

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Fingerprint Anti Spoofing – Domain Adaptation and Adversarial Learning

João Afonso Pinto Pereira

DISSERTATION FOR THE DEGREE OF MASTER IN BIOENGINEERING



MSc in Bioengineering - Biomedical Engineering

Supervisor: Jaime dos Santos Cardoso

Co-supervisor: Ana Filipa Sequeira

June 28, 2020

Fingerprint Anti Spoofing – Domain Adaptation and Adversarial Learning

João Afonso Pinto Pereira

MSc in Bioengineering - Biomedical Engineering

June 28, 2020

Resumo

A impressão digital é a característica biométrica usada há mais tempo como forma de identificação. Com a crescente implementação de sensores biométricos em dispositivos móveis, a sua utilização está a tornar-se um hábito no dia a dia de milhões de pessoas. Cada vez mais se controla acções que envolvem informação sensível e pessoal através da identificação correta dos indivíduos com as impressões digitais. Este crescimento na utilização de sistemas baseados em impressão digital traz preocupações acerca da segurança dos mesmos, que podem ser facilmente enganados com uma grande variedade de materiais e técnicas simples que estão em constante evolução. Desta forma, diversos métodos anti-falsificação de impressões digitais têm sido implementados como um esforço para solucionar este problema, com a obtenção de desempenhos significativamente bons. No entanto, grande parte das soluções considera que é conhecido antecipadamente que tipo de material será usado pelos atacantes, o que não corresponde ao que acontece numa aplicação real, em que o sistema estará vulnerável a diversos ataques desconhecidos.

Esta dissertação teve como objetivo estudar e desenvolver possíveis abordagens para a detecção de falsificações de impressões digitais, com foco no uso de métodos de *deep learning*, ou seja, na aplicação de Redes Neurais Artificiais. Como ponto de partida consolidou-se o trabalho preliminar desenvolvido anteriormente com classificadores convencionais de *machine learning*. As características extraídas manualmente das imagens foram o histograma da intensidade, da *Local Binary Patterns* e da *Local Phase Quantization*. O classificador que acabou por ser usado como base de comparação foi o *Support Vector Machines* (SVM) devido à sua superioridade em relação aos restantes. A partir daqui o foco tornou-se o *deep learning*, começando com a aplicação de um *Multi-layer Perceptron* (MLP) que usa as mesmas características texturais que o SVM. Esse modelo foi posteriormente regularizado com técnicas de transferência de aprendizagem e de treino adversarial, de forma a melhorar a capacidade de detetar ataques não observados na fase de treino (ataques desconhecidos). Caminhou-se de seguida para as soluções *end-to-end* através das Redes Neurais Convolucionais, otimizando-se a sua performance e testando a mesma regularização que foi aplicada ao MLP, o que levou à submissão de um artigo científico para a *19th International Conference of the Biometrics Special Interest Group* (BIOSIG 2020). Por fim, foi estudado o potencial das Redes Generativas Adversariais para esta aplicação, incluindo diferentes experiências que resultaram numa abordagem inovadora e promissora. A solução com a estratégia de treino por fases ("*Image-to-image GAN with phased training strategy*") contribui com um avanço científico na detecção de falsificações de impressões digitais, tendo capacidade de generalização melhorada para lidar com tipos desconhecidos de materiais de ataque. Ainda estão a ser realizadas experiências de consolidação desta estratégia e está em preparação uma segunda submissão de um artigo científico.

Abstract

The fingerprint is the oldest biometric trait used as a form of identification. With the increasing implementation of biometric sensors in mobile devices, its use is becoming a habit in the everyday life of millions of people, and operations involving sensitive and personal information are becoming constrained by the correct identification of the individuals through their fingerprint. This growth has brought concerns about the security of the available systems, which can be easily deceived with a large variety of simple materials and techniques that are constantly evolving. The detection of the presentation of a fake or altered sample to the sensor was referred to as spoofing attack detection and included liveness detection methods. These later meant to determine if the samples belonged to a live person. Although fingerprint presentation attack detection methods have been achieving very good performances, most solutions consider that it is known in advance what type of materials will be used by the attackers, which does not correspond to what happens in a real environment, where a fingerprint based system is vulnerable to unknown attacks.

This dissertation aimed to intensively study different approaches in fingerprint presentation attack detection, with a focus on the use of deep learning methods, that is, the application of Artificial Neural Networks. As a starting point, the preliminary work previously performed with conventional machine learning classifiers was consolidated. The features extracted manually were the histogram of the intensity, the histogram of the Local Binary Patterns and the histogram of the Local Phase Quantization. The classifier used as a baseline of comparison was the Support Vector Machines (SVM) because of its superiority against the others. From there the focus became deep learning, starting with the development of a Multi-layer Perceptron (MLP) that uses the same handcrafted-features as the SVM. This model was later regularized with transfer learning and adversarial training techniques in an attempt to increase robustness against presentation attacks not observed in the training phase (unknown/unseen-attacks). The next step was the development of end-to-end solutions through Convolutional Neural Networks, optimizing their performance and testing the same regularization that was applied to the MLP, which led to the submission of a paper for the 19th International Conference of the Biometrics Special Interest Group (BIOSIG 2020). Finally, the potential of Generative Adversarial Networks for this application was studied, comprising different experiments which resulted in an innovative and promising approach. The "Image-to-image GAN with phased training strategy" contributes with a scientific advance in fingerprint presentation attack detection specifically by developing models with improved presentation attack detection (PAD) generalization capacity to deal with unknown types of attack materials. There are more experiments on-going related to this strategy and a submission to a journal is in preparation

Agradecimentos

Antes de tudo, gostaria de deixar umas fortes palavras de agradecimento ao professor Jaime Cardoso e à professora Ana Filipa Sequeira por todo o seu apoio, hospitalidade, simpatia e ensinamentos que me deram ao longo deste ano. Posso, com toda a confiança, dizer que este foi um dos anos mais importantes no meu desenvolvimento como estudante, como profissional e como pessoa. Ao mesmo tempo deixo uma enorme palavra de apreço a todo o grupo, que me recebeu de braços abertos, e, em particular, gostaria de deixar um abraço ao Diogo e ao Ricardo, que no meio das suas teses de doutoramento estiveram sempre disponíveis e sem eles esta dissertação não tinha chegado ao que chegou. Que nos encontremos todos num futuro próximo.

Como nunca irei esquecer as minhas origens e a minha cidade, Torres Novas, deixo aqui uma palavra aos meus amigos mais antigos, que mesmo com a distância sempre me puxaram e nunca deixaram que se perdesse a ligação que é tão importante. Ainda por Torres Novas, um grande obrigado à professora Rosário Caldeirão por me ter dado a paixão pelo conhecimento, e à Teresa, por todas as horas de trabalho e os reforços, que me deram um método de trabalho e uma dedicação que nunca mais desapareceram. Sem vocês não estaria onde estou.

A seguir, um enorme abraço a toda a gente que me acompanhou estes cinco anos, neste percurso que nunca sairá da minha memória. Por todas as noites mal dormidas, por todas as conversas aleatórias no corredor da sala de Bio, por todas as almoços e todos os jantares. Termino este percurso com amigos para a vida, com vários colegas de casa e, principalmente, com uma quantidade imensurável de histórias. Que continuemos a viajar juntos, a encher as casas uns dos outros e a passar férias completamente alucinadas. Um obrigado não é suficiente!

Por fim, o maior dos obrigados à minha família pelo apoio incondicional que me dão desde sempre, por nunca desistirem de mim e por me fazerem ser uma pessoa melhor todos os dias. A ti Inês, que acabaste este ano a saber muito mais sobre impressões digitais e inteligência artificial, que me ouves, acalmas e motivas, o meu agradecimento vai ser dar-te de volta tudo o que fazes por mim e muito mais, sempre. Aos meus pais, às minhas irmãs, aos meus tios e aos meus avós, não tenho como vos agradecer. Em especial, ao Pinto e ao Velho, por toda a vossa dedicação incondicional à família, por serem uma fonte de inspiração e duas forças da natureza.

A vocês, Senhor Pinto e Engenheiro Velho, dedico esta dissertação.

João Afonso Pinto Pereira

“Your focus determines your reality.”

Qui-Gon Jinn

Contents

1	Introduction	1
1.1	Context	1
1.2	Motivation	2
1.3	Objectives	3
1.4	Contributions	3
1.5	Document Structure	4
2	Image Processing and Machine Learning Background Knowledge	5
2.1	Image Textural Features	5
2.1.1	Histogram	5
2.1.2	Local Binary Patterns	5
2.1.3	Local Phase Quantization	7
2.2	Machine Learning Classifiers	7
2.2.1	K-Nearest Neighbors	7
2.2.2	Naive Bayes	8
2.2.3	Support Vector Machines	8
2.3	Artificial Neural Networks	9
2.3.1	Multi-layer Perceptrons	11
2.3.2	Convolutional Neural Networks	12
2.3.3	Generative Adversarial Networks	14
3	Fundamental Concepts of Biometric Systems	17
3.1	Biometric Recognition	17
3.1.1	Biometric Modalities	18
3.2	Biometric Systems	19
3.2.1	Architecture of a Biometric System	19
3.2.2	Operating Mode of a Biometric System	19
3.2.3	Vocabulary	20
3.3	Presentation Attack Detection	20
3.3.1	Vocabulary	21
3.4	The Biometric Menagerie	21
3.5	Fingerprint Based Recognition	22
3.5.1	Fingerprint Features	23
3.5.2	Fingerprint Sensing	25
3.5.3	Vulnerabilities of Fingerprint Recognition Systems	27

4	Presentation Attack Detection Methods	31
4.1	Hardware-based Presentation Attack Detection Methods	32
4.2	Software-based Presentation Attack Detection Methods	32
4.2.1	Dynamic Solutions	32
4.2.2	Static Solutions	34
4.3	Fingerprint Presentation Attack Detection Databases	41
4.4	Summary	41
5	Methodology	43
5.1	Evaluation Framework	43
5.2	Pre-processing	45
5.3	Feature Extraction	46
5.4	Handcrafted Features with Machine Learning	46
5.4.1	Support Vector Machines	47
5.5	Handcrafted Features with Deep Learning	48
5.5.1	Multi-layer Perceptron	48
5.5.2	Regularization of the Multi-layer Perceptron	49
5.6	Pure Deep Learning	51
5.6.1	First Convolutional Neural Network	51
5.6.2	VGG Neural Network	53
5.6.3	Second Convolutional Neural Network	56
5.6.4	Regularization of the Second Convolutional Neural Network	57
5.6.5	One-class Generative Adversarial Network	58
5.6.6	Image-to-image Generative Adversarial Network	62
5.7	Summary	65
6	Experimental Work	67
6.1	Datasets	67
6.2	Performance Metrics	68
6.3	Experiments Description	70
6.4	Implementation Details	71
6.5	Results and Discussion	71
6.5.1	Experiment A - SVM vs MLP	72
6.5.2	Experiment B - MLP Regularization (MLP_{reg})	73
6.5.3	Experiment C - First CNN (CNN1)	75
6.5.4	Experiment D - VGG	76
6.5.5	Experiment E - Second CNN (CNN2)	77
6.5.6	Experiment F - VGG vs CNN2	79
6.5.7	Experiment G - CNN2 Regularization ($CNN2_{reg}$)	80
6.5.8	Experiment H - VGG vs $CNN2_{reg}$	82
6.5.9	Experiment I - MLP vs CNN2	83
6.5.10	Experiment J - One-class GAN	85
6.5.11	Experiment K - Image-to-image GAN	87
7	Conclusions and Future Work	91
7.1	Final Remarks	92
7.2	Future Work	92

A	Complete Results	A1
A.1	SVM vs MLP Complete Results	A1
A.2	MLP Regularization (MLP_{reg}) Complete Results	A12
A.3	First CNN (CNN1) Complete Results	A21
A.4	VGG Complete Results	A24
A.5	Second CNN (CNN2) Complete Results	A30
A.6	CNN2 Regularization (CNN2_{reg}) Complete Results	A41
A.7	One-class GAN Complete Results	A47
A.8	Image-to-image GAN Complete Results	A49
B	One-class GAN outputs	B1
C	Image-to-image GAN outputs	C1

List of Figures

2.1	Histogram of a fingerprint image	6
2.2	Histogram of the LBP of a fingerprint image	6
2.3	Histogram of the LPQ of a fingerprint image	7
2.4	The principle diagram of the KNN classification algorithm	8
2.5	Support Vector Machines hyperplane	9
2.6	Schematic of a single artificial neuron	10
2.7	Activation functions	11
2.8	Multi-layer Perceptron with two hidden layers	12
3.1	Biometric system architecture	19
3.2	Presentation Attack Methods classification	21
3.3	Fingerprint's ridges and valleys	23
3.4	Level one fingerprint details	24
3.5	Loop pattern with details	24
3.6	Level two fingerprint details	25
3.7	Pores in fingerprint ridges	25
3.8	Fingerprint sensing technologies	26
3.9	Vulnerable points of attack in a biometric system	28
3.10	Types of Presentation Attacks	29
3.11	Fingerprints from one bona fide and several PAIs samples	30
3.12	Artificial fingerprint molding process	30
4.1	A taxonomy of the existing PAD approaches	31
4.2	Multiple frame of a live a a spoof fingerprint	33
5.1	Example of data splits for cross-validation	44
5.2	Frameworks: One-Attack and Unseen-Attack	44
5.3	Images with more and less background area comparing to the foreground area	45
5.4	Flow diagram of the segmentation method	45
5.5	Assembly of the arrays for classification	46
5.6	Segmented and cropped images	47
5.7	Scheme of the SVM classifier for PAD	47
5.8	Scheme of the MLP for PAD	48
5.9	Architecture of the Multi-layer Perceptron	49
5.10	Scheme of the regularized Multi-layer Perceptron	49
5.11	Scheme of the end-to-end solutions (CNNs) for PAD	51
5.12	Architecture of the First Convolutional Neural Network	52
5.13	Architecture of the VGG11 network already modified for fingerprint PAD	53
5.14	Examples of images after the two pre-processing methods for the VGG model	55

5.15	Architecture of the Second Convolutional Neural Network (CNN2)	56
5.16	Scheme of the regularized Second Convolutional Neural Network	58
5.17	Scheme of the proposed GAN for fingerprint PAD	59
5.18	Scheme of the proposed GAN's architecture	60
5.19	Scheme of the proposed image-to-image GAN for fingerprint PAD	62
5.20	Matcher's analysis with the detected minutiae	64
5.21	Scheme of the proposed image-to-image GAN's architecture	64
6.1	LiveDet2015 sample images	67
6.2	Generated fingerprints - Cross Match	85
6.3	Generated fingerprints - Digital Persona	86
6.4	Generated/modified fingerprints - Cross Match and Digital Persona	87
6.5	Evolution of the Classifier's loss and accuracy throughout training.	88
B.1	Examples of generated fingerprints from all sensors - One-class GAN.	B2
C.1	Examples of modified fingerprints from all sensors - Image-to-image GAN.	C2

List of Tables

3.1	Comparative analysis of common biometric traits	18
3.2	Source of characteristics in static physical reproductions.	29
4.1	Summary of the literature reviewed on fingerprint PAD	42
5.1	SVM parameters evaluated in the optimization stage.	48
5.2	Multi-Layer Perceptron training parameters.	49
5.3	Evaluation and optimization of the Adversarial and Transfer Weights.	51
5.4	First CNN architecture details and training parameters.	53
5.5	Summary of the input processing approaches.	54
5.6	VGG11 architecture details and training parameters.	55
5.7	CNN2's architecture details and training parameters.	57
5.8	CNN2's parameters that were evaluated and optimized	57
5.9	Proposed one-class GAN's architecture details and training parameters.	61
5.10	Proposed image-to-image GAN's architecture details and training parameters. . .	65
6.1	LivDet2015 dataset	68
6.2	Complete experimental work and corresponding codes.	70
6.3	Performance differences between PAISp	72
6.4	SVM vs MLP (A) - One-attack framework	72
6.5	SVM vs MLP (A) - Unseen-attack framework	73
6.6	MLP regularization (B) - Unseen-attack framework	74
6.7	First CNN experiments (C)	75
6.8	VGG11 (D) - One-attack framework	76
6.9	VGG11 (D) - Unseen-attack framework	76
6.10	Second CNN (E) - One-attack framework	77
6.11	Second CNN (E) - Unseen-attack framework	78
6.12	VGG vs CNN2 (F) - One-attack framework	79
6.13	VGG vs CNN2 (F) - Unseen-attack framework	80
6.14	CNN2 regularization (G) - Unseen-attack framework	81
6.15	Literature and proposed <i>CNN2_{reg}</i>	82
6.16	VGG vs <i>CNN2_{reg}</i> (H) - Unseen-attack framework	83
6.17	MLP vs CNN2 (I) - One-attack framework	83
6.18	MLP vs CNN2 (I) - Unseen-attack framework	84
6.19	One-class GAN results (J) - Unseen-attack	86
6.20	Image-to-image GAN results - Unseen-attack framework - Joint training	88
6.21	Image-to-image GAN results (K) - Unseen-attack framework - Phased training .	90
A.1	SVM vs MLP complete results - One-Attack - Cross Match	A2

A.2	SVM vs MLP complete results - Unseen-Attack - Cross Match	A3
A.3	SVM vs MLP complete results - One-Attack - Digital Persona	A4
A.4	SVM vs MLP complete results - Unseen-Attack - Digital Persona	A5
A.5	SVM vs MLP complete results - One-Attack - Green Bit	A6
A.6	SVM vs MLP complete results - Unseen-Attack - Green Bit	A7
A.7	SVM vs MLP complete results - One-Attack - Hi Scan	A8
A.8	SVM vs MLP complete results - Unseen-Attack - Hi Scan	A9
A.9	SVM vs MLP complete results - One-Attack - Time Series	A10
A.10	SVM vs MLP complete results - Unseen-Attack - Time Series	A11
A.11	MLP regularization complete results - Unseen-Attack - Cross Match	A13
A.12	MLP reg. complete results without segmentation - Unseen-Attack - Digital Persona	A14
A.13	MLP reg. complete results with segmentation - Unseen-Attack - Digital Persona .	A15
A.14	MLP reg. complete results without segmentation - Unseen-Attack - Green Bit . .	A16
A.15	MLP reg. complete results with segmentation - Unseen-Attack - Green Bit	A17
A.16	MLP reg. complete results without segmentation - Unseen-Attack - Hi Scan . . .	A18
A.17	MLP reg. complete results with segmentation - Unseen-Attack - Hi Scan	A19
A.18	MLP regularization complete results - Unseen-Attack - Time Series	A20
A.19	CNN1 complete results - One-Attack - Cross Match	A22
A.20	CNN1 complete results - Unseen-Attack - Cross Match	A23
A.21	VGG11 complete results - Cross Match	A25
A.22	VGG11 complete results - Digital Persona	A26
A.23	VGG11 complete results - Green Bit	A27
A.24	VGG11 complete results - Hi Scan	A28
A.25	VGG11 complete results - Time Series	A29
A.26	CNN2 complete results - One-Attack - Cross Match	A31
A.27	CNN2 complete results - Unseen-Attack - Cross Match	A32
A.28	CNN2 complete results - One-Attack - Digital Persona	A33
A.29	CNN2 complete results - Unseen-Attack - Digital Persona	A34
A.30	CNN2 complete results - One-Attack - Green Bit	A35
A.31	CNN2 complete results - Unseen-Attack - Green Bit	A36
A.32	CNN2 complete results - One-Attack - Hi Scan	A37
A.33	CNN2 complete results - Unseen-Attack - Hi Scan	A38
A.34	CNN2 complete results - One-Attack - Time Series	A39
A.35	CNN2 complete results - Unseen-Attack - Time Series	A40
A.36	CNN2 regularization complete results - Unseen-Attack - Cross Match	A42
A.37	CNN2 regularization complete results - Unseen-Attack - Digital Persona	A43
A.38	CNN2 regularization complete results - Unseen-Attack - Green Bit	A44
A.39	CNN2 regularization complete results - Unseen-Attack - Hi Scan	A45
A.40	CNN2 regularization complete results - Unseen-Attack - Time Series	A46
A.41	One-class GAN complete results	A48
A.42	Image-to-image GAN complete results - Unseen-Attack - Cross Match	A50
A.43	Image-to-image GAN complete results - Unseen-Attack - Digital Persona	A51
A.44	Image-to-image GAN complete results - Unseen-Attack - Green Bit	A52
A.45	Image-to-image GAN complete results - Unseen-Attack - Hi Scan	A53
A.46	Image-to-image GAN complete results - Unseen-Attack - Time Series	A54

Abbreviations

2D	Two-dimensional
3D	Three-dimensional
ACER	Average Classification Error Rate
ANN	Artificial Neural Network
APCER	Attack Presentation Classification Error Rate
BMM	Bernoulli Mixture Model
BPCER	Bona Fide Presentation Classification Error Rate
BSIF	Binarized Statistical Image Feature
CNN	Convolutional Neural Network
DET	Detection Error-Tradeoff
EER	Equal Error Rate
FDR	False Detection Rate
FNR	False Negative Rate
FPAD	Fingerprint Presentation Attack Detection
FPR	False Positive Rate
FTIR	Frustrated Total Internal Reflection
GAN	Generative Adversarial Network
GLCM	Gray Level Co-Occurrence Matrices
GMM	Gaussian Mixture Model
IDE	Integrated Development Environment
KNN	K-Nearest Neighbors
LDA	Linear Discriminant Analysis
LBP	Local Binary Patterns
LPQ	Local Phase Quantization
LSTM	Long Short-Term Memory
MLP	Multi-layer Perceptron
NB	Naive Bayes
PAD	Presentation Attack Detection
PAI	Presentation Attack Instrument
PAISp	Presentation Attack Instrument Species
PCA	Principal Component Analysis
PIL	Python Image Library
QDA	Quadratic Discriminant Analysis
ReLU	Rectified Linear Unit
Tanh	Hyperbolic Tangent
TDR	True Detection Rate
TPR	True Positive Rate
SIFT	Scale Invariant Feature Transform
SVM	Support Vector Machines
wLBP	Weighted Local Binary Patterns

Chapter 1

Introduction

1.1 Context

Biometrics can be defined as the use of technology to recognize individuals based on biological and behavioral characteristics and, consequently, is often applied in security systems, either in civil or government applications, which want to recognize individuals as friends, foes or unknown to the records [79]. The rise of biometric recognition technologies motivated the development of subversion of these by intruders that aim at obscure their identity or impersonate someone else's unlawfully. It became well acknowledged the need for developing countermeasures to detect and prevent unauthorized recognition attempts, or spoofing attacks, also known as presentation attacks. These attacks are undertaken by biometric capture subjects at the point of presentation and collection of the relevant biometric characteristics [32]. Therefore, the issue of effective and secure people identification and the performance of digital tasks safely and securely is one of the fundamental issues of our time [26].

Fingerprint-based identification or verification of an individual has been in use for more than a century in different scenarios among a long list of applications such as authentication of identity against identification documents, identification of individuals in criminal contexts or authorization of financial transactions, among others. The preference for these biometric systems can essentially be credited to their proven high accuracy and also to the uniqueness and persistence of fingerprints [18, 54]. Fingerprints are formed around the seventh month of fetus development and are a result of genetic and environmental factors, thus invariant throughout life and completely individualized, as even identical twins have different fingerprints [6, 79]. The worldwide use for many years of the fingerprint as a feature for biometric identification has led to a huge amount of data available, with recognition algorithms being trained and tested with hundreds of millions of records [6].

While the accuracy and verification performance are very attractive features, there is a stronger concern for the reliability, as the widespread use of fingerprints for biometric security made these systems a prime attacking target [18]. The problem worsens because of the ability to generate artifacts simply and cost-effectively by using, for example, latent fingerprints. Recent efforts have resulted in several recent approaches to tackle presentation attacks with the help of advanced sensing

techniques like Multispectral Imaging and Advanced Optical Tomographies, Short Wave Infra-red or Laser Contrast Speckle Imaging to detect artifacts. However, despite the great ability to detect attacks, these technologies are very expensive and not completely developed and optimized, so the ideal solution would be to achieve a reliable distinction between live samples, also called bona fide samples, and attack artifacts through the traditional sensors, that is, using the primary outputs of the already implemented systems [54]. This is achieved by only-software-based approaches, which started with handcrafted features and traditional machine learning algorithms and are now tending towards deep learning techniques, with Convolutional Neural Networks (CNNs) at the center of these latest efforts. However, in spite of the greater performance and robustness, CNN-based solutions require a large amount of training data to avoid over-fitting, which brings an additional problem, as there are lots of different attack materials with few available data to train these algorithms [18].

Even with all the developments in hardware and software methods, and even with the merging of super sensors with deep learning, the challenge of detecting unseen attacks is still open, that is, the capability of classifying an artifact as an attack even if that artifact is not present in the algorithm training set, since an ideal method to solve this problem has not yet been found [1]. Thus, although the evolution of the anti-spoofing solutions, there is still a major limitation regarding the poor generalization performance across novel presentation attack materials [18].

1.2 Motivation

Although fingerprint was the first biometric trace to be used as a means of individual identification, only in the 21st century the threat of presentation attacks was recognized. Before Matsumoto et al. (2002) [56] demonstrated that simple artificial fingers made with gelatin can successfully mislead most of the sensors available at the time, the reference to real and fake presentations was almost non-existent in the literature. From then on, the security of the systems against these attacks began to be considered and is currently one of the major research topics in the field of biometrics.

The use of biometrics traces and, more specifically, fingerprints, is no longer specific to high-security environments, like airports or military bases, but rather a daily habit for millions of people around the world. As the use of smartphones is becoming an essential part of everyday life, its use for sensitive tasks like financial transactions is growing significantly and fingerprint identification is increasingly an important part of these processes [31]. Although reliable for matching and recognition, when it comes to detecting presentation attacks, current smartphones' fingerprint sensors are not secure and can be easily deceived [31]. With the use of biometrics in smartphones boosting, this problem is becoming more serious. In fact, Most (2017) [58] forecasts that by 2022, 100% of the shipped smartphones worldwide will have biometrics technologies implemented and 97.17% of these devices will use biometrics to perform transactions, which will be reflected in an annual market of over US\$ 50 billion.

Attackers may not only want to access data by illegal authentication but also to access resources or even to perform identity theft, and their motivation leads to an effort to create new ways

to fool sensors [45]. Therefore, presentation attacks evolve, with new methods and new materials or sets of materials constantly misleading fingerprint sensors and even detection algorithms, so the latter need to follow and adopt new and more powerful methods to ensure that a fake sample is detected even when that type of material has not been found before in any other device. This is leading to great efforts in creating novel algorithms for the last years.

The most classic software-based approaches include varied machine learning techniques that take advantage of the characteristics of two-dimensional fingerprint images, but the new challenges have led to the use of neural networks as an attempt to increase robustness, particularly when coupled with tools such as adversarial learning and domain adaptation, which are used to improve the robustness and generalization capabilities of machine learning models. For example, by introducing regularization constraints during the learning phase, the models will be forced to be more flexible and less overfitted to the training data. Therefore, there is room for development and enhancement of current solutions to make systems as robust as possible and able to generalize from the known attacks to deal with unknown attacks.

1.3 Objectives

The main goal of this dissertation is to take a journey through the domain of deep learning and develop a series of methods for fingerprint presentation attack detection. The preparation for this dissertation ended with an intensive study of the basic concepts of Biometrics and the state of the art of fingerprint PAD. Furthermore, a good understanding of the practical aspects of developing these algorithms was obtained alongside an extensive experimental work. Therefore, the present dissertation aims to start to update the literature review and to consolidate the work done with conventional machine learning classifiers. The next step is to develop deep learning methods, testing and optimizing models, recreating published works and creating innovative approaches that can contribute with improvements to the state of the art solutions.

1.4 Contributions

From the different methodologies developed in this dissertation, there were several relevant contributions:

1. Baseline Convolutional Neural Network that is competitive with state-of-the-art baseline approaches.
2. Application of an adversarial training and transfer learning based regularization method that outperformed the state of the art solutions for unseen attacks. Led to the submission of a paper for the 19th International Conference of the Biometrics Special Interest Group (BIOSIG 2020).
3. Innovative methodology with Image-to-image Generative Adversarial Networks, which led to a better PAD generalization for all types of attack materials. Experiments to consolidate

this approach are still ongoing and a submission for a scientific journal is being prepared. This submission can be focused on the application for fingerprint PAD but also on the innovation for the fields of machine learning and domain adaptation.

1.5 Document Structure

This present introduction chapter comprises the contextualization and motivation of this dissertation, followed by the contributions to the literature and by the present section, which summarizes the overall structure of the document. The next chapter includes a study on the image processing and artificial intelligence background knowledge necessary for the development of the experimental work of this dissertation (Chapter 2). The following two chapters correspond to the initial study of biometric concepts (Chapter 3) and to the study of the literature related to fingerprint presentation attack detection (Chapter 4). Then, the entire methodology of the experimental work is explained (Chapter 5), followed by the details of its implementation, as well as the obtained results (Chapter 6). Finally, the document ends with the conclusions and proposals for future work (Chapter 7), as well as the appendixes (Appendixes A to C), which include the complete results.

Chapter 2

Image Processing and Machine Learning Background Knowledge

2.1 Image Textural Features

Many classification methods, especially the most conventional ones, do not directly use images as input, but representations of them. Therefore, these methods require an initial step to manually obtain representations from the images. This manual process corresponds to the extraction of specific characteristics/features from the images, which emphasize different properties of their texture. The following sections present the three textural features used in this dissertation.

2.1.1 Histogram

The histogram of an image is a description of its statistical distribution in terms of the number of pixels at each digital value DV , denominated as "bin". It is calculated by dividing the number of pixels in each "bin" by the image's total number of pixels N (Equation 2.1).

$$hist_{DV} = \frac{\sum_{i=0}^{N-1} p_i}{N} \quad (2.1)$$

In Equation 2.1, p_i is a binary variable with the value 1 if the intensity of the pixel positioned in the index i of the image is equal to DV and 0 otherwise. Therefore, the histogram of an image is the histogram of the intensities of that image (Figure 2.1).

2.1.2 Local Binary Patterns

The Local Binary Patterns (LBP) [67] thresholds the value of a pixel around a pre-defined neighbourhood based on the central pixel's value and calculated in a circular and symmetric way, depending on the interpolation of the locations checked against that central intensity [54]. This

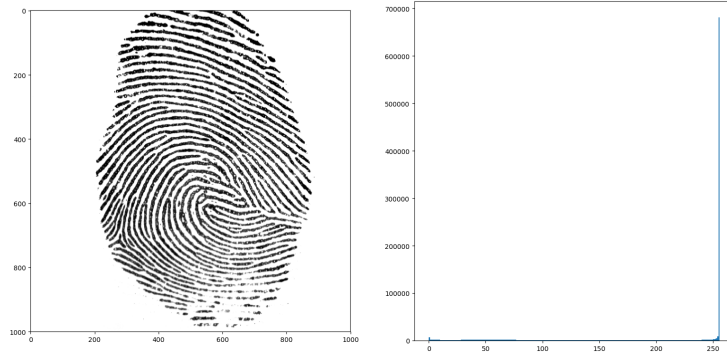


Figure 2.1: Histogram of a fingerprint image from LivDet2015 dataset [47].

computation efficiently captures local spatial patterns and contrasts in the image, thus being one of the most effective texture descriptors [12]. The LBP descriptor is defined by Equation 2.2.

$$LBP = \sum_{i=0}^{P-1} p(i - c) \times 2^i \quad (2.2)$$

Where P is the number of neighborhood pixels at a specified radius and $p(i - c)$ represents the binary function that returns 1 if the neighbor pixel i is greater than the central pixel c and 0 otherwise. By multiplying each value by 2^i , the final value is a decimal number instead of a binary array, which replaces the central pixel value. [54].

The frequency of occurrence of each LBP value, which can be defined as the histogram of all LBP results, is a good texture descriptor of an image and more cost-effective than all LBP labels, as the number of values to be processed by a classifier greatly decreases [49]. The resulting LBP image and its histogram is represented in Figure 2.2.

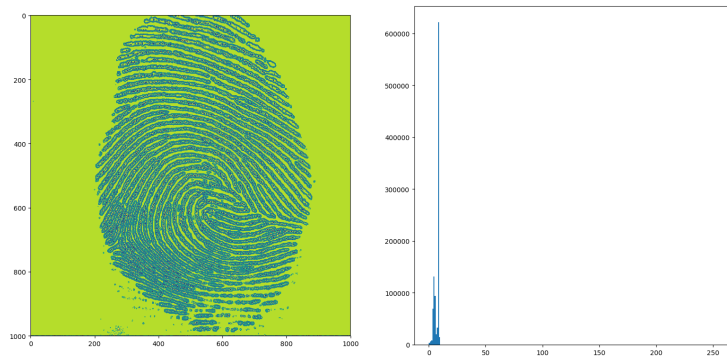


Figure 2.2: Histogram of the LBP of a fingerprint image from LivDet2015 dataset [47]. The original image is in Figure 2.1.

2.1.3 Local Phase Quantization

The Local Phase Quantization (LPQ) textural feature [68] was developed to be insensitive to blurring but has proved to achieve better classification performances even in images that are not blurred. It is obtained by computing the Short-Term Fourier Transform at four frequency points (2D frequencies), $u_1 = [a, 0]^T$, $u_2 = [0, a]^T$, $u_3 = [a, a]^T$, $u_4 = [a, -a]^T$, where a is any scalar that ensures a positive spectral response of the four points [54]. The Short-Term Fourier Transform, F , is given by the following 2D convolutions at each pixel position:

$$F(u, x) = f(x) * e^{-j2\pi u^T x} \quad (2.3)$$

Where u is the frequency, x is the pixel position and $f(x)$ is the pixel value at the position x . The result is then binarized, with the value one corresponding to when the response is greater than one, and zero otherwise. Furthermore, the result is converted to a range between 0 and 255 with a common binary to decimal operation:

$$f = \sum_{j=1}^8 R_j \times 2^{(j-1)} \quad (2.4)$$

Where R and f are the binary and decimal results, respectively [54]. The LPQ is commonly applied with its histogram instead of all pixel values, as advised by the original authors (Figure 2.3).

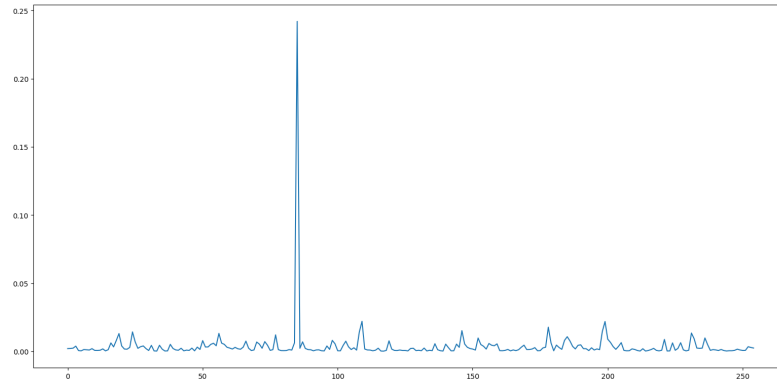


Figure 2.3: Histogram of the LPQ of a fingerprint image from LivDet2015 dataset [47]. The original image is in Figure 2.1.

2.2 Machine Learning Classifiers

2.2.1 K-Nearest Neighbors

K-Nearest Neighbors (KNN) [80] is one of the simplest classification methods but at the same time one of the most effective in predicting the labels of new data. Furthermore, this algorithm can deal with data limited to a specific range of values, also known as categorical data [63].

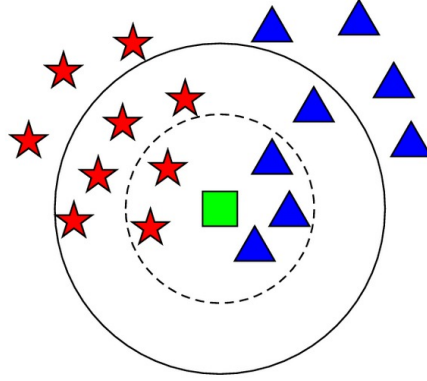


Figure 2.4: The principle diagram of the KNN classification algorithm. From Wu et al. (2014) [91].

The main goal is to find the nearest K neighbors of a data point, using a defined distance metrics such as the Euclidean distance. The class assigned to the point corresponds to the most represented class among the k -nearest neighbors [63, 64].

Figure 2.4 shows the principle of the KNN algorithm. If $K=3$, the label chosen for the new datapoint is the class represented by the blue triangles, as the three closest points of this class are closer than the three that belong to opposite class.

2.2.2 Naive Bayes

The Naive Bayes (NB) [55] classifier is a probabilistic algorithm based on independence assumptions between features and applying to them the Bayes theorem. Therefore, it is created a probabilistic model of the training features, which is then used to make a prediction of a new sample's label [70].

Bayes theorem is a conditional probability model that defines the probability of an event Y to occur if the event X has already happened, and is defined by Equation 2.5 [83].

$$P(Y|X) = \frac{P(X|Y) \times P(Y)}{P(X)} \quad (2.5)$$

Naive Bayes is a supervised classifier with the objective of predicting an initially unknown test set using the features and the label of the training data. To achieve this, the features must be conditionally independent, given the class, and no hidden attributes should be present to negatively affect the classification process [70].

2.2.3 Support Vector Machines

Support Vector Machines (SVM) [9] are one of the most popular supervised classification methods in the present and considered one of the best ones to solve classification problems. It has been successfully used in various applications, either in two class separation or in multiclass problems [28].

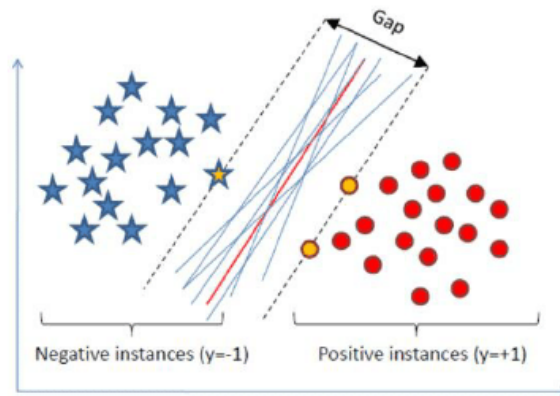


Figure 2.5: Support Vector Machines hyperplane for two classes. From Upadhyaya et al. (2014) [85].

The main goal of SVM is to separate data with the smallest possible error but in the most general manner possible, using a classification criterion, which can be for example a decision function [24]. SVM computes the optimal separating line between two classes, also denominated hyperplane, by maximizing the distance between this line and the closest points of each label, also known as support vectors. Figure 2.5 demonstrates the process of finding the ideal hyperplane to separate the data from the two classes, with several possibilities tested until the gap is maximized. Therefore, this separating line should be as far as possible from the support vectors of each class but at the same time have a minimal error in the separation of all points [28, 62].

2.3 Artificial Neural Networks

Artificial Neural Networks (ANNs) are parametric computational systems that aim to replicate the operations that occur in the human brain. ANNs are formed by a large number of processing units, just as the brain is made up of millions of neurons. These units, the artificial neurons, are only capable of performing simple operations individually, but when they are strongly interconnected they form a very complex structure that can learn, recognize patterns and make predictions even when the information at its disposal is not complete. The human brain has exactly the same behavior: an individual neuron is just a simple cell that carries an action potential through its own body, but millions of neurons connected together become an organ capable of learning to recognize people, to speak, to write poems and even to have abstract thoughts [86]. The simplest ANNs, known as Perceptrons, are made up of just one neuron or a single layer of neurons. As more layers are added, the structure becomes more complex and, consequently, the processing abilities increase. Deep ANNs have a very high number of layers and, therefore, can reach hundreds of millions of parameters. When associated with large datasets and efficient hardware, these models can achieve much better performance than the simpler ones [86, 87]. The oldest type of ANNs, known as Feedforward Neural Networks, moves the information in a single direction, that is, from the input neurons to the output ones. On the other hand, a Recurrent Neural Network features

connections towards the inputs as an addition, like a feedback mechanism [76]. Mathematically, each neuron of an ANN acts like an operator that receives a set of inputs and applies to them a linear transformation followed by a nonlinear function (Figure 2.6).

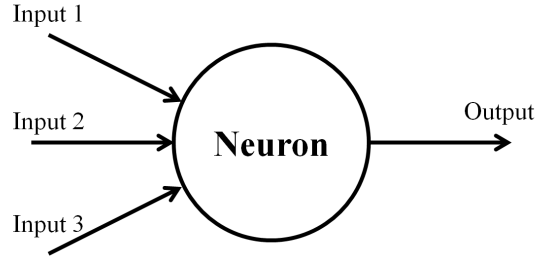


Figure 2.6: Schematic of a single artificial neuron that receives three inputs.

The output of a single neuron can be defined by Equation 2.6, where y is the output, x_i is the input of index $i \in [0, n]$, w_i the weight associated with x_i , θ the bias of the neuron, a parameter that is not associated with any input, and f the nonlinear function, also known as activation function.

$$y = f\left(\sum_{i=1}^n w_i x_i + \theta\right) \quad (2.6)$$

The activation function acts like a small decision for the neuron and can be of different types, for example:

- Logistic Sigmoid

$$f(x) = \frac{1}{1 + e^{-x}} \quad (2.7)$$

- Hyperbolic Tangent (Tanh)

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \quad (2.8)$$

- Rectified Linear Unit (ReLU)

$$f(x) = \begin{cases} x & x > 0 \\ 0 & x \leq 0 \end{cases} \quad (2.9)$$

The first two functions are very similar in behavior, however, the Hyperbolic Tangent (Equation 2.8) has a more gradual gradient, with the output between -1 and 1, whereas the Logistic Sigmoid (Equation 2.7) outputs between 0 and 1. Despite being widely used, these two functions often bring problems of vanishing gradients, where the more initial layers become stagnant with zero gradients due to the saturation of subsequent layers at the extreme outputs of the function (-1 and 1 in the case of Tanh). This problem leads to training difficulties, and it becomes often

impossible to convert to an effective model. The ReLU activation function is an alternative that takes into account the problems of sigmoid-type nonlinearities and is defined by Equation 2.9.

Therefore, a Rectified Linear Unit is activated when the input is positive and with a partial derivative equal to 1. One of the problems of this solution is the fact that the gradient is always zero whenever the input is non-positive. This can lead to some neurons never being able to activate and update their weights in order to learn, thus leading to problems similar to the vanishing gradients. To solve this problem, a variation of ReLU was defined, where a saturated unit (non-positive input) results in a non-zero but small gradient. This variation is known as Leaky ReLU and is given by Equation 2.10.

$$f(x) = \begin{cases} x & x > 0 \\ \alpha x & x \leq 0 \end{cases} \quad (2.10)$$

Where α is the negative slope of the function, commonly a small constant. Hence, this last function is very similar to the ReLU, but with a slight slope with negative inputs [48].

The four described activation functions are demonstrated together in Figure 2.7. Despite the advantages and disadvantages of each one, all four were used in different parts of the experimental work developed for this dissertation.

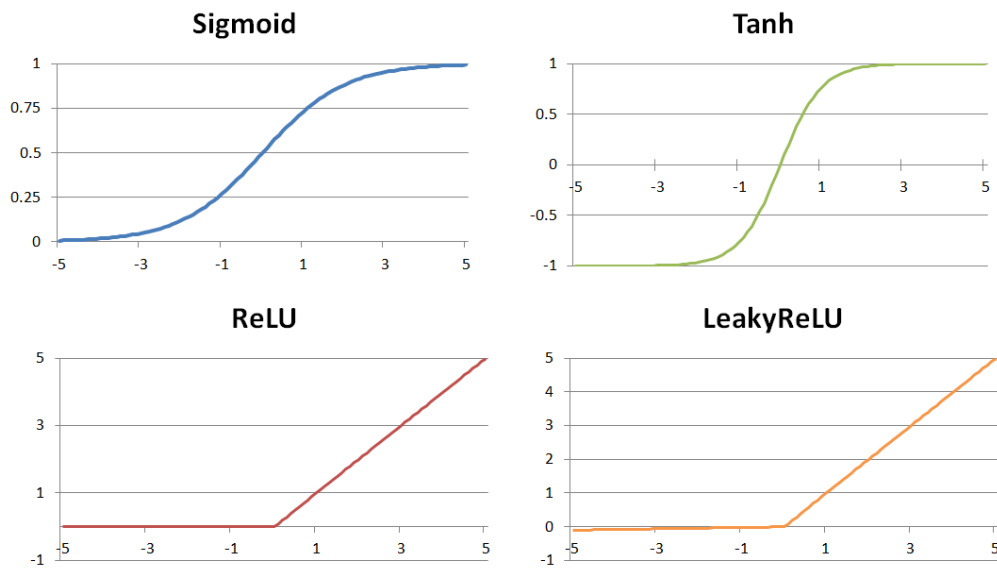


Figure 2.7: Four examples of activation functions.

2.3.1 Multi-layer Perceptrons

A Multi-layer Perceptron (MLP) is an Artificial Neural Network that has neurons, or layers of neurons, not directly connected to the output, known as hidden neurons/layers. These models are the first approach to solve nonlinear separation problems, and can achieve high complexity, with

a large number of hidden layers formed by a large number of neurons (Deep Artificial Neural Networks) [86]. Figure 2.8 shows an example of a Multi-layer Perceptron with two hidden layers.

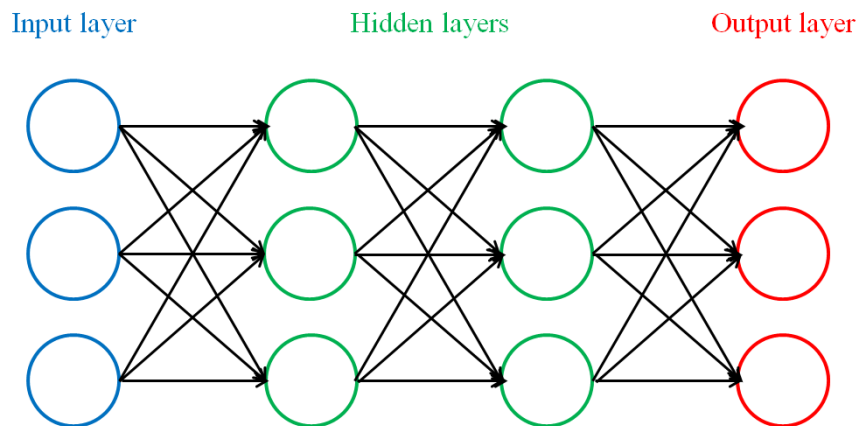


Figure 2.8: Multi-layer Perceptron with two hidden layers.

2.3.2 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are one of the most used types of ANNs today, largely due to their success and very good performance, mainly with multi-dimensional data, such as images. These neural networks use filters that are spatially similar to the input data and that are applied successively with convolutions, in order to detect and enhance patterns and characteristics of the images.

A CNN is formed by a group of layers, which are the basic units of the model, and each one implements different operations. Just like the neurons of a basic ANN, the joint work of all layers of a CNN is the reason for their capabilities. There are several types of layers, with different functions that have different complexities, of which:

- **Convolutional Layers** – A convolution layer is made of a group of filters, also known as kernels, which are convolved with the input data of the layer, returning a feature map as output. The filters are grids where each cell is a weight to be updated during training. For example a two-dimensional kernel with size 3×3 is a square matrix with 9 elements. During the convolution operation, the filters can take different steps after every convolution, which is called stride. Moreover, the dimension of the input can be increased to give more flexibility to the model by applying sets of zeros around it, which is called padding.
- **Transposed Convolutional Layers** – A transposed convolution layer has the same behavior and properties of a convolution layer, but the data flows in the opposite direction, like a backward pass. Therefore, the output of these layers is relatively “bigger” than the input, contrarily to what happens in convolution layers, so this operation “up-samples” the input.

- **Pooling Layers** – A pooling layer applies pooling functions such like Maximum (Max) and Average to the input/feature map through a window with a defined size (for example, a 2x2 matrix) and with a defined step/stride. As the names indicate, the Max Pooling function keeps the maximum value inside the window and the Average Pooling computes the average of the values inside the window. Hence, pooling layers are used to “down sample” the data, which makes the representations more compact and invariant to modifications.
- **Fully Connected Layers** – A fully connected layer in a CNN is the same as a layer of neurons in a Multi-layer Perceptron, so every unit is connected to all units of the previous layer. The operation performed by this layer is the same that occurs in non-convolutional ANNs, which is already described in Section 2.3.

By grouping different types of layers, it is possible to automatically extract features from the input data and classify those features with a predicted label, which is an end-to-end solution, also known as pure deep learning. Instead of optimizing the weights for just the classification task, these models also learn the best features to extract from the input by optimizing the weights of the initial layers. The convolution layers are most commonly used for the feature extraction step and the fully connected for the classification task, however, there are endless possibilities to organize the layers of a CNN and define its architecture. For example, the Fully Convolutional Neural Networks don't have any fully connected layers but can also be used for classification. Each layer is normally followed by a non-linear activation function, just like a classic ANN (Section 2.3).

Regarding the training of CNNs, they have a considerable number of parameters, which makes it easier to overfit on the training data. To avoid this, there are some possibilities to regularize a model and guarantee a correct learning:

- **Data Augmentation (images)** – by making copies of the images and performing simple operations to them, such like rotating, cropping and flipping, the size of the training dataset can increase significantly. This is very helpful when the original dataset is small and there is few images for the model to learn, as well as to force more varied inputs into the model, hence reducing the chances of overfitting. When the training dataset is already large but some variance is needed, it is common to define a probability to perform the operations to the images. For example, if the probability is set to 50%, half of the images entering the model are a result of a data augmentation operation, thus maintain the number of samples but increasing the variability.
- **Dropout** – during the training phase of an Artificial Neural Network, it is possible to define a probability for the activation of the neurons, that is, at each training step, only a random sub-sample of the model is used and updated. This technique introduces “noise” to the training, so the model learns to generalize better, thus avoiding overfitting.
- **Batch Normalization** – if the output activations of the layers are normalized, instead of just normalizing the input data, the distributions inside the model are maintained similar, which helps it to learn and consequently reduces significantly the time needed to converge.

- **Weight Constraints** – the weight decay technique penalizes large weights concentrated on a small number of neurons by adding a hyper-parameter to the loss, hence enforcing a good distribution of weights through the entire model.

In summary, CNNs are the most emergent deep learning solutions today and increasingly show potential in their tasks, even outperforming the more traditional approaches, mainly with large quantities of training data. This success comes from the variety of layers and computations that can be made inside the models, greatly increasing their complexity and, thus, their capability of learning [42].

2.3.3 Generative Adversarial Networks

The Generative Adversarial Network (GAN) [27] is a recent framework proposed to improve the performance of generative models by introducing adversarial training techniques. For this, two different Artificial Neural Networks are trained simultaneously: a Generator, G , and a Discriminator, D . The Generator learns the distribution of the training data and tries to generate new samples from a random noise input, as similar as possible to the original samples. At the same time, the Discriminator wants to distinguish between real and generated samples. In conclusion, a GAN is a two-player game between the two ANNs, where G tries to minimize the following function and D tries to maximize it:

$$E_x[\log(D(x))] + E_{G(z)}[\log(1 - D(G(z)))] \quad (2.11)$$

Where x is a real sample, z a random noise, $G(z)$ the output of the Generator, $D(x)$ the Discriminator's output for real samples (probability to be real) and $D(G(z))$ the Discriminator's output for generated samples (probability to be generated). The values E_x and $E_{G(z)}$ are the expected outputs of the Discriminator for real and generated samples, respectively, so they are in effect the ground truth labels. As the first term of Equation 2.11 is not affected directly by the Generator, its loss can be simplified to just the second term:

$$E_{G(z)}[\log(1 - D(G(z)))] \quad (2.12)$$

The training of a GAN is made by alternatively passing the real samples and then the generated samples through the Discriminator, computing the classification loss. The Discriminator is then updated by backpropagation to minimize the classification loss of all samples and the Generator is updated to maximize the Discriminator's loss for the generated samples, that is, is updated to generate better and better samples, increasingly similar to the real ones. After enough training, the Generator would ideally be capable of generating samples so real that the probabilities of the Discriminator become always close to 50%. This means that the Discriminator is no longer capable of distinguishing between real and generated samples.

Although GANs are relatively recent, they are one of the most emergent types of neural networks and are being used for a large variety of tasks. These models are very flexible in terms of

architectures, from only fully connected layers to fully convolutional networks (Fully Deep Convolutional GANs), and in terms of training methods for the Generator and Discriminator. Consequently, there are innumerable possible extensions and variations of GANs, like the Wasserstein GAN [3], which improves the training stability, and the Conditional GANs, where additional data is introduced to the Generator, for example, an image label. This last example led, in its turn, to the creation of image-to-image translation models, where the input and output of the Generator are both images, like the Pix2Pix GAN [99], and the Cycle-GAN [34].

Chapter 3

Fundamental Concepts of Biometric Systems

3.1 Biometric Recognition

At the beginning of human civilization, the population of each settlement was rather small, so it was easy for the individuals to know and identify each other. The growth of the human population and the evolution of mobility led to a great expansion where individuals no longer can manage all identities alone, and, therefore, identity management systems have emerged to establish an association between individuals and their personal identity. In this way, it became possible to determine the identity or confirm a person's identity claim, which is also known as person recognition [35]. The traditional methods used for person recognition are based on what a person holds and knows, also referred to as token-based and knowledge-based, respectively. The first one uses personal items such as identification cards, passports or driver's licenses and the second relies on a piece of information that is theoretically solely known by the individual, like a password, an identification number or a numeric key. Although their wide utilization, these techniques are not reliable and have great risks, as it is very common to lose a personal card or forgot personal information, for example. To overcome these disadvantages, a third method emerged by using a person's physical or behavioral characteristics for recognition, which is known as biometric recognition or simply biometrics [35, 78].

Biometric recognition can be defined as the use of natural and inherent human characteristics to achieve personal recognition. Biometric characteristics are universal and unique to each individual, as well as more permanent than the traditional techniques, so they are a much more reliable form of identification. Biometric technologies are currently used in several different applications worldwide. In forensics, they are used to identify criminals, corpses or missing people, and in commercial applications, they are already present in bank ATMs, as well widely used in mobile phones and access control systems in companies or other facilities. Finally, the governments use biometrics in national identification cards, driver's licenses and to control border crossing [60, 78].

3.1.1 Biometric Modalities

Biometric modalities refer to the inherent characteristics used for biometric recognition, also known as traits. These traits must take into account different properties to constitute good identification tools: *Universality* means every person has that trait, *Uniqueness* means the trait is different in every single person, *Collectability* refers to the easiness to collect and quantify the trait with external sensors and *Permanence* is the immutable nature of the feature through a person's life. In addition, *Performance* relates to the accuracy and speed of the system, *Acceptability* to the population acceptance of its usage and *Circumvention* to the possibilities of deceiving the same [41].

Biometric traits can be genotypic or phenotypic, if permanent through a person's life or dependent on the environment, and also can be physical or behavioral. Physical or anatomical traits are parts of the human body that can be used for personal identification, such as fingerprint, face, iris, retina, hand geometry and veins, DNA, facial thermogram, palm print, and ear shape. On the other hand, behavioral traits describe a person's conduct, such as gait, signature and the way a keyboard is typed. Finally, some traits include anatomical and behavioral characteristics, like an individual's voice. In Table 3.1, the enumerated traits are classified in terms of the first four properties defined previously. Both performance, acceptability and circumvention of a biometric trait are also dependent on the technology used and not only on the trait itself [41, 78].

Table 3.1: Comparative analysis of common biometric traits. Adapted from Sequeira (2015) [78] and Manivannan et al. (2011) [50].

Traits	Properties			
	<i>Universality</i>	<i>Uniqueness</i>	<i>Collectability</i>	<i>Permanence</i>
Fingerprint	Medium	High	High	Medium
Face	High	Low	High	Medium
Iris	High	High	Medium	High
Retina	High	High	Low	Medium
Hand geometry	Medium	Medium	High	Medium
Hand veins	Medium	High	High	Medium
DNA	High	High	Low	High
Facial thermogram	High	High	High	Low
Palm print	Medium	High	Medium	High
Ear shape	Medium	Medium	Medium	High
Gait	Low	Low	High	Low
Signature	Medium	Low	High	Low
Keystroke	Low	Low	Medium	Low
Voice	Medium	Low	Medium	Low

Among all the possible modalities, fingerprint, face and iris are the most accepted and used worldwide, while hand geometry is the least implemented [41]. Fingerprints are the most important trait in forensics but face recognition is growing in this domain with large datasets being created, whereas in civilian applications, both fingerprint, iris and face recognition systems are

widely used. For commercial purposes, fingerprint and face traits are increasingly implemented in mobile phones, and, in conjunction with iris and voice, are widely used for access control [78].

3.2 Biometric Systems

3.2.1 Architecture of a Biometric System

All biometric recognition systems can be divided into two different stages: enrollment and identification/verification (Figure 3.1). The enrollment phase consists of the first time a user presents its biometric data, that is, corresponds to the phase when the database is constructed with the data that will be later used for biometric comparison. On the other hand, the identification/verification stage has the main goal of recognizing a new input by comparing it to the data present in the database. Both stages require a pre-classification process which can include pre-processing techniques to prepare the input data for the comparison phase. Then, a set of features is extracted and compared to the same set of features from images present in the database, which is the feature comparison phase, followed, by the recognition [78].

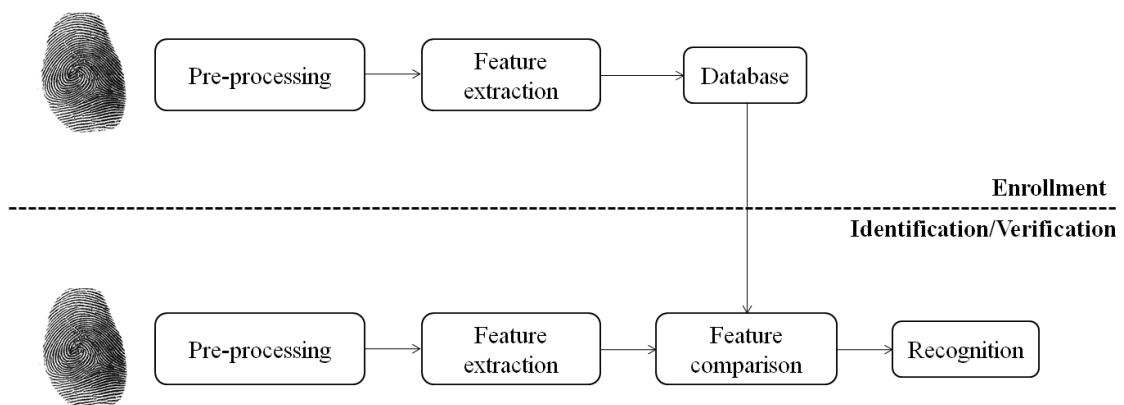


Figure 3.1: Biometric system architecture. Adapted from Sequeira (2015) [78].

3.2.2 Operating Mode of a Biometric System

A biometric system can be used for both biometric verification and biometric identification. The first one consists of confirming an identity claim by matching it to the corresponding sample in the database, whereas the second aims to detect an individual's personal identity by searching and comparing it to all the identities in the database. Verification is thus a 1 to 1 comparison, where the person's collected trait is matched to the trait of the same identity present in the database, and identification is a 1 to N comparison where N is the number of samples in the system's database [78].

3.2.3 Vocabulary

In biometrics, there are some terms that must be recognized. A *biometric sample*, also known as raw biometric data, is the initial data used for the enrollment stage and to create templates, before any processing stage being applied. For example, in fingerprint recognition, a sample is a fingerprint image and in voice recognition, a sample is a sound recording.

A *biometric template* is obtained after the feature extraction stage and consists of a mathematical description of the sample, which is the data used for matching. A template resultant from the enrolment phase is called a *stored template* while one obtained in the verification/identification stage is a *live template*.

Presentation is the process of presenting the biometric data to the system's hardware, also known as the *accusation device*, either at the enrollment or at the verification/identification stages. The accusation device depends on the trait that is being used: a sensor for fingerprints, a camera for face and iris or a microphone for voice recognition [41].

3.3 Presentation Attack Detection

Presentation attack detection (PAD) methods have the main goal of determining if a biometric sample is authentic or if it is fake or has been modified, that is, these methods, also known as liveness methods, aim to classify a presentation as real or fake. Presentation attacks occur at the data capture stage of a biometric system and try to subvert it by presenting an imitation of a real and live sample. The task of detecting these attacks can then be defined as a binary classification task, where one class represents the real presentations and the other the fake ones. This is achieved by extracting relevant features from the data to facilitate the distinction between the two classes [32].

A PAD method can be classified into four different types depending on the features extracted and its timing on a real biometric system. A *Perfect Matching Model* performs biometric recognition and liveness detection at the same time and with the same features whereas a *Simultaneous Measuring Model* uses different features collected at the same time. On the other hand, a *Same Biometric Measuring Model* obtain the same features for recognition and PAD at different times and, finally, a *Independent Measuring Model* uses different features collected at different times [78]. A summary of this classification of PAD methods is shown in Figure 3.2.

A presentation attack can have different objectives and be performed in various ways depending on the biometric trait that is being used, which will be explained and exemplified for the specific case of fingerprints in this dissertation. At the same time, a presentation attack detection solution can be achieved with different approaches and techniques, depending also on the biometric trait. The various ways of fabricating attacks and the various methods for detecting them are described in more detail for the specific case of fingerprints in Subsection 3.5.3 and Chapter 4, respectively.

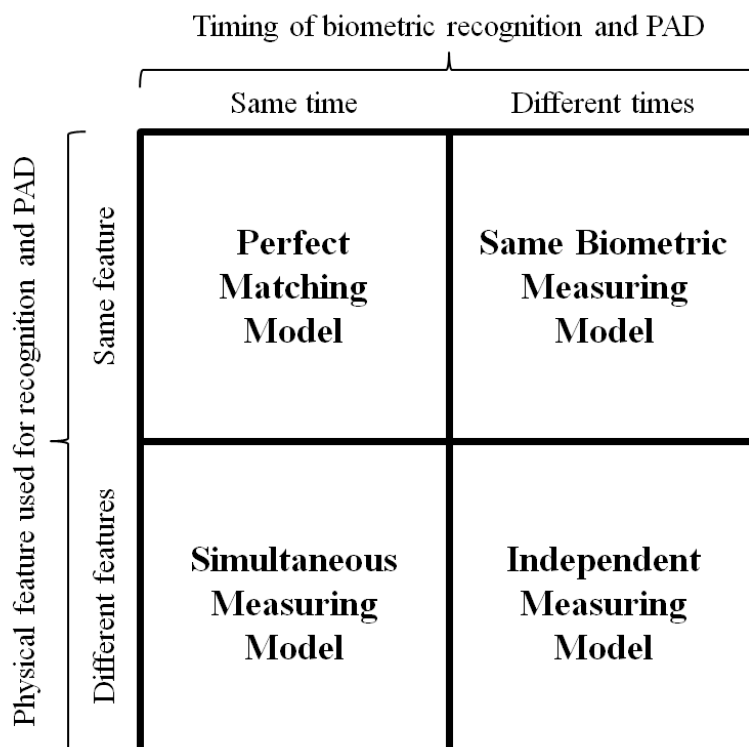


Figure 3.2: Presentation Attack Methods classification.

3.3.1 Vocabulary

In presentation attack detection, some terms must be recognized. A biometric presentation can be either a *presentation attack* when it aims to subvert the system or a *bona fide presentation* otherwise. Presentation attacks can be referred to as spoofs or artifacts, and bona fide presentation is analogous to normal or live presentation.

A *presentation attack instrument* (PAI) corresponds to the biometric feature or object used to perform a presentation attack. A *PAI species* (PAISp) is a class of PAIs made with the same materials or methods but for different biometric identities, whereas a *PAI series* is a group of PAIs made for the same identity and with the same fabrication process [32, 33].

3.4 The Biometric Menagerie

Biometric systems sometimes have high recognition errors even when all the users perform in compliance with the policies, that is, without considering presentation attacks. These lower performances are a consequence of certain types of users that affect the recognition task, and by identifying them it becomes possible to target each one of the problems and develop true robust solutions. The Biometric Menagerie was first formalized by Doddington et al. (1998) [11] and divides the users into four groups designated by animals that metaphorically represent the users' characteristics:

- **Sheep** - A sheep match well against itself and does not match well against others. The sheep are the most common users and are not problematic.
- **Goats** - A goat matches poorly against itself, resulting in a high number of false rejects, that is, false-negative verifications.
- **Lambs** - Lambs are very easy to impersonate, that is, if someone matches against a lamb, there is a high probability of a positive result, even if it is a different individual.
- **Wolves** - Wolves are experts in impersonating lambs, achieving lots of false-positive matching results, that is, high false-acceptance rates.

The scores' distributions of the problematic populations (goats, lambs, and wolves) are very different from the distributions of the most common group (sheep), and, usually, an individual cannot be classified entirely as a goat, a lamb or a wolf but will rather display different degrees of "goat-like", "lamb-like" or "wolf-like" behaviors [92]. Regarding presentation attacks, individuals from the lamb class are logically those who most easily suffer from successful identity frauds, as someone impersonating a lamb can subvert more easily a biometric verification system.

Therefore, by understanding the population distribution through its different classes and the intrinsic problems of each class, it becomes easier to increase the matching performance of a biometric system and at the same time its robustness against either purposeful or not purposeful recognition errors.

3.5 Fingerprint Based Recognition

A fingerprint is the unique characteristic of the fingertip's skin and corresponds to a pattern of ridges and valleys (Figure 3.3) that is unique to each individual, even in identical twins. In addition, this pattern remains unchangeable through an individual's life until death, unless in very specific cases of accidents where the fingers suffer bruises or the fingertips are cut off or severely damaged [6, 30]. These features have made the fingerprint an excellent identification tool and thus the most widely used biometric feature in the world, with records of their use dating back thousands of years [20, 90].

There are several hypotheses regarding the biological processes that lead to the beginning of the formation of fingerprints between the third and fourth month of fetus development, which is finalized around the seven month of pregnancy [20, 90]. It is believed that exists a strong genetic component, but also an influence of differential forces in the skin or even biochemical processes. The mechanical point of view defends that the fingerprint ridges correspond to folds in the skin surface caused by differences between the dermis and epidermis growth, whereas the chemical hypothesis considers that the presence of substances in the skin surface leads to folding [20]. In the case of identical twins, the uniqueness is a consequence of the flow of amniotic fluids and the position of the fetus in the uterus, which changes the growth patterns of the skin cells. These

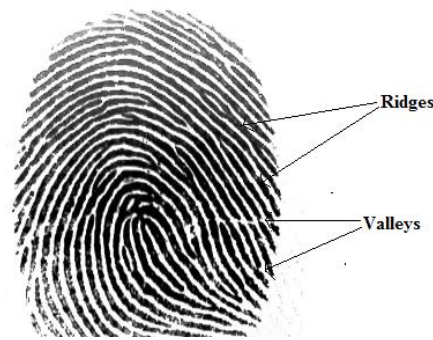


Figure 3.3: Fingerprint's ridges and valleys. From Rao et al. (2015) [73].

microscopic changes in each cell are amplified by their differentiation, originating macroscopic differences that lead to differences between twins' fingerprints [30].

In terms of anatomical purpose, fingerprints are thought to improve the ability to hold and grasp objects more securely, which can be the cause of the greater primate's dexterity relative to all other mammals [90]. This idea only refers to rough objects, as the fingertip roughness caused by the ridges and valleys of fingerprints increases the friction between the two surfaces, however, when a larger contact area is an important factor, such as in rubbery materials, the existence of fingerprints is no longer an advantage [88]. Another hypothesis lies in the optimization of the sense of touch through spectral selection and amplification of tactile information that leads to an easier processing by the nervous receptors, which may also explain the more advanced capabilities of primates in the use of the hands for more precise tasks [77], however, there is still no consensus about this subject.

3.5.1 Fingerprint Features

A fingerprint can be characterized by its global information, that is, the overall pattern, or by local information that derives from the ridges. There are three different levels of detail when describing the patterns, which brings different types of fingerprints, depending on how the ridges are disposed of.

3.5.1.1 Level One Features

The first level focus on global macro details, mainly the shape formed by the ridges and valleys in large regions of the fingerprint, which can be classified as an arch, a loop or a whorl (Figure 3.4) [51].

Arch fingerprints are the rarest pattern of all three, and as the name indicates, the ridges are curved like an arch, moving from one side to another with no reversed turns. There are different categories of arch patterns, which refer to differences in the arch size and orientation at the center of the print. On the other hand, loop patterns are the most common and result from re-curves of the ridges, forming at least one core point and one delta point, as seen in Figure 3.5. Finally, whorl



Figure 3.4: Level one fingerprint details. From Marasco et al. (2014) [51].

fingerprints have at least two delta points and thus some ridges turn through one circuit, forming a circular and content structure [90].

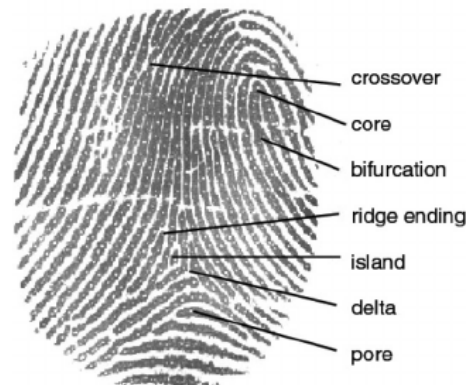


Figure 3.5: Loop pattern with details. From Marasco et al. (2014) [51].

3.5.1.2 Level Two Features

The second level features are focused on the local level and classify different minor details in the pattern, such as ridge endings and bifurcations or dots and islands, as observed in Figures 3.5 and 3.6. These features are denominated Galton characteristics or minutiae points and are the foundation of all the traditional classification and matching algorithms for fingerprints. These methods rely on good image pre-processing to optimize minutiae extraction and thus achieving a good classification performance, as each minutiae is represented by its two-dimensional coordinates and a value representing the direction of the ridge on that specific point, constituting the main features that are used by the classifiers [10, 51].

3.5.1.3 Level Three Features

The third level is the finest one, where small details like sweat pores, scars and incipient ridges, that is, ridges that are not fully developed, are captured. These features demonstrate great potential

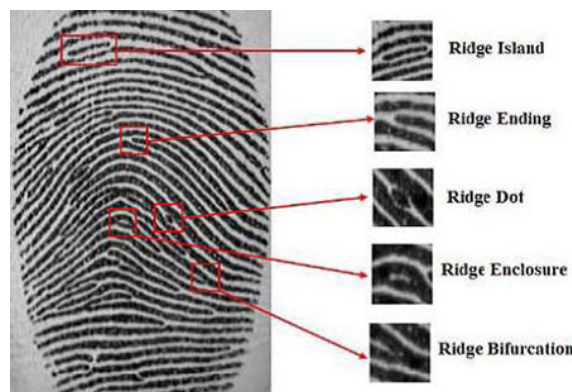


Figure 3.6: Level two fingerprint details. From Bayometric (2020) [5].

in the optimization of matching algorithms [36, 40], however, they need high-resolution sensors to be effectively scanned, which are not as available as the less advanced ones and are usually expensive. Figure 3.7 shows two images of fingerprint pores obtained with two different sensors, where it is even possible to observe different positions of the pores inside the ridges, which is a consequence of whether or not they are open [51].

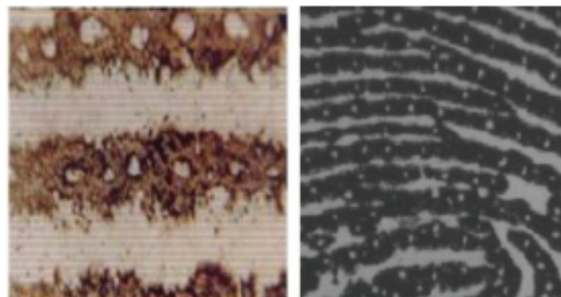


Figure 3.7: Pores in fingerprint ridges. From Marasco et al. (2014) [51].

3.5.2 Fingerprint Sensing

Fingerprints can be acquired by various technologies, with numerous principles being utilized for ridge/valley structure capturing. The fingerprint sensing methods can be classified as optical or as solid state (Figure 3.8). Optical sensors use light sources, prisms, lens or optical fibers associated with a photosensitive surface to obtain the pattern, thus capable of penetrating thicker glass, whereas solid state sensors consist of a single chip solution, with the sensing technology integrated on the chip, making them more reliable for small and portable solutions [6, 51, 82].

Regarding the optical sensors, there are several principles and technologies applied. The first to be developed was the Frustrated Total Internal Reflection (FTIR), which takes advantage of a light source, a prism and an optical camera (CCD/CMOS) to detect differences in light reflection and absorption caused by the ridges and valleys of the fingerprints, making it reliable against fake two-dimensional artifacts. However, the necessity for these components makes these sensors

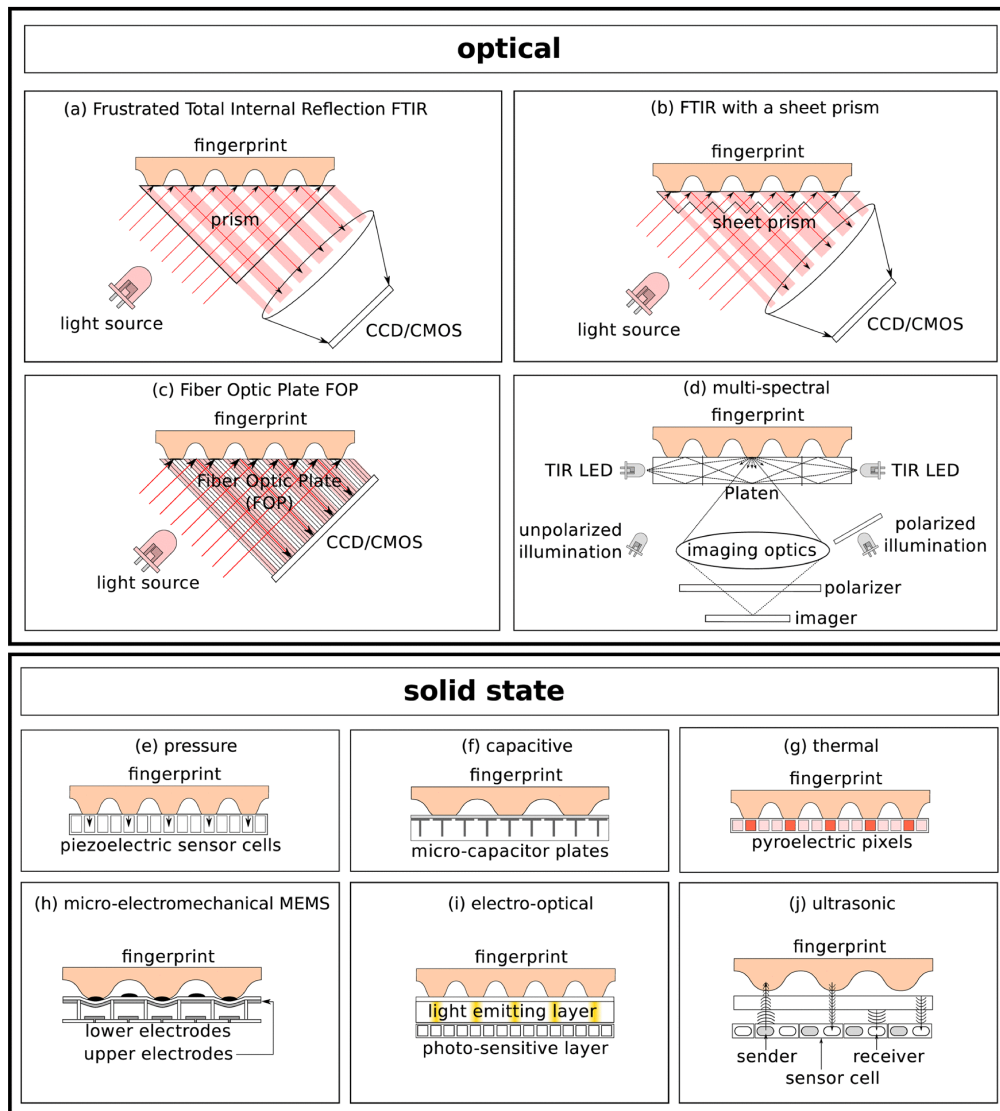


Figure 3.8: Fingerprint sensing technologies. From Busch et al. (2014) [6].

larger, which is solved through the sheet prism FTIR by employing a large number of small adjacent prisms with losses in image quality [6]. These sensors can be spoofed with materials that have reflective properties similar to that of the skin and can vary across manufacturers because of differences in the components utilized [51]. The Fiber Optic Plate technique has a compact design because it uses a grid of optical fibers instead of a prism or set of prisms, and the multi-spectral technology capture various images of the fingertip under different light sources, performing well in difficult conditions [6].

In contrast, the solid-state sensors can be very compact and cost-effective because of the single-chip integration. The pressure type fingerprint sensor is based on the piezoelectric effect, so when the fingertip is pressed against the surface, the ridges contact with the sensor units, which are piezoelectric cells that produce voltage when pressure is applied [6, 82].

Capacitive fingerprint sensors are the most widely used scans because they are cost and size effective, thus are used on mobile devices. This technology obtains fingerprint images by measuring the difference in capacitance between the fingerprint ridges and valleys and the sensor, modeling the finger as the upper electrode of a capacitor and the plate as the lower electrode [51, 82].

On the other hand, thermal sensors measure the difference between temperature caused by the skin that is in contact with the pyroelectric material (ridges) and the air in the valleys, because of the thermal equilibrium that occurs between the fingertip and the chip [6, 51], whereas ultrasonic sensors emit acoustic waves and adopt the same principle of medical ultrasound to create a visual image of the fingerprint [82].

Finally, two additional techniques should be referenced: the micro-electromechanical and the electro-optical, which, as the name indicates, measure pressure differences caused by the texture in the sensing cells and the variabilities in the electrical potential induced by the pattern, respectively [6].

Alternatively, there is an additional classification that focuses on what action the individual needs to perform for the fingerprint sensing technologies to acquire the image: swipe, touch and touchless sensors. Touch-based sensors are the most widely used, requiring the appliance of pressure by the finger, whereas swipe sensors just need the finger to swipe over the surface, which reduces costs but increases error rates, and touchless systems can capture the fingerprint in its original condition, that is, without the deformations caused by pressing the fingertip onto the sensor [6].

All these technologies are vulnerable to attack artifacts that have skin-like properties correspondent to the properties measured by the sensor, that is, a material with similar thermal properties to that of the skin will perform well when trying to attack a thermal-based system, or a fake finger with the same echoing characteristics of a real finger can spoof ultrasonic sensors [51].

This problem led to efforts in improving the sensing technologies that resulted in various recent methods that perform well when tested against presentation attacks. These advanced sensing techniques, like Multispectral Imaging, advanced Optical Tomographies, Short Wave infra-red, Laser Contrast Speckle imaging or even three-dimensional image acquiring can be fused with deep learning algorithms to achieve more robust systems, however, the costs are high and the performance is not yet ideal, making the detection of spoofing attacks a problem that still remains to be solved [1, 54].

3.5.3 Vulnerabilities of Fingerprint Recognition Systems

Attacks on fingerprint sensing systems can target various points and instances of the process, as observed in Figure 3.9. Thus, attacks can focus on the final decision phase, like changing the sensor's final classification (numbers 5 and 8 in Figure 3.9) or tampering with the data used as a model (numbers 6 and 7 in Figure 3.9) during the processing phase, as input (numbers 2, 3 and 4 in Figure 3.9) or at the fingerprinting stage (number 1 in Figure 3.9) [51]. This last point of vulnerability then corresponds to the presentation attacks and therefore is the focus of this work.

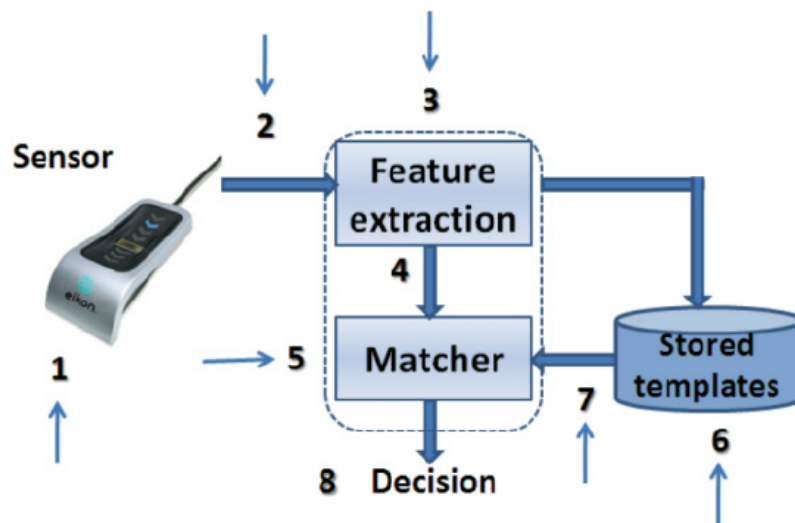


Figure 3.9: Vulnerable points of attack in a biometric system. From Marasco et al. (2014) [51]. Where: 1 - Presentation Attack, 2 - Biometric Signal Replication, 3 - Feature Modification, 4 - Feature Replacement, 5 - Matcher Overriding, 6 - Templates Replacement, 7 - Data Modification, 8 - Decision Alteration.

A fingerprint presentation attack is therefore intended to mislead the sensors through untruthful fingerprinting so that the attacker is identified as a different individual, called a biometric impostor, or not recognized as an individual known to the system, also called biometric concealer. In turn, an impostor may want to be recognized as a specific system individual or simply recognized without any specification. The concealer simply wants to hide his biometric characteristics by using an artifact or even physically altering his natural fingerprints [32].

The artifact used in a presentation attack, the presentation attack instrument (PAI), can be artificial or human-based, accordingly with the method used to obtain it, which in turn can be artificial or take advantage of human characteristics and behaviors, as seen in Figure 3.10 [32].

The use of real human fingers involves the use of lifeless samples, that is, a finger from a cadaver or the alteration of living ones, which can include the destruction or the surgical replacement of the original fingerprint. Non-conformability is related to the act of altering the capture with the intent of performing an attack, so that the sensor's decision is corrupted, whereas a conformant case occurs without the presence of an impostor, thus being an incorrect classification by the sensor. Lastly, a coerced presentation attack happens when the fingerprint of an individual is used when the same individual is being forced or unconscious, that is, corresponds to the use of biometric characteristics under duress [33].

Artificial attacks do not use real biological tissue but attempt to replicate it with an identical fingerprint structure. This can be achieved by generating a synthetic sample generation, that is, building a PAI not based on the biometric characteristics of any specific individual or based on a standard template [7, 19], or by creating a static physical reproduction, which in turn can be produced by casting, direct rendering and masking. Casting is a two-step process, consisting of molding followed by casting from the mold, direct rendering involves two-dimensional (2D) or

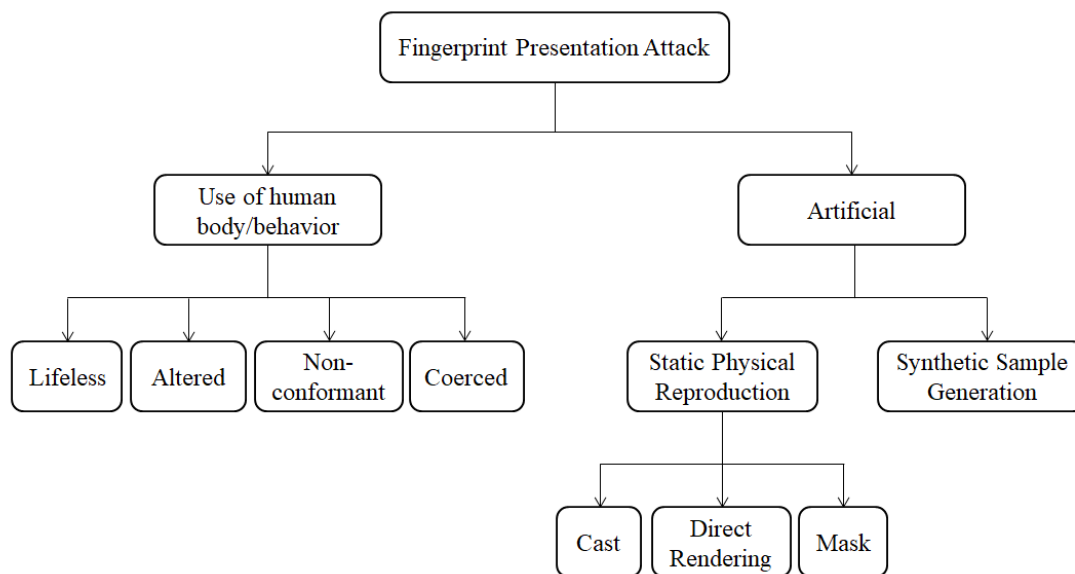


Figure 3.10: Types of Presentation Attacks. From ISO/IEC (2017) [33].

three-dimensional (3D) printing or etching a fingerprint on metal, and a mask can be simply a layer of glue on a finger to conceal the original fingerprint [33].

Table 3.2: Source of characteristics in static physical reproductions. From ISO/IEC (2017) [33].

	Description	Example
Cooperative	Captured directly from another individual with assistance	Finger mould with silicone, plasticine, candle wax, etc.
Latent (non-co-operative)	Captured indirectly through a latent fingerprint sample	Enhanced fingerprint on transparent film, printed circuit board, etc.

Another classification of presentation attacks is based on the distinction of the source of the characteristics, that is, the source of the original model, for generating the PAI, dividing them into two main types: Co-operative and Non-co-operative. As the denomination suggests, a co-operative spoofing happens when the real fingerprint owner participates willingly on the process, whereas a non-co-operative spoofing is achieved against the will or without the existence of an original individual. All human-based PAIs are non-cooperative, by definition, and the synthetic generation does not involve a real individual or a real origin, so it only makes sense to refer to this classification when referring to the methods of static physical reproduction, which is summarized in Table 3.2 [33, 44].

The spoof fingerprints can be made by a huge variety of materials, from latex to wood glue (Figure 3.11), with new possibilities often arising, as well as adaptations or combinations of the already known ones. For example, in the casting process, a mold is made by a plastic-based material and then filled with a moisture-based material to form the PAI (Figure 3.12), and this is where the possibilities are almost endless. The quality of an artifact depends on their thickness,

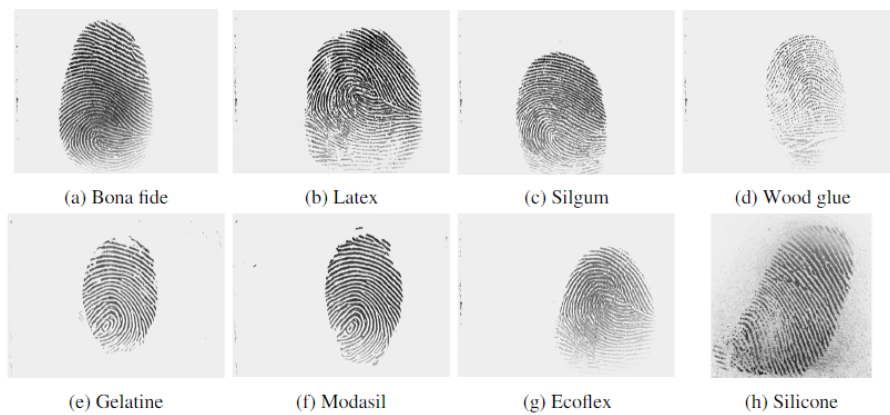


Figure 3.11: Fingerprints from one bona fide and several PAIs samples. From González Soler et al. (2019) [25].

drying time and a potential defects during creation, and different materials perform differently in different types of sensors, as silicone PAIs are normally rejected by capacitive sensors but usually surpass optical sensors, in example [44, 51].



Figure 3.12: Artificial fingerprint molding process. From Marasco et al. (2014) [51].

Given this variability and endless possibilities for obtaining a PAI, as well as the constant emergence of new PAI species or the evolution of existing ones, it is increasingly important to create robust algorithms for these attacks, hence the effort that has been registered in recent years.

Chapter 4

Presentation Attack Detection Methods

Knowing what fingerprint presentation attacks are and the risk they impose (Section 3.5.3), it is now important to study the different ways to detect them. Presentation attack detection (PAD) methods can be hardware-based or software-based, depending on the technologies used to develop it. Figure 4.1 shows an overall visualization of the more traditional methods and the emerging ones for fingerprint PAD, each one detailed in the following sections.

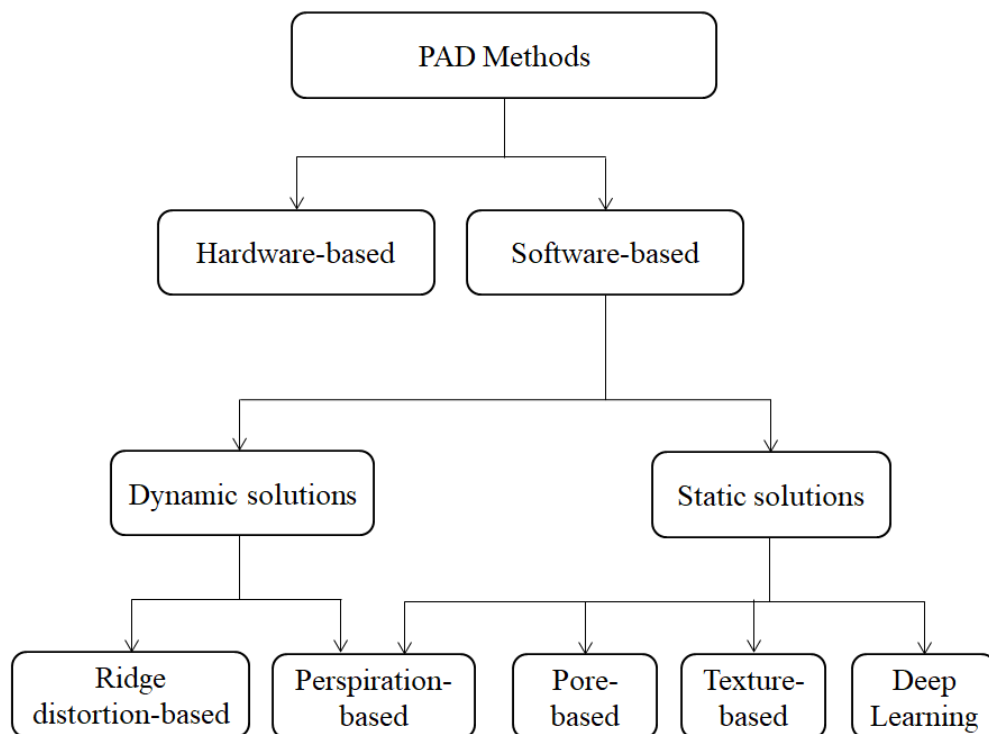


Figure 4.1: A taxonomy of the existing PAD approaches with focus on the software based ones. Inspired from Marasco et al. (2014) [51].

4.1 Hardware-based Presentation Attack Detection Methods

Hardware-based approaches involve adding sensors or updating the already available hardware of the fingerprint reading systems in order to capture additional information besides the image for the matching algorithm and thus making them difficult to deceive. There is a huge variety of features that can be collected with these additional sensors, such as odor through an electronic nose, heartbeat due to slight volumetric changes, thermal output, blood flow, pulse oximetry, electrical and multi-spectral properties, or even imaging methods to observe 3D models of the finger and non-superficial structures such as sub-dermal fingerprints [6, 13]. These characteristics can be classified into static if they are extracted from a single fingerprint print or in a single measurement such as odor and electrical properties, or dynamic when it is necessary to analyze and process several sequential frames of the same sample to obtain information, such as skin elasticity or blood flow.

Besides the necessity to add physical components, these methods also have limitations as it is possible to fool the supplemental sensors. For example, if the silicone layer is thin enough, no differences in the temperature can be detected when comparing it to the normal values for the human skin, or a sufficient translucent material allows the blood flow of the finger underneath to be measured. Also, the addition of saliva to an artificial fingerprinting can mislead electrical conductivity checks, and when the analysis consists of mechanical properties, it is sufficient for the PAI to have the same as human skin, which is not difficult to achieve. Thus, although they appear robust, as the properties of a finger beyond fingerprint are virtually universal, these solutions are easily bypassed with some ingenuity of the attackers [44].

All hardware-based solutions also need additional software or upgrades in the existing software, so these methods are often referred to as hybrid solutions, however, either term is considered correct [57].

4.2 Software-based Presentation Attack Detection Methods

Software-only techniques, commonly referred to as software-based techniques, do not augment fingerprint sensing hardware and only use the existing designs, making use of the same type of data collected for fingerprint matching. These solutions only require updating the algorithms already running on the sensors, so they have low cost and can be applied almost universally. Hence, they have become very attractive and widely used, following the exponential growth of computer processing capabilities. Software-based approaches can exploit the dynamic behaviors of live fingerprints or the static properties of a single fingerprint image, that is, they may require a sequential set of 2D images or a single 2D sample per fingerprint, respectively [6, 51, 57].

4.2.1 Dynamic Solutions

Dynamic features are derived by processing multiple frames of the same fingerprint which are captured within a finite time interval, so, these methods analyze the time series of the fingerprint

images obtained by the scan, in spite of analyzing a single one (Figure 4.2). When observing these changes over time, it is possible to detect slight differences between a real finger and a PAI, as the material has different properties and characteristics. These approaches are divided into two main types: ridge distortion based and perspiration based [6, 44, 51].

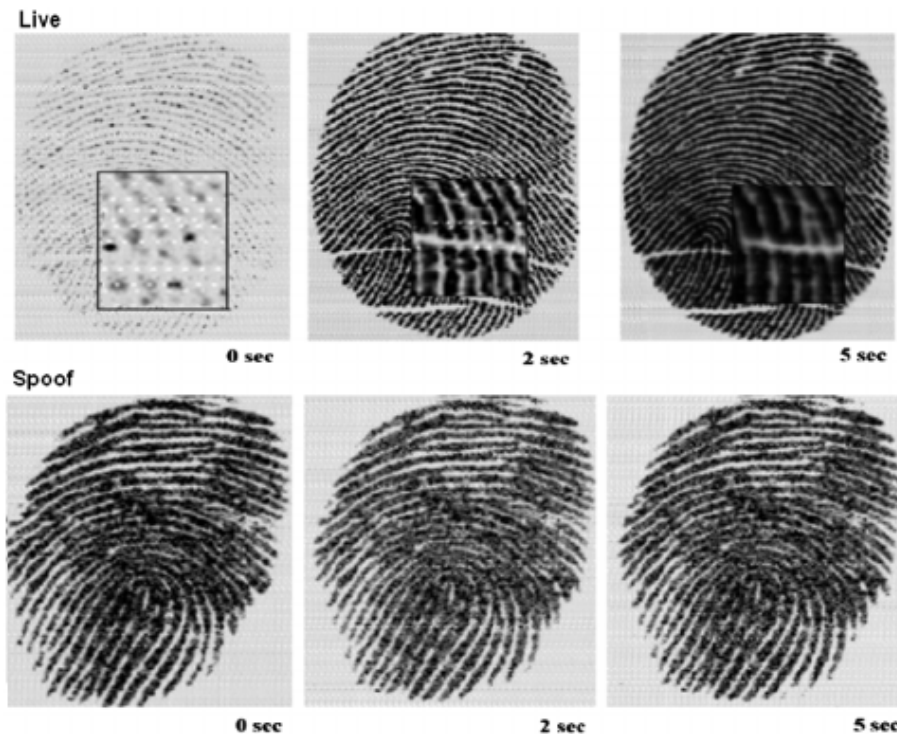


Figure 4.2: Multiple frame of a bona fide and presentation attack fingerprint. From Marasco et al. (2014) [51].

The ridge distortion based approaches rely on the fact that the distortion of a real finger, when pressed, is greater than that of a fake. The assessment of this distortion is achieved by pressing and slightly rotating the finger in the counterclockwise direction and at the same time acquiring a sequence of fingerprint frames at a high frame rate. The finger is considered to have no distortion at the beginning and the optical flows of the movements are calculated to obtain distortion maps, as well as minutiae displacement during the process. Therefore, the success of this technique depends on the quality of the extraction and matching of minutia, as well as on obtaining rigorous sequential scans [6, 44, 51].

The perspiration based approaches focus on detecting perspiration between the finger and the material of the sensor. Perspiration is a phenomenon typical of the human skin so when the finger is pressed, its pores start to release sweat that diffuses through the fingerprint ridges, making them look darker in the images and thus losing uniformity. As spoof materials don't have this property, showing high uniformity, the moisture patterns can be recognized and used as a way to differentiate live and fake fingerprints, mainly when analyzing the temporal evolution of sweat diffusion by obtaining sequential frames of the finger. To increase efficiency, this method requires

further investigation in various affecting factors, such as variability across subjects, environmental conditions and the pressure made by the finger [44, 51].

4.2.2 Static Solutions

Presentation attack detection software static solutions use only a two-dimensional image scanned by a classical two-dimension sensor and apply various transformations to it in order to detect slight discrepancies between bona fide and attack samples caused by differences in elastic properties or even minor imperfections present in the presentation attack instrument. As it only needs a single image, the same used for fingerprint matching, this type of method is the cheapest and fastest one [6, 51]. As seen in Figure 4.1, static solutions can be divided into four different types: perspiration-based, pore-based, texture-based and deep learning methods.

Perspiration-based approaches are also described in the dynamic solutions (Section 4.2.1), where the changes in the scan over time caused by the spread of sweat fluids are detected, however, this natural phenomenon can also be used with a single image, as the contact of the finger with the sensor instantly changes the state of sweat pores, which is recorded in the images and observed as different between real and a fake fingers. In fact, perspiration leads to different ridge signals, obtained with the skeleton of the ridge structure, and causes periodicity in the middle ridge signal, computed from the ridges' centers, a property not found in fake or cut off fingers [6]. Associated with this approach is the analysis of the pores present in the image. These structures are not always visible in a fingerprint scan, but the analysis of the spacing between them and of their relative position in the ridges indicate or not the existence of perspiration, which in turn indicates that the sample corresponds to a real or a fake finger, respectively [51].

On the other hand, texture-based approaches look for small differences between the texture properties of bona fide and presentation attack images, caused by the different characteristics of the materials in relation to the skin. These textural features may relate to the orientation, morphology or, for example, the smoothness of the image [51]. The analysis of an image texture can be through the ridge and valley signals, or by applying various texture enhancement techniques and computing texture descriptors to obtain features for the classification task to detect attack presentations. Among the possible texture-related features, there are simple examples, like first order statistics, such as the Entropy and the Median of the image and second order statistics like the gray-level distribution, as enumerated in Marasco et al. [51], or more advanced descriptors like the Local Binary Pattern, Local Phase Quantization or Binarized Statistical Image Features, all described in Marcel et al. (2019) [54].

More recently, there are the deep learning methods, with greater potential for good detection results. Deep Convolutional Neural Networks have been widely used for image classification with very good performance, however, they need a large amount of training data to achieve the best prediction results, which brings a problem when trying to detect a novel type of presentation attack instrument and even more so when it comes to unknown attacks. The CNN-based methods are also known as pure deep learning approaches or end-to-end solutions, as the input data is the original fingerprint image.

To avoid the need of large datasets, pre-trained neural networks, as well as transfer learning and domain adaptation methods have been increasingly used in the detection of presentation attacks, where smaller volumes of data can be handled [54]. Neural networks can switch to different domains and adapt to the characteristics of the new input data, and when a simple linear model is associated with a pre-trained network which works as a feature extractor, the accuracy of classification is very likely to increase [89]. The use of deep learning techniques to solve the PAD problem is increasingly found in recent research work and has enabled major innovations and developments in the proposed algorithms.

The scope of the present work is based on static methods, namely: i) the classification through handcrafted features and ii) the application of pure deep learning techniques. Therefore, the next sections present a description of several state of the art methods that also focus in these approaches for fingerprint PAD and some innovations in other areas of biometric liveness, such as iris PAD. Moreover, a third section summarizes the literature regarding unknown attacks, from their first reference to the most recent proposed methods.

4.2.2.1 Traditional Classification

Jin et al. (2010) [39] explored the potential of signals obtained through the patterns formed by the ridges and valleys, also known as middle ridge and middle valley line signals, together with the spectral energy of the image. The classification was implemented with Support Vector Machines (SVM) and Quadratic Discriminant Analysis (QDA). In the same year, Marasco et al. (2010) [52] extracted a large set of different types of features, like pore spacing and residual noise, which required some signal processing, various first-order statistics and also an unusual way of describing the intensity of the image, consisting of defining two ranges of intensities and computing the ratio of those ranges in relation to total number of pixels. To reduce the number of features actually used it was applied a Sequential Forward Selection technique, and, with this, it was possible to obtain a different set of features for each database or sensor available, maximizing the performance for all. This work also used SVM, as well as a Bayesian classifier, a Decision Tree and a Multilayer Perceptron to compare results.

Ghiani et al. (2012) [21] introduced the use of the Local Phase Quantization (LPQ) feature due to its blurring insensitivity being a possible advantage when detecting loss of information caused by the PAIs' manufacturing process. More specifically, it was used a set of multiple features derived from a rotation-invariant extension of LPQ to detect spectrum differences between live and fake samples, with results very similar to the more classical Local Binary Pattern (LBP) feature when applying an SVM for classification, but showing the potential of LPQ.

Similarly to LBP and LPQ, the Binarized Statistical Image Features (BSIF) characteristic, based on these and first employed in fingerprint PAD by Ghiani et al. (2013) [22], encodes the image texture on a feature vector. BSIF automatically learns a fixed group of filters just from a small collection of natural images to extract a large number of features, unlike its predecessors, which use handcrafted filters, an advantage that led to promising results, mainly when associated with other texture descriptors. The classification task in this work was in charge of an SVM algorithm.

In the same year, Gragnaniello et al. (2013) [29] introduced Weber Local Image descriptors to detect discontinuity zones and thus find minor differences between live and spoof fingerprints. The work joined this new feature with LBP to increase the algorithm's performance using the SVM classifier and Linear Discriminant Analysis (LDA).

The use of deep learning, more specifically Convolution Neural Networks (CNN), in fingerprint liveness detection was first introduced in Nogueira et al. (2015) [66], where a single layer CNN was trained to extract different patterns from the image, which were then used as input features for SVM classification. CNNs need large amounts of data to achieve good performances and to solve this, several images were created from each real image in the dataset through minor modifications, which greatly increased the number of images and at the same time helped the classifier to be more robust. This process, never before applied to fingerprint PAD, is called Data Augmentation and allowed the neural network to extract a large number of features, which was then diminished by a Principal Component Analysis (PCA).

Sequeira et al. (2015) [79] used a set of features derived from Gray Level Co-Occurrence Matrices (GLCM), which represents the frequency of co-occurrences of gray levels combinations in an image, and an innovative feature that joins together a Scale Invariant Feature Transform (SIFT) and LBP, called Weighted Local Binary Patterns (wLBP) [97]. This last feature, as the name indicates, results in weighted maps, producing thus a large dimensional feature. These features were then used as input for the classifiers in two different approaches: i) binary classification with the SVM classifier and ii) one-class classification, where the classifiers evaluated were a one-class SVM and Gaussian Mixture Models (GMM).

More recently, Jiang et al. (2018) [38] approached the problems of scale invariance and high dimensional features by using a Gaussian pyramid and LBP. A Gaussian pyramid is a group of images obtained by iteratively downsampling the original image with a Gaussian kernel operation. The classifier used was also the SVM.

Finally, González Soler et al. (2019) [25] extracted a series of texture descriptors, such as LBP, BSIF, a dense version of SIFT, Histogram of Oriented Gradients, Speed-Up Robust Features and more, conducted a Principal Component Analysis and modeled the feature distribution with GMM and Bernoulli Mixture Models (BMM). The modeled features were then encoded with a Fisher Vector before passing to the SVM classifier. This method avoided the use of the trending deep learning techniques and allowed the use of more classical machine learning classification methods by optimizing the separation between real and fake fingerprints labels with a more complex feature processing.

The possibilities regarding the features to use for fingerprint presentation attack detection are endless, with new features being discovered and adapted from other purposes constantly. Regarding the classification task, most of the described works use SVM because of its known good results. Deep learning is bringing up the quality and efficiency of the feature extraction process, but at the same time questioning the need for multiple classification stages with the called pure deep learning solutions, where the input is the original image and the output is the classification result, better described in Subsection 4.2.2.2.

4.2.2.2 Pure Deep Learning

The pure deep learning solutions, also known as end-to-end approaches, do not rely on pre-computed features from an image but instead the whole process is performed by the Artificial Neural Network. Among the several types of Artificial Neural Networks, the CNNs are increasingly used in classification tasks involving images, such as fingerprint PAD. In fact, most of the works submitted in the recent Fingerprint Liveness Detection Competitions are pure deep learning approaches.

Nogueira et al. (2016) [65] used various CNNs to detect spoofs and compared their performance with a CNN-based feature extractor followed by a PCA and SVM, and with LBP, PCA and SVM sequentially. To avoid the need for large data to train the networks, this work used Transfer Learning techniques to use pre-trained CNNs, modeled to recognize real-life objects, for fingerprint PAD. This Transfer Learning approach thus consisted of re-using a neural network that targeted a specific classification problem to solve a different problem. The features resultant from the first layer of a neural network are very general, somewhat similar to color blobs and Gabor filters, and thus can relate to almost every type of image, and only the features of the more internal layers are related to the type of image, but those layers depend all on the output of the general initial layer [94].

In Mirzaalian et al. (2019) [57], a more advanced sensing technique called Laser Speckle Contrast Image was used to acquire fingerprint images, which were then classified by five different neural networks, including a Long Short-Term Memory (LSTM) neural network, which shares the internal weights across time and has very good performances with sequential data. Thus, in this method, the input image was transformed into a one-dimensional array.

Engelsma et al. (2019) [13] introduced the concept of Adversarial Learning for fingerprint PAD with the utilization of a Generative Adversarial Network (GAN), which has a Generator and a Discriminator. The Generator tries to synthesize live fingerprint images in the training phase and the Discriminator tries to distinguish between real live samples and synthesized live samples, and in the testing phase, the Discriminator has learned all the characteristics of bona fide samples and can theoretically detect spoof attacks. Therefore, the Discriminator is a one-class classifier that learns the features of live fingerprints and labels as a presentation attack everything outside this class. On the other hand, the Generator tries to deceive the Discriminator in the training stage. The dataset used in this work was obtained by the authors through a device built for this purpose, obtaining images of 3 different types: normal, raw FTIR and processed FTIR. The obtained results were unsatisfactory, even with a rich and large dataset, so this approach proved to be very limited. Kim et al. (2019) [43] also used GANs but for generating a large dataset with synthesized fingerprints image. For this, an initial GAN generated a map of minutiae and a second GAN used that map to generate a fingerprint image. Additionally, an end-to-end solution for fingerprint PAD was proposed, using the synthesized images. The method consisted on dividing the image into patches and classifying each patch, with a Fully-Convolutional Neural Network, into one of three different classes: bona fide, presentation attack and background. The overall

fingerprint liveness score is then computed through an algorithm that uses the patches' predicted classes.

Furthermore, Yuan et al. (2020) [96] developed a model that can be divided into a stacked autoencoder for feature extraction and a deep learning classifier for fingerprint PAD. The autoencoder, in its turn, is divided into an encoder and a decoder. The encoder outputs features from the images, which are the input of the classifier, and the decoder tries to recreate the original image by using those same features. This way the encoder learns the best features to extract, for both the decoder and classifier. To avoid overfitting, data augmentation techniques are applied to the fingerprint images. On the other hand, Tolosana et al. (2020) [84] captured images within the short wave infrared spectrum (SWIR) and tested different end-to-end solutions, including Residual Networks trained from scratch and several pre-trained models adapted and finetuned for the purpose of fingerprint PAD. Finally, Liu et al. (2020) [46] trained an autoencoder with only bona fide images and used the differences in the reconstruction errors of bona fide and presentation attack samples to obtain a PAD score through a Gaussian Model. The images, obtained with an advanced sensing technique, were pre-processed to remove noise and the autoencoder has a Fine-grained Map architecture in order to refine the reconstruction error before entering the autoencoder. The resulting model obtained good robustness and generalization, but showed high dependence on a correct extraction of the region of interest (fingerprint).

The approaches described above demonstrate the most recent work based on pure deep learning to detect fingerprint presentation attacks, and, despite the referred limitations of neural networks, techniques like Transfer and Adversarial Learning have shown promising results that confirm the potential of deep learning models.

4.2.2.3 Unknown Presentation Attacks

New materials and ways of creating presentation attack instruments are constantly emerging, so even a system with high spoof detection performance can be tricked by using one of these, which can only be avoided if the algorithm is robust to any PAI species, even if it is not included in its training stage. The main goal of PAD is to detect a spoof and not which of spoof, so when developing a detection algorithm it is assumed that the samples used for training represent the complete population. Hence, the binary classifiers create a model that separate these samples from bona fide samples, which can be easily deceived when using a PAI that does not completely match the attack class characteristics. These unknown attacks have been identified as a problem for some years, but only more recently have appeared research works that tackle the detection of unknown attacks.

Marasco et al. (2011) [53] carried out a set of experiments comparing the detection accuracy of PAD methods varying the types of PAI in the training step. With these experiments, the authors were able to conclude that the performance significantly decreases when it comes to unknown materials, however, this work did not provide a solution to overcome the problem. Some years later, one approach to tackle the problem was proposed by Rattani et al. (2014) [74]. This work was based on the automatic adaptation of the algorithm when encountering new materials in the

operational stage, which improved the performance when the differences between the materials were small but did not constitute a solution to the problem when the new material was significantly different from the previous ones. Rattani et al. (2015) [75] proposed then a novel approach where a one-class SVM was combined with a binary SVM by a Weibull-calibrated formulation, which resulted in a better generalization for unknown materials caused by the improvement of the model in the decision boundaries.

In Sequeira et al. (2015) [79], the "leave one material out" strategy was used to show the vulnerabilities of biometric systems and a one-class classification strategy was implemented aiming at an increase of the robustness of the model. By concentrating efforts in learning the characteristics of real samples without having to consider another class, the model became more robust to unknown attacks in some situations. The experimental method employed in this work is described in Subsection 4.2.2.1.

The increasing awareness regarding unknown presentation attacks has led researchers to consider it when developing new detection algorithms, and several of the most recent works already attempted to develop approaches focused on generalizing and improving the robustness of the models regarding previously unseen spoof materials, most of them combining hardware and software based methods. Some of the solutions described in Subsection 4.2.2.2 were developed to solve this problem, with techniques such as Transfer Learning and Adversarial Training showing promising results but yet with many possible improvements.

Finally, the mention of unknown presentation attacks has not only been addressed in the fingerprint domain but also in other widely used biometric traces like face and iris. Although the type of information and the nature of the PAIs used for face and iris are significantly different from the ones in the fingerprint domain, the basic principles of PAD and of the problem of dealing with unknown spoofs are the same and thus many of the methods applied in these two domains can be transferred to fingerprint PAD with probable good performances.

A face PAI can be a video or photography, a face mask or even a sketch, and there are several methods to detect these spoofs, such as motion detection and face texture analysis. On the other hand, an iris PAI can be a printed contact lense, an artificial eyeball, a photography or a video. The detection methods include also motion and texture-based, as well as other approaches like spectrum or reflectance analysis [2]. The emergence of new types of spoof artifacts is also a problem yet to solve in all biometric domains, and increasingly complex methods have been recently carried out to generalize solutions. Chugh et al. (2018) [8] did not apply any regularization or advanced techniques, but conducted an intensive study on a set of 12 fingerprint PAI species (PAISp), concluding that 6 of them covered almost the entire feature space of all presentation attack samples. When training the model with only those 6 fake materials, it became generalized to all 12 types. The conducted study using a considerable number of PAI species was very interesting, however, the presented solution is only proved to be good for the 12 materials used, not being able to prove that it is robust to a new material that is not present in the dataset, as it may be represented outside the obtained feature map for presentation attacks, thus leading to detection errors. On the other hand, a novel approach for iris PAD, introduced in Ferreira et al. (2019)

[15] stands out due to its use of Adversarial and Transfer Learning, which is on the focus of this dissertation. In this work, an Artificial Neural Network called *species-classifier network* tries to learn the type of attack material (PAISp) used by receiving fingerprint latent representations from an *encoder network*. This second ANN tries to fool the first but also to provide, at the same time, good representations to a third ANN that learns to distinguish between live and fake iris, called *task-classifier network*. Therefore, the main goal is to force the *species-classifier network* to be a random predictor and minimize the error of the *task-classifier network*, making the detection of presentation attacks invariant to the PAISp used and, thus, robust to unknown attacks.

The emergence of generative models, especially the GANs in recent years, has led to a whole new domain of presentation attacks, as it has become possible to generate synthetic fingerprints almost indistinguishable from real samples or digitally modify fingerprints in order to deceive PAD systems. These modifications do not even need to be visible, as shown in Fei et al. (2020) [14], where several algorithms were studied to introduce perturbations in fingerprints, evaluating their ability to deceive PAD systems, as well as the robustness of the perturbed images regarding transformations such as rotation and resizing. These perturbations, not visible for those who observe the image, are sufficient for the PAD system to change its prediction from fake to real, imposing thus a significant threat to anti-spoofing solutions.

Therefore, the capability of detecting any type of spoof attack is a key characteristic for future fingerprint systems, which currently can be deceived, even with PAD algorithms implemented, by using previously unknown PAIs. This problem is now one of the main focus on fingerprint liveness detection, as well as in other biometric traces, with many developments in the last years but still with a large margin of improvement.

4.2.2.4 Performance of Fingerprint Presentation Attack Methods

The accuracy of a PAD algorithm is highly variant with the dataset used for evaluation and even with the type of attack samples presented. Showing a good performance regarding one PAI species is in no way a guarantee that the PAD method will have similar performance when facing a different PAI species. Overlooking the aforementioned problem, when the evaluation is done with PAI species that were known to the models in the training step, the performances are overly optimistic. Using a binary traditional approach, the performance of the solutions in current literature works has become increasingly better, with errors lower than 1%, as concluded in the results of the Liveness Detection Competition of 2019, regarding fingerprint. This is the most famous competition for PAD algorithms and its last edition was devoted to the integration of anti spoofing with matching algorithms, that is, to evaluate the performance of PAD solutions in a real-world application scenario [69]. The published results demonstrate the development stage of the present approaches, however, the problem of PAD generalization capacity to unknown PAI species still remains to be solved with optimal performances.

To better understand the performances achieved by the different works described in Subsections 4.2.2.1 and 4.2.2.2, a summary of the methods described, including the resultant evaluation metrics, is presented in Table 4.1.

4.3 Fingerprint Presentation Attack Detection Databases

One important tool for developing and improving PAD methods is the availability of a high quantity of data samples, either from bona fide presentations or from different attack materials. There is a lot of work going into solving this problem, however many of the databases used are not public [54]. Of those available to the public, three will be mentioned because of its frequent use in state of the art's research works:

- **Fingerprint LivDet Databases:** These large datasets were created for the various editions of the Liveness Detection Competition Series, hosted every two years since 2009. This competition has the main goal of comparing innovative PAD methods with vast quantities of data and standardized tests and is open to all institutions to submit their PAD methods [47, 54].
- **ATVS-Fake Fingerprint (FFp) Database:** With more than 3000 images from different types of sensors (optical, thermal and capacitive) and 68 fingers, this dataset has the additional feature of providing samples of both cooperative and uncooperative attacks [4, 54].
- **Michigan State University Fingerprint Presentation Attack Dataset:** This database was created to simulate fingerprint spoofing attacks and stimulate the creation of solutions for them. It offers tens of thousands of images of various types of cooperative attacks and bona fide samples collected by two different sensors [59].

4.4 Summary

The revision of the literature about fingerprint recognition and, in particular, fingerprint presentation attack detection resulted in an extensive description of more general topics such as the formation and characteristics of a fingerprint, how the different types of scans acquire images and the different ways an attacker can deceive these sensors and their recognition algorithms.

Furthermore, the possible approaches to detect spoofs, either hardware or software-based, were deeply understood and described, more specifically the software static approaches, which learn the differences between live and fake fingerprints with the same image used for individual recognition. The evolution of the solutions presented by research works for the last decade was described, demonstrating the progressive shift from conventional machine learning approaches relying on handcrafted features to the combination of learnt features with traditional classifiers ending in pure deep learning techniques, that is, ANNs based end-to-end solutions.

Although the fact that the research related to fingerprint PAD has been intensive in the last decade, the concern about unknown attacks has just started to be considered in novel implemented methods, demonstrating thus the need for improvements in the generalization and robustness of the algorithms.

Table 4.1: Summary of the software-static solutions for fingerprint PAD. When unknown-attack is not referred, the methodology only considered known attacks.

Name	Year	Features	Classifier	Method	Average results
Jin et al. (2010) [39]	2010	Ridge and valley signals, Spectral Energy	SVM, QDA	Live sample-based calibration	ACER=6.5%
Marasco et al. (2010) [52]	2016	Signal processing, first order, gray level	SVM, NB, Decision Tree	Sequential Forward Selection Technique	ACER=12.5%
Ghiani et al. (2012) [21]	2012	LPQ	SVM	Rotation invariant LPQ extension	EER=12.3%
Ghiani et al. (2013) [22]	2013	BSIF	SVM		EER=7.22%
Gragnaniello et al. (2013) [29]	2013	Weber descriptor, LBP	SVM, LDA		ACER=3.7%
Nogueira et al. (2015) [66]	2015	CNN extractor	SVM	Dataset augmentation, PCA	ACER=4.71%
Sequeira et al. (2015) [79]	2015	Weighted LBP, GLCM	SVM, GMM, One-class SVM	Segmentation, PCA, Unknown-attack	ACER=4%
Nogueira et al. (2016) [65]	2016	Various CNNs		Transfer Learning	ACER=2.9%
Jiang et al. (2018) [38]	2018	LBP, Gaussian Pyramid	SVM		ACER=21.2%
Mirzaalian et al. (2019) [57]	2019	LSTM and other CNNs		Laser Speckle Image, Unknown-attack	ACER=5.2%
Engelsma et al. (2019) [13]	2019	GAN		FTIR images, Adversarial Learning, One-class, Unknown-attack	TDR@(FDR=0.2%) = 49.8%
González Soler et al. (2019) [25]	2019	Multiple texture descriptors	SVM	PCA, GMM/BMM, Fisher Vector	ACER=2.8%
Liu et al. (2020) [46]	2020	Autoencoder and Gaussian Model		Advanced imaging (B-scans), Fine-grained Map architecture	TPR@(FPR=5%) = 93.18%

Chapter 5

Methodology

The experimental work developed for this dissertation follows a path that begins with a more traditional approach and gradually moves to pure deep learning methods, towards innovative contributions to the state of the art of fingerprint presentation attack detection. In short, a Support Vector Machines classifier was trained with handcrafted-features (Section 5.4.1), transiting then to the domain of deep learning by training a Multi-layer Perceptron (MLP) with the same features (Section 5.5), a method that was further regularized in an attempt to make it more robust to unknown attacks. The next steps turned the focus to pure deep learning solutions (Section 5.6) with Convolution Neural Networks working as end-to-end approaches, receiving fingerprint images and returning their classification as presentation attack or bona fide. These networks were also regularized in the same way as the MLP, using adversarial training techniques. Finally, the advantages of using generative models, more specifically Generative Adversarial Networks, as well as Transfer Learning techniques, for increasing the robustness of PAD fingerprint solutions were studied.

5.1 Evaluation Framework

In this experimental work, two different evaluation procedures were implemented, both considered to have high relevance for PAD algorithms. These different methods differ in how dataset images are divided into data for model training, validation, and testing. The training phase consists of providing a set of features/images and the label of the corresponding images so that the classifiers can create their prediction model, whereas in the testing phase the model is evaluated with a set of data that was not used in the train or the validation processes, that is, data unknown to the model. Regarding the validation, in deep learning classifiers, the validation data is different from the training data, and at the end of each training cycle the models are evaluated with the validation set, which is the reference to verify if the training is going well. In non-deep learning (machine learning) methods, the validation data is used to optimize the classifiers' parameters, as different parameters affect the performance of the models. This validation process is often implemented with a cross-validation algorithm, where the training data is divided into a pre-defined number of subsets and the ones used for training and validation change iteratively, as shown in Figure 5.1.

The end result of cross-validation score corresponds to the average value of all results from all iterations.

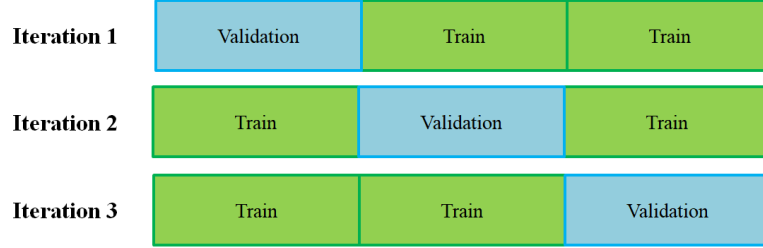


Figure 5.1: Example of data splits for cross-validation with three subsets and iterations.

Therefore, taking the description above into account, two types of methods were implemented to understand the effect of different PAI species (PAISp) on the classification performance, namely to investigate the capability of the models to detect unknown PAISp. Figure 5.2 is a visual representation of these two evaluation procedures: one-attack and unseen-attack.



Figure 5.2: Frameworks: One-Attack and Unseen-Attack. The train/validation and test data are represented by blue and red limits, respectively.

In the one-attack procedure, only one PAISp is considered and used for both training, validating and testing the model, whereas in the unseen-attack, one of the materials is left out and all the images correspondent to it form the testing set, in conjunction with bona fide images. In this way, the main goal is to evaluate the differences between the performance of a classifier with an attack known during training against an unknown attack: one-attack and unseen-attack, respectively.

5.2 Pre-processing

The pre-processing of an image, before it enters the feature extraction and classification steps, can include several operations to enhance it, which will help the models to learn better and improve their performance. These operations comprise resize processes, segmentation steps, or even morphological transformations that improve image quality.

The images from the used database have different background content, as shown in Figure 5.3, due to some sensors being capable of acquiring a scan focused on the fingerprint area whereas others scans always include content external to the finger.

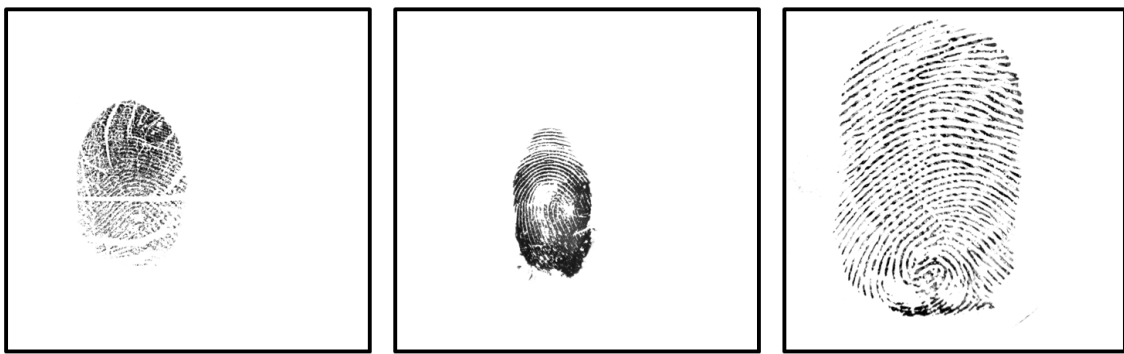


Figure 5.3: Example of fingerprint images with more (first and second) and less (third) background area comparing to the foreground area. From LivDet2015 database.

To solve this problem and evaluate the effect of the presence of background in the detection of presentation attacks, the images were segmented and then cropped in a way that the resulting image would be mostly filled with the fingerprint (region of interest) and no pattern content is lost. The segmentation was performed with the method described in Ferreira et al. [17], whose main steps are presented in Figure 5.4.

Firstly, the ridges of the fingerprint are enhanced with a range filter, followed by the clustering process with the Fuzzy c-means method and the merge of the resulting cluster to obtain a binarized image. Finally, the resulting foreground mask is computed by applying a series of post-processing morphological operations to the binary image [79].

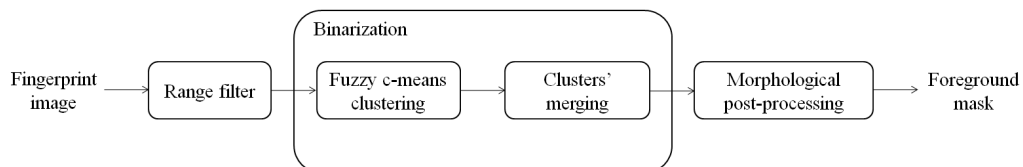


Figure 5.4: Flow diagram of the segmentation method. From Ferreira et al. [17].

5.3 Feature Extraction

During the feature extraction step were computed the histogram of the intensity, the histogram of the Local Binary Patterns and the histogram of the Local Phase Quantization as descriptors of each image. These three histograms were the concatenated to form the feature vector of the image. Therefore, globally for the entire dataset, was obtained a n_{image} by $n_{features}$ array, where n_{image} is the number of images and $n_{features}$ the size of the feature vector that describes each image. This number of values corresponds to the sum of the number of bins in the intensity, LBP and LPQ histograms. All three textural descriptors are fully explained in Chapter 2. This phase also constructs the output array with the labels of each image, with zero and one corresponding to bona fide and attack presentations, respectively. The process of assembling these two arrays is demonstrated in Figure 5.5.

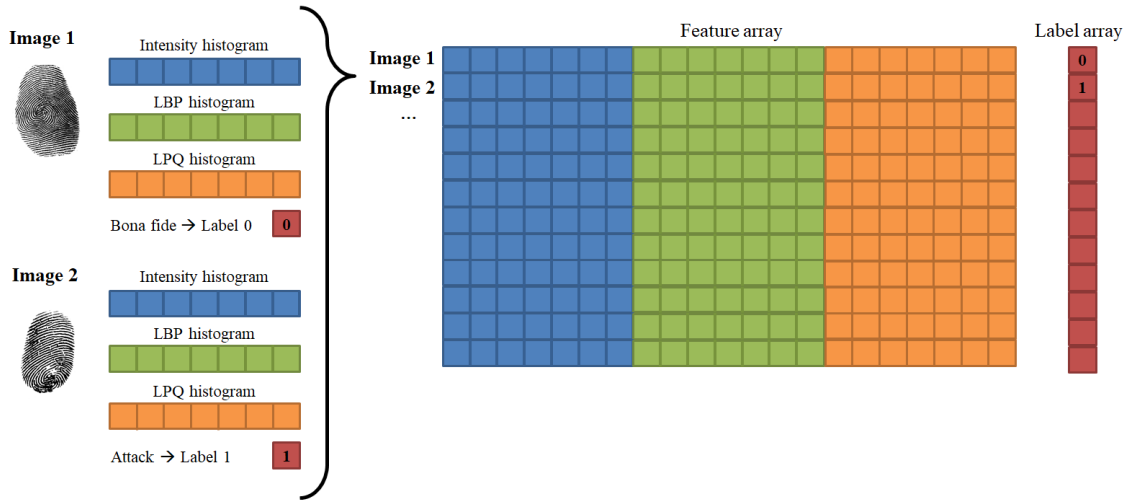


Figure 5.5: Assembly of the arrays for classification.

To evaluate the impact that the background has in detecting fingerprint presentation attacks, the features were extracted from both the original images and the images resulting from the segmentation and crop method described previously (Subsection 5.2). Figure 5.6 shows three examples of segmented images, the same ones of Figure 5.3.

5.4 Handcrafted Features with Machine Learning

In the work carried out in the preparation for this dissertation, several experiments were performed with three different Machine Learning classifiers: Support Vector Machines, Naive Bayes, and K-Nearest Neighbors. Despite the initial conclusions drawn about the best classifier being the KNN, a subsequent consolidation of the work revealed that the joint use of the three handcrafted features mentioned in Section 5.3 followed by a feature selection step (PCA) not only makes the SVM classifier much superior to the others but also improves the classification performance in general.



Figure 5.6: Segmented and cropped images correspondent to the samples shown in Figure 5.3.

Therefore, the first approach followed in this work was the association of handcrafted textural features with the SVM classifier. This method is then the baseline comparison for all further approaches, which is in line with what is observed in the state of the art publications.

5.4.1 Support Vector Machines



Figure 5.7: Scheme of the SVM classifier for PAD. X represents an image, h the corresponding handcrafted features and $\hat{y}$ the predicted class.

The SVM classifier, fully explained in Chapter 2, is the most common baseline method in the state of the art of fingerprint presentation attack, and in various recent and innovative methods, Convolutional Neural Networks are used only as feature extractors but the final classifier is still a SVM.

Figure 5.7 schematizes the framework of the first method of this dissertation. The input of the SVM classifier is the feature array resultant from the feature extraction phase, which was divided into training, validation, and testing. The SVM classifier is strongly affected by data normalization, so the training array was normalized and the resultant minimum and maximum values were then used to normalize the testing array. This way, it is guaranteed that both inputs (training and testing) are equally normalized. Additionally, a Principal Component Analysis (PCA) was performed to automatically select the features that better separate the two classes. The classifier is then optimized by iteratively changing some of its parameters and employing a 5-fold cross-validation, keeping the accuracy scores and then choosing the parameter values that brought the

highest value, which were then the ones used to train again and finally test the model. The parameters considered in the optimization and their respective range of values are specified in Table 5.1.

Table 5.1: SVM parameters evaluated in the optimization stage.

Parameter	Values
Kernel	"linear", "poly", "rbf", "sigmoid"
Regularization parameter	$[10^{-2}, 10^2]$ with 0.5 increment
Degree	$[1, 5]$ with 0.5 increment

5.5 Handcrafted Features with Deep Learning

After using a set of features extracted manually from the images and testing a conventional machine learning classifier for PAD, the potential of Artificial Neural Networks will now be evaluated using the same input data. Therefore, the following sections mark the beginning of the application of deep learning methods.

5.5.1 Multi-layer Perceptron

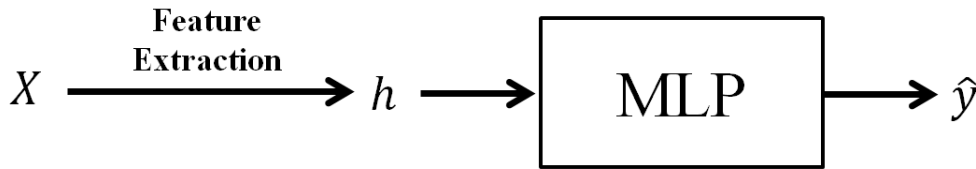


Figure 5.8: Scheme of the MLP classifier for PAD. X represents an image, h the corresponding handcrafted features and $\hat{y}$ the predicted class.

The first deep learning method evaluated in this dissertation follows the approach described in Ferreira et al. (2019) [15], originally developed for iris PAD, but using different handcrafted features. Instead of the Weighted LBP of the original version, this classifier uses the same input array as the one used for the experiments with the SVM (Section 5.4.1), thus comprising the histograms of intensity, of the LBP, and of the LPQ. This way is possible to compare the results of both methods.

The first ANN consisted of a Multi-Layer Perceptron (MLP) with three fully connected layers with 128, 128 and 2 neurons, and an input X of 275 values, which corresponds to the number of features per image. Therefore, the output of the model consists of 2 values, corresponding to the probabilities for each target class: bona fide and presentation attack. The final class prediction, $\hat{y}$, is obtained by choosing the class that obtained the highest probability (Figure 5.9).

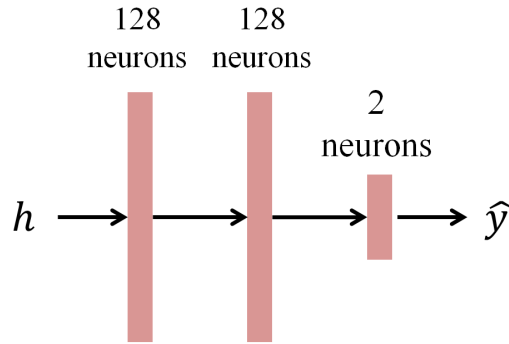


Figure 5.9: Architecture of the Multi-layer Perceptron used for fingerprint PAD. h represents the input array, and $\hat{y}$ the predicted class.

Each layer is followed by a Rectified Linear Unit (ReLU) activation function. The model was trained for 50 epochs with Cross Entropy as the loss function and the Adam optimizer with a learning rate of 10^{-4} and a weight decay of 10^{-4} (Table 5.2).

Table 5.2: Multi-Layer Perceptron training parameters.

Parameter	Values
Number of epochs	50
Loss function	Cross Entropy
Optimizer	Adam
Learning rate	10^{-4}
Weight Decay	10^{-4}

5.5.2 Regularization of the Multi-layer Perceptron

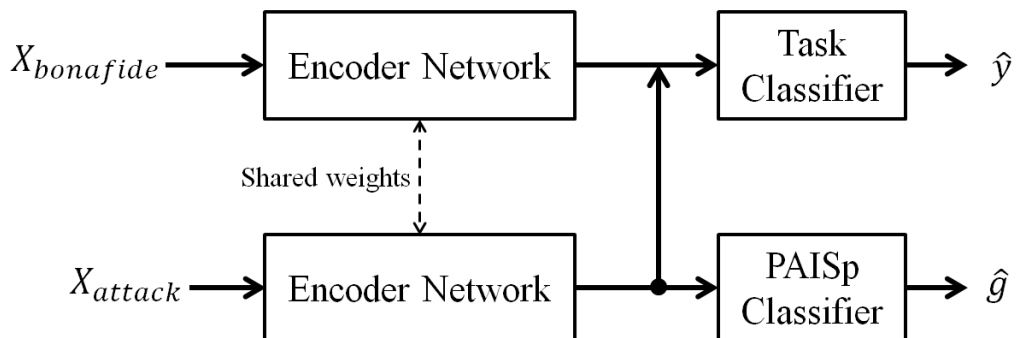


Figure 5.10: Scheme of the regularized Multi-layer Perceptron. X represents the input array, $\hat{y}$ the task prediction and $\hat{g}$ the PAISp prediction.

The next model, fully schematized in Figure 5.10, corresponds to a regularization of the previous MLP, as described in the original work developed for iris PAD [15], where an adversarial training and a PAISp-transfer training objectives were added. The main goal of the regularization

is to attempt a better performance with unseen attacks, and this part of the dissertation recreates the original work but now for fingerprint PAD and with a different set of handcrafted features. The model comprises a neural network called the “Encoder Network”, which maps the input array into a correspondent latent representation, and that array is then the input to two different networks: the “Task Classifier”, which aims to classify each input as bona fide or presentation attack, and the “PAISp Classifier”, which aims to classify a presentation attack sample into its correspondent presentation attack instrument. The Encoder Network tries to generate representations that worsen the performance of the PAISp Classifier but that are at the same time good for the Task Classifier, theoretically leading to a more general model that is less dependent on the PAISp and, consequently, more robust to unknown attacks. Additionally, the latent representations of presentation attack samples are encouraged to be PAISp-invariant, that is, the Encoder Network is stimulated to generate similar latent representations for all fake fingerprints, independently of their correspondent presentation attack instrument.

Referring to Figure 5.10, the Encoder Network uses the feature array X as input and outputs a latent representation h . The Task Classifier uses the latent representations h as input and outputs the binary classification $\hat{y}$ of bona fide or presentation attack. Finally, the PAISp Classifier uses also the representations h (only of fake samples) as input and outputs the prediction $\hat{g}$ regarding what is the PAI species. The learning objective can be summarized with the following loss function, where θ_h , θ_f and θ_g are the parameters of the Encoder Network, Task Classifier and PAISp Classifier, respectively:

$$\min\{\mathcal{L}(\theta_h, \theta_f, \theta_g)\} = \min\{\mathcal{L}_{task}(\theta_h, \theta_f) + \lambda \mathcal{L}_{species}(\theta_h, \theta_g) + \gamma \mathcal{L}_{transfer}(\theta_h)\} \quad (5.1)$$

$\mathcal{L}_{task}$ corresponds to the loss function of the presentation attack classification task, which is the Cross-Entropy between the predictions of the classifier and their respective labels. $\mathcal{L}_{species}$ is the Adversarial Loss, also denominated Soft Cross-Entropy Uniform Loss of the PAISp Classifier, which enforces the fake material predictions to be close to uniform, making this classifier similar to random guesser. Finally, $\mathcal{L}_{transfer}$ is the PAISp-transfer loss, which ensures that the latent representations of the fake fingerprints are as similar as possible. λ and γ are the respective weights of the Adversarial and Transfer losses. In conclusion, the Encoder Network and the Task Classifier are optimized together to minimize the loss function of Equation 5.1, and the PAISp Classifier is optimized to minimize its own loss function that corresponds to the Cross-Entropy between the PAISp label and the PAISp prediction.

Regarding the model architecture, each part of it is similar to the MLP without the regularization: the Encoder Network is constituted of three fully connected layers with 128 neurons each, the Task Classifier has three fully connected layers with 128, 128 and 2 neurons and the PAISp Classifier also has three fully connected layers, where the first two have 128 neurons and the last one has a number of neurons equal to the number of labels to predict, that is, the number of PAISp used for training. The training specificities were the same as the baseline MLP and can be seen in

Table 5.2.

The values of the Adversarial and Transfer Weights are important factors for the success of this regularization, so a series of experiments were conducted regarding these parameters. Firstly, the Transfer Weight (γ) was set to zero and the model was tested with the Adversarial Weight (λ) varying between 0 and 1, keeping the value that led to the best performance. Then, the Adversarial Weight was set to zero and the model was tested with the Transfer Weight varying between 0 and 1, keeping also the value that led to the best performance. Finally, it was conducted a test with both weights ranging between 0 and 1 with cross-validation, keeping the combination that led to the best results. These three different experiments evaluate the impact of having just one of the penalties or both, and the optimization of the values ensures that the model achieves the best possible performance. This set of experiments is summarized in Table 5.3.

Table 5.3: Evaluation and optimization of the Adversarial and Transfer Weights.

Experiments \ Values	Adversarial Weight (λ)	Transfer Weight (γ)
Just Adversarial Weight	[0, 1] with 0.1 increment	0
Just Transfer Weight	0	[0, 1] with 0.1 increment
Adversarial + Transfer Weight	[0, 1] with 0.1 increment	[0, 1] with 0.1 increment

5.6 Pure Deep Learning

The transition to pure deep learning, that is, without any handcrafted features, was conducted through a set of experiments with Convolution Neural Networks (CNNs). Figure 5.11 outlines the basic framework of the end-to-end solutions using CNNs.



Figure 5.11: Scheme of the end-to-end solutions (CNNs) for PAD. X represents an image and $\hat{y}$ the predicted class.

5.6.1 First Convolutional Neural Network

The first pure deep learning model was developed only for initial testing and understanding of the capabilities of CNNs. The ANN used here consisted of five convolutional layers and two fully connected layers, all followed by batch normalization, except the last one, whose output corresponds to the final classification of bona fide or presentation attack. The architecture of this CNN is schematized in Figure 5.12 and its training specificities are summarized in Table 5.4.

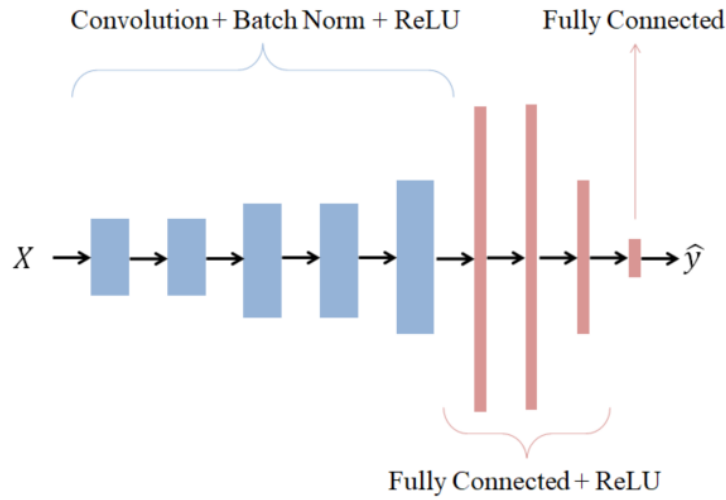


Figure 5.12: Architecture of the First Convolutional Neural Network. X represents the input image and $\hat{y}$ the predicted class.

Three approaches were evaluated with this model, regarding the input image:

- A. Entire image.
- B. Region of interest.
- C. Patches.

In the first case, the entire image is the input of the CNN, with all the background present. In the second approach, the images were segmented and cropped with the method described in Subsection 5.2, to obtain mainly the fingerprint content (region of interest). The fingerprints have different sizes inside the images, so the resultant cropped images have also different sizes. As this first proposed CNN has two fully connected layers, all input images must have the same size so that there are no mismatches with the internal dimensions of the model. To achieve this, two approaches were followed and evaluated:

- B.1. Add padding to all images to match the size of the larger image of the entire dataset.
- B.2. Crop all images to match the size of the smaller image of the entire dataset.

Finally, as observed in some state of the art publications, it is not uncommon to extract patches from the fingerprints and apply the presentation attack methods to the patches, so this was also evaluated in two different approaches:

- C.1. Concatenating a set of 10 patches (32x32) from the same image to classify the group as one.
- C.2. Classify each one of the 10 patches individually and use the majority rule.

Table 5.4: First CNN architecture details and training parameters.

Architecture	Values
Number of convolution layers	5
Convolution filters	[32, 32, 64, 64, 128]
Convolution kernels	[3, 3, 3, 3, 3]
Convolution stride	[1, 2, 1, 2, 1]
Convolution padding	[1, 1, 1, 1, 1]
Number of fully connected layers	2
Fully connected dimensions	[128, 2]
Training parameter	Values
Number of epochs	100
Loss function	Cross Entropy
Optimizer	Adam
Learning rate	10^{-4}
Weight Decay	10^{-4}

The main goal of this initial neural network was to evaluate the potential of this type of approach, as well as to evaluate different input formats and choose the most appropriate one for the following models presented in this dissertation.

5.6.2 VGG Neural Network

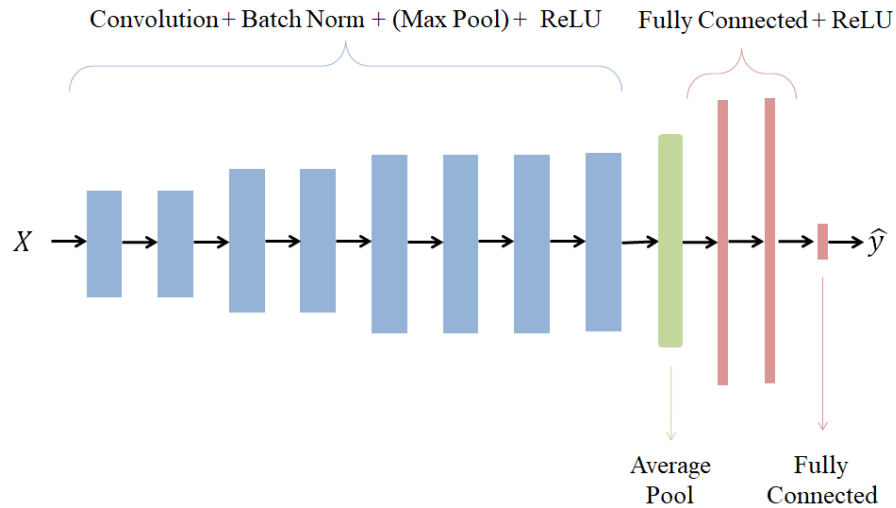


Figure 5.13: Architecture of the VGG11 network already modified for fingerprint PAD. X represents the input image and $\hat{y}$ the predicted class. Adapted from [81].

The second pure deep learning-based model used in this dissertation was the Convolutional Neural Network VGG11, with batch normalization [81]. This model was chosen due to its very good results in several different applications, including fingerprint presentation attack detection.

The number 11 comes from the 11 weighted layers that this version of VGG has, and although there are VGG networks with 19 layers, it was decided to follow with the least deep version. To match the model's architecture with the number of labels, the last fully connected layer of the VGG was modified to output two values instead of the 1000 values of the original network. The final architecture can be observed in Figure 5.13.

An initial evaluation led to the conclusion that fine-tuning the pre-trained (with the ImageNet dataset) VGG for fingerprint PAD achieves the best performance, so this was the version used in further experiments. This CNN has two main restrictions: the input images must have 3 channels and a minimum size of 224x224 pixels. To address the number of channels requirement, each image was stacked with two copies of itself, resulting in a three-channel image: the shape of each sample is changed from $1 \times \text{Width} \times \text{Height}$ to $3 \times \text{Width} \times \text{Height}$. Additionally, to address the minimum size requirements, the fingerprint images, already segmented and cropped, passed through the following processing:

Method 1 | Step 1 - Center cropping to match the size of the smaller image in the dataset.

Method 1 | Step 2 - Resizing the images to match the minimum 224x224 size but maintaining the ratio between the width and the height.

The resulting images had no background content, which is positive, but the resize operation can distort the fingerprints and affect the performance. Therefore, another processing method was conducted as an alternative to the previous one:

Method 2 | Step 1 - Center crop both dimensions to match the 224 required size in the smaller one, but maintaining the ratio between the two dimensions.

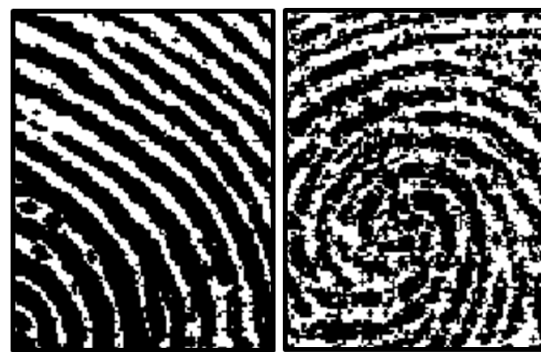
Method 2 | Step 2 - Center crop the larger dimension to match the 224 required size.

Therefore, the resulting images do not pass through a resize operation, but have some background content, which can also affect the performance of the model.

From now on, the first approach (Method 1), which uses center cropping and resizing, will be called "**Resize**" and the second one (Method 2), which only uses center cropping, will be called "**Crop**" (Table 5.5). Figure 5.14 shows a set of images processed with these two approaches. The model was trained and tested with both these methods so that it was possible to conclude which one brings the best performance. The specificities of the training conducted with the VGG11, as well as its architecture specifications are shown in Table 5.6.

Table 5.5: Summary of the input processing approaches.

Method code name	Description
Resize (method 1)	Crop to match the size of the smaller image and then resize
Crop (method 2)	Two cropping operations to match the size requirement



(a) Resize.



(b) Crop.

Figure 5.14: Examples of images after the two pre-processing methods for the VGG model: Resize and Crop.

Table 5.6: VGG11 architecture details and training parameters.

Architecture	Values
Number of convolution layers	8
Convolution filters	[64, 128, 256, 256, 512, 512, 512, 512]
Convolution kernels	[3, 3, 3, 3, 3, 3, 3, 3]
Convolution stride	[1, 1, 1, 1, 1, 1, 1, 1]
Convolution padding	[1, 1, 1, 1, 1, 1, 1, 1]
Max Pooling	[None, (kernel = 2; stride 2), None, (kernel = 2; stride 2), None, None, None, (kernel = 2; stride 2)]
Average Pooling output size	7x7
Number of fully connected layers	3
Fully connected dimensions	[4096, 4096, 2]
Training parameter	Values
Number of epochs	100
Loss function	Cross Entropy
Optimizer	Adam
Learning rate	10^{-4}
Weight Decay	10^{-4}

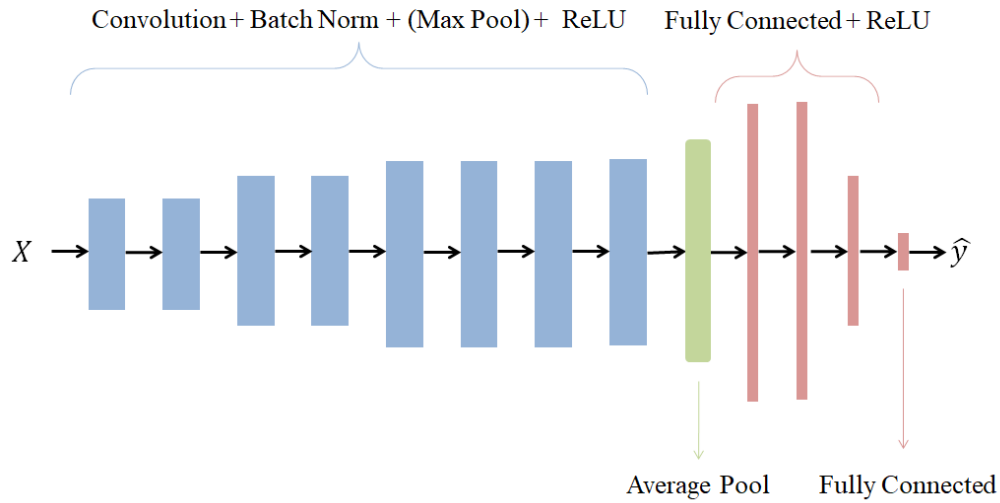


Figure 5.15: Architecture of the Second Convolutional Neural Network (CNN2). X represents the input image and $\hat{y}$ the predicted class.

5.6.3 Second Convolutional Neural Network

After the adaptation of the VGG11, a second custom CNN (CNN2) was developed and trained from scratch for a direct comparison with the pre-trained model. This CNN is very similar to the VGG19 [81], the more complex VGG available, but with only the first eight convolutional layers. Additionally, instead of replacing the last fully connected layer of 1000 neurons with 2 neurons, a new layer was added to output the two value prediction, so the CNN2 model has four fully connected layers instead of three (Figure 5.15). This higher number of fully connected layers was defined to compensate the smaller number of convolutional layers in comparison to the VGG19, and proved to be advantageous after initial experiments.

This final architecture of the CNN2 network was defined after a set of initial experiments where various possible architectures were tested through cross-validation. The complete characteristics of this model, as well as its training parameters, are summarized in Table 5.7.

This model was tested with different experiments to optimize its performance. Regarding the architecture, three different sets of kernel sizes were used in the convolution layers:

1. All 3x3.
2. All 5x5.
3. One 7x7 kernel followed by three 5x5 kernels and then four 3x3 kernels.

The last approach was tested because different kernel sizes may catch different characteristics of the fingerprints to help distinguish between bona fide and presentation attack samples. Additionally, the input images for this network were also pre-processed with the two methods described

Table 5.7: CNN2's architecture details and training parameters.

Architecture	Values
Number of convolution layers	8
Convolution filters	[64, 64, 128, 128, 256, 256, 256, 256]
Convolution stride	[1, 1, 1, 1, 1, 1, 1, 1]
Convolution padding	[1, 1, 1, 1, 1, 1, 1, 1]
Max Pooling	[None, (kernel = 2; stride 2), None, (kernel = 2; stride 2), None, None, None, (kernel = 2; stride 2)]
Average Pooling output size	7x7
Number of fully connected layers	4
Fully connected dimensions	[4096, 4096, 1000, 2]
Training parameter	Values
Number of epochs	100
Loss function	Cross Entropy
Optimizer	Adam
Learning rate	10^{-4}
Weight Decay	10^{-4}

for the VGG network, and the model was tested to discover which one is the best approach regarding the input images (Table 5.5). The summary of this optimization experiments can be observed in Table 5.8.

Table 5.8: CNN2's parameters that were evaluated and optimized. The pre-processing methods are the same that were applied to the VGG (Table 5.5).

Parameter to optimize	Values
Convolution kernel sizes	[3, 3, 3, 3, 3, 3, 3, 3]
	[5, 5, 5, 5, 5, 5, 5, 5]
	[7, 5, 5, 5, 3, 3, 3, 3]
Input images pre-processing	Resize
	Crop

5.6.4 Regularization of the Second Convolutional Neural Network

The CNN2 was the baseline model to apply the same regularization that was applied to the MLP model (Section 5.5.2). Therefore, a new model was developed (CNN2_{reg}) following the methodology of the regularization explained before. The eight convolutional layers of the CNN2 correspond to the Encoder (Feature Extractor) of the CNN2_{reg}, with the last one being regularized with the Transfer Loss to force similar representations for different presentation attack instruments. The two classifiers (Task and PAISp) of the CNN2_{reg} have the same architecture of the fully connected layers of the baseline CNN2. The overall architecture and of this regularized version of the CNN2 can be analyzed in Figure 5.16 and its specificities are the same as the baseline CNN2 (Table 5.7).

This approach is then a transition of the regularization from handcrafted features to pure deep learning and Deep Convolutional Neural Networks, which was already evaluated but for a different

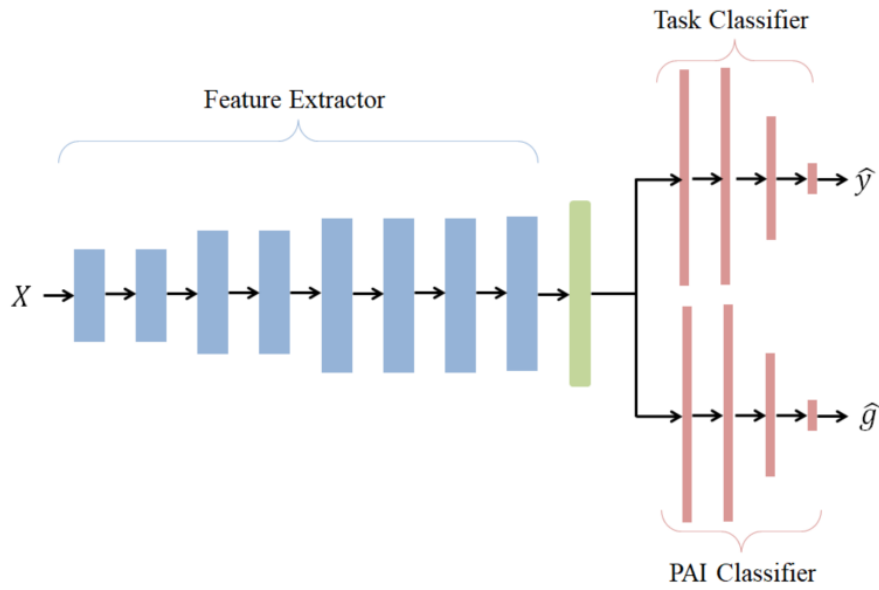


Figure 5.16: Scheme of the regularized Second Convolutional Neural Network. X represents the input image, $\hat{y}$ the task prediction and $\hat{g}$ the PAISp prediction.

objective, in Ferreira et al. (2020) [16]. This work applies the regularization to a sign language translator to try to increase the robustness to different signers. Hence, the Task Classifier focuses on the sign itself and the Adversarial Classifier, supposed to perform bad, focuses on the signer that is making the gestures. Apart from having a different goal, area of application and using a different network architecture, this approach applied to fingerprint PAD has an important difference from the original work: for the sign language objective, the Adversarial Classifier looks at all images, as all gestures have an associated signer, whereas in the case of fingerprints, only the fake ones have an associated fabrication material, so the Adversarial Classifier only looks at the images corresponding to presentation attacks. Taking into account the similarity with the regularization of the MLP, the CNN2_{reg} was also evaluated and optimized concerning the weights of the Adversarial and Transfer losses. These experiments followed the same protocol, which is summarized in Table 5.3.

This proposed methodology, innovative in its application for fingerprint presentation attack detection, led to the submission of a paper for the 19th International Conference of the Biometrics Special Interest Group (BIOSIG 2020).

5.6.5 One-class Generative Adversarial Network

GANs are one of the most emerging types of Artificial Neural Networks today, managing to generate good quality images from just random noise inputs. Therefore, it was decided to study the capabilities of these models regarding: i) generation of fingerprint images and ii) fingerprint presentation attack detection.

The first GAN-based model developed in this dissertation corresponds to a Deep Convolutional GAN with a Pix2Pix Discriminator, also known as Patch Discriminator [34]. This classifier

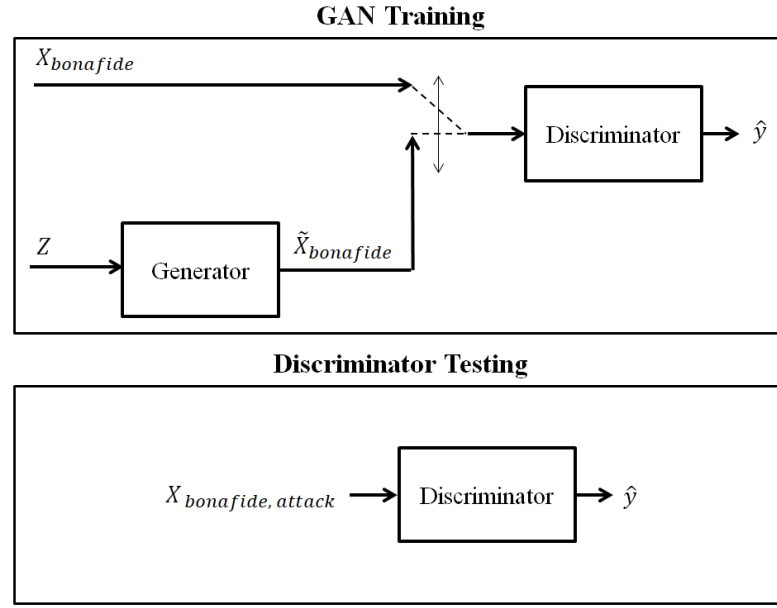


Figure 5.17: Scheme of the proposed GAN for fingerprint PAD. $X_{bona fide}$ and $\tilde{X}_{bona fide}$ are the original and generated bona fide images, respectively, Z is the random noise Generator's input and $\hat{y}$ is the output of the Discriminator.

outputs a matrix of probabilities instead of a single probability value, which is the same as classifying various patches of the same image instead of the entire image as one. For this, the loss function has to be computed for each one of the matrix elements individually. This choice for the Discriminator network was taken after a series of initial experiments in which it proved to be advantageous to classify images with this approach, including for a better quality image generation.

The idea of this model is to train the Generator network to learn the distribution of bona fide fingerprints and generate images as similar as possible to them, and the Discriminator to distinguish between the real bona fide ($X_{bona fide}$) and the generated bona fide ($\tilde{X}_{bona fide}$). After enough training, the Discriminator network is tested with a new set of bona fide and presentation attack samples. Therefore, in the test phase the Discriminator works as a one-class classifier and its output corresponds only to the probability of a sample being bona fide or not being bona fide, since during training only real samples were used. Theoretically, if the Discriminator has to be capable of distinguish between real bona fide and generated bona fide fingerprints, it automatically becomes capable of detecting presentation attack samples. This method follows the solution presented by Engelsma et al. (2019) [13] but with a different dataset and using Patch Discriminator. The training and testing framework is summarized in Figure 5.17.

Due to memory capacities and in order to facilitate the processing inside the neural networks, all images have been processed to have dimensions of 128x128, which is the minimum dimension for which the details remained visible. However, in order to maintain the original width-to-height ratio and thus not distort fingerprints, the images went through the following processing:

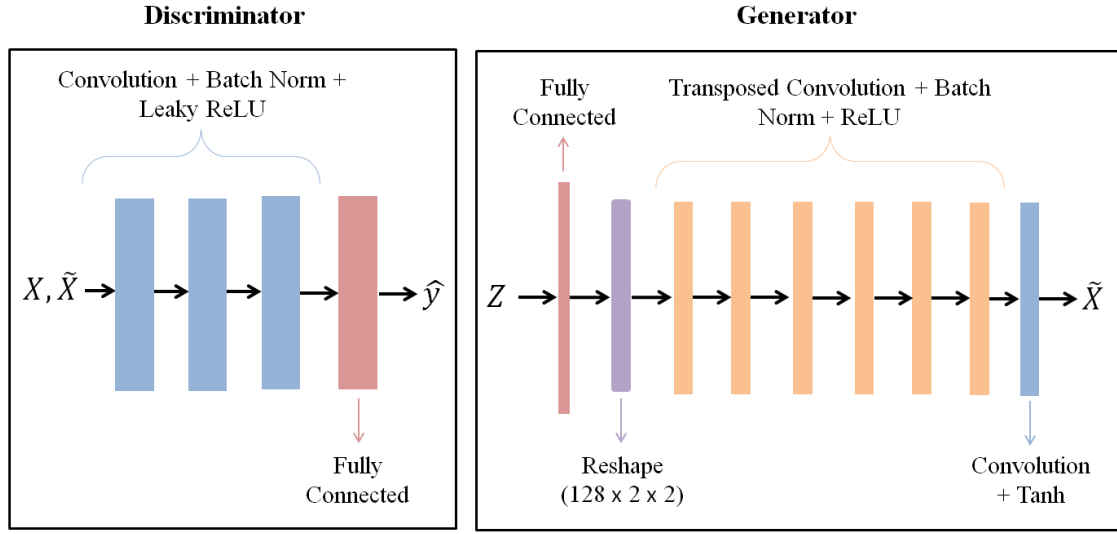


Figure 5.18: Scheme of the proposed GAN's architecture. X represents an original image, $\tilde{X}$ a generated image, Z a random noise and $\hat{y}$ the output of the Discriminator.

Step 1 - Add centered padding so that the two dimensions (width and height) have the same value, thus obtaining a square image.

Step 2 - Resize the image to 128x128 pixels.

For the initial tests with this model, the images were simply center cropped to match the 128x128 dimension, so only a small sample of the entire fingerprint is used. The variability of patterns and the few information provided to the model (the images did not even have the circular/ovular shape characteristic of the fingerprints) did not allow the Generator to learn the distribution of the training samples. Thus, it was decided to follow the approach described above.

The Discriminator's loss is the same as explained in Section 2.3.3 of the background knowledge chapter, given by Equation 2.3.3. On the other hand, the loss of the Generator (Equation 2.12) was altered by adding a diversity term (d_t) to the expression, another innovation in relation to the work developed in Engelsma et al. (2019) [13]. This term, proposed by Yang et al. (2019) [93], forces the Generator not to ignore the input noise and produce more diverse outputs instead of becoming restricted to the same ideal output and generating it regardless of the input (mode collapse). The diversity term depends on two different input noises and their corresponding Generator's output:

$$d_t = \frac{\|z_2 - z_1\|_2^2}{\|G(z_2) - G(z_1)\|_2^2 + \epsilon} \quad (5.2)$$

Where z_i and $G(z_i)$ are the inputs and their corresponding Generator's output. Therefore, the numerator is equal to the squared Euclidean 2-norm between the two inputs and the denominator is equal to the squared Euclidean 2-norm between the two corresponding outputs. If two outputs are exactly the same (distance 0) for two different inputs, the denominator will be equal to ϵ ,

which should be a small value, that is, the diversity term will be very high. Likewise, the more different the outputs of the Generator are, for equally different inputs, the smaller the diversity term is. Therefore, by adding this term to the loss function the Generator is forced to diversify its outputs.

Table 5.9: Proposed one-class GAN's architecture details and training parameters.

Discriminator's architecture	Values
Number of convolution layers	3
Convolution filters	[128, 128, 128]
Convolution kernels	[5, 5, 5]
Convolution stride	[2, 2, 2]
Convolution padding	[0, 0, 0]
Number of fully connected layers	1
Fully connected dimensions	[1]
Generator's architecture	Values
Number of fully connected layers	1
Fully connected dimensions	[512]
Number of transposed convolution layers	6
Transposed convolution filters	[128, 128, 128, 128, 128, 128]
Transposed convolution kernels	[4, 4, 4, 4, 4, 4]
Transposed convolution stride	[2, 2, 2, 2, 2, 2]
Transposed convolution padding	[1, 1, 1, 1, 1, 1]
Number of convolution layers	1
Convolution filters	[1]
Convolution kernels and stride	[1]
Convolution padding	[0]
Training parameters	Values
Number of epochs	500
Loss function	Cross Entropy
Optimizer	Adam
Learning rate	10^{-4}
Weight Decay	10^{-4}

Regarding the architecture, the Discriminator comprises three convolutional layers followed by batch normalization and a Leaky ReLU with a negative slope of 0.2 as activation function. The last layer is a fully connected layer that converts the number of channels to 1, and it is not followed by any activation function, as the loss function used already applies the Sigmoid function to the outputs. In its turn, the Generator begins with a fully connected layer and batch normalization, followed by six transposed convolution layers, each one followed by batch normalization and a ReLU activation function. The last layer is a convolutional layer that converts the feature map into a 1 channel 128x128 image, followed by a Hyperbolic Tangent activation function. This function outputs all pixels between -1 and 1, so the real samples should be normalized to match this range. The input of the Generator is a random noise array of 100 elements with a normal distribution of mean 0 and standard deviation 1. After a set of initial experiments, the minimum number of

epochs for the Generator to output fingerprint images of all datasets was defined as 500. The architecture details and training parameters of the model are summarized in Figure 5.18 and Table 5.9.

5.6.6 Image-to-image Generative Adversarial Network

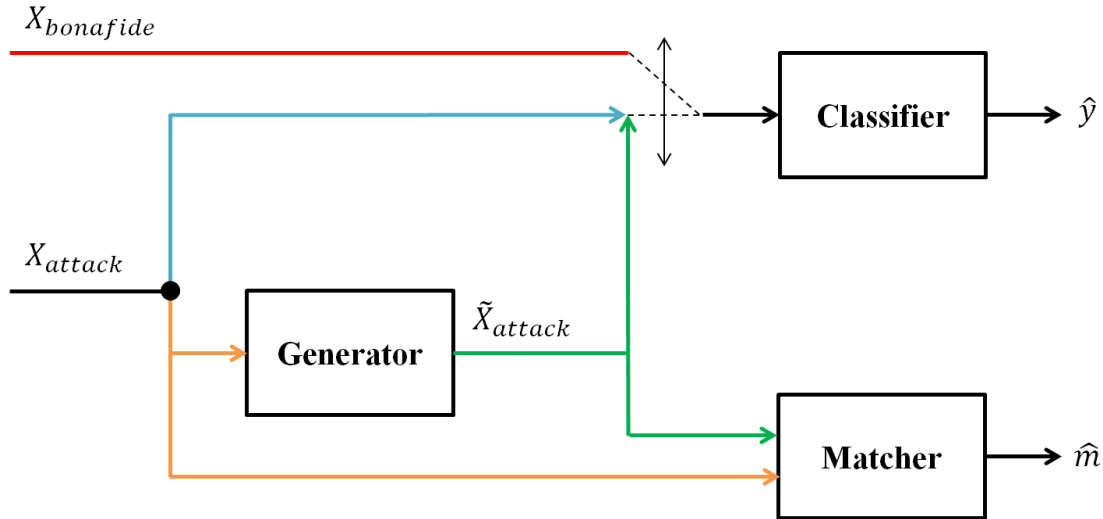


Figure 5.19: Scheme of the proposed image-to-image GAN for fingerprint PAD. $X_{bonafide}$ represents the bona fide images, X_{attack} the original fake images, $\tilde{X}_{attack}$ the modified fake images, $\hat{m}$ the matching score and $\hat{y}$ the output of the Classifier. The blue and orange lines are the attack samples not to be modified and the attack samples to be modified, respectively.

The purpose of a presentation attack is to deceive a biometric system into thinking that the presented sample is bona fide, either being allowed to enrol with that fake identity or to impersonate an authorized user, thus passing the security as another individual. Thus, the more real the scan of a false fingerprint appears, the greater the likelihood that it will deceive the system, so the development of new PAISp focuses on trying to make them more similar to bona fide fingerprints.

This is the main idea behind the last PAD method of this dissertation: instead of just using the capabilities of GANs to generate new fingerprints, it was decided to use the image-to-image translation capabilities of these models to make presentation attacks more similar to bona fide samples, thus forcing the system to be more restricted when learning to distinguish the two classes. Theoretically, if the Classifier is trained with a greater variety of PAISp, including samples more similar to bona fide fingerprints, it will become more robust to new and unknown presentation attacks. This “training enrichment” approach is possible through GANs, in which the Generator does not receive random noise as input but an image of a fake fingerprint, modifying it to become more similar to the real fingerprints of the training dataset. An image-to-image Generator must have an encoder and a decoder part so that the output has the same dimensions of the input. These models are used in numerous applications, from segmentation tasks to image colorization [34].

In summary, this model comprises a Generator and a Classifier/Discriminator network, just like a normal GAN. The Generator receives an image of a presentation attack and tries to modify it to become similar to the bona fide samples and the Discriminator tries to distinguish between bona fide and presentation attack samples. Hence, the Discriminator is a PAD classifier that is trained dynamically alongside the Generator, in an adversarial framework. With enough training, the Generator network should learn to output fake samples very similar to the real ones, and the Classifier has to be more restrictive, which may make it more robust to unknown attacks. The losses of these two adversarial ANNs ($\mathcal{L}_{Classifier}$ and $\mathcal{L}_{Generator}$) are given by Equations 5.3 and 5.4, where $\mathcal{L}_{task}$ represents the loss between the predictions and the ground truth, $\mathcal{L}_{adv}$ is the adversarial loss of the Generator, $\hat{y}_{X_{bonafide}}$ is the output of the Classifier for bona fide samples, $y_{X_{bonafide}}$ is the bona fide ground truth label, $\hat{y}_{X_{attack}}$ is the output of the Classifier for all presentation attack samples, $\hat{y}_{\tilde{X}_{attack}}$ is the output of the Classifier for modified presentation attack samples, and $y_{X_{attack}}$ is the presentation attack ground truth label. The overall framework of this model can be observed in Figure 5.19.

$$\mathcal{L}_{Classifier} = \min\{\mathcal{L}_{task}(\hat{y}_{X_{bonafide}}, y_{X_{bonafide}}) + \mathcal{L}_{task}(\hat{y}_{X_{attack}}, y_{X_{attack}})\} \quad (5.3)$$

$$\mathcal{L}_{Generator} = \max\{\mathcal{L}_{adv}(\hat{y}_{\tilde{X}_{attack}}, y_{X_{attack}})\} \quad (5.4)$$

To include some of the original presentation attack samples, just a random half of them are fed into the Generator to be modified, the other half remaining unchanged. The fake samples, modified and not modified, are then joint and fed together into the Discriminator, so this approach can be considered a data augmentation of the fake fingerprints, where each sample has a 50% chance of entering the Generator and being modified.

Additionally, if the Generator is not regularized, it may be capable of finding an ideal modification that always deceives the Classifier and output always the same image, ignoring the input fingerprint, which is not the goal of this model since it is important to maintain the biometric identity of the samples. To avoid this possibility, a third part is added to the framework: the fingerprint Matcher. This Matcher could be an ANN, pre-trained ou trained from scratch alongside the Generator and the Classifier, but for this dissertation it was decided to use a publicly available method [37] that only uses image processing techniques to obtain a matching score between two fingerprints. As seen in Figure 5.20, the Matcher algorithm enhances the input images and detects the fingerprint's minutiae for comparison, computing the final score from the number of mismatches.

The lower the score, the closer the two fingerprints are. Therefore, every modified fingerprint image should be matched with the corresponding original image, and a high matching score (biometric identity is changed) should penalize the Generator. To achieve this, the matching score, $\hat{m}$, was added to the Generator's loss:

$$\mathcal{L}_{Generator} = \max\{\mathcal{L}_{adv}(\hat{y}_{\tilde{X}_{attack}}, y_{X_{attack}}) - \hat{m}\} \quad (5.5)$$

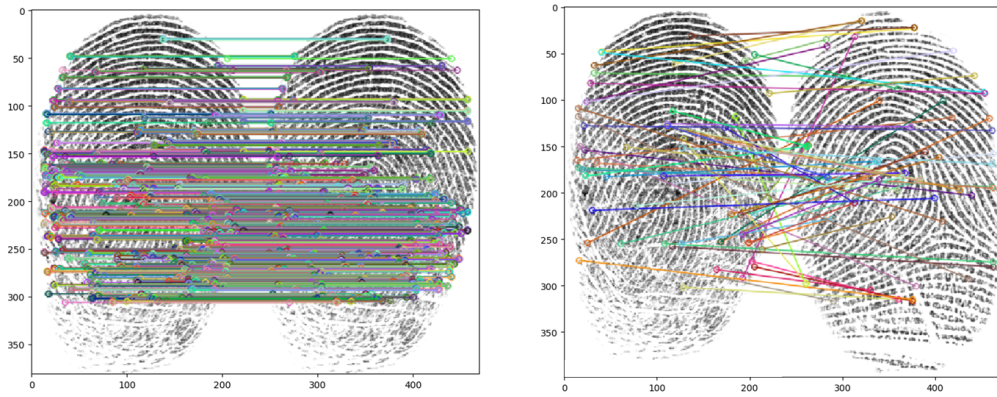


Figure 5.20: Matcher's analysis with the detected minutiae. The left and right graphs correspond to two equal and two different fingerprints, respectively.

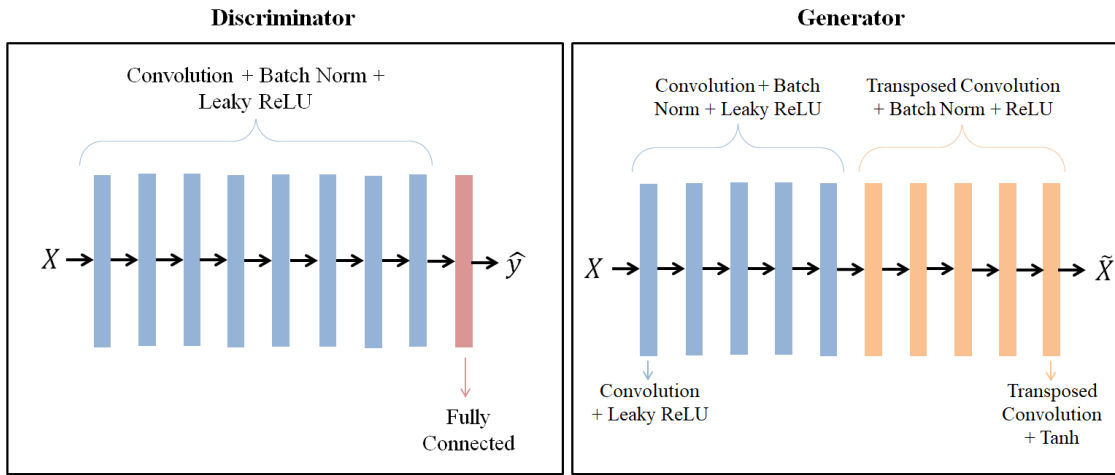


Figure 5.21: Scheme of the proposed image-to-image GAN's architecture. X represents an image, $\tilde{X}$ the corresponding modified image and $\hat{y}$ the output of the Discriminator.

Regarding the architecture, the Classifier is similar to a Patch Discriminator, like the one described in Subsection 5.6.5, so the output is a matrix of PAD scores. This network has eight convolutional layers followed by a LeakyReLU activation with negative slope equal to 0.2. After them, there is a fully-connected layer that converts the number of channels to 1 in order to obtain a 2D matrix as output. This last layer is not followed by any activation function, as the loss function used already implements a Sigmoid activation to the outputs. In its turn, the Generator network can be divided into two parts: the encoder and the decoder. The encoder is made by 5 convolutional layers, all followed by batch normalization and a LeakyReLU activation function with a negative slope of 0.2. The decoder is the symmetric of the encoder, so it has 5 transposed convolutional layers, all followed by batch normalization and with ReLU activation, apart from the last layer, which is not followed by batch normalization and has a Tanh activation. Similarly to the first GAN developed in this dissertation (Section 5.6.5), the Generator of this model needs a significant amount of time to learn to modify the fingerprints but maintaining the same resolution

Table 5.10: Proposed image-to-image GAN's architecture details and training parameters.

Discriminator's architecture	Values
Number of convolution layers	8
Convolution filters	[64, 64, 128, 128, 256, 256, 256, 256]
Convolution kernels	[3, 3, 3, 3, 3, 3, 3, 3]
Convolution stride	[1, 2, 1, 2, 1, 1, 1, 2]
Convolution padding	[0, 0, 0, 0, 0, 0, 0, 0]
Number of fully connected layers	1
Fully connected dimensions	[1]
Generator's architecture	Values
Number of convolution layers (encoder)	5
Convolution filters	[64, 64, 64, 64, 64]
Convolution kernels	[3, 3, 3, 3, 3]
Convolution stride	[2, 1, 1, 1, 1]
Convolution padding	[1, 1, 1, 1, 1]
Number of transposed convolution layers (decoder)	5
Transposed convolution filters	[64, 64, 64, 64, 64]
Transposed convolution kernels	[3, 3, 3, 3, 3]
Transposed convolution stride	[1, 1, 1, 1, 2]
Transposed convolution padding	[1, 1, 1, 1, 1]
Training parameters	Values
Number of epochs	500
Loss function	Cross Entropy
Optimizer	Adam
Learning rate	10^{-4}
Weight Decay	10^{-4}

and the same content in terms of the ridge-valley patterns. Initial tests showed that this Generator needs also at least 500 training epochs to satisfy the requirements. Furthermore, as it takes some epochs to start returning clear images and with a defined ridge-valley pattern, the matching regularization is only applied in the last 125 epochs of the training routine, functioning in this way as a final finetuning of the Generator. The architecture details and the training parameters of this ultimate model are summarized in Figure 5.21 and Table 5.10.

5.7 Summary

The experimental protocol developed in this dissertation is a journey that goes from traditional machine learning techniques and ends with more complex deep learning frameworks, always with the goal to address the development of methods for fingerprint PAD.

The first developed method comprised the extraction of textural features and the classification through conventional machine learning algorithms, such as the SVM classifier. Then the approaches entered the field of deep learning, using the same handcrafted features with an MLP

as a classifier. This MLP was later regularized with adversarial training and transfer learning (domain adaptation) techniques, in an attempt to increase the robustness to unknown attacks. Up to this point, the solutions developed can be divided into two parts: feature extraction and classification. Therefore, this dissertation moved next to the field of pure deep learning, that is, end-to-end solutions, in which an image enters the model and it returns the prediction regarding the authenticity of the fingerprint. The first end-to-end method, a simple CNN, allowed an initial analysis regarding the ideal input format. Moving on to more complex architectures, the performance of the pre-trained VGG11 neural network was evaluated, as well as of a developed and trained from scratch CNN. This second custom CNN was then regularized in the same way as the MLP, again to assess whether it is possible to increase the robustness to unknown attacks. Finally, the emerging GANs were used to create an adversarial framework and enrich the training dataset, also with a prospect of increasing the performance of the classifiers concerning previously unseen PAISp.

In conclusion, the methodology of this dissertation is a path that includes the recreation of previously presented solutions but then moved towards the most recent and emerging branches of artificial intelligence.

Chapter 6

Experimental Work

6.1 Datasets

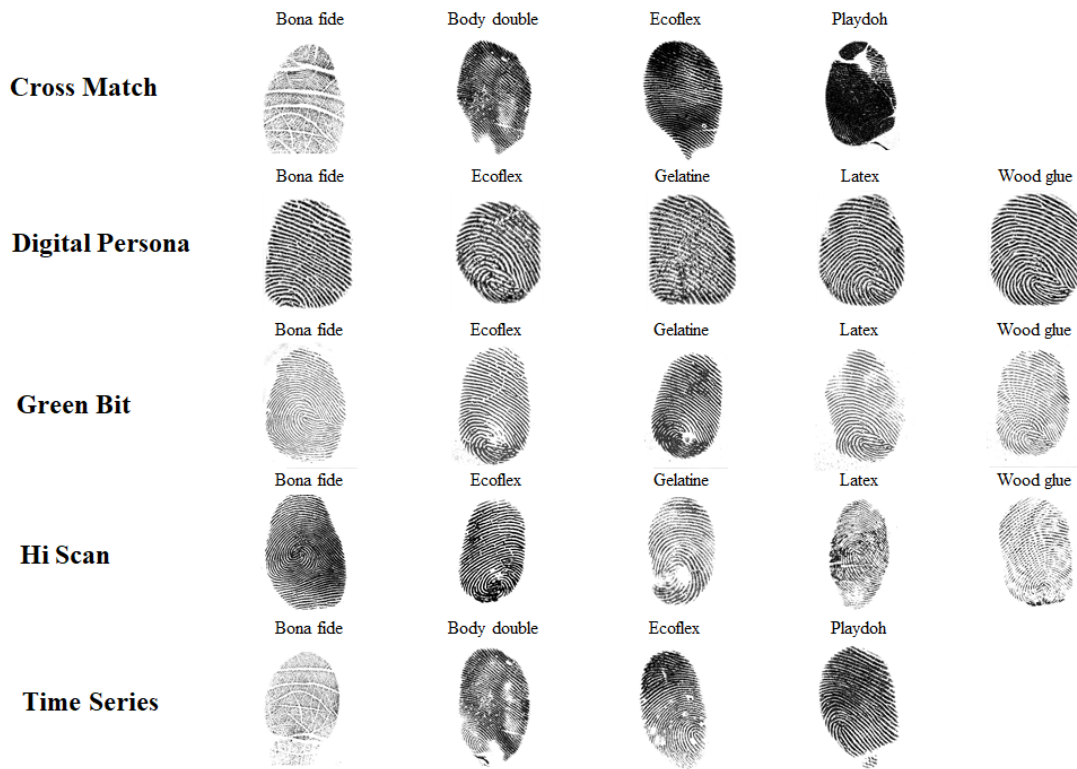


Figure 6.1: LiveDet2015 sample images.

The dataset used for the experimental procedures of this work was the 2015 edition of the Fingerprint Liveness Detection Competition Series [61], described in Section 4.3. This specific database is composed by five subsets, as each sensor is considered to be a separate dataset: Cross Match, Digital Persona, Green Bit, Hi Scan and Time Series. The number of images, as well as the types of materials used to simulate presentation attacks, are specified in Table 6.1 and a set of sample images from each dataset is shown in Figure 6.1.

Table 6.1: LivDet2015 dataset.

Sensor	Bona fide images	Attack images	PAI materials
Cross Match	1510	1478	Body double, Ecoflex, Playdoh
Digital Persona	1000	1000	Ecoflex, Gelatine, Latex, Wood glue
Green Bit	1000	1000	Ecoflex, Gelatine, Latex, Wood glue
Hi Scan	1000	1000	Ecoflex, Gelatine, Latex, Wood glue
Time Series	4440	4486	Body double, Ecoflex, Playdoh

6.2 Performance Metrics

The metrics presented in the following subsections were the ones used to evaluate the performance of the various classifiers in the multiple procedures, as they are recommended by *ISO/IEC FDIS 30107-3 Information Technology — Biometric presentation attack detection — Part 3: Testing and reporting* [33], as well as for their utilization in presentation attacks detection recent research works.

6.2.0.1 Attack Presentation Classification Error Rate

The Attack Presentation Classification Error Rate (APCER) is defined in ISO/IEC (2017) [33] as the proportion of attack images incorrectly classified as bona fide images, given by Equation 6.1.

$$APCER = 1 - \left(\frac{1}{A}\right) \sum_{i=1} pred_i \quad (6.1)$$

Where A is the number of presentation attack images in the set and $pred_i$ is classification prediction for the image i , that is, takes value 1 if the classification result is "presentation attack" and 0 otherwise. Therefore, the sum in Equation 6.1 corresponds to the total number of classifications as presentation attack, that is, corresponds to all positive classifications, whether correct (true positive) or incorrect (false positive). In this way, APCER can be defined as shown in Equation 6.2.

$$\begin{aligned} APCER &= 1 - \frac{TP+FP}{A} \\ &= \frac{A-TP-FP}{A} \end{aligned} \quad (6.2)$$

Where TP and FP correspond to the number of true positive and false positive classifications, respectively, which if subtracted to the total number of positive images A given the number of false negative classifications that indicates the number of presentation attack images incorrectly labeled as bona fide. Therefore, APCER can be defined as presented in Equation 6.3, with conclusion that

this metric corresponds to the false negative rate (FNR) [57].

$$\begin{aligned} APCER &= \frac{FN}{A} \\ &= 1 - \frac{TP}{A} \end{aligned} \quad (6.3)$$

6.2.0.2 Bona Fide Presentation Classification Error Rate

The Bona fide Presentation Classification Error Rate (BPCER) is also defined in ISO/IEC (2017) [33] and can be defined as the proportion of bona fide presentations incorrectly classified as presentation attacks (Equation 6.4).

$$BPCER = \frac{\sum_{i=1}^{B_f} pred_i}{B_f} \quad (6.4)$$

Where $pred_i$ is a binary variable that takes value 1 if a bona fide image is classified as an attack and 0 otherwise, and B_f correspond to the number of bona fide images in the set. Thus, it can be stated that the sum of Equation 6.4 is equal to the number of false positives (FP), so the BPCER can be reformulated and originate Equation 6.5 as the same as the false positive rate (FPR).

$$\begin{aligned} BPCER &= \frac{FP}{B_f} \\ &= 1 - \frac{TN}{B_f} \end{aligned} \quad (6.5)$$

6.2.0.3 Average Classification Error Rate

The Average Classification Error Rate (ACER) is given by the mean of APCER and BPCER [25], as defined in Equation 6.6, and is one of the most common evaluation metrics observed in fingerprint PAD research works, however, it was recently disregarded as a correct evaluation method by ISO/IEC (2017) [33]. Nevertheless, ACER is still widely used, hence its definition in the present document. Although it was not applied and considered for the discussion of the preliminary results, it may be considered in the dissertation work to enable comparison of the new method with those of the state of the art.

$$ACER = \frac{APCER + BPCER}{2} \quad (6.6)$$

6.2.0.4 BPCER at a specific APCER value (BPCER@APCER)

The values of APCER and BPCER described above are calculated for a threshold of 0.5 on the probability scores outputted from the classifier. By obtaining the probability scores for both classes for each samples it is possible to obtain the two probability distributions for each class. This will allow to analyse other points of the curves as well as obtain other metrics, such as the value of BPCER for a specific value of APCER, as suggested in ISO/IEC (2017) [33] and applied in various research works [57, 25]. For the experiments conducted in this dissertation, the reference

values of APCER were 1%, 5% and 10%, which will be denoted as BPCER@1, BPCER@5 and BPCER@10, respectively.

6.2.0.5 Equal Error Rate

The Equal Error Rate (EER), also known as Detection Equal Error Rate (D-EER), analyses the distribution of the false negative rate and the false positive rate scores for different classification thresholds, that is, different operation points. This metric corresponds to the minimum value where FPR and FNR are equal (Equation 6.7), thus, the better the performance, the lower it gets [25].

$$EER = \text{minimum}(FPR = 1 - TPR = FNR) \quad (6.7)$$

Where FPR , TPR and FNR correspond to the false positive rate, total positive rate and false negative rate, respectively. As known, in fingerprint PAD the FPR corresponds to the BPCER metric and the FNR to the APCER, so EER can be redefined with this terms (Equation 6.8). Therefore, it can be concluded that this is one of the most valuable metrics because it takes into account both attack and bona fide detection errors.

$$EER = \text{minimum}(BPCER = APCER) \quad (6.8)$$

A PAD solution can achieve a very low attack detection error (APCER) but with a high bona fide detection error (BPCER), which is not a good performance, so the EER is a good method to analyze both errors at the same time and has not been disregarded like ACER.

6.3 Experiments Description

Table 6.2: Complete experimental work and corresponding codes.

Experiment	Code
SVM vs MLP	A
MLP Regularization	B
First CNN (CNN1)	C
VGG	D
Second CNN (CNN2)	E
CNN2 vs VGG	F
CNN2 Regularization	G
CNN2 (regularized) vs VGG	H
MLP vs CNN2	I
Simple GAN	J
Image-to-image GAN	K

This dissertation includes several approaches for fingerprint PAD, each one comprising different models and each model, in turn, was tested with several variations. Thus, to facilitate the

interpretation of the experimental work as a whole and to assist interpretation and discussion of the results, a code system was implemented. In this system, to each experiment was assigned a a code of letters and numbers, which can be observed in Table 6.2.

6.4 Implementation Details

This section has the main goal of explaining the technical details about the experimental implementation of the methods described in Chapter 5.

The image segmentation and cropping stage was performed in MATLAB by running an executable file with the segmentation method presented in Ferreira et al. (2015) [17], which was provided by the authors of the same. However, all the other experiments were developed in Python 3.7 in the Anaconda Spyder Integrated Development Environment (IDE). The image files were read using the *glob* module and the *Python Image Library* (PIL).

Regarding the feature extraction stage, the LPB was extracted with the *scikit-image* library, which is specialized for image processing, and the LPQ was extracted with the function provided by the original publication. For non-deep learning methods, the classification stage, including data preparation and computation of the evaluation metrics, was employed with the machine learning *scikit-learn* library. On the other hand, the deep learning approaches were developed using the *Pytorch* library, with some evaluation metrics also being obtained through the *scikit-learn* library. The ANNs were trained with the help of Graphics Processing Units (GPUs).

Finally, all basic operations regarding arrays were possible with the *Numpy* package and all graphical outputs (charts and images) were performed by the *Matplotlib* library.

6.5 Results and Discussion

In this section, the results of the experiments described in Chapter 5 are presented and discussed (Tables 6.4 to 6.21). To simplify the analysis, the presented tables only contain the average of all materials for each sensor, however, by doing this some information is lost, as the performance can vary a lot when the attack material is different. This variability is shown in Table 6.3, which is a sample from the complete results of the baseline SVM, with the three PAISp of the Cross Match dataset. As observed, different presentation attack materials bring different classification performances. Despite the loss of information, the average results per sensor were considered sufficient to have a rich discussion about the results of the different approaches followed in this dissertation. In addition, as a way of making it easier for comparison, the results presented in this section will only feature three of the evaluation metrics described in Section 6.2: APCER, EER and BPCER@5 (BPCER at a fixed APCER of 5%). The APCER directly evaluates the capability of detecting attacks and the other two use the distribution of both APCER and BPCER, which makes them stronger points of comparison than just looking at the BPCER. The complete results, for all sensors, separated by PAISp and with all the metrics applied are presented in Appendix A.

Table 6.3: Performance differences between PAISp. Sample of the results of the SVM classifier for the unseen-attack framework and with the Cross Match sensor.

Cross Match - Unseen-attack - SVM							
PAI	With segmentation						
	Features = Intensity + LBP + LPQ						
	Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10
Body Double	86.50%	19.40%	7.60%	11.60%	29.50%	20.40%	13.00%
Ecoflex	96.60%	1.60%	7.00%	3.80%	9.20%	2.20%	1.00%
Playdoh	80.00%	10.20%	48.90%	23.20%	84.20%	63.30%	49.30%

The following subsections follow the experimental codes displayed in Table 6.2. In each table, the metrics corresponding to the approach considered the best were highlighted with different colors for different methods. The choice of the best performance was not based from just one of the metrics but through an overall conclusion employed by observing all errors.

6.5.1 Experiment A - SVM vs MLP

Table 6.4: SVM vs MLP (A) - One-attack framework.

A - One attack							
Sensor	Classifier	Without segmentation			With segmentation		
		APCER	EER	BPCER@5	APCER	EER	BPCER@5
Cross Match	SVM	9.10%	4.77%	4.77%	9.77%	4.30%	3.83%
	MLP	1.01%	2.17%	0.09%	0.23%	0.10%	0.00%
Digital Persona	SVM	11.30%	11.15%	24.40%	5.53%	2.00%	18.80%
	MLP	0.80%	1.90%	2.03%	0.63%	1.45%	0.68%
Green Bit	SVM	7.05%	5.30%	8.73%	14.05%	16.18%	14.15%
	MLP	0.65%	0.63%	1.95%	10.10%	6.38%	7.23%
Hi Scan	SVM	9.75%	13.83%	15.35%	31.63%	14.95%	24.50%
	MLP	1.03%	0.95%	0.25%	0.50%	4.50%	6.98%
Time Series	SVM	10.53%	8.33%	11.20%	8.73%	19.83%	32.33%
	MLP	0.13%	0.07%	0.00%	0.23%	0.20%	0.00%

The first comparison to be made was between the best conventional machine learning approach, the SVM classifier (Section 5.4.1), and the first deep learning approach, a simple MLP (Section 5.5.1), both using handcrafted features as described in Chapter 5 (Section 5.3). From Tables 6.4 and 6.5, several conclusions can be made:

- Regarding the classifiers, the superiority of deep learning is evident, since the MLP obtained better results for all frameworks and sensors, except for the unseen-attack scenario with the non-segmented images of the GreenBit sensor (Table 6.5).
- Regarding the segmentation and cropping of the images, in the one-attack approach (Table 6.4) the results are better with the original images for 3 of the 5 sensors, while for the unseen-attack (Table 6.5), the segmentation brought a better performance in 4 out of 5 sensors. This

Table 6.5: SVM vs MLP (A) - Unseen-attack framework.

A - Unseen attack							
<i>Sensor</i>	<i>Classifier</i>	<i>Without segmentation</i>			<i>With segmentation</i>		
		APCER	EER	BPCER@5	APCER	EER	BPCER@5
Cross Match	SVM	3.40%	9.93%	15.43%	10.40%	12.87%	28.63%
	MLP	0.60%	6.33%	10.80%	0.07%	4.33%	7.57%
Digital Persona	SVM	4.60%	9.50%	19.10%	3.05%	14.90%	35.90%
	MLP	0.00%	1.50%	4.13%	0.00%	0.45%	0.53%
Green Bit	SVM	2.60%	3.00%	1.90%	2.83%	4.80%	4.30%
	MLP	0.00%	8.98%	21.30%	0.70%	1.10%	0.20%
Hi Scan	SVM	6.48%	5.10%	5.70%	24.00%	23.20%	40.20%
	MLP	0.00%	0.58%	0.48%	0.30%	3.03%	2.83%
Time Series	SVM	28.60%	17.53%	42.73%	41.87%	16.50%	42.90%
	MLP	0.17%	1.43%	1.00%	0.00%	0.60%	0.03%

can be explained by the possibility that when only one PAISp is involved in the training (one-attack), the model may be taking advantage of the differences in tones between the real and fake fingerprints, using the background as the comparison term. When it is used more than one PAISp in the training step, that tone analysis may be no longer advantageous and that is why it is better to focus the image on the region of interest (fingerprint) before extracting the features. Nevertheless, by counting amidst both frameworks, the segmentation and cropping brought better performance in 6 out of 10 experiments (2 frameworks times 5 sensors), so it is possible to conclude the positive effect of excluding the background from the images. Despite this conclusion, the next experiment will also include the possibility of features extracted from both the original and cropped images.

- Regarding the two frameworks, one-attack and unseen-attack, the first one brings the best performance, as expected, because the PAISp used for testing is seen by the classifier during the training phase. The higher errors in the unseen-attack framework show the necessity of developing new models that are capable of generalizing better the differences between bona fide fingerprints and presentation attacks, hence being more robust to a novel (unseen) PAISp.

6.5.2 Experiment B - MLP Regularization (MLP_{reg})

The MLP regularization aims to try to increase the robustness against unknown attacks, that is, to improve the results for the unseen-attack framework. Table 6.6 shows the results of the MLP's regularization for the unseen-attack framework. As the adversarial classifier needs at least two different PAISp in the training phase, the one-attack framework would have no meaning in this experience. For each sensor there are 4 approaches presented, which are in line with the methodology explained in Section 5.5.2: **None** (without regularization - corresponds to the MLP's results presented in Table 6.5), **Adversarial** (Adversarial Weight is positive and Transfer Weight equal

Table 6.6: MLP regularization (B) - Unseen-attack framework.

B - Unseen attack							
<i>Sensor</i>	<i>Regularization MLP</i>	<i>Without segmentation</i>			<i>With segmentation</i>		
		APCER	EER	BPCER@5	APCER	EER	BPCER@5
Cross Match	None	0.60%	6.33%	10.80%	4.33%	4.33%	8.00%
	Adversarial	3.33%	8.63%	13.20%	4.77%	4.77%	7.27%
	Transfer	1.33%	7.23%	10.27%	4.63%	4.63%	7.13%
	Adversarial + Transfer	2.53%	9.17%	15.40%	3.70%	3.70%	4.30%
Digital Persona	None	0.00%	1.50%	4.13%	0.45%	0.45%	1.00%
	Adversarial	0.00%	0.78%	1.68%	0.50%	0.50%	0.10%
	Transfer	0.25%	0.50%	0.00%	0.68%	0.68%	1.20%
	Adversarial + Transfer	0.00%	0.30%	1.85%	0.30%	0.30%	0.00%
Green Bit	None	0.00%	8.98%	21.30%	0.70%	1.10%	0.20%
	Adversarial	0.00%	7.28%	18.43%	0.60%	0.93%	0.18%
	Transfer	0.00%	7.40%	20.90%	0.90%	0.93%	0.48%
	Adversarial + Transfer	0.00%	5.25%	10.75%	0.70%	0.93%	0.63%
Hi Scan	None	0.00%	0.58%	0.48%	0.30%	3.03%	2.83%
	Adversarial	0.00%	0.38%	0.00%	0.60%	3.80%	6.05%
	Transfer	0.00%	0.65%	0.43%	1.50%	5.78%	7.90%
	Adversarial + Transfer	0.00%	0.23%	0.00%	1.30%	3.38%	3.60%
Time Series	None	0.60%	1.43%	1.00%	0.00%	0.60%	0.03%
	Adversarial	0.63%	4.73%	15.95%	0.00%	0.30%	0.03%
	Transfer	0.97%	2.73%	2.30%	0.00%	1.27%	0.53%
	Adversarial + Transfer	0.47%	1.10%	0.70%	0.00%	0.10%	0.03%

to zero), **Transfer** (Adversarial Weight is equal to zero and Transfer Weight positive), and **Adversarial + Transfer** (Adversarial and Transfer Weights are both positive). It is possible to draw several conclusions from the table:

- The segmentation and cropping of the images before feature extraction brings the best performance in 4 out of 5 sensors, thus reinforcing its superiority in comparison to the use of the original images.
- The regularization with both penalties (Adversarial and Transfer) successfully improved the performance for unseen-attacks (Table 6.6), leading to the best performance in 3 out of 5 sensors using segmented and cropped images. However, as some errors obtained with the baseline MLP are already low, the differences are not very significant.
- Regarding the isolation of the Adversarial and Transfer penalties (looking only at the "With segmentation" columns of Table 6.6), the model regularized only with the Adversarial penalty obtained better results than the model regularized only with the Transfer penalty. Regardless of this, the combination of both penalties continues to be the better approach against unknown attacks.

In conclusion, the regularization of MLP proved to be effective in improving the performance of the model against unknown attacks. Once again, the use of features extracted from the segmented and cropped images demonstrated to be advantageous and, therefore, any future comparison with the MLP will be with the results corresponding to those.

6.5.3 Experiment C - First CNN (CNN1)

Table 6.7: First CNN experiments (C) - one-attack and unseen-attack.

C							
Sensor	Method	One Attack			Unseen Attack		
		APCER	EER	BPCER@5	APCER	EER	BPCER@5
Cross Match	Entire image	7.86%	5.52%	11.67%	2.07%	5.33%	19.23%
	ROI (padding)	6.66%	4.85%	8.17%	1.90%	26.10%	29.57%
	ROI (cropping)	3.45%	2.37%	2.67%	5.57%	7.33%	21.13%
	Patches (method 1)	18.31%	15.35%	20.13%	8.53%	15.83%	38.27%
	Patches (method 2)	17.93%	13.13%	19.90%	18.93%	20.40%	40.37%

To recap, the first CNN (CNN1) served as an initial experiment of an end-to-end solution, as well as to test what would be the most effective approach with respect to the format of the input (experience detailed in Section 5.6.1). There were five variations of input, each one taking a significant amount of time to pre-process, as well as to train the model, so the CNN1 was only tested with images from the Cross Match sensor, making it possible to continue the journey with deep learning solutions and study approaches with greater potential. The evaluated pre-processing methods were the following:

1. Original images (**entire image**).
2. Segmented and cropped images with padding, to match the dimensions of all images (**ROI padding**).
3. Segmented and cropped images with additional cropping, to match the dimensions of all images (**ROI cropping**).
4. Concatenated patches (**patches method 1**).
5. Individualized patches (**patches method 2**).

The results are shown in Table 6.7, from where it can be concluded that segmented and cropped images are the best form of input for CNNs. These results are in line with the conclusions drawn previously with the handcrafted features. Regarding the processing to ensure that all images have the same dimensions, for the one-attack framework, it was better to crop, whereas for the unseen-attack it is hard to define the best approach. Nevertheless, as the models are dealing with fingerprint images, with specific and detailed patterns, the best processing option is to crop, as the resize, if not applied carefully, can distort the patterns of ridges and valleys and modify the fingerprint.

Despite its simplicity, this model achieves satisfactory results, but there is still much space for improvement. One of the most recently used approaches in fingerprint PAD is the use of pre-trained models, which was, therefore, explored in the solution discussed in the following section.

6.5.4 Experiment D - VGG

The second end-to-end solution explored was the pre-trained VGG11 with batch normalization. The input restrictions imposed by this model were addressed in two different ways (Resize and Crop), which is fully described in Section 5.6.2. From Tables 6.8 and 6.9 it is straightforward to conclude that cropping the input images leads to better results than resizing. The cropping approach achieved the best performance with all sensors in the one-attack framework (Table 6.8) and in the unseen-attack framework (Table 6.9).

Table 6.8: VGG11 (D) - One-attack framework.

D - One attack				
<i>Sensor</i>	<i>Pre-processing</i>	APCER	EER	BPCER@5
Cross Match	Resize	7.78%	7.13%	10.00%
	Crop	5.89%	3.63%	3.23%
Digital Persona	Resize	20.90%	17.27%	30.80%
	Crop	15.68%	8.91%	12.19%
Green Bit	Resize	5.81%	3.69%	3.36%
	Crop	4.24%	2.74%	1.90%
Hi Scan	Resize	6.49%	4.88%	7.56%
	Crop	8.54%	2.73%	6.60%
Time Series	Resize	11.87%	13.37%	29.91%
	Crop	0.97%	0.47%	0.76%

Table 6.9: VGG11 (D) - Unseen-attack framework.

D - Unseen attack				
<i>Sensor</i>	<i>Pre-processing</i>	APCER	EER	BPCER@5
Cross Match	Resize	14.00%	17.30%	25.57%
	Crop	3.13%	8.10%	10.23%
Digital Persona	Resize	22.30%	20.78%	29.83%
	Crop	5.50%	11.25%	16.95%
Green Bit	Resize	4.55%	4.95%	5.45%
	Crop	2.03%	3.63%	8.95%
Hi Scan	Resize	4.10%	6.10%	8.53%
	Crop	3.50%	3.48%	2.33%
Time Series	Resize	11.40%	17.62%	33.59%
	Crop	0.43%	1.70%	3.93%

Despite being a model pre-trained in a database that does not include fingerprint images (ImageNet database), the model's finetuning allows it to adapt to the new images, obtaining good results with both known and unknown attacks but still leaving space to improvements.

6.5.5 Experiment E - Second CNN (CNN2)

Knowing the good performance of the VGG, the next focus was to try to match these results with a model developed and trained from scratch. Therefore, a second CNN (CNN2) was created and evaluated in relation to the pre-processing of the input (the same two variants tested with the VGG - **Resize** and **Crop**) and also in relation to the size of the kernels applied in the convolution layers (Section 5.6.3).

Table 6.10: Second CNN (E) - One-attack framework.

E - One attack					
<i>Sensor</i>	<i>Pre-processing</i>	<i>Kernels</i>	<i>APCER</i>	<i>EER</i>	<i>BPCER@5</i>
Cross Match	Resize	3	8.36%	8.18%	9.99%
		5	7.51%	5.62%	8.17%
		7,5,3	8.60%	7.04%	8.58%
	Crop	3	4.81%	2.92%	2.27%
		5	5.92%	4.80%	5.09%
		7,5,3	7.21%	3.53%	2.94%
Digital Persona	Resize	3	21.32%	14.72%	17.95%
		5	21.89%	18.13%	27.83%
		7,5,3	22.82%	20.07%	27.73%
	Crop	3	7.82%	4.33%	4.71%
		5	9.43%	7.48%	8.82%
		7,5,3	10.14%	6.89%	5.67%
Green Bit	Resize	3	5.49%	3.68%	3.20%
		5	8.75%	4.23%	4.17%
		7,5,3	5.35%	3.23%	2.42%
	Crop	3	3.45%	2.48%	2.13%
		5	5.89%	3.03%	2.67%
		7,5,3	4.29%	1.55%	3.11%
Hi Scan	Resize	3	10.01%	5.59%	9.58%
		5	12.67%	10.27%	10.70%
		7,5,3	13.56%	10.63%	15.81%
	Crop	3	7.77%	7.08%	8.84%
		5	10.16%	6.28%	8.12%
		7,5,3	9.77%	3.98%	5.87%
Time Series	Resize	3	13.61%	14.57%	24.57%
		5	11.90%	12.51%	22.58%
		7,5,3	12.98%	13.41%	27.41%
	Crop	3	2.33%	1.76%	2.83%
		5	2.04%	1.30%	1.90%
		7,5,3	3.13%	1.81%	1.94%

In Tables 6.10 and 6.11, the results are displayed for all combinations of pre-processing and kernel size. The number 3 on the kernel column corresponds to all kernels having a dimension of 3x3, the number 5 to all kernels having a 5x5 dimension and 7,5,3 means that the kernel of the first

Table 6.11: Second CNN (E) - Unseen-attack framework.

E - Unseen attack					
<i>Sensor</i>	<i>Pre-processing</i>	<i>Kernels</i>	<i>APCER</i>	<i>EER</i>	<i>BPCER@5</i>
Cross Match	Resize	3	8.36%	8.18%	9.99%
		5	7.51%	5.62%	8.17%
		7,5,3	10.53%	17.37%	25.07%
	Crop	3	5.00%	8.70%	6.25%
		5	5.92%	4.80%	5.09%
		7,5,3	5.87%	12.80%	17.67%
Digital Persona	Resize	3	20.00%	21.28%	35.08%
		5	26.90%	22.95%	29.60%
		7,5,3	27.60%	28.33%	45.90%
	Crop	3	5.60%	7.28%	10.80%
		5	5.10%	8.58%	11.58%
		7,5,3	4.70%	9.40%	11.60%
Green Bit	Resize	3	4.53%	5.13%	7.95%
		5	3.43%	5.03%	5.70%
		7,5,3	4.65%	6.83%	9.68%
	Crop	3	3.03%	7.05%	14.13%
		5	4.23%	5.70%	9.63%
		7,5,3	2.60%	7.05%	13.90%
Hi Scan	Resize	3	5.60%	9.88%	10.63%
		5	6.60%	9.50%	10.15%
		7,5,3	7.40%	6.85%	10.40%
	Crop	3	5.60%	11.25%	20.15%
		5	4.40%	7.23%	11.78%
		7,5,3	8.40%	8.70%	11.73%
Time Series	Resize	3	12.80%	18.33%	34.87%
		5	10.23%	16.90%	28.67%
		7,5,3	14.73%	16.17%	35.37%
	Crop	3	0.83%	3.83%	6.83%
		5	1.37%	4.07%	9.10%
		7,5,3	0.70%	3.97%	9.40%

layer is 7x7, the kernels of the next 3 layers are 5x5 and the rest are 3x3. From the observation of the tables, several conclusions can be drawn:

- As with the VGG11 model, cropping the images proved to be the most proper pre-processing, being involved in the best performance in 9 out of 10 experiments (5 sensors times 2 frameworks).
- Regarding the size of the kernels, it is not simple to decide which approach is the best. Looking at Table 6.10 (one-attack framework), the use of only 3x3 filters was the top option in most of the sensors. On the other hand, looking at Table 6.11 (unseen-attack framework), the option of all kernels to be 5x5 was superior in 3 of the 5 sensors, while the 3x3 option was

better in the rest. Therefore, taking into account the disparity of the results in relation to this variable, the kernel size 3x3 was chosen to be applied in future models, due to the fact that it obtained better results in 5 of the 10 experiments, more than the other two possibilities.

In conclusion, the second CNN reached a very positive performance, mainly with known attacks, and this can be observed when comparing it with the literature. For example, Yuan et al. (2019) [95] obtained an average ACER (Average Classification Error Rate) of 3.7% and 6.45% with the LivDet2013 and LivDet2011 databases, respectively. Furthermore, Zhang et al. (2019) [98] obtained an average ACER of 3.07% with the LivDet2017 database and won the same competition, and Park et al. (2016) [72] obtained an ACER of 3.42% with the LivDet2009 database. All these three works used more than a simple CNN, such like Residual Networks or Image Scale Equalization. The average ACER obtained with the proposed CNN2 for all datasets was 5.36%, which is in the range of what is found in the literature, even without more than just a plain CNN. Although the ACER was not used in this dissertation, the APCER and BPCER values found in the complete results (Appendix A) were used to calculate the average ACER in order to confirm that the baseline CNN2's performance matched the state of the art solutions. However, as the one-attack scenario was not the focus of this dissertation and the databases are not exactly the same (even though some subsets are shared), a thorough comparison was not presented.

In this way, the second CNN is optimized and with promising results, but still having ground for improvements. To make further conclusions it is important now to compare its performance with the performance of the VGG11, results that are discussed in the next section.

6.5.6 Experiment F - VGG vs CNN2

Table 6.12: VGG vs CNN2 (F) - One-attack framework.

F - One attack				
<i>Sensor</i>	<i>Model</i>	APCER	EER	BPCER@5
Cross Match	VGG	5.89%	3.63%	3.23%
	CNN2	4.81%	2.92%	2.27%
Digital Persona	VGG	15.68%	8.91%	12.19%
	CNN2	7.82%	4.33%	4.71%
Green Bit	VGG	4.24%	2.74%	1.90%
	CNN2	3.45%	2.48%	2.13%
Hi Scan	VGG	8.54%	2.73%	6.60%
	CNN2	7.77%	7.08%	8.84%
Time Series	VGG	0.97%	0.47%	0.76%
	CNN2	2.33%	1.76%	2.83%

Tables 6.12 and 6.13 summarize the comparison between the two end-to-end solutions described previously, the pre-trained VGG11 and CNN2, both of which already optimized. That said, several conclusions can be drawn:

Table 6.13: VGG vs CNN2 (F) - Unseen-attack framework.

F - Unseen attack				
Sensor	Model	APCER	EER	BPCER@5
Cross Match	VGG	3.13%	8.10%	10.23%
	CNN2	5.00%	8.70%	6.25%
Digital Persona	VGG	5.50%	11.25%	16.95%
	CNN2	5.60%	7.28%	10.80%
Green Bit	VGG	2.03%	3.63%	8.95%
	CNN2	3.03%	7.05%	14.13%
Hi Scan	VGG	3.50%	3.48%	2.33%
	CNN2	5.60%	11.25%	20.15%
Time Series	VGG	0.43%	1.70%	3.93%
	CNN2	0.83%	3.83%	6.83%

- In the one-attack framework (Table 6.12), the CNN2 is better in 3 out of 5 sensors (Cross Match, Digital Persona and Green Bit) and VGG is better with the remaining two (Hi Scan and Time Series). In this way, the CNN2 is slightly superior when it comes to detecting known presentation attacks.
- Regarding unknown attacks (Table 6.13), the VGG is better in 3 out of 5 sensors, so it is slightly better than the CNN2. Although in the first sensor (Cross Match) the VGG obtained a slightly lower APCER and EER, the BPCER@5 was significantly higher than with the CNN2. To further analyze these results, the tables with the complete results obtained with the VGG (Table A.21) and with the CNN2 (Table A.27) for this sensor were analyzed and the average values for all metrics were computed. Consequently, it was observed that the CNN2 clearly performed better than the VGG, since the results were better with almost all the evaluation metrics.

In conclusion, both solutions had very satisfactory results with known attacks and were very promising when encountering unseen attacks. Overall, the CNN2 achieved results equivalent to the VGG, which is very positive considering that it was developed and trained from scratch. Thus, while there is still room for improvement in the scope of unknown attacks, CNN2 was chosen as the baseline, mainly due to the greater ease of adaptation and regularization, as well as the lower computational cost and greater efficiency in learning. Having proven the success of regularizing the MLP with features extracted manually, the next step was then to apply this same methodology to the pure deep learning approaches, that is, to the CNN2.

6.5.7 Experiment G - CNN2 Regularization (CNN2_{reg})

As with the regularization applied to the MLP, the CNN2's regularization (Section 5.6.4) was only tested for the unseen attack framework, since it does not make sense to apply the one-attack framework. The results of this experiment are summarized in Table 6.14. Once again, 4 possibilities

Table 6.14: CNN2 regularization (G) - Unseen-attack framework.

G - Unseen attack				
<i>Sensor</i>	<i>Regularization CNN2</i>	APCER	EER	BPCER@5
Cross Match	None	5.00%	8.70%	6.25%
	Adversarial	1.40%	3.33%	6.54%
	Transfer	1.00%	7.63%	14.07%
	Adversarial + Transfer	1.07%	2.82%	4.65%
Digital Persona	None	5.60%	7.28%	10.80%
	Adversarial	1.50%	2.45%	3.98%
	Transfer	1.80%	3.20%	3.95%
	Adversarial + Transfer	0.60%	2.45%	3.85%
Green Bit	None	3.03%	7.05%	14.13%
	Adversarial	0.60%	3.13%	5.43%
	Transfer	0.50%	4.80%	7.50%
	Adversarial + Transfer	0.60%	1.63%	2.93%
Hi Scan	None	5.60%	11.25%	20.15%
	Adversarial	2.40%	1.80%	3.20%
	Transfer	2.20%	2.00%	2.00%
	Adversarial + Transfer	1.20%	1.04%	1.21%
Time Series	None	1.37%	4.07%	9.10%
	Adversarial	0.87%	2.33%	5.20%
	Transfer	0.47%	2.67%	6.33%
	Adversarial + Transfer	0.60%	2.70%	6.30%

were tested for each sensor: **without regularization** (the same as the original CNN2), **regularized only with the adversarial penalty**, **regularized only with the PAISp-transfer penalty**, and, finally, **regularized with both penalties**. Several conclusions can be drawn from the results obtained:

- The regularization with both penalties successfully improved the performance for unknown-attacks, achieving the best approach with 4 of the 5 sensors. The Time Series sensor was the only one where this approach was not the best. In this case, the best was regularization with just the adversarial penalty, but even so, the regularization with both penalties had better results than the original model, with the APCER lowering from 1.37% to 0.60%, the EER from 4.07% to 2.70% and the BPCER@5 from 9.10% to 6.30%.
- Regarding the isolation of the adversarial and PAISp-transfer penalties, the model regularized only with the adversarial penalty is superior in general. The only case where the transfer penalty isolated is better than the adversarial isolated is the experiment with the Hi Scan sensor, but with a small difference, as the APCER and the EER are almost equal.

Additionally, a comparison was made with the best literature results with similar approaches (software static solutions). Although fingerprint PAD is a widely discussed topic, the detection of unknown attacks is still a little mentioned topic, so not many recent publications have tested

solutions with unseen attacks. However, the 2015 edition of the Liveness Detection Competition (LivDet2015) included the objective of detecting unknown attacks, so it is an excellent comparative term considering that the dataset used in this dissertation was the LivDet2015 database. Table 6.15 presents the results of the regularized CNN2 side by side with the best results from LivDet2015, as well as the results of a recent publication that also addressed this topic. The metrics presented are the same as those presented in the publications, and only 3 sensors are presented because they were the ones tested in common.

Table 6.15: Literature and proposed *CNN2_{reg}*.

Method	PAD metrics (%)							
	Cross Match		Digital Persona		GreenBit		Average	
	APCER	BPCER@5	APCER	BPCER@5	APCER	BPCER@5	APCER	BPCER@5
Proposed <i>CNN2_{reg}</i>	1.07	4.65	0.60	3.85	0.60	2.93	0.76	3.81
<i>LivDet2015</i> [23, 61]	1.68	≈ 0.80	0.60	≈ 10.00	4.00	≈ 5.00	2.09	≈ 5.27
<i>Park et al</i> [71]	0.00	-	11.00	-	8.00	-	6.33	-

From the observation of the table, it is easy to conclude the superiority of the proposed model (*CNN2_{reg}*) compared to solutions in the literature. The regularized CNN2 achieved the best performance in 2 of the 3 sensors and the average results were significantly better, with an average APCER of 0.76% and an average BPCER@5 of 3.81%.

In conclusion, the presented regularization was once again a success in improving the performance of the model in detecting unknown attacks. In some cases, this improvement was very significant, lowering errors by almost 20%. Within the studied regularization possibilities, the combination of the two penalties was the most successful approach. Now, knowing the results of this solution against unseen attacks and its success when comparing it with the literature, it makes sense to go back and compare it with other methods already tested, as is the case of the VGG and the handcrafted features-based approaches, namely the MLP.

The results of the regularized CNN2 were presented in a submission for the 19th International Conference of the Biometrics Special Interest Group (BIOSIG 2020).

6.5.8 Experiment H - VGG vs *CNN2_{reg}*

The first comparison using the regularized version of the CNN2 (*CNN2_{reg}*) was with the pre-trained VGG11. In the previous comparison between the VGG and the baseline CNN2, it was concluded that the performance of the two models was almost equivalent, however, the VGG proved to be better when dealing with unseen-attacks. Therefore, it makes sense to understand whether the regularization of the CNN2 has led to it outperform the VGG or not.

Looking at Table 6.16, it is easy to conclude that the *CNN2_{reg}* has a better ability to detect unknown attacks than the pre-trained VGG. The regularized version of the CNN2, developed and trained from scratch, obtained the lowest errors in 4 of the 5 sensors. In some of them, the differences are significant, as in the case of the sensor Digital Persona, in which the VGG has an APCER of 5.50%, an EER of 11.25% and a BPCER@5 of 16.95%, while the *CNN2_{reg}* has the APCER, EER and BPCER@5 equal to 0.60%, 2.45% and 3.85%, respectively. In the case of the

Table 6.16: VGG vs $CNN2_{reg}$ (H) - Unseen-attack framework.

H - Unseen attack				
<i>Sensor</i>	<i>Model</i>	APCER	EER	BPCER@5
Cross Match	VGG	3.13%	8.10%	10.23%
	CNN2reg	1.07%	2.82%	4.65%
Digital Persona	VGG	5.50%	11.25%	16.95%
	CNN2reg	0.60%	2.45%	3.85%
Green Bit	VGG	2.03%	3.63%	8.95%
	CNN2reg	0.60%	1.63%	2.93%
Hi Scan	VGG	3.50%	3.48%	2.33%
	CNN2reg	1.20%	1.04%	1.21%
Time Series	VGG	0.43%	1.70%	3.93%
	CNN2reg	0.60%	2.70%	6.30%

Time Series sensor, with which the VGG was better, the difference in APCER is 0.17%, in EER is 1% and in BPCER@5 is 2.37%.

It can, consequently, be concluded that the regularization of the CNN2 clearly led to a performance improvement over the VGG11. Nevertheless, in future work it would be interesting to study the performance of more pre-trained models and even address the possibility of applying the regularization to them, always maintaining the weights resulting from previous training, that is, maintaining the approach of just finetuning the models.

6.5.9 Experiment I - MLP vs CNN2

Table 6.17: MLP vs CNN2 (I) - One-attack framework.

I - One attack				
<i>Sensor</i>	<i>Model</i>	APCER	EER	BPCER@5
Cross Match	MLP baseline	1.01%	0.10%	0.00%
	CNN baseline	4.81%	2.92%	2.27%
Digital Persona	MLP baseline	0.80%	1.45%	1.00%
	CNN baseline	7.82%	4.33%	4.71%
Green Bit	MLP baseline	0.65%	6.38%	7.00%
	CNN baseline	3.45%	2.48%	2.13%
Hi Scan	MLP baseline	1.03%	4.50%	7.00%
	CNN baseline	7.77%	7.08%	8.84%
Time Series	MLP baseline	0.13%	0.20%	0.00%
	CNN baseline	2.33%	1.76%	2.83%

The second comparison using the $CNN2_{reg}$ was with the similarly regularized MLP. The idea of this comparison is not only to observe side by side the performance of the handcrafted features-based method with the end-to-end solution, but also to evaluate the impact that the regularization had on both approaches individually, so these results also include the performance of the baseline

Table 6.18: MLP vs CNN2 (I) - Unseen-attack framework.

I - Unseen attack					
<i>Sensor</i>	<i>Model</i>		APCER	EER	BPCER@5
Cross Match	MLP	Baseline	4.33%	4.33%	8.00%
		Regularized	3.70%	3.70%	4.30%
	CNN	Baseline	5.00%	8.70%	6.25%
		Regularized	1.07%	2.82%	4.65%
Digital Persona	MLP	Baseline	0.45%	0.45%	1.00%
		Regularized	0.30%	0.30%	0.00%
	CNN	Baseline	5.60%	7.28%	10.80%
		Regularized	0.60%	2.45%	3.85%
Green Bit	MLP	Baseline	0.70%	1.10%	0.20%
		Regularized	0.70%	0.93%	0.63%
	CNN	Baseline	3.03%	7.05%	14.13%
		Regularized	0.60%	1.63%	2.93%
Hi Scan	MLP	Baseline	0.30%	3.03%	2.83%
		Regularized	1.30%	3.38%	3.60%
	CNN	Baseline	5.60%	11.25%	20.15%
		Regularized	1.20%	1.04%	1.21%
Time Series	MLP	Baseline	0.00%	0.60%	0.03%
		Regularized	0.00%	0.10%	0.03%
	CNN	Baseline	1.37%	4.07%	9.10%
		Regularized	0.60%	2.70%	6.30%

models (MLP and CNN), apart from their regularized versions. In this way, several conclusions can be drawn from Tables 6.17 and 6.18:

- In Table 6.18 (unseen-attack), by looking at the two lines of the MLP and the two lines of the CNN individually, for each sensor, it is possible to conclude that the regularization had a much more significant positive impact when applied to the CNN. While with the MLP the differences are smaller and there is even a case where regularization clearly worsens the performance (sensor Hi Scan), with the CNN the regularization always improves the results and with larger differences. Certainly, this is greatly influenced by the fact that the MLP baseline had already obtained very satisfactory results, while the CNN baseline had obtained errors of up to 20%, meaning that it had much more ground for improvements.
- Still in the unseen-attack framework (Table 6.18), comparing the four methods for each sensor at the same time (best results are highlighted), the regularized MLP obtained the best performance in 2 of the 5 sensors (Digital Persona and Time Series), while the regularized CNN was superior in other 2 (Cross Match and Hi Scan). With the remaining sensor (Green Bit), although it is difficult to define the better approach among the four, the regularized MLP obtained better results than the regularized CNN. Therefore, the method of hand-crafted features is slightly superior when it comes to detecting unknown attacks.

- Regarding known attacks (Table 6.17), the MLP shows that it is superior to the CNN, obtaining the best results in 4 of the 5 experiments. However, as discussed previously, the baseline CNN has achieved good results that are competitive with the state-of-the-art baseline end-to-end solutions. Nevertheless, these results demonstrate the superiority that the methods of handcrafted features can still achieve.

In conclusion, although both solutions have obtained good results in the carried out experiments, the use of features extracted manually proved to be better than the pure deep learning solutions such as the CNN2. This may be explained by the fact that the extracted features are very specific and detailed, mainly the LBP and the LPQ, while the CNN uses directly the images, and it is known that these models need larger datasets to demonstrate their maximum potential. Consequently, taking into account that the dataset used for this dissertation is not very large, if more data was provided to the CNN it would be expected to outperform the MLP, both for known and unknown attacks. Finally, the most important conclusion to be drawn from these experiments is the success of the regularization when it comes to generalizing the models for any type of PAI, regardless of the type of approach (handcrafted features or automatically learned features). This method of adversarial training with domain adaptation techniques has shown yet again to have a positive impact on this application of detecting fingerprint presentation attacks.

6.5.10 Experiment J - One-class GAN

The first GAN-based model developed in this dissertation had two main goals: i) study the ability to generate fingerprints and ii) study a generalized PAD approach using the Discriminator as a one-class classifier (Section 5.6.5).



(a) Without diversity term.

(b) With diversity term.

Figure 6.2: Example of generated fingerprints without and with the diversity term. Training images from Cross Match sensor.

Regarding the generation of images, GANs usually need large amounts of training data in order to achieve their maximum potential, as all CNNs normally need. Nevertheless, the Generator of

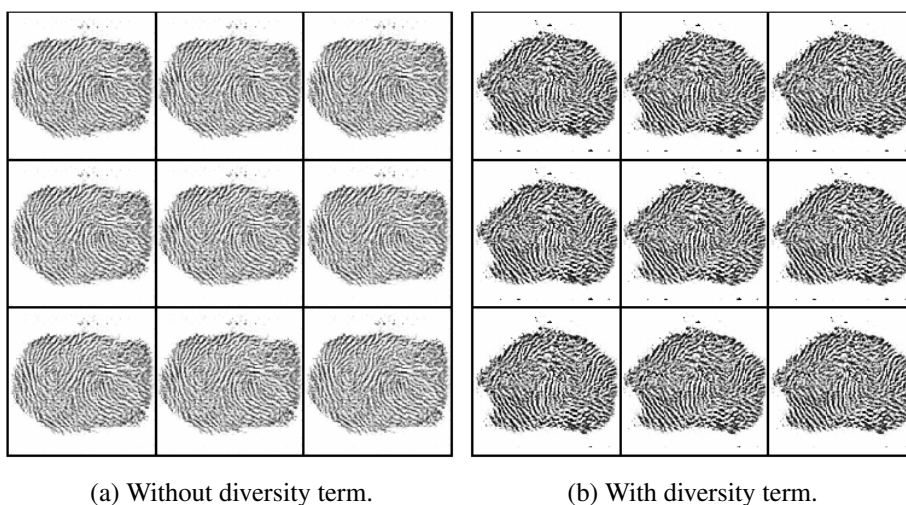


Figure 6.3: Example of generated fingerprints without and with the diversity term. Training images from Digital Persona sensor.

the model in study was capable of producing fingerprints from just random noise inputs, even with the relatively small dataset that was used. In most sensors, the diversity term was crucial to avoid mode collapsing, that is, to ensure variety in the Generator's output. Figure 6.2 shows examples of fingerprints generated by the model without and with the diversity term included in the Generator's loss. As observed, the fingerprints synthesized with this term are not only more diverse but also slightly sharper, having a better defined ridge-valley pattern. However, with the sensor Digital Persona, the diversity term did not have a significant effect on the synthesized images regarding variety, as seen in Figure 6.3. The original bona fide images obtained with this sensor often have a less defined pattern and are not as clean as those of others, which may be the reason behind this outcome. Several examples of generated fingerprints with the diversity term, for all sensors, are shown in the Appendix B.

Table 6.19: One-class GAN results (J) - Unseen-attack.

J - Unseen attack			
<i>Sensor</i>	APCER	EER	BPCER@5
Cross Match	7.85%	38.90%	68.41%
Digital Persona	79.99%	49.94%	81.99%
Green Bit	70.43%	60.64%	93.46%
Hi Scan	43.70%	62.50%	96.38%
Time Series	79.42%	50.56%	96.67%

Despite the success in terms of generating fingerprints, the objective of using the Discriminator as a one-class classifier for PAD was unsuccessful, as shown in Table 6.19. Theoretically, by learning the distribution of bona fide images during GAN training, the Discriminator should be capable of detecting fake fingerprints, which will all serve as unknown presentation attacks. However, this did not happen in practice, with very negative results for all sensors. These results,

although far from what is intended, are not very far from what was obtained with the similar one-class approach presented in Engelsma et al. (2019) [13]. This work, which proposes the same methodology using a self-made large database comprising three types of input images (direct, raw FTIR and processed FTIR), only obtained an average True Detection Rate (TDR) of 49.8% with a False Detection Rate of 0.2%, and for some PAISp the value of TDR was less than 5%, which is very unsatisfactory. There is some work developed with the one-class approach, however, the performance of those solutions still remains below that of the binary classification approaches.

In conclusion, despite the successful generation of fingerprints, the use of the Discriminator as a one-class classifier for PAD did not meet the expectations. Nevertheless, these experiments were very useful to understand the whole concept of GANs, both theoretically and in terms of practical implementation and training routines, thus allowing to advance to the final model of this dissertation, an innovative approach also based on adversarial training.

6.5.11 Experiment K - Image-to-image GAN

The ultimate model developed in this dissertation is one more approach towards the goal of a robust solution to unknown attacks. Using the capabilities of the emerging GANs, the core idea is to enrich the Discriminator/Classifier's training with images synthesized by a *Liveness Generator*. Instead of random noise, the Generator's input is a presentation attack sample to be modified to look more similar to a bona fide fingerprint, but maintaining its biometric identity. This methodology is fully explained in Section 5.6.6.

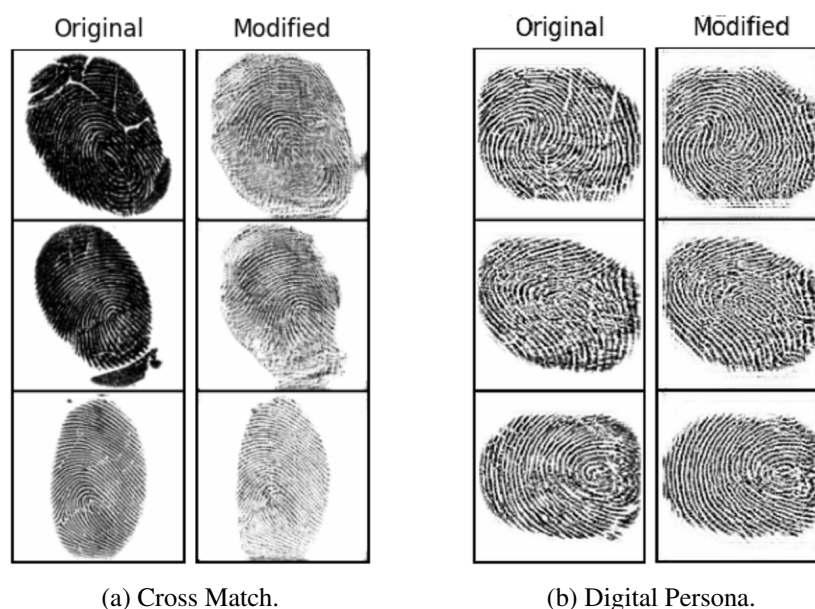


Figure 6.4: Examples of generated/modified fingerprints from two different sensors: Cross Match and Digital Persona.

First of all, by observing Figure 6.4 it is easy to note the success of the *Liveness Generator*. For example, with the sensor Cross Match, the main modification was to lighten the images, which

is the first difference noticed when comparing bona fide and attack samples from this subset. On the other hand, the main modification made by the Generator with images from the sensor Digital Persona was to clean the fingerprint by eliminating artifacts such as scratches and bubbles, details also commonly observed in the presentation attack samples of this subset. On other subsets, where sometimes the fake fingerprints have, at the same time, a different tone and small artifacts, the Generator was capable of performing both modifications, producing fake fingerprints more similar to real ones. Additionally, it is also important to note that the ridge-valley pattern (and, consequently, the biometric identity of the samples) was preserved regardless of the modifications, which was the ultimate goal for this part of the model. Several examples of modified fingerprints for all datasets are shown in Appendix C.

Table 6.20: Image-to-image GAN results - Unseen-attack framework - Joint training.

K - Unseen attack - Joint Training				
<i>Sensor</i>	<i>With Generator</i>	APCER	EER	BPCER@5
Cross Match	No	1.27%	3.97%	18.87%
	Yes	32.80%	12.77%	40.37%
Digital Persona	No	3.90%	7.38%	13.33%
	Yes	3.10%	7.50%	22.15%
Green Bit	No	4.48%	13.80%	26.58%
	Yes	4.68%	12.43%	26.90%
Hi Scan	No	2.40%	4.80%	12.38%
	Yes	16.20%	10.33%	21.28%
Time Series	No	0.50%	0.60%	5.13%
	Yes	12.00%	6.23%	13.13%

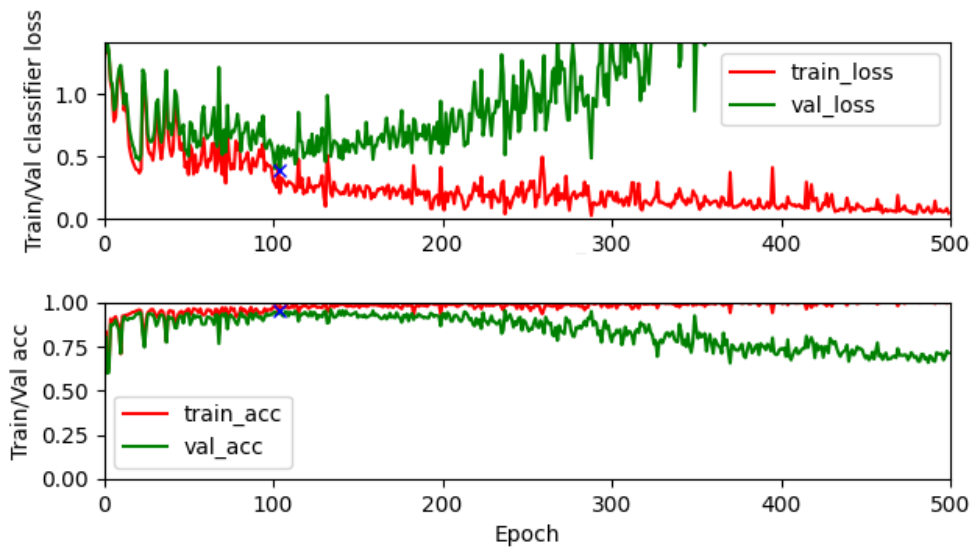


Figure 6.5: Evolution of the Classifier's loss and accuracy throughout training.

Despite the success of the Generator, this model has not achieved good performance regarding the PAD objective. Table 6.20 shows the results of this solution, where the baseline method ("Without Generator") corresponds to the use of the Discriminator as a simple CNN classifier. This experiment only included the unseen-attack framework because it was the single focus of the proposed methodology.

As observed in Table 6.20, the use of modified images did not bring improvements in detecting unknown attacks. In fact, the performance worsened with all sensors when the Generator was included and forced an adversarial training. The results of both approaches were similar solely with the Green Bit sensor, however, the use of the Generator only improved 1 of the 3 metrics presented. Observing the evolution of the Classifier's loss and accuracy throughout the training routine, shown in Figure 6.5, it is clear that overfitting may have occurred since the validation loss began to increase and the validation accuracy to decrease, while the training performance stayed at optimal values.

This brings a problem regarding the adversarial training approach that is being followed: on one hand, the Generator needs a large number of training epochs to generate good quality images, on the other hand, the long training routine leads to a decrease in the Classifier's performance with the validation data and, consequently, leads to poor results with the test data. Nevertheless, there is also the possibility that this methodology is not good for the purpose of detecting unknown attacks, as the perception of things by a human being is not the same as that of an artificial intelligence algorithm. Taking this into account, an additional experiment was conducted with the final model of this dissertation. The idea was to use the trained Generator to retrain solely the Classifier with the enriched data. Therefore, the training phase was divided into two parts:

1. Train the Generator and Classifier together for 500 epochs so that the Generator learns how to correctly modify the images.
2. Train the Classifier alone using the pre-trained Generator to modify the presentation attack samples and enrich the training data.

With this new phased training approach, the model lost its end-to-end identity, as we have to pre-train the Generator, however, it is a possibility to solve the overfitting problem observed in the joint training and, ultimately, help the model to improve its performance when facing unknown attacks. Despite overfitting starting at around 100 epochs in the joint training, that did not happen in the phased training, so after some initial experiments, the number of epochs was set to 250, keeping the model that achieved the best performance throughout training.

Table 6.21 summarizes the results of the new phased solution. It is straightforward to note a significant improvement regarding the use of the Generator, clearly outperforming the baseline method in 2 of the 5 (Digital Persona and Hi Scan). The baseline method achieved better performance with also 2 sensors (Cross Match and Green Bit). With the remainder, Time Series, although it is not straightforward to choose the better approach, by looking at Table A.46 (Appendix A, Section A.8) and computing the average for all evaluation metrics, the proposed model

Table 6.21: Image-to-image GAN results (K) - Unseen-attack framework - Phased training.

K - Unseen attack - Phased Training				
<i>Sensor</i>	<i>With Generator</i>	APCER	EER	BPCER@5
Cross Match	No	1.20%	4.47%	7.50%
	Yes	1.80%	5.27%	12.70%
Digital Persona	No	2.80%	8.33%	18.15%
	Yes	2.30%	5.58%	8.45%
Green Bit	No	4.88%	15.35%	32.10%
	Yes	4.98%	16.40%	33.70%
Hi Scan	No	1.90%	7.68%	13.58%
	Yes	2.00%	4.83%	13.33%
Time Series	No	0.50%	0.60%	5.13%
	Yes	0.57%	1.17%	0.73%

(with Generator) clearly obtained better results. Therefore, it is now correct to conclude that the proposed method improved the performance with 3 of the 5 sensors, thus ending positively the journey carried out through the domain of deep learning.

In conclusion, the final solution presented in this dissertation was a clear success in terms of the ability to modify presentation attack images to look more similar to bona fide fingerprints. This brings a whole new range of possibilities when it comes to improving the training of PAD systems using these synthesized images, especially when the available datasets are relatively small. At the same time, it raises concerns about the potential for attackers to take advantage of these generative models to create optimal fake fingerprints that can fool the systems much more efficiently, leading to a completely new type of presentation attacks. Moreover, the proposed PAD solution obtained quite satisfactory results, outperforming the baseline method in more than half of the experiments, but still left space for future developments and improvements. In any case, one can confidently state that the final experience of this dissertation was a successful innovation and a relevant contribution to the state of the art of fingerprint presentation attack detection.

This methodology is being further evaluated and, at the same time, a second submission for a scientific journal is being prepared. Due to the innovation of this model for the field of domain adaptation, the journal in question may be focused both in the areas of biometrics and machine learning.

Chapter 7

Conclusions and Future Work

The fingerprint is probably the most widely used biometric trait since the beginning of individual identification. More recently, the integration of biometric systems in smartphones and other devices used in daily life has led to the exponential growth in the use of fingerprints to perform frequent tasks, from account logins to online payments. Even with the emergence of new biometric traits, the fingerprint remains significant in the everyday life of millions of people and crucial to maintain security. Despite the almost optimal recognition performance of the algorithms running through the systems, they are still vulnerable to security threats. Thus, an individual can easily surpass security by using an artificially made fingerprint (presentation attack) that copies another one, this way matching to another individual or just concealing the attacker's identity. The most alarming fact is the ease with which anyone can build a fake fingerprint using countless different materials that are easy to obtain, such as silicone and gelatin. This led to the emergence of fingerprint presentation attack detection (PAD) solutions, which aim at detect if a presented fingerprint is either authentic or artificially fabricated. Among these solutions, the software only approaches are the most used and investigated due to the greater ease in integrating them into the existing systems. Since these solutions started to be developed, the growth has been enormous and the algorithms are now able to obtain minimal errors, however, almost always with optimistic positions in which they assume they know all possible attack materials, which is not true. In parallel with the evolution of PAD solutions, attackers have also evolved and new attack instruments are constantly emerging, more sophisticated and similar to bona fide fingerprints. In this way, a PAD system has a high probability of encountering an unknown attack material, which increases its detection error and creates a whole new security risk.

Taking this into account, this dissertation aimed to study different approaches for fingerprint PAD solutions, also addressing the problem of unknown attacks. The ideal method should achieve a low detection error even when encountering previously unseen attacks, that is, should be generalized to all PAI species and thus robust to possible novel ones. The extensive literature review, uptaded to include the most recent state of the art solutions, made it possible to acknowledge the most traditional approaches and also the most innovative and recent ones. The emergence of deep learning techniques, particularly CNNs, have been proving greater potential than other techniques,

however, they still need improvements and growth. Among all different approaches, domain adaptation and adversarial training methods have been showing very good results in many different tasks, including biometric PAD, so they were the focus of the experiential work of this dissertation regarding the robustness to unknown attacks.

7.1 Final Remarks

The experimental work of this dissertation can be represented as a journey through different PAD methods, towards contributions to the state of the art. The journey started where the preparation for this dissertation ended, with the consolidation of the preliminary experiments and optimization of the best conventional approach: the SVM classifier with handcrafted features. From here, the focus on deep learning solutions began, starting with a MLP using the same handcrafted features as the SVM. This MLP, despite its satisfactory performance, was later regularized with adversarial training and domain adaptation techniques towards an improvement regarding the detection of unknown attacks. After developing the methods that use features manually extracted from the images, the study of end-to-end approaches began. The initial experiments were conducted with CNNs, both pre-trained and trained from scratch, achieving good results that are inline with the state of the art solutions. One of the proposed models was then regularized with the same methodology as the MLP, which ended up outperforming the best results of the literature regarding unknown attacks and led to the submission of a research paper. Finally, the emerging GANs were tested and in their context, two models were developed. The first one, a simple GAN, was capable of generating fingerprint images from random noise but did not succeed in acting as a one-class classifier. The second one, the final model of this dissertation, was an image-to-image GAN that proved to be very effective in modifying presentation attack fingerprints to look more similar to bona fide samples. At the same time, it proved to be advantageous in improving the Classifier's performance when it comes to detecting unknown attacks, once again showing the potential of adversarial training techniques. This last approach had never been addressed for this purpose before and, therefore, composes one more innovative contribution of this dissertation to the field of fingerprint and biometric PAD.

All in all, one can confidently state that this dissertation ended up with a deep understanding in PAD fingerprint but also in Biometrics and Artificial Intelligence. At the same time, the experimental work led to the development of innovative solutions that surpassed the results of the literature and contributed with relevant advances for the state of the art.

7.2 Future Work

Future work based on this dissertation must, first of all, have access to more datasets and, therefore, a larger amount of data. There are many datasets for fingerprint PAD, including those from different editions of the Liveness Detection Competition (LivDet), which will continue to happen and is already in preparation for its 2021 edition. In fact, during this dissertation, a registration

was made for the next edition of this competition, which submission period ends in November of this year. In this way, a larger amount of data should be used not only to further validate the methods developed in this dissertation, but also to assist in refining them and in developing new approaches towards a possible submission to the Fingerprint LivDet2021 competition.

Regarding the methods themselves, different CNN architectures can be evaluated, with and without the proposed regularization, to further consolidate and optimize the results. These new models can be either pre-trained or trained from scratch. Moreover, the image-to-image GAN still left ground for progress and, despite the good results, it is possible to improve the results and try some variations. Firstly, different Classifier and Generator architectures should be evaluated, and the problem of overfitting should be further addressed. Finally, as an additional variation, one can experiment the implementation of the Matcher as an Artificial Neural Network to be trained or at least finetuned alongside the Classifier and the Generator.

Bibliography

- [1] Wael AbdAlmageed. “Anti-spoofing’s state-of-the-art: deep learning meets super sensors”. In: *Biometric Technology Today* 2019.7 (2019), pp. 8–11. ISSN: 0969-4765. DOI: [https://doi.org/10.1016/S0969-4765\(19\)30099-2](https://doi.org/10.1016/S0969-4765(19)30099-2). URL: <http://www.sciencedirect.com/science/article/pii/S0969476519300992>.
- [2] Zahid Akhtar, Christian Micheloni, and G.L. Foresti. “Biometric Liveness Detection: Challenges and Research Opportunities”. In: *IEEE Security Privacy* 13 (Sept. 2015), pp. 63–72. DOI: [10.1109/MSP.2015.116](https://doi.org/10.1109/MSP.2015.116).
- [3] Martín Arjovsky, Soumith Chintala, and Léon Bottou. “Wasserstein GAN”. In: *ArXiv abs/1701.07875* (2017).
- [4] ATVS. *ATVS Fake Fingerprint Database*. http://atvs.ii.uam.es/atvs/ffp_db.html. (Accessed on 01/08/2020). 2020.
- [5] Bayometric. *Minutiae Based Extraction in Fingerprint Recognition*. <https://www.bayometric.com/minutiae-based-extraction-fingerprint-recognition/>. (Accessed on 01/08/2020). 2020.
- [6] Christoph Busch and Ctirad Soušek. “Presentation attack detection methods for fingerprint recognition systems: A survey”. In: *IET Biometrics* 3 (Dec. 2014), pp. 219–233. DOI: [10.1049/iet-bmt.2013.0020](https://doi.org/10.1049/iet-bmt.2013.0020).
- [7] R. Cappelli et al. “Synthetic fingerprint-image generation”. In: *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*. Vol. 3. Sept. 2000, 471–474 vol.3. DOI: [10.1109/ICPR.2000.903586](https://doi.org/10.1109/ICPR.2000.903586).
- [8] Tarang Chugh and Anil Jain. *Fingerprint Presentation Attack Detection: Generalization and Efficiency*. Dec. 2018.
- [9] Corinna Cortes and Vladimir Vapnik. “Support-vector networks”. In: *Machine Learning* 20.3 (Sept. 1995), pp. 273–297. ISSN: 1573-0565. DOI: [10.1007/BF00994018](https://doi.org/10.1007/BF00994018). URL: <https://doi.org/10.1007/BF00994018>.
- [10] Luke Darlow and Benjamin Rosman. “Fingerprint minutiae extraction using deep learning”. In: Oct. 2017, pp. 22–30. DOI: [10.1109/BTAS.2017.8272678](https://doi.org/10.1109/BTAS.2017.8272678).

- [11] George Doddington et al. “Sheep, goats, lambs and wolves: A statistical analysis of speaker performance in the NIST 1998 speaker recognition evaluation”. In: *National Institut of Standards and Technology Gaithersburg* (1998), pp. 1–4.
- [12] Fadi Dornaika et al. “Chapter 32 - A Comparative Study of Image Segmentation Algorithms and Descriptors for Building Detection”. In: *Handbook of Neural Computation*. Ed. by Pijush Samui, Sanjiban Sekhar, and Valentina E. Balas. Academic Press, 2017, pp. 591–606. ISBN: 978-0-12-811318-9. DOI: <https://doi.org/10.1016/B978-0-12-811318-9.00032-6>. URL: <http://www.sciencedirect.com/science/article/pii/B9780128113189000326>.
- [13] Joshua J. Engelsma and Anil K. Jain. “Generalizing Fingerprint Spoof Detector: Learning a One-Class Classifier”. In: *CoRR* abs/1901.03918 (2019). arXiv: [1901.03918](https://arxiv.org/abs/1901.03918). URL: <http://arxiv.org/abs/1901.03918>.
- [14] Jianwei Fei et al. “Adversarial attacks on fingerprint liveness detection”. In: *EURASIP Journal on Image and Video Processing* 2020.1 (Jan. 2020), p. 1. ISSN: 1687-5281. DOI: [10.1186/s13640-020-0490-z](https://doi.org/10.1186/s13640-020-0490-z). URL: <https://doi.org/10.1186/s13640-020-0490-z>.
- [15] Pedro M. Ferreira et al. “Adversarial learning for a robust iris presentation attack detection method against unseen attack presentations”. In: *2019 International Conference of the Biometrics Special Interest Group, BIOSIG 2019 - Proceedings* (2019), pp. 1–12.
- [16] Pedro M. Ferreira et al. “Learning signer-invariant representations with adversarial training”. In: *Twelfth International Conference on Machine Vision (ICMV 2019)*. Ed. by Wolfgang Osten and Dmitry P. Nikolaev. Vol. 11433. International Society for Optics and Photonics. SPIE, 2020, pp. 918–926. DOI: [10.1117/12.2559534](https://doi.org/10.1117/12.2559534). URL: <https://doi.org/10.1117/12.2559534>.
- [17] Pedro Ferreira, Ana Sequeira, and Ana Rebelo. “A Fuzzy C-Means Algorithm for Fingerprint Segmentation”. In: vol. 9117. June 2015. DOI: [10.1007/978-3-319-19390-8n](https://doi.org/10.1007/978-3-319-19390-8n).
- [18] Rohit Gajawada et al. “Universal Material Translator : Towards Spoof Fingerprint Generalization”. In: (2019), pp. 1–8.
- [19] Javier Galbally et al. “An evaluation of direct attacks using fake fingers generated from ISO templates”. In: *Pattern Recognition Letters* 31 (June 2010), pp. 725–732. DOI: [10.1016/j.patrec.2009.09.032](https://doi.org/10.1016/j.patrec.2009.09.032).
- [20] Diego Garzón-Alvarado and Angélica Ramírez-Martínez. “A biochemical hypothesis on the formation of fingerprints using a turing patterns approach”. In: *Theoretical biology medical modelling* 8 (June 2011), p. 24. DOI: [10.1186/1742-4682-8-24](https://doi.org/10.1186/1742-4682-8-24).
- [21] Luca Ghiani, Gian Marcialis, and F. Roli. “Fingerprint liveness detection by local phase quantization”. In: Jan. 2012, pp. 537–540. ISBN: 978-1-4673-2216-4.
- [22] Luca Ghiani et al. “Fingerprint Liveness Detection using Binarized Statistical Image Features”. In: Sept. 2013, pp. 1–6. DOI: [10.1109/BTAS.2013.6712708](https://doi.org/10.1109/BTAS.2013.6712708).

- [23] Luca Ghiani et al. “Review of the Fingerprint Liveness Detection (LivDet) competition series: 2009 to 2015”. In: *Image and Vision Computing* 58 (2017), pp. 110–128. ISSN: 0262-8856. DOI: <https://doi.org/10.1016/j.imavis.2016.07.002>. URL: <http://www.sciencedirect.com/science/article/pii/S0262885616301111>.
- [24] Raoof Gholami and Nikoo Fakhari. “Chapter 27 - Support Vector Machine: Principles, Parameters, and Applications”. In: *Handbook of Neural Computation*. Ed. by Pijush Samui, Sanjiban Sekhar, and Valentina E. Balas. Academic Press, 2017, pp. 515–535. ISBN: 978-0-12-811318-9. DOI: <https://doi.org/10.1016/B978-0-12-811318-9.00027-2>. URL: <http://www.sciencedirect.com/science/article/pii/B9780128113189000272>.
- [25] Lázaro González Soler et al. “On the Impact of Different Fabrication Materials on Fingerprint Presentation Attack Detection”. In: June 2019.
- [26] Alan Goode. “Digital identity: solving the problem of trust”. In: *Biometric Technology Today* 2019.10 (2019), pp. 5–8. ISSN: 0969-4765. DOI: [https://doi.org/10.1016/S0969-4765\(19\)30142-0](https://doi.org/10.1016/S0969-4765(19)30142-0). URL: <http://www.sciencedirect.com/science/article/pii/S0969476519301420>.
- [27] Ian J. Goodfellow et al. “Generative Adversarial Networks”. In: *ArXiv* abs/1406.2661 (2014).
- [28] Robert Gove and Jorge Faytong. “Chapter 4 - Machine Learning and Event-Based Software Testing: Classifiers for Identifying Infeasible GUI Event Sequences”. In: ed. by Ali Hurson and Atif Memon. Vol. 86. *Advances in Computers*. Elsevier, 2012, pp. 109–135. DOI: <https://doi.org/10.1016/B978-0-12-396535-6.00004-1>. URL: <http://www.sciencedirect.com/science/article/pii/B9780123965356000041>.
- [29] Diego Gragnaniello et al. “Fingerprint Liveness Detection based on Weber Local Image Descriptor”. In: Sept. 2013. DOI: [10.1109/BIOMS.2013.6656148](https://doi.org/10.1109/BIOMS.2013.6656148).
- [30] Y. Han et al. “A Study on Evaluating the Uniqueness of Fingerprints Using Statistical Analysis”. In: vol. 3506. May 2005, pp. 75–76. DOI: [10.1007/11496618_34](https://doi.org/10.1007/11496618_34).
- [31] Zahra Hosseini. “Fingerprint vulnerability: A survey”. In: Apr. 2018, pp. 70–77. DOI: [10.1109/ICWR.2018.8387240](https://doi.org/10.1109/ICWR.2018.8387240).
- [32] ISO/IEC. “Information technology — Biometric presentation attack detection — Part 1: Framework”. In: *ISO/IEC 30107-1:2016* (Jan. 2016). URL: <https://www.iso.org/standard/53227.html>.
- [33] ISO/IEC. “Information Technology — Biometric presentation attack detection — Part 3: Testing and reporting”. In: *ISO/IEC 30107-3:2017* (Sept. 2017). URL: <https://www.iso.org/standard/67381.html>.
- [34] Phillip Isola et al. “Image-to-Image Translation with Conditional Adversarial Networks”. In: *CoRR* abs/1611.07004 (2016). arXiv: [1611.07004](https://arxiv.org/abs/1611.07004). URL: <http://arxiv.org/abs/1611.07004>.

- [35] A.K. Jain, A.A. Ross, and K. Nandakumar. *Introduction to Biometrics*. SpringerLink : Bücher. Springer US, 2011. ISBN: 9780387773261. URL: <https://books.google.pt/books?id=ZPt2xrZFtzkC>.
- [36] Anil Jain, Yi Chen, and Meltem Demirkus. “Pores and ridges: high-resolution fingerprint matching using level 3 features”. In: *IEEE transactions on pattern analysis and machine intelligence* 29 (Feb. 2007), pp. 15–27. DOI: [10.1109/TPAMI.2007.17](https://doi.org/10.1109/TPAMI.2007.17).
- [37] Kristijan Jankoski. *Python Fingerprint Recognition*. <https://github.com/kjanko/python-fingerprint-recognition>. 2019.
- [38] Yujia Jiang and Xin Liu. “Uniform Local Binary Pattern for Fingerprint Liveness Detection in the Gaussian Pyramid”. In: *Journal of Electrical and Computer Engineering* 2018 (Jan. 2018), pp. 1–9. DOI: [10.1155/2018/1539298](https://doi.org/10.1155/2018/1539298).
- [39] Changlong Jin et al. “Fingerprint Liveness Detection Based on Multiple Image Quality Features.” In: Jan. 2010, pp. 281–291.
- [40] Latha Jothi and Subbarayan Arumugam. “High-Resolution Fingerprint Matching using Level 3 Incipient Ridges and Scars”. In: *International Journal of Computer Applications* 48 (June 2012). DOI: [10.5120/7368-0130](https://doi.org/10.5120/7368-0130).
- [41] Gursimarpreet Kaur and DrChander Kant Verma. “Comparative Analysis of Biometric Modalities”. In: *International Journal of Advanced Research in Computer Science and Software Engineering* 4.4 (2014), pp. 2277–128.
- [42] Salman Khan et al. “A Guide to Convolutional Neural Networks for Computer Vision”. In: *Synthesis Lectures on Computer Vision* 8 (Feb. 2018), pp. 1–207. DOI: [10.2200/S00822ED1V01Y201712COV015](https://doi.org/10.2200/S00822ED1V01Y201712COV015).
- [43] Hakil Kim et al. “Fingerprint Generation and Presentation Attack Detection using Deep Neural Networks”. In: Mar. 2019, pp. 375–378. DOI: [10.1109/MIPR.2019.00074](https://doi.org/10.1109/MIPR.2019.00074).
- [44] Samruddhi S. Kulkarni and Hemprasad Y. Patil. “Article: Survey on Fingerprint Spoofing, Detection Techniques and Databases”. In: *IJCA Proceedings on National Conference on Advances in Computing* NCAC 2015.7 (Dec. 2015). Full text available, pp. 30–33.
- [45] Douglas Kunda and Mumbi Chishimba. “A Survey of Android Mobile Phone Authentication Schemes”. In: *Mobile Networks and Applications* (Aug. 2018). DOI: [10.1007/s11036-018-1099-7](https://doi.org/10.1007/s11036-018-1099-7).
- [46] Haozhe Liu et al. *A Zero-Shot based Fingerprint Presentation Attack Detection System*. 2020. arXiv: [2002.04908](https://arxiv.org/abs/2002.04908) [cs.CV].
- [47] LivDet. *LivDet Competition*. <http://livdet.org/>. (Accessed on 01/08/2020). 2020.
- [48] Andrew L. Maas. “Rectifier Nonlinearities Improve Neural Network Acoustic Models”. In: 2013.

- [49] Aakarsh Malhotra et al. “Chapter 6 - Fingerphoto Authentication Using Smartphone Camera Captured Under Varying Environmental Conditions”. In: *Human Recognition in Unconstrained Environments*. Ed. by Maria De Marsico, Michele Nappi, and Hugo Proença. Academic Press, 2017, pp. 119–144. ISBN: 978-0-08-100705-1. DOI: <https://doi.org/10.1016/B978-0-08-100705-1.00006-3>. URL: <http://www.sciencedirect.com/science/article/pii/B9780081007051000063>.
- [50] Manivannan and Padma. “Comparative and Analysis of Biometric Systems”. In: *International Journal on Computer Science and Engineering* 3.5 (2011), pp. 2156–2162.
- [51] Emanuela Marasco and Arun Ross. “A Survey on Antispoofing Schemes for Fingerprint Recognition Systems”. In: *ACM Comput. Surv.* 47.2 (Nov. 2014), 28:1–28:36. ISSN: 0360-0300. DOI: [10.1145/2617756](https://doi.org/10.1145/2617756). URL: <http://doi.acm.org/10.1145/2617756>.
- [52] Emanuela Marasco and Carlo Sansone. “An anti-spoofing technique using multiple textural features in fingerprint scanners”. In: Oct. 2010, pp. 8–14. DOI: [10.1109/BIOMS.2010.5610440](https://doi.org/10.1109/BIOMS.2010.5610440).
- [53] Emanuela Marasco and Carlo Sansone. “On the Robustness of Fingerprint Liveness Detection Algorithms against New Materials used for Spoofing.” In: Jan. 2011, pp. 553–558.
- [54] Sébastien Marcel et al., eds. *Handbook of Biometric Anti-Spoofing - Presentation Attack Detection, Second Edition*. Advances in Computer Vision and Pattern Recognition. Springer, 2019. ISBN: 978-3-319-92626-1. DOI: [10.1007/978-3-319-92627-8](https://doi.org/10.1007/978-3-319-92627-8). URL: <https://doi.org/10.1007/978-3-319-92627-8>.
- [55] M. E. Maron. “Automatic Indexing: An Experimental Inquiry”. In: *J. ACM* 8.3 (July 1961), pp. 404–417. ISSN: 0004-5411. DOI: [10.1145/321075.321084](https://doi.org/10.1145/321075.321084). URL: <https://doi.org/10.1145/321075.321084>.
- [56] Tsutomu Matsumoto et al. “Impact of Artificial "Gummy" Fingers on Fingerprint Systems.” In: *Datenschutz und Datensicherheit* 26 (Apr. 2002). DOI: [10.1117/12.462719](https://doi.org/10.1117/12.462719).
- [57] Hengameh Mirzaalian, Mohamed E. Hussein, and Wael Abd-Almageed. “On the Effectiveness of Laser Speckle Contrast Imaging and Deep Neural Networks for Detecting Known and Unknown Fingerprint Presentation Attacks”. In: *CoRR* abs/1906.02595 (2019). arXiv: [1906.02595](https://arxiv.org/abs/1906.02595). URL: <http://arxiv.org/abs/1906.02595>.
- [58] C. Maxine Most. *The Global Biometrics and Mobility Report: The Convergence of Commerce and Privacy. Market Analysis and Forecasts 2016-2020, a preview*. Tech. rep. Acuity Market Intelligence, Aug. 2017.
- [59] MSU-FPAD. *Michigan State University Fingerprint Presentation Attack Dataset*. http://biometrics.cse.msu.edu/Publications/Databases/MSU_FPAD/. (Accessed on 01/08/2020). 2020.
- [60] Biometrics Research Group MSU. *What is Biometrics?* <http://biometrics.cse.msu.edu/info/index.html>. (Accessed on 01/29/2020). 2009.

- [61] Valerio Mura et al. “LivDet 2015 - Fingerprint Liveness Detection Competition 2015”. In: Sept. 2015. DOI: [10.1109/BTAS.2015.7358776](https://doi.org/10.1109/BTAS.2015.7358776).
- [62] Muhammad-Sajid Mushtaq and Abdelhamid Mellouk. “2 - Methodologies for Subjective Video Streaming QoE Assessment”. In: *Quality of Experience Paradigm in Multimedia Services*. Ed. by Muhammad-Sajid Mushtaq and Abdelhamid Mellouk. Elsevier, 2017, pp. 27–57. ISBN: 978-1-78548-109-3. DOI: <https://doi.org/10.1016/B978-1-78548-109-3.50002-3>. URL: <http://www.sciencedirect.com/science/article/pii/B9781785481093500023>.
- [63] Prakash Nadkarni. “Chapter 10 - Core Technologies: Data Mining and “Big Data””. In: *Clinical Research Computing*. Ed. by Prakash Nadkarni. Academic Press, 2016, pp. 187–204. ISBN: 978-0-12-803130-8. DOI: <https://doi.org/10.1016/B978-0-12-803130-8.00010-5>. URL: <http://www.sciencedirect.com/science/article/pii/B9780128031308000105>.
- [64] R.C. Neath and M.S. Johnson. “Discrimination and Classification”. In: *International Encyclopedia of Education (Third Edition)*. Ed. by Penelope Peterson, Eva Baker, and Barry McGaw. Third Edition. Oxford: Elsevier, 2010, pp. 135–141. ISBN: 978-0-08-044894-7. DOI: <https://doi.org/10.1016/B978-0-08-044894-7.01312-9>. URL: <http://www.sciencedirect.com/science/article/pii/B9780080448947013129>.
- [65] R. F. Nogueira, R. de Alencar Lotufo, and R. Campos Machado. “Fingerprint Liveness Detection Using Convolutional Neural Networks”. In: *IEEE Transactions on Information Forensics and Security* 11.6 (June 2016), pp. 1206–1213. ISSN: 1556-6021. DOI: [10.1109/TIFS.2016.2520880](https://doi.org/10.1109/TIFS.2016.2520880).
- [66] Rodrigo Nogueira, Roberto Lotufo, and Rubens Machado. “Evaluating software-based fingerprint liveness detection using Convolutional Networks and Local Binary Patterns”. In: (Aug. 2015). DOI: [10.1109/BIOMS.2014.6951531](https://doi.org/10.1109/BIOMS.2014.6951531).
- [67] T. Ojala, M. Pietikainen, and T. Maenpaa. “Multiresolution gray-scale and rotation invariant texture classification with local binary patterns”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24.7 (July 2002), pp. 971–987. ISSN: 1939-3539. DOI: [10.1109/TPAMI.2002.1017623](https://doi.org/10.1109/TPAMI.2002.1017623).
- [68] Ville Ojansivu and Janne Heikkilä. “Blur Insensitive Texture Classification Using Local Phase Quantization”. In: *Image and Signal Processing*. Ed. by Abderrahim Elmoataz et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 236–243. ISBN: 978-3-540-69905-7.
- [69] Giulia Orrù et al. “LivDet in Action - Fingerprint Liveness Detection Competition 2019”. In: (2019). arXiv: [1905.00639](https://arxiv.org/abs/1905.00639). URL: <http://arxiv.org/abs/1905.00639>.

- [70] Ranjit Panigrahi and Samarjeet Borah. “1 - Classification and Analysis of Facebook Metrics Dataset Using Supervised Classifiers”. In: *Social Network Analytics*. Ed. by Nilanjan Dey et al. Academic Press, 2019, pp. 1–19. ISBN: 978-0-12-815458-8. DOI: <https://doi.org/10.1016/B978-0-12-815458-8.00001-3>. URL: <http://www.sciencedirect.com/science/article/pii/B9780128154588000013>.
- [71] E. Park et al. “Presentation Attack Detection Using a Tiny Fully Convolutional Network”. In: *IEEE Transactions on Information Forensics and Security* 14.11 (2019), pp. 3016–3025.
- [72] Eunsoo Park et al. “Fingerprint Liveness Detection Using CNN Features of Random Sample Patches”. In: Sept. 2016, pp. 1–4. DOI: [10.1109/BIOSIG.2016.7736923](https://doi.org/10.1109/BIOSIG.2016.7736923).
- [73] A N Mukunda Rao and Sumitha M. “MULTIPLE FEATURES BASED FINGERPRINT IDENTIFICATION SYSTEM”. In: *International Journal of Research in Engineering and Technology* 4 (Sept. 2015), p. 142.
- [74] Ajita Rattani and Arun Ross. “Automatic Adaptation of Fingerprint Liveness Detector to New Spoof Materials”. In: Sept. 2014. DOI: [10.1109/BTAS.2014.6996254](https://doi.org/10.1109/BTAS.2014.6996254).
- [75] Ajita Rattani, Walter Scheirer, and Arun Ross. “Open Set Fingerprint Spoof Detection Across Novel Fabrication Materials”. In: *IEEE Transactions on Information Forensics and Security* (Nov. 2015). DOI: [10.1109/TIFS.2015.2464772](https://doi.org/10.1109/TIFS.2015.2464772).
- [76] Murat Sazli. “A brief review of feed-forward neural networks”. In: *Communications, Faculty Of Science, University of Ankara* 50 (Jan. 2006), pp. 11–17. DOI: [10.1501/0003168](https://doi.org/10.1501/0003168).
- [77] J. Scheibert et al. “The Role of Fingerprints in the Coding of Tactile Information Probed with a Biomimetic Sensor”. In: *Science* 323.5920 (2009), pp. 1503–1506. ISSN: 0036-8075. DOI: [10.1126/science.1166467](https://doi.org/10.1126/science.1166467). eprint: <https://science.sciencemag.org/content/323/5920/1503.full.pdf>. URL: <https://science.sciencemag.org/content/323/5920/1503>.
- [78] Ana Sequeira. “Liveness Detection and Robust Recognition in Iris and Fingerprint Biometric Systems”. PhD thesis. Dec. 2015.
- [79] Ana F. Sequeira and Jaime S. Cardoso. “Fingerprint liveness detection in the presence of capable intruders”. In: *Sensors (Switzerland)* 15.6 (2015), pp. 14615–14638. ISSN: 14248220. DOI: [10.3390/s150614615](https://doi.org/10.3390/s150614615).
- [80] B.W. Silverman and M. C. Jones. “E. Fix and J.L. Hodges(1951): an important contribution to nonparametric discriminant analysis and density estimation”. In: *International Statistical Review* 57.3 (1989), pp. 233–247. URL: <http://oro.open.ac.uk/28327/>.
- [81] Karen Simonyan and Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2014. arXiv: [1409.1556](https://arxiv.org/abs/1409.1556) [cs.CV].
- [82] Kai Tang et al. “A Novel Fingerprint Sensing Technology Based on Electrostatic Imaging”. In: *Sensors* 18 (Sept. 2018), p. 3050. DOI: [10.3390/s18093050](https://doi.org/10.3390/s18093050).

- [83] Sergios Theodoridis. “Chapter 2 - Probability and Stochastic Processes”. In: *Machine Learning*. Ed. by Sergios Theodoridis. Oxford: Academic Press, 2015, pp. 9–51. ISBN: 978-0-12-801522-3. DOI: <https://doi.org/10.1016/B978-0-12-801522-3.00002-1>. URL: <http://www.sciencedirect.com/science/article/pii/B9780128015223000021>.
- [84] Rubén Tolosana et al. “Biometric Presentation Attack Detection: Beyond the Visible Spectrum”. In: *IEEE Transactions on Information Forensics and Security* 15 (2020), pp. 1261–1275.
- [85] Shruti Upadhyaya and RAAJ Ramsankaran. “Support Vector Machine (SVM) based Rain Area Detection from Kalpana-1 Satellite Data”. In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* (Dec. 2014). DOI: [10.5194/isprsannals-II-8-21-2014](https://doi.org/10.5194/isprsannals-II-8-21-2014).
- [86] Leonardo Vanneschi and Mauro Castelli. “Multilayer Perceptrons”. In: *Encyclopedia of Bioinformatics and Computational Biology*. Ed. by Shoba Ranganathan et al. Oxford: Academic Press, 2019, pp. 612–620. ISBN: 978-0-12-811432-2. DOI: <https://doi.org/10.1016/B978-0-12-809633-8.20339-7>. URL: <http://www.sciencedirect.com/science/article/pii/B9780128096338203397>.
- [87] René Vidal et al. “Mathematics of Deep Learning”. In: *CoRR* abs/1712.04741 (2017). arXiv: [1712.04741](https://arxiv.org/abs/1712.04741). URL: <http://arxiv.org/abs/1712.04741>.
- [88] Peter Warman and Roland Ennos. “Fingerprints are unlikely to increase the friction of primate fingerpads”. In: *The Journal of experimental biology* 212 (Aug. 2009), pp. 2016–22. DOI: [10.1242/jeb.028977](https://doi.org/10.1242/jeb.028977).
- [89] Karl Weiss, Taghi M. Khoshgoftaar, and Ding Ding Wang. *A survey of transfer learning*. Vol. 3. 1. Springer International Publishing, 2016. ISBN: 4053701600. DOI: [10.1186/s40537-016-0043-6](https://doi.org/10.1186/s40537-016-0043-6).
- [90] Khin Nandar Win et al. “Fingerprint classification and identification algorithms for criminal investigation: A survey”. In: *Future Generation Computer Systems* (2019). ISSN: 0167-739X. DOI: <https://doi.org/10.1016/j.future.2019.10.019>. URL: <http://www.sciencedirect.com/science/article/pii/S0167739X19315109>.
- [91] Jian Wu et al. “Mixed Pattern Matching-Based Traffic Abnormal Behavior Recognition”. In: *TheScientificWorldJournal* 2014 (Mar. 2014), p. 834013. DOI: [10.1155/2014/834013](https://doi.org/10.1155/2014/834013).
- [92] N. Yager and T. Dunstone. “The Biometric Menagerie”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.2 (Feb. 2010), pp. 220–230. ISSN: 1939-3539. DOI: [10.1109/TPAMI.2008.291](https://doi.org/10.1109/TPAMI.2008.291).
- [93] Dingdong Yang et al. “Diversity-Sensitive Conditional Generative Adversarial Networks”. In: *International Conference on Learning Representations*. 2019. URL: <https://openreview.net/forum?id=rJliMh09F7>.

- [94] Jason Yosinski et al. “How transferable are features in deep neural networks?” In: Jan. 2014, pp. 3320–3328.
- [95] C. Yuan et al. “Fingerprint Liveness Detection Using an Improved CNN With Image Scale Equalization”. In: *IEEE Access* 7 (2019), pp. 26953–26966.
- [96] Chengsheng Yuan et al. “Semi-supervised stacked autoencoder-based deep hierarchical semantic feature for real-time fingerprint liveness detection”. In: *Journal of Real-Time Image Processing* 17.1 (Feb. 2020), pp. 55–71. ISSN: 1861-8219. DOI: [10.1007/s11554-019-00928-0](https://doi.org/10.1007/s11554-019-00928-0). URL: <https://doi.org/10.1007/s11554-019-00928-0>.
- [97] Hui Zhang, Zhenan Sun, and Tieniu Tan. “Contact Lens Detection Based on Weighted LBP”. In: Aug. 2010, pp. 4279–4282. DOI: [10.1109/ICPR.2010.1040](https://doi.org/10.1109/ICPR.2010.1040).
- [98] Yongliang Zhang et al. “Slim-ResCNN: A deep Residual Convolutional Neural Network for Fingerprint Liveness Detection”. In: *IEEE Access* PP (July 2019), pp. 1–1. DOI: [10.1109/ACCESS.2019.2927357](https://doi.org/10.1109/ACCESS.2019.2927357).
- [99] Jun-Yan Zhu et al. “Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks”. In: *CoRR* abs/1703.10593 (2017). arXiv: [1703.10593](https://arxiv.org/abs/1703.10593). URL: <http://arxiv.org/abs/1703.10593>.

Appendix A

Complete Results

In this appendix it is possible to visualize the complete results for each dataset and for each different PAI, rather than an average of all materials. These results are presented (from Table [A.1](#) to [A.46](#)) with the performance metrics described in Section [6.2](#).

A.1 SVM vs MLP Complete Results

Table A.1: SVM vs MLP complete results - One-Attack framework - Cross Match.

SVM vs MLP - One attack - CrossMatch									
Attack	Classifier	Without segmentation Features = Intensity + LBP + LPQ							
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Body Double	SVM MLP	89.50%	19.70%	1.40%	6.70%	18.80%	7.70%	3.00%	
		99.00%	0.33%	1.67%	0.00%	3.93%	0.00%	0.00%	
Ecoflex	SVM MLP	90.40%	5.80%	17.40%	3.40%	14.90%	2.80%	0.80%	
		98.00%	2.33%	1.63%	6.50%	4.63%	0.27%	0.00%	
Playdoh	SVM MLP	92.10%	1.80%	25.90%	4.20%	14.50%	3.80%	2.40%	
		99.67%	0.37%	0.33%	0.00%	0.00%	0.00%	0.00%	
Attack	Classifier	With segmentation Features = Intensity + LBP + LPQ							
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Body Double	SVM MLP	94.90%	4.10%	6.10%	4.60%	9.10%	4.00%	2.40%	
		99.83%	0.00%	0.33%	0.00%	0.00%	0.00%	0.00%	
Ecoflex	SVM MLP	92.20%	0.60%	22.00%	2.60%	5.90%	2.00%	1.40%	
		99.33%	0.40%	0.94%	0.30%	0.30%	0.00%	0.00%	
Playdoh	SVM MLP	80.20%	24.60%	5.50%	5.70%	9.90%	5.50%	5.10%	
		99.50%	0.30%	0.69%	0.00%	2.59%	0.00%	0.00%	

Table A.2: SVM vs MLP complete results - Unseen-Attack framework - Cross Match.

SVM vs MLP - Unseen attack - CrossMatch									
Attack	Classifier	Without segmentation							
		Features = Intensity + LBP + LPQ							
Body Double	SVM MLP	Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
		85.90%	7.70%	20.40%	14.40%	40.90%	26.90%	17.20%	
		80.90%	0.00%	38.50%	6.30%	31.70%	8.50%	3.70%	
Ecoflex	SVM MLP	80.40%	0.30%	57.90%	7.80%	31.90%	10.80%	7.00%	
		88.70%	1.60%	21.10%	5.60%	24.30%	8.60%	3.10%	
		95.10%	2.20%	13.00%	7.60%	22.20%	8.60%	7.20%	
Playdoh	SVM MLP	78.30%	0.20%	44.10%	7.10%	43.20%	15.30%	5.90%	
Attack	Classifier	With segmentation							
		Features = Intensity + LBP + LPQ							
Body Double	SVM MLP	Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
		86.50%	19.40%	7.60%	11.60%	29.50%	20.40%	13.00%	
		91.20%	0.20%	17.40%	1.30%	1.70%	0.40%	0.00%	
Ecoflex	SVM MLP	96.60%	1.60%	7.00%	3.80%	9.20%	2.20%	1.00%	
		98.80%	0.00%	2.40%	0.00%	0.00%	0.00%	0.00%	
		80.00%	10.20%	48.90%	23.20%	84.20%	63.30%	49.30%	
Playdoh	SVM MLP	54.00%	0.00%	93.80%	11.70%	45.10%	22.30%	11.30%	

Table A.3: SVM vs MLP complete results - One-Attack framework - Digital Persona.

SVM vs MLP - One attack - Digital Persona									
Attack	Classifier	Without segmentation Intensity + LBP + LPQ							
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	SVM MLP	98.80%	0.80%	1.60%	2.00%	0.00%	0.00%	0.00%	
		96.00%	0.00%	7.50%	2.30%	45.30%	1.70%	0.00%	
Gelatine	SVM MLP	66.70%	0.00%	100.00%	17.70%	79.50%	53.80%	36.50%	
		98.00%	2.50%	1.30%	0.40%	3.10%	0.00%	0.00%	
Latex	SVM MLP	78.50%	25.30%	10.00%	9.60%	36.10%	22.50%	13.30%	
		93.70%	0.70%	12.70%	2.70%	22.90%	5.30%	1.00%	
Wood glue	SVM MLP	82.20%	19.10%	12.40%	15.30%	40.60%	21.30%	16.10%	
		96.30%	0.00%	7.10%	2.20%	3.90%	1.10%	0.60%	
Attack	Classifier	With segmentation Intensity + LBP + LPQ							
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	SVM MLP	98.00%	3.20%	0.80%	2.00%	2.40%	0.00%	0.00%	
		96.70%	0.70%	5.70%	1.60%	26.80%	0.00%	0.00%	
Gelatine	SVM MLP	66.70%	0.00%	0.80%	2.00%	63.10%	38.20%	24.10%	
		98.30%	0.60%	2.60%	1.60%	2.20%	0.00%	0.00%	
Latex	SVM MLP	87.90%	0.50%	0.80%	2.00%	38.20%	14.90%	8.40%	
		99.30%	1.20%	0.00%	0.50%	0.50%	0.00%	0.00%	
Wood glue	SVM MLP	83.50%	18.40%	0.80%	2.00%	37.80%	22.10%	16.90%	
		95.30%	0.00%	9.10%	2.10%	10.20%	2.70%	1.10%	

Table A.4: SVM vs MLP complete results - Unseen-Attack framework - Digital Persona.

SVM vs MLP - Unseen attack - Digital Persona									
Attack	Classifier	Without segmentation Intensity + LBP + LPQ							
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	SVM MLP	87.40%	0.80%	24.40%	7.20%	20.00%	8.80%	4.80%	
		98.80%	0.00%	2.40%	0.70%	4.40%	0.00%	0.00%	
Gelatine	SVM MLP	66.70%	0.00%	100.00%	14.80%	74.00%	44.80%	26.40%	
		96.80%	0.00%	6.40%	0.70%	0.70%	0.00%	0.00%	
Latex	SVM MLP	92.70%	5.60%	12.40%	9.20%	47.20%	12.40%	7.20%	
		94.40%	0.00%	11.20%	4.60%	49.20%	16.50%	1.20%	
Wood glue	SVM MLP	89.80%	12.00%	2.80%	6.80%	55.60%	10.40%	3.60%	
		99.20%	0.00%	1.60%	0.00%	0.00%	0.00%	0.00%	
Attack	Classifier	With segmentation Intensity + LBP + LPQ							
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	SVM MLP	90.80%	0.00%	18.40%	3.20%	4.80%	2.00%	1.20%	
		97.20%	0.00%	5.60%	0.00%	2.60%	0.00%	0.00%	
Gelatine	SVM MLP	66.70%	0.00%	100.00%	24.00%	93.60%	78.80%	63.60%	
		99.20%	0.00%	1.60%	0.00%	0.00%	0.00%	0.00%	
Latex	SVM MLP	86.60%	7.60%	30.80%	19.20%	62.40%	38.40%	27.60%	
		95.40%	0.00%	9.20%	1.40%	16.20%	2.10%	0.00%	
Wood glue	SVM MLP	90.90%	4.60%	27.20%	13.20%	52.80%	24.40%	15.60%	
		98.80%	0.00%	2.40%	0.40%	4.80%	0.00%	0.00%	

Table A.5: SVM vs MLP complete results - One-Attack framework - Green Bit.

SVM vs MLP - One attack - Green Bit									
Attack	Classifier	Without segmentation Intensity + LBP + LPQ							
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	SVM MLP	95.80%	6.80%	1.60%	3.60%	4.80%	1.60%	0.80%	
		98.30%	0.60%	2.60%	1.10%	1.10%	0.00%	0.00%	
		87.70%	0.80%	35.30%	1.60%	3.20%	0.00%	0.00%	
Gelatine	SVM MLP	99.30%	0.00%	1.20%	0.00%	3.50%	0.00%	0.00%	
		85.60%	16.30%	8.40%	10.80%	15.30%	11.20%	10.80%	
Latex	SVM MLP	93.00%	2.00%	11.20%	1.40%	23.70%	7.80%	1.40%	
		91.20%	4.30%	26.50%	5.20%	22.10%	22.10%	21.30%	
Wood glue	SVM MLP	99.30%	0.00%	1.30%	0.00%	0.00%	0.00%	0.00%	
Attack	Classifier	With segmentation Intensity + LBP + LPQ							
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	SVM MLP	92.60%	14.50%	0.40%	26.50%	1.20%	0.40%	0.40%	
		96.70%	4.30%	1.90%	2.50%	7.90%	0.00%	0.00%	
		77.20%	0.00%	68.30%	3.20%	3.60%	2.00%	1.20%	
Gelatine	SVM MLP	91.00%	15.00%	2.70%	6.70%	15.50%	8.90%	4.80%	
		82.40%	14.60%	26.50%	18.50%	28.50%	25.30%	23.30%	
Latex	SVM MLP	95.00%	5.40%	4.50%	6.70%	18.90%	4.00%	2.10%	
		74.70%	27.10%	26.50%	16.50%	34.90%	28.90%	26.50%	
Wood glue	SVM MLP	87.70%	15.70%	9.10%	9.60%	22.30%	16.00%	7.00%	

Table A.6: SVM vs MLP complete results - Unseen-Attack framework - Green Bit.

SVM vs MLP - Unseen attack - Green Bit									
Attack	Classifier	Without segmentation Intensity + LBP + LPQ							
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	SVM MLP	94.80%	7.20%	3.20%	4.80%	12.00%	4.80%	1.60%	
		97.40%	0.00%	5.20%	0.80%	2.80%	0.00%	0.00%	
Gelatine	SVM MLP	86.30%	0.00%	41.20%	2.80%	5.60%	1.20%	0.80%	
		59.00%	0.00%	81.60%	30.40%	97.50%	82.20%	70.60%	
Latex	SVM MLP	96.00%	0.10%	15.60%	1.60%	4.80%	0.40%	0.40%	
		94.80%	0.00%	10.40%	0.90%	0.90%	0.00%	0.00%	
Wood glue	SVM MLP	97.10%	3.10%	2.00%	2.80%	4.80%	1.20%	0.80%	
		92.40%	0.00%	15.20%	3.80%	37.00%	3.00%	1.20%	
Attack	Classifier	With segmentation Intensity + LBP + LPQ							
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	SVM MLP	94.60%	7.60%	3.20%	5.20%	12.40%	4.40%	2.00%	
		99.60%	0.80%	0.00%	0.50%	0.50%	0.00%	0.00%	
Gelatine	SVM MLP	90.40%	1.20%	26.40%	6.00%	27.60%	6.40%	3.20%	
		95.20%	0.00%	9.60%	0.80%	0.80%	0.00%	0.00%	
Latex	SVM MLP	95.50%	2.40%	10.80%	4.80%	21.60%	4.40%	1.20%	
		98.80%	0.40%	2.00%	1.60%	1.60%	0.00%	0.00%	
Wood glue	SVM MLP	89.00%	0.10%	54.40%	3.20%	14.00%	2.00%	1.20%	
		97.60%	1.60%	3.20%	1.50%	5.40%	0.80%	0.50%	

Table A.7: SVM vs MLP complete results - One-Attack framework - Hi Scan.

SVM vs MLP - One attack - Hi Scan									
Attack	Classifier	Without segmentation Intensity + LBP + LPQ							
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	SVM MLP	91.80%	9.60%	6.80%	8.00%	13.70%	8.40%		6.00%
		99.00%	0.00%	2.00%	0.50%	0.50%	0.00%		
		81.80%	0.00%	54.60%	8.40%	10.80%	8.80%		6.80%
Gelatine	SVM MLP	98.00%	2.50%	1.30%	0.50%	0.50%	0.50%		0.00%
		85.50%	0.30%	57.00%	11.20%	18.50%	12.90%		12.00%
Latex	SVM MLP	96.70%	0.00%	7.10%	1.50%	1.50%	0.00%		0.00%
		72.90%	29.10%	19.30%	27.70%	33.30%	31.30%		30.90%
Wood glue	SVM MLP	97.70%	1.60%	3.30%	1.30%	1.80%	0.50%		0.50%
Attack	Classifier	With segmentation Intensity + LBP + LPQ							
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	SVM MLP	92.00%	13.30%	2.80%	5.60%	25.70%	8.40%		2.40%
		96.50%	0.60%	6.40%	1.90%	19.90%	3.00%		0.00%
		77.50%	0.00%	67.50%	9.20%	30.10%	14.90%		11.20%
Gelatine	SVM MLP	93.90%	0.80%	11.30%	4.90%	13.80%	5.50%		2.10%
		67.00%	40.70%	10.00%	22.90%	39.40%	33.30%		31.30%
Latex	SVM MLP	92.30%	0.60%	16.20%	11.20%	27.80%	19.40%		18.30%
		41.60%	72.50%	2.00%	22.10%	44.60%	41.40%		40.20%
Wood glue	SVM MLP	98.80%	0.00%	2.30%	0.00%	1.80%	0.00%		0.00%

Table A.8: SVM vs MLP complete results - Unseen-Attack framework - Hi Scan.

SVM vs MLP - Unseen attack - Hi Scan									
Attack	Classifier	Without segmentation Intensity + LBP + LPQ							
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	SVM MLP	89.80%	0.40%	20.00%	5.60%	16.80%	9.60%	2.80%	
		99.00%	0.00%	2.00%	0.00%	0.00%	0.00%	0.00%	
Gelatine	SVM MLP	92.80%	9.60%	2.40%	4.00%	9.20%	3.20%	1.60%	
		95.80%	0.00%	8.40%	1.90%	20.50%	1.90%	0.00%	
Latex	SVM MLP	95.80%	3.60%	6.00%	4.80%	25.60%	4.00%	1.20%	
		86.60%	0.00%	26.80%	0.40%	0.40%	0.00%	0.00%	
Wood glue	SVM MLP	89.90%	12.30%	1.20%	6.00%	24.40%	6.00%	1.20%	
		95.80%	0.00%	8.40%	0.00%	25.50%	0.00%	0.00%	
Attack	Classifier	With segmentation Intensity + LBP + LPQ							
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	SVM MLP	86.60%	21.20%	5.60%	13.20%	42.00%	32.00%	16.40%	
		96.40%	1.20%	6.00%	2.90%	10.90%	1.70%	0.40%	
Gelatine	SVM MLP	75.70%	1.00%	70.80%	17.60%	66.40%	46.00%	32.80%	
		88.80%	0.00%	22.40%	6.20%	34.30%	8.80%	4.60%	
Latex	SVM MLP	44.20%	68.80%	16.80%	29.60%	58.00%	40.40%	36.40%	
		95.00%	0.00%	10.00%	1.50%	8.90%	0.80%	0.00%	
Wood glue	SVM MLP	87.50%	5.00%	42.40%	32.40%	67.60%	42.40%	38.00%	
		99.00%	0.00%	2.00%	1.50%	1.50%	0.00%	0.00%	

Table A.9: SVM vs MLP complete results - One-Attack framework - Time Series.

SVM vs MLP - One attack - Time Series									
Attack	Classifier	Without segmentation Features = Intensity + LBP + LPQ							
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Body Double	SVM MLP	87.10%	11.30%	14.40%	12.40%	34.70%	20.70%	14.30%	
		99.80%	0.00%	0.30%	0.00%	2.70%	0.00%	0.00%	
Ecoflex	SVM MLP	92.90%	3.60%	14.10%	6.30%	16.80%	6.80%	4.10%	
		99.40%	0.40%	0.80%	0.20%	5.10%	0.00%	0.00%	
Playdoh	SVM MLP	82.00%	16.70%	21.80%	6.30%	17.10%	6.10%	4.90%	
		99.80%	0.00%	0.30%	0.00%	3.80%	0.00%	0.00%	
Attack	Classifier	With segmentation Feature = Intensity + LBP + LPQ							
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Body Double	SVM MLP	91.40%	15.10%	2.30%	6.10%	23.10%	7.80%	2.60%	
		99.90%	0.00%	0.10%	0.00%	0.00%	0.00%	0.00%	
Ecoflex	SVM MLP	74.50%	0.00%	76.50%	4.20%	12.90%	3.60%	2.40%	
		99.70%	0.30%	0.30%	0.00%	0.30%	0.00%	0.00%	
Playdoh	SVM MLP	67.80%	11.10%	95.40%	49.20%	93.50%	85.60%	84.00%	
		99.30%	0.40%	1.00%	0.60%	1.30%	0.00%	0.00%	

Table A.10: SVM vs MLP complete results - Unseen-Attack framework - Time Series.

SVM vs MLP - One attack - Time Series									
Attack	Classifier	Without segmentation Features = Intensity + LBP + LPQ							
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Body Double	SVM	81.20%	19.50%	18.10%	18.50%	55.70%	38.80%	27.10%	
	MLP	86.20%	0.30%	27.30%	4.20%	10.70%	3.00%	1.10%	
Ecoflex	SVM	73.50%	37.80%	3.20%	17.30%	74.30%	46.00%	31.40%	
	MLP	99.60%	0.10%	0.70%	0.00%	0.00%	0.00%	0.00%	
Playdoh	SVM	77.10%	28.50%	5.60%	16.80%	67.30%	43.40%	27.90%	
	MLP	98.90%	0.10%	2.10%	0.10%	0.10%	0.00%	0.00%	
Attack	Classifier	With segmentation Feature = Intensity + LBP + LPQ							
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Body Double	SVM	81.30%	33.90%	3.40%	10.00%	42.80%	15.40%	10.00%	
	MLP	82.40%	0.00%	35.30%	1.20%	1.90%	0.10%	0.00%	
Ecoflex	SVM	71.70%	0.30%	85.50%	15.80%	63.10%	37.30%	23.80%	
	MLP	98.40%	0.00%	3.10%	0.50%	1.00%	0.00%	0.00%	
Playdoh	SVM	31.00%	91.40%	0.50%	23.70%	92.40%	76.00%	53.40%	
	MLP	99.40%	0.00%	1.30%	0.10%	1.30%	0.00%	0.00%	

A.2 MLP Regularization (MLP_{reg}) Complete Results

Table A.11: MLP regularization complete results - Unseen-Attack framework - Cross Match.

MLP - Unseen Attack - Cross Match										
Without segmentation										
Attack	Regularization	Features = Intensity + LBP + LPQ								
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10		
Body Double	TASK TASK+ ADV TASK + TRANSFER	80.90%	0.00%	38.50%	6.30%	31.70%	8.50%	3.70%		
		81.20%	2.40%	35.40%	9.50%	48.80%	13.70%	8.40%		
		84.10%	1.60%	30.40%	9.20%	38.60%	12.70%	7.20%		
		83.20%	4.60%	29.10%	10.10%	38.90%	19.40%	12.00%		
Ecoflex	TASK TASK+ ADV TASK + TRANSFER	88.70%	1.60%	21.10%	5.60%	24.30%	8.60%	3.10%		
		88.00%	6.80%	17.30%	8.40%	26.90%	10.60%	8.10%		
		93.60%	2.20%	10.60%	4.80%	16.90%	5.00%	2.00%		
		92.90%	1.80%	12.40%	5.40%	14.50%	3.90%	1.70%		
Playdoh	TASK TASK+ ADV TASK + TRANSFER	78.30%	0.20%	44.10%	7.10%	43.20%	15.30%	5.90%		
		83.80%	0.80%	32.20%	8%	44%	15.30%	6.20%		
		74.50%	0.20%	51.80%	7.70%	52.80%	13.10%	5.40%		
		79.10%	1.20%	41.40%	12.00%	43.30%	22.90%	14.50%		
With segmentation										
Attack	Regularization	Features = Intensity + LBP + LPQ								
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10		
Body Double	TASK TASK+ ADV TASK + TRANSFER	91.20%	0.20%	17.40%	1.30%	1.70%	0.40%	0.00%		
		92.20%	0.00%	15.80%	1.10%	1.10%	0.20%	0.00%		
		95.10%	0.40%	9.50%	0.80%	1.20%	0.60%	0.00%		
		94.40%	0.40%	10.90%	2.00%	2.90%	0.20%	0.20%		
Ecoflex	TASK TASK+ ADV TASK + TRANSFER	98.80%	0.00%	2.40%	0.00%	0.00%	0.00%	0.00%		
		98.40%	0.00%	3.20%	0.40%	0.40%	0.00%	0.00%		
		99.00%	0.00%	2.00%	0.00%	0.00%	0.00%	0.00%		
		100.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%		
Playdoh	TASK TASK+ ADV TASK + TRANSFER	54.00%	0.00%	93.80%	11.70%	45.10%	22.30%	11.30%		
		52.10%	0.00%	97.70%	12.80%	43.90%	21.60%	11.70%		
		52.40%	0.00%	97.10%	13.10%	46.20%	20.80%	12.80%		
		53.10%	0.00%	95.60%	9.10%	30.00%	12.70%	6.00%		

Table A.12: MLP regularization complete results - Unseen-Attack framework - Digital Persona.

MLP - Unseen Attack - Digital Persona									
Without segmentation									
Attack	Regularization	Features = Intensity + LBP + LPQ							
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	TASK TASK+ ADV TASK +TRANSFER TASK + ADV + TRANSFER	98.80%	0.00%	2.40%	0.70%	4.40%	0.00%	0.00%	
		99.00%	0.00%	2.00%	0.00%	0.00%	0.00%	0.00%	
		99.00%	0.30%	2.00%	0.40%	1.50%	0.00%	0.00%	
		99.00%	0.00%	2.00%	0.00%	1.90%	0.00%	0.00%	
		96.80%	0.00%	6.40%	0.70%	0.70%	0.00%	0.00%	
Gelatine	TASK+ ADV TASK +TRANSFER TASK + ADV + TRANSFER	99.00%	0.00%	2.00%	0.00%	0.00%	0.00%	0.00%	
		97.00%	0.50%	6.00%	0.00%	0.00%	0.00%	0.00%	
		97.20%	0.00%	5.60%	0.00%	0.00%	0.00%	0.00%	
		94.40%	0.00%	11.20%	4.60%	49.20%	16.50%	1.20%	
Latex	TASK+ ADV TASK +TRANSFER TASK + ADV + TRANSFER	95.40%	0.00%	9.20%	2.70%	39.30%	6.70%	0.50%	
		96.00%	0.00%	8.00%	1.60%	1.60%	0.00%	0.00%	
		95.60%	0.00%	8.80%	1.20%	19.50%	7.40%	0.00%	
		99.20%	0.00%	1.60%	0.00%	0.00%	0.00%	0.00%	
Wood glue	TASK+ ADV TASK +TRANSFER TASK + ADV + TRANSFER	99.40%	0.00%	1.20%	0.40%	0.40%	0.00%	0.00%	
		99.40%	0.20%	1.20%	0.00%	0.00%	0.00%	0.00%	
		99.20%	0.00%	1.60%	0.00%	0.00%	0.00%	0.00%	
		99.20%	0.00%	1.60%	0.00%	0.00%	0.00%	0.00%	

Table A.13: MLP regularization complete results - Unseen-Attack framework - Digital Persona.

MLP - Unseen Attack - Digital Persona									
With segmentation									
Attack	Regularization	Accuracy	APCER	BPCER	EER	Features = Intensity + LBP + LPQ			
Ecoflex	TASK	97.20%	0.00%	5.60%	0.00%	2.60%	0.00%	0.00%	0.00%
	TASK + ADV	98.80%	0.00%	2.40%	0.40%	0.40%	0.00%	0.00%	0.00%
	TASK + TRANSFER	98.20%	0.00%	3.60%	0.40%	10.10%	0.00%	0.00%	0.00%
	TASK + ADV + TRANSFER	99.80%	0.00%	0.40%	0.00%	0.00%	0.00%	0.00%	0.00%
	TASK	99.20%	0.00%	1.60%	0.00%	0.00%	0.00%	0.00%	0.00%
Gelatine	TASK + ADV	100.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	TASK + TRANSFER	99.20%	0.00%	1.60%	0.30%	0.30%	0.00%	0.00%	0.00%
	TASK + ADV + TRANSFER	99.80%	0.00%	0.40%	0.00%	0.00%	0.00%	0.00%	0.00%
	TASK	95.40%	0.00%	9.20%	1.40%	16.20%	2.10%	0.00%	0.00%
Latex	TASK + ADV	94.60%	0.00%	10.80%	1.60%	15.60%	0.40%	0.00%	0.00%
	TASK + TRANSFER	95.60%	0.00%	8.80%	1.60%	17.10%	4.80%	0.00%	0.00%
	TASK + ADV + TRANSFER	96.00%	0.00%	8.00%	1.20%	7.00%	0.00%	0.00%	0.00%
	TASK	98.80%	0.00%	2.40%	0.40%	4.80%	0.00%	0.00%	0.00%
Wood glue	TASK + ADV	99.80%	0.00%	0.40%	0.00%	0.00%	0.00%	0.00%	0.00%
	TASK + TRANSFER	98.80%	1.20%	1.20%	0.40%	5.60%	0.00%	0.00%	0.00%
	TASK + ADV + TRANSFER	99.80%	0.00%	0.40%	0.00%	0.00%	0.00%	0.00%	0.00%
	TASK	99.80%	0.00%	0.40%	0.00%	0.00%	0.00%	0.00%	0.00%

Table A.14: MLP regularization complete results - Unseen-Attack framework - Green Bit.

MLP - Unseen Attack - Green Bit								
Without segmentation								
Attack	Regularization	Features = Intensity + LBP + LPQ						
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10
Ecoflex	TASK	97.40%	0.00%	5.20%	0.80%	2.80%	0.00%	0.00%
	TASK+ ADV	95.20%	0.00%	9.60%	0.00%	8.20%	0.00%	0.00%
	TASK + TRANSFER	98.40%	0.00%	3.20%	0.40%	1.40%	0.00%	0.00%
	TASK + ADV + TRANSFER	97.80%	0.00%	4.40%	0.00%	1.60%	0.00%	0.00%
	TASK	59.00%	0.00%	81.60%	30.40%	97.50%	82.20%	70.60%
Gelatine	TASK+ ADV	58.80%	0.00%	82.00%	25.60%	90.10%	71.70%	58.20%
	TASK + TRANSFER	58.40%	0.00%	82.80%	23.60%	85.10%	74.50%	50.50%
	TASK + ADV + TRANSFER	64.60%	0.00%	70.40%	19.00%	88.40%	42.30%	32.40%
Latex	TASK	94.80%	0.00%	10.40%	0.90%	0.90%	0.00%	0.00%
	TASK+ ADV	96.80%	0.00%	6.40%	0.30%	0.30%	0.00%	0.00%
	TASK + TRANSFER	97.00%	0.00%	6.00%	0.90%	0.90%	0.00%	0.00%
	TASK + ADV + TRANSFER	87.70%	0.00%	24.40%	0.00%	1.30%	0.00%	0.00%
	TASK	92.40%	0.00%	15.20%	3.80%	37.00%	3.00%	1.20%
Wood glue	TASK+ ADV	91.50%	0.00%	16.80%	3.20%	16.90%	2.00%	0.40%
	TASK + TRANSFER	92.20%	0.00%	15.60%	4.70%	55.10%	9.10%	2.30%
	TASK + ADV + TRANSFER	93.40%	0.00%	13.20%	2.00%	14.20%	0.70%	0.00%

Table A.15: MLP regularization complete results - Unseen-Attack framework - Green Bit.

MLP - Unseen Attack - Green Bit									
With segmentation									
Attack	Regularization	Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	TASK	99.60%	0.80%	0.00%	0.50%	0.50%	0.00%	0.00%	
	TASK+ ADV	99.60%	0.40%	0.40%	0.00%	0.00%	0.00%	0.00%	
	TASK+TRANSFER	99.20%	1.20%	0.40%	0.50%	0.50%	0.00%	0.00%	
	TASK + ADV + TRANSFER	99.40%	0.80%	0.40%	0.40%	0.40%	0.00%	0.00%	
Gelatine	TASK	95.20%	0.00%	9.60%	0.80%	0.80%	0.00%	0.00%	
	TASK+ ADV	94.40%	0.40%	10.80%	1.50%	1.50%	0.40%	0.40%	
	TASK+TRANSFER	96.80%	1.20%	5.20%	1.40%	4.10%	1.10%	0.00%	
	TASK + ADV + TRANSFER	95.80%	0.80%	7.60%	1.90%	3.80%	1.50%	0.40%	
Latex	TASK	98.80%	0.40%	2.00%	1.60%	1.60%	0.00%	0.00%	
	TASK+ ADV	98.60%	0.80%	2.00%	0.80%	2.80%	0.00%	0.00%	
	TASK+TRANSFER	99.00%	0.40%	1.60%	0.40%	0.40%	0.40%	0.00%	
	TASK + ADV + TRANSFER	99.20%	0.80%	0.80%	0.40%	0.40%	0.00%	0.00%	
Wood glue	TASK	97.60%	1.60%	3.20%	1.50%	5.40%	0.80%	0.50%	
	TASK+ ADV	96.80%	0.80%	5.60%	1.40%	2.80%	0.30%	0.00%	
	TASK+TRANSFER	98.00%	0.80%	3.20%	1.40%	5.80%	0.40%	0.00%	
	TASK + ADV + TRANSFER	97.20%	0.40%	5.20%	1.00%	2.10%	1.00%	0.50%	

Table A.16: MLP regularization complete results - Unseen-Attack framework - Hi Scan.

MLP - Unseen Attack - Hi Scan									
Without segmentation									
Attack	Regularization	Features = Intensity + LBP + LPQ							
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	TASK	99.00%	0.00%	2.00%	0.00%	0.00%	0.00%	0.00%	
	TASK+ ADV	100.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	
	TASK+TRANSFER	100.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	
	TASK + ADV + TRANSFER	100.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	
	TASK	95.80%	0.00%	8.40%	1.90%	20.50%	1.90%	0.00%	
Gelatine	TASK+ ADV	96.80%	0.00%	6.40%	0.80%	4.80%	0.00%	0.00%	
	TASK+TRANSFER	97.60%	0.00%	4.80%	2.60%	3.60%	1.70%	0.00%	
	TASK + ADV + TRANSFER	97.20%	0.00%	5.60%	0.00%	0.00%	0.00%	0.00%	
	TASK	86.60%	0.00%	26.80%	0.40%	0.40%	0.00%	0.00%	
Latex	TASK+ ADV	85.00%	0.00%	30.00%	0.00%	0.00%	0.00%	0.00%	
	TASK+TRANSFER	89.00%	0.00%	22.00%	0.00%	0.00%	0.00%	0.00%	
	TASK + ADV + TRANSFER	83.40%	0.00%	33.20%	0.00%	0.00%	0.00%	0.00%	
	TASK	95.80%	0.00%	8.40%	0.00%	25.50%	0.00%	0.00%	
Wood glue	TASK+ ADV	92.80%	0.00%	14.40%	0.70%	3.00%	0.00%	0.00%	
	TASK+TRANSFER	96.80%	0.00%	6.40%	0.00%	2.90%	0.00%	0.00%	
	TASK + ADV + TRANSFER	95.40%	0.00%	9.20%	0.90%	3.20%	0.00%	0.00%	

Table A.17: MLP regularization complete results - Unseen-Attack framework - Hi Scan.

MLP - Unseen Attack - Hi Scan									
With segmentation									
Attack	Regularization	Accuracy	APCER	Features = Intensity + LBP + LPQ					
			BPCER	EER	BPCER@1	BPCER@5	BPCER@10		
Ecoflex	TASK	96.40%	1.20%	6.00%	2.90%	10.90%	1.70%	0.40%	
	TASK + ADV	99.00%	0.40%	1.60%	1.10%	1.10%	0.00%	0.00%	
	TASK + TRANSFER	96.60%	1.20%	5.60%	4.00%	7.40%	0.00%	0.00%	
	TASK + ADV + TRANSFER	96.20%	1.20%	6.40%	2.70%	5.70%	1.60%	0.30%	
	TASK	88.80%	0.00%	22.40%	6.20%	34.30%	8.80%	4.60%	
Gelatine	TASK + ADV	87.80%	0.80%	23.60%	5.20%	21.30%	12.60%	3.10%	
	TASK + TRANSFER	92.00%	2.40%	13.60%	8.90%	29.50%	14.50%	4.40%	
	TASK + ADV + TRANSFER	91.80%	2.80%	13.60%	6.70%	24.70%	9.50%	4.10%	
	TASK	95.00%	0.00%	10.00%	1.50%	8.90%	0.80%	0.00%	
Latex	TASK + ADV	92.40%	0.40%	14.80%	7.20%	18.60%	11.60%	3.60%	
	TASK + TRANSFER	92.20%	1.60%	14.00%	9.50%	42.30%	17.10%	7.40%	
	TASK + ADV + TRANSFER	93.80%	1.20%	11.20%	4.10%	5.70%	3.30%	2.00%	
	TASK	99.00%	0.00%	2.00%	1.50%	1.50%	0.00%	0.00%	
Wood glue	TASK + ADV	99.20%	0.80%	0.80%	1.70%	1.70%	0.00%	0.00%	
	TASK + TRANSFER	98.40%	0.80%	2.40%	0.70%	2.70%	0.00%	0.00%	
	TASK + ADV + TRANSFER	99.80%	0.00%	0.40%	0.00%	0.00%	0.00%	0.00%	
	TASK								

Table A.18: MLP regularization complete results - Unseen-Attack framework - Time Series.

MLP - Unseen Attack - Time Series											
Without segmentation											
Attack	Regularization	Features = Intensity + LBP + LPQ									
Body Double	TASK TASK+ ADV TASK + TRANSFER TASK + ADV + TRANSFER	Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10			
		86.20%	0.30%	27.30%	4.20%	10.70%	3.00%	1.10%			
		80.20%	1.40%	38.20%	12.70%	59.90%	31.80%	19.30%			
		87.10%	1.10%	24.60%	6.70%	20.50%	6.30%	3.70%			
		89.80%	0.20%	20.10%	2.10%	9.40%	1.10%	0.50%			
Ecoflex	TASK TASK+ ADV TASK + TRANSFER TASK + ADV + TRANSFER	99.60%	0.10%	0.70%	0.00%	0.00%	0.00%	0.00%			
		99.00%	0.20%	2.00%	0.50%	nan	nan	nan			
		99.60%	0.40%	0.30%	0.10%	0.20%	0.10%	0.10%			
		99.00%	0.50%	1.40%	0.30%	nan	nan	nan			
		98.90%	0.10%	2.10%	0.10%	0.10%	0.00%	0.00%			
Playdoh	TASK TASK+ ADV TASK + TRANSFER TASK + ADV + TRANSFER	98.10%	0.30%	3.40%	1%	3%	0.10%	0.10%			
		98.30%	1.40%	2.10%	1.40%	9.80%	0.50%	0.30%			
		98.40%	0.70%	2.10%	0.90%	9.70%	0.30%	0.10%			
		With segmentation									
Attack	Regularization	Features = Intensity + LBP + LPQ									
Body Double	TASK TASK+ ADV TASK + TRANSFER TASK + ADV + TRANSFER	Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10			
		82.40%	0.00%	35.30%	1.20%	1.90%	0.10%	0.00%			
		79.50%	0.00%	41.00%	0.80%	2.30%	0.10%	0.00%			
		76.50%	0.00%	47.10%	3.30%	10.30%	1.60%	0.70%			
		86.80%	0.00%	26.50%	0.20%	0.50%	0.10%	0.00%			
Ecoflex	TASK TASK+ ADV TASK + TRANSFER TASK + ADV + TRANSFER	98.40%	0.00%	3.10%	0.50%	1.00%	0.00%	0.00%			
		99.80%	0.00%	0.50%	0.10%	0.10%	0.00%	0.00%			
		98.80%	0.00%	2.30%	0.20%	0.20%	0.00%	0.00%			
		99.80%	0.00%	0.50%	0.10%	0.10%	0.00%	0.00%			
		99.40%	0.00%	1.30%	0.10%	1.30%	0.00%	0.00%			
Playdoh	TASK+ ADV TASK + TRANSFER TASK + ADV + TRANSFER	99.50%	0.00%	1.00%	0.00%	1.00%	0.00%	0.00%			
		99.50%	0.00%	1.10%	0.30%	1.80%	0.00%	0.00%			
		99.90%	0.00%	0.30%	0.00%	0.30%	0.00%	0.00%			

A.3 First CNN (CNN1) Complete Results

Table A.19: CNN1 complete results - One-Attack framework - Cross Match.

C - One Attack - CrossMatch								
Attack	Method	Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10
Body Double	Entire image ROI (padding) ROI (cropping) Patches (method 1) Patches (method 2)	84.94%	15.00%	14.72%	9.00%	34.70%	22.20%	10.80%
		88.58%	9.80%	12.58%	8.54%	22.00%	12.20%	6.00%
		93.14%	4.26%	9.66%	3.68%	11.10%	3.80%	0.90%
		72.74%	27.42%	26.94%	26.04%	64.40%	41.00%	31.90%
Ecoflex	Entire image ROI (padding) ROI (cropping) Patches (method 1) Patches (method 2)	80.46%	27.16%	11.96%	18.32%	40.20%	33.60%	28.00%
		85.76%	7.80%	21.34%	7.55%	25.70%	12.50%	6.50%
		91.10%	9.02%	8.72%	5.52%	22.00%	12.20%	6.00%
		93.30%	6.08%	7.30%	3.44%	12.20%	3.80%	1.70%
Playdoh	Entire image ROI (padding) ROI (cropping) Patches (method 1) Patches (method 2)	82.60%	21.20%	13.28%	16.42%	47.10%	16.30%	10.90%
		83.20%	20.70%	12.86%	16.98%	46.00%	24.70%	6.60%
		98.72%	0.78%	1.82%	0.00%	19.80%	0.30%	0.00%
		99.48%	1.16%	0.00%	0.48%	0.70%	0.10%	0.00%
Playdoh	Entire image ROI (padding) ROI (cropping) Patches (method 1) Patches (method 2)	99.80%	0.00%	0.40%	0.00%	5.10%	0.40%	0.00%
		93.90%	6.30%	5.56%	3.58%	14.20%	3.10%	0.80%
		95.38%	5.94%	3.26%	4.10%	11.20%	1.40%	0.70%

Table A.20: CNN1 complete results - Unseen-Attack framework - Cross Match.

C - Unseen Attack - CrossMatch									
Attack	Method	Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Body Double	Entire image	82.30%	1.40%	34.00%	8.50%	43.20%	21.80%	14.60%	
	ROI (padding)	68.20%	1.60%	61.90%	10.90%	62.90%	44.90%	23.40%	
	ROI (cropping)	67.60%	2.20%	62.60%	15.10%	49.30%	37.90%	26.60%	
	Patches (method 1)	63.80%	10.50%	61.90%	25.00%	65.60%	50.50%	31.70%	
	Patches (method 2)	57.60%	13.80%	71.10%	35.80%	79.50%	52.50%	34.30%	
Ecoflex	Entire image	87.80%	1.80%	22.70%	6.60%	27.20%	15.10%	7.20%	
	ROI (padding)	60.00%	0.20%	79.70%	26.60%	61.90%	36.60%	26.00%	
	ROI (cropping)	89.20%	6.00%	15.70%	6.20%	61.50%	24.70%	14.00%	
	Patches (method 1)	82.40%	7.00%	29.10%	12.50%	82.30%	58.20%	42.30%	
	Patches (method 2)	80.90%	16.70%	21.50%	18.90%	75.80%	59.80%	33.70%	
Playdoh	Entire image	98.90%	2.30%	0.00%	0.70%	30.50%	20.80%	5.40%	
	ROI (padding)	97.60%	4.60%	0.20%	0.90%	10.50%	7.20%	0.50%	
	ROI (cropping)	95.70%	8.50%	0.00%	0.70%	2.50%	0.80%	0.40%	
	Patches (method 1)	85.30%	8.10%	21.20%	10.00%	25.10%	6.10%	5.00%	
	Patches (method 2)	86.40%	26.30%	0.80%	6.50%	28.20%	8.80%	4.30%	

A.4 VGG Complete Results

Table A.21: VGG11 complete results - Cross Match.

VGG11 - One Attack - Cross Match								
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10
Body Double	VGG crop	95.83%	6.30%	2.37%	3.40%	6.40%	2.17%	1.27%
	VGG resize	90.30%	7.43%	11.80%	9.27%	23.50%	13.40%	8.57%
Ecoflex	VGG crop	94.83%	6.87%	3.43%	4.67%	9.20%	4.47%	2.93%
	VGG resize	89.83%	9.97%	10.43%	8.83%	26.00%	8.97%	6.03%
Playdoh	VGG crop	96.97%	4.50%	1.57%	2.83%	4.25%	3.05%	2.70%
	VGG resize	96.30%	5.93%	1.30%	3.30%	8.37%	7.63%	7.07%
VGG11 - Unseen Attack - Cross Match								
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10
Body Double	VGG crop	81.90%	3.40%	33.00%	12.60%	29.00%	18.70%	13.90%
	VGG resize	77.30%	14.20%	31.40%	21.90%	54.40%	38.30%	32.00%
Ecoflex	VGG crop	90.80%	2.60%	15.90%	5.40%	13.70%	5.70%	2.50%
	VGG resize	88.50%	16.80%	6.20%	10.90%	21.50%	11.20%	9.50%
Playdoh	VGG crop	89.20%	3.40%	18.50%	6.30%	17.20%	6.30%	3.60%
	VGG resize	77.90%	11.00%	33.70%	19.10%	31.60%	27.20%	22.70%

Table A.22: VGG11 complete results - Digital Persona.

VGG11 - One Attack - Digital Persona									
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	VGG crop	88.00%	12.37%	11.90%	8.10%	18.77%	11.23%	5.13%	
	VGG resize	78.00%	22.30%	22.30%	15.53%	41.83%	26.37%	20.50%	
	VGG crop	86.33%	17.17%	10.37%	10.03%	21.33%	13.50%	9.47%	
Gelatine	VGG resize	72.67%	23.53%	29.70%	21.27%	43.70%	33.07%	31.00%	
	VGG crop	88.33%	15.03%	7.67%	9.77%	12.70%	11.17%	5.77%	
	VGG resize	82.67%	15.30%	18.77%	12.70%	40.90%	25.50%	23.80%	
Latex	VGG crop	88.67%	18.13%	5.37%	7.73%	21.40%	12.85%	9.40%	
	VGG resize	75.67%	22.47%	25.23%	19.57%	52.07%	38.27%	33.83%	
VGG11 - Unseen Attack - Digital Persona									
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	VGG crop	93.20%	8.00%	5.60%	5.90%	10.60%	6.80%	2.70%	
	VGG resize	84.40%	20.80%	10.40%	13.81%	30.19%	18.77%	14.68%	
	VGG crop	68.80%	3.60%	58.80%	25.40%	57.20%	42.20%	39.70%	
Gelatine	VGG resize	65.60%	18.80%	50.00%	33.00%	67.89%	53.97%	44.63%	
	VGG crop	88.20%	4.40%	19.20%	8.50%	34.00%	13.60%	9.50%	
	VGG resize	83.00%	23.20%	10.80%	19.03%	34.18%	22.80%	20.06%	
Latex	VGG crop	95.40%	6.00%	3.20%	5.20%	10.20%	5.20%	3.30%	
	VGG resize	82.00%	26.40%	9.60%	17.27%	34.45%	23.79%	17.86%	

Table A.23: VGG11 complete results - Green Bit.

VGG11 - One Attack - Green Bit									
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	VGG crop	96.67%	5.67%	1.23%	2.27%	6.43%	1.70%	0.53%	
	VGG resize	97.00%	3.60%	2.67%	1.33%	4.03%	0.00%	0.00%	
Gelatine	VGG crop	93.33%	7.97%	5.30%	6.37%	6.37%	4.60%	4.60%	
	VGG resize	91.00%	6.57%	11.13%	4.10%	11.37%	4.10%	3.00%	
Latex	VGG crop	95.33%	2.70%	6.67%	1.90%	7.60%	0.87%	0.87%	
	VGG resize	93.33%	9.77%	3.80%	7.07%	14.70%	7.87%	2.13%	
Wood glue	VGG crop	99.00%	0.63%	1.27%	0.43%	0.43%	0.43%	0.00%	
	VGG resize	96.67%	3.30%	3.70%	2.27%	1.87%	1.47%	0.83%	
VGG11 - Unseen Attack - Green Bit									
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	VGG crop	96.40%	4.50%	2.80%	1.50%	5.70%	0.50%	0.50%	
	VGG resize	97.60%	3.20%	1.60%	1.50%	1.50%	0.70%	0.30%	
Gelatine	VGG crop	76.70%	0.80%	45.60%	11.90%	58.60%	35.00%	14.10%	
	VGG resize	80.10%	3.60%	36.00%	10.90%	25.50%	16.50%	12.60%	
Latex	VGG crop	99.20%	0.80%	0.80%	0.80%	0.80%	0.30%	0.30%	
	VGG resize	94.80%	6.10%	4.40%	4.60%	4.90%	3.10%	1.60%	
Wood glue	VGG crop	99.00%	2.00%	0.00%	0.30%	0.30%	0.00%	0.00%	
	VGG resize	96.40%	5.30%	2.00%	2.80%	4.10%	1.50%	0.80%	

Table A.24: VGG11 complete results - Hi Scan.

VGG11 - One Attack - Hi Scan									
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	VGG crop	93.00%	8.67%	5.07%	2.87%	7.67%	7.67%	2.63%	
	VGG resize	91.33%	7.33%	10.37%	5.20%	21.50%	14.37%	4.67%	
	VGG crop	96.00%	5.77%	2.67%	1.87%	4.27%	4.27%	0.40%	
Gelatine	VGG resize	93.33%	2.00%	11.40%	1.70%	3.40%	1.60%	1.60%	
	VGG crop	91.00%	12.07%	6.43%	4.73%	15.57%	13.00%	7.73%	
Latex	VGG resize	90.67%	11.33%	7.33%	8.83%	10.75%	10.75%	4.35%	
	VGG crop	96.00%	7.67%	0.73%	1.47%	1.47%	1.47%	1.47%	
Wood glue	VGG resize	94.67%	5.30%	5.63%	3.80%	3.53%	3.53%	1.20%	
VGG11 - Unseen Attack - Hi Scan									
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	VGG crop	95.60%	2.00%	6.80%	1.60%	2.20%	1.60%	0.50%	
	VGG resize	94.40%	4.80%	6.40%	3.47%	8.24%	4.86%	1.71%	
	VGG crop	85.40%	6.40%	22.80%	8.70%	nan	nan	nan	
Gelatine	VGG resize	84.60%	1.20%	29.60%	9.55%	18.14%	15.96%	6.43%	
	VGG crop	94.00%	3.60%	8.40%	3.20%	6.90%	5.00%	1.20%	
Latex	VGG resize	91.20%	4.40%	13.20%	6.45%	8.90%	7.75%	2.87%	
	VGG crop	97.60%	2.00%	2.80%	0.40%	0.40%	0.40%	0.00%	
Wood glue	VGG resize	92.60%	6.00%	8.80%	4.92%	9.92%	5.54%	2.60%	

Table A.25: VGG11 complete results - Time Series.

VGG11 - One Attack - Time Series									
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Body Double	VGG crop	99.00%	0.53%	1.53%	0.57%	0.57%	0.43%	0.07%	
	VGG resize	80.43%	18.20%	21.17%	16.90%	46.23%	40.97%	22.90%	
Ecoflex	VGG crop	98.63%	0.90%	1.87%	0.67%	1.83%	1.67%	0.07%	
	VGG resize	86.17%	9.20%	18.00%	13.07%	31.10%	26.00%	12.40%	
Playdoh	VGG crop	98.93%	1.47%	0.70%	0.17%	0.17%	0.17%	0.10%	
	VGG resize	87.83%	8.20%	15.93%	10.13%	25.63%	22.77%	10.57%	
VGG11 - Unseen Attack - Time Series									
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Body Double	VGG crop	90.00%	0.30%	19.70%	4.00%	12.00%	9.80%	2.20%	
	VGG resize	68.81%	9.05%	53.35%	27.04%	70.12%	56.12%	44.49%	
Ecoflex	VGG crop	98.80%	0.50%	1.80%	0.50%	0.70%	0.40%	0.10%	
	VGG resize	86.70%	11.42%	15.14%	11.43%	44.21%	21.92%	15.42%	
Playdoh	VGG crop	96.70%	0.50%	6.10%	0.60%	1.60%	1.60%	0.20%	
	VGG resize	85.10%	13.72%	16.08%	14.40%	40.76%	22.72%	16.60%	

A.5 Second CNN (CNN2) Complete Results

Table A.26: CNN2 complete results - One-Attack framework - Cross Match.

CNN2 - One Attack - Cross Match								
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10
Body Double	CNN (3, resize)	85.90%	11.00%	16.87%	11.50%	32.00%	16.13%	11.70%
	CNN (5, resize)	90.80%	5.63%	13.00%	6.47%	22.67%	11.80%	7.27%
	CNN (7,5,3, resize)	90.30%	2.47%	16.77%	7.83%	19.43%	11.03%	7.43%
	CNN (3, crop)	94.00%	6.93%	5.20%	4.03%	11.63%	3.80%	1.10%
	CNN (5, crop)	94.33%	6.60%	4.70%	5.53%	7.23%	5.00%	4.70%
Ecoflex	CNN (7,5,3, crop)	92.13%	11.83%	3.43%	4.47%	10.20%	4.10%	3.23%
	CNN (3, resize)	90.50%	7.90%	11.00%	7.90%	17.80%	9.33%	5.80%
	CNN (5, resize)	91.50%	10.47%	6.60%	6.93%	20.83%	10.20%	3.03%
	CNN (7,5,3, resize)	89.50%	14.67%	6.00%	8.53%	19.70%	10.20%	8.13%
	CNN (3, crop)	95.33%	3.63%	5.70%	3.03%	5.30%	2.40%	2.13%
Playdoh	CNN (5, crop)	91.33%	9.20%	8.50%	7.23%	15.67%	9.47%	7.40%
	CNN (7,5,3, crop)	95.50%	6.47%	2.83%	3.93%	7.83%	2.83%	2.20%
	CNN (3, resize)	94.43%	6.17%	4.97%	5.13%	6.10%	4.50%	2.27%
	CNN (5, resize)	96.23%	6.43%	0.67%	3.47%	3.05%	2.50%	1.60%
	CNN (7,5,3, resize)	95.07%	8.67%	1.00%	4.77%	5.07%	4.50%	3.40%
Playdoh	CNN (3, crop)	97.63%	3.87%	0.77%	1.70%	2.03%	0.60%	0.33%
	CNN (5, crop)	97.47%	1.97%	3.00%	1.63%	1.63%	0.80%	0.50%
	CNN (7,5,3, crop)	98.00%	3.33%	0.70%	2.20%	3.10%	1.90%	1.63%

Table A.27: CNN2 complete results - Unseen-Attack framework - Cross Match.

CNN2 - Unseen Attack - Cross Match								
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10
Body Double	CNN (3, resize)	85.90%	11.00%	16.87%	11.50%	32.00%	16.13%	11.70%
	CNN (5, resize)	90.80%	5.63%	13.00%	6.47%	22.67%	11.80%	7.27%
	CNN (7,5,3, resize)	69.50%	12.00%	49.20%	26.30%	54.90%	41.00%	33.30%
	CNN (3, crop)	82.10%	7.00%	28.90%	13.00%	nan	nan	nan
	CNN (5, crop)	94.33%	6.60%	4.70%	5.53%	7.23%	5.00%	4.70%
Ecoflex	CNN (7,5,3, crop)	73.30%	4.60%	49.00%	17.50%	50.10%	29.60%	20.50%
	CNN (3, resize)	90.50%	7.90%	11.00%	7.90%	17.80%	9.33%	5.80%
	CNN (5, resize)	91.50%	10.47%	6.60%	6.93%	20.83%	10.20%	3.03%
	CNN (7,5,3, resize)	84.90%	7.60%	22.70%	13.60%	36.00%	20.40%	16.90%
	CNN (3, crop)	90.70%	2.80%	15.90%	6.70%	17.20%	6.50%	4.50%
Playdoh	CNN (5, crop)	91.33%	9.20%	8.50%	7.23%	15.67%	9.47%	7.40%
	CNN (7,5,3, crop)	92.80%	4.80%	9.60%	7.60%	18.00%	7.00%	5.40%
	CNN (3, resize)	94.43%	6.17%	4.97%	5.13%	6.10%	4.50%	2.27%
	CNN (5, resize)	96.23%	6.43%	0.67%	3.47%	3.05%	2.50%	1.60%
	CNN (7,5,3, resize)	82.20%	12.00%	23.90%	12.20%	14.90%	13.80%	11.90%
Playdoh	CNN (3, crop)	90.30%	5.20%	14.30%	6.40%	8.60%	6.00%	4.40%
	CNN (5, crop)	97.47%	1.97%	3.00%	1.63%	1.63%	0.80%	0.50%
	CNN (7,5,3, crop)	86.00%	8.20%	20.00%	13.30%	21.90%	16.40%	11.10%

Table A.28: CNN2 complete results - One-Attack framework - Digital Persona.

CNN2 - One Attack - Digital Persona									
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	CNN (3, resize)	87.67%	18.50%	5.93%	12.40%	20.83%	11.23%	9.43%	
	CNN (5, resize)	83.67%	11.50%	20.97%	9.70%	22.27%	15.27%	11.80%	
	CNN (7,5,3, resize)	87.00%	17.20%	9.30%	12.27%	24.23%	13.43%	11.77%	
	CNN (3, crop)	94.33%	6.07%	5.43%	2.77%	9.60%	4.87%	1.77%	
	CNN (5, crop)	89.00%	10.63%	10.70%	7.07%	10.05%	6.70%	4.35%	
Gelatine	CNN (7,5,3, crop)	94.33%	6.70%	4.60%	5.23%	2.45%	0.90%	0.00%	
	CNN (3, resize)	74.67%	38.77%	10.83%	23.87%	43.13%	34.93%	29.50%	
	CNN (5, resize)	76.67%	32.33%	15.27%	26.37%	42.47%	33.77%	32.57%	
	CNN (7,5,3, resize)	72.33%	29.23%	26.93%	29.73%	43.73%	36.90%	32.23%	
	CNN (3, crop)	88.00%	12.90%	10.43%	8.10%	12.03%	6.93%	4.77%	
Latex	CNN (5, crop)	91.67%	7.33%	9.20%	5.40%	12.13%	4.30%	3.17%	
	CNN (7,5,3, crop)	88.33%	16.37%	7.90%	11.10%	24.60%	10.65%	8.95%	
	CNN (3, resize)	92.67%	8.13%	6.60%	5.73%	13.50%	6.63%	4.30%	
	CNN (5, resize)	80.67%	14.67%	21.67%	11.27%	28.70%	18.63%	16.80%	
	CNN (7,5,3, resize)	80.67%	19.67%	17.90%	20.20%	40.85%	34.80%	31.35%	
Wood glue	CNN (3, crop)	91.67%	4.60%	11.40%	3.57%	8.43%	4.13%	2.30%	
	CNN (5, crop)	87.33%	13.80%	11.13%	13.87%	38.40%	21.10%	20.30%	
	CNN (7,5,3, crop)	91.67%	5.83%	11.13%	6.37%	14.83%	5.40%	2.90%	
	CNN (3, resize)	86.33%	19.87%	7.30%	16.87%	30.35%	19.00%	18.10%	
	CNN (5, resize)	75.33%	29.07%	20.23%	25.20%	48.37%	43.67%	39.27%	
Wood glue	CNN (7,5,3, resize)	78.33%	25.17%	16.50%	18.07%	40.15%	25.80%	21.00%	
	CNN (3, crop)	93.00%	7.70%	6.10%	2.87%	9.97%	2.90%	1.10%	
	CNN (5, crop)	93.33%	5.97%	7.47%	3.60%	5.10%	3.17%	3.17%	
	CNN (7,5,3, crop)	89.33%	11.67%	9.40%	4.87%	12.97%	5.73%	3.80%	

Table A.29: CNN2 complete results - Unseen-Attack framework - Digital Persona.

CNN2 - Unseen Attack - Digital Persona								
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10
Ecoflex	CNN (3, resize)	81.80%	32.00%	4.40%	19.40%	31.10%	24.70%	20.90%
	CNN (5, resize)	82.20%	30.00%	5.60%	20.20%	33.50%	26.80%	23.10%
	CNN (7,5,3, resize)	79.60%	18.40%	22.40%	18.90%	34.40%	27.30%	21.50%
	CNN (3, crop)	95.20%	5.60%	4.00%	3.70%	9.60%	3.40%	2.30%
	CNN (5, crop)	93.40%	8.40%	4.80%	7.20%	13.50%	7.30%	4.20%
	CNN (7,5,3, crop)	94.80%	6.40%	4.00%	4.30%	9.00%	3.90%	3.50%
Gelatine	CNN (3, resize)	62.60%	10.80%	64.00%	29.80%	67.10%	55.30%	45.60%
	CNN (5, resize)	59.80%	16.80%	63.60%	34.20%	80.80%		58.30%
	CNN (7,5,3, resize)	64.40%	30.40%	40.80%	35.00%	80.30%	72.70%	65.00%
	CNN (3, crop)	79.60%	4.00%	36.80%	14.10%	35.70%	18.50%	13.20%
	CNN (5, crop)	76.60%	3.60%	43.20%	13.30%	44.30%	20.00%	15.30%
	CNN (7,5,3, crop)	72.80%	2.40%	52.00%	21.50%	48.30%	28.10%	23.30%
Latex	CNN (3, resize)	80.60%	15.20%	23.60%	19.90%	41.30%	29.80%	27.40%
	CNN (5, resize)	77.60%	36.00%	8.80%	19.30%	42.80%	33.10%	26.10%
	CNN (7,5,3, resize)	58.80%	34.40%	48.00%	39.00%	62.50%	50.90%	48.50%
	CNN (3, crop)	89.00%	6.00%	16.00%	7.80%	29.30%	15.90%	11.30%
	CNN (5, crop)	86.80%	2.80%	23.60%	9.50%	31.30%	14.70%	9.60%
	CNN (7,5,3, crop)	88.00%	5.20%	18.80%	8.90%	24.80%	11.90%	6.20%
Wood glue	CNN (3, resize)	82.60%	22.00%	12.80%	16.00%	42.60%	30.50%	23.00%
	CNN (5, resize)	81.00%	24.80%	13.20%	18.10%	39.80%	28.90%	22.60%
	CNN (7,5,3, resize)	78.00%	27.20%	16.80%	20.40%	41.80%	32.70%	26.90%
	CNN (3, crop)	94.00%	6.80%	5.20%	3.50%	14.20%	5.40%	0.70%
	CNN (5, crop)	94.60%	5.60%	5.20%	4.30%	7.30%	4.30%	1.10%
	CNN (7,5,3, crop)	94.60%	4.80%	6.00%	2.90%	13.10%	2.50%	1.20%

Table A.30: CNN2 complete results - One-Attack framework - Green Bit.

CNN2 - One Attack - Green Bit								
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10
Ecoflex	CNN (3, resize)	95.33%	5.27%	4.60%	3.13%	5.70%	3.13%	2.63%
	CNN (5, resize)	92.67%	7.67%	7.37%	3.60%	6.20%	2.73%	0.47%
	CNN (7.5,3, resize)	90.67%	6.47%	12.07%	4.53%	15.23%	4.53%	2.93%
	CNN (3, crop)	95.33%	5.10%	4.13%	2.83%	7.97%	1.57%	0.53%
	CNN (5, crop)	94.67%	7.27%	3.50%	2.67%	5.77%	2.80%	0.00%
	CNN (7.5,3, crop)	94.00%	7.43%	3.70%	2.70%	14.00%	10.10%	9.63%
Gelatine	CNN (3, resize)	93.33%	6.50%	6.40%	5.07%	7.73%	4.07%	3.13%
	CNN (5, resize)	88.33%	12.70%	10.40%	5.20%	10.63%	5.73%	5.07%
	CNN (7.5,3, resize)	93.00%	7.07%	6.67%	2.63%	8.03%	2.63%	1.67%
	CNN (3, crop)	97.67%	3.03%	1.33%	1.83%	1.83%	0.53%	0.53%
	CNN (5, crop)	94.00%	6.87%	5.97%	3.90%	6.03%	3.90%	2.10%
	CNN (7.5,3, crop)	95.00%	1.97%	7.83%	1.63%	5.40%	1.00%	0.43%
Latex	CNN (3, resize)	95.33%	5.10%	3.93%	3.27%	10.80%	2.80%	1.45%
	CNN (5, resize)	93.00%	6.97%	6.47%	4.53%	14.00%	5.50%	3.57%
	CNN (7.5,3, resize)	97.33%	0.67%	4.43%	1.17%	2.43%	0.00%	0.00%
	CNN (3, crop)	96.33%	3.87%	3.30%	3.63%	4.10%	2.03%	2.03%
	CNN (5, crop)	95.00%	6.87%	3.13%	3.20%	4.40%	3.40%	2.45%
	CNN (7.5,3, crop)	96.67%	4.97%	1.90%	1.17%	1.17%	0.63%	0.63%
Wood glue	CNN (3, resize)	95.33%	5.10%	3.93%	3.27%	10.80%	2.80%	1.45%
	CNN (5, resize)	92.67%	7.67%	7.37%	3.60%	6.20%	2.73%	0.47%
	CNN (7.5,3, resize)	94.67%	7.20%	3.60%	4.60%	5.25%	2.50%	0.85%
	CNN (3, crop)	97.00%	1.80%	3.93%	1.63%	5.05%	4.40%	0.85%
	CNN (5, crop)	96.67%	2.57%	4.07%	2.33%	2.03%	0.57%	0.00%
	CNN (7.5,3, crop)	96.67%	2.80%	3.80%	0.70%	1.40%	0.70%	0.00%

Table A.31: CNN2 complete results - Unseen-Attack framework - Green Bit.

CNN2 - Unseen Attack - Green Bit								
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10
Ecoflex	CNN (3, resize)	97.40%	4.00%	1.20%	1.50%	4.10%	0.80%	0.40%
	CNN (5, resize)	95.60%	4.90%	4.00%	2.80%	7.60%	3.60%	1.80%
	CNN (7,5,3, resize)	94.60%	5.30%	5.60%	5.30%	9.10%	3.40%	0.70%
	CNN (3, crop)	97.40%	1.60%	3.60%	2.30%	3.40%	0.40%	0.00%
	CNN (5, crop)	96.40%	2.80%	4.40%	2.10%	7.20%	1.10%	1.10%
Gelatine	CNN (7,5,3, crop)	96.40%	4.00%	3.20%	0.70%	7.50%	0.30%	0.30%
	CNN (3, resize)	75.50%	2.40%	46.40%	15.80%	45.50%	29.30%	23.80%
	CNN (5, resize)	82.50%	1.60%	33.20%	11.10%	29.10%	15.00%	9.10%
	CNN (7,5,3, resize)	80.50%	3.20%	35.60%	16.00%	42.40%	32.00%	20.00%
	CNN (3, crop)	65.40%	1.60%	67.20%	21.80%	74.50%	54.00%	40.20%
Latex	CNN (5, crop)	72.40%	3.20%	51.60%	17.50%	46.20%	35.60%	23.00%
	CNN (7,5,3, crop)	65.60%	0.80%	67.60%	22.10%	62.80%	47.70%	38.40%
	CNN (3, resize)	96.80%	4.00%	2.40%	1.40%	3.40%	1.00%	0.40%
	CNN (5, resize)	95.00%	4.00%	6.00%	5.30%	8.90%	3.60%	1.60%
	CNN (7,5,3, resize)	92.80%	4.00%	10.40%	4.80%	10.60%	2.90%	2.00%
Wood glue	CNN (3, crop)	96.80%	4.00%	2.40%	1.20%	2.40%	0.40%	0.40%
	CNN (5, crop)	96.80%	4.00%	2.40%	1.90%	5.40%	1.40%	0.90%
	CNN (7,5,3, crop)	97.20%	2.80%	2.80%	2.70%	5.40%	3.80%	1.20%
	CNN (3, resize)	96.20%	7.70%	0.00%	1.80%	2.30%	0.70%	0.40%
	CNN (5, resize)	97.40%	3.20%	2.00%	0.90%	3.70%	0.60%	0.60%
	CNN (7,5,3, resize)	96.40%	6.10%	1.20%	1.20%	3.50%	0.40%	0.00%
	CNN (3, crop)	95.60%	4.90%	4.00%	2.90%	4.70%	1.70%	0.70%
	CNN (5, crop)	96.60%	6.90%	0.00%	1.30%	1.30%	0.40%	0.00%
	CNN (7,5,3, crop)	97.20%	2.80%	2.80%	2.70%	5.40%	3.80%	1.20%

Table A.32: CNN2 complete results - One-Attack framework - Hi Scan.

CNN2 - One Attack - Hi Scan									
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	CNN (3, resize)	95.33%	3.60%	5.87%	1.67%	7.33%	2.30%	0.60%	
	CNN (5, resize)	86.67%	13.10%	12.13%	13.27%	28.47%	14.67%	12.03%	
	CNN (7,5,3, resize)	90.00%	10.77%	9.23%	7.30%	16.10%	13.40%	5.57%	
	CNN (3, crop)	91.67%	6.87%	9.93%	5.47%	18.67%	16.67%	4.70%	
	CNN (5, crop)	92.00%	10.23%	5.87%	5.13%	8.43%	5.13%	3.57%	
	CNN (7,5,3, crop)	94.33%	6.43%	4.63%	3.03%	6.30%	6.30%	4.15%	
Gelatine	CNN (3, resize)	90.33%	14.77%	5.10%	8.33%	12.77%	8.47%	7.20%	
	CNN (5, resize)	89.67%	10.23%	10.40%	8.30%	10.40%	8.55%	6.30%	
	CNN (7,5,3, resize)	90.67%	11.07%	6.57%	8.50%	12.17%	12.17%	6.37%	
	CNN (3, crop)	90.67%	10.57%	8.27%	9.67%	5.40%	1.80%	1.80%	
	CNN (5, crop)	89.67%	14.70%	7.20%	11.07%	15.17%	12.37%	8.10%	
	CNN (7,5,3, crop)	91.00%	11.50%	7.10%	4.00%	7.20%	7.20%	3.33%	
Latex	CNN (3, resize)	89.00%	13.47%	8.73%	7.60%	23.47%	16.43%	15.47%	
	CNN (5, resize)	87.67%	16.97%	7.30%	13.57%	27.30%	15.20%	13.65%	
	CNN (7,5,3, resize)	81.00%	19.10%	18.70%	16.30%	23.03%	22.23%	17.47%	
	CNN (3, crop)	90.67%	6.03%	12.43%	7.47%	11.80%	11.80%	5.97%	
	CNN (5, crop)	93.00%	8.47%	5.70%	6.17%	11.03%	10.27%	3.53%	
	CNN (7,5,3, crop)	89.67%	15.90%	4.33%	7.70%	12.43%	9.57%	5.20%	
Wood glue	CNN (3, resize)	92.33%	8.20%	6.90%	4.77%	20.65%	11.10%	3.65%	
	CNN (5, resize)	91.67%	10.37%	6.27%	5.93%	13.67%	4.37%	2.87%	
	CNN (7,5,3, resize)	89.33%	13.30%	7.80%	10.40%	18.57%	15.43%	6.27%	
	CNN (3, crop)	95.00%	7.60%	2.17%	5.70%	5.10%	5.10%	1.17%	
	CNN (5, crop)	94.67%	7.23%	3.40%	2.73%	7.10%	4.70%	1.07%	
	CNN (7,5,3, crop)	96.33%	5.23%	2.23%	1.20%	1.20%	0.40%	0.40%	

Table A.33: CNN2 complete results - Unseen-Attack framework - Hi Scan.

CNN2 - Unseen Attack - Hi Scan								
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10
Ecoflex	CNN (3, resize)	94.60%	6.00%	4.80%	5.10%	12.20%	4.70%	2.50%
	CNN (5, resize)	92.00%	7.20%	8.80%	8.30%	26.40%	8.30%	4.20%
	CNN (7,5,3, resize)	92.60%	8.40%	6.40%	4.10%	13.00%	5.60%	2.40%
	CNN (3, crop)	92.20%	10.00%	5.60%	5.30%	11.90%	8.10%	5.00%
	CNN (5, crop)	92.20%	4.00%	11.60%	4.00%	7.70%	7.70%	5.50%
Gelatine	CNN (7,5,3, crop)	92.00%	8.80%	7.20%	4.80%	11.00%	11.00%	4.10%
	CNN (3, resize)	72.40%	1.60%	53.60%	20.50%	32.80%	23.50%	21.00%
	CNN (5, resize)	81.20%	5.60%	32.00%	12.50%	24.30%	14.20%	13.70%
	CNN (7,5,3, resize)	89.00%	10.00%	12.00%	10.80%	18.40%	16.50%	9.10%
	CNN (3, crop)	64.20%	3.60%	68.00%	35.30%	63.80%	62.80%	49.40%
Latex	CNN (5, crop)	78.40%	4.00%	39.20%	15.90%	24.80%	21.60%	13.30%
	CNN (7,5,3, crop)	75.00%	6.00%	44.00%	16.70%	nan	nan	nan
	CNN (3, resize)	89.80%	10.00%	10.40%	9.20%	23.90%	10.90%	8.60%
	CNN (5, resize)	90.00%	8.40%	11.60%	10.00%	22.00%	12.10%	7.70%
	CNN (7,5,3, resize)	86.20%	5.20%	22.40%	7.60%	16.40%	12.80%	5.60%
Wood glue	CNN (3, crop)	95.80%	4.40%	4.00%	2.70%	6.80%	4.70%	0.70%
	CNN (5, crop)	92.20%	4.80%	10.80%	5.00%	10.20%	9.20%	4.20%
	CNN (7,5,3, crop)	88.00%	11.20%	12.80%	9.20%	16.30%	13.70%	6.10%
	CNN (3, resize)	95.80%	4.80%	3.60%	4.70%	8.50%	3.40%	0.90%
	CNN (5, resize)	93.20%	5.20%	8.40%	7.20%	14.80%	6.00%	2.00%
Wood glue	CNN (7,5,3, resize)	92.80%	6.00%	8.40%	4.90%	11.50%	6.70%	3.00%
	CNN (3, crop)	92.80%	4.40%	10.00%	1.70%	6.40%	5.00%	1.40%
	CNN (5, crop)	92.80%	4.80%	9.60%	4.00%	9.30%	8.60%	1.80%
	CNN (7,5,3, crop)	93.20%	7.60%	6.00%	4.10%	12.10%	10.50%	4.30%

Table A.34: CNN2 complete results - One-Attack framework - Time Series.

CNN2 - One Attack - Time Series									
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Body Double	CNN (3, resize)	78.73%	22.80%	19.67%	20.90%	52.90%	35.97%	29.13%	
	CNN (5, resize)	83.57%	12.87%	20.13%	13.97%	47.57%	29.73%	20.40%	
	CNN (7,5,3, resize)	80.33%	19.67%	19.47%	17.33%	44.07%	39.47%	21.70%	
	CNN (3, crop)	96.17%	2.73%	4.80%	2.30%	5.17%	4.23%	1.23%	
	CNN (5, crop)	96.47%	1.00%	5.90%	1.23%	3.50%	2.40%	0.20%	
Ecoflex	CNN (7,5,3, crop)	96.50%	4.57%	2.43%	2.13%	3.77%	3.10%	1.07%	
	CNN (3, resize)	85.87%	10.73%	17.43%	12.80%	39.60%	21.60%	13.03%	
	CNN (5, resize)	86.50%	11.03%	15.87%	13.20%	41.57%	21.23%	13.33%	
	CNN (7,5,3, resize)	87.67%	11.20%	13.40%	12.20%	27.53%	22.97%	10.40%	
	CNN (3, crop)	97.00%	3.07%	2.93%	1.57%	3.50%	3.00%	0.33%	
Playdoh	CNN (5, crop)	97.27%	2.30%	3.10%	1.43%	3.17%	2.23%	0.30%	
	CNN (7,5,3, crop)	96.77%	3.13%	3.23%	1.90%	2.20%	1.50%	0.60%	
	CNN (3, resize)	89.33%	7.30%	14.10%	10.00%	27.33%	16.13%	11.30%	
	CNN (5, resize)	88.00%	11.80%	12.13%	10.37%	37.53%	16.77%	8.80%	
	CNN (7,5,3, resize)	87.77%	8.07%	16.57%	10.70%	24.25%	19.80%	10.15%	
Playdoh	CNN (3, crop)	97.57%	1.20%	3.63%	1.40%	1.40%	1.27%	0.40%	
	CNN (5, crop)	97.30%	2.83%	2.57%	1.23%	1.23%	1.07%	0.33%	
	CNN (7,5,3, crop)	98.03%	1.70%	2.27%	1.40%	1.70%	1.23%	0.10%	

Table A.35: CNN2 complete results - Unseen-Attack framework - Time Series.

CNN2 - Unseen Attack - Time Series									
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Body Double	CNN (3, resize)	68.80%	9.10%	53.30%	28.00%	69.60%	55.40%	45.70%	
	CNN (5, resize)	69.70%	6.50%	54.10%	24.60%	60.40%	46.10%	38.10%	
	CNN (7,5,3, resize)	73.80%	8.90%	43.50%	23.40%	54.40%	51.40%	32.70%	
	CNN (3, crop)	84.00%	0.70%	31.30%	9.70%	21.30%	17.00%	8.00%	
	CNN (5, crop)	84.90%	1.60%	28.50%	9.80%	29.00%	23.80%	8.70%	
Ecoflex	CNN (7,5,3, crop)	83.80%	0.60%	31.80%	10.30%	30.50%	25.90%	9.90%	
	CNN (3, resize)	85.80%	16.60%	11.90%	12.80%	46.00%	22.10%	15.30%	
	CNN (5, resize)	87.40%	12.00%	13.20%	11.10%	39.30%	18.50%	12.60%	
	CNN (7,5,3, resize)	85.30%	16.50%	12.90%	11.60%	33.10%	28.60%	12.00%	
	CNN (3, crop)	98.10%	1.00%	2.80%	0.70%	2.80%	2.20%	0.10%	
Playdoh	CNN (5, crop)	98.40%	0.60%	2.60%	1.00%	2.30%	1.80%	0.10%	
	CNN (7,5,3, crop)	98.20%	0.70%	2.80%	0.70%	2.40%	1.40%	0.10%	
	CNN (3, resize)	83.40%	12.70%	20.50%	14.20%	44.20%	27.10%	18.20%	
	CNN (5, resize)	84.00%	12.20%	19.90%	15.00%	38.20%	21.40%	16.80%	
	CNN (7,5,3, resize)	84.10%	18.80%	13.00%	13.50%	30.60%	26.10%	14.00%	
Playdoh	CNN (3, crop)	97.00%	0.80%	5.30%	1.10%	1.50%	1.30%	0.10%	
	CNN (5, crop)	97.60%	1.90%	2.90%	1.40%	2.20%	1.70%	0.20%	
	CNN (7,5,3, crop)	96.90%	0.80%	5.30%	0.90%	1.10%	0.90%	0.10%	

A.6 CNN2 Regularization ($CNN2_{reg}$) Complete Results

Table A.36: CNN2 regularization complete results - Unseen-Attack framework - Cross Match.

CNN2reg - Unseen attack - Cross Match									
		Accuracy	APCER	BP CER	EER	BP CER@1	BP CER@5	BP CER@10	
Body Double	TASK TASK + ADV TASK + TRANSFER TASK + ADV + TRANSFER	82.10%	7.00%	28.90%	13.00%	nan	nan	nan	
		80.78%	1.40%	37.25%	7.51%	14.97%	14.97%	8.16%	
		65.60%	1.40%	67.80%	19.00%	37.10%	34.70%	22.60%	
		83.00%	2.00%	32.19%	6.26%	9.29%	9.29%	4.17%	
Ecoflex	TASK TASK + ADV TASK + TRANSFER TASK + ADV + TRANSFER	90.70%	2.80%	15.90%	6.70%	17.20%	6.50%	4.50%	
		91.08%	1.20%	16.67%	2.47%	4.65%	4.65%	2.27%	
		85.00%	0.60%	29.30%	3.70%	7.30%	7.30%	2.10%	
		90.20%	0.60%	20.10%	2.20%	nan	nan	nan	
Playdoh	TASK TASK + ADV TASK + TRANSFER TASK + ADV + TRANSFER	90.30%	5.20%	14.30%	6.40%	8.60%	6.00%	4.40%	
		99.18%	1.60%	0.00%	0.00%	0.00%	0.00%	0.00%	
		99.20%	1.00%	0.60%	0.20%	0.20%	0.20%	0.20%	
		99.39%	0.60%	0.62%	0.00%	0.00%	0.00%	0.00%	

Table A.37: CNN2 regularization complete results - Unseen-Attack framework - Digital Persona.

CNN2reg - Unseen attack - Digital Persona								
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10
Ecoflex	TASK	95.20%	5.60%	4.00%	3.70%	9.60%	3.40%	2.30%
	TASK+ ADV	98.20%	2.40%	1.20%	0.00%	0.00%	0.00%	0.00%
	TASK + TRANSFER	97.80%	1.20%	3.20%	0.00%	0.00%	0.00%	0.00%
	TASK + ADV + TRANSFER	99.40%	1.20%	0.00%	0.00%	0.00%	0.00%	0.00%
Gelatine	TASK	79.60%	4.00%	36.80%	14.10%	35.70%	18.50%	13.20%
	TASK+ ADV	66.80%	1.20%	65.20%	8.10%	13.40%	13.40%	6.20%
	TASK + TRANSFER	79.80%	3.60%	36.80%	10.50%	13.50%	13.50%	10.10%
	TASK + ADV + TRANSFER	62.80%	0.00%	74.40%	7.20%	11.50%	11.50%	3.90%
Latex	TASK	89.00%	6.00%	16.00%	7.80%	29.30%	15.90%	11.30%
	TASK+ ADV	98.80%	0.40%	2.00%	0.40%	0.40%	0.40%	0.00%
	TASK + TRANSFER	97.80%	2.00%	2.40%	2.00%	2.00%	2.00%	1.30%
	TASK + ADV + TRANSFER	94.80%	0.40%	10.00%	0.60%	0.60%	0.60%	0.60%
Wood glue	TASK	94.00%	6.80%	5.20%	3.50%	14.20%	5.40%	0.70%
	TASK+ ADV	94.00%	2.00%	10.00%	1.30%	2.10%	2.10%	0.90%
	TASK + TRANSFER	97.60%	0.40%	4.40%	0.30%	0.30%	0.30%	0.00%
	TASK + ADV + TRANSFER	95.40%	0.80%	8.40%	2.00%	3.30%	3.30%	1.20%

Table A.38: CNN2 regularization complete results - Unseen-Attack framework - Green Bit.

CNN2reg - Unseen attack - Green Bit								
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10
Ecoflex	TASK	97.40%	1.60%	3.60%	2.30%	3.40%	0.40%	0.00%
	TASK+ ADV	99.20%	0.00%	1.60%	0.50%	0.50%	0.50%	0.00%
	TASK +TRANSFER	98.60%	0.00%	2.80%	0.30%	0.30%	0.30%	0.00%
	TASK + ADV + TRANSFER	99.60%	0.80%	0.00%	0.00%	0.00%	0.00%	0.00%
Gelatine	TASK	65.40%	1.60%	67.20%	21.80%	74.50%	54.00%	40.20%
	TASK+ ADV	73.80%	0.80%	51.20%	8.80%	12.90%	12.90%	7.60%
	TASK +TRANSFER	66.80%	2.00%	64.00%	17.40%	27.60%	27.60%	17.30%
	TASK + ADV + TRANSFER	76.10%	0.80%	46.80%	4.40%	5.40%	5.40%	4.00%
Latex	TASK	96.80%	4.00%	2.40%	1.20%	2.40%	0.40%	0.40%
	TASK+ ADV	95.30%	0.80%	8.40%	2.00%	2.90%	2.90%	0.60%
	TASK +TRANSFER	94.20%	0.00%	11.60%	0.30%	0.90%	0.90%	0.30%
	TASK + ADV + TRANSFER	92.00%	0.00%	16.00%	0.80%	2.20%	2.20%	1.70%
Wood glue	TASK	95.60%	4.90%	4.00%	2.90%	4.70%	1.70%	0.70%
	TASK+ ADV	94.20%	0.80%	10.80%	1.20%	nan	nan	nan
	TASK +TRANSFER	91.30%	0.00%	17.20%	1.20%	1.20%	1.20%	0.00%
	TASK + ADV + TRANSFER	90.70%	0.80%	17.60%	1.30%	4.10%	4.10%	0.00%

Table A.39: CNN2 regularization complete results - Unseen-Attack framework - Hi Scan.

CNN2reg - Unseen attack - Hi Scan								
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10
Ecoflex	TASK	92.20%	10.00%	5.60%	5.30%	11.90%	8.10%	5.00%
	TASK+ ADV	97.60%	1.20%	3.60%	0.30%	nan	nan	nan
	TASK + TRANSFER	98.20%	2.00%	1.60%	1.40%	1.40%	1.40%	1.40%
	TASK + ADV + TRANSFER	98.60%	2.40%	0.40%	1.30%	1.30%	1.30%	0.00%
	TASK	64.20%	3.60%	68.00%	35.30%	63.80%	62.80%	49.40%
Gelatine	TASK+ ADV	87.40%	2.00%	23.20%	2.00%	2.00%	2.00%	1.10%
	TASK + TRANSFER	85.80%	2.40%	26.00%	4.60%	4.60%	4.60%	4.10%
	TASK + ADV + TRANSFER	91.20%	1.20%	16.40%	1.34%	1.34%	1.34%	0.94%
	TASK	95.80%	4.40%	4.00%	2.70%	6.80%	4.70%	0.70%
	TASK+ ADV	94.80%	1.20%	9.20%	0.70%	1.70%	1.70%	0.70%
Latex	TASK + TRANSFER	96.40%	1.60%	5.60%	0.90%	0.90%	0.90%	0.00%
	TASK + ADV + TRANSFER	93.20%	0.40%	13.20%	0.50%	0.50%	0.50%	0.00%
	TASK	92.80%	4.40%	10.00%	1.70%	6.40%	5.00%	1.40%
	TASK+ ADV	94.40%	2.00%	9.20%	2.20%	2.90%	2.90%	1.70%
	TASK + TRANSFER	95.20%	2.80%	6.80%	1.10%	1.10%	1.10%	1.10%
Wood glue	TASK + ADV + TRANSFER	91.40%	0.80%	16.40%	1.02%	1.69%	1.69%	0.62%

Table A.40: CNN2 regularization complete results - Unseen-Attack framework - Time Series.

CNN2reg - Unseen attack - Time Series								
		Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10
Body Double	TASK TASK+ ADV TASK +TRANSFER TASK + ADV + TRANSFER	84.90%	1.60%	28.50%	9.80%	29.00%	23.80%	8.70%
		86.30%	1.40%	26.00%	6.10%	13.20%	13.20%	5.40%
		82.90%	0.50%	33.70%	6.90%	16.00%	16.00%	6.20%
		83.60%	0.50%	32.00%	6.10%	14.00%	13.60%	4.90%
Ecoflex	TASK TASK+ ADV TASK +TRANSFER TASK + ADV + TRANSFER	98.40%	0.60%	2.60%	1.00%	2.30%	1.80%	0.10%
		97.90%	0.30%	3.90%	0.30%	1.20%	1.20%	0.10%
		99.10%	0.70%	1.10%	0.20%	0.30%	0.30%	0.10%
		98.50%	1.10%	1.90%	0.30%	0.60%	0.60%	0.00%
Playdoh	TASK TASK+ ADV TASK +TRANSFER TASK + ADV + TRANSFER	97.60%	1.90%	2.90%	1.40%	2.20%	1.70%	0.20%
		97.70%	0.90%	3.70%	0.60%	1.20%	1.20%	0.30%
		97.30%	0.20%	5.30%	0.90%	2.90%	2.70%	0.90%
		94.00%	0.20%	11.90%	1.70%	4.70%	4.70%	1.20%

A.7 One-class GAN Complete Results

Table A.41: One-class GAN complete results.

One-class GAN - Unseen Attack									
Sensor	Attack	Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Cross Match	Body Double	66.70%	0.61%	97.00%	45.28%	93.68%	87.17%	80.99%	
	Ecoflex	73.40%	5.60%	93.00%	33.76%	74.09%	59.84%	53.76%	
	Playdoh	70.70%	17.35%	92.60%	37.66%	68.19%	58.23%	52.15%	
Digital Persona	Ecoflex	65.00%	53.87%	49.20%	53.07%	81.62%	76.00%	71.04%	
	Gelatine	64.40%	100.00%	0.00%	33.97%	76.54%	66.08%	59.43%	
	Latex	57.80%	91.20%	26.40%	55.36%	98.94%	96.61%	94.21%	
	Wood glue	51.50%	74.90%	33.20%	57.37%	93.48%	89.28%	85.95%	
	Ecoflex	75.50%	49.20%	85.02%	65.98%	93.84%	89.11%	85.51%	
Green Bit	Gelatine	59.70%	96.50%	22.27%	61.70%	100.00%	99.48%	96.54%	
	Latex	53.90%	95.20%	8.10%	64.74%	96.09%	92.61%	91.05%	
	Wood glue	58.40%	40.80%	68.83%	50.15%	98.83%	92.62%	86.82%	
	Ecoflex	72.80%	98.00%	0.40%	50.10%	95.86%	94.16%	85.37%	
Hi Scan	Gelatine	53.00%	37.20%	85.20%	78.56%	100.00%	100.00%	100.00%	
	Latex	50.00%	22.40%	96.00%	63.25%	98.67%	97.69%	93.90%	
	Wood glue	51.20%	17.20%	96.00%	58.09%	99.38%	93.65%	90.53%	
Time Series	Body Double	52.30%	100.00%	0.00%	60.30%	100.00%	100.00%	100.00%	
	Ecoflex	50.80%	74.00%	64.59%	51.44%	100.00%	100.00%	100.00%	
	Playdoh	53.60%	64.26%	64.59%	39.93%	100.00%	90.00%	100.00%	

A.8 Image-to-image GAN Complete Results

Table A.42: Image-to-image GAN complete results - Unseen-Attack framework - Cross Match.

Image-to-image GAN - Joint Training - Unseen attack - Cross Match									
Attack	With Generator	Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Body Double	No	82.30%	1.20%	34.40%	8.30%	81.40%	17.40%	6.10%	
	Yes	81.30%	0.80%	36.80%	7.60%	86.20%	48.90%	33.00%	
Ecoflex	No	95.90%	0.80%	7.40%	1.20%	81.80%	38.20%	13.00%	
	Yes	69.50%	47.20%	13.70%	25.70%	81.80%	60.30%	50.90%	
Playdoh	No	96.60%	1.80%	5.00%	2.40%	4.10%	1.00%	0.40%	
	Yes	73.40%	50.40%	1.90%	5.00%	43.50%	11.90%	3.20%	
Image-to-image GAN - Phased Training - Unseen attack - Cross Match									
Attack	With Generator	Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Body Double	No	80.00%	1.00%	39.30%	8.50%	35.60%	19.40%	9.80%	
	Yes	80.80%	1.00%	37.70%	8.90%	23.10%	12.30%	5.80%	
Ecoflex	No	97.10%	1.00%	4.80%	2.70%	8.80%	1.80%	0.90%	
	Yes	91.50%	2.80%	14.30%	5.50%	55.40%	25.30%	15.30%	
Playdoh	No	96.50%	1.60%	5.40%	2.20%	6.30%	1.30%	0.40%	
	Yes	98.20%	1.60%	2.10%	1.40%	2.20%	0.50%	0.00%	

Table A.43: Image-to-image GAN complete results - Unseen-Attack framework - Digital Persona.

Image-to-image GAN - Phased Training - Unseen attack - Digital Persona									
	With Generator	Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	No	95.80%	6.80%	1.60%	1.90%	7.40%	0.80%	0.00%	
	Yes	97.20%	4.00%	1.60%	2.40%	4.90%	1.60%	0.80%	
Gelatine	No	70.20%	1.60%	58.00%	19.50%	74.90%	46.00%	29.80%	
	Yes	70.80%	2.40%	56.00%	22.40%	77.80%	58.90%	38.50%	
Latex	No	92.40%	3.60%	11.60%	6.10%	25.10%	6.10%	5.00%	
	Yes	92.80%	3.60%	10.80%	4.80%	11.10%	3.10%	2.30%	
Wood glue	No	97.60%	3.60%	1.20%	2.00%	5.70%	0.40%	0.40%	
	Yes	98.00%	2.40%	1.60%	0.40%	50.00%	25.00%	12.50%	
Image-to-image GAN - Phased Training - Unseen attack - Digital Persona									
	With Generator	Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	No	98.20%	2.40%	1.20%	0.80%	3.60%	2.30%	0.40%	
	Yes	98.20%	2.80%	0.80%	1.10%	3.90%	0.70%	0.00%	
Gelatine	No	70.20%	2.00%	57.60%	18.40%	79.50%	52.50%	34.30%	
	Yes	72.40%	2.40%	52.80%	13.10%	49.80%	27.80%	18.00%	
Latex	No	89.80%	3.60%	16.80%	10.70%	27.20%	15.10%	7.20%	
	Yes	91.80%	1.20%	15.20%	5.70%	14.10%	4.90%	2.10%	
Wood glue	No	97.00%	3.20%	2.80%	3.40%	5.70%	2.70%	0.80%	
	Yes	97.80%	2.80%	1.60%	2.40%	3.20%	0.40%	0.00%	

Table A.44: Image-to-image GAN complete results - Unseen-Attack framework - Green Bit.

Image-to-image GAN - Phased Training - Unseen attack - Green Bit									
	With Generator	Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	No	92.00%	5.30%	10.80%	3.90%	12.50%	6.50%	3.50%	
	Yes	89.70%	5.30%	15.20%	1.50%	15.70%	7.80%	4.80%	
Gelatine	No	59.40%	2.00%	78.80%	35.80%	89.00%	78.40%	70.70%	
	Yes	56.70%	2.40%	83.60%	40.80%	84.40%	79.10%	75.30%	
Latex	No	89.70%	4.50%	16.00%	9.20%	33.40%	12.60%	7.60%	
	Yes	86.90%	6.50%	19.60%	5.20%	39.80%	17.10%	9.70%	
Wood glue	No	93.40%	6.10%	7.20%	6.30%	28.20%	8.80%	4.30%	
	Yes	94.60%	4.50%	6.40%	2.20%	10.00%	3.60%	1.40%	
Image-to-image GAN - Phased Training - Unseen attack - Green Bit									
	With Generator	Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	No	92.00%	4.50%	11.60%	7.60%	37.90%	12.00%	5.90%	
	Yes	90.10%	5.70%	14.00%	10.80%	43.00%	27.00%	12.10%	
Gelatine	No	55.70%	1.60%	86.40%	33.70%	86.80%	75.40%	67.90%	
	Yes	56.10%	1.60%	85.60%	37.10%	84.70%	72.90%	66.20%	
Latex	No	89.70%	6.50%	14.00%	11.00%	47.10%	16.30%	10.90%	
	Yes	89.90%	5.30%	14.80%	11.10%	47.00%	27.60%	8.50%	
Wood glue	No	90.50%	6.90%	12.00%	9.10%	46.00%	24.70%	6.60%	
	Yes	92.80%	7.30%	7.20%	6.60%	22.90%	7.30%	3.10%	

Table A.45: Image-to-image GAN complete results - Unseen-Attack framework - Hi Scan.

Image-to-image GAN - Phased Training - Unseen attack - Hi Scan									
	With Generator	Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	No	96.20%	2.80%	4.80%	1.50%	39.20%	1.20%	0.00%	
	Yes	83.20%	30.40%	3.20%	13.90%	40.30%	23.40%	16.90%	
Gelatine	No	69.40%	0.80%	60.40%	5.50%	39.80%	30.00%	17.60%	
	Yes	69.80%	2.00%	58.40%	5.80%	34.00%	25.00%	19.50%	
Latex	No	94.60%	1.60%	9.20%	4.50%	12.20%	4.10%	1.60%	
	Yes	92.40%	8.00%	7.20%	6.30%	30.10%	9.50%	3.10%	
Wood glue	No	91.40%	4.40%	12.80%	7.70%	23.00%	14.20%	6.00%	
	Yes	80.40%	24.40%	14.80%	15.30%	42.70%	27.20%	24.00%	
Image-to-image GAN - Phased Training - Unseen attack - Hi Scan									
	With Generator	Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Ecoflex	No	96.00%	2.80%	5.20%	3.50%	10.50%	2.20%	0.70%	
	Yes	97.40%	1.20%	4.00%	1.10%	10.10%	1.60%	0.50%	
Gelatine	No	67.80%	1.20%	63.20%	17.40%	62.90%	44.90%	23.40%	
	Yes	73.80%	1.20%	51.20%	9.50%	59.00%	39.80%	34.70%	
Latex	No	95.80%	2.00%	6.40%	4.30%	16.10%	5.30%	0.80%	
	Yes	93.80%	2.00%	10.40%	4.80%	20.80%	5.90%	2.20%	
Wood glue	No	95.40%	1.60%	7.60%	5.50%	18.00%	1.90%	1.10%	
	Yes	95.60%	3.60%	5.20%	3.90%	27.60%	6.00%	0.00%	

Table A.46: Image-to-image GAN complete results - Unseen-Attack framework - Time Series.

Image-to-image GAN - Joint Training - Unseen attack - Time Series									
Attack	With Generator	Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Body Double	No	91.70%	0.50%	16.20%	0.40%	17.90%	7.90%	2.80%	
	Yes	85.60%	12.80%	16.10%	14%	52.10%	30.80%	19.50%	
Ecoflex	No	98.80%	1.60%	0.90%	1.20%	1.60%	0.10%	0.00%	
	Yes	87.80%	23.20%	1.40%	4.80%	23.30%	4.20%	0.80%	
Playdoh	No	97.20%	0.10%	5.50%	0.80%	11.90%	0.10%	0.00%	
	Yes	84.20%	27.20%	4.50%	9.90%	42.50%	21.10%	10.20%	
Image-to-image GAN - Phased Training - Unseen attack - Time Series									
Attack	With Generator	Accuracy	APCER	BPCER	EER	BPCER@1	BPCER@5	BPCER@10	
Body Double	No	90.90%	0.70%	17.40%	0.40%	17.90%	9.70%	4.50%	
	Yes	93.60%	0.50%	12.40%	3.00%	9.60%	1.40%	0.50%	
Ecoflex	No	99.40%	0.40%	0.80%	0.10%	7.40%	5.30%	1.10%	
	Yes	99.40%	0.90%	0.30%	0.30%	0.30%	0.10%	0.00%	
Playdoh	No	96.80%	0.40%	5.90%	1.30%	19%	0.40%	0%	
	Yes	96.50%	0.30%	6.80%	0.20%	19.40%	0.70%	0.00%	

Appendix B

One-class GAN outputs

In this appendix it is possible to visualize nine examples of fingerprints from all sensors, generated by the first GAN (one-class GAN) of this dissertation (Figure [B.1](#)).

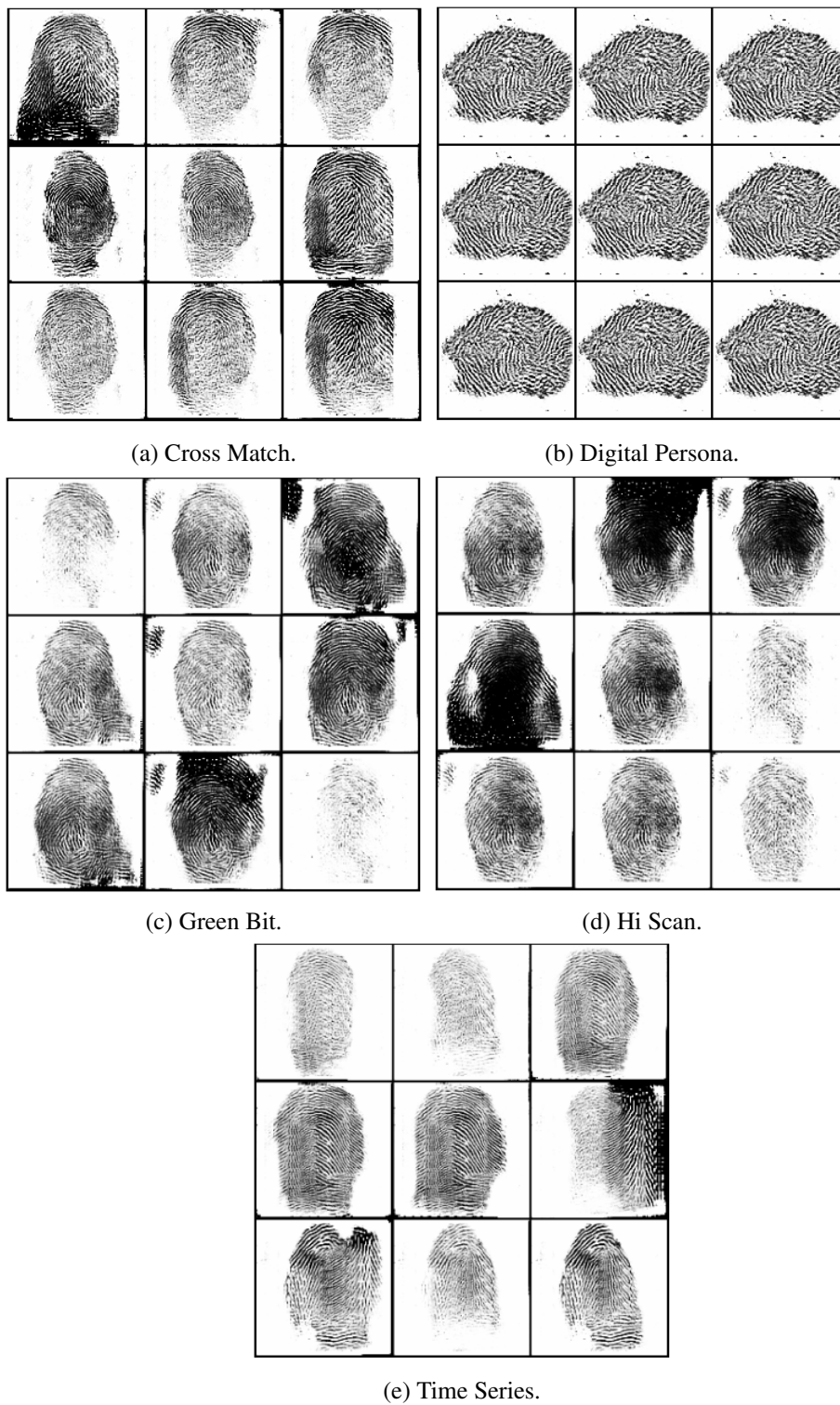


Figure B.1: Examples of generated fingerprints from all sensors - One-class GAN.

Appendix C

Image-to-image GAN outputs

In this appendix it is possible to visualize three examples of fingerprints from all sensors and their corresponding modified versions returned by the Generator of the final model developed in this dissertation: the image-to-image GAN (Figure [C.1](#)).

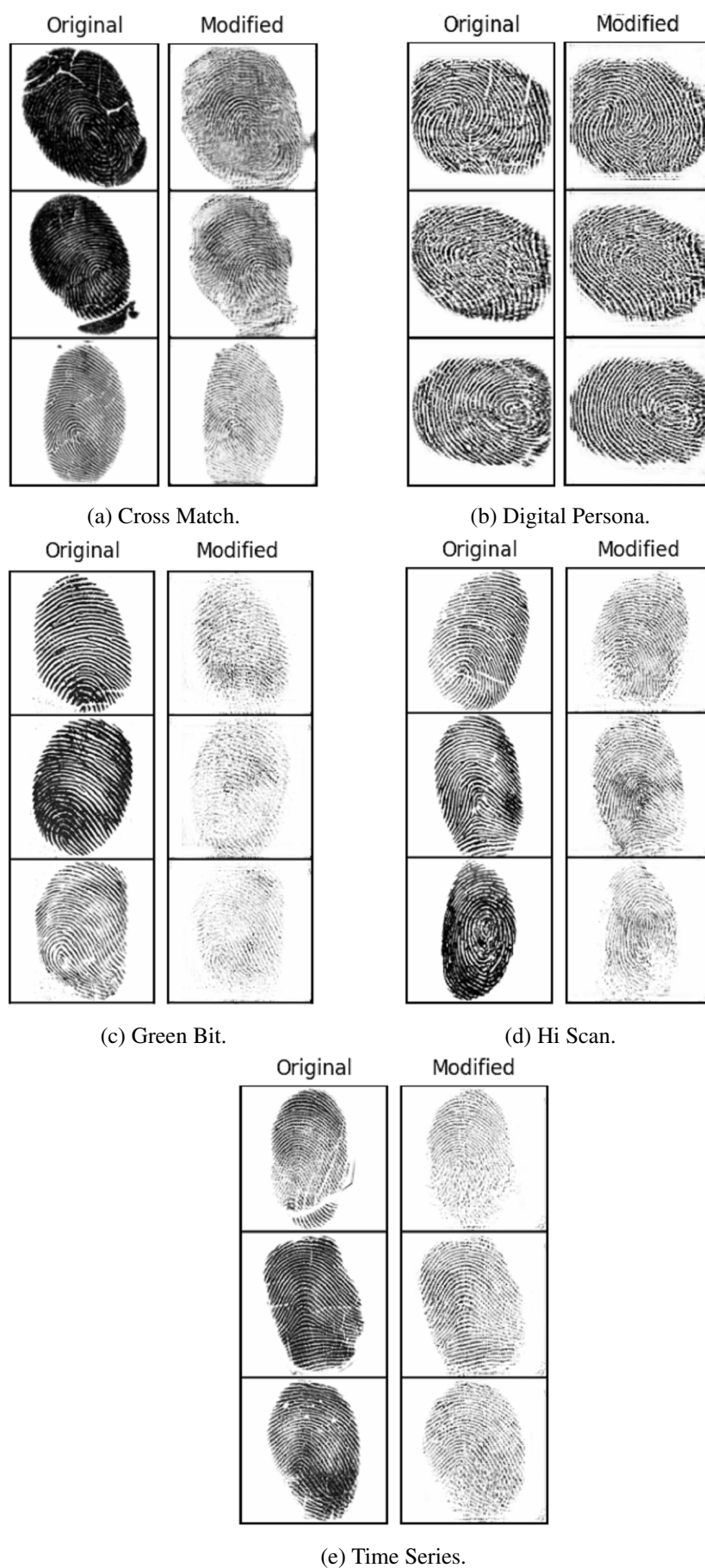


Figure C.1: Examples of modified fingerprints from all sensors - Image-to-image GAN.