University of Padova

Department of Mathematics "Tullio Levi-Civita"

Master Thesis in Cybersecurity

Generative Fingerprint Augmentation against Membership Inference Attacks

Supervisor
Prof. Simone Milani
University of Padova

Co-supervisor
Prof. Mauro Conti
University of Padova

Master Candidate
Saverio Cavasin

Advisor
Dr. Daniele Mari

Academic Year
2022-2023

"Not a mathematician. Not an engineer. All he knows is what he's not. If only he could see what he is."

— Balto (kinda)

Abstract

Generative models and biometric systems are two of the most prominent technologies currently investigated both in industry, academia and cyberthreat intelligence. Generative models are gaining significant attention as the potential catalyst for the next industrial revolution (DALL- E, ChatGPT, Midjourney, etc), while biometric systems serve as efficient and widely adopted means of authentication across various devices and systems. Since automated sample generation can be useful to solve privacy and data scarcity issues that usually affect learned biometric models, such technologies are becoming widely spread in this field as well.

This thesis aims at investigating an extended privacy protection assessment for neural networks, focusing on vulnerabilities related to databases of fingerprints, which have increasingly gained significance in various domains, including banking operations and daily smartphone usage. The research efforts were focused on assessing the vulnerabilities of generative machine learning models, particularly those trained on fingerprint images (by nature and purpose harder to recover, and thus, often linked to less generalized and smaller datasets), because they are especially prone to specific privacy inferring attacks. The objective is to examine the black box vulnerabilities arising from privacy concerns associated with these models. Through a comprehensive evaluation based on the training and testing of several different networks, it is possible to understand the potential privacy risks faced by generative models and contribute to the development of effective defense strategies for safeguarding sensitive biometric data.

The analysis assessed the vulnerabilities of generative machine learning models concerning identity protection by designing and testing identity inference attack strategies on fingerprint datasets carried by means of attacking network, engineered and fine tuned with the sole purpose of gathering relevant information from the "victim" models .

Several different attack strategies have been carried, based on the vulnerability (membership/identity inference) intended to be exploited and the models (Generative Adversarial Networks / Autoencoders) target of the attacks.

By testing and analyzing these frameworks with custom membership inference attacks (MIA) and user-level membership attacks, that are modified and renamed as identity inference attacks (IIA), it was possible to assess the quality of some proposed defensive mechanism and employ the gathered information to improve the robustness of the models and the aforementioned defenses.

Experimental results show that the proposed solutions prove to be effective under different configurations and easily extendable to other biometric measurements.

Contents

Listing of figures

Listing of tables

Listing of acronyms

GAN . . . . . Generative Adversarial Network

CTI . . . . . Cyberthreat Intelligence

MIA . . . . . Membership Inference Attack

ULMIA . . . . . User-Level Membership Inference Attack

IIA . . . . . Identity Inference Attack

ACS . . . . . Access Control System

VAE . . . . . Variational Autoencoders

DC . . . . . Deep Convolutional

LZW . . . . . Lempel-Ziv-Welch

ANN . . . . . Artificuial Neural Network

PSNR . . . . . Peak Signal-to-Noise Ratio

API . . . . . Application Programming Interface

MSH . . . . . Mean and Scale Hyperprior

IS . . . . . Inception Score

MSE . . . . . Mean Squared Error

PSNR . . . . . Peak Signal to Noise Ratio

BPP . . . . . Bits Per Pixel

KL . . . . . Kullback-Leibler

1 Introduction

1.1 Biometrics

Biometrics refers to the science and technology of identifying and verifying individuals based on their unique physical, chemical, and behavioral characteristics.

Such characteristics, also known as biometric traits, serve as distinctive identifiers that can be acquired, processed, and compared to establish the identity of a person with a certain degree of accuracy.

These technologies represent a useful tool for security and authentication, as they provide an efficient and powerful alternative to traditional systems.

Indeed, we can divide authentication means by the relationship these last share with the users.

Shortly,

1. What the user **knows**
2. What the user **owns**
3. What the user **is**

The first category represents the most traditional and practical means: passwords, pins, codewords, etc. These ones are really efficient, easily spreadable and scalable through large Access Control Systems (ACS).

The second one introduces the concept of physical ownership of the authentication means tokens, smartcards, etc. These are often more secure than passwords, nevertheless, they introduce some issues about the need of being always carried by the user when required, implying worsened scalability as well.

Finally, biometrics represent precisely the last category of authentication means. What the user is; physically, biologically, behaviourally. There are several ways of additionally dividing biometrics categories as well. For instance [1] provides a distinction between hard biometrics, soft biometrics, and hidden biometrics. Recently, on top of these more traditional means, the behavioral category was introduced as well.

1. **Hard** biometrics: These are considered the most classic or traditional, such as, for instance: faces, fingerprints, and the ones related to eyes, like iris, retina, sclera, etc.

2. **Soft** biometrics : these vary among weaker identifiers from the ones related to the head, as skin color, hair color or facial measurements, to the whole body, as height or weight, and even in some cases to accessories, such as glasses/hats/specific clothing and ornaments.

3. **Hidden** biometrics: These are also named intrinsic biometrics, and they are basically related to medical data, such as biosignals, MRI images, or X-Ray images. These identifiers represent very informative information, even though are by nature way harder to acquire, process, and share.

4. **Behavioural** biometrics: novel kinds of traits, based on the way each individual differently moves and carries his body around the space. Some of the main examples are voices, gestures, and gait, which refer to the position and the movements exhibited by the human body during the process of walking, running, and more generally, moving.

Even though these means can be used jointly to improve reliability and security, second or third levels of authentication can introduce the issues of slowing down the whole access control and increasing the costs of the related technologies and systems.

Several biometric characteristics are being evaluated in different scenarios and applications. Each biometric identifier carries its own perks and flaws, therefore, choosing to use a specific biometric trait or another for a particular application depends on a variety of issues besides its matching performance and economic costs. Jain et al. [2] have identified seven factors that determine the suitability of a physical or a behavioral trait to be used in a biometric application.

1. **Universality**. Every individual requiring access to the application should possess the trait and should be able to provide it to analysis in the same fashion.

2. **Uniqueness**. Each biometric trait from an individual should be sufficiently different across other individuals in the population.

3. **Permanence**. The trait should be sufficiently invariant over a relevant period of time with respect to the same (or similar) matching algorithms. A trait that introduces several significant changes over time is not a useful nor reliable identifier.

4. **Measurability**. It should be possible to acquire and digitize the biometric trait using suitable devices that do not cause unexpected inconvenience to the individuals. Furthermore, the acquired raw data should be suitable for processing and manipulation in order to extract representative feature sets and key point descriptors.

5. **Performance**. The recognition accuracy and the resources required to achieve such accuracy should meet the constraints imposed by the applications and the systems of the specific task.

6. **Acceptability**. Individuals in the target population that will utilize the application should be willing to present their biometric trait to the system without discomfort or annoyance with respect to their moral, social, religious, and ethical beliefs.

7. **Circumvention**. The trait of an individual should introduce significant challenges in being imitated using artifacts (e.g., synthetic fingers, deep fakes, etc.), in the case of physical traits, and mimicry, in the case of behavioral traits (audio manipulations, forged signatures, etc). [3]

No biometric identifier should be expected to effectively meet every single requirement (e.g., accuracy, practicality, cost) imposed by all applications (access control, surveillance, diagnosis, etc.). Therefore, no biometric trait is "the best" but many are "suitable" for the task. The effectiveness of different identifiers with respect to a specific application is established depending on the nature and requirements of the application and the related systems, and the properties of the biometric characteristics.

1.1.1 Fingerprints

During this work, the efforts are specifically focused on fingerprints, as they are known to be among the most used and reliable biometric systems.

Fingerprints are constituted by the raised papillary ridges that run across the skin's surface. Humans and other mammals display them on their fingers, thumbs, and palms, jointly with the toes and soles of the feet. Ridges are an evolutionary trait developed to provide friction in order to aid grip and locomotion. Their flow usually builds patterns, but often ridges themselves

don't run continuously due to the presence of some breaks and deviations in their structure (i.e. where the ridges end or deviate, known as *minutiae*) [4].

A fingerprint can therefore be defined as a distinct pattern of ridges and valleys on the finger surface of an individual.

1. **Ridge**: single curved segment whereas
2. **Valley**: area between two adjacent ridges.

So, in a given sample, the dark areas of the fingerprint are the ridges while the white areas between them are the valleys[5].

In an identification system based on fingerprints, the captured image needs to be matched against the stored fingerprint templates of every user present in the database. This eventually involves several computations and complex searching loops. Therefore there is a need for a fingerprint classification system that restricts the size of the challenged templates database. To accomplish this, the minutiae features are first extracted and then matched against the fingerprint submitted in the system. The template size of fingerprint biometric systems is often limited and compact as most of these identification systems are based on minutiae.

The ridge patterns and minutiae occurrences are random in nature and are used as the basis for establishing the identity of individuals. This is accurately efficient because given two different skin regions (bearing papillary ridge systems) the same arrangement of minutiae has never been found (not even in the case of monozygotic twins). This is due to environmental factors that affect their development inside the womb, hence it's absolutely impossible even for identical twins to have the exact same fingerprints. [6]). Similarly, the same concept holds true for palmprints as well, but the area of ridged skin is much larger, containing more detail. This means that expanding the acquisition area will also significantly increase the dimension and cost requirements for the sensor modules.

Instead, on fingerprints, minutiae are the major features of each sample and can be efficiently used in the matching process. Indeed, minutiae key points are used to determine the uniqueness of a given fingerprint image. A good quality (both the devices and the skin are sufficiently clean) fingerprint image can have between 25 to 80 minutiae. This depends on the acquiring system scanner resolution and the way the finger was placed upon the sensor. [5]

Generally speaking, minutiae can be defined as the points where the ridge lines end or fork. So the minutiae points can be seen as the local ridge discontinuities and can be of many types. Let's introduce the most common:

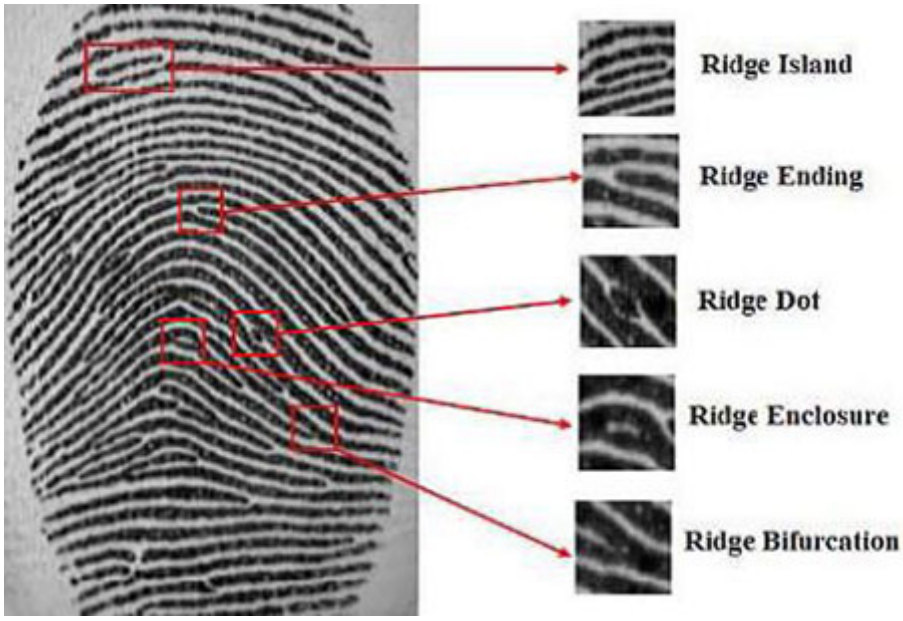


Figure 1.1: Common Minutiae Classification

1. **Ending** is the point where the ridge ends suddenly.
2. **Bifurcation** is the point where a single ridge branches out into two or more ridges.
3. **Dots** are very small ridges.
4. **Islands**: slightly longer than dots and occupy a middle space between two diverging ridges.
5. **Ponds or Lakes**: empty space between two diverging ridges.
6. **Spurs**: notch protruding from a ridge.
7. **Bridges**: small ridges that join two longer adjacent ridges.
8. **Crossovers**: when two ridges cross each other.

Endings and bifurcations are the most common among all the different types, moreover, the other minutiae are based on a combination of these two types. Figure 1.1 displays some of the common minutiae patterns.

Biometric recognition systems acquire and digitize the minutiae arrangement and the flow and orientation of the ridges to create the biometric template. These templates are then stored

in the final dataset that will provide the required information to carry either the verification or identification, upon the application usage.

Fingerprints can be acquired through different means and technologies.

1. Previously, a widely spread method was using **ink and paper** but scalability, efficiency, and comfort suggested the development of novel systems.
2. **scanning** allows for different sample acquisitions, either when the fingers are placed upon the scanner or when they get rolled across a specific platen.
3. Additional practicality, both concerning the quality of the impressions and hygienic regulations introduced the **contactless** method that captures the required level of detail at a proximal distance.

Matching Modelization

A biometric system can be generally modeled by four main modules[3]: a sensor module; a quality assessment and feature extraction module; a matching module; and a database module.

1. **Sensor module**: A suitable biometric reader or scanner is required to acquire the raw biometric data of an individual. The sensor module defines the human-machine interface and is, therefore, pivotal to the performance of the biometric system. A poorly designed interface can result in an acquisition failure and, consequently, deny acceptability. Since most biometric modalities are acquired as images (differently from voice which is audio-based and odor which is chemical-based), the quality of the raw data is impacted by the characteristics of the employed camera technology.
2. **Feature extraction module**: The quality of the biometric data acquired by the sensor is first assessed in order to determine its suitability for further processing. Typically, the acquired data is subjected to a signal enhancement algorithm in order to improve the sample. However, often, the quality of the data may be too poor for the system (resolution/noise/dirt, etc.) and the user is then asked to submit the biometric data again. After additional processing, a set of salient discriminatory features is extracted to represent the underlying trait.
3. **Matching module**: The extracted features are compared against the stored templates to generate matching scores. The matching score may be moderated by the quality of the presented biometric data. The matcher module also encapsulates a decision-making module, where the matching scores are stored and evaluated to either validate a claimed identity for verification or provide a ranking of the enrolled identities for identification (I'll explain more about the difference soon).

4. **Database module**: The database acts as the repository of biometric data. During enrollment, the feature set extracted from the raw biometric sample (i.e., the template) is stored in the database (possibly) along with some biographic information (such as name, Personal Identification Number (PIN), address, etc.) characterizing the user. The acquisition process may or may not be supervised by a human depending on the application.

As metioned earlier, depending on the applications, a biometric system may operate either in **verification** or **identification** mode (see Figure 1.3).

1. In **verification** mode, the system validates someone's identity by comparing the biometric data acquired in a given moment with the related biometric template(s) stored in the system database. This way, an individual who desires to be recognized is claiming an identity, and the system conducts a one-to-one comparison to determine whether such a claim is true or not. Verification is typically used for what is called *positive recognition*, where the aim is to prevent multiple people from accessing resources granted to the same identity.

2. In **identification** mode, the system aims at recogniziing an individual by looking up to the templates of all the users present in the database in order to find a match. Therefore, differently from the verification mode, the system is tasked with a one-to-many comparison to establish someone's identity without the subject having to claim an identity (usually, if the system is not able to find any match at all, the process is terminated with a *failed identification* outcome). Identification is hence employed in what is referred to as "negative recognition" applications, where the system is tasked with establishing fairness when an individual denies being related to a certain identity in favor of another. The purpose of negative recognition is to prevent a single person from using multiple identities. Identification may also be used in positive recognition for convenience (the user is not required to claim an identity). While traditional methods of personal recognition such as passwords, PINs, keys, and tokens may work for positive recognition, negative recognition can only be established through biometrics.

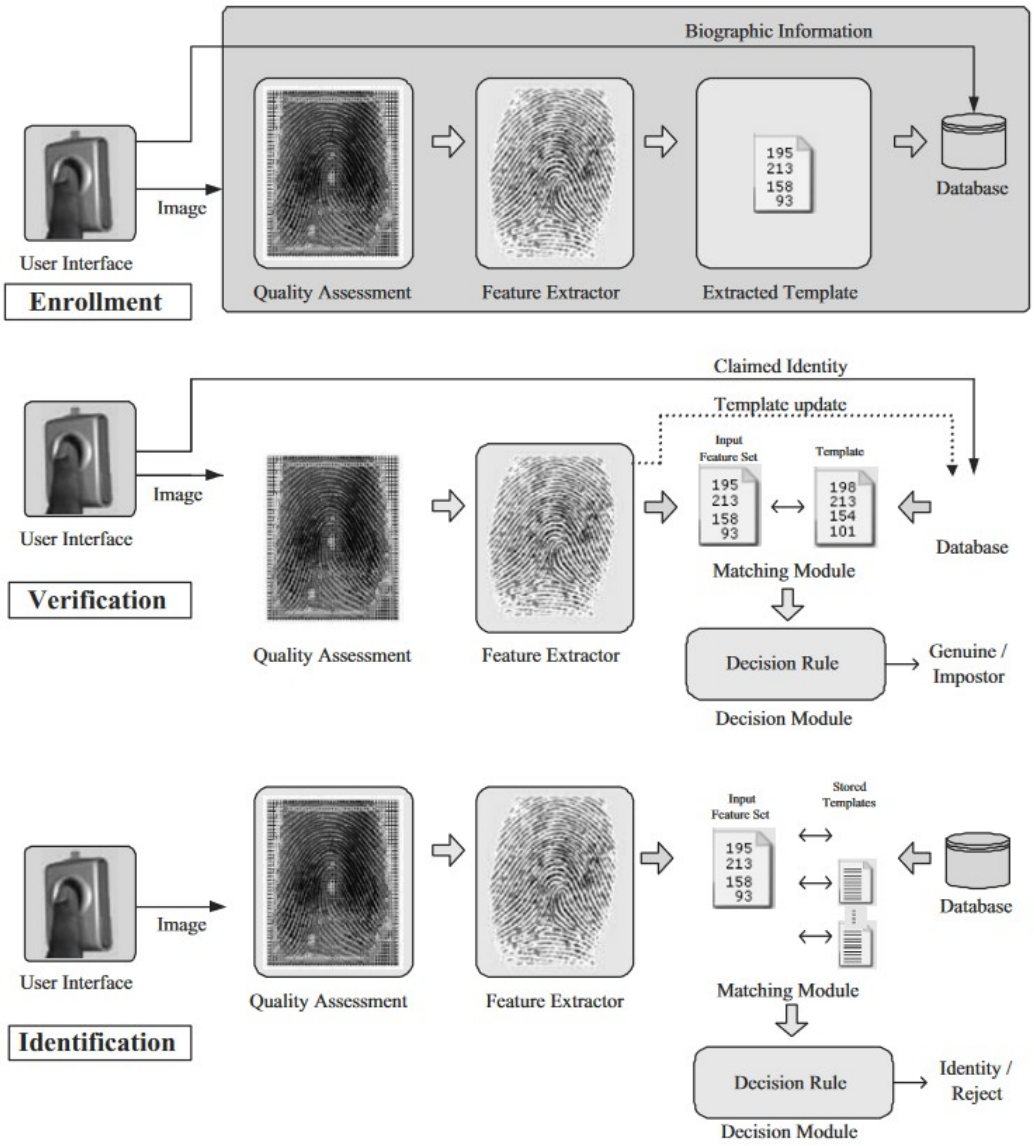


Figure 1.2: Identification vs Verification mode

To evaluate how accurate a biometric system is, i.e. to measure its biometric performance, many genuine and fraudulent attempts are made with the system, and all the matching scores are saved. By applying a varying scoring threshold to such scores, pairs of outcome instance ratings can be calculated and different reliability/usability assessments about the system can be made.

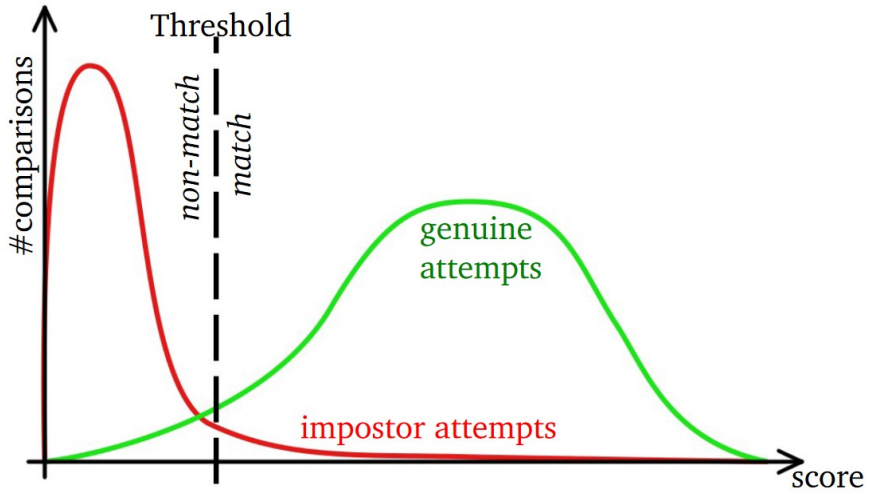


Figure 1.3: General Performance Problem

Let's define some of the most useful and common rating methods.

1. **FMR - False Match Rate** Proportion of fraudolent attempts that are falsely declared to match a template of another object.

2. **FNMR - False NonMatch Rate** Proportion of genuine attempts that are falsely declared not to match a template of the same object.

3. **FTA - Failure to Acquire Rate** Proportion of the attempts for which the system fails to produce a sample of sufficient quality.

4. **FAR - False Accept Rate** and **FRR - False Reject Rate**. Similar to FMR and FNMR respectively, but the definition distinguishes between *attempts* and *transactions*. A transaction may consist of a sequence of attempts. Depending on the system configurations the outcome of individual attempts may affect the transaction differently. FAR and FRR also consider the FTA. In the case of a transaction consisting of exactly one attempt, FAR and FRR are computed as follows:

$$FAR = FMR \cdot (1 - FTA) \quad , \quad FRR = FTA + FNMR \cdot (1 - FTA)$$

5. **EER - Equal Error Rate** When the proportion of False Matches is the same as False Non Matches (FNMR = FMR).

There are two common ways of using these rates to plot performance-evaluating results:

1. **DET (Detection Error Tradeoff)** graph plots. FRR on the y-axis and FAR on the x-axis, i.e. false negative vs. false positive rate, often employing a logarithmic scale (at least for the FAR axis). As the y-axis shows the number of match errors, the closest curve to the bottom of the plot corresponds to the best biometric performance.

2. **ROC (Receiver Operating Characteristic)** graph plots true positive (1 - FRR) and false positive (FAR) rates. In these, the best biometric performance can be observed near the top of the plots. DET curves are generally better at highlighting areas of interest and critical operating levels. Therefore the former is the most commonly used [7].

Biometric performance evaluation is actually standardized by ISO/IEC in the 19795 series of standards.

Performance evaluation may be divided according to three aspects: technology, scenario, and operational evaluation.

1. **Technology**: the evaluation uses the saved data, e.g. previously acquired fingerprint images.

2. **Scenario**: The evaluation of the whole end-to-end system adopts a prototype, almost *digital twin* or simulated environment.

3. **Operational**: The performance of a complete biometric system is determined in a specific application environment with a specific population.

The first one is by far the most common and often most feasible [7]. Since this type of evaluation is carried out using already saved samples, the results are reproducible at any moment with any system, and the evaluation is not overly time-consuming nor complex.

Nevertheless, the main disadvantage of technology evaluations is that they don't necessarily represent the exact conditions of the system that will eventually be used. Therefore it could be beneficial to collect a dedicated set of samples trying to mimic the specifics of the target application when carrying out an evaluation.

1.2 AI-driven Vulnerabilities

In the last several years biometric identifiers have become crucial means for authentication and reliability, with continual daily usage (smartphones, laptops, banking operations and transactions, job establishment access, etc.). Therefore, the need for (person-tailored) identification methods have widely and commonly spread. Tracking devices, surveillance cameras, and biomedical smart apps constantly collect, use, and share people's vulnerable information.

Identity inference (I.I.) represents a group of strategies allowing us to understand and connect (infer) the profile of individuals (identity), using different pieces of information. In relation to cybersecurity and privacy issues, this capability becomes highly critical when related to multimedia content. Pictures, videos, and audio are easily shared through the internet, especially due to the diffusion of social networks, and are constantly exploited by investigators and malicious actors to extract specific knowledge about targeted profiles.

Training and tuning of big data algorithms requires, as the name suggest, huge quantity of data samples. This implies that softwares trained to optimize acquisition and processing of biometric information (smartphones sensors, camera quality adjustments, etc.) are deeply data hungry and lead to a significant increase of privacy issues when insufficient samples are provided. Indeed, when machine learning models needs to deal with data scarcity, they often introduce either worsened quality results or specific predictable patterns, both related to the **overfitting** problem.

Such tendency can be exploited by an attacker to gain relevant information about the model and its the training set. Therefore, when the training set concerns biometric data, such vulnerabilities directly translate in privacy inference attacks.

In the next chapters, an extended analysis of these issues is going to be carried out, providing a complete assessment of the risks of such inferences.

1. Firstly, the second chapter will discuss the state of the art in the generation of artificial fingerprints and the related security concerns, with a particular focus on the adversarial assessment of data re-identification and the related privacy leaking issues.

2. Starting in the third chapter with the introduction of a general, standard dataset, the information about the data available in this research will be discussed and the processing and data transformations required to provide the best scalability and generality possible will be shown.

3. The fourth chapter will then explain the details of the employed models from their theoretical introduction to the custom engineering choice specified for the following tasks.

4. In the fifth chapter, several attacking strategies are going to be introduced, as well as how different models were targeted by different means, and the related malicious objectives.
5. Eventually, the last chapter will analyze the results of these attacks and finally suggest some protective mechanisms to mitigate the discussed vulnerabilities.

2
Related Works

2.1 Synthetic Fingerprint Generation

As introduced earlier, there are many reasons why the artificial generation of biometric data spread so much. These kinds of samples, and therefore the related datasets, are not trivial to recover and share, due to many privacy regulations and different legal systems incompatibilities. Training, validating, and testing software and models based upon such data has become an increasingly harder task since state-of-the-art systems require huge amounts of data, significant data preprocessing, and many other requirements about the reliability of the samples, their distribution, and their characteristics.

As a matter of fact, in the past, institutions such as NIST have even taken down some of their previously public fingerprint datasets (NIST SD27, NIST SD14, and NIST SD4) due to stringent privacy regulations. [8]

Employing synthetic data may allow a free manipulation of biometric data by AI-driven technologies. The main issues are obviously related to the need for reliable and representative artificial samples (otherwise the quality of the new data won't be good enough or may introduce unrealistic features or distributions in the datasets) and must be secure (it needs to be safe against privacy attacks, such as membership inference, which may reveal private information of the people whose samples trained such models).

Several methods have been suggested for the generation of synthetic fingerprints. These tech-

niques can be broadly divided into:

1. **Model-based** strategies

2. **Learning-based** approaches

Although learning-based methods demand substantial volumes of training data, they excel in generating fingerprints that closely resemble reality, surpassing the capabilities of model-based counterparts [8].

The limitations of existing model-based approaches encompass the following:

1. **Pre-assumption of independent ridge orientations and minutiae models**. Consequently, minutiae distributions might be generated without a valid ridge orientation field.

2. **Assumption of a fixed fingerprint ridge-width**, whereas actual fingerprints exhibit diverse ridge widths. In these cases, they may report really high accuracies in discriminating real fingerprints from synthetic ones using only a couple of ridge spacing characteristics, not modeling realistic distributions.

3. **Use of masterprints in the ridge structure model**. Owing to Gabor filtering and the AM/FM models, the generated masterprints have inconsistent ridge flow patterns, leading to implausible fingerprint images.

4. **Generation of unrealistic minutiae configurations**. Often minutiae models do not account for local minutiae arrangements. Notably, researchers in [30] demonstrated the ability to completely differentiate between genuine and synthetic [21] fingerprints based solely on minutiae configurations.

   None of these techniques are known to produce fingerprint datasets comparable in size to extensive operational datasets.

2.1.1 GANs

Generative Adversarial Networks (GANs) [9] emerged as a breakthrough in the field of artificial intelligence and machine learning, offering innovative approaches to the generation of realistic diverse data.

These gained widespread attention due to their remarkable capability to synthesize highly realistic data that can get to the point of being virtually indistinguishable with respect to authentic samples. GANs revolutionized the field of generative modeling by adopting a novel adversarial training paradigm that employs jointly a generator and a discriminator network to engage in a min-max game [9]. This adversarial process leads to the generation of data that captures intricate underlying patterns and distributions, making GANs invaluable for various applications, including image synthesis, style transfer, data augmentation, and many others.

The core architecture of a GAN consists of two neural networks: the generator and the discriminator. The generator network takes noise from a random distribution as input, generating fake data that aims to mimic the real distribution. Meanwhile, the discriminator network serves as a binary classifier, distinguishing between genuine data and data produced by the generator. Through a competitive feedback loop, these networks engage in an adversarial exchange, where the generator strives to improve its output to deceive the discriminator, while the discriminator aims to enhance its ability to differentiate between real and generated samples.

The adversarial training process in GANs can be conceptualized as a minmax game between the generator and the discriminator. The generator aims to minimize the discriminator's ability to correctly classify its outputs as fake, while the discriminator seeks to maximize its accuracy in distinguishing real data from generated data. This dynamic balance between the two networks results in a Nash equilibrium, where (if successfully trained) the generator produces data that closely approximates the real data distribution.

Several loss functions can be employed to train GANs effectively, the most common are the following:

1. **Binary Cross-Entropy (BCE) Loss**

   The BCE loss is used in GANs for binary classification. For real samples, the discriminator loss (L_D) and the generator loss (L_G) are defined as:

   For real samples:

   $$L_D = -\log(D(x))$$

   For fake samples:

   $$L_G = -\log(1 - D(G(z)))$$

2. **Least Squares Loss (LSGAN)**

   LSGAN uses mean squared error (MSE) loss for both the discriminator and generator. The loss functions are:

   For real samples:

$$L_D = (D(x) - 1)^2$$

   For fake samples:

$$L_G = (D(G(z)))^2$$

3. **Wasserstein Loss (WGAN)**

   WGAN uses the Wasserstein distance and introduces a critic. The critic loss (L_{critic}) and the generator loss (L_G) are:

   For the critic:

$$L_{\text{critic}} = -\left(\frac{1}{m}\sum_{i=1}^{m} D(x_i) - \frac{1}{m}\sum_{i=1}^{m} D(G(z_i))\right)$$

   For the generator:

$$L_G = -\frac{1}{m}\sum_{i=1}^{m} D(G(z_i))$$

4. **Hinge Loss (HINGE GAN)**

   Hinge loss is another modification of the GAN loss using the hinge function. The discriminator loss (L_D) and generator loss (L_G) are defined as:

   For the discriminator:

$$L_D = \max(0, 1 - D(x)) + \max(0, 1 + D(G(z)))$$

   For the generator:

$$L_G = -D(G(z))$$

5. **Feature Matching**

   Finally, Feature matching is an auxiliary loss function that encourages feature alignment. The feature matching loss is defined as follows:

$$L_{\text{FM}} = \|\mathbb{E}[f(x)] - \mathbb{E}[f(G(z))]\|^2$$

The generator's loss is typically defined by a measure of dissimilarity between the discriminator's predictions for generated samples and the genuine data. Commonly used loss functions

include the Binary Cross-Entropy Loss (BCE) and the Wasserstein distance. The discriminator's loss, on the other hand, is designed to maximize its accuracy in classifying real and fake samples. The overall training objective involves alternating between updating both networks to achieve stable convergence.

Since their inception, GANs witnessed significant advancements. Progressive GANs introduced by Karras et al. employ a multi-stage training process to generate high-resolution images progressively [10]. CycleGANs, proposed by Zhu et al., enable unpaired image-to-image translation through a cycle-consistency loss [11]. StyleGAN and StyleGAN2, pioneered by Karras et al., allow for fine-grained control over generated images' style and attributes [12]. These developments underscore GANs' adaptability and potential across several different fields.

Concerning the learning-based generation of fingerprints, GANs have been widely used in the literature as a tool to generate new fingerprints with high-quality minutiae [13, 8, 14, 15] or to reconstruct partially degraded ones [16, 17].

While several of these methodologies enhance the authenticity of fingerprint images through substantial training datasets of real fingerprints, they overlook the distinctiveness and individuality of the generated fingerprints. In the absence of additional guidance, a generator might repeatedly fabricate a limited subset of fingerprints, corresponding to only a few identities. This phenomenon was quantitatively proved in [18], where search accuracy against real fingerprint galleries was inferior to that against synthetic fingerprint galleries.

In [8], they extend the capabilities of cutting-edge learning-based synthesis algorithms to capitalize on their enhanced realism. They also incorporate supplementary guidance (identity loss) to incentivize the production of fingerprints representing distinct identities. Following enhancements to both the distinctiveness and realism of synthetic fingerprints, they are able to generate a dataset of 100 million fingerprints (the largest gallery documented in existing literature) for an extensive fingerprint search assessment. The authors use I-WGAN [18] together with an identity loss that allows generating fingerprints from the same user.

Figure 2.1: 100mil Samples

In [13] they propose a synthetic fingerprint generation framework based on deep convolutional generative adversarial networks (DC-GAN) [19], which is able to generate realistic fingerprint images, which are hard to be distinguished from real ones. They additionally provide a connectivity regularization loss to generate faithful fingerprint images with a suitable term to the GAN loss function, imposing generated fingerprints to improve line connection. Empirically, they achieve such an outcome by adding the total variation [21] of the generated images to the loss function.

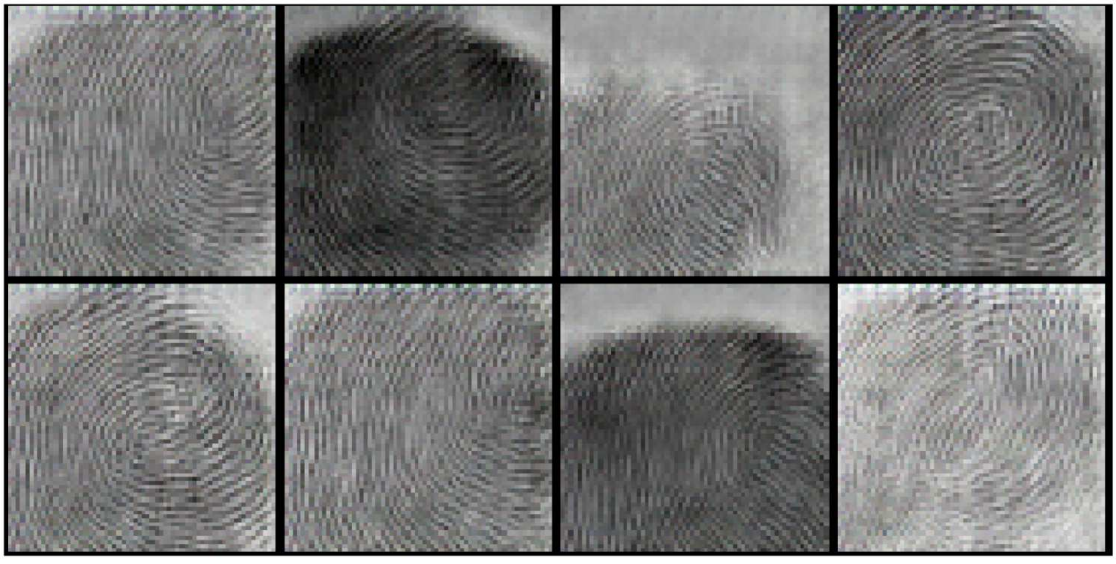


Figure 2.2: Finger-gan Samples

In [14], fingerprints are generated in a StyleGAN2 and CycleGAN based dual generative adversarial networks (GAN) system:

1. A **StyleGAN-2** [20] based model is used to generates distinct fingerprint skeletons from preprocessed data.

2. Then **CycleGAN** [21] is applied to perform style transfer and transform these skeletons into actual realistic images.

Figure 2.3: Fingan Samples

This model can generate high-quality 256 by 256 fingerprints that can be turned into a variety of fingerprint styles. Synthesized fingerprints from this model also retain features of real fingerprints that can be used in the related search system. Experimentation of the model includes visual image quality, quantitative image quality, distinctiveness test, and human perception test.

Lastly, the approach in [15] also uses Style-GAN2 to generate samples, but the input signal is created by an encoder that allows for selecting the position of the minutiae thus generating fingerprints with consistent identities.

They propose a novel fingerprint synthesis and reconstruction framework based on the Style-Gan2 architecture joined with the derivation of a computational approach, to modify the attributes of the generated fingerprint while preserving their identity. This allows for the simultaneous synthesis of multiple different fingerprint images per finger.

They are therefore able to introduce the SynFing synthetic fingerprints dataset consisting of a hundred thousand image pairs, each pair corresponding to the very same identity. Their proposed framework focused on both fingerprint synthesis and reconstruction. The authors remark on the realism of the generated fingerprints, both visually and in terms of their ability to spoof fingerprint-based verification systems.

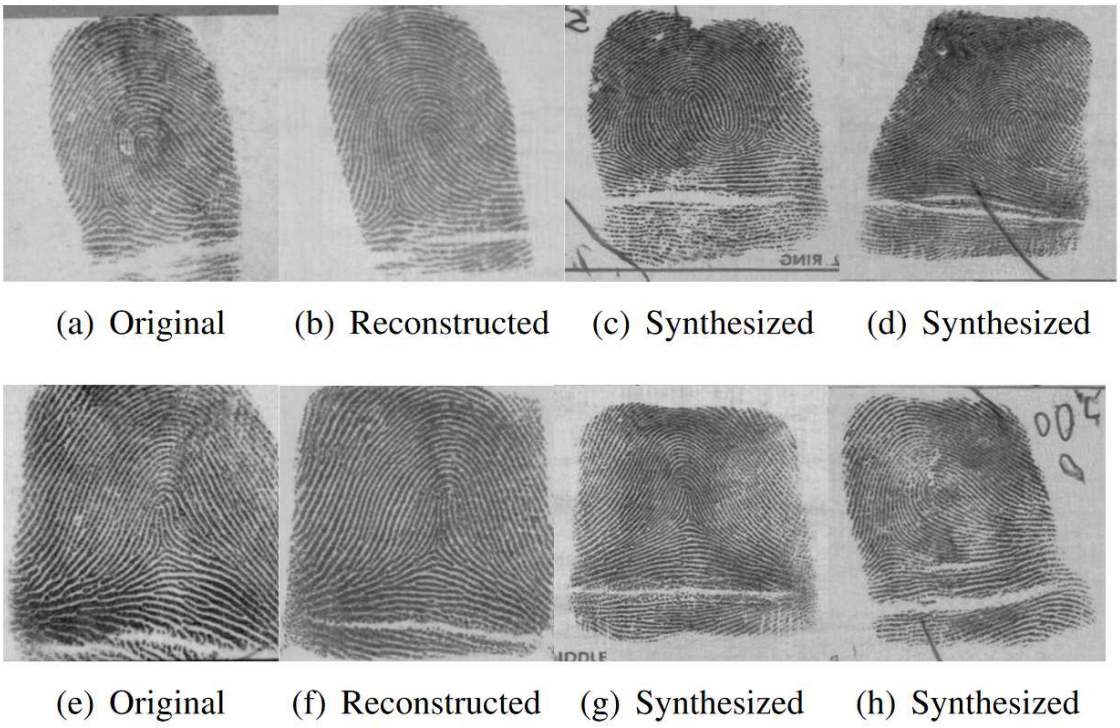


Figure 2.4: Generative Process

2.2 Membership Inference

Lately, industry leaders like Google, Microsoft, and Amazon introduced the provision of APIs to afford customers convenient access for integrating machine learning functionalities into their applications. Businesses can leverage the capabilities of this paradigm, referred to as Machine Learning as a Service (MLaaS) in frameworks developed to externalize many complex processes, such as training classifiers, analyzing predictions, and carrying out clustering tasks. Moreover, they can enable external parties to query models trained on their proprietary data, often involving a corresponding expense [22].

Nevertheless, the potential exposure of training data utilized to develop these models to malicious actors introduces a critical issue.

This scenario can lead to severe consequences due to the potential exposure of sensitive information. Notably, organizations have limited influence over the architectural aspects of the models and the specifics of training parameters within the MLaaS platform. This lack of control raises the possibility of stimulated overfitting, a situation where a model's effectiveness purposely diminishes outside the scope of its training data to gather relevant information. Consequently, attackers can exploit this vulnerability to query and reconstruct training data [23].

Given a set of data points, and a machine learning model, the Membership Inference Attack (MIA) consists in guessing which of the samples were used to train the algorithm i.e. are members of the training set.

Generally, the MIA is said to be

1. **White-box** if the attacker has access to the weights of the model
2. **Black-box** if he can only analyze the predictions. This can be further subdivided by incrementally removing information (e.g. when attacking a model that only returns the top-n predicted classes w.r.t. a model that returns the probabilities for each class.).

One of the first works to introduce MIAs is [24] where "machine learning as a service" classification models were targeted.

In this investigation, the researchers quantitatively explore the phenomenon of information leakage from machine learning models, specifically pertaining to the individual data records used for their training. The primary focus of the study centers on the analysis of the basic membership inference attack. This attack involves the task of determining, with only black-box access to a model, whether a given data record was included in the model's training dataset.

To execute the membership inference attack on a chosen target model, they adopt an adversarial approach by creating their own inference model, trained to detect disparities in the predictions made by the target model. Such processes are assessed between inputs that the target model was trained on and those that it was not.

The evaluation of their inference techniques is conducted empirically. The researchers specifically consider classification models that have been trained by commercial "machine learning as a service" providers, such as Google and Amazon. The study incorporates real-world datasets and practical classification tasks, among which is a hospital discharge dataset that holds privacy-sensitive membership information. The outcomes of the investigation reveal that these models are indeed susceptible to membership inference attacks.

Additionally, the researchers delve into an analysis of the factors that contribute to this form of information leakage. They proceed to examine strategies aimed at mitigating the identified risks associated with membership inference attacks.

This is a good example of a solely black box setting since even the Machine Learning (ML) algorithm is unknown to the attacker. The authors' own strategy was to train multiple shadow models (one for each class) to mimic the behavior of the attacked one. Afterward, they additionally train a classifier to assess the membership of the samples using the shadow models as a proxy for the attacked one allowing the classifier to be later used to attack the actual black box cloud API.

Following this idea, many works have proposed increasingly reliable and effective attacks and counter-measures [25, 26, 27]. In [26] the correctness of the prediction and the loss value are used as discriminative factors. This work examines the impact of overfitting and its influence on an attacker's ability to gain insights into the training data occurring via attacks involving membership inference or attribute inference. Through a combination of formal analysis and empirical investigations, the paper establishes a distinct correlation between these factors and the ensuing privacy risks associated with various well-known machine learning algorithms.

The study reveals that overfitting alone can provide attackers with the means to conduct membership inference attacks. Moreover, in cases where the target attribute adheres to specific influence conditions, it can also facilitate attribute inference attacks. Intriguingly, the formal analysis unveils that overfitting isn't a prerequisite for such attacks, hinting at the presence of other contributing factors that warrant exploration.

Lastly, the paper delves into the interrelation between membership inference and attribute inference, showcasing deep connections between the two, with such connections pave the way

for innovative attack strategies.

In [27] the techniques allow the removal of some stright assumptions made in the aforementioned work. As the authors state, others include the utilization of multiple "shadow models," possessing knowledge about the structure of the target models with additional access to huge fragments of the original datasets, mirroring the distribution of the target model's training data. In their study, they purposely relax these critical assumptions. By doing so, they try to reveal significantly broader applicability with lower cost and complexities of implementation, thereby presenting a more significant risk than previously perceived.

This research presents an extensive examination of the evolving inference threats, surpassing previous endeavors. They accomplish this by employing eight diverse datasets, which collectively highlight the potential of the proposed attacks across various domains. Furthermore, this paper introduces the first effective defense mechanisms against this expanded category of membership inference attacks. These mechanisms manage to uphold the utility of the machine learning model at a high level, addressing the pressing need for protective measures.

Additionally, many other MIAs have been designed to work on different types of models (e.g.generative ones). For example in [22] the authors propose the first MIA attack on GANs. In the white-box scenario, the authors use the discriminator prediction confidence as the discriminative factor between members and non-members, while, in the black-box scenario, they train a GAN to mimic the target model and then they exploit the newly trained discriminator to perform the white-box attack.

In this study, the researchers explore the viability of membership inference attacks directed at generative models as is important to note that membership inference on generative models tends to be more intricate compared to discriminative models (as highlighted in [24]).

Discriminative models are designed to predict labels based on input data, allowing attackers to leverage the model's prediction confidence for the attack. However, generative models lack such signals, posing challenges in both detecting overfitting and deducing membership information. The absence of this signal makes it difficult to achieve both tasks effectively within the realm of generative models.

The central question therefore involves determining whether an attacker, armed with access to a generative model and an individual data record, can discern if the specific record was employed during the model's training phase.

Still about generative models inference, in [28] the authors use Monte Carlo integration to

approximate the probability that the item is a member.

The paper introduces two advanced information leakage attacks that surpass previous efforts in the realm of membership inference against generative models. The first attack stands out by enabling membership inference without imposing assumptions about the specific type of generative model used. Notably, unlike past evaluation metrics like Kernel Density Estimation, this approach exclusively focuses on samples generated by the model that closely resemble training data records.

The second attack is tailored specifically for Variational Autoencoders (VAEs), demonstrating notably high accuracy in membership inference. Moreover, a shift in perspective is advocated – moving beyond single-record membership inference adversaries to incorporate regulatory actors performing set membership inference. This extension allows for the identification of specific datasets employed for training, offering a broader context for assessment.

The efficacy of these attacks is evaluated across two prevalent generative model architectures, namely Generative Adversarial Networks (GANs) and Variational Autoencoders (VAEs), using standard image datasets. The results underscore the superior success rates achieved by these two attacks compared to prior endeavors, even under comparatively mild assumptions. These findings suggest potential applications, particularly when combined with formalized membership inference attack strategies. This combination could prove beneficial in maintaining data privacy standards and automating the evaluation of model quality within machine learning as service setups.

Notably, this research encourages the utilization of Generative Adversarial Networks (GANs) due to their demonstrated resilience against information leakage attacks, coupled with their ability to produce intricate and detailed samples.

Expanding MIAs, some researchers have shown that it is not only possible to assess the membership of a sample in the training set but also the user depicted in the photo himself. In particular, in [29] it is shown that it is possible to get to a point of understanding if photos of a user were used to train a metric embedding learned system by looking at how tightly some images of the person are clustered by the network. Additionally, these images don't even need to have been used as training samples, showing that the model is actually memorizing the user identity in its weights.

Additional works concerning other multimedia content apart from images have also shown that IIAs can be carried out to assess the authorship of samples used to train language models [30] and even to audio signals, by finding out if someone's voice was used to train some voice

services [31].

2.3 Codecs

All the purposes of fingerprint (and in general, biometric) employment require minimum quality levels mandated by the systems and the institutions that develop the frameworks for the related technologies. This means that the manipulation of such data needs to oblige specific and accurate requirements. This can cause a bottleneck in machine-learning-related implementations, as the quantity of data and its quality are often restricted by their encoding.

Data compression is the process of reducing the size of data to save storage space or transmission bandwidth while preserving its essential information. Traditional compression techniques, such as Huffman coding [32] and Lempel-Ziv-Welch (LZW) algorithms [33], have been widely used to achieve this goal. Nevertheless, as introduced in [34], several machine learning models, as Variational Autoencoders (VAEs), offer an innovative approach to data compression. These models leverage their capacity to learn patterns and representations from training data. In [35] the idea of exploiting neural networks for end-to-end optimized image compression has been presented, where both encoding and decoding processes are jointly optimized. This approach enhances compression efficiency and demonstrates the adaptability of machine learning techniques. Lossy image compression has always been discussed using several methods, including recently those based on machine learning [36]. It highlights the progress made in leveraging machine learning for data compression and its relevance in modern applications.

Machine learning-based data compression involves training models on a dataset containing representative samples of the data to be compressed. As explained in "Data Compression and Machine Learning: A Tutorial Overview" by Hinton and Salakhutdinov (2006), these models learn to capture and represent the inherent patterns and redundancies within the data. Once trained, such models can then be used to compress new data by encoding it into a more compact representation. When decompression is required, the model is used to decode the compact representation and reconstruct the original data. This process aligns with the minimum description length principle, as discussed in [37], where the model's learned representation serves as a succinct description of the original data.

These as some of the reasons why machine learning models offer a promising avenue for data compression by learning to efficiently represent and reconstruct data, leveraging their ability to

capture intricate patterns and structures. As the field of machine learning continues its advance, further innovations in data compression techniques can be expected, combining the strengths of both traditional methods and modern machine learning approaches.

Even though they are being developed in the state of the art of the field, additional strategies and models introduce additional vulnerabilities, especially in relation to privacy threats, and as later is going to be shown, require further attention and assessments.

During the experiments introduced in this thesis, the work was built upon the TensorFlow (TF) implementation of the TensorFlowCompression (TFC) [38]. This library has been developed to build own ML models with end-to-end optimized data compression built in. It's useful to find storage-efficient representations of multiple data (images, features, examples, etc.) while only sacrificing a small fraction of the model performance.

2.3.1 Mean and scale hyperprior model

During this work, the **mean and scale hyperprior** model [39] has been employed, retrained, and tuned. It is described as an end-to-end trainable codec for image compression based on variational autoencoders. The model incorporates a hyperprior to effectively capture spatial dependencies in the latent representation. This solution is being adopted in the most advanced learned codecs using artificial neural networks (ANNs). Unlike existing autoencoder compression methods, this model trains a complex prior jointly with the underlying autoencoder. This model leads to state-of-the-art [39] image compression when measuring visual quality using the MS-SSIM index, and yields remarkable rate–distortion performance when evaluated using a more traditional metric based on squared error (PSNR).

3 Dataset

3.1 Biometric Data

When it comes to evaluating the performance of biometric systems, such as fingerprint recognition, it's important to recognize that errors in these evaluations are not evenly distributed among individuals. Some individuals possess biometric features that are inherently challenging to capture accurately. For instance, factors like:

1. worn friction ridges due to manual labor,
2. dry skin caused by cold weather,
3. skin diseases such as dermatitis or psoriasis.

can all contribute to issues in obtaining reliable biometric data.

Furthermore, there are cases where individuals present fingerprints with limited unique characteristics. This could be due to too few minutiae in the ridge pattern or an inability to provide a proper or complete portion of their fingers, often only revealing the top part with fewer characteristic structures.

Indeed, in the pursuit of calculating biometric performance, one approach to consider is selectively removing the worst samples from a database. This can lead to an improvement in

the overall performance of the system. However, it's essential to note that the resulting performance metrics are often derived from a subset of the database, which may not accurately represent the system's performance across all individuals.

In this context, it's also worth mentioning that metrics already introduced, like the Failure to Acquire Rate (FAR) and the methods used to filter out low-quality samples play a significant role in assessing system reliability. Ignoring these aspects can render performance evaluations less informative and relevant.

These consideration are the reason why models for fingerprints require huge, variable datasets, built upon the contribution of several subjects and acquired after accurate, thoughtful and systematic planning.

3.1.1 Casia

Portions of the research in this thesis use the CASIA-FingerprintV5, collected by the Chinese Academy of Sciences' Institute of Automation (CASIA) [40].

The CASIA Fingerprint Dataset Version 5.0 (or CASIA-FingerprintV5) contains 20.000 fingerprint images of 500 subjects. The fingerprint images of CASIA-FingerprintV5 were captured using the URU4000 fingerprint sensor in one single session. The volunteers that allowed the acquisitions for the CASIA-FingerprintV5 include graduate students, workers, and other collaborators. Each volunteer contributed 40 fingerprint images per finger.

They were asked to rotate their fingers with different levels of pressure to generate significant intra-class variations.

All the fingerprint images in the original dataset are 8-bit gray-level BMP files and the image resolution is 328*356 and show significant differences in the quality of the different acquisitions (some of them are partial, some display cuts and bruises, others introduce a lot of dirt, some are blurry).

The first preprocessing step carried out during this work was about converting the images into PNGs, allowing to display and manipulate better the data samples better. That enabled an improvement regarding the ease of monitoring the quality and the additional usability of the computer vision Python libraries. During the research, particular attention was paid to keeping the images consistent within all the transformations that the experiments introduced, from the processing to the training and the evaluations (e.g., synthetic generation, compression, decompression). Please notice that this constant and accurate compatibility assessment was provided whether they were the original data samples from the CASIA or the images we

artificially generated through the use of the GANs that will be introduced later on.

Training the attacking strategies and the different models required a specific partitioning of the dataset.

The first division regarded the processed CASIA as 16 thousand available samples and 4 thousand "leftovers".

The reason is that such a split was used to train a Codec on all the 16k images and a 2k subset of these, in order to evaluate the potential and vulnerabilities of these different models.

Additionally, From the 16k images, a second splitting strategy was employed as well, by taking exactly 3 acquisitions per finger. These allowed to create both a training query for the Gans (starting from 9600 samples) and a query specifically meant for identity inference (starting from 5400 samples).

The 4k images that were left untouched are needed as validation and testing samples for the different attack strategies and defense assessments.

(a) 011_L1_0 **(b)** 011_L1_1

(c) 011_L1_2 **(d)** 011_L1_3

(e) 011_L1_4

Figure 3.1: 5 impressions of the same fingerprint: The left index of the 11th person.

4 Models

4.1 GANs

Different Convolutional Gans (C-GANs) were trained in the experimental setup, using the impressions from the CASIA-FingerprintV5 dataset. These networks represented the backbone both for the simulations of industrial APIs (the image-generating services we may want to provide) and the malicious attacking networks, trying to uncover the highest information available by the victim networks.

Convolutional Generative Adversarial Networks (cGANs) [9] represent a relevant advancement within the domain of generative modeling, encapsulating the enforcement of convolutional neural networks (CNNs) with the adversarial framework. Such fusion develops generative modeling capabilities providing improved intricate and authentic data samples.

cGANs employ deep convolutional architectures in both the generator and discriminator components. The generator, designed as a deconvolutional neural network, effectively traverses the inverse path of convolutional layers, progressively crafting data samples with increasing complexity. [19].

The convolutional architecture employed in cGANs effectively captures spatial hierarchies and correlations present in the data, and thereby holds a distinct advantage over fully connected architectures [41]. This spatial awareness and local feature extraction capability endow cGANs with the prowess to generate coherent and realistic data samples, focusing on preserving global structures and fine-grained details. The **generator**, implemented using transposed convolutional layers, progressively transforms random noise into data samples that adhere to the training data distribution, whilst the task of the related **discriminator** is to underpin the interpretability of convolutional layers [42] in unraveling the visual elements detected by the network. The implementations that allowed for the best quality of fingerprint was evaluated with respect to the Inception Score (IS) [43]. The former is shown in figure 4.1, whilst the latter can be observed in figure 4.2.

```python
def make_generator_model():
    model = tf.keras.Sequential(name='generator')

    model.add(Input(shape=(256,)))

    model.add(Dense(64*89*82))
    model.add(LeakyReLU())

    model.add(Reshape((89,82,64)))

    model.add(Conv2D(64,4,padding="same"))
    model.add(LeakyReLU())

    model.add(Conv2DTranspose(64,4,padding="same",strides=2))
    model.add(LeakyReLU())

    model.add(Conv2D(64,4,padding="same"))
    model.add(LeakyReLU())

    model.add(Conv2DTranspose(64,4,padding="same",strides=2))
    model.add(LeakyReLU())

    model.add(Conv2D(64,3,padding="same"))
    model.add(LeakyReLU())

    model.add(Conv2D(64,3,padding="same"))
    model.add(LeakyReLU())

    model.add(Conv2D(1,7,padding="same",activation="tanh"))

    return model

generator = make_generator_model()
generator.build(input_shape=(1, 256))


noise = tf.random.normal([1, 256])
generated_image = generator(noise, training=False)
```

Figure 4.1: Generator

```python
def make_discriminator_model():
    model = tf.keras.Sequential(name='discriminator')
    model.add(Input(shape=(356, 328, 1)))

    # Increase the depth of the model
    model.add(Conv2D(64, 3, padding='same'))
    model.add(BatchNormalization())
    model.add(LeakyReLU())

    model.add(Conv2D(128, 3, strides=2, padding='same'))
    model.add(BatchNormalization())
    model.add(LeakyReLU())

    model.add(Conv2D(128, 3, strides=2, padding='same'))
    model.add(BatchNormalization())
    model.add(LeakyReLU())

    model.add(Conv2D(256, 3, strides=2, padding='same'))
    model.add(BatchNormalization())
    model.add(LeakyReLU())

    model.add(Conv2D(512, 3, strides=2, padding='same'))
    model.add(BatchNormalization())
    model.add(LeakyReLU())

    # Use larger kernel sizes and increase the width of the model
    model.add(Conv2D(1024, 4, strides=1, padding='valid'))
    model.add(BatchNormalization())
    model.add(LeakyReLU())

    # Use dropout to prevent overfitting
    model.add(Dropout(0.4))

    model.add(Flatten())
    model.add(Dense(1, activation='sigmoid'))

    return model

discriminator = make_discriminator_model()
discriminator.build(input_shape=(1, 356, 328, 1))


decision = discriminator(generated_image, training=False)
print(decision)
```

Figure 4.2: Discriminator

4.1.1 Inception Score

The IS seeks to capture two properties of a collection of generated images:

1. **Image Quality.** Evaluating if the images look visually appealing with respect to the objects intended to be generated. Therefore mages that contain meaningful objects should have a conditional label distribution $p(y|x)$ with low entropy.

2. **Image Diversity.** Evaluating if the generated images are representing a wide enough range of required objects, so the marginal $\int p(y|x = G(z))dz$ should have high entropy.

It leverages a pre-trained deep convolutional neural network (CNN) called InceptionV_3, which was originally designed for image classification tasks.

Combining all the requirements, the score proposed in [43] is:

$$\exp(\mathbf{E}_x[KL(p(y|x)||p(y)])$$

, where:

1. KL is the **Kullback–Leibler divergence** (also known as relative entropy), denoted by $D_{KL}(P||Q)$, is a type of statistical distance which measures how much the probability distribution P is different from a second, reference probability distribution, Q.

2. The results are modeled through exponentiation to allow easier comparisons between values.

4.1.2 Victim

The attacked architecture is defined in Figure 4.3.

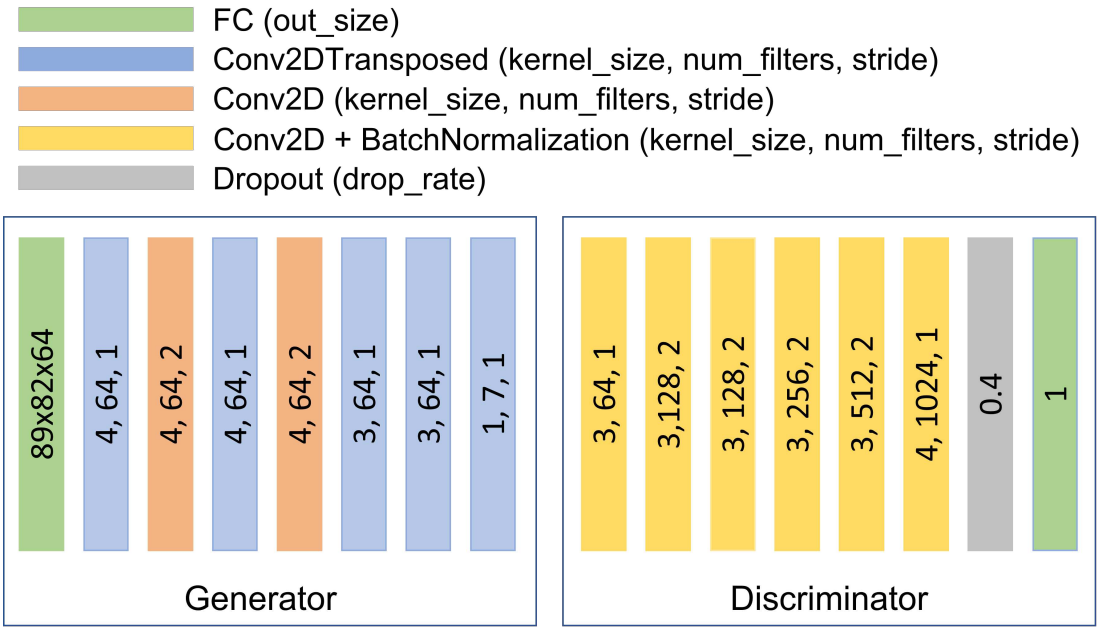


Figure 4.3: General Network Scheme

LeakyReLU was used as an activation function making exceptions for

1. the last layer of the generator where **TANH** was chosen:

$$f(x) = \tanh(x) = \frac{e^{2x} - 1}{e^{2x} + 1}$$

2. the last layer of the discriminator where **Sigmoid** was employed:

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

(these implementation choices are rather generic and shared by most of the GAN architectures in the literature).

Using the hyperbolic tangent (TANH) activation function as the last layer in the **generator** of a GAN is a common choice for several reasons:

1. **Output Range**: TANH outputs values $\in$ [-1, 1], which is often desired in image generation tasks. Many image datasets normalize pixel values to the [-1, 1] range, so this function ensures that the generator's output matches such a range, making it easier to directly compare it with real images.

2. **Symmetry and Zero centering**: TANH being an odd function implies it's symmetric around the origin. Such symmetry can help the generator learn to produce images with more balanced features, opposed as favoring too much either positive or negative values.This can also help in generating images with a better contrast.

3. **Smoothness**: TANH is a smooth and differentiable function. This differentiability is crucial for training GANs using techniques like backpropagation and gradient descent. Smoothness also helps in generating images with smooth transitions between different features, resulting in visually pleasing outputs.

On the other hand, the sigmoid activation function as the last layer in the **discriminator** of these networks is common choice because:

1. **Binary Classification**: The discriminator's primary task is binary classification: distinguishing between real and fake data. The sigmoid function maps its input to the range [0, 1], which is suitable for modeling binary classification problems. The output of the sigmoid can be interpreted as the probability that the input data is real (close to 1) or fake (close to 0).

2. **Probability Interpretation**: with this function, the output of the discriminator can be interpreted as a probability score. Such score makes it easier to assess the confidence of the discriminator in its classification decision.

3. **Smoothness**: as TANH, the sigmoid function is a smooth and differentiable function as well. This properties are essential for gradient-based optimization methods, allowing the discriminator to be trained effectively using many useful common methods, as backpropagation and stochastic gradient descent.

4. **Compatibility with common Loss Functions**: The sigmoid output naturally fits with the binary cross-entropy loss that was mainly used in this work, which is also very a common choice for training binary classifiers. As stated, such a loss function properly quantifies the difference between predicted probabilities and the true labels, making it well-suited for training the discriminator in a GAN setup.

5. **Consistency with the Generator's Output**: Using a sigmoid output, remarkable consistency is obatined with the choice of TANH activation in the generator's output layer. Indeed, as TANH activation maps the generator's output to the [-1, 1] range, the sigmoid activation in the discriminator maps it back to the [0, 1] probability space.

In order to assess the robustness of the trained models against MIAs, the considered C-GANs were retrained using all the dataset splits specified in 4.1. Nevertheless, only 3 acquisitions per finger were kept in the training dataset in all the data splits, while the others are added to the IIA query dataset. These choices will be thoroughly explained in the next chapter as they are intentionally planned to provide the best environment for the attacks.

Name	Real Images	D1	D2	D3	D4
Samples / Users / F.prints	2000/50/5	9600/400/3	4800/200/3	2400/100/3	1200/50/3
IS Mean	1639.2	1620.13	1563.79	1570.89	1571.28
IS STD	7.92	11.95	15.99	18.66	10.29

Table 4.1: Datasets splits and inception score metrics on the trained GANs for each of them

(a) GAN 9600 **(b)** GAN 4800

(c) GAN 2400 **(d)** GAN 1200

Figure 4.4: Generated fingerprint from GANs trained with variable numbers of samples

4.1.3 Attacker

The attacking GANs share similar architectures with respect to the victim. This is due to the similarity of the best implementation available using the researched framework and the fact that, even though the dataset size is different in each experiment, the data distribution and the types of layers used in the networks were the same in all experiments. This has obviously led to the best models (selected using Inception score [43] as the discriminating factor) with similar hyperparameter and thus similar network architectures. Such a model was trained using the **Adam** optimizer [44] and the standard Binary Cross-Entropy (BCE) losses.

However, it was noticed that the discriminator was producing a very skewed prediction distribution making it hard to distinguish between samples that had high and low confidence. To address this issue, a label smoothing with a **smoothing factor** equal to 0.2 was implemented as displayed in figure 4.2, which immediately provided more significant predictions. The latter is an effective regularization tool for deep neural networks, that generates soft labels by applying a weighted average between the uniform distribution and the hard label [45] (e.g. replacing the 0 and 1 targets for a classifier with smoothed values, like .9 or .1, was recently shown to reduce the vulnerability of neural networks to adversarial examples [43]) .

```
def discriminator_loss(real_output, fake_output, smooth_factor=0.2):

    real_loss = cross_entropy(tf.ones_like(real_output) * (1 - smooth_factor), real_output)
    fake_loss = cross_entropy(tf.zeros_like(fake_output), fake_output)
    total_loss = real_loss + fake_loss

    return total_loss

def generator_loss(fake_output):

    gen_loss = cross_entropy(tf.ones_like(fake_output), fake_output)

    return gen_loss
```

Figure 4.5: Attacking GANs losses

Concerning the "attackers", model selection was performed using the IS metric (as mentioned above) following the logic of what a black-box attacker would need to do in a real scenario (i.e., training his own networks until the generated images resembled the attacked ones to a certain degree. This way, the attacker can assume enough information could have been retained in order for the discriminator to provide valid inferences). As a result, diverse architectures with different parameters were created, depending on the dataset that was originally used to train the specific generator and discriminator (the various dataset splits). Also in this

case, the networks were obtained by stacking convolutional layers (with a structure similar to the one of the attacked GAN) whose number was varied in order to satisfy the IS criterion.

The advancement of cGANs urged scientific investigation to handle many training issues such as mode collapse, where the generator converges to produce a limited variety of samples. Strategies proposed in [43] have been adapted to cGANs, introducing stability to training and enhancing the diversity of generated samples. These issues challenges the attackers as much as the victims, as they need to eventually avoid the malicious discriminator models being biased on predictions/inferences related to a limited/specific amount of features, in order to provide effective attacks.

As discussed, overfitting poses a serious concern inherent to any machine learning model and can affect these cGANs as well. Indeed, in [46] the author provides insights into overfitting phenomena in neural networks, which may resonate with cGANs when generating data samples that excessively mimic the training data.

This is actually the reason for the different dataset splits. Several models trained with different tuning on a different, scalable amount of data allowed to highlight which properties led to more vulnerable models. Understanding which of these provide the highest degree of vulnerability is a main concern in the realm of model explainability.

4.2 Codec

As shown in the introduction, the Codec studied in this work was based on the mean and scale hyperprior (bmshj2018) model [39] and the dataset employed for its retraining was made of 16 thousand samples to provide the best generalization possible.

Figure 4.6 represents the network architecture of the hyperprior model. The left side shows an image autoencoder architecture, the right side corresponds to the autoencoder implementing the hyperprior. The analysis and synthesis transforms g_a and g_s use the classical symmetrical hourglass architecture often found in autoencoders.

The Q block performs quantization of the latents to obtain symbols that can be entropy coded, while AE and AD represent arithmetic encoder and arithmetic decoder, respectively [39].

The hyperprior architecture is a variational autoencoder used to estimate the symbol probabilities of the latents thus allowing for a better modelization of their probability distribution which in turn leads to a smaller compressed bitstream.

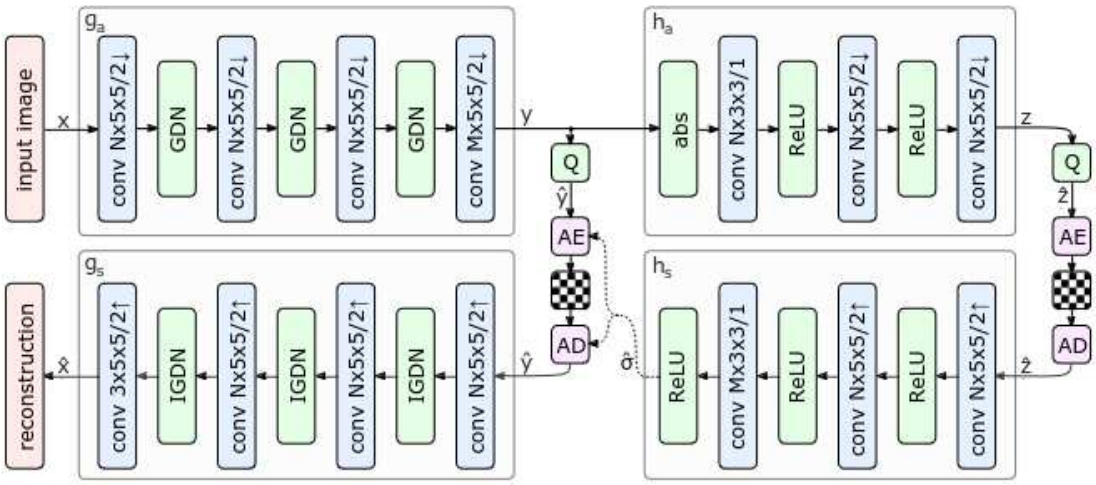


Figure 4.6: BMSHJ2018 hyperprior model

Most of the hyperparameters selected during the re-training have been left as proposed by the authors of TensorFlow compression (TFC), to try to replicate a faithful and general framework that could have been used by anyone (so as to avoid being too dependent on specific configurations that may not be translated to real-life scenarios).

```
# Loop over all the files in the input directory
for filename in os.listdir(original_dir):
    if filename.endswith('.png'):
        # Construct the output filename
        output_filename = os.path.splitext(filename)[0] + '.tfci'
        output_path = os.path.join(compression_dir, output_filename)

        # Run the compression command for each image file
        command = f"python Codec_1 compress {os.path.join(original_dir, filename)} {output_path}"
        os.system(command)

# Loop over all the files in the input directory
for filename in os.listdir(compression_dir):
    if filename.endswith('.tfci'):
        # Construct the output filename
        output_filename = os.path.splitext(filename)[0] + '.png'
        output_path = os.path.join(decompression_dir, output_filename)

        # Run the decompression command for each compressed file
        command = f"python Codec_1 decompress {os.path.join(compression_dir, filename)} {output_path}"
        os.system(command)
```

Figure 4.7: Codec custom command

Obviously considering the scope of this research and the differences between CASIA and the Datasets used to train the Mean and scale hyperprior (MSH) in the first place, some adjustments regarding the number of epochs and updates per epoch were still introduced to better fit the properties of the fingerprint dataset. In figure 4.7 the custom encoding strategy implemented from the TFC library can be observed.

The Convolutional Autoencoders were tested for quality of compression over 4 different lambda values (which controls the trade-off between bitrate and distortion that the model will be optimized for):

1. $\lambda = 0.01$
2. $\lambda = 0.001$
3. $\lambda = 0.003$
4. $\lambda = 0.0003$

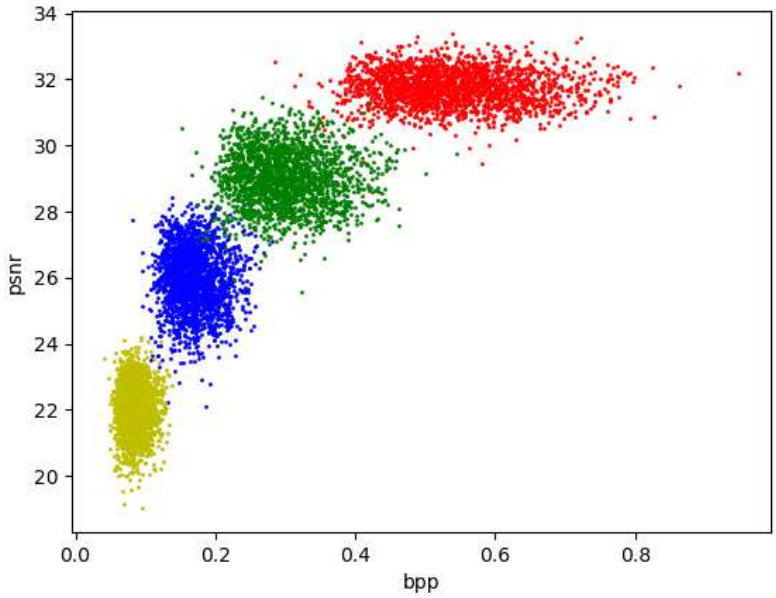
The retrained Codec reached proper convergence, providing consistent and stable compressions and decompressions at different lambda values. This is proved both by the side by side comparisons provided in figure 4.8 and in the following plots, that displays the relationships between the lambda values and the compression metrics. In figure 4.9 are shown the clusters representing the compression rate and the psnr (Peak Signal to Noise Ratio) for each image

tested, colored by the different λ value. In figure 4.10 are shown the averages of such clusters and their linear approximation.

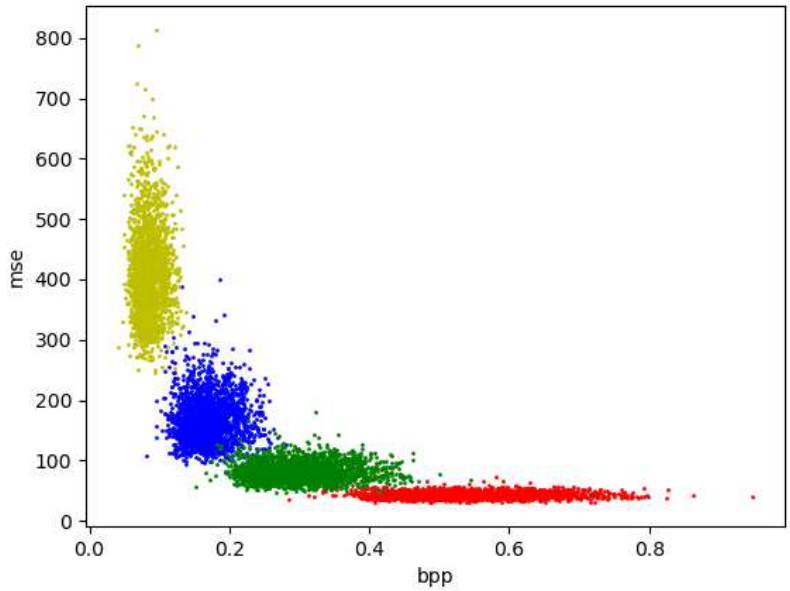
(a) Original **(b)** λ = 0.01 **(c)** λ = 0.001

(d) λ = 0.003 **(e)** λ = 0.0003

Figure 4.8: Same fingerprint compression comparison

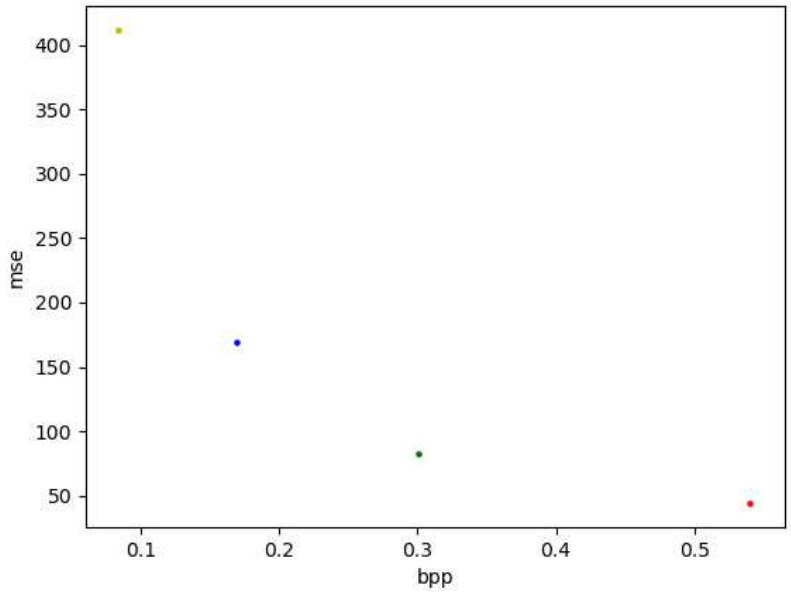


(a) PSNR vs BPP plot for the different lambda values

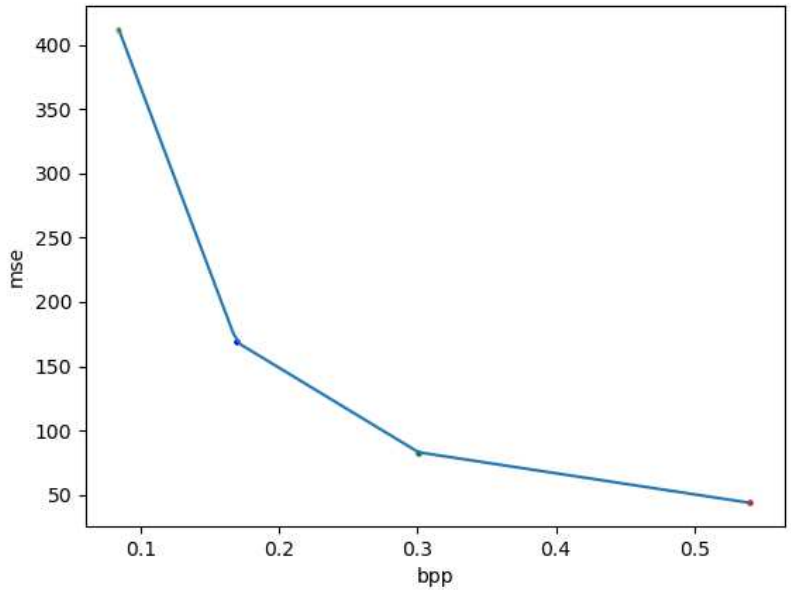


(b) MSE vs BPP plot for the different lambda values

Figure 4.9: Compression clusters



(a) average of the previous clusters



(b) linearized approximation of the previous averages

Figure 4.10: Clusters averages

5 Attacks

In order to evaluate the feasibility of identity inference for biometric generative models, the considered Convolutional Generative Adversarial Networks (C-GANs) approaches represent the most frequently adopted in biometric applications and allow to generalize the obtained results to the whole family of models. The black box attack carried out against all the trained models is inspired by the one proposed in [22] (see scheme in figure 5.1). The main assumptions are that the attacker has access to some APIs that allow generating from the target C-GAN G_a, D_a as many samples as he wants (the structure of the target model is not known) and he/she is in possession of a dataset containing some of the samples used to train G_a, D_a (called the query dataset in order of confidence). The attacker trains a shadow GAN G_s, D_s to mimic G_a, D_a. More precisely, G_s is trained to generate samples as close as possible to those produced by G_a, while the discriminator D_s classifies whether the sample was generated by G_a or G_s. In this way, D_s will infer some peculiar features that G_a inadvertently introduces in the generated samples (likely due to overfitting). At this point, the confidence values predicted by D_s make it possible to sort the samples in the query dataset. Ideally, the top-k samples are going to have a higher likelihood of being members of the training set, thus achieving a successful membership inference attack. The main intuition behind this attack is that it should be able to better recognize images used to train G_a, D_a since they should present features that are more similar to the ones displayed in samples generated by G_a. In [22] the authors show how the attack performance improves with the training time, but knowing in advance a sufficient number of iterations allows for the reduction of the computational burden.

In this work, by exploiting the Inception Score (IS) [43] as the metric for early stopping, is possible to obtain remarkable MIA performance. In particular, the training was stopped when the IS of the shadow model reached a similar outcome as the one of the attacked one.

Lastly, Identity Inference Attacks are developed as well. In this case, the aim is not only to determine if a given impression was used during training but also to infer if the specific finger (i.e., identity) itself was used in the training process. Here, the assumptions formulated above are slightly changed. In particular, the query dataset is modified so that it does not contain the same impression of the finger used in the training datasets. This is a more realistic scenario since it is unlikely for an attacker to use the same biometric samples. The black box attack remains exactly identical to the one carried out in the MIA case since also in IIA the objective is for D_s to recognize features of the fingerprints that were present in the training set, and thus also on other impressions of the same fingerprints.

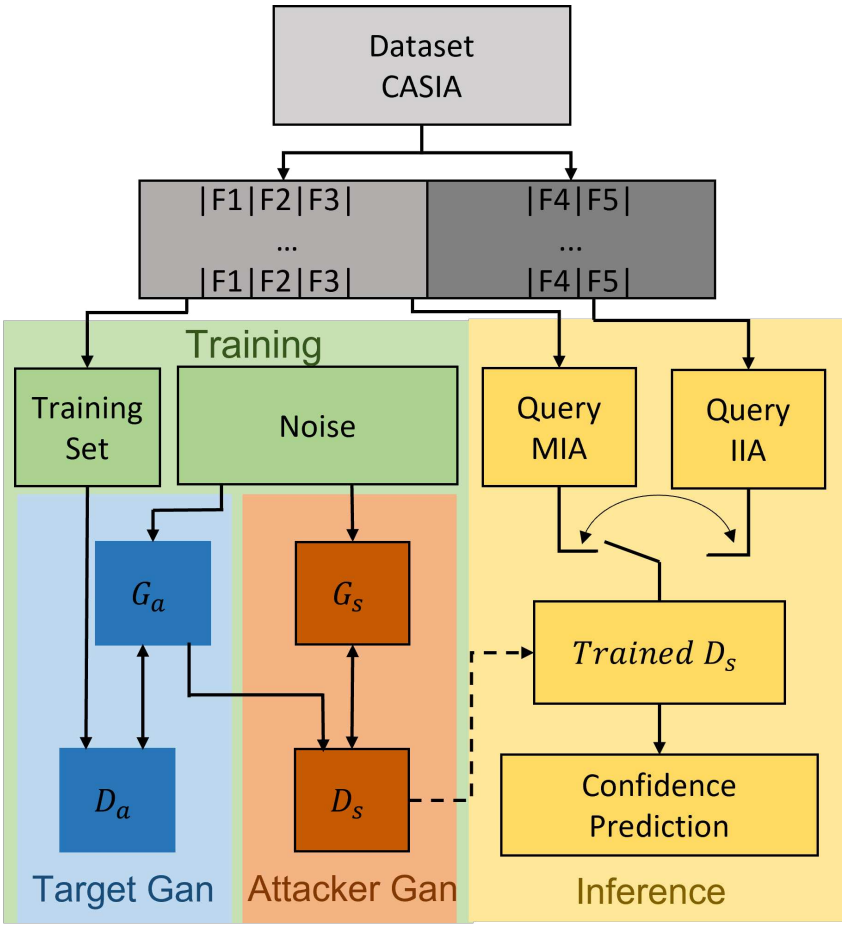


Figure 5.1: Attack scheme

5.1 Results

First experimental considerations concern the quality of the generated images and the effectiveness in using IS as a termination criterion.

Figure 4.4 shows that the generated fingerprint images consistently present level 1 (ridge orientation and singular points) and 2 (minutiae) features, with the sporadic presence of level 3 (pores) features as well.

This is further highlighted by the IS values showed accordingly in table 4.1. In figure 5.2 4 generated fingerprints impression are proposed. On the left, highest confidence fingerprints that are actual members. On the right, highest scoring images that are different impressions of training samples. (Gan 2400 and Gan 4800) obtained on the generated images, which are comparable to those computed on fingerprints. It is also possible to notice that the lower the amount of training images the lower the IS value.

Figure 5.2: Highest Inference results side by side comparison

In table 5.1 it is possible to find the number of members and corresponding fingerprints

found in the top 20/200/2000 samples by confidence (except for the cases where the training set size is smaller than 2000). It is possible to see that for all the trained GANs the percentage of samples in the first n $\in$ 20, 200, 2000 is always above 50 %, showing that this type of attack actually allows to gather some information about the training dataset since it achieves better performance than random guessing. Additionally, it is possible to see that the performance of the attack is only slightly worse when performing IIA compared to MIA indicating that, even if this kind of attack is more challenging, it can be carried out with significant success.

This demonstrates the effectiveness of both MIA and IIA in detecting sample and finger membership in the training dataset for a GAN. As a matter of fact, the attacks lead to near-perfect recognition of many fingerprints as belonging to the targeted sample in both scenarios.

An additional proof that the success of the attack is provided by a visual inspection of images where D_s confidence suggests that they are more likely to belong to the training dataset. These indeed exhibit various similarities especially within the Level 1 features as can be seen from figure 5.2.

Remarkable results can be observed especially in the case of the GAN trained on 4800 samples, wher out of the 20 flagged images with the highest scores, 19 of them are indeed from the training dataset.

Once the actual training images are swapped with a set of "remaining" acquisitions, i.e., different samples of the same fingers, the result remains high with 18 out of 20 correctly identified showing how much information these types of models can actually leak even though they were designed in the first place to avoid privacy concerns.

	MIA			IIA		
GAN	top 20	top 200	top 50%	top 20	top 200	top 50%
Gan_9600	14/20	126/200	1172/2000	12/20	114/200	1161/2000
Gan_4800	19/20	144/200	1133/2000	18/20	140/200	1133/2000
Gan_2400	12/20	108/200	869/1600	13/20	105/200	869/1600
Gan_1200	13/20	132/200	526/800	12/20	137/200	540/800

Table 5.1: MIA and IIA Results.

6 Conclusion

This work illustrates the main vulnerabilities of generative adversarial networks as means to solve the data shortage and privacy issues for learned biometric architectures. More precisely, Membership and Identity Inference Attacks on fingerprint GANs are described and evaluated showing how it is possible to infer users' identity from a trained black box model. This is the first work to notice the severity of the problem on fingerprint data showing that it is possible to assess membership of a sample and to detect if a person contributed to the creation of a dataset without having access to any of the training data.

6.1 Future Works

In future works, researchers should analyze how this problem affects other kinds of biometric samples and architectures. Additionally, some defense strategies should be designed in order to make fingerprint generation a truly secure way to solve the data shortage problem in the biometric field.

In the last efforts of this thesis some work has been already developed to extend the Membership/Identity Inference Analysis over the convolutional autoencoders for data compression as well (i.e. the Codecs defined priorly).

Such models represent a novel and really interesting technology that is remarkably keen to many different vulnerabilities, but also introduces stochastic properties that can provide several obfuscation techniques to shield generative models from the attacks assessed in this work.

This time the privacy attacking strategy concerning overfitting is based on the quality of the compression/decompression systems, measured through the means of MSE, PSNR and the losses defined for the models.

The idea is to adapt the MIA attack to these different kind of network, applying the similar concept inherited by the previous literature but generalized over the different self-imposed tasks. Indeed, with this model an attacker looks at leveraging the different quality of the compression that the codecs exhibits. Therefore the attack is going to be based on quality inference of the differences in compression/decompression of images already observed - and images never observed - expecting to capture different remarkable tendencies. An experimented model of this kind of attack can be observed in figure 6.1.

```
#evaluate the loss for the compression (loss = bpp + self.lmbda * mse)

#calculate the loss for training set
loss_train_01 = [bpp_train_01[i] + 0.01 * mse_train_01[i] for i in range(len(bpp_train_01))]
loss_train_0001 = [bpp_train_001[i] + 0.01 * mse_train_001[i] for i in range(len(bpp_train_001))]
#calculate the loss for validation set
loss_val_01 = [bpp_val_01[i] + 0.01 * mse_val_01[i] for i in range(len(bpp_val_01))]
loss_val_001 = [bpp_val_001[i] + 0.01 * mse_val_001[i] for i in range(len(bpp_val_001))]

#compare the means of loss for training set
print("Training set")
print("Loss 0.1: ", np.mean(loss_train_01))
print("Loss 0.01: ", np.mean(loss_train_001))
#compare the means of loss for validation set
print("Validation set")
print("Loss 0.1: ", np.mean(loss_val_01))
print("Loss 0.01: ", np.mean(loss_val_001))
```

Figure 6.1: Compression-quality based attack

Currently, when applying the same attacking strategy described in this thesis, it is already possible to observe a really interesting behaviour as reported in table 6.1, as the different sets of images are indeed introducing a statistically relevant difference between compression quality of data used to train the Codecs and data that the model never saw. (I reported the most explicative results, based on the lambda values $\lambda = \{0.01, 0.001, 0.003\}$, which consistently with the IS logic for gans, provided the best compressions, as noticeable in figure 4.8).

	MIA		
GAN	top 20	top 200	top 50%
Codec_0.01	17/20	164/200	1203/2000
Codec_0.001	14/20	142/200	1122/2000
Codec_0.003	16/20	147/200	1163/2000

Table 6.1: MIA results for the experimental codec attack

The issue with this approach at the moment is that sometimes a better compression/decompression performance on images not used to train the Codecs was found. At the moment different experimenting techniques are being carried to figure the reason of this strange underfitting and sometimes random outcomes.

Nevertheless, the fact that most of the times the models actually inherited some intrinsic divergences between the two sets outcome similar results to the GANs' ones, proving how close this strategy is to figure a systemic and generalized way to accurately infer and motivate such differences, hence proving again the efficiency of these attacks (and therefore the need for protection).

Once such vulnerabilities will be generalized and assessed, the need to find a way to neutralize the inference capabilities of these attacks will come along.

As the crowning strategy of all this research, the defense mechanism could combine these models together by re-introducing the GAN as a **twin model** to defend the Codecs.

As proved, different GANs training provide different levels of vulnerability, and this remark will pose itself as the base of a privacy preserving generation of images that will be able to **additionally protect a Codec, by becoming the actual training set of the latter**.

This will imply that the Codec will be trained on generated images (not actual/real fingerprints), and these generated images will be created by an already privacy aware GAN (tested with the most efficient and secure techniques introduced in this thesis) that will still generate high level samples (that will not significantly affect the quality of the compressions).

Indeed, during this work the generated images have always been checked for the quality and reliability of the models and of the attack/defense strategies in both a quantitative and qualitative way. The GANs are therefore going to be tested again using the methods already thoroughly described. Moreover, the Codecs are assessed by the joint analysis over mse,psnr,bpp and the model losses themself (providing the same level of analysis displayed in the "model" chapter).

Additionally, to prove the effectiveness of the compression standards, the statistical properties of the key features of the fingerprints (cores,ridges,minutiae, etc) [47] are going to be tested as well, to prove the feasibility in the real world application of the proposed processes.

References

[1] A. El Kissi and N. ESSOUKRI BEN AMARA, "Soft and hard biometrics for the authentication of remote people in front and side views," vol. 11, pp. 8120–8127, 01 2016.

[2] R. Bolle, S. Pankanti, and A. K. Jain, *Biometrics, Personal Identification in Networked Society: Personal Identification in Networked Society*. USA: Kluwer Academic Publishers, 1998.

[3] A. Jain, P. Flynn, and A. Ross, *Handbook of Biometrics*. Springer US, 2007.

[4] B. Institute. Types of biometrics - fingerprint: Key considerations. [Online]. Available: https://www.biometricsinstitute.org/types-of-biometrics-fingerprint-key-considerations/

[5] Bayometric. Minutiae-based extraction in fingerprint recognition. [Online]. Available: https://www.bayometric.com/minutiae-based-extraction-fingerprint-recognition/

[6] Healthline. Do identical twins have the same fingerprints? [Online]. Available: https://www.healthline.com/health/do-identical-twins-have-the-same-fingerprints

[7] P. Biometrics. (2014) Understanding biometric performance evaluation. [Online]. Available: https://precisebiometrics.com/wp-content/uploads/2014/11/White-Paper-Understanding-Biometric-Performance-Evaluation-QR.pdf

[8] V. Mistry, J. J. Engelsma, and A. K. Jain, "Fingerprint synthesis: Search with 100 million prints," 2020.

[9] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014.

[10] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," 2017.

[11] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision (ICCV)*, 2017.

[12] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, 2019.

[13] S. Minaee and A. Abdolrashidi, "Finger-gan: Generating realistic fingerprint images using connectivity imposed gan," 2018.

[14] A. Sams, H. Shomee, and S. Rahman, "Hq-fingan: High-quality synthetic fingerprint generation using gans," *Circuits, Systems, and Signal Processing*, vol. 41, pp. 1–16, 07 2022.

[15] R. Bouzaglo and Y. Keller, "Synthesis and reconstruction of fingerprints using generative adversarial networks," 2023.

[16] X. Huang, P. Qian, and M. Liu, "Latent fingerprint image enhancement based on progressive generative adversarial network," 06 2020, pp. 3481–3489.

[17] I. Joshi, A. Anand, M. Vatsa, R. Singh, S. Dutta Roy, and P. Kalra, "Latent fingerprint enhancement using generative adversarial networks," 01 2019, pp. 895–903.

[18] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of wasserstein gans," 03 2017.

[19] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," 2016.

[20] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," 2020.

[21] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2242–2251.

[22] J. Hayes, L. Melis, G. Danezis, and E. D. Cristofaro, "Logan: Membership inference attacks against generative models," 2018.

[23] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," 2015.

[24] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," 2017.

[25] H. Hu, Z. Salcic, L. Sun, G. Dobbie, P. S. Yu, and X. Zhang, "Membership inference attacks on machine learning: A survey," 2022.

[26] S. Yeom, I. Giacomelli, M. Fredrikson, and S. Jha, "Privacy risk in machine learning: Analyzing the connection to overfitting."

[27] A. Salem, Y. Zhang, M. Humbert, P. Berrang, M. Fritz, and M. Backes, "Ml-leaks: Model and data independent membership inference attacks and defenses on machine learning models," 2018.

[28] B. Hilprecht, M. Härterich, and D. Bernau, "Reconstruction and membership inference attacks against generative models," 2019.

[29] G. Li, S. Rezaei, and X. Liu, "User-level membership inference attack against metric embedding learning," 2022.

[30] C. Song and V. Shmatikov, "Auditing data provenance in text-generation models," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. New York, NY, USA: Association for Computing Machinery, 2019, p. 196–206.

[31] Y. Miao, M. Xue, C. Chen, L. Pan, J. Zhang, B. Z. H. Zhao, D. Kaafar, and Y. Xiang, "The audio auditor: User-level membership inference in internet of things voice services," 2021.

[32] D. A. Huffman, "A method for the construction of minimum-redundancy codes," *Proceedings of the IRE*, vol. 40, no. 9, pp. 1098–1101, 1952.

[33] J. Ziv and A. Lempel, "A universal algorithm for sequential data compression," *IEEE Trans. Inf. Theory*, vol. 23, pp. 337–343, 1977. [Online]. Available: https://api.semanticscholar.org/CorpusID:9267632

[34] D. P. Kingma and M. Welling, "An introduction to variational autoencoders," *Foundations and Trends® in Machine Learning*, pp. 1–18, 2019.

[35] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," 2017.

[36] Y. Hu, W. Yang, Z. Ma, and J. Liu, "Learning end-to-end lossy image compression: A benchmark," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PP, pp. 1–1, 03 2021.

[37] D. Sculley and C. Brodley, "Compression and machine learning: A new perspective on feature space vectors," 04 2006, pp. 332– 341.

[38] J. Ballé, S. J. Hwang, and E. Agustsson, "TensorFlow Compression: Learned data compression," 2023. [Online]. Available: http://github.com/tensorflow/compression

[39] J. Ballé, D. Minnen, S. Singh, S. J. Hwang, and N. Johnston, "Variational image compression with a scale hyperprior," 2018.

[40] Biometric ideal test dataset - fingerprint. [Online]. Available: http://biometrics.idealtest.org/downloadDesc.do?id=7#/datasetDetail/7

[41] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, pp. 2278 – 2324, 12 1998.

[42] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," 2013.

[43] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," 2016.

[44] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.

[45] C.-B. Zhang, P.-T. Jiang, Q. Hou, Y. Wei, Q. Han, Z. Li, and M.-M. Cheng, "Delving deep into label smoothing," *Trans. Img. Proc.*, vol. 30, 2021.

[46] D. Arpit, S. Jastrzębski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio, and S. Lacoste-Julien, "A closer look at memorization in deep networks," 2017.

[47] D.-L. Nguyen, K. Cao, and A. K. Jain, "Robust minutiae extractor: Integrating deep networks and fingerprint domain knowledge," 2017.