Siamese loss (b) d-prime loss

Figure 4.1. Score distributions produced with a CNN trained using: (a) the classical Siamese and (b) the proposed d-prime loss. Notice the better separation of the genuine/impostor distributions with the new loss.

4.2 Comparison to Siamese networks

Figure 4.1 shows the difference between the genuine and impostor score distributions produced by our CNN using the novel d-prime loss function compared to the traditional *Siamese loss* [Bromley et al., 1993; Chopra et al., 2005] which is applied to individual scores rather than to distributions

$$l_{siam} = \frac{1}{2}||\mathbf{x} - \mathbf{x}^+||^2 + \max\{0, M - ||\mathbf{x} - \mathbf{x}^-||\}, \tag{4.5}$$

where $\mathbf{x}, \mathbf{x}^+$ and $\mathbf{x}^-$ denote the feature vectors of a genuine, and an impostor, subject respectively. The newly proposed *d*-prime loss results in a better separation of the genuine and impostor score distributions.

4.3 Experiments

This section presents an extensive evaluation of our method on several different standard benchmarks. A short review of the datasets used in this work is presented first. Next, the CNN architecture used in all our experiments is discussed. Subsequently, we compare our performance to some of the existing state-of-the-art methods. Finally, we provide visual reasoning behind what has the model learnt.

4.3.1 Datasets

The proposed approach was evaluated on two standard contactless palmprint datasets.

The *IIT Delhi database* [Kumar, 2008; Kumar and Shekhar, 2011], which contains 5 samples of segmented palmprint images for each of left and right hands of 230 different subjects. Only right hand samples are used, and the dataset is split into disjoint equally-sized training and test sets, containing different subjects. Two-fold cross-validation is employed, always having 50% of subjects for training and 50% for testing. This process is repeated 8 times having a different split of the subjects in each iteration (in each iteration, there are 5 samples for each subject and therefore five times leave-one-out cross-validation is performed followed by averaging the results to get the performance for the current iteration).

The *CASIA database* [Sun et al., 2005] contains 5,502 images captured from 312 different subjects with approximately 9 images per subject (both left and right hands). The dataset did not provide extracted palmprint regions. A palmprint region-of-interest (ROI) extraction algorithm described in Zhang et al. [2003] was used and only the subjects for which at least 5 images were successfully extracted were kept (successful extraction was assumed if the ROI coordinates were within image bounds). This results in a dataset including 283 subjects for the right and 282 subjects for the left hand, always with 5 images per subject. The splitting and cross-validation settings are the same as for the IIT Delhi dataset.

4.3.2 CNN architecture

In all experiments, a CNN that is applied to a 128 × 128 image of palmprint region extracted using methods described in Genovese et al. [2014] has been used. The CNN architecture is a sequence of convolutional layers applying banks of filters to the input image or the output of the previous layer, pooling (non-linear averaging and downsampling), and linear combination. The weights of the filters are learnable parameters of the network, selected by an optimization procedure minimizing a task-specific loss function, the *d*-prime loss.

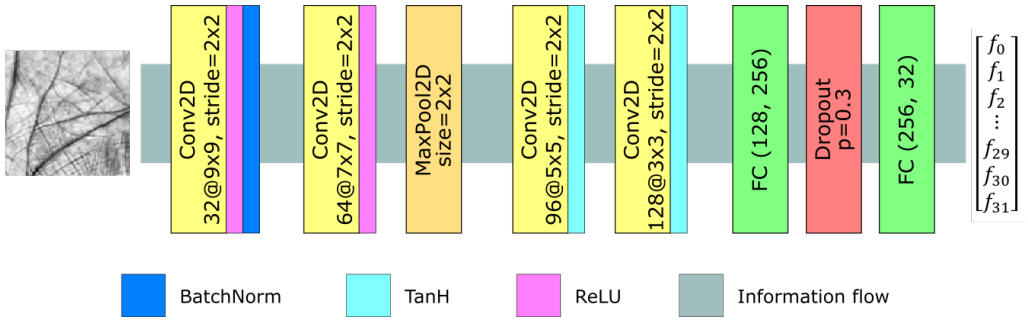


Figure 4.2. The CNN architecture, containing four convolutional layers followed by two fully connected layers and outputting a 32-dimensional feature vector.

The architecture of the network is illustrated in Figure 4.2. It comprises of four convolutional, one pooling, one fully connected, and an output layer of 32 units. The output of the CNN is thus a 32-dimensional feature vector describing the input palmprint image. The first two convolutional layers are a simplified version of the first two layers of AlexNet [Krizhevsky et al., 2012]. The batch-normalization is applied only after the first layer's ReLU, as applying it also after the successive layers showed neither performance nor convergence improvement. The output of the last convolutional layer is fed to the linear fully connected layer with a dropout of 30%. This layer is linked to the last fully connected layer with 32 linear output units. The method thus produces feature vectors of 32 values with 32-bit floating-point precision, which is 128-bytes in total.

4.3.3 Performance evaluation

Table 4.1 and Table 4.2 summarize the performance of the compared methods on the IIT Delhi and CAISA datasets respectively. The proposed *d*-prime CNN

method outperforms the other approaches in terms of EER and performs well also in terms of the *d*-prime index. The output feature vector size of the *d*-prime approach is 128 bytes, the smallest amongst the compared methods. Figure 4.3 shows the ROC curves of different methods. The novel method achieves significantly lower false acceptance rates, which indicates its potential in large-scale applications. Figure 4.4 depicts the genuine and impostor score distributions for different approaches. The aim is to reduce the overlap of the two distributions, which is speaking clearly in favor of the *d*-prime based solution.

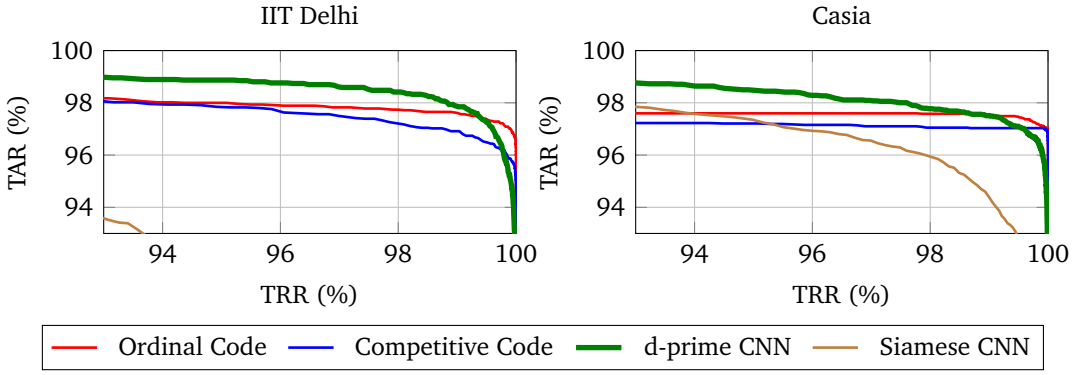


Figure 4.3. ROC curve (tradeoff between acceptance and rejection rates) of the evaluated methods on the IIT Delhi and Casia datasets.

Table 4.1. Performance in terms of EER (the lower the better).

Method	IIT Delhi	CASIA	Feature size
Competitive Code	2.33%	2.90%	384 Bytes
Ordinal Code	2.08%	2.41%	384 Bytes
Siamese CNN	6.08%	3.15%	128 Bytes
d-prime CNN	1.64%	1.86%	128 Bytes

4.3.4 Visual analysis of results

To validate our approach, visual inspection of information represented by the output feature vector is provided. One can follow Zeiler's visualization [Zeiler and Fergus, 2014] in our *d*-prime CNN in order to obtain a saliency map for each dimension of the output feature vector $\mathbf{x} \in \mathbb{R}$. The saliency map is produced

Table 4.2. Genuine/impostor distributions separability in terms of d-prime index (the higher the better).

Method	IIT Delhi	CASIA
Competitive Code [Kong and Zhang, 2004]	3.32	4.72
Ordinal Code [Sun et al., 2005]	3.92	5.75
Siamese CNN	2.91	2.20
d-prime CNN	4.84	5.73

to visually verify that dimensions of $\mathbf{x}$ represent meaningful information - some part of a palm line, wrinkle or their combination.

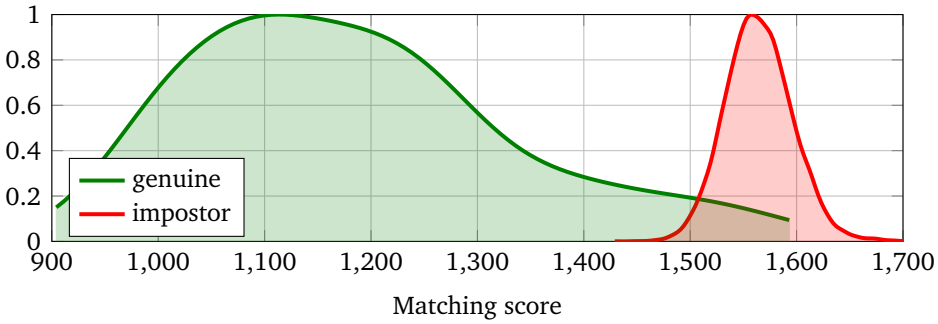
Figure 4.5 shows results for several individual dimensions of an output feature vector on the right. It is clearly visible that different elements of the feature vector represent different lines and wrinkles on the palmprint. The bottom leftmost image then shows a combination of visualizations from all 32 dimensions of the output feature vector. This suggests that the novel d-prime based method in combination with modified backpropagation can be used for palmprint segmentation as well.

4.4 Discussion

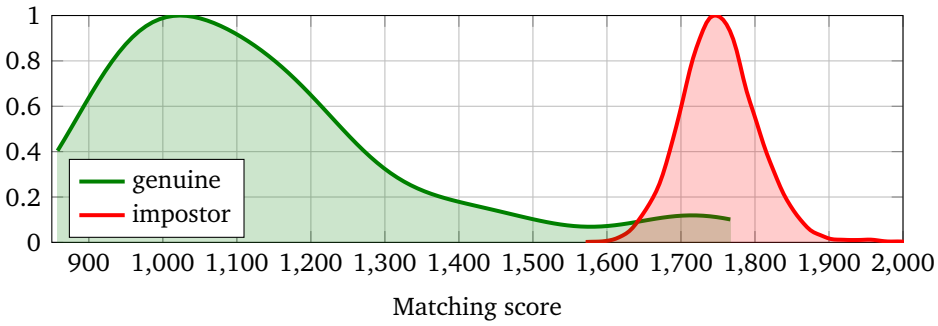
This chapter has explored the use of deep learning techniques for biometric palmprint image recognition. It presented a novel siamese-type convolutional neural network architecture, which is designed specifically for contactless palmprint. Unlike previous methods based on "handcrafted" features requiring manual adjustments, this approach automatically adapts the features from the data and does not need cumbersome parameter tuning. Instead of training the network with the traditional Siamese loss, it employs the novel d-prime loss, which aims to maximize the separation of the genuine and impostor score distributions. The aforementioned separation of score distributions results in improved recognition performance and better scalability.

Compared to its non-learning based counterparts, this approach is easily transferable across different datasets. The network architecture stays the same and it is only necessary to retrain the filters. This way, one can obtain very good classifier for any palmprint database without the need for tedious parameter tuning.

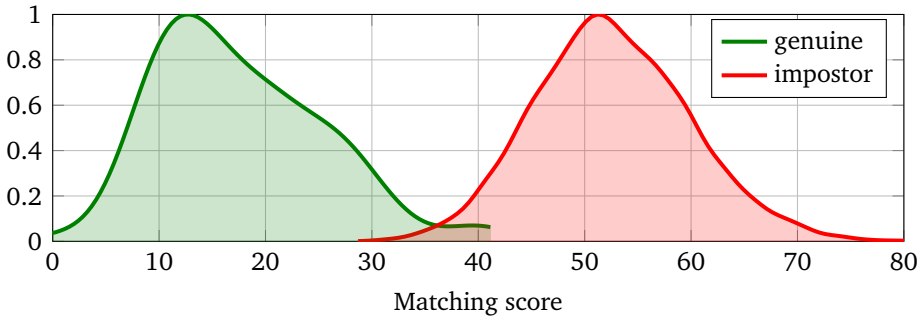
The results presented on the identification task are very encouraging. The



(a) Competitive code



(b) Ordinal code



(c) d-prime CNN

Figure 4.4. Score distributions on the IIT Delhi dataset for two baseline approaches (a,b) and the proposed method (c). Genuine and impostor distributions are shown in green and red, respectively. Higher score corresponds to smaller similarity between the palmprint images. The aim is to minimize the overlap between the two distributions, which is better achieved by d-prime in comparison to the other state-of-the-art approaches.

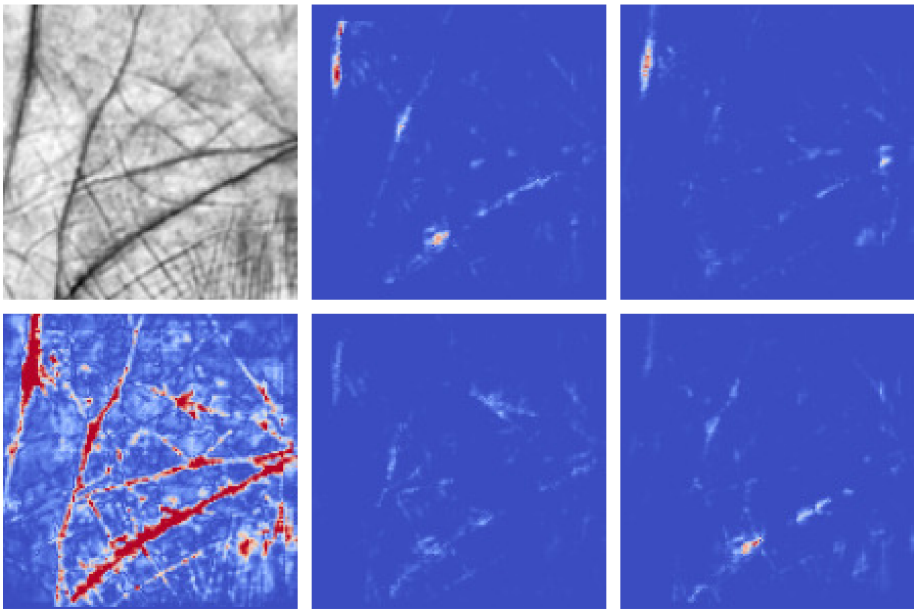


Figure 4.5. Representation of an input image by the output feature vector of our CNN. The input image is in the top row on the left, the whole representation by all the features is in the bottom row on the left (red color indicates the relative importance of the pixel). The remaining four images on the right are representation of the input image by four random dimensions of the output feature vector.

success of deep-learning-based approaches for contactless palmprint recognition has been later on followed by e.g. Ariyanto et al. [2018]; Wang et al. [2018]; Bensid et al. [2018]; Genovese et al. [2019]; Fei et al. [2019b]; Ungureanu et al. [2020]. Besides, the visual inspection of what the output feature vector represents supports the desire of exploiting this approach in palmprint segmentation tasks in the future. Last but not least, d-prime learning has presented itself as a general learning paradigm, which has been successfully adopted by many other works in various fields [Vijay Kumar et al., 2016; Wang et al., 2018a; Gainza et al., 2019; Guo et al., 2020].

Chapter 5

Towards low-quality fingerprint improvement

This chapter is based on Svoboda et al. [2017]. It presents a method to enhance and reconstruct missing details in low-quality partial fingerprint impressions – for example, latent fingerprints.

In addition to inspecting the palm of the hand, one can view the hand as a whole, including the fingers. In an ideal scenario, access to fingerprints – one of the oldest and most powerful biometric modalities – would be readily available. Fingerprints in conjunction with hand geometry and palmprint may result in a very robust, contactless biometric system. Contactless, image-based fingerprint sensing is an extremely challenging task, even more so in unconstrained environments. This may result in samples containing only partial fingerprints, very low-quality impressions, etc. Similar challenges are typical for processing of latent fingerprints, which we have decided to use as our model use-case.

The centerpiece of the model is an autoencoder architecture, which receives damaged fingerprints as input and learns to reconstruct the complete original. Such denoising autoencoder models, however, typically require a fair number of original/damaged pairs as training samples, which no existing fingerprint dataset provides. Training of our model is made possible by accumulating a large dataset of synthetic fingerprints using an open-source implementation of SFinGE [Cappelli et al., 2000] called Anguli [Ansari, 2011] and adding certain noise to the samples as post-processing. The experiments show it is possible to train a model able to reconstruct fingerprints fully on synthetic data, therefore overcoming the large database requirement. Fingerprints reconstructed using this method are then fed to several standard feature-extraction and matching pipelines, yielding state-of-the-art results on the IIIT Delhi latent fingerprint database [Sankaran et al., 2011, 2015].

5.1 Fingerprint similarity measure

The proposed method is based on Convolutional Autoencoders (CAE). Gradient analysis of the fingerprint image ridge pattern is, therefore, a natural choice as the computation of the image gradient can be implemented using the convolutional operator. In particular, gradient analysis allows extracting the ridge positions and orientations.

Ridge positions. Ridge positions can be recovered simply by computing gradient magnitude image using directional derivatives. Directional derivative of the fingerprint sample I is computed using directional kernel S_θ as follows:

$$G_\theta(I) = I \star S_\theta, \tag{5.1}$$

where $\theta \in \{0, 45, 90, 135\}$ is the gradient direction.

Criterion of similarity of a target image I_t and its reconstruction I_r is then defined as average Mean Squared Error (MSE) over all the directions:

$$E_{grad}(I_t, I_r) = \frac{\sum_{\theta \in \mathscr{T}} \frac{1}{n} \|(I_t - I_r) \star S_\theta\|_2^2}{|\mathscr{T}|}, \tag{5.2}$$

where $\mathscr{T} = \{0, 45, 90, 135\}$ is the set of considered orientation angles (in degrees) and n is the number of image pixels.

Ridge pattern orientation. Orientation of the ridge pattern (see Chapter 2) is defined through image moments based on image gradients. Considering $G_x = G_0$ and $G_y = G_{90}$, the covariance matrix of the image is defined using second order central moments (μ'_{20}, μ'_{02}, μ'_{11}) Hu [1962]:

$$\mathrm{Cov}(I(x,y)) = \begin{pmatrix} \mu'_{20} & \mu'_{11} \\ \mu'_{11} & \mu'_{02} \end{pmatrix} = \begin{pmatrix} G_{xx} & G_{xy} \\ G_{xy} & G_{yy} \end{pmatrix}, \tag{5.3}$$

where $G_{xx} = g_{\Sigma_s} \star (G_x \cdot G_x)$, $G_{yy} = g_{\Sigma_s} \star (G_y \cdot G_y)$ and $G_{xy} = g_{\Sigma_s} \star (G_x \cdot G_y)$. In the above, g_Σ represents a Gaussian smoothing kernel with covariance Σ and $\cdot$ denotes the element-wise product.

Eigenvectors of the covariance matrix $\mathrm{Cov}(I(x,y))$ point in the directions of the major and minor intensity of image I. This information is enough to compute the orientation as the angle between the eigenvector corresponding to the largest eigenvalue and the x-axis, which is expressed by formula:

$$\Theta = \frac{1}{2} \tan^{-1}\left(\frac{2G_{xy}}{G_{xx} - G_{yy}}\right). \tag{5.4}$$

Further, the *reliability orientation field* [Khalil, 2011] is calculated in order to strengthen the similarity between the reconstructed image and the associated ground-truth. The ridge orientation image is converted into a continuous vector field as

$$(\Phi_x, \Phi_y) = (\cos(2\Theta), \sin(2\Theta)), \tag{5.5}$$

Subsequently, the resulting vector field is denoised applying a low-pass Gaussian filter

$$(\Phi'_x, \Phi'_y) = (g_{\Sigma_o} \star \Phi_x, g_{\Sigma_o} \star \Phi_y). \tag{5.6}$$

The reliability measure R is then defined by means of minimum inertia I_{min} and maximum inertia I_{max} as follows:

$$I_{min} = \frac{(G_{yy} + G_{xx}) - (G_{xx} - G_{yy})\Phi'_x - G_{xy}\Phi'_y}{2}, \tag{5.7}$$

$$I_{max} = G_{yy} + G_{xx} - I_{min}, \tag{5.8}$$

$$R = 1 - \frac{I_{min}}{I_{max}}. \tag{5.9}$$

The intuition behind is that when the ratio I_{min}/I_{max} approaches 1, there is very little orientation information at that point.

Finally, the orientation energy E_{ori} and reliability energy E_{rel} is defined as the MSE of the orientation and reliability measures computed on the target image I_t and the reconstructed image I_r,

$$E_{ori} = \frac{1}{n}\|\Theta(I_t) - \Theta(I_r)\|_2^2, \tag{5.10}$$

$$E_{rel} = \frac{1}{n}\|R(I_t) - R(I_r)\|_2^2. \tag{5.11}$$

5.2 Fingerprint reconstruction model

Following the principles introduced in the previous section, the model is a fully convolutional autoencoder network designed according to the guidelines presented in [Radford et al., 2015]. The full architecture is depicted in Figure 5.1. In the encoder, each convolutional layer is equipped with REctified Linear Unit (ReLU) [Nair and Hinton, 2010] nonlinearity. Batch normalization is applied to each layer for faster convergence [Ioffe and Szegedy, 2015].

The output of the encoder is directly fed to the decoder, which copies the architecture of the encoder with the following changes. Convolutional layers

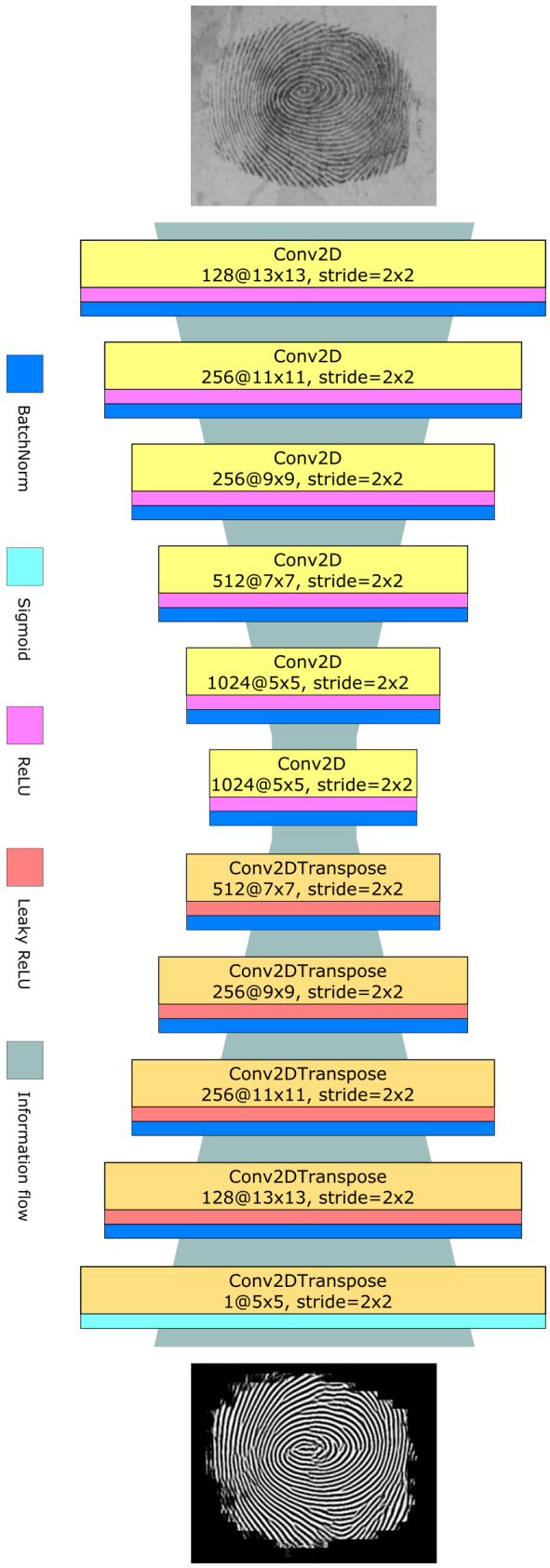


Figure 5.1. The fully convolutional autoencoder architecture employed in this work.

are replaced with de-convolutional layers (i.e. fractionally strided convolutional layers [Radford et al., 2015]) and ReLU nonlinearities with Leaky-ReLU [Maas et al., 2013]. The original grayscale image is reproduced at the output by means of an additional convolutional layer equipped with a sigmoid activation which is attached to the end of the decoder.

Training objective. The chosen objective function is based on the analysis of gradient-based fingerprint similarity measures described in Section 5.1. It is composed as a weighted average of three different losses, in particular:

$$E = E_{grad} + \lambda(E_{ori} + E_{rel}), \tag{5.12}$$

where E_{grad} enforces the model to reconstruct the fingerprint ridges at the same positions, while E_{ori} and E_{rel} regularize the model in order to produce consistent orientation of the fingerprint ridge pattern with respect to the original input. Parameter $\lambda = 0.1$ weights the regularization losses.

Training procedure. Training of autoencoder models is subject to the availability of a sufficient dataset. This approach overcomes this shortcoming by training purely on synthetic data. The generated fingerprints undergo transformations such as rotation, translation, directional blur, morphological dilation and finally blending with several different backgrounds in order to simulate latent fingerprint impressions.

The ground truth (target) images are further binarized using approach based on Bartunek et al. [2006]. The presented autoencoder model, therefore, learns not only to reconstruct the original image (target) but also directly produces its binarized version, which is a more favorable representation for feature extraction and matching modules.

5.3 Experiments

We provide an extensive evaluation of the proposed approach on two different standard benchmarks, which are presented first. The section follows with a description of the experimental setup. Subsequently, we compare our performance to the baseline performances reported by the authors of the datasets. Finally, NBIS NIST [2007] utility is used to evaluate the quality of the samples reconstructed by our method.

5.3.1 Datasets

The proposed autoencoder model has been evaluated on publicly available latent fingerprint datasets which are shortly described next.

IIIT-Delhi Latent Fingerprint [Sankaran et al., 2011] dataset consists of latent fingerprint impressions of all ten fingers from 15 different subjects lifted from two different backgrounds. Multiple instances are captured for each fingerprint to allow latent-to-latent fingerprint comparisons. In total, there are 1046 latent impressions

IIIT-Delhi Multi-sensor Optical and Latent Fingerprint (MOLF) [Sankaran et al., 2015] contains 100 subjects and a total of 19200 fingerprints acquired using three different sensors (Lumidigm Venus IP65 Shell, Secugen Hamster-IV, Cross-Match L-Scan Patrol) and two versions of lifting latent impressions (single latent fingerprints, simultaneous latent fingerprints) lifted using black powder dusting process.

5.3.2 Experimental setup

First, performance is evaluated by applying standard fingerprint recognition algorithms on original fingerprints and those enhanced by the presented method. Two different feature extraction method are employed: one proposed by Abraham et al. [2011] (further abbreviated as ABR) and MINDTCT from NBIS [NIST, 2007]. Extracted features are subsequently compared using two different matching methods, namely BOZORTH3 [NIST, 2007] (abbreviated to BOZ in the following text) and Minutiae Cylinder Code (MCC) [Cappelli et al., 2010, 2011; Ferrara et al., 2012, 2014].

For all the remaining evaluations, the best performing method on the latent fingerprint matching, which is a combination of ABR + MCC, is chosen.

All the results are presented in terms of TopX measure and Cumulative Match Characteristic (CMC) curves.

5.3.3 Latent-to-Latent matching

The evaluation was carried out using the protocol described in Sankaran et al. [2011]. The whole IIIT-Delhi latent fingerprint dataset contains 1046 samples of all ten fingerprints collected from 15 different subjects. The split strategy of Sankaran et al. [2011] is adopted, randomly choosing 395 images as gallery and 520 as probes, making sure that each class contains at least one gallery sample. 131 images were left out since Sankaran et al. [2011] used them for

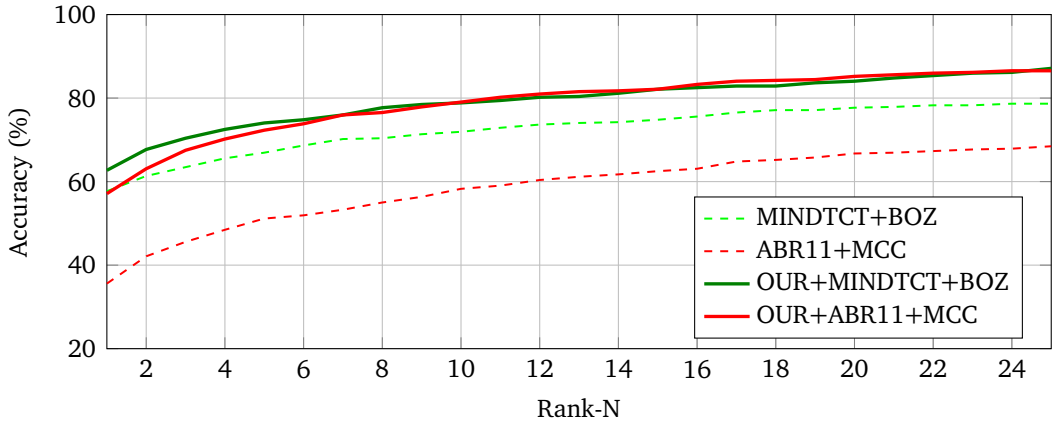


Figure 5.2. Latent-to-Latent matching on IIIT-Delhi latent database. CMC curve showing the performance improvement of our enhancement approach (solid lines) against matching raw latent fingerprints (dashed lines).

training. Ten-fold cross-validation is performed in order to ensure the random splitting does not influence the reported results.

Table 5.1 shows Rank-1 and Rank-10 accuracy for both recognition methods with and without the proposed enhancement clearly demonstrating the boost in performance while enhancing input samples using the proposed approach. The novel approach outperforms all the fingerprint matching methods evaluated by Sankaran et al. [2011]. It is worth pointing out that as opposed to Sankaran et al. [2011], the proposed method does not need to be trained on a subset of the data.

Careful examination of the CMC curves in Figure 5.2 shows that the proposed approach boosts the performance of combination ABR + MCC much more than that of MINDTCT + BOZ. This can be attributed to the fact that the energy minimized during the training of the model is composed of very similar operations to the ridge binarization part of the ABR feature extraction algorithm.

5.3.4 Latent-to-Sensor matching

MOLF dataset was employed to evaluate the approach on a much more challenging task of latent-to-sensor fingerprint matching. The dataset contains all ten fingerprints of 100 different subjects. The samples are of very different quality, including some very poor samples where no ridge structure is visible. Fingerprints of each subject are capture with several commercial fingerprint scanners (Lumidigm, Secugen and Crossmatch). In addition, each participant pro-

Method		Accuracy	
Enhancement	Extract + Match	Rank-1	Rank-10
Raw	ABR + MCC	35.58%	58.27%
Raw	MINDTCT + BOZ	57.69%	71.92%
Our	ABR + MCC	57.12%	79.04%
Our	MINDTCT + BOZ	62.69%	78.85%

Table 5.1. Latent-to-Latent matching on IIIT-Delhi latent database. Matching was performed on images obtained with the proposed fingerprint enhancement (Our) and with no enhancement (Raw).

vides a set of latent fingerprint impressions. This allows performing matching of latent fingerprints to the ones acquired by a sensor. Following the test protocol of Sankaran et al. [2015], first and second instance fingerprints for each user from a sensor scanned database are considered as the gallery. The whole latent fingerprint dataset consisting of 4400 samples is considered the probe set.

We compare to the evaluation of Sankaran et al. [2015], where minutiae were extracted automatically and matched using one of the standard algorithms. Sankaran et al. [2015] evaluated the performance of publicly available MINDTCT + BOZ and commercial Verifinger [NeuroTechnology, 1998] fingerprint matching methods, reporting very poor performance for both. Results for the proposed approach are presented using the combination ABR feature extractor and MCC matching algorithm (ABR + MCC) and summarized in Table 5.2 and Figure 5.3. It shows that ABR + MCC performs very poorly on the original samples while enhancing the samples using the novel approach results in a significant performance boost.

5.3.5 Cross-dataset Latent-to-Sensor matching

To evaluate the influence of different sensors on the quality while matching latent fingerprints to the sensor scans, an additional comparison of matching latent fingerprints to Secugen and Crossmatch sensors using the ABR + MCC matching method is presented. Quality of the acquired fingerprint samples is of-course bound to the type of sensor that is used. Therefore, the performance of matching latent fingerprints to samples from different sensors can slightly differ. Table 5.3 and Figure 5.4 however both indicate that enhancing fingerprints using the proposed method results in significant performance boost independently on the sensor at hand.

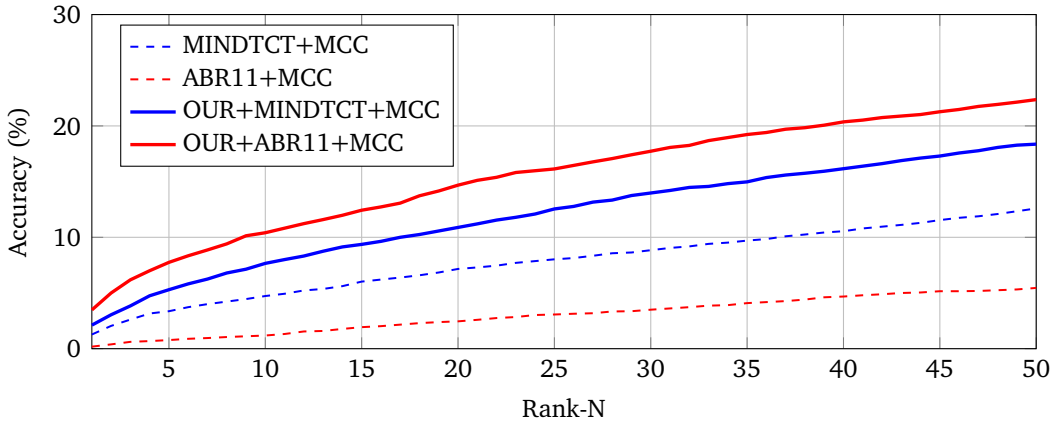


Figure 5.3. Latent-to-Sensor matching on IIIT-Delhi MOLF database. CMC curves showing comparison of our enhancement approach (solid lines) against matching raw latent fingerprints (dashed lines) to the Lumidigm database.

Method		Accuracy	
Enhancement	Extract + Match	Rank-25	Rank-50
Raw	MINDTCT + MCC	8.02%	12.59%
Raw	ABR + MCC	3.07%	5.45%
Our	MINDTCT + MCC	12.55%	18.36%
Our	ABR + MCC	16.14%	22.36%
Sankaran et al. [2015]	MINDTCT + BOZ	N/A	6.06%
Sankaran et al. [2015]	VeriFinger	N/A	6.80%

Table 5.2. Latent-to-Sensor matching on IIIT-Delhi MOLF database. Shown as the performance of matching Lumidigm images to latent fingerprints with enhancement (Our) and with no enhancement (Raw). Two more methods listed in Sankaran et al. [2015] are compared.

5.3.6 Fingerprint quality

To qualitatively evaluate the reconstruction of latent fingerprints done by the proposed model, quality assessment using the NIST Fingerprint Image Quality (NFIQ) utility (part of NBIS [NIST, 2007]) was carried out. It assigns a fingerprint image a numerical score from 1 (great quality) to 5 (very poor quality). The distribution of score values shown in Figure 5.5 supports the claimed benefits of using the proposed enhancement method, being compared to raw fingerprints

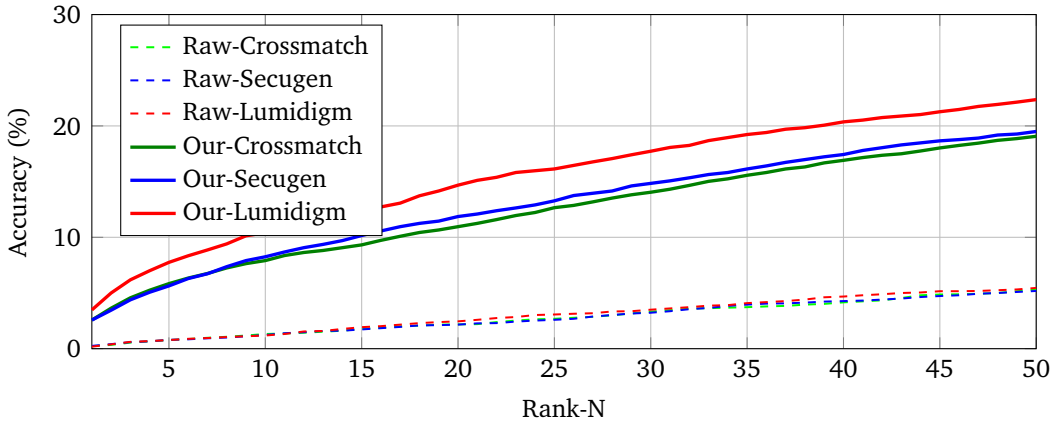


Figure 5.4. Cross-dataset Latent-to-Sensor matching on IIIT-Delhi MOLF database. CMC curve showing the performance improvement of our enhancement approach (solid lines) against matching raw latent fingerprints (dashed lines) to the Lumidigm, Secugen, and Crossmatch scanner datasets.

Method		Accuracy	
Enhancement	Gallery dataset	Rank-25	Rank-50
Raw	Lumidigm	3.07%	5.45%
Raw	Secugen	2.59%	5.18%
Raw	Crossmatch	2.68%	5.32%
Our	Lumidigm	16.14%	22.36%
Our	Secugen	13.27%	19.50%
Our	Crossmatch	12.66%	19.07%

Table 5.3. Cross-dataset Latent-to-Sensor matching on IIIT-Delhi MOLF database. Performance of matching Lumidigm, Secugen, and Crossmatch sensor images to latent fingerprints enhanced by our model (Our) and with no enhancement (Raw). In all cases, combination ABR + MCC was used for feature extraction and matching.

and an alternative enhancement method based on histogram equalization.

Successful and unsuccessful results of latent fingerprint reconstruction for both real and synthetic data are shown in Figure 5.6 and Figure 5.7 respectively. The proposed method enhances the ridge information well in cases where it is still somehow present. For very difficult cases, where no discernible ridges are visible, the proposed approach expectedly fails. This is however not wrong, since

producing ridge pattern where there is none visible could result in the generation of false minutiae and negatively affect the matching results.

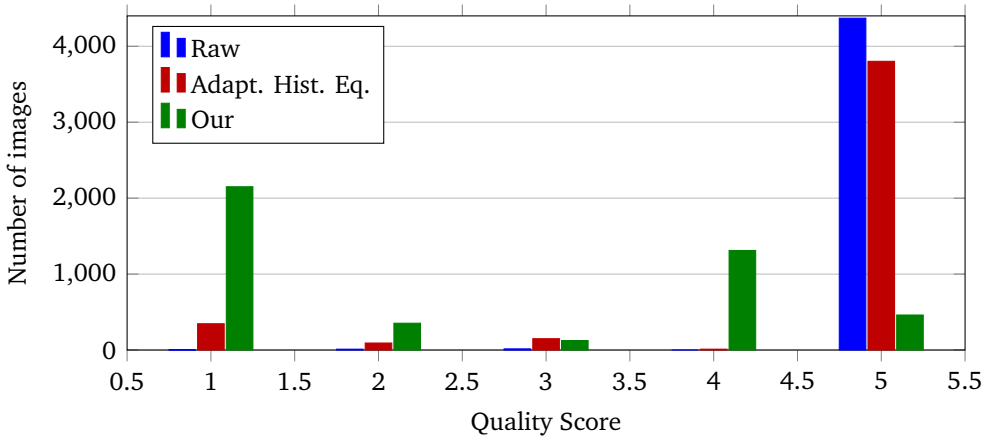


Figure 5.5. Image quality assessment using NFIQ utility. We compare the raw data with data improved using adaptive histogram equalization and data improved using our enhancement. The quality assessment score ranges from 1-5, where the lower the better.

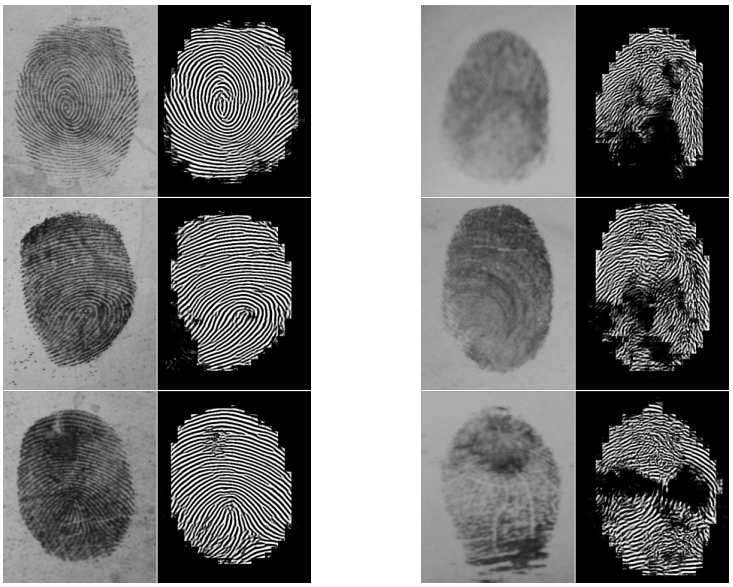
5.4 Discussion

In this chapter, a deep-learning-based technique for enhancement of damaged and incomplete latent fingerprints has been presented. It is based on convolutional autoencoder models popular in deep learning nowadays. The objective function is based on gradient analysis of the fingerprint ridge structure in order to make the autoencoder learn accurate reconstructions of fingerprints.

The proposed approach overcomes the burden of large dataset requirement for training of the model by employing a synthetic fingerprint generation software in order to generate sufficient, theoretically unlimited, amount of training samples.

The trained model has been evaluated on standard benchmarks for latent-to-latent fingerprint matching as well as latent-to-sensor fingerprint matching. In both cases, the novel approach outperforms the current state-of-the-art on these datasets by a significant margin.

Nevertheless, the model is not always successful and some of the failure cases have been evaluated qualitatively, showing the potential risk of generating false minutiae. This issue can influence the performance of fingerprint recognition



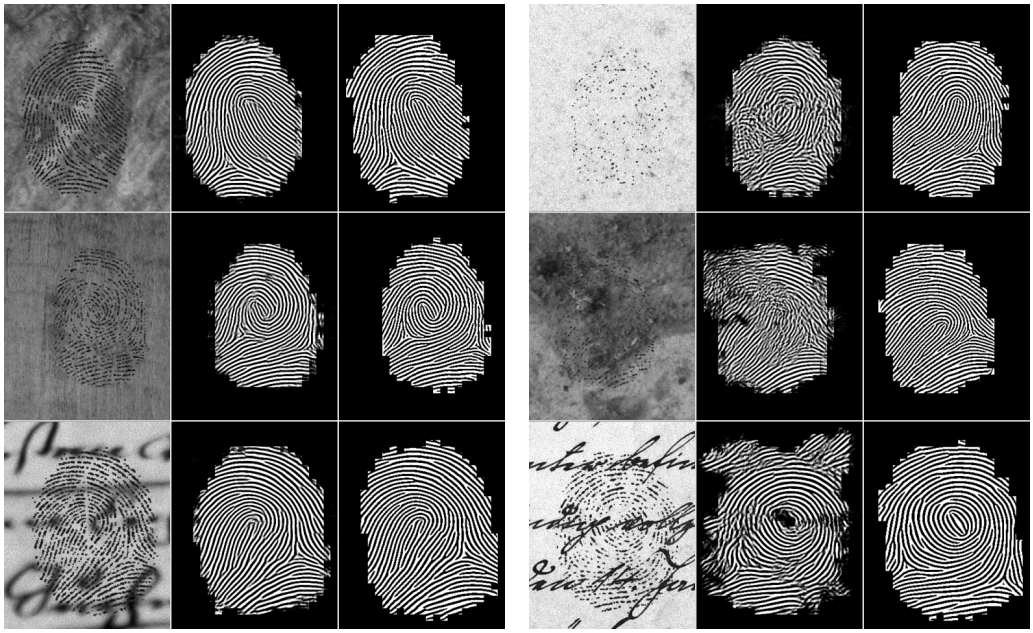
(a) Successful reconstructions

(b) Fail cases

Figure 5.6. Examples of the fingerprint reconstructions on real latent fingerprints. Each pair is (input, reconstruction) (a) successfully reconstructed samples (b) cases where the model fails to reconstruct the fingerprint well.

greatly and should, therefore, be addressed as future work. Meanwhile, others have followed this work [Schuch et al., 2017; Li et al., 2018; Dabouei et al., 2018; Joshi et al., 2019]. In particular, Schuch et al. [2017] have proposed a conditional generative autoencoder-based model, which should address the aforementioned issue, setting new state-of-the-art on the benchmark datasets.

As our approach does not directly extract minutiae or perform matching but instead reconstructs the correct ridge pattern from a poor-quality fingerprint, it has potential for broad alternative applications such as the reconstruction of fingerprints affected by diseases or enhancement of fingerprints from camera-based contactless fingerprint sensors.



(a) Successful reconstructions (b) Fail cases

Figure 5.7. Examples of the fingerprint reconstructions on synthetic data. Each triplet is from left to right (input, reconstruction, target) (a) successfully reconstructed samples (b) cases where the model fails to reconstruct the fingerprint well.

Chapter 6

Improving robustness of neural networks

Previous chapters in this thesis covered novel methods for hand biometrics based on image analysis using deep learning. Their performance is, however, heavily influenced by the reliability of the trained model. Unfortunately, such systems have been proven vulnerable to adversarial attacks [Szegedy et al., 2014], making them easy to fool. This raises some serious concerns for the application of deep learning in biometrics. It is, therefore, necessary to seek innovative design methodologies to ensure deep neural networks be robust to such attacks.

This chapter is based on Svoboda et al. [2019]. It presents PeerNets, a novel family of convolutional networks alternating classical Euclidean convolutions with graph convolutions to harness information from a graph of peer samples. This introduces the effect of non-local propagation within the model, where latent features become conditioned on the global structure induced by the graph. Extensive evaluations show that PeerNets are, indeed, more robust to many types of adversarial attacks and, therefore, offer a way to mitigate the negative effect of adversarial noise.

6.1 Peer Regularized Networks

A new deep neural network architecture that takes advantage of the data space structure is proposed. The centerpiece of the model is the *Peer Regularization* (PR) layer, designed as follows. Let $\mathbf{X}^1, \ldots, \mathbf{X}^N$ be $n \times d$ matrices representing the feature maps of N images, which are referred to as *peers* (here n denotes the number of pixels and d is the dimension of the feature in each pixel). Given a pixel of image i, we consider its K nearest neighbor graph in the space of d-dimensional feature maps of all pixels of all the peer images, where the neighbors are computed using e.g. the cosine distance. The kth nearest neighbor of the pth pixel $\mathbf{x}_p^i$ taken

from image i is the q_kth pixel $\mathbf{x}_{q_k}^{j_k}$ taken from peer image j_k, with $k = 1, \ldots, K$ and $j_k \in \{1, \ldots, N\}$, $q_k \in \{1, \ldots, n\}$. A variant of graph attention network (GAT) by Veličković et al. [2018] is applied to the nearest-neighbor graph constructed this way,

$$\tilde{\mathbf{x}}_p^i = \sum_{k=1}^{K} \alpha_{ij_kpq_k} \mathbf{x}_{q_k}^{j_k}, \qquad \alpha_{ij_kpq_k} = \frac{\text{LeakyReLU}(\exp(a(\mathbf{x}_p^i, \mathbf{x}_{q_k}^{j_k})))}{\sum_{k'=1}^{K} \text{LeakyReLU}(\exp(a(\mathbf{x}_p^i, \mathbf{x}_{q_{k'}}^{j_{k'}})))} \tag{6.1}$$

where $a()$ denotes a fully connected layer mapping from $2d$-dimensional input to scalar output, and $\alpha_{ij_kpq_k}$ are attention scores determining the importance of contribution of the q_kth pixel of image j to the output pth pixel $\tilde{\mathbf{x}}_p^i$ of image i. This way, the output feature map $\tilde{\mathbf{X}}^i$ is a pixel-wise weighted aggregate of the peers. Such Peer Regularization is reminiscent of non-local means denoising of Buades et al. [2005], which operates on a single image and computes the attention between different pixels based on their neighborhood similarity. Matching pixels in different images has some analogies with the PatchMatch algorithm [Barnes et al., 2009]. Our Peer Regularization, instead, takes the pixels from multiple images rather than from the same image, and the attention weights are learnable.

In principle, one would use a graph constructed by performing nearest neighbor search across all the available training samples. This is unfortunately not feasible in practice due to memory and computation limitations. Instead, an approximation using the Monte Carlo method can be computed as follows.

Let $\mathbf{X}^1, \ldots, \mathbf{X}^{N'}$ denote the images of the training set. M smaller batches of $N \ll N'$ peers are selected randomly with uniform distribution, batch m containing images $\{l_{m1}, \ldots, l_{mN}\} \subset \{1, \ldots, N'\}$. The nearest-neighbor graph is constructed separately for each batch m, such that the kth nearest neighbor of pixel p in image i is pixel q_{mk} in image j_{mk}, where $m = 1, \ldots, M$, $j_{mk} \in \{l_{m1}, \ldots, l_{mN}\}$, and $q_{mk} \in \{1, \ldots, n\}$. The output of the filter is approximated by empirical expectation on the M batches, which is estimated as follows:

$$\tilde{\mathbf{x}}_p^i = \frac{1}{M} \sum_{m=1}^{M} \sum_{k=1}^{K} \alpha_{ij_{mk}pq_{mk}} \mathbf{x}_{q_{mk}}^{j_{mk}}. \tag{6.2}$$

As it is desired to limit the computation overhead, $M = 1$ is used during training, whereas larger M is used during inference (see Section 6.3 for details).

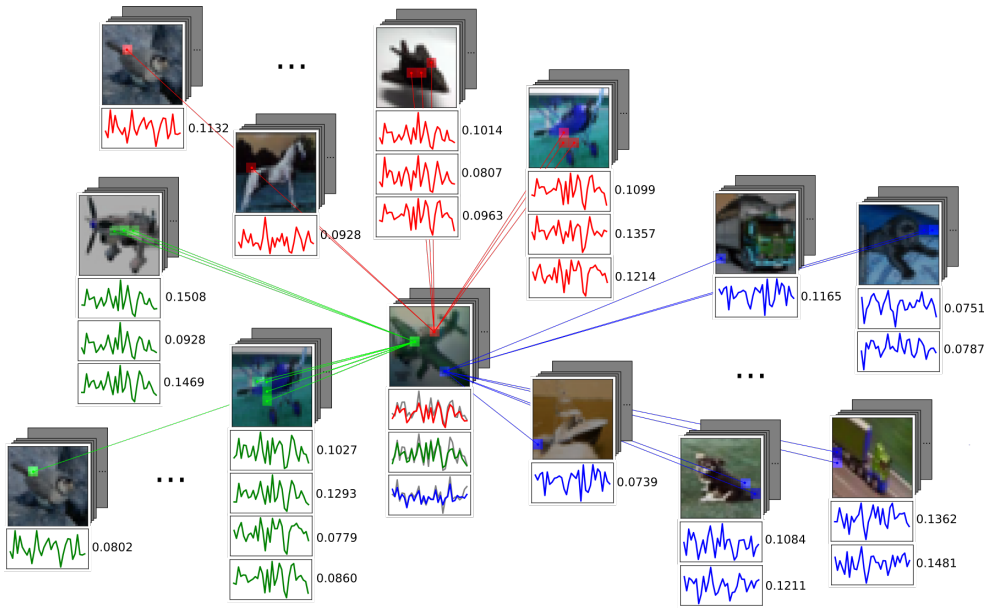


Figure 6.1. Our Peer Regularization illustrated on three pixels (red, green, blue) of a CIFAR image (center). For each pixel, five nearest neighbors are found in peer images. Plots represent the feature maps in the respective pixels; numbers represent the attention scores.

6.2 Graph construction

There are different approaches to graph construction during the training and testing phase, which are shortly described below.

During training, it is possible to construct the graph using all the images of the current batch as peers. This is done by computing all-pairs distances followed by the selection of the K-nearest neighbors. To mitigate the influence of the small batch sizes during the training of PeerNets, a high level of stochasticity is introduced inside the Peer Regularization layer by using dropout of 0.2 on the all-pairs distances while performing K-nearest neighbors and 0.5 on the attention weights right before the *softmax* nonlinearity.

For testing, N fixed peer images are randomly selected from the training set and their distances with respect to all the testing samples are computed. Feeding a batch of test samples, each of them can be adjacent only to the N samples in the fixed graph, and not to other test samples. Because of the approximation in the graph construction described in the previous section, one needs to ensure that the random pre-selection of the graph nodes does not influence the performance. This is done using Monte Carlo sampling over M forward passes with different

uniformly sampled graphs with N nodes and averaging the predictions over the M runs at the end.

6.3 Experiments

The robustness of proposed Peer Regularization against adversarial perturbations is evaluated on standard benchmarks in computer vision, which are described first. The section continues with presenting the evaluated model architectures and subsequently follows evaluation on different adversarial attacks.

6.3.1 Datasets

Robustness of PeerNets is evaluated on the standard datasets MNIST [LeCun, 1998a], CIFAR-10 and CIFAR-100 [Krizhevsky, 2009] which are shortly described next.

MNIST is a dataset of handwritten digits with a training set of 60000 examples and test set of 10000 examples. Each example belongs to one of the 10 classes (numbers 0 to 9). The digits have been further size-normalized and centered in a fixed-image size of 28×28 pixels.

CIFAR-10 and CIFAR-100 are both labelled subsets of a big dataset of tiny natural color images introduced by Torralba et al. [2008]. The size of each image is 32×32 pixels. In particular, *CIFAR-10* consists of 50000 training images and other 10000 images for testing, and has 10 classes, meaning 6000 images per class. *CIFAR-100* is similar, it has 50000 images in the training set and test set of 10000 images, but has 100 classes in total, meaning 600 images per class. The 100 classes are additionally grouped into 20 superclasses, which provides two different labellings: on a coarse and fine level.

6.3.2 Experimental setup

Evaluation has been done using several common architectures such as LeNet [LeCun, 1998a] or ResNet [He et al., 2015]. This is mainly to be able to compare to the previous state-of-the-art in the field of adversarial attacks. The modifications of the aforementioned architectures adding the novel PR layers are referred to as PR-LeNet and PR-ResNet (see Figure 6.2) and are described next.

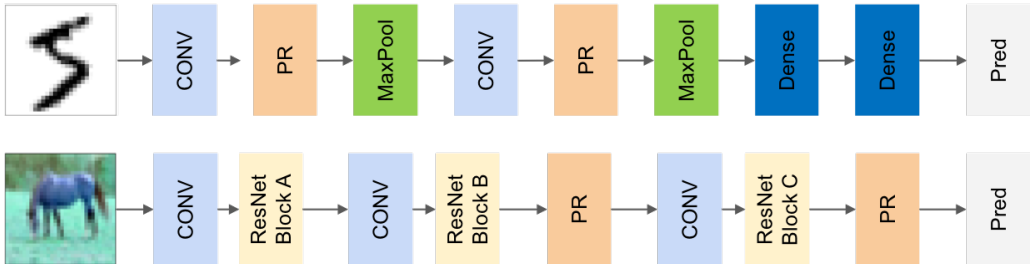


Figure 6.2. Modification of LeNet-5 and ResNet with the novel Peer Regularization Layer into PR-LeNet and PR-ResNet.

For *MNIST*, modified LeNet-5 architecture is introduced. Two PR layers are added after each convolutional layer and before max-pooling (Figure 6.2 top). Performance on *CIFAR-10* is evaluated using modified ResNet-32 model. With $A = 16$, $B = 32$ and $C = 64$ as depicted in Figure 6.2 (bottom). In particular, two PR layers are added at the last change of dimensionality and before the classifier. Each ResNet block is a sequence of Conv + BatchNorm + ReLU + Conv + BatchNorm + ReLU layers. Finally, ResNet-110 is used for *CIFAR-100*, with $A = 16$, $B = 32$ and $C = 64$, modified in the same way as for CIFAR-10. Each block is of size 18.

Timing. During training on CIFAR-10 with an unoptimized PR-ResNet-32 implementation, batch of 64 samples is processed in $0.15s$ compared to traditional ResNet-32 baseline which takes $0.07s$. At inference time a batch of 100 samples with a graph size of 100 is processed in approximately $1.5s$ whereas the baseline takes only $0.4s$. Nevertheless, much can be done in order to speed-up the PR layer if needed as future work.

Gradient obfuscation. Athalye et al. [2018] has shown that many defenses against adversarial attacks indeed do not work. It has been demonstrated that the majority of the methods, in fact, obfuscate the gradients, which in other words simply means that the gradients of the network cannot be propagated through the defense mechanism and the attacker, therefore, does not know how to foster the perturbation. Such methods block the gradient-based attacks but are not of any use in case of e.g. black-box attacks. It also raises the concern that the proposed defense mechanisms do not actually work, and better performance is achieved only due to gradient obfuscation. Athalye et al. [2018] proposed several ways to suggest whether a method is obfuscating gradients or not. Further experiments show that PeerNets do not exhibit the behavior which typically accompanies this

problem (e.g. the PR-models can be always fooled with an unbounded attack, iterative attacks are more efficient than the single-step ones, etc.).

6.3.3 Sample specific attacks

PeerNets have been evaluated on different types of sample-specific non-targeted attacks, defined as follows. Having a testing sample with label y, it aims to generate an adversarial sample particularly for this testing sample, which will have label $\hat{y} \neq y$, while satisfying $||\mathbf{x} - \hat{\mathbf{x}}||_\infty \leq \epsilon$, where ϵ is some small value to produce a perturbation $\mathbf{v}$ that is ideally imperceivable. For reproducibility, an implementation of the following methods provided by Foolbox by Rauber et al. [2017] has been used. In this experiment we have tested against Gradient Descent, Fast Gradient Sign Method (FGSM) and Projected Gradient Descent (PGD) attacks that are described in Section 3.6.2.

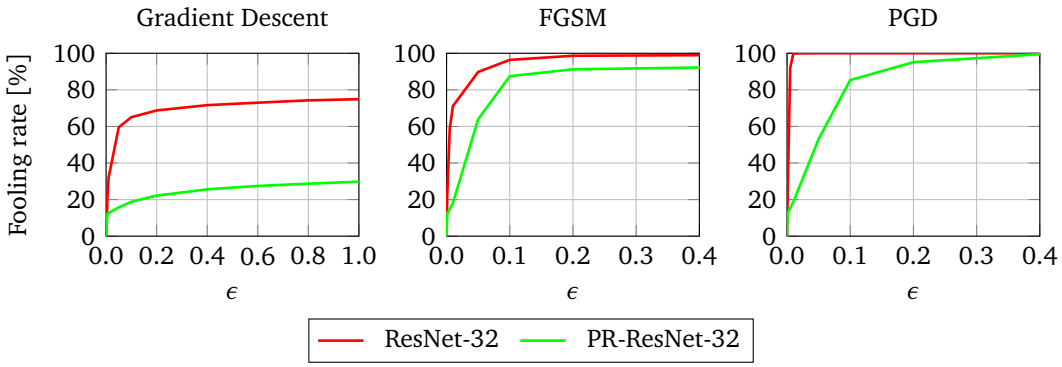
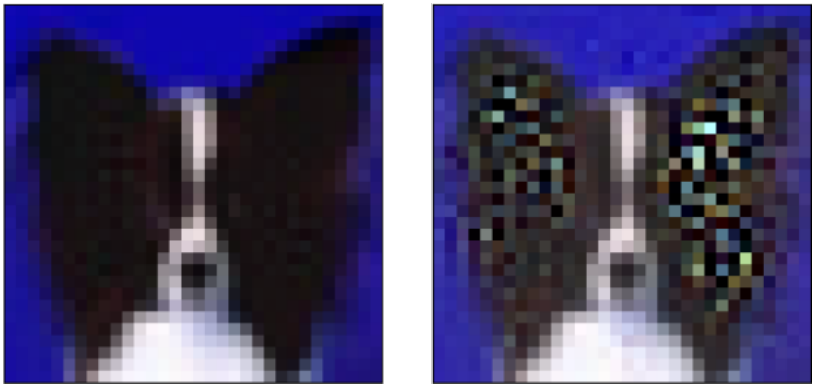


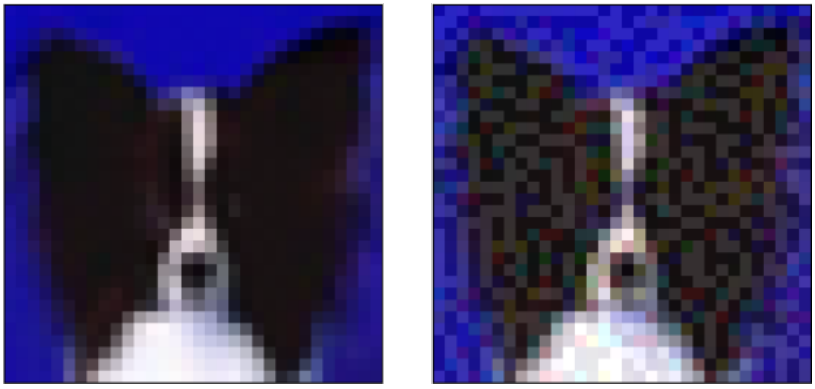
Figure 6.3. Comparison of PeerNets and classical CNNs on the CIFAR-10 dataset using various gradient-based sample-specific attacks.

Figure 6.4 clearly demonstrates that the sample-specific gradient-based attacks have to generate much more significant perturbation in order to fool the PeerNets architectures. The visual interpretation is shown in Figure 6.4, suggesting that it is hard to fool PeerNets without generating perturbation observable by a human eye.

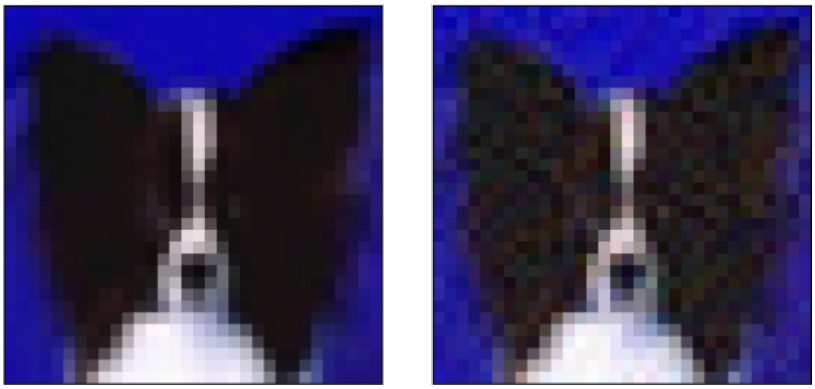
Presented results further confirm that PGD, the strongest attack that was tested, can generate a perturbation for all the samples in the test set even while using PeerNets. However, the generated perturbations are generally much stronger. For classical ResNet-32, it is enough to set $\epsilon \geq 0.013$, while PR-ResNet-32 requires $\epsilon \geq 0.651$ in order to be fooled, which obviously generates much stronger perturbation. The success of PGD is due to its iterative nature and it also supports



(a) Gradient Descent



(b) FGSM



(c) PGD

Figure 6.4. Comparison of perturbation strength sufficient to fool the network, as generated by different sample-specific gradient-based attacks. ResNet-32 is always shown on the left, whereas our PR-ResNet-32 on the right.

the fact that our method does not suffer from previously mentioned gradient obfuscation, as a viable perturbation can be found based on the iterative processing of network gradients for any test sample.

6.3.4 Universal adversarial perturbations

We evaluate PeerNets on the universal adversarial perturbation as well, using the publicly available implementation of the adversarial attack from the *DeepFool* Moosavi-Dezfooli et al. [2017] toolbox. The strength of the perturbation was bound by $||\mathbf{v}||_2 \leq \rho \mathbb{E}||\mathbf{x}||_2$, where the expectation is taken over the set of training images and the parameter ρ controls the strength.

The evaluation was carried out on MNIST dataset [LeCun, 1998a] and compared to LeNet-5 as a baseline. The same evaluations were performed also on CIFAR-10 and CIFAR-100 datasets, using ResNet-32 and ResNet-110 as baselines, respectively. On CIFAR-10, additional comparison to other baselines is provided, namely BRELU with Gaussian Additive Noise by Zantedeschi et al. [2017], and MagNet of Meng and Chen [2017]. It has been shown that PeerNets perform significantly better than MagNet in particular, and better than BRELU for strong noise, as shown in Figure 6.5.

Considering MagNet, in particular, the key advantage of PeerNets is that it can be fully trained end-to-end together with regularization layers instead of as a separate independent module, which should allow learning more discriminative and robust features. Moreover, training a separate module is often prone to the gradient obfuscation problem mentioned before.

It should be noted that a minor loss in accuracy at $\rho = 0$ is observed while using PeerNets. This is a consequence of the regularization due to the averaging in feature space and could be possibly mitigated by increasing the model capacity. This is demonstrated by an experiment with doubled number of maps in PR layers (marked as v2 in Figure 6.5), which indeed shows notable improvement while preserving its original robustness, as opposed to the ResNet baselines.

Universal perturbations generated for both classical CNNs and our PR-Nets are depicted in Figure 6.6. It can be observed that perturbations generated for PR-Nets have more localized structure. This suggests that the attack might be trying to fool the KNN mechanism of the PR layers, which results in strong noise in background areas rather than in the central parts usually containing the object.

6.3.5 Black-box attacks

To show the generality of PeerNets against various classes of adversarial attacks, an evaluation on some of the strong black-box attacks has been done as well. A small subset of 100 test samples has been randomly sampled from the test set. Both untargeted single-step and iterative black-box attacks computed using finite difference method have been used, compared to CW-loss based white-box attack.

Achieved results in Table 6.1 demonstrate that PeerNets are efficient on black-box attacks as well. Besides, it verifies that this novel approach does not cause gradient masking. Athalye et al. [2018] explains that iterative attacks should be stronger than single step and that black-box attacks should be a strict subset of white-box attacks, which holds in all the presented results.

Table 6.1. Evaluation of PeerNets on finite-difference based query black-box attacks using CIFAR-10 dataset.

Method	Attack type	ϵ	Fooling rate [%]	
			White-box	Black-box
ResNet-32	single step	8.0	82.00	82.00
PR-ResNet	single step	8.0	34.00	14.00
ResNet-32	iterative	0.5	41.00	41.00
ResNet-32	iterative	8.0	100.00	100.00
PR-ResNet	iterative	8.0	45.00	28.00
PR-ResNet	iterative	12.0	63.00	37.00

6.3.6 Adversarial training

It has been also studied what is the effect of adversarial training [Madry et al., 2017] compared to PeerNets. The baseline model was the same ResNet-32 that appeared in the previous paragraphs. Two different configurations were used producing two different baseline models, in particular:

1. *ResNet-32 A* - the default hyperparameters provided by the repository;
2. *ResNet-32 B* - the same hyperparameters as in our paper.

PeerNets model was trained traditionally using SGD with momentum without adversarial training.

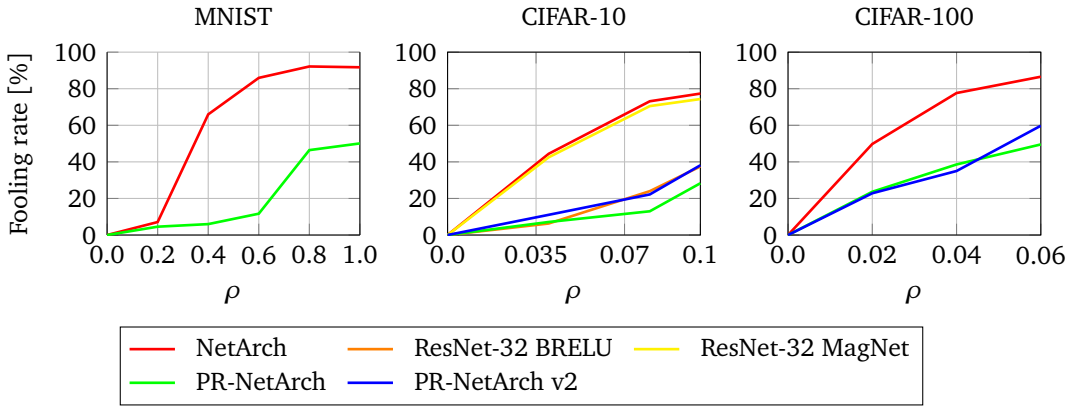


Figure 6.5. Fooling rate on various datasets for different levels ρ of universal adversarial noise. The NetArch is Lenet-5, ResNet-32 and ResNet-110 for MNIST, CIFAR-10 and CIFAR-100 datasets respectively.

Results presented in the Table 6.2 show the superiority of PeerNets compared to adversarial training. PR-ResNet outperforms the classical ResNet CNN by a margin of 20%. Moreover, it should be noted that PR-ResNet was trained without having any knowledge about the attack, as opposed to ResNet-32 trained adversarially on the specific attack.

Table 6.2. Comparison of PeerNets to adversarially trained version of ResNet-32 baseline on CIFAR-10 dataset.

Method	Acc. orig. [%]	Acc. pert. [%]
ResNet-32 A	78.86	45.47
ResNet-32 B	75.59	42.53
PR-ResNet	77.44	64.76

6.3.7 Interpretation of Peer Regularization

In order to verify and better understand what Peer Regularization presented in Section 6.1 actually does, one can take the attention scores from the Peer Regularization layer, backpropagate them to the input and compose a new image as a weighted sum of the peer pixels. Results of this experiment depicted in Fig-

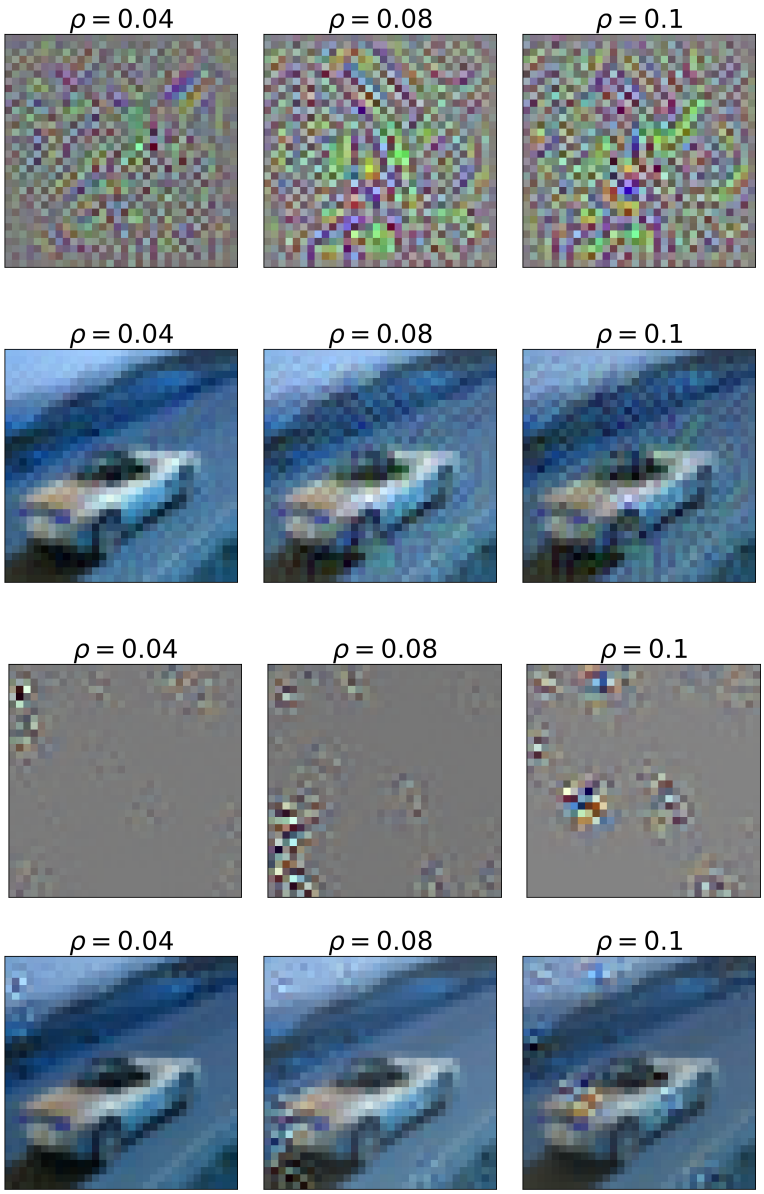


Figure 6.6. Examples of generated universal adversarial perturbations for CIFAR-10 dataset and their applications to a sample image for different values of ρ. Shown are results of ResNet-32 (top couple of rows) and PR-ResNet-32 (bottom couple of rows).

ure 6.7 confirm that the Peer Regularization layer tries to recompose the original feature maps using pixels of peer feature maps, resulting in a new representation

of the same object.

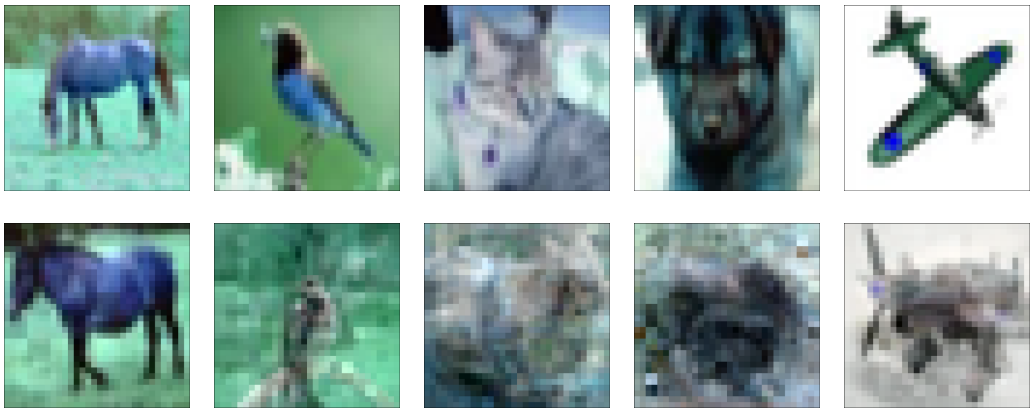


Figure 6.7. Examples of "Franken-images" (second row) constructed by back-propagating the attention scores to the input and using them to compose a new image as a weighted sum of the peer pixels. Original images are shown in the first row.

6.4 Discussion

A novel family of deep networks, so-called PeerNets, alternating Euclidean and Graph convolutions to harness information from peer images was introduced. It has been demonstrated that such models are more robust to adversarial attacks, showing their robustness in a variety of scenarios through extensive experiments for both white-box and black-box attacks. The framework is straightforward to use and Peer Regularization layers can be easily incorporated into any baseline model with minimum effort. Using PeerNets allows achieving much lower fooling rates with negligible loss in performance. Interestingly, the amount of noise required to fool PeerNets is much higher and results in the generation of new images where the noise has a clear structure and is significantly more perceivable to the human eye.

Moreover, the analysis of the effect of Peer Regularization in feature space shows the ability to recompose features from the graph of peers. This shows potential for diverse applications in image processing such as inpainting or neural style transfer [Svoboda et al., 2020a], and also in biometrics(e.g. using the graph of peers for dynamic database search) and possibly many more.

Chapter 7

End-to-end 3D hand geometry biometrics

With respect to methods involving fusion in biometrics, a more complete solution is to have a 3D hand biometric-recognition system based solely on geometric properties of the hand shape. Such systems provide an interesting alternative in places where fingerprints and palmprints cannot be used (e.g. wearing latex gloves, very dirty hands) and face recognition is not an option (e.g. wearing face masks, helmets, goggles or other protective equipment). Solutions have been proposed in the past (see Section 2.3), which, however, neither offer satisfactory performance nor are easy to use, often imposing strict constraints on the acquisition environment. Recent advances in deep learning and 3D sensing suggest that many of the acquisition constraints could be dropped. Attempts to propose a system that would take advantage of the latest trends, however, requires a new dataset, as evaluation data for such approaches are missing at the moment.

This chapter is based on Svoboda et al. [2020b]. It first presents a new dataset for less-constrained 3D hand biometric recognition. The dataset was acquired using a low-cost acquisition device (an off-the-shelf 3D camera) in variable environmental conditions (there were no constraints on where the system was placed during acquisition). Each sample is a short RGB-D recording (a video) of a user performing a predefined gesture, which allows researchers to capture frames of different hand poses or to perform recognition directly from the short video sequences.

Secondly, this chapter introduces a novel method presenting an end-to-end 3D hand-geometry biometric system based on geometric deep learning architecture called Dynamic Graph CNN [Wang et al., 2019]. The novel architecture is compared to two baselines, namely the popular PointNet [Qi et al., 2016] and

the original Dynamic Graph CNN architectures for processing point cloud data. Successful training of such models requires a considerable amount of annotated data, which is typically not present in biometrics. To overcome this limitation, we train the proposed model fully on synthetic data and then transfer the model knowledge to a real dataset while preserving the performance.

7.1 NNHand RGB-D: Hand RGB-D Sequence Dataset

This section introduces a new dataset of human hands collected for the purpose of evaluating hand biometric systems on real world data. At the moment, the first version of the dataset, with suffix *v1* [1], has been released and comprises 79 individuals in total. An extension to at least 200 different identities is planned (version *v2*) for the future.

The dataset is collected using an off-the-shelf range camera in different environments and lighting conditions. Each person contributing to the dataset is asked to repeatedly perform three different series of gestures with their hand in front of the camera, resulting in three RGB-D video sequences collected for each participant. Each subject in the dataset has the following annotations: *User ID*, *Gender* and *Age*. The dataset is mainly targeting three-dimensional shape recognition. However, the presence of RGB-D information also allows attempting 2D shape or palmprint recognition. Attempting palmprint recognition on this dataset might be, however, an extremely challenging task due to the poor lighting conditions of the RGB data in many sequences.

7.1.1 Acquisition device

The vast majority of biometric datasets for hand geometry recognition provide data from expensive devices and constrained acquisition environment. Such solutions are rather hard or even impossible to deploy in industrial applications. This dataset is aimed to be more practical and is supposed to provide data from affordable easy-to-deploy devices and close-to-real environments. Evaluating new methods on such datasets will provide a better indication of the deployment potential of the proposed algorithms in real-world applications.

In particular, this dataset has been collected using *Intel RealSense SR-300* RGB-D camera [Zabatani et al., 2019], which is based on structured light technology (see Section 2.3.1). This technology provides a cost-performance trade-off that is desirable with low-cost range scanners. The sensor is known for satisfying

[1] https://handgeometry.nnaisense.com

reconstruction results while sacrificing some robustness against diverse lighting conditions (e.g. direct sunlight hinders the reconstruction by destroying the infrared pattern projected into the scene). The RealSense SR-300 allows capturing RGB-D video sequences with a depth resolution of 640×480 accompanied by the RGB stream at a resolution up to 1920×1080. With its framerate of 30fps at the maximum resolution, it is well suitable for recording RGB-D video sequences. The minimum distance for a successful data acquisition is approximately 0.25m, which also speaks in its favor while scanning human hands.

7.1.2 Video sequences

Storing video sequences of human hands performing gestures instead of just images allows to evaluate pose-agnostic hand shape recognition as the hand can be viewed in different poses. It also makes the dataset inter-disciplinary, because it gives the possibility to use it for evaluation of hand pose estimation systems.

There are three types of gestures that each participant is asked to perform repeatedly, four times in particular. Between the gestures, the participants are asked to remove their hands from the scene and re-enter. This naturally forces them to re-introduce the hand in the scene each time and provides more diverse and realistic samples.

The recorded video sequences are depicted in Figure 7.1. More examples of the described sequences can be found on the project website[2] together with a more detailed description of the dataset.

7.1.3 Applications of the dataset

Capturing of RGB-D video sequences instead of static images gives the possibility to utilize the dataset for evaluation of new methods on real world data in different areas of research. Moreover, inspired by Afifi [2019], the additional information such as age and gender that is stored for each subject allows investigating the possibility of gender and/or age recognition based on three-dimensional hand shape as well.

Hand shape recognition The main purpose of the dataset is to serve as a new evaluation benchmark for three-dimensional hand shape recognition based on a low-cost three-dimensional sensor. Three different RGB-D video sequences are being captured. The dataset, therefore, allows to experiment with non-rigid

[2] https://handgeometry.nnaisense.com

Figure 7.1. The three sequences (one in each row) recorded for each subject in the dataset. The first sequence is sliding hand vertically into the scene with open palm and removing it again, repeatedly. In the second sequence, rotation of the hand is added when the hand is upright. In the last sequence, user closes and reopens the fist while the hand is upright.

three-dimensional shape recognition as well as attempting to perform recognition viewing the hand from both palm and the dorsal side of the hand.

Palmprint recognition Each video sequence contains the RGB information as well. This means that one can attempt to perform palmprint image recognition on this dataset. However, it should be noted that the illumination conditions are often suboptimal (see Figure 7.2) which makes palmprint recognition on this dataset a rather challenging task.

Figure 7.2. The poor quality of the palmprint images in the NNHand RGB-D dataset.

Age and gender recognition The gender and age together with three-dimensional hand shape have not been recorded previously. They are provided in our dataset as additional details for each subject, which opens the door to the investigation of gender and age estimation of people from the three-dimensional shape of their hand.

Hand pose estimation Even though this dataset does not provide annotations of the hand pose, it can still be used as an additional test benchmark for qualitative evaluation of the hand pose estimation methods, in addition to the recently published hand pose estimation datasets [Tzionas et al., 2016; Kulon et al., 2020].

7.1.4 Examples

Figure 7.3 displays some examples of the data stored in the dataset. In particular, it emphasizes the fact that both palm side and dorsal side of the hand is available, together with different poses of the hand.

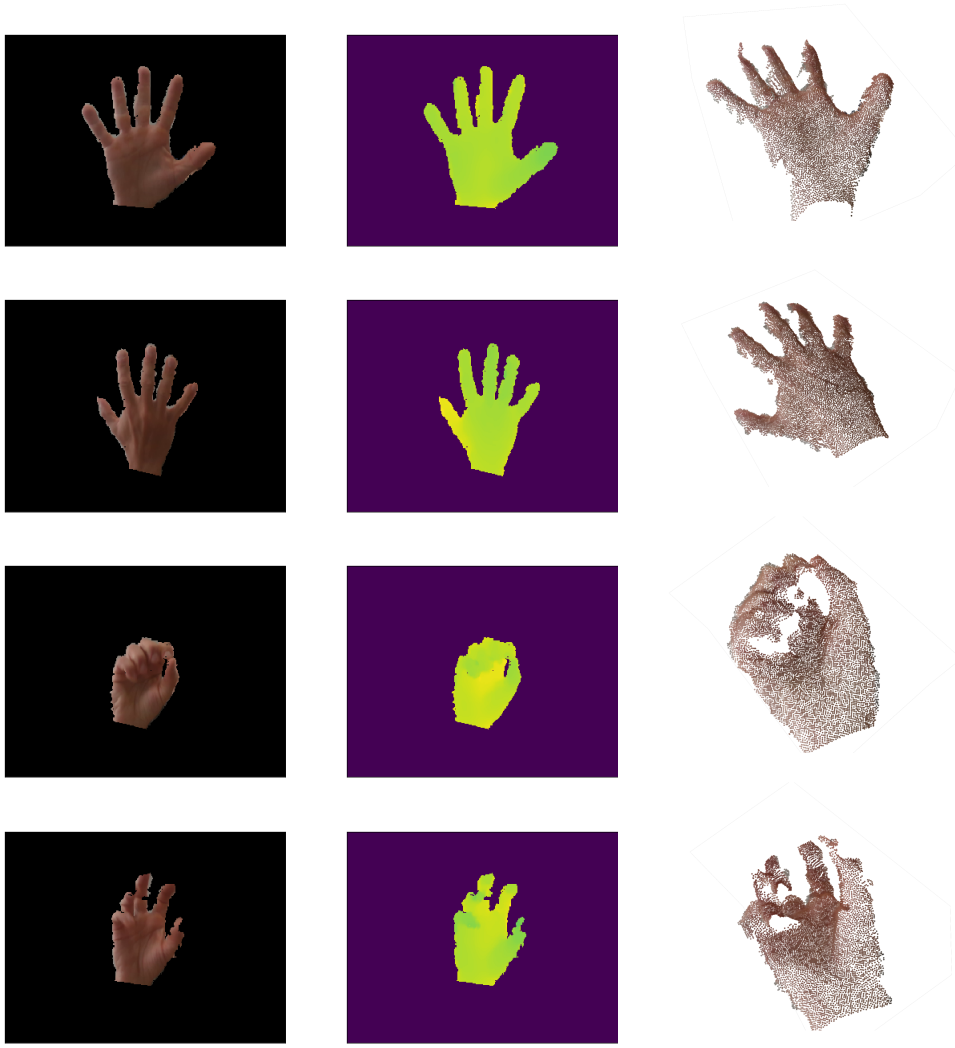


Figure 7.3. Each row is a sample taken from one of the RGB-D sequences in the dataset. Left-to-right are shown the color image, the depth map and the corresponding point cloud.

7.2 Hand Recognition using Geometric Deep Learning

The state-of-the-art methods in three-dimensional biometric hand recognition presented earlier in Section 2.3 show that only classical hand-crafted approaches have been used in the past. The existing solutions are mostly based on the differential geometry of surfaces and analysis of different parts of the hand surface separately, e.g. separate fingers, palm, etc. By the nature of these methods, they are not robust against changes in pose and therefore a good alignment of the hand is required to achieve good performance. This calls for sophisticated pre-processing of the range scans involving extraction of different parts of the hand or estimating its orientation. Undesirably, the quality of the pre-processing has a significant direct impact on recognition performance.

Following the trends of the last decade (see Chapter 3), it seems very appealing to take advantage of deep neural networks to learn good features for three-dimensional hand recognition automatically. However, the data typically acquired by a range scanner are in the form of a point cloud. Applying deep neural networks to point clouds has been a great challenge for years. There have been attempts to solve the problem by using e.g. voxel grids [Maturana and Scherer, 2015], but only some more recent works [Qi et al., 2016, 2017; Shen et al., 2017; Wang et al., 2019; Deng et al., 2018] have really shown how to perform convolution-like operations on clouds of points as noted in Chapter 3. Some of the most notable approaches are PointNet [Qi et al., 2016, 2017] and Dynamic Graph CNN by Wang et al. [2019], which have defined new state-of-the-art performance on many benchmarks ranging from object classification to semantic scene segmentation.

It is very appealing to adapt such methods also in biometrics when dealing with point clouds. However, a big challenge of using deep learning in biometric applications remains the lack of training data, which is discussed next.

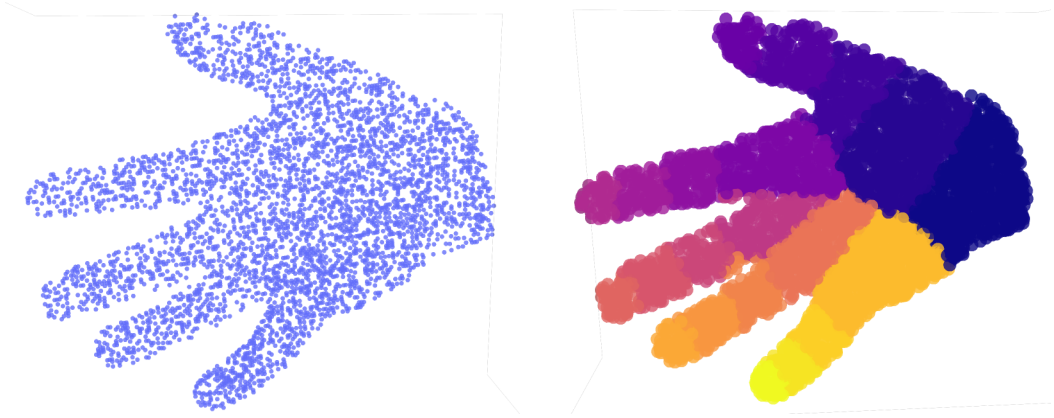
7.2.1 Challenge of training samples

Lack of training samples for deep neural network-based systems is a typical problem in biometrics. The data is often personal and therefore complicated to collect, even more so to distribute. In particular, the datasets available for three-dimensional hand recognition are clearly not of a sufficient size that would allow reliable training of a deep neural network.

The new dataset from Section 7.1 does not come to rescue either. With its currently limited size of 79 subjects, it might not span the space of possible hand shapes well. Moreover, the dataset is released strictly for academic use, in order

to demonstrate the potential of newly developed algorithms. Being bound by the dataset license, any model trained on this new dataset cannot be employed in practical commercial scenarios. It is, therefore, desirable to seek more independent ways of training deep learning models in biometrics.

It has been already shown in Chapter 5 that one can overcome such limitation by using synthetic training data. An essential building element of such an approach is a good synthetic data generator. Recent developments in hand pose estimation have provided us with a very convenient deformable model of three-dimensional hands called MANO [Romero et al., 2017], which is publicly available. It allows generating hands of arbitrary shapes in arbitrary poses. The generated hand sample is controlled by two sets of parameters. First are the so-called shape parameters $\mathscr{S} \subseteq \mathbb{R}^{10}$ that define the overall size of the hand and lengths and thickness of the fingers. The second group of parameters are the pose parameters $\mathscr{P} \subseteq \mathbb{R}^{12}$, where the first 9 parameters define the hand pose in terms of non-rigid deformations (e.g. bending fingers, etc.) and the last 3 parameters define the orientation of the whole hand in the three-dimensional space.



(a) (b)

Figure 7.4. Clustering of points for the synthetic samples using the skeleton joints into semantically meaningful parts. (a) The original point cloud; (b) Result of clustering the point cloud.

One can use the pre-trained MANO hand model to generate a theoretically unlimited amount of three-dimensional hands controlling their shape and pose via $\mathbf{s} \in \mathscr{S}$ and $\mathbf{p} \in \mathscr{P}$. Such three-dimensional models can be easily reprojected into range data. This is very suitable for training a person identification system since the parameters of the MANO model were estimated from real world data.

Each subject in the dataset will be represented by its shape parameters **s**. Those are allowed to vary only slightly for the same individual, but also have to vary noticeably for different individuals. Many samples for the same individual can be generated either with an arbitrary pose by randomly varying the pose parameters **p** or in a constant pose by keeping **p** fixed.

Synthetic training dataset For training of all models presented in this chapter, a synthetic dataset of 200 subjects with 50 samples per subject has been generated using the MANO model. This yields a total of 10000 training samples, a number that would be rather difficult to collect from real individuals, especially for commercial purposes. Each sample in the dataset is stored as a three-dimensional point cloud together with positions of the skeleton joints and the parameter vectors $\mathbf{s} \in \mathscr{S}$ and $\mathbf{p} \in \mathscr{P}$ that were used to generate it. The skeleton joints can be used to generate an approximate semantic segmentation of the hand surface into clusters, as shown in Figure 7.4. Additionally, each sample has a label assigning it to a particular subject.

7.2.2 PointNet++ baseline

As the first baseline method for the new dataset NNHand RGB-D, a PointNet++ [Qi et al., 2017] architecture, the successor of the famous PointNet by Qi et al. [2016], has been used. PointNet++ builds a multi-level grouping of points. At each level, always larger local regions are being abstracted along the hierarchy via sampling and grouping operations. This hierarchical sampling and grouping operation allows capturing local details better, as opposed to traditional PointNet, which uses only a single max-pooling operation over the whole point set for information aggregation.

The centerpiece of PointNet++ architecture is so-called *Set Abstraction (SA)* layer, which does sampling and grouping of points followed by PointConv [Qi et al., 2017] operation, which is internally a Multilayer Perceptron (MLP) operating on a set of points. SA module typically has three parameters. The sampling ratio r, the neighborhood radius ρ and the MLP subnetwork, which we can specify as MLP$(m, n, \ldots)$, where $m, n, \ldots$ are the number of parameters of each layer of the MLP. A sequence of SA modules is then followed by Global Aggregation (GA) module, which is just an MLP followed by global max-pooling operation. To reduce the dimensionality of the output and increase the capacity of the network, the GA module is usually followed by another MLP, which produces the final output.

In the above terms, we can define the baseline PointNet architecture as follows. It has two SA modules. The first SA module has $r = 0.5, \rho = 0.2$ and MLP$(3, 64, 64, 128)$. It is followed by second SA module with $r = 0.25, \rho = 0.4$ and MLP$(3 + 128, 128, 128, 256)$. The output of the second SA module is forked into two parallel branches. The first branch is supposed to output the shape parameters $\mathbf{s} \in \mathscr{S}$. It is composed by a GA module with MLP$(3{+}256, 256, 512, 1024)$ followed by another MLP subblock defined as MLP$(1024, 512, 256, 10)$. The second branch, instead, is outputting the pose parameters $\mathbf{p} \in \mathscr{P}$ and is composed of a GA module with MLP$(3 + 256, 256, 512, 1024)$ whose output is fed to an MLP module MLP$(1024, 512, 256, 12)$. The described architecture is depicted graphically in Figure 7.5.

Big PointNet++ A second version with more parameters has been evaluated in parallel. This model has a bigger subnetwork for the shape regression. In particular, the GA module is equipped with MLP$(3{+}256, 256, 512, 1024{\times}21)$ whose output is fed to an MLP module MLP$(1024 \times 21, \frac{1024\times21}{12}, \frac{1024\times21}{24}, 10 \times 21, 10)$.

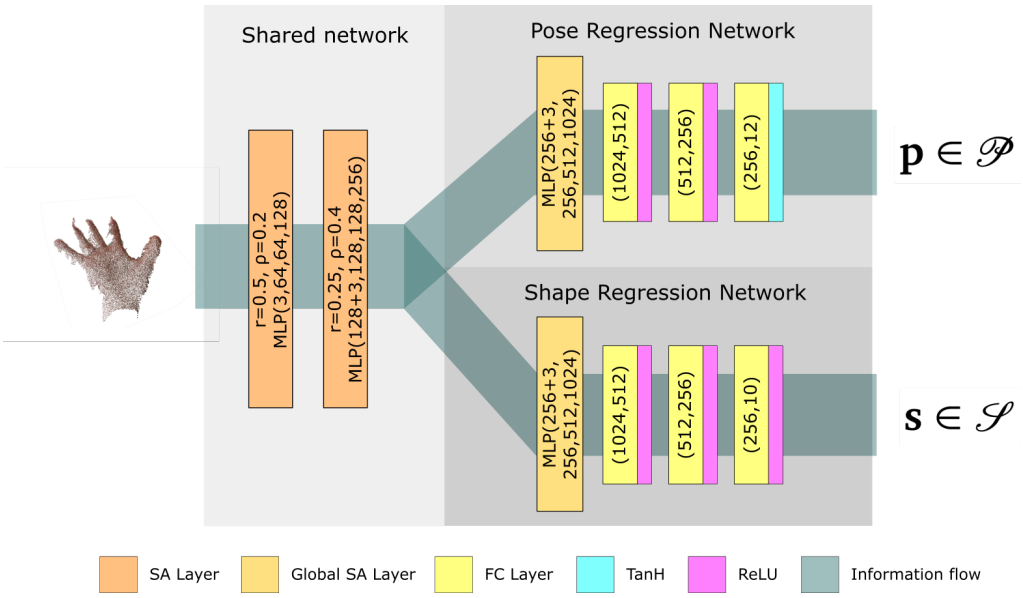


Figure 7.5. t PointNet++ architecture used as a baseline method.

Training. The model is trained on the synthetic dataset presented in Section 7.2.1. In particular, the optimization problem is posed as a regression over the shape and pose parameters $\mathbf{s} \in \mathscr{S}$ and $\mathbf{p} \in \mathscr{P}$ feeding a three-dimensional point cloud

as an input. It is defined using the following objective function for a batch of $M \in \mathbb{N}$ samples:

$$E = E_{\mathscr{S}} + \lambda E_{\mathscr{P}} \tag{7.1}$$

where $E_{\mathscr{S}}$ is the MSE loss for the regression of the shape parameters

$$E_{\mathscr{S}} = \frac{1}{M} \sum_{m=0}^{M-1} |\hat{\mathbf{s}}_m - \mathbf{s}_m|^2, \tag{7.2}$$

$E_{\mathscr{P}}$ is the MSE loss for the regression of the pose parameters

$$E_{\mathscr{P}} = \frac{1}{M} \sum_{m=0}^{M-1} |\hat{\mathbf{p}}_m - \mathbf{p}_m|^2, \tag{7.3}$$

and λ is a hyperparameter weighting the importance of regressing the pose parameters $\mathbf{p}$ with respect to the shape parameters $\mathbf{s}$.

7.2.3 Dynamic Graph CNN baseline (DGCNN)

Another baseline method that was evaluated on the NNHand RGB-D dataset is the Dynamic Graph CNN (DGCNN) by Wang et al. [2019]. Opposed to PointNet, DGCNN is able to exploit the local geometric structures better by constructing a local neighborhood graph and applying convolution-like operations on it. The graph is constructed dynamically in each layer of DGCNN using k-Nearest Neighbors search to determine the neighborhood of each point. This allows for non-local propagation of the information along the point cloud.

The main building block of DGCNNs is the EdgeConv module [Wang et al., 2019], which computes the graph dynamically at every layer of the network and performs the convolution-like operation based on an MLP. EdgeConv module has three parameters: the MLP used internally, the number of neighbors for the KNN search k and the type of the local aggregation performed in each neighborhood $aggr$, which is most often *mean* or *max*.

Borrowing some terminology from the previous subsection, we can define the baseline DGCNN architecture as follows. The model starts with two EdgeConv modules, both with $k = 10$ and *max* aggregation type. The first module has MLP$(3+3, 64, 64, 128)$ and the latter one MLP$(128+128, 256)$. Outputs of both EdgeConv modules are concatenated and passed forward. The model is then forked into two branches, one regressing the pose parameters $\mathbf{p} \in \mathscr{P}$ and the other one the shape parameters $\mathbf{s} \in \mathscr{S}$ of the input point cloud. The first branch is composed of a GA module with MLP$(128+256, 1024)$ followed by another MLP

subblock defined as MLP$(1024, 512, 256, 12)$. The second branch is almost the same, with only one difference. The final MLP block's output is 10-dimensional as it outputs the shape parameters $\mathbf{s}$. For more details, the architecture is depicted in Figure 7.6.

Big DGCNN A second version with more parameters has been evaluated in parallel. This model has a bigger subnetwork for the shape regression. In particular, the GA module is equipped with MLP$(128 + 256, 1024 \times 21)$ whose output is fed to an MLP module MLP$(1024 \times 21, \frac{1024\times21}{12}, \frac{1024\times21}{24}, 10 \times 21, 10)$.

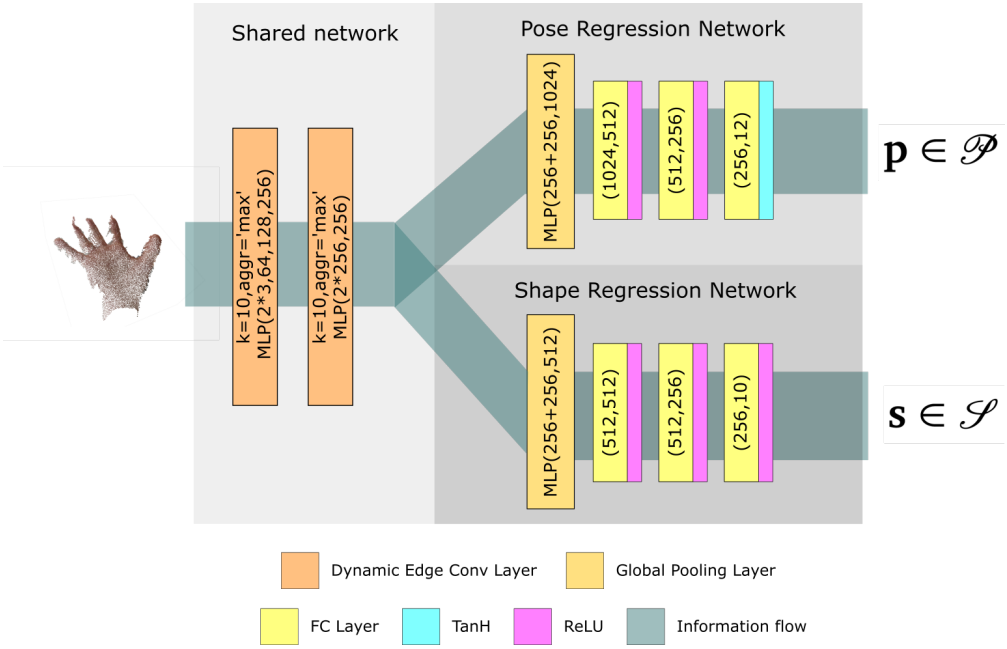


Figure 7.6. DGCNN architecture used as a baseline method.

Training. The model is trained on the synthetic dataset presented in Section 7.2.1. As for the PointNet++ baseline, also here the optimization problem is defined as a regression over the shape and pose parameters $\mathbf{s} \in \mathscr{S}, \mathbf{p} \in \mathscr{P}$ feeding a three-dimensional point cloud as an input. The objective is the same as in Equation 7.1.

7.2.4 Learning to cluster hand point clouds

The human hand is a complex and highly non-rigid surface. Moreover, RGB-D scans are often noisy. Matching noisy samples of hands using a global descriptor seems very challenging. An easier task would be to rather aim at describing the hand surface divided into semantically meaningful parts. These parts can be predefined based on human anatomy, for example by looking at the skeletal structure of the hand. Such clustered description (see Figure 7.7) retains more information and should be robust against noise and, possibly non-rigid, transformations.

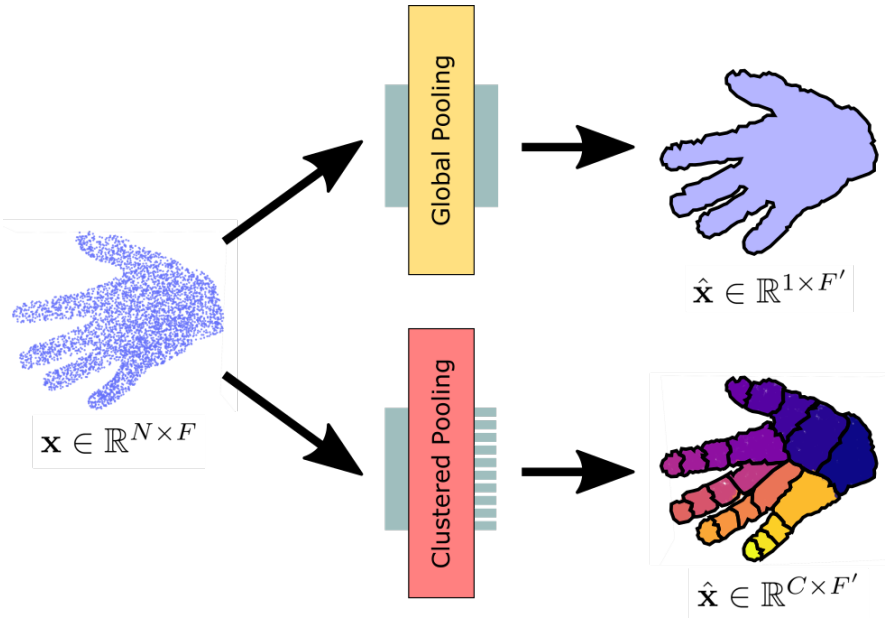


Figure 7.7. The difference between global and clustered pooling of the N input feature points. Global pooling creates a single new descriptor for the whole hand shape, while clustered pooling creates a new descriptor for each of the C semantically meaningful clusters.

To this end, inspired by the differentiable graph pooling [Ying et al., 2018; Cangea et al., 2018; Bianchi et al., 2019], the *Global Pooling Module* in the shape regression sub-network of architecture from Section 7.2.3 is replaced with a novel module called *Clustered Pooling Module* (see Figure 7.7), which is described next.

Clustered Pooling Module It allows to dynamically learn a clustering function $l : \mathbb{R}^F \rightarrow \mathbb{R}^C$, which produces cluster assignment probability vector $\mathbf{c} \in \mathbb{R}^{N \times C}$ into $C \in \mathbb{N}$ clusters for a vector of $N \in \mathbb{N}$ feature points $\mathbf{x} \in \mathbb{R}^{N \times F}$ as

$$\mathbf{c} = \text{softmax}(l(\mathbf{x})). \tag{7.4}$$

To get the clustered representation, the input feature points $\mathbf{x} \in \mathbb{R}^{N \times F}$ further undergo a non-linear transformation defined as $f : \mathbb{R}^F \rightarrow \mathbb{R}^{F'}$ and are subsequently aggregated into the C clusters as:

$$\mathbf{x}_f = f(\mathbf{x}), \tag{7.5}$$

$$\hat{\mathbf{x}} = \frac{\mathbf{c}^T \mathbf{x}_f}{\mathbf{D}}, \tag{7.6}$$

where the division represents a Hadamard division, $\mathbf{D} \in \mathbb{R}^{C \times F'}$ is a matrix with identical columns, where each column is defined as $\left(\sum_{i=1}^{N} \mathbf{c}_i\right)^T \in \mathbb{R}^{C \times 1}$ and $\hat{\mathbf{x}} \in \mathbb{R}^{C \times F'}$ is the pooled representation of the transformed input $\mathbf{x}_f \in \mathbb{R}^{F'}$. The novel clustered DGCNN architecture which we call *Cluster DGCNN* is schematically depicted in Figure 7.8.

Training. Similarly to the baselines, this model is trained on the synthetic dataset presented in Section 7.2.1. The optimization problem is posed as a regression over the shape and pose parameters $\mathbf{s} \in \mathscr{S}$ and $\mathbf{p} \in \mathscr{P}$, and simultaneous classification of the point clusters while feeding a three-dimensional point cloud as an input. It is defined using the following objective function for a batch of $M \in \mathbb{N}$ samples:

$$E = E_{\mathscr{S}} + \lambda_1 E_{\mathscr{P}} + \lambda_2 E_{clust}, \tag{7.7}$$

where $E_{\mathscr{S}}$ and $E_{\mathscr{P}}$ are described by the Equation 7.1. E_{clust} is a cross-entropy loss which enforces the classification of points into correct clusters. It is defined as:

$$E_{clust} = \frac{1}{M} \sum_{m=0}^{M-1} -\log\left(\frac{\exp(\mathbf{c}_m^{y_m})}{\sum_j \exp(\mathbf{c}_m^j)}\right), \tag{7.8}$$

where $\mathbf{c}_n$ is the vector of cluster probabilities for points in a point cloud and y are the cluster labels of these points. Hyperparameter λ_1 is weighting the importance of regressing the pose parameters $\mathbf{p} \in \mathscr{P}$ with respect to the shape parameters $\mathbf{s} \in \mathscr{S}$ and λ_2 is a hyperparameter weighting the importance of the cluster classification loss.

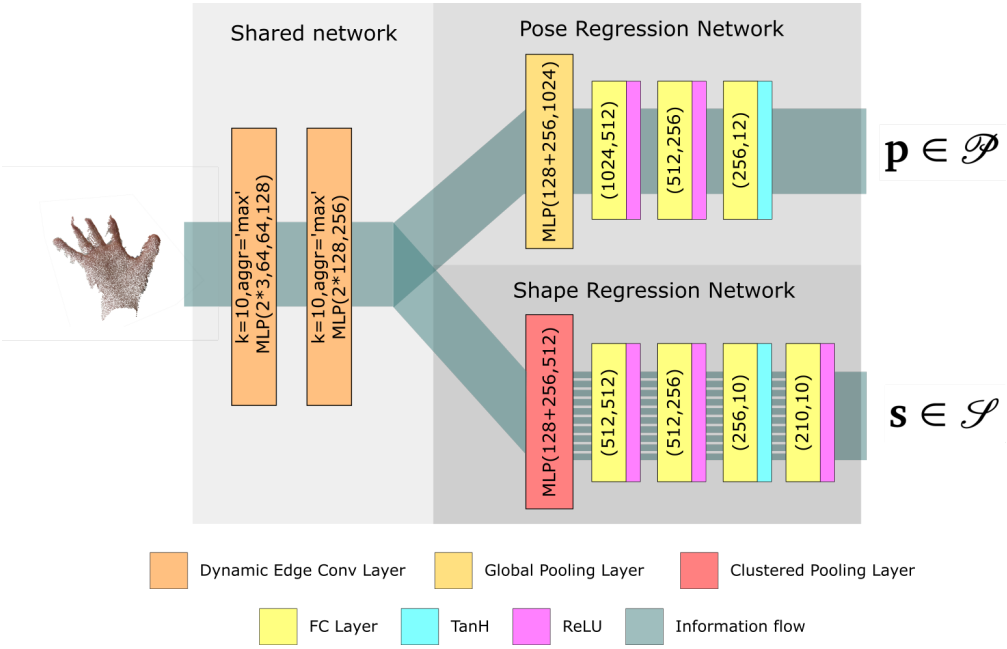


Figure 7.8. Improved DGCNN architecture with hand point clustering. The Global Pooling Layer in Shape Regression Network is replaced with our novel Clustered Pooling Layer.

7.3 Experiments

All models presented in this section have been extensively tested on the newly collected dataset *NNHand RGB-D* in order to provide baseline results. The experiments are designed so to highlight the possible usage of the dataset as well as point out some relevant challenges that come along the way. The new dataset consists of video sequences and therefore requires some pre-processing in order to obtain RGB-D frames that the baseline evaluation has been done on. Experiments with both All-To-All matching as well as splitting the database into reference and probe subsets and performing Reference-Probe matching have been conducted.

For matching, we consider the per-cluster shape parameters as the output feature vector in case of our novel *Cluster DGCNN*. There are 21 different clusters which results in a vector of 210 dimensions. For a fair comparison, in case of *PointNet++* and *DGCNN* baselines, which both perform a global pooling, we take the output of the layer before the last in the shape regression network as the feature vector, which has 256-dimensions. Different metrics have been tried for

computing the distance, where the L1 metric has shown to be the most suitable one.

In addition, we evaluate the proposed baselines on the current standard benchmark in the field, which is the Hong Kong Polytechnic University Contact Free 3D/2D Hand Images database[3] [Kanhangad et al., 2009, 2011], referred to as *HKPolyU v1 and HKPolyU v2 database* further in the text.

HKPolyU v1 database. A dataset of 177 subjects containing in total 1770 RGB-D samples that were acquired with high precision Minolta Vivid 910 range scanner. Each subject has been scanned in two sessions in different time periods, obtaining 5 samples per session. The precision of the data is enough to perform both 3D hand geometry and 3D palmprint recognition.

HKPolyU v2 database. A dataset of 114 subjects with a total of 570 RGB-D samples that were acquired using the Minolta Vivid 910 range scanner. Each subject has been scanned 5 times, each time presenting his hand on different global orientation. Besides, the precision of the data is enough to perform both 3D hand geometry and 3D palmprint recognition.

7.3.1 Data pre-processing

A dataset of fixed RGB-D frames has been sampled from the video sequences. For each subject, sequence number 1 has been taken and 10 samples have been acquired while the hand is held straight up with the fingers extended and palm facing the camera. The dataset currently contains 79 subjects, which gives a total of 790 samples. Similarly, sequence number 2 has been used to obtain a second set of 790 samples. For reproducibility of this evaluation, the acquired subset of RGB-D frames is stored together with the original NNHand RGB-D dataset.

Each frame captured from the video sequences undergoes several pre-processing steps. First, the background is removed using the depth information. Subsequently, to avoid problems with objects or other parts of the body appearing in the frames, a mask keeping only the central area of each frame is applied (see Figure 7.9). Next, an OpenPose-based [Cao et al., 2017] single RGB image hand pose estimator[4] is used to estimate the hand keypoints. Thanks to the one-to-one mapping between RGB and depth information, this allows to filter out the unde-

[3] https://www4.comp.polyu.edu.hk/ csajaykr/myhome/database$_r equest$/3$dhand$/$Hand3D.htm$

[4] https://github.com/erezposner/HandKeyPointDetector

sired part of the hand below the wrist in the whole RGB-D frame. Step-by-step preprocessing of a random frame is depicted in Figure 7.9.

In the pre-processed subset, each sample is stored in two files. The first file is the pre-processed colored point cloud stored in PLY format. The second file is then additional information for the point cloud, namely the color and depth frame the point cloud was taken from together with the detected hand keypoints.

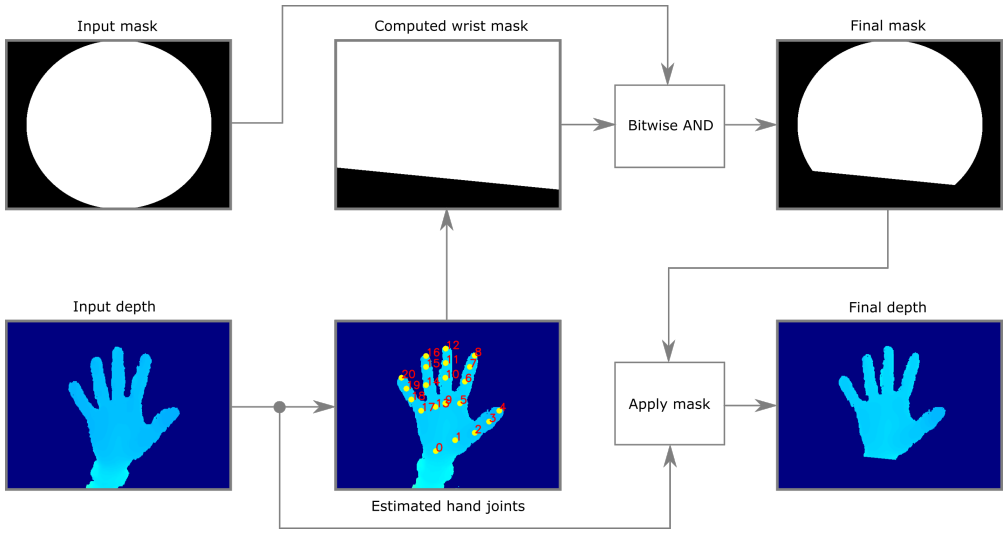


Figure 7.9. Each depth sample in the dataset undergoes the following pre-processing steps before it is saved as a point cloud.

Point cloud pre-processing. The hand point cloud extracted from the pre-processed depth image further undergoes the following pre-processing steps before feature extraction and matching takes places. First, each point cloud is subsampled using Furthest Point Sampling (FPS) [Eldar, 1992; Eldar et al., 1997] to 4096 points. Consequently, each sample is aligned to a reference hand point cloud using the Iterative Closest Point (ICP) [Zhang, 1994] algorithm.

7.3.2 All-To-All matching

In this experiment, each feature vector is taken and its distance to feature vectors of all other samples in the dataset is computed. The sample with the shortest distance is taken as the matching class. The performance is shown in terms of Top-1 accuracy and EER in Table 7.1. The ROC curve for this experiment is shown in Figure 7.10.

Table 7.1. All-To-All matching performance of presented methods on different datasetes in terms of Top-1 accuracy and EER.

Method	NNHand RGB-D		HKPolyU v1		HKPolyU v2	
	Top-1 [%]	EER [%]	Top-1 [%]	EER [%]	Top-1 [%]	EER [%]
PointNet++	53.42	47.19	30.40	34.28	9.12	37.55
Big PointNet++	48.35	34.79	40.62	38.51	17.72	44.24
DGCNN	76.20	21.70	84.63	19.03	39.12	27.40
Big DGCNN	73.54	22.05	73.79	19.66	29.30	27.62
Cluster DGCNN	**98.23**	**14.45**	**99.27**	**7.92**	**59.65**	**25.08**

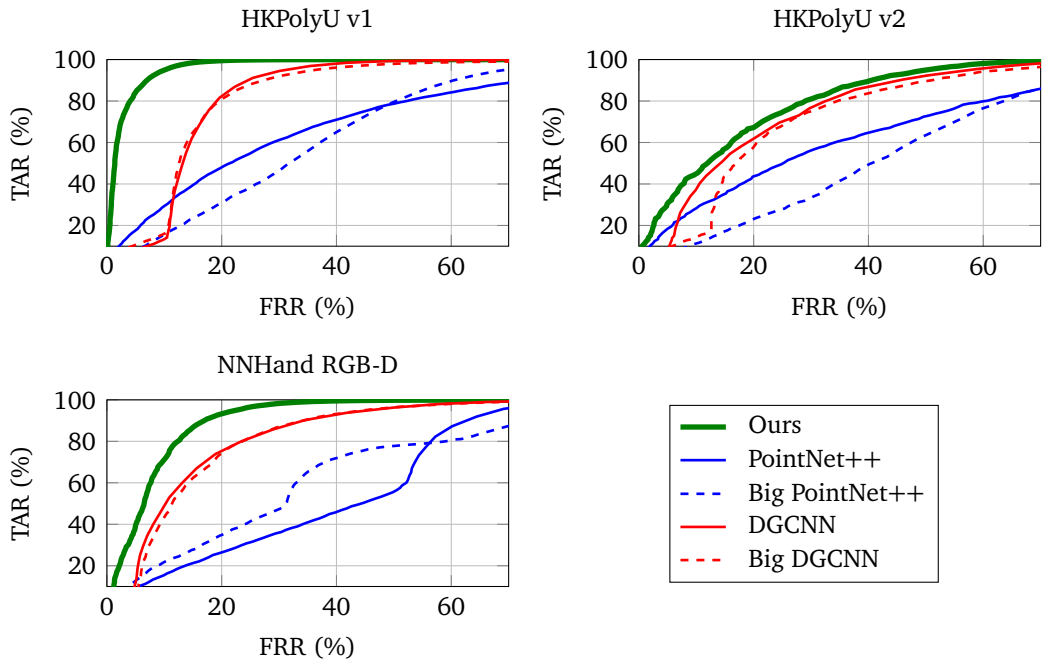


Figure 7.10. All-To-All matching ROC curves (tradeoff between acceptance and rejection rates) of the presented methods on different datasets.

The baseline performance is set by the results obtained using PointNet++ and DGCNN methods. First, increasing the number of parameters of the *Shape Regression Network* does not help and could be attributed to overfitting due to use of an over-parametrized model, which results in a drop in performance during test time. Second, our novel Cluster DGCNN outperforms both baselines by a margin and sets new state-of-the-art on the NNHand RGB-D dataset as well as HKPolyU v1 and v2 standard benchmarks. The original works presenting HKPolyU datasets

do not perform this experiment and we, therefore, cannot compare to them directly.

7.3.3 Reference-Probe matching

A very popular way of evaluating biometric algorithms on diverse datasets is performing so-called reference - probe matching, where the dataset is split into two parts, one is the reference (i.e. the database) and the rest is the probe (i.e. the samples one wants to identify). Different splitting strategies have been applied depending on the dataset at hand.

For the HKPolyU v1 dataset, the splitting strategy proposed by Kanhangad et al. [2009] is followed, choosing the 5 samples from the first session as the reference and the 5 samples from the second session as the probe for each user.

In case of HKPolyU v2, we use the splitting strategy used in Kanhangad et al. [2011], where 1 sample is chosen as probe and all the other 4 as reference. This process is repeated 5 times, always picking different sample as the probe to produce the genuine and impostor scores for the generation of the ROC curve and computation of the EER.

NNHand RGB-D database has 10 samples per user from sequence 1 and another 10 samples from sequence 2. For each user, the 10 samples from sequence 1 are selected as the reference and the other 10 samples from sequence 2 are left as the probe.

Results for all datasets obtained using different methods are summarized in Table 7.2 in terms of Top-1 accuracy and EER. The ROC curves are then presented in Figure 7.11.

Table 7.2. Reference-probe matching performance of presented methods on different datasets in terms of Top-1 accuracy and EER.

Method	NNHand RGB-D		HKPolyU v1		HKPolyU v2	
	Top-1 [%]	EER [%]	Top-1 [%]	EER [%]	Top-1 [%]	EER [%]
PointNet++	24.05	39.08	17.29	33.14	12.10	35.39
Big PointNet++	18.86	40.54	22.37	34.72	13.68	37.70
DGCNN	24.56	24.31	54.24	16.79	42.98	16.58
Big DGCNN	28.48	23.45	45.54	15.83	31.92	19.53
Cluster DGCNN	**62.28**	**13.75**	**85.65**	**7.26**	**69.82**	**10.48**

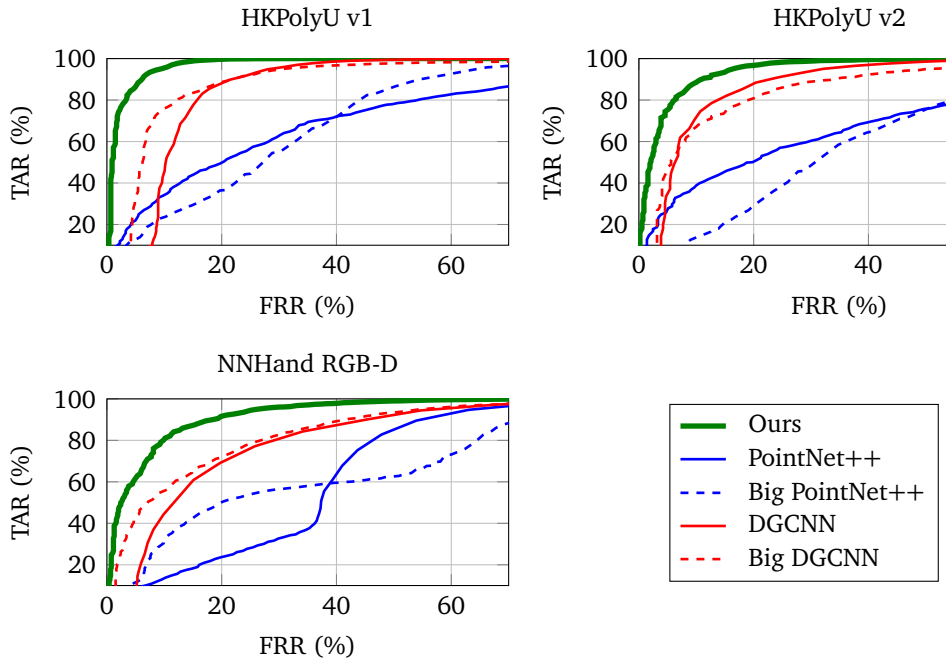


Figure 7.11. Reference-probe matching ROC curves (tradeoff between acceptance and rejection rates) of the presented methods on different datasets.

The baseline performance is again represented by results on PointNet++ and DGCNN methods. As in the All-To-All case, also for the reference-probe matching the new state-of-the-art on NNHand RGB-D dataset is set by our novel Cluster DGCNN, which superates both baselines by a margin.

On the HKPolyU v1 dataset, comparing to results of Kanhangad et al. [2009], our method performs on-par, with Equal Error Rate (EER) higher by a few percent. This is accounted to the fact, that as opposed to Kanhangad et al. [2009], which analyzes HKPolyU samples in full resolution, here the evaluation is done on subsampled point clouds of only 4096 points per model. In case of the HKPolyU v2 dataset, we can compare to the results presented by Kanhangad et al. [2011], where we outperform their method in terms of EER by a margin of 7% (which is an improvement of 60% compared to their EER of 17.2%). This further supports the high potential of our novel method.

7.3.4 Semantic segmentation analysis

Our method (which we call *Cluster DGCNN*), besides others, outputs the semantic segmentation of the point cloud into parts, which the network was enforced to

learn during training by the cluster assignment loss (see Equation 7.7) using the cluster annotations provided with the synthetic training samples (see Figure 7.4).

There is no ground truth segmentation for the testing data, we, however, provide a qualitative evaluation in Figure 7.12, which supports that the *Cluster DGCNN* has learnt to segment the point cloud in a meaningful way. Aggregating information inside each cluster, therefore, provides a meaningful piece-wise representation of the point cloud.

One should notice that due to the presence of noise in the input point clouds, the segmentation is prone to produce some outliers in the finger regions (see Figure 7.12). Influence of such inconsistencies on the final descriptor is reduced by averaging feature vectors in each semantic region in order to produce the global segment descriptor.

7.4 Discussion

A new dataset of RGB-D sequences of human hands, NNHand RGB-D, has been presented. Each participant performs several predefined hand gestures in front of the camera. This allows analyzing the surface of the hand from both palm and dorsal side, as well as different positions. To the best of our knowledge, it is the first dataset of this kind acquired mainly for biometric purposes. It opens the door to new research in three-dimensional hand shape biometrics, in particular non-rigid identity matching as well as matching from dynamic RGB-D sequence.

Together with the dataset, a transfer learning approach for three-dimensional hand shape feature extraction, which is based on geometric deep learning principles, is presented. It is shown how to use synthetic data to train models which are shown to generalize well on real data during testing. Several baseline methods are presented together with our novel approach, which builds upon the DGCNN model and learns semantic segmentation of the hand surface into parts. Such segmentation describes the hand surface in a very structured way. We show its efficiency both on current standard benchmark HKPolyU as well as NNHand RGB-D dataset on which it sets the new state-of-the-art.

This is however only the first step in exploiting the possibilities of the NNHand RGB-D dataset. The frames used during the experiments in this chapter were captured in a rather good position, the hand being upright and palm facing the camera. There is more such as fully non-rigid hand matching and matching from RGB-D video sequences, which both remain open problems.

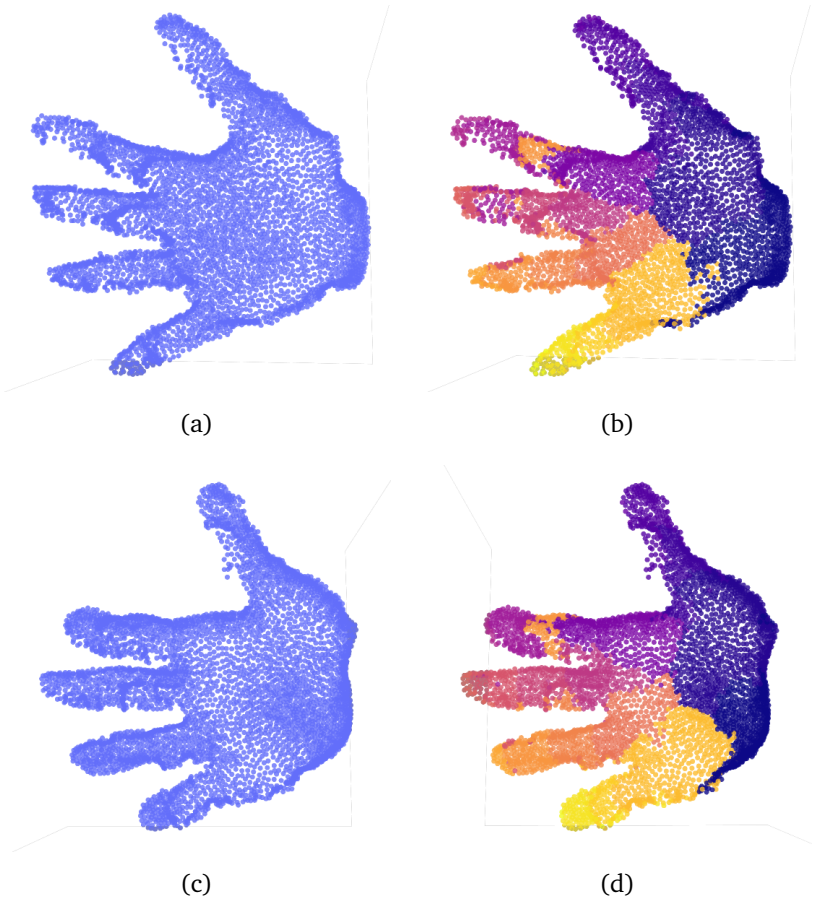


Figure 7.12. Clustering of points computed by Clustered DGCNN for a real sample. One can notice some classification errors around some of the fingers due to high level of noise in the input data. (a,c) The original point cloud; (b,d) Result of clustering the point cloud.

Chapter 8

Conclusions

This work brings the state-of-the-art methods from deep learning for computer vision into three-dimensional hand-geometry biometrics and opens new possibilities of advancing the field. Chapter 2 provides an introduction into hand geometry biometrics and some related biometric modalities. Chapter 3 presents the advances in computer vision and machine learning that motivated this work. Utilizing this inspiration, Chapters 4-7 present novel methods to improve hand geometry biometrics, either directly or via fusion with other hand-based biometric modalities. After a short review of what has been presented thus far, this chapter concludes by proposing possible future directions.

Biometric fusion

Hand geometry biometrics often analyze data which provide not only hand-shape information but also the texture of the hand surface. A natural extension of hand-geometry-based systems is, therefore, the analysis of palmprint or, if visible, fingerprint information.

Discriminative index CNN's. [Svoboda et al., 2016] These are an alternative to handcrafted palmprint-feature extractors that allow learning the optimal features to describe a palmprint using a metric-learning approach. We replace the traditional Siamese loss, taking into account the desire to separate genuine and impostor score distributions during matching. These distributions are typically modelled statistically as two normal distributions, and the aim is to minimize their overlap, facilitated by the discriminative index optimization.

We experimentally show the advantage of using d-prime loss over Siamese loss. Evaluations on standard benchmarks for palmprint biometrics show the

superiority of our approach in terms of Receiver Operating Curve (ROC) and the Equal Error Rate (EER) scores. We compute the discriminative index for our solution, beating that of the baseline, which is expected since we derive our loss based on that quantity.

Additionally, we propagate the computed feature vectors back through the network in order to analyze what they represent. We show that each element of the output feature vector activates for a different combination of palm lines or wrinkles of the hand. Altogether, the output feature vector can, therefore, be considered a global representation of the hand palm lines and wrinkles.

Latent fingerprint autoencoders. [Svoboda et al., 2017] Performing identification from poorly-visible, partial fingerprints is a highly challenging task. A representative example of this class of fingerprints are latent impressions lifted from surfaces, for example, at crime scenes. State-of-the-art approaches in the field design robust feature extractors or matching techniques, which aim to be resilient to many missing minutiae points.

We take a different approach that operates solely as a pre-processing stage and allows for subsequent reuse of existing feature-extraction and matching techniques. Inspired by the performance of convolutional autoencoders and generative models in image processing, we employ a convolutional autoencoder to "fill in" the missing pieces of a partial fingerprint. We train the network using a carefully designed set of losses which enforces the model to preserve the consistency of the original ridgelines.

To overcome the challenge of missing training data, a synthetic dataset of latent fingerprints is generated using open-source fingerprint generators. We show how to efficiently train the model on synthetic data and perform a successful transfer on real-world samples afterwards.

Our fingerprint enhancement technique allows the existing feature extraction and matching methods to reach state-of-the-art performance on the standard benchmarks IIIT-Delhi Latent Fingerprint and IIIT-Delhi MOLF datasets on both latent-to-latent and latent-to-sensor matching tasks. Subsequently, the fingerprints enhanced using this novel approach are analyzed using the NIST Fingerprint Image Quality tool, which confirms that they are, indeed, of better quality.

Hand geometry biometrics

In addition to proposing improvements related to biometric fusion, we devote ourselves to improving the hand-geometry analysis itself. Inspired by the current state-of-the-art in deep learning for 3D shape analysis and hand pose estimation,

we propose a novel approach to 3D hand-geometry processing for biometric applications. Furthermore, we provide a novel dataset to test our method on.

NNHand RGB-D dataset. The available biometric datasets for 3D hand shape recognition are not very suitable for our application. Namely, HK PolyU v1 and v2 both provide very high precision range scans with restricted environmental conditions during capture.

We, therefore, resort to collecting our own dataset [Svoboda et al., 2020b] of hand-range scans using an off-the-shelf 3D camera Intel RealSense SR-300. The dataset currently contains 79 subjects and three short RGB-D video sequences per subject. In each sequence, the user is performing a pre-defined hand gesture several times repeatedly.

Collecting dataset of video sequences with different hand gestures for many identities opens the door to a variety of new research directions in hand geometry recognition. Besides palm or dorsal side facing hand samples, one can think of fully non-rigid hand shape recognition from an arbitrary pose or recognition from a video sequence instead of static frames.

Two baseline methods have been evaluated on the newly collected dataset for the basic setting: the hand being held upright with the palm facing the camera and fingers opened. In particular, we have applied two state-of-the-art methods for 3D point cloud classification PointNet++ and Dynamic Graph CNN. Due to a lack of training data, we train both models adapting the transfer learning approach we have used for latent fingerprint autoencoders. The MANO hand generative model is used in order to generate a synthetic training database of hands. Each hand is defined by a set of shape and pose parameters. The task of PointNet++ and DGCNN is to regress these parameters given the 3D point cloud as input. Features generated from the regressed shape parameters are then used for matching. DGCNN shows superior compared to PointNet++ in all of our experiments which we address to the fact that it constructs a neighborhood graph dynamically and is, therefore, able to better exploit local neighborhood relations.

Clustered DGCNN. Describing the whole hand surface globally by a small set of shape parameters can be rather challenging and too ambiguous for practicality. We use the skeleton joints provided by the MANO hand model and then cluster each synthetic hand sample into semantically meaningful parts. Taking the better performing DGCNN model and replacing the Global Pooling Layer with our novel Clustered Pooling Layer [Svoboda et al., 2020b], we leverage the cluster annotations of the synthetic data in order to train the model to, besides regress-

ing shape and pose parameters, semantically clustering the hand into parts. This allows us to generate a set of shape parameters per-cluster and create a piece-wise descriptor of the input point cloud.

The Clustered DGCNN is evaluated both on NNHand RGB-D dataset, as well as subsampled point clouds from HK PolyU v1 and HKPolyU v2. It yields competitive results on HK PolyU v1 while setting the new state-of-the-art performance for NNHand RGB-D and HKPolyU v2 datasets. Results obtained on different datasets confirm our assumption that piece-wise description of the point-cloud will have more expressive power than a single global descriptor for the whole hand shape. Furthermore, to show that our model yields meaningful and consistent semantic clustering on the real-world datasets as well, despite it was trained on synthetic data, some qualitative clustering results are provided.

Neural networks robustness

One would likely notice that all the novel methods introduced in this text are based on deep learning, in particular convolutional neural networks or their equivalents for graph-structured data. Despite outstanding performance of the state-of-the-art deep learning approaches on many tasks, convolutional and graph neural networks have been shown to be unstable, breaking down with a rather negligible structured noise (adversarial perturbation) added to the input sample. This poses a direct threat to biometric systems based on such models.

We have therefore devoted some time to studying adversarial perturbations and how to defend against them. As a result, Peer Regularized Networks (PeerNets) [Svoboda et al., 2019] have been presented. PeerNets extend the existing models, e.g. LeNet, ResNet, by interleaving convolutional layers with peer regularization. For each input sample, feature maps are recombined from feature maps of peer samples obtained from a graph of peers by means of KNN search during peer regularization.

Robustness of PeerNets has been evaluated using LeNet-5 and ResNet on standard datasets MNIST, CIFAR-10 and CIFAR-100 defending against various types of adversarial attacks showing increased robustness. We have performed additional evaluations to show that PeerNets do not suffer from gradient obfuscation, which has just recently become a real concern for adversarial attack defenses.

8.1 Potential future work

The presented state-of-the-art methods and results, together with our new dataset, open doors to new lines of research:

Palm lines and wrinkle segmentation using d-prime loss.

Palmprint recognition systems based on convolutional neural networks clearly benefit from utilization of the d-prime loss for model training. Besides utilizing the output feature vectors directly for identification, one can use them together with modified backpropagation in order to produce saliency maps. This has been shown to highlight the palm lines and more prominent palm wrinkles and may be used to segment them. Palm-line segmentation opens new possibilities for applying ridge-based or line-based palmprint-recognition methods.

Hand shape recognition from video sequences.

The new NNHand RGB-D dataset is the first biometric dataset for hand shape recognition to store RGB-D video sequences instead of only static frames. This gives researchers the opportunity to work on hand-positioning systems, which could detect the ideal position of the hand before saving a snapshot. Likewise, it gives way for a potential new branch of hand-geometry recognition from video sequences, where not only the hand shape but also the behavioral characteristics of each user can be studied in order to create a stronger imprint of a person's identity.

Non-rigid hand shape recognition.

There exists a significant amount of quality research on non-rigid shape matching in the literature. Nevertheless, none have attempted to employ such methods in biometric hand-geometry recognition. Supposedly, this was due to the lack of data to evaluate such methods on. Our NNHand RGB-D dataset allows capturing hand snapshots in various poses and therefore removes the hurdle to this previously-unexplored area of research.

Determining gender and age from the shape of the hand.

Besides assigning an anonymous ID to each subject in the NNHand RGB-D dataset, we also stored the gender and, where given, the age information. Besides the plethora of other applications mentioned already in this text, the dataset possibly

allows investigating whether it is possible to tell the gender and/or the age of an individual based on the shape of his/her hand.

Genetic traits in hands

In 2014, Claes et al. [2014] have set the first steps in the recovery of human facial shape from the DNA based on a generative model. Performance of such generative models can be potentially improved with the recent introduction of 3D mesh convolutional autoencoders applied to 3D face [Ranjan et al., 2018] and hand [Tretschk et al., 2019; Kulon et al., 2019] reconstruction and generation. Having a dataset of human hands with DNA information available, one could investigate the recovery of a hand shape of an individual from their DNA information.

Image style transfer and inpainting with PeerNets.

PeerNets represent a novel family of neural network architectures interleaving traditional Euclidean convolutions with graph convolutions in order to drastically mitigate the effects of adversarial attacks. Such architectures have broad applications in machine learning, proven by their recent application to arbitrary-image style transfer [Svoboda et al., 2020a]. Similarly, we believe PeerNets can be applied to image inpainting and, outside of image processing, might find use in e.g. transfer learning, biometrics, etc.

Publications during the PhD program

Svoboda, J., Bronstein, M. M. and Drahansky, M. [2015]. Contactless biometric hand geometry recognition using a low-cost 3d camera, *International Conference on Biometrics (ICB)*.

Svoboda, J., Masci, J. and Bronstein, M. M. [2016]. Palmprint recognition via discriminative index learning, *International Conference on Pattern Recognition (ICPR)*.

Svoboda, J., Monti, F. and Bronstein, M. M. [2017]. Generative convolutional networks for latent fingerprint reconstruction, *International Joint Conference on Biometrics (IJCB)*.

Monti, F., Boscaini, D., Masci, J., Rodolà, E., Svoboda, J. and Bronstein, M. M. [2017]. Geometric deep learning on graphs and manifolds using mixture model CNNs, *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Svoboda, J., Cashman, T. J. and Fitzgibbon, A. W. [2018]. QRkit: Sparse, Composable QR Decompositions for Efficient and Stable Solutions to Problems in Computer Vision, *Winter Conference on Applications of Computer Vision (WACV)*.

Svoboda, J., Masci, J., Monti, F., Bronstein, M. M. and Guibas, L. J. [2019]. PeerNets: Exploiting Peer Wisdom Against Adversarial Attacks, *International Conference on Learning Representations (ICLR)*.

Svoboda, J., Anoosheh, A., Osendorfer, C. and Masci, J. [2020a]. Two-Stage Peer-Regularized Feature Recombination for Arbitrary Image Style Transfer, *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Svoboda, J., Astolfi, P., Boscaini, D., Masci, J. and Bronstein, M. M. [2020b]. Clustered Dynamic Graph CNN for Biometric 3D Hand Shape Recognition, *To be published soon.*

Bibliography

Abraham, J., Kwan, P. and Gao, J. [2011]. *Fingerprint Matching using A Hybrid Shape and Orientation Descriptor*, State of the art in Biometrics, InTech.

Afifi, M. [2019]. 11k hands: gender recognition and biometric identification using a large dataset of hand images, *Multimedia Tools and Applications* .

Allegion [2017]. HandKey II, `https://us.allegion.com/en/home/products/\categories/biometrics/schlage-handkeyII.html`.

Ansari, A.-H. [2011]. *Generation and storage of large synthetic fingerprint database*, Master's thesis, IISc. M.E. Thesis.

Ariyanto, A., Djamal, E. C. and Ilyas, R. [2018]. Personality identification of palmprint using convolutional neural networks, *2018 International Symposium on Advanced Intelligent Informatics (SAIN)* pp. 90–95.

Athalye, A., Carlini, N. and Wagner, D. A. [2018]. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples, *International Conference on Machine Learning (ICML)* .

Atwood, J. and Towsley, D. [2016]. Diffusion-convolutional neural networks, *Annual Conference on Neural Information Processing Systems (NIPS)*.

Barnes, C., Shechtman, E., Finkelstein, A. and B., G. D. [2009]. PatchMatch: A randomized correspondence algorithm for structural image editing, *ACM Transactions on Graphics (Proc. SIGGRAPH)* **28**(3).

Bartunek, J. S., Nilsson, M., Nordberg, J. and Claesson, I. [2006]. Adaptive fingerprint binarization by frequency domain analysis, Asilomar Conference on Signals, Systems and Computers, pp. 598–602.

Bensid, K., Samai, D., Laallam, F. Z. and Meraoumia, A. [2018]. Deep learning feature extraction for multispectral palmprint identification, *Journal of Electronic Imaging* **27**(3).

Bianchi, F. M., Grattarola, D. and Alippi, C. [2019]. Mincut pooling in graph neural networks, *arXiv preprint arXiv:1907.00481* .

Bouritsas, G., Bokhnyak, S., Ploumpis, S., Bronstein, M. M. and Zafeiriou, S. [2019]. Neural 3d morphable models: Spiral convolutional networks for 3d shape representation learning and generation, *International Conference on Computer Vision (ICCV)* .

Bromley, J., Guyon, I., LeCun, Y., Säckinger, E. and Shah, R. [1993]. Signature verification using a "siamese" time delay neural network, Annual Conference on Neural Information Processing Systems (NIPS).

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A. and Vandergheynst, P. [2017a]. Geometric deep learning: going beyond euclidean data, *IEEE Signal Process. Mag.* **34**(4): 18–42.

Bronstein, M. M., Bruna, J., Szlam, A., Bresson, X. and LeCun, Y. [2017b]. Geometric deep learning on graphs and manifolds, `http://geometricdeeplearning.com/slides/NIPS-GDL.pdf`. NIPS 2017 Tutorial.

Bruna, J., Zaremba, W., Szlam, A. and LeCun, Y. [2014]. Spectral networks and locally connected networks on graphs, *International Conference on Learning Representations (ICLR)* .

Buades, A., Coll, B. and Morel, J.-M. [2005]. A non-local algorithm for image denoising, *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Cangea, C., Veličković, P., Jovanović, N., Kipf, T. and Liò, P. [2018]. Towards sparse hierarchical graph classifiers, *arXiv preprint arXiv:1811.01287* .

Cao, K. and Jain, A. K. [2015]. Latent orientation field estimation via convolutional neural network, *International Conference on Biometrics (ICB)*.

Cao, K., Liu, E. and Jain, A. K. [2014]. Segmentation and enhancement of latent fingerprints: A coarse to fine ridgestructure dictionary, *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* **36**(9): 1847–1859.

Cao, Z., Hidalgo, G., Simon, T., Wei, S.-E. and Sheikh, Y. [2017]. Realtime multi-person 2d pose estimation using part affinity fields, *Conference on Computer Vision and Pattern Recognition (CVPR)* .

Cappelli, R., Ferrara, M. and Maltoni, D. [2010]. Minutia cylinder-code: A new representation and matching technique for fingerprint recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* **32**(12): 2128–2141.

Cappelli, R., Ferrara, M. and Maltoni, D. [2011]. Fingerprint indexing based on minutia cylinder-code, *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* **33**(5): 1051–1057.

Cappelli, R., Maio, D. and Maltoni, D. [2000]. Synthetic fingerprint-image generation, *International Conference on Pattern Recognition (ICPR)*.

Caron, M., Bojanowski, P., Joulin, A. and Douze, M. [2018]. Deep clustering for unsupervised learning of visual features, *European Conference on Computer Vision (ECCV)* .

Caruana, R., Lawrence, S. and Giles, L. [2000]. Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping, Annual Conference on Neural Information Processing Systems (NIPS).

Chakraborty, A., Alam, M., Dey, V., Chattopadhyay, A. and Mukhopadhyay, D. [2018]. Adversarial attacks and defences: A survey, *arXiv preprint arXiv:1810.00069* .

Chen, X., Wang, G., Guo, H. and Zhang, C. [2017]. Pose guided structured region ensemble network for cascaded hand pose estimation, *Neurocomputing* .

Chopra, S., Hadsell, R. and LeCun, Y. [2005]. Learning a similarity metric discriminatively, with application to face verification, Conference on Computer Vision and Pattern Recognition (CVPR).

Ciresan, D. C., Meier, U., Gambardella, L. M. and Schmidhuber, J. [2011a]. Convolutional neural network committees for handwritten character classification, *International Conference on Document Analysis and Recognition (ICDAR)*.

Ciresan, D. C., Meier, U., Masci, J., Gambardella, L. M. and Schmidhuber, J. [2011b]. Flexible, high performance convolutional neural networks for image classification, *International Joint Conferences on Artificial Intelligence (IJCAI)*.

Claes, P., Liberton, D. K., Daniels, K., Rosana, K. M., Quillen, E. E., Pearson, L. N., McEvoy, B., Bauchet, M., Zaidi, A. A., Yao, W., Tang, H., Barsh, G. S., Absher, D. M., Puts, D. A., Rocha, J., Beleza, S., Pereira, R. W., Baynam, G., Suetens,

P., Vandermeulen, D., Wagner, J. K., Boster, J. S. and Shriver, M. D. [2014]. Modeling 3d facial shape from dna, *PLOS Genetics* **10**(3): 1–14.

Clevert, D.-A., Unterthiner, T. and Hochreiter, S. [2015]. Fast and accurate deep network learning by exponential linear units (elus), *arXiv preprint arXiv:1511.07289* .

Cybenko, G. [1989]. Approximation by superpositions of a sigmoidal function, *Mathematics of Control, Signals and Systems* **2**: 303–314.

Dabouei, A., Soleymani, S., Kazemi, H., Iranmanesh, S. M., Dawson, J. and Nasrabadi, N. M. [2018]. ID preserving generative adversarial network for partial latent fingerprint reconstruction, *International Conference on Biometrics Theory, Applications and Systems (BTAS)*.

Darlow, L. N. and Rosman, B. [2017]. Fingerprint minutiae extraction using deep learning, *International Joint Conference on Biometrics (IJCB)*.

Daugman, J. [2000a]. Biometric decision landscapes, University of Cambridge, Technical Report No. 482.

Daugman, J. [2000b]. Combining multiple biometrics, `https://www.cl.cam.ac.uk/~jgd1000/combine/combine.html`.

de Santos-Sierra, A., Sánchez-Ávila, C., del Pozo, G. B. and Guerra-Casanova, J. [2011]. Unconstrained and contactless hand geometry biometrics, *Sensors* **11**(11): 10143–10164.

Defferrard, M., Bresson, X. and Vandergheynst, P. [2016]. Convolutional neural networks on graphs with fast localized spectral filtering, *Annual Conference on Neural Information Processing Systems (NIPS)*.

Deng, H., Birdal, T. and Ilic, S. [2018]. Ppfnet: Global context aware local features for robust 3d point matching, *Conference on Computer Vision and Pattern Recognition (CVPR)* .

Devlin, J., Chang, M.-W., Lee, K. and Toutanova, K. [2019]. Bert: Pre-training of deep bidirectional transformers for language understanding, *Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT)*.

Dhillon, G. S., Azizzadenesheli, K., Lipton, Z. C., Bernstein, J., Kossaifi, J., Khanna, A. and Anandkumar, A. [2018]. Stochastic activation pruning for robust adversarial defense, *International Conference on Learning Representations (ICLR)* .

Ding, Y., Zhuang, D. and Wang, K. [2005]. A study of hand vein recognition method, *International Conference Mechatronics and Automation*.

Dorai, C. and Jain, A. K. [1997]. Cosmos-a representation scheme for 3d free-form objects, *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* **19**(10): 1115–1130.

Drahanský, M. [2018]. *Hand-Based Biometrics: Methods and Technology*.

Duta, N. [2009]. A survey of biometric technology based on hand shape, *ACM Pattern Recognition* **42**(11): 2797–2806.

Duta, N., Jain, A. K. and Mardia, K. V. [2002]. Matching of palmprints, *ACM Pattern Recognition Letters* **23**(4): 477–485.

Duvenaud, D. K., Maclaurin, D., Iparraguirre, J., Bombarell, R., Hirzel, T., Aspuru-Guzik, A. and Adams, R. P. [2015]. Convolutional networks on graphs for learning molecular fingerprints, *Annual Conference on Neural Information Processing Systems (NIPS)*.

Egan, J. P. [1975]. *Signal Detection Theory and ROC Analysis*, Series in Cognition and Perception, Academic Press.

Eldar, Y. [1992]. Irregular image sampling using the voronoi diagram.

Eldar, Y., Lindenbaum, M., Porat, M. and Zeevi, Y. Y. [1997]. The farthest point strategy for progressive image sampling, *IEEE Transactions on Image Processing* **6**(9): 1305–1315.

Fakhar, K., El Aroussi, M., Nabil Saidi, M. and Aboutajdine, D. [2016]. Fuzzy pattern recognition-based approach to biometric score fusion problem, *Fuzzy Sets and Systems* **315**: 149–159.

Fawzi, A., Fawzi, O. and Frossard, P. [2018]. Analysis of classifiers' robustness to adversarial perturbations, *Machine Learning* **107**: 481–508.

Fei, L., Lu, G., Jia, W., Teng, S. and Zhang, D. [2019a]. Feature extraction methods for palmprint recognition: A survey and evaluation, *IEEE Transactions on Systems, Man, and Cybernetics: Systems* **49**(2): 346–363.

Fei, L., Zhang, B., Xu, Y., Guo, Z., Wen, J. and Jia, W. [2019b]. Learning discriminant direction binary palmprint descriptor, *IEEE Transactions on Image Processing* **28**(8): 3808–3820.

Feng, J. [2008]. Combining minutiae descriptors for fingerprint matching, *Pattern Recognition* **41**(1): 342–352.

Feng, J., Zhou, J. and Jain, A. K. [2013]. Orientation field estimation for latent fingerprint enhancement, *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* **35**(4): 925–940.

Ferrara, M., Maltoni, D. and Cappelli, R. [2012]. Noninvertible minutia cylinder-code representation, *IEEE Transactions on Information Forensics and Security* **7**(6): 1727–1737.

Ferrara, M., Maltoni, D. and Cappelli, R. [2014]. A two-factor protection scheme for mcc fingerprint templates, *International Conference of the Biometrics Special Interest Group (BIOSIG)*.

Fierrez, J., Morales, A., Vera-Rodriguez, R. and Camacho, D. [2018]. Multiple classifiers in biometrics. part 1: Fundamentals and review, *Information Fusion* **44**: 57–64.

Frasconi, P., Gori, M. and Sperduti, A. [1998]. A general framework for adaptive processing of data structures, *IEEE Transactions on Neural Networks* **9**(5): 768–786.

Fukushima, K. [1980]. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position, *Biological Cybernetics* **36**(4): 193–202.

Gainza, P., Sverrisson, F., Monti, F., Rodolà, E., Boscaini, D., Bronstein, M. M. and Correia, B. E. [2019]. Deciphering interaction fingerprints from protein molecular surfaces using geometric deep learning, *Nature Methods* **17**: 184–192.

Galton, F. [1892]. *Finger Prints*, number 57490-57492, Macmillan and Company.

Genovese, A., Piuri, V., Plataniotis, K. N. and Scotti, F. [2019]. Palmnet: Gabor-pca convolutional networks for touchless palmprint recognition, *IEEE Transactions on Information Forensics and Security* **14**(12): 3160–3174.

Genovese, A., Piuri, V. and Scotti, F. [2014]. *Touchless Palmprint Recognition Systems*, Advances in Information Security, Springer International Publishing.

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O. and Dahl, G. E. [2017]. Neural message passing for quantum chemistry, *International Conference on Machine Learning (ICML)* .

Goller, C. and Küchler, A. [1995]. Learning Task-Dependent Distributed Representations by Backpropagation Through Structure, *Ar-report*, Institut für Informatik, Technische Universität München.

Goodfellow, I., Bengio, Y. and Courville, A. [2016]. *Deep Learning*, MIT Press. `http://www.deeplearningbook.org`.

Goodfellow, I., Shlens, J. and Szegedy, C. [2015]. Explaining and harnessing adversarial examples, *International Conference on Learning Representations (ICLR)*.

Gori, M., Monfardini, G. and Scarselli, F. [2005]. A new model for learning in graph domains, *International Joint Conference on Neural Networks (IJCNN)*.

Guo., B. H., Nixon, M. S. and Carter, J. N. [2019]. Soft biometric fusion for subject recognition at a distance, *IEEE Transactions on Biometrics, Behavior, and Identity Science* **1**(4): 292–301.

Guo, J., Wang, H., Cheng, Z., Zhang, X. and Yan, D.-M. [2020]. Learning local shape descriptors for computing non-rigid dense correspondence, *Computational Visual Media* **6**: 95–112.

Hadsell, R., Chopra, S. and LeCun, Y. [2006]. Dimensionality reduction by learning an invariant mapping, Conference on Computer Vision and Pattern Recognition (CVPR).

Hamilton, W., Ying, Z. and Leskovec, J. [2017]. Inductive representation learning on large graphs, *Annual Conference on Neural Information Processing Systems (NIPS)*.

He, K., Zhang, X., Ren, S. and Sun, J. [2015]. Deep residual learning for image recognition, *International Conference on Computer Vision (ICCV)* .

Henaff, M., Bruna, J. and LeCun, Y. [2015]. Deep convolutional networks on graph-structured data, *arXiv preprint arXiv:1506.05163* .

Hershey, J. R., Chen, Z., Le Roux, J. and Watanabe, S. [2016]. Deep clustering: Discriminative embeddings for segmentation and separation, *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Hjelm, R. D., Fedorov, A., Lavoie-Marchildon, S., Grewal, K., Bachman, P., Trischler, A. and Bengio, Y. [2018]. Learning deep representations by mutual information estimation and maximization, *arXiv preprint arXiv:1808.06670* .

Hoffer, E. and Ailon, N. [2015]. Deep metric learning using triplet network, Springer International Publishing.

Holmström, L. [1989]. *Classification of unaveraged evoked cortical magnetic fields*, Research reports / Rolf Nevanlinna Institute. A.

Hornik, K., Stinchcombe, M. and White, H. [1989]. Multilayer feedforward networks are universal approximators, *Neural Networks* **2**(5): 359 – 366.

Hu, G., Yang, Y., Yi, D., Kittler, J., Christmas, W. J., Li, S. Z. and Hospedales, T. M. [2015]. When face recognition meets with deep learning: an evaluation of convolutional neural networks for face recognition, *Conference on Computer Vision and Pattern Recognition (CVPR)* .

Hu, M. K. [1962]. Visual Pattern Recognition by Moment Invariants, *IRE Transactions on Information Theory* **8**(2): 179–187.

Huang, D.-S., Jia, W. and Zhang, D. [2008]. Palmprint verification based on principal lines, *ACM Pattern Recognition* **41**(4): 1316–1328.

Huang, D., Tang, Y., Wang, Y., Chen, L. and Wang, Y. [2014]. Hand-dorsa vein recognition by matching local features of multisource keypoints, *IEEE Transactions on Cybernetics* **45**(9): 1823–1837.

Huang, Y. S. and Suen, C. Y. [1995]. A method of combining multiple experts for the recognition of unconstrained handwritten numerals, *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* **17**(1): 90–94.

Intel [2015]. RealSense, `http://www.intel.com/realsense`.

Ioffe, S. and Szegedy, C. [2015]. Batch normalization: Accelerating deep network training by reducing internal covariate shift, *International Conference on Machine Learning (ICML)* .

Ivakhnenko, A. G. [1971]. Polynomial theory of complex systems, *IEEE Transactions on Systems, Man and Cybernetics* **SMC-1**(4): 364–378.

Ivakhnenko, A. G. and Lapa, V. G. [1965]. Cybernetic predicting devices, *CCM Information Corporation* .

Izadpanahkakhk, M., Razavi, S. M., Taghipour-Gorjikolaie, M., Zahiri, S. H. and Uncini, A. [2018]. Deep region of interest and feature extraction models for palmprint verification using convolutional neural networks transfer learning, *Applied Sciences* .

Jain, A. K. and Feng, J. [2011]. Latent fingerprint matching, *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* **33**(1): 88–100.

Jain, A. K., Feng, J. and Nandakumar, K. [2010b]. Fingerprint matching, *Computer* **43**(2): 36–44.

Jain, A. K., Flynn, P. and Ross, A. A. [2010a]. *Handbook of Biometrics*, 1st edn, Springer Publishing Company, Incorporated.

Jain, A. K., K., A., Ross, A. and Pankanti, S. [1999]. A prototype hand geometry-based verification system, *International Conference on Audio- and Video-Based Biometric Person Authentication (AVBPA)* .

Jain, A. K., Prabhakar, S., Hong, L. and Pankanti, S. [2000]. Filterbank-based fingerprint matching, *IEEE transactions on Image Processing* **9**(5): 846–859.

Jain, A. K., Ross, A. and Prabhakar, S. [2001]. Fingerprint matching using minutiae and texture features, *International Conference on Image Processing (ICIP)*.

Jalali, A., Mallipeddi, R. and Lee, M. [2015]. Deformation invariant and contactless palmprint recognition using convolutional neural network, *International Conference on Human-Agent Interaction (HAI)*.

Jiang, X. and Yau, W.-Y. [2000]. Fingerprint minutiae matching based on the local and global structures, *International Conference on Pattern Recognition (ICPR)*.

Joshi, I., Anand, A., Vatsa, M., Singh, R., Roy, S. D. and Kalra, P. [2019]. Latent fingerprint enhancement using generative adversarial networks, *Winter Conference on Applications of Computer Vision (WACV)*.

Kanhangad, V., Kumar, A. and Zhang, D. [2009]. Combining 2d and 3d hand geometry features for biometric verification, *Conference on Computer Vision and Pattern Recognition (CVPR)* .

Kanhangad, V., Kumar, A. and Zhang, D. [2011]. Contactless and pose invariant biometric identification using hand surface, *IEEE Transactions on Image Processing* **20**(5): 1415–1424.

Kazi, A., Cosmo, L., Navab, N. and Bronstein, M. M. [2020]. Differentiable graph module (dgm) graph convolutional networks, *arXiv preprint arXiv:2002.04999* .

Khalil, M. S. [2011]. Deducting fingerprint singular points using orientation field reliability, *International Conference on Robot, Vision and Signal Processing (RVSP)*.

Kipf, T. N. and Welling, M. [2017a]. Semi-supervised classification with graph convolutional networks, *International Conference on Learning Representations (ICLR)* .

Kipf, T. N. and Welling, M. [2017b]. Semi-supervised classification with graph convolutional networks.

Kleene, S. C. [1956]. Representation of events in nerve nets and finite automata., *Automata Studies, Princeton University Press* pp. 3–42.

Klette, R., Schluns, K. and Koschan, A. [1998]. *Computer Vision: Three-Dimensional Data from Images*, Springer.

Komarinski, P. [2004]. *Automated Fingerprint Identification Systems (AFIS)*, Academic Press.

Kong, A. W. K. and Zhang, D. [2004]. Competitive coding scheme for palmprint verification, *International Conference on Pattern Recognition (ICPR)*.

Kong, A., Zhang, D. and Kamel, M. [2009]. A survey of palmprint recognition, *ACM Pattern Recognition* **42**(7): 1408–1418.

Krizhevsky, A. [2009]. Learning multiple layers of features from tiny images.

Krizhevsky, A., Sutskever, I. and Hinton, G. E. [2012]. Imagenet classification with deep convolutional neural networks, Annual Conference on Neural Information Processing Systems (NIPS).

Küchler, A. and Goller, C. [1996]. Inductive learning in symbolic domains using structure-driven recurrent neural networks, *Annual German Conference on Artificial Intelligence: Advances in Artificial Intelligence*.

Kulon, D., Güler, R. A., Kokkinos, I., Bronstein, M. M. and Zafeiriou, S. [2020]. Weakly-supervised mesh-convolutional hand reconstruction in the wild, *Conference on Computer Vision and Pattern Recognition (CVPR)* .

Kulon, D., Wang, H., Güler, R. A., Bronstein, M. M. and Zafeiriou, S. [2019]. Single image 3d hand reconstruction with mesh convolutions, *British Machine Vision Conference (BMVC)* .

Kumar, A. [2008]. Incorporating cohort information for reliable palmprint authentication, *Indian Conference on Computer Vision, Graphics Image Processing*.

Kumar, A. and Shekhar, S. [2011]. Personal identification using multibiometrics rank-level fusion, *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* **41**(5): 743–752.

Kumar, A. and Shen, H. C. [2004]. Palmprint identification using palmcodes, *International Conference on Image and Graphics (ICIG)*.

Kumar, A., Wong, D. C., Shen, H. C. and Jain, A. K. [2003]. Personal verification using palmprint and hand geometry biometric, *International Conference on Audio-and Video-Based Biometric Person Authentication*, Springer.

Kuncheva, L. I. [2004]. *Combining Pattern Classifiers: Methods and Algorithms*, Wiley-Interscience, New York, NY, USA.

Kurakin, A., Goodfellow, I. J., Bengio, S., Dong, Y., Liao, F., Liang, M., Pang, T., Zhu, J., X., H., Xie, C., Wang, J., Zhang, Z., Ren, Z., Yuille, A. L., Huang, S., Zhao, Y., Zhao, Y., Han, Z., Long, J., Berdibekov, Y., Akiba, T., Tokui, S. and Abe, M. [2018]. Adversarial attacks and defences competition, *Annual Conference on Neural Information Processing Systems (NIPS)* .

Laadjel, M., Bouridane, A., Kurugollu, F., Nibouche, O. and Yan, W. [2010]. *Transactions on Data Hiding and Multimedia Security*, Springer, chapter Partial Palmprint Matching Using Invariant Local Minutiae Descriptors.

Lam, L. and Suen, S. Y. [1997]. Application of majority voting to pattern recognition: an analysis of its behavior and performance, *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* **27**(5): 553–568.

LeCun, Y. [1998a]. The MNIST Database of Handwritten Digits, `http://yann.lecun.com/exdb/mnist/`.

LeCun, Y., Bottou, L., Bengio, Y. and Haffner, P. [1998b]. Gradient-based learning applied to document recognition, *Proceedings of the IEEE* **86**(11): 2278–2324.

Lee, H. C. and Gaensslen, R. E. [2001]. *Advances in Fingerprint Technology*, Elsevier.

Lee, H. C., Ramotowski, R. and Gaensslen, R. E. [2001]. *Advances in Fingerprint Technology*, CRC Press.

Levie, R., Monti, F., Bresson, X. and Bronstein, M. M. [2018]. Cayleynets: Graph convolutional neural networks with complex rational spectral filters, *IEEE Transactions on Signal Processing* **67**(1): 97–109.

Li, J., Feng, J. and Jay Kuo, C.-C. [2018]. Deep convolutional neural network for latent fingerprint enhancement, *Signal Processing: Image Communication* **60**: 52 – 63.

Li, Y., Tarlow, D., Brockschmidt, M. and Zemel, R. [2016]. Gated graph sequence neural networks, *International Conference on Learning Representations (ICLR)*.

Lin, C. and Kumar, A. [2018]. Contactless and partial 3d fingerprint recognition using multi-view deep representation, *Pattern Recognition* **83**: 314 – 327.

Linnainmaa, S. [1976]. Taylor expansion of the accumulated rounding error, *BIT Numerical Mathematics* **16**: 146–160.

Liu, E., Jain, A. K. and Tian, J. [2013]. A coarse to fine minutiae-based latent palmprint matching, *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* **35**(10): 2307–2322.

Lu, J., Zhao, Y. and Hu, J. [2009]. Enhanced Gabor-based region covariance matrices for palmprint recognition, *Electronics Letters* **45**(17): 880–881.

Lu, X., Deng, Z. and Chen, W. [2016]. A robust scheme for feature-preserving mesh denoising, *IEEE Transactions on Visualization and Computer Graphics* **22**(3): 1181–1194.

Maas, A., Hannun, A. and Ng, A. [2013]. Rectifier nonlinearities improve neural network acoustic models, *International Conference on Machine Learning (ICML)*.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D. and Vladu, A. [2017]. Towards deep learning models resistant to adversarial attacks, *International Conference on Learning Representations (ICLR)* .

Mahdizadehaghdam, S., Panahi, A., Krim, H. and Dai, L. [2018]. Deep dictionary learning: A parametric network approach, *IEEE Transactions on Image Processing* **28**(10): 4790–4802.

Malassiotis, S., Aifanti, M. and Strintzis, M. G. [2006]. Personal authentication using 3-d finger geometry, *IEEE Transactions on Information Forensics and Security* **1**(1): 12–21.

Maltoni, D., Maio, D., Jain, A. K. and Prabhakar, S. [2009]. *Handbook of Fingerprint Recognition*, 2nd edn, Springer Publishing Company, Incorporated.

Martin, A., Doddington, G., Kamm, T., Ordowski, M. and Przybocki, M. [1997]. The det curve in assessment of detection task performance, *European Conference on Speech Communication and Technology (EUROSPEECH)*.

Masci, J., Meier, U., Cireşan, D. and Schmidhuber, J. [2011]. *Stacked Convolutional Auto-Encoders for Hierarchical Feature Extraction*.

Maturana, D. and Scherer, S. [2015]. Voxnet: A 3d convolutional neural network for real-time object recognition, *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

McCulloch, W. S. and Pitts, W. [1943]. A logical calculus of ideas immanent in nervous activity, *Bulletin of Mathematical Biophysics* **5**: 115–133.

Meng, D. and Chen, H. [2017]. Magnet: a two-pronged defense against adversarial examples, *ACM SIGSAC Conference on Computer and Communications Security (CCS)* .

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. and Dean, J. [2013]. Distributed representations of words and phrases and their compositionality, *Annual Conference on Neural Information Processing Systems (NIPS)* .

Minaee, S. and Wang, Y. [2016]. Palmprint recognition using deep scattering convolutional network, *International Symposium on Circuits and Systems (ISCAS)* .

Mirmohamadsadeghi, L. and Drygajlo, A. [2011]. Palm vein recognition with local binary patterns and local derivative patterns, *International Joint Conference on Biometrics (IJCB)*.

Moenssens, A. [1971]. *Fingerprint Techniques*, Chilton Book Company.

Monti, F., Boscaini, D., Masci, J., Rodolà, E., Svoboda, J. and Bronstein, M. M. [2017a]. Geometric deep learning on graphs and manifolds using mixture model CNNs, *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Monti, F., Bronstein, M. M. and Bresson, X. [2017b]. Geometric matrix completion with recurrent multi-graph neural networks, *Annual Conference on Neural Information Processing Systems (NIPS)*.

Monti, F., Otness, K. and Bronstein, M. M. [2018]. Motifnet: a motif-based graph convolutional network for directed graphs, *IEEE Data Science Workshop (DSW)* .

Moosavi-Dezfooli, S., Fawzi, A., Fawzi, O. and Frossard, P. [2017]. Universal adversarial perturbations, *Conference on Computer Vision and Pattern Recognition (CVPR)* .

Moosavi-Dezfooli, S., Fawzi, A. and Frossard, P. [2016]. Deepfool: a simple and accurate method to fool deep neural networks, *Conference on Computer Vision and Pattern Recognition (CVPR)* .

Mráček, S., Dvořák, R., Váňa, J., Novotný, T. and Drahanský, M. [2015]. 3d face recognition utilizing a low-cost depth sensor, *International Conference on Automatic Face and Gesture Recognition (FG)*.

Nair, V. and Hinton, G. E. [2010]. Rectified linear units improve restricted boltzmann machines, *Proceedings of the 27th International Conference on International Conference on Machine Learning*, International Conference on Machine Learning (ICML).

NeuroTechnology [1998]. Verifinger. www.neurotechnology.com/verifinger.html.

Nguyen, D., Cao, K. and Jain, A. K. [2018]. Robust minutiae extractor: Integrating deep networks and fingerprint domain knowledge, *International Conference on Biometrics (ICB)*.

Nibouche, O. and Jiang, J. [2013]. Palmprint matching using feature points and {SVD} factorisation, *Digital Signal Processing* **23**(4): 1154–1162.

NIST [2007]. NBIS (NIST Biometric Image Software). http://www.nist.gov/itl/iad/ig/nbis.cfm.

Qi, C. R., Su, H., Mo, K. and Guibas, L. J. [2016]. Pointnet: Deep learning on point sets for 3d classification and segmentation, *Conference on Computer Vision and Pattern Recognition (CVPR)* .

Qi, C. R., Yi, L., Su, H. and Guibas, L. J. [2017]. Pointnet++: Deep hierarchical feature learning on point sets in a metric space, *Annual Conference on Neural Information Processing Systems (NIPS)* .

Radford, A., Metz, L. and Chintala, S. [2015]. Unsupervised representation learning with deep convolutional generative adversarial networks, *arXiv preprint arXiv:1511.06434* .

Raghavendra, R. and Busch, C. [2014]. Robust palmprint verification using sparse representation of binarized statistical features: A comprehensive study, *Workshop on Information Hiding and Multimedia Security (IHMMSEC)*.

Ranjan, A., Bolkart, T., Sanyal, S. and Black, M. J. [2018]. Generating 3d faces using convolutional mesh autoencoders, *European Conference on Computer Vision (ECCV)* .

Rauber, J., Brendel, W. and Bethge, M. [2017]. Foolbox v0.8.0: A python toolbox to benchmark the robustness of machine learning models, *arXiv preprint arXiv:1707.04131* .

Ribaric, S. and Marcetic, M. [2012]. Personal recognition based on the gabor features of colour palmprint images, *International Convention on Information, Communication and Electronic Technology (MIPRO)*.

Romero, J., Tzionas, D. and Black, M. J. [2017]. Embodied hands: Modeling and capturing hands and bodies together, *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)* **36**(6).

Rosenblatt, F. [1958]. The perceptron: A probabilistic model for information storage and organization in the brain, *Psychological Review* pp. 65–386.

Ross, A. A. and Govindarajan, R. [2005]. Feature level fusion of hand and face biometrics, International Society for Optics and Photonics.

Ross, A. A., Nandakumar, K. and Jain, A. K. [2006]. *Handbook of Multibiometrics*, Springer.

Ross, A. A., Nandakumar, K. and Jain, A. K. [2011]. *Handbook of Multibiometrics*, 1st edn, Springer Publishing Company, Incorporated.

Ross, A., Jain, A. K. and Pankati, S. [1999]. A prototype hand geometry-based verification system, *International Conference on Audio-and Video-Based Biometric Person Authentication*.

Sanchez-Reillo, R., Sanchez-Avila, C. and Gonzalez-Marcos, A. [2000]. Biometric identification through hand geometry measurements, *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* .

Sankaran, A., Dhamecha, T. I., Vatsa, M. and Singh, R. [2011]. On matching latent to latent fingerprints, *International Joint Conference on Biometrics (IJCB)*.

Sankaran, A., Vatsa, M. and Singh, R. [2015]. Multisensor optical and latent fingerprint database, *IEEE Access* **3**: 653–665.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M. and Monfardini, G. [2009]. The graph neural network model, *IEEE Transactions on Neural Networks and Learning Systems* **20**(1): 61–80.

Schmidhuber, J. [2015]. Deep learning in neural networks: An overview, *Neural Networks* **61**: 85 – 117.

Schuch, P., Schulz, S. and Busch, C. [2016]. De-convolutional auto-encoder for enhancement of fingerprint samples, *International Conference on Image Processing Theory, Tools and Applications (IPTA)*.

Schuch, P., Schulz, S. and Busch, C. [2017]. Minutia-based enhancement of fingerprint samples, International Carnahan Conference on Security Technology (ICCST).

Shen, Y., Feng, C., Yang, Y. and Tian, D. [2017]. Neighbors do help: Deeply exploiting local structures of point clouds, *arXiv preprint arXiv:1712.06760* .

Shu, W. and Zhang, D. [1998]. Automated personal identification by palmprint, *Optical Engineering* **37**(8): 2359–2362.

Sidlauskas, D. P. [1988]. 3d hand profile identification apparatus, `https://www.google.com/patents/US4736203`. US Patent 4,736,203.

Sperduti, A. and Starita, A. [1997]. Supervised neural networks for the classification of structures, *IEEE Transactions on Neural Networks* **8**(3): 714–735.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R. [2014]. Dropout: A simple way to prevent neural networks from overfitting, *Journal of Machine Learning Research* **15**(56): 1929–1958.

StemmerImaging [2020]. Time-Of-Flight camera, `https://www.stemmer-imaging.com/fr-fr/donnees/cameras-3d-time-of-flight-cameras/`.

Su, H., Chen, K., Wong, W. J. and Lai, S. [2017]. A deep learning approach towards pore extraction for high-resolution fingerprint recognition, *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*.

Su, J., Vasconcellos Vargas, D. and Sakurai, K. [2019]. One pixel attack for fooling deep neural networks, *IEEE Transactions on Evolutionary Computation* **23**(5): 828–841.

Sun, X., Rosin, P., Martin, R. and Langbein, F. [2007]. Fast and effective feature-preserving mesh denoising, *IEEE Transactions on Visualization and Computer Graphics* **13**(5): 925–938.

Sun, Y., Wang, X. and Tang, X. [2014]. Deep learning face representation from predicting 10,000 classes, *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Sun, Z., Tan, T., Wang, Y. and Li, S. Z. [2005]. Ordinal palmprint representation for personal identification, *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Svoboda, J., Anoosheh, A., Osendorfer, C. and Masci, J. [2020a]. Two-Stage Peer-Regularized Feature Recombination for Arbitrary Image Style Transfer, *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Svoboda, J., Astolfi, P., Boscaini, D., Masci, J. and Bronstein, M. M. [2020b]. Clustered Dynamic Graph CNN for Biometric 3D Hand Shape Recognition, *To be published soon.*

Svoboda, J., Bronstein, M. M. and Drahansky, M. [2015]. Contactless biometric hand geometry recognition using a low-cost 3d camera, *International Conference on Biometrics (ICB)*.

Svoboda, J., Masci, J. and Bronstein, M. M. [2016]. Palmprint recognition via discriminative index learning, *International Conference on Pattern Recognition (ICPR)*.

Svoboda, J., Masci, J., Monti, F., Bronstein, M. M. and Guibas, L. J. [2019]. PeerNets: Exploiting Peer Wisdom Against Adversarial Attacks, *International Conference on Learning Representations (ICLR)*.

Svoboda, J., Monti, F. and Bronstein, M. M. [2017]. Generative convolutional networks for latent fingerprint reconstruction, *International Joint Conference on Biometrics (IJCB)*.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. and Fergus, R. [2014]. Intriguing properties of neural networks, *International Conference on Learning Representations (ICLR)* .

Taigman, Y., Yang, M., Ranzato, M. and Wolf, L. [2014]. Deepface: Closing the gap to human-level performance in face verification, *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Tang, D., Taylor, J., Kohli, P., Keskin, C., Kim, T. and Shotton, J. [2015]. Opening the black box: Hierarchical sampling optimization for estimating human hand pose, *International Conference on Computer Vision (ICCV)*.

Tang, Y., Gao, F. and Feng, J. [2017]. Latent fingerprint minutia extraction using fully convolutional network, *International Joint Conference on Biometrics (IJCB)*.

Tang, Y., Gao, F., Feng, J. and Liu, Y. [2017b]. Fingernet: An unified deep network for fingerprint minutiae extraction, *International Joint Conference on Biometrics (IJCB)*.

Thakku, S. G., Tham, Y.-C., Baskaran, M., Mari, J.-M., Strouthidis, N. G., Aung, T., Cheng, C.-Y. and Girard, M. J. A. [2015]. A global shape index to characterize anterior lamina cribrosa morphology and its determinants in healthy indian eyeslc-gsi to characterize lc morphology, *Investigative Ophthalmology & Visual Science* **56**(6): 3604–3614.

Tian, F., Gao, B., Cui, Q., Chen, E. and Liu, T.-Y. [2014]. Learning deep representations for graph clustering, *AAAI Conference on Artificial Intelligence*.

Tico, M. and Kuosmanen, P. [2003]. Fingerprint matching using an orientation-based minutia descriptor, *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* **25**(8): 1009–1014.

Torralba, A., Fergus, R. and Freeman, W. T. [2008]. 80 million tiny images: A large data set for nonparametric object and scene recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* .

Tramèr, F., Kurakin, A., Papernot, N., Goodfellow, I., Boneh, D. and McDaniel, P. [2018]. Ensemble adversarial training: Attacks and defenses, *International Conference on Learning Representations (ICLR)*.

Tretschk, E., Tewari, A., Zollhöfer, M., Golyanik, V. and Theobalt, C. [2019]. DEMEA: deep mesh autoencoders for non-rigidly deforming objects, *arXiv preprint arXiv:1905.10290* .

Tzionas, D., Ballan, L., Srikantha, A., Aponte, P., Pollefeys, M. and Gall, J. [2016]. Capturing hands in action using discriminative salient points and physics simulation, *International Journal of Computer Vision (IJCV)* .

Ungureanu, A. S., Salahuddin, S. and Corcoran, P. [2020]. Toward unconstrained palmprint recognition on consumer devices: A literature review, *IEEE Access* **8**: 86130–86148.

van den Oord, A., Li, Y. and Vinyals, O. [2018]. Representation learning with contrastive predictive coding, *arXiv preprint arXiv:1807.03748* .

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P. and Bengio, Y. [2018]. Graph attention networks, *International Conference on Learning Representations (ICLR)* .

Vijay Kumar, B. G., Carneiro, G. and Reid, I. D. [2016]. Learning local image descriptors with deep siamese and triplet convolutional networks by minimising global loss functions, *Conference on Computer Vision and Pattern Recognition (CVPR)* .

Waibel, A. [1987]. Phoneme recognition using time-delay neural networks, *Meeting of Institute of Electronics, Information and Communication Engineers (IEICE)* .

Wan, C., Wang, L. and Phoha, V. V. [2018]. A survey on gait recognition, *ACM Comput. Surv.* **51**(5).

Wang, C., Liu, H. and Liu, X. [2014a]. Contact-free and pose-invariant hand-biometric-based personal identification system using rgb and depth data, *Journal of Zhejiang University SCIENCE C* **15**: 525–536.

Wang, G., Kang, W., Wu, Q., Wang, Z. and Gao, J. [2018]. Generative adversarial network (gan) based data augmentation for palmprint recognition, *Digital Image Computing: Techniques and Applications (DICTA)*.

Wang, H., Guo, J., Yan, D.-M., Quan, W. and Zhang, X. [2018a]. Learning 3D keypoint descriptors for non-rigid shape matching, *European Conference on Computer Vision (ECCV)*.

Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, J., Philbin, J., Chen, B. and Wu, Y. [2014b]. Learning fine-grained image similarity with deep ranking, *Conference on Computer Vision and Pattern Recognition (CVPR)* .

Wang, K., Zhang, Y., Yuan, Z. and Zhuang, D. [2006a]. Hand vein recognition based on multi supplemental features of multi-classifier fusion decision, *International Conference on Mechatronics and Automation*.

Wang, R., Han, C., Wu, Y. and Guo, T. [2014c]. Fingerprint classification based on depth neural network, *arXiv preprint arXiv:1409.5188* .

Wang, X., L., L. and Wang, M. [2012]. Palmprint verification based on 2d – gabor wavelet and pulse-coupled neural network, *Knowledge-Based Systems* **27**: 451–455.

Wang, X., Liang, J. and Wang, M. [2013]. On-line fast palmprint identification based on adaptive lifting wavelet scheme, *Knowledge-Based Systems* **42**: 68–73.

Wang, Y. and Ruan, Q. [2006b]. Kernel Fisher discriminant analysis for palmprint recognition, *International Conference on Pattern Recognition (ICPR)*.

Wang, Y., Sun, Y., Liu, Z., Sarma, S. E., Bronstein, M. M. and Solomon, J. M. [2019]. Dynamic graph CNN for learning on point clouds, *ACM Transactions on Graphics (TOG)* **38**(5).

Weng, J., Ahuja, N. and Huang, T. S. [1993]. Learning recognition and segmentation of 3-d objects from 2-d images, *International Conference on Computer Vision (ICCV)* .

Werbos, P. J. [1982]. Applications of advances in nonlinear sensitivity analysis, *System Modeling and Optimization*, pp. 762–770.

Woodard, D. L. and Flynn, P. J. [2005]. Finger surface as a biometric identifier, *Computer Vision and Image Understanding (CVIU)* **100**(3): 357–384.

Wu, X., Wang, K. and Zhang, D. [2004]. HMMs based palmprint identification, *International Conference on Biometric Authentication (ICBA)* .

Xia, X. and O'Gorman, L. [2003]. Innovations in fingerprint capture devices, *ACM Pattern Recognition* **36**: 361–369.

Xie, J., Girshick, R. B. and Farhadi, A. [2015]. Unsupervised deep embedding for clustering analysis, *International Conference on Machine Learning (ICML)* .

Xu, L., Krzyzak, A. and Suen, C. Y. [1992]. Methods of combining multiple classifiers and their applications to handwriting recognition, *IEEE Transactions on Systems, Man, and Cybernetics* **22**(3): 418–435.

Yang, W. l. and Wang, L. l. [2010]. Research of palmprint identification method using Zernike moment and neural network, *International Conference on Computing, Networking and Communications (ICNC)*.

Yi, D., Lei, Z., Liao, S. and Li, S. Z. [2014]. Learning face representation from scratch, *arXiv preprint arXiv:1411.7923* .

Ying, R., You, J., Morris, C., Ren, X., Hamilton, W. L. and Leskovec, J. [2018]. Hierarchical graph representation learning with differentiable pooling, *Annual Conference on Neural Information Processing Systems (NIPS)* .

Yoon, S., Feng, J. and Jain, A. K. [2010]. On latent fingerprint enhancement, *SPIE Biometric Technology for Human Identification VII* .

Yoon, S., Feng, J. and Jain, A. K. [2011]. Latent fingerprint enhancement via robust orientation field estimation, *International Joint Conference on Biometrics (IJCB)* .

Young, T., Hazarika, D., Poria, S. and Cambria, E. [2017]. Recent trends in deep learning based natural language processing, *IEEE Computational Intelligence Magazine* **13**(3): 55–75.

Yuan, X., He, P., Zhu, Q., Rana Bhat, R. and Li, X. [2019]. Adversarial examples: Attacks and defenses for deep learning, *IEEE Transactions on Neural Networks and Learning Systems* **30**(9): 2805–2824.

Zabatani, A., Surazhsky, V., Sperling, E., Ben Moshe, S., Menashe, O., Silver, D. H., Karni, T., Bronstein, A. M., Bronstein, M. M. and Kimmel, R. [2019]. Intel realsense sr300 coded light depth camera, *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* .

Zantedeschi, V., Nicolae, M. and Rawat, A. [2017]. Efficient defenses against adversarial attacks, *ACM Workshop on Artificial Intelligence and Security* .

Zeiler, M. D. and Fergus, R. [2014]. Visualizing and understanding convolutional networks, *European Conference on Computer Vision (ECCV)* .

Zhang, D., Kong, W.-K., You, J. and Wong, M. [2003]. Online palmprint identification, *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* **25**(9): 1041–1050.

Zhang, Z. [1994]. Iterative point matching for registration of free-form curves and surfaces, *International Journal of Computer Vision* **13**(2): 119–152.

Zhong, D., Shao, H. and Du, X. [2019]. A hand-based multi-biometrics via deep hashing network and biometric graph matching, *IEEE Transactions on Information Forensics and Security* **14**(12): 3140–3150.

Zhou, X., Wan, Q., Zhang, W., Xue, X. and Wei, Y. [2016]. Model-based deep hand pose estimation, *arXiv preprint arXiv:1606.06854* .

Zhu, J.-Y., Park, T., Isola, P. and Efros, A. A. [2017]. Unpaired image-to-image translation using cycle-consistent adversarial networks, *International Conference on Computer Vision (ICCV)* .

Zimmermann, C. and Brox, T. [2017]. Learning to estimate 3d hand pose from single RGB images, *International Conference on Computer Vision (ICCV)* .

Zunkel, R. L. [1996]. Hand geometry based verification, *Biometrics: Personal Identification in Networked Society*, Springer, pp. 87–101.