BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF INTELLIGENT SYSTEMS
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

GENERATIVE ADVERSARIAL NETWORKS APPLIED FOR PRIVACY PRESERVATION IN BIO-METRIC-BASED AUTHENTICATION AND IDENTIFICATION

GENERATIVNÍ ADVERSARIALNÍ NEURONOVÉ SÍTĚ VYUŽITY NA OCHRANU SOUKROMÍ PŘI BIOMETRICKÉ AUTENTIFIKACI A IDENTIFIKACI

MASTER'S THESIS
DIPLOMOVÁ PRÁCE

AUTHOR — **Bc. ĽUBOŠ MJACHKY**
AUTOR PRÁCE

SUPERVISOR — **Ing. IVAN HOMOLIAK, Ph.D.**
VEDOUCÍ PRÁCE

BRNO 2021

Brno University of Technology
Faculty of Information Technology

Department of Intelligent Systems (DITS) Academic year 2020/2021

Master's Thesis Specification

Student:	**Mjachky Ľuboš, Bc.**
Programme:	Information Technology
Field of study:	Information Technology Security
Title:	**Generative Adversarial Networks Applied for Privacy Preservation in Bio-Metric-Based Authentication and Identification**
Category:	Artificial Intelligence

Assignment:

1. Get familiar with existing GAN approaches, such as dual-GAN, cyclic-GAN, disco-GAN. Compare these methods in terms of performance.
2. Acquaint yourself with principles of differential privacy and homomorpic encryption and discuss their application to privacy preservation in bio-metric data, such as face pictures, fingerprints, etc.
3. Compare collaborative and centralized machine learning approaches applicable in the context of authentication and identification. Focus on privacy aspects.
4. Propose a new method for privacy preservation in bio-metric-based authentication and identification using GANs.
5. Evaluate the proposed method on an arbitrary dataset of faces and arbitrary supervised classifier performing authentication/identification. Focus on comparison of performance drop with use of the proposed method.
6. Perform adversarial experiments trying to evade the classifier trained on transformed data of your method.

Recommended literature:

- Yi, Zili, et al. "Dualgan: Unsupervised dual learning for image-to-image translation." *Proceedings of the IEEE international conference on computer vision*. 2017.
- Hitaj, Briland, Giuseppe Ateniese, and Fernando Perez-Cruz. "Deep models under the GAN: information leakage from collaborative deep learning." *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2017.
- Bathen, Luis, et al. "SelfIs: Self-Sovereign Biometric IDs." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2019.

Requirements for the semestral defence:

- Items 1 to 3.

Detailed formal requirements can be found at https://www.fit.vut.cz/study/theses/

Supervisor:	**Homoliak Ivan, Ing., Ph.D.**
Head of Department:	Hanáček Petr, doc. Dr. Ing.
Beginning of work:	November 1, 2020
Submission deadline:	May 19, 2021
Approval date:	November 11, 2020

Abstract

Biometric-based authentication systems are getting broadly adopted in many areas. However, these systems do not allow participating users to influence the way their data will be used. Furthermore, the data may leak and can be misused without the users' knowledge. In this thesis, we propose a new authentication method which preserves the privacy of an individual and is based on a generative adversarial network (GAN). Concretely, we suggest using the GAN for translating images of faces to a visually private domain (e.g., flowers or shoes). Classifiers, which are used for authentication purposes, are then trained on the images from the visually private domain. Based on our experiments, the method is robust against attacks and still provides meaningful utility.

Abstrakt

Systémy založené na biometrickej autentizácii sa stávajú súčasťou nášho každodenného bytia. Tieto systémy však nedovoľujú používateľom priamo alebo nepriamo meniť spôsob, akým sa k ich dátam pristupuje a ako sa s nimi bude zaobchádzať ďalej v budúcnosti. Dôsledkom tohto môžu vyplynúť riziká spojené s uniknutím identity jedinca. Táto práca sa zaoberá návrhom systému, ktorý zachováva privátnosť a zároveň umožňuje autentizáciu na základe biometrických čŕt používateľov, a to za pomoci generatívnej neurónovej siete (GAN). V práci sa konkrétne uvažuje o tom, že GAN je použitá na transformáciu obrázkov tvárí napríklad na obrázky kvetov. Autentizačný systém sídliaci na serveri je v konečnom dôsledku učený rozlišovať používateľov podľa obrázkov kvetov a nie tvárí. Na základe vykonaných experimentov môžeme potvrdiť, že navrhovaná metóda je robustná voči útokom, pričom stále vykazuje kvalitatívne požiadavky kladené na štandardný autentizačný systém.

Keywords

privacy preservation, machine learning, generative adversarial networks, biometric systems

Klíčová slova

ochrana súkromia, strojové učenie, generatívne adversariálne siete, biometrické systémy

Reference

MJACHKY, Ľuboš. *Generative Adversarial Networks Applied for Privacy Preservation in Bio-Metric-Based Authentication and Identification.* Brno, 2021. Master's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Ing. Ivan Homoliak, Ph.D.

Rozšířený abstrakt

Neurónové siete sú v dnešnej dobe používané na rôzne účely. Môžu rozpoznávať objekty, identifikovať používateľa v rámci systému alebo slúžiť k prevodu reči na text či transformácií obrázkov z jednej domény do druhej. Pri transformácií obrázkov sa využívajú tzv. generatívne neurónové siete. Generatívna neurónová sieť (ďalej len GAN) pozostáva z dvoch neurónových sietí, kde jedna z nich sa nazýva diskriminátor a druhá generátor. Generátor generuje obrázky a diskriminátor určuje, či obrázok, ktorý je aktuálne spracovávaný, pochádza z generátora alebo zo skutočného datasetu. Generátor sa na základe výstupov diskriminátora učí, ako generovať také obrázky, aby sa čo najviac podobali tým reálnym.

Štandardný GAN model nie je dostatočne robustný a preto sa do trénovacej procedúry pridávajú ďalšie loss funkcie, ktoré zohľadňujú napr. cyklickú konzistenciu alebo geometrickú konzistenciu medzi transformovanými obrázkami. V práci sú diskutované moderné GAN architektúry, ktoré dosahujú skvelé výsledky pri menšom počte trénovacích dát alebo kratšom čase potrebnom na trénovanie. Konkrétne sa vedie diskusia o CycleGAN, DualGAN, DiscoGAN, GANimorph, GcGAN, StarGAN, GANHopper, SPA-GAN, TraVeLGAN a U-GAT-IT.

Po predstavení GAN architektúr sa práca ďalej venuje popisu rozdielov medzi centralizovaným a kolaboratívnym učením neurónových sietí. Centralizované učenie zastáva typický model trénovania, kde neurónová sieť je učená priamo na celom datasete. Problém centralizovaného učenia spočíva v tom, že ak dataset, obsahujúci privátne dáta používateľov, unikne alebo sa dostane do rúk neoprávnenej osoby, môže to mať vážne následky. V prípade kolaboratívneho učenia je dataset rozdistribuovaný medzi niekoľkými zariadeniami a spoločne trénujú jednu alebo viac neurónových sietí.

Väčšina systémov, ktoré sú postavené na kolaboratívnom učení, využíva možnosť paralelizácie algoritmu SGD. Naviac, takáto konfigurácia umožňuje trénovať neurónové siete tak, aby bola zachovaná privátnosť jednotlivých datasetov, nakoľko neexistuje priamy prístup ku kompletnému datasetu. Na zabezpečenie privátnosti je však stále nutné používať techniky, ktoré dodatočne zabezpečia paralelizáciu algoritmu SGD. To je napríklad vynútená agregácia dát a pridávanie šumu.

Pridávanie šumu musí byť určitým spôsobom korigované. Diferenciálna privátnosť (ďalej ako DP) práve pracuje s takýmto šumom. Podstata DP spočíva v tom, že ak objekt v trénovacej sade neovplyvní výsledok trénovania, potom informácia o ňom ani nemôže byť zapamätaná. Teda, privátnosť tohto objektu je v rámci trénovacej sady potom zaručená.

V práci je popísaných niekoľko variantov DP. V každom prípade sa na vloženie šumu do dát používa tzv. randomizačný mechanizmus. Tento mechanizmus ovplyvňuje to, aká bude pravdepodobnosť toho, že objekt ovplyvní výsledok trénovania. Pravdepodobnosť ďalej určuje, aký je maximálny únik dát pri celkovom trénovaní, čomu prislúcha budget privátnosti. Platí, že čím menší je budget privátnosti, tým horšie sa dá určiť, či sa objekt nachádza v trénovacej sade a tým viac šumu sa vkladá do dát. Výsledkom príliš malého budget-u privátnosti je, že neurónová sieť bude mať nízku výkonnosť a stane sa nepoužiteľnou.

Mimo DP existuje ďalšia známa možnosť ochrany, a to homomorfné šifrovanie (HE). HE umožňuje pracovať so zašifrovanými dátami bez toho, aby sa niekedy museli odšifrovať. Tento koncept je však mimoriadne náročný na realizáciu a oproti DP nie je tak ľahko použiteľný pri trénovaní neurónových sietí.

Systémy založené na biometrickej autentizácii často používajú neurónové siete, ktoré sa učia identifikovať používateľov na základe ich biometrických čŕt. Ak tieto biometrické dáta uniknú, útočník získa šablónu biometrických čŕt a dokáže spätne určiť tváre všetkých používateľov systému.

V tejto práci navrhujeme riešenie, ktoré zaručuje privátnosť používateľov aj v prípade centralizovaného učenia. Konkrétne navrhujeme použiť GAN na transformáciu obrázkov tvárí napr. na obrázky kvetov alebo topánok. Transformácia prebieha ešte na zariadení používateľa a server, na ktorom sa nachádza neurónová sieť slúžiaca k autentizácii, má k dispozícii obrázky len zo zabezpečenej domény. Toto riešenie umožňuje používateľom sa prihlasovať do systému na základe svojich biometrických čŕt bez toho, aby niekedy boli skutočne odtajnené.

Zo všetkých predstavených GAN architektúr vyhovoval našej metóde najviac TraVeLGAN. TraVeLGAN modely, trénované na datasete tvárí celebrít a kvetov, boli použité pri finálnom vyhodnocovaní a testovaní navrhnutej metódy. Experimenty ukázali, že v prípade, že sa použili na identifikáciu používateľov obrázky kvetov, celkové zníženie výkonnosti systému z pohľadu F-1 skóre neprekročilo 6%.

Ďalej boli v rámci experimentovania uskutočnené dva typy útokov na našu metódu. Prvý typ útoku pozostával z natrénovania rovnakej GAN architektúry, ale s prehodenými doménami. Neurónová sieť sa teda učila transformovať obrázky kvetov na obrázky tvárí. Tento útok nebol v žiadnom z pokusov úspešný, ak útočník chcel zrekonštruovať tvár používateľa z odchyteného obrázku kvetu. Druhý typ útoku pozostával zo znalosti správnych párov tvár-kvet, ktoré generuje používateľský GAN model. V tomto type útoku má útočník k dispozícii všetky informácie okrem privátneho datasetu používateľa (obrázkov jeho tváre). Útok bol úspešný z hľadiska určenia pohlavia, približnej geometrie tváre a štýlu vlasov. Avšak skutočná identita používateľa nemohla byť nikdy jasne deklarovaná.

Počas vykonávania druhého útoku sme zistili, že ak útočník pretransformuje rekonštruovanú tvár späť do domény kvetov, tak sa v systéme môže autentizovať ako reálny používateľ v 45% prípadov. Pridanie Gaussovského šumu do transformovaných obrázkov čiastočne problém zredukovalo, avšak za cenu presnosti klasifikátora.

Záverom práce zhrňujeme, že ak útočník získa priamy prístup k používateľskému GAN modelu a odchytí niekoľko obrázkov kvetov identifikujúcich reálnu tvár, ochrana súkromia je vždy do určitého spektra zaručená. Utočník ale môže z odchytených obrázkov vygenerovať nové obrázky kvetov a nimi sa úspešne autentizovať na serveri takmer v jednom z dvoch pokusov.

Aj tento problém sa dá vyriešiť tým, že používateľský GAN model bude v zariadení zabezpečený a prístup k nemu bude vždy autorizovaný napr. operačným systémom tak, aby útočník nemohol čítať váhy natrénovanej siete.

V tejto práci sme predstavili koncept použitia GAN architektúr na zachovanie privátnosti používateľov pri autentizácii. Do budúcna navrhujeme otestovať robustnosť systému s 500 a viac používateľmi, nakoľko experimenty prebiehali len v systéme s takmer 100 používateľmi. Podobne odporúčame ďalej experimentovať s DP, keďže sme počas testov zistili, že komplexné GAN architektúry sú príliš citlivé na vkladanie akéhokoľvek šumu do procesu trénovania. Ak sme použili DP v modeloch TraVeLGAN, generované obrázky neboli dostatočne rôznorodé, čo je jedna z podmienok nutných na autentizáciu používateľov. Na záver navrhujeme otestovať novšie GAN architektúry a zlepšiť tak kvalitu pretransformovaných obrázkov v zabezpečenej doméne.

Generative Adversarial Networks Applied for Privacy Preservation in Bio-Metric-Based Authentication and Identification

Declaration

I hereby declare that this Master's thesis was prepared as an original work by the author under the supervision of Ing. Ivan Homoliak, Ph.D. I have listed all the literary sources, publications and other sources, which were used during the preparation of this thesis.

.....
Ľuboš Mjachky
May 12, 2021

Acknowledgements

I would like to express my deepest gratitude to my supervisor Ing. Ivan Homoliak, Ph.D. for his guidance and valuable advice. Also, I would like to thank the National Grid Infrastructure that is operated by MetaCentrum. Computational resources were supplied by the project "e-Infrastruktura CZ" (e-INFRA LM2018140) provided within the program Projects of Large Research, Development and Innovations Infrastructures.

Contents

Chapter 1

Introduction

Biometric-based authentication systems are being used by millions of people daily. Every modern smartphone is equipped with a fingerprint sensor or facial recognition hardware. However, information about fingerprints or faces may end up in an unreliable place, and users cannot influence that. In general, the users are not able to affect the way biometric systems manipulate their data.

This opens a question of trust because biometric data are not easily cancelable. Once a regular password leaks, there is a way to change it. Changing a user's face when it was exposed to an attacker is not really feasible.

The privacy of an individual should be always protected unless there is a public interest in revealing it. In this thesis, there are discussed current limitations and possibilities of how privacy-preserving authentication and identification systems work. Furthermore, we aim our attention on defining a novel method for preserving privacy by utilizing a generative adversarial network (GAN). The GAN is used to translate images of faces to a visually private domain (e.g., flowers), totally unrelated to the domain of faces. Thus, the privacy is assured solely by the GAN and the learnt mapping function between the domains.

In the thesis, we assume scenarios, where a user is authenticated by images of flowers. Such scenarios simulate standard environments where users log in to the biometric-based systems without revealing their actual identities.

Overall, our contributions are twofold. First, we proposed a method that does not require any extra transformation to perform a translation to a visually private domain, as opposed to the existing works. Second, we carried out experiments on multiple target domains and successfully validated the proposed method on real-world binary classification tasks, representing a centralized authentication use case.

The thesis is organized as follows: in Chapter 2, there are clarified basic concepts of GANs. Then, a variety of GAN architectures, used for image-to-image translation, is compared between each other. Chapter 3 presents two frequently used techniques for privacy preservation, i.e., differential privacy (DP) and homomorphic encryption (HE). In Chapter 4, we compare centralized learning with collaborative learning while focusing on privacy aspects. Chapter 5 contains a description of the proposed method. In Chapter 6, we assess and execute experiments to validate the proposed method. Then, we devise a plan for attacking our method in Chapter 7. In the end, a conclusion is drawn from the performed experiments and possible future improvements are discussed.

Chapter 2

Generative Adversarial Networks

Generative adversarial networks (GANs) are used in many settings. For instance, one can employ the GANs in generating realistic pictures of landscapes, animals, or even human faces.

At the beginning of this chapter, we explain the notion of the standard GAN architecture. After that, we extend this concept to recent architectures and compare these approaches in terms of performance. Finally, for an overall comparison of the architectures, with regards to loss functions or mapping capabilities, see Section 2.3.

2.1 Basic Concepts

A GAN consists of two base models: a generator and a discriminator. The generative model produces fake samples and the discriminative model strives to determine whether a sample comes from the real data distribution or the model distribution. Both of these models are trained simultaneously and they play the two-player minimax game with the following objective:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[\log D(x)] + \mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z)))] \tag{2.1}$$

Where $p_z(z)$ represents input noise variables (referred to as a latent vector), $G(z)$ describes a differentiable mapping function to the data space, and $D(x)$ stands for a function which outputs a single scalar which defines the probability that x comes from the data source and not from the generator's distribution. $\mathbb{E}$ is the expected value over instances given by the subscripts. D is trained to maximize the probability of assigning correct labels for examples. On the other hand, G is trained to minimize $\log(1 - D(G(z)))$, because it cannot affect the term $\log(D(x))$ directly [26].

In Figure 2.1 below, we show how the training of the standard GAN framework looks like. The discriminator is fed with the real samples and the generated samples. The output of the discriminator is exploited during the backpropagation. The weights of the discriminator and generator are then updated correspondingly.

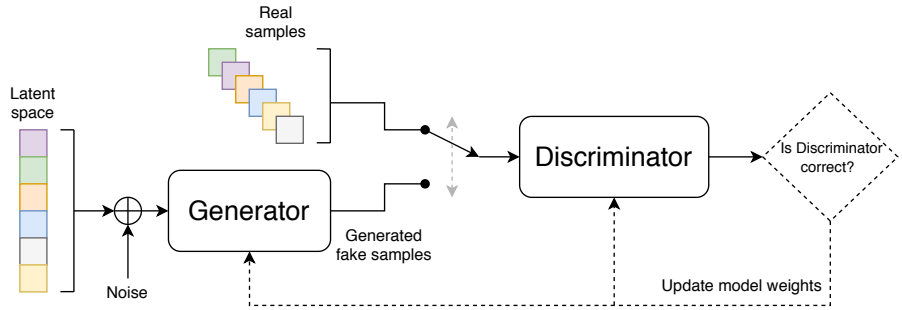

Figure 2.1: A conceptual illustration of the standard GAN [21].

2.2 Architectures

This whole section contains a summary of commonly known GANs and considers only the networks designed for image-to-image translation tasks. The goal of the image-to-image translation is to learn the mapping between two or more domains. Once there is discovered such mapping, we can transfer styles or textures from one domain to another.

2.2.1 CycleGAN

Pictures within a specific domain always have something in common. By capturing the characteristics and examining how they may be translated into other domains, we solve an image-to-image translation task. CycleGAN studies those characteristics in an unsupervised manner by leveraging a pair of GANs.

CycleGAN incorporates two mappings $G_{XY} : X \to Y$ and $G_{YX} : Y \to X$. For each mapping, there is a separate generator network, accompanied by a discriminator, that focus on single domain transfer. The first discriminator tries to correctly assess generated samples from the first generator. While the second discriminator aims to determine whether the generated samples come from the second generator. CycleGAN has therefore two adversarial losses which are based on Equation 2.1. For the generator G_{XY} and discriminator D_Y, the equation looks like this:

$$\mathcal{L}_{GAN}(G_{XY}, D_Y, X, Y) = \mathbb{E}_{y \sim p_{data}y}[\log D_Y(y)] + \mathbb{E}_{x \sim p_{data}(x)}[\log(1 - D_Y(G_{XY}(x)))] \quad (2.2)$$

The adversarial losses alone cannot assure that the learned mapping between two domains is flawless. A network can still map the same set of inputs to random permutations of images in the target domain. Due to that, CycleGAN uses cycle-consistency loss to reduce the space of possible mapping functions. The idea behind the cycle-consistency loss is that for every single image x from a domain X, the translation should return the original image x, thus, $x \to G_{XY}(x) \to G_{YX}(G_{XY}(x)) \approx x$. The same analogy is applied to an image y from a domain Y. Following these notions, the cycle-consistency loss is formally defined as:

$$\mathcal{L}_{cyc}(G_{XY}, G_{YX}) = \mathbb{E}_{x \sim p_{data}x}[\|G_{YX}(G_{XY}(x)) - x\|_1] + \mathbb{E}_{y \sim p_{data}(y)}[\|G_{XY}(G_(y)) - y\|_1] \quad (2.3)$$

The distance between the vectors is processed by the L1 norm, also known as the Manhattan distance. Figure 2.2 exhibits the concept of CycleGAN.

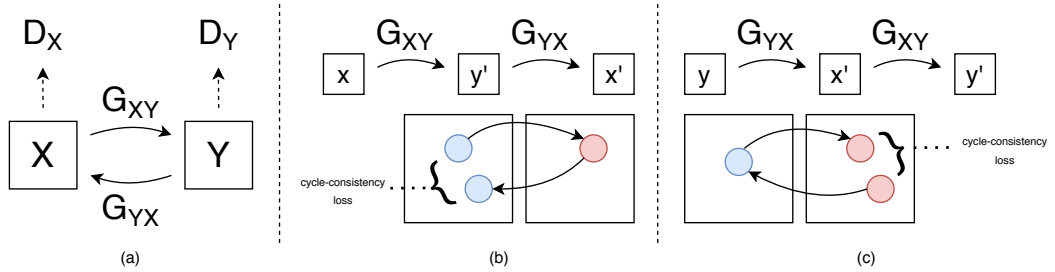

Figure 2.2: (a) CycleGAN contains two mapping functions G_{XY}, G_{YX}, and two discriminators D_X, D_Y. (b) depicts the forward cycle-consistency loss, whereas (c) visualizes the backward cycle-consistency loss [86].

In the end, CycleGAN aims to solve the main objective by combining the two losses mentioned above via the two-player minimax game, like so:

$$G^*_{XY}, G^*_{YX} = \arg \min_{G_{XY}, G_{YX}} \max_{D_X, D_Y} \mathcal{L}_{CycleGAN}(G_{XY}, G_{YX}, D_X, D_Y) \tag{2.4}$$

According to the authors of CycleGAN, the training was more stable when they replaced the standard GAN loss with LSGAN [48]. So, the generator and discriminator were eventually trained to minimize:

$$\mathcal{L}_{LSGAN}(G_{XY}) = \mathbb{E}_{x \sim p_{data}(x)}[(D_Y(G_{XY}(x)) - 1)^2] \tag{2.5}$$

$$\mathcal{L}_{LSGAN}(D_Y) = \mathbb{E}_{y \sim p_{data}(y)}[(D_Y(y) - 1)^2] + \mathbb{E}_{x \sim p_{data}(x)}[D_Y(G_{XY}(x))^2] \tag{2.6}$$

In terms of the implemented architecture, both generators consist of three convolutions and multiple residual blocks. Such an architecture was first proposed by Johnson et al. [37]. The discriminators make use of the PatchGAN architecture [33]. Here, pixels beyond a particular area are considered to be independent of each other. Thanks to that, the network pays attention to pixels just at the patch level rather than the full image. Therefore, features that are situated more frequently in images are correctly captured. This configuration allows CycleGAN to better discriminate overlapping image patches as real or fake.

CycleGAN was extensively evaluated on image-to-image translation problems that mainly consisted of transferring styles between various domains. Such problems included zebra → horse, winter → summer scene, or landscape → Van Gogh painting transfer tasks. In Figure 2.3, we provide results of the translations retrieved from the official landing page[1] of the CycleGAN project [86].

2.2.2 DualGAN

A similar approach was made by the authors of DualGAN [79]. In this case, the loss functions are supported by Wasserstein GAN (WGAN) [4]. WGAN has proven to improve the stability of training and generator's convergence while boosting the quality of generated images. In the original GAN, there is used the sigmoid cross-entropy loss (a.k.a. logarithmic loss).

[1] `https://junyanz.github.io/CycleGAN/`

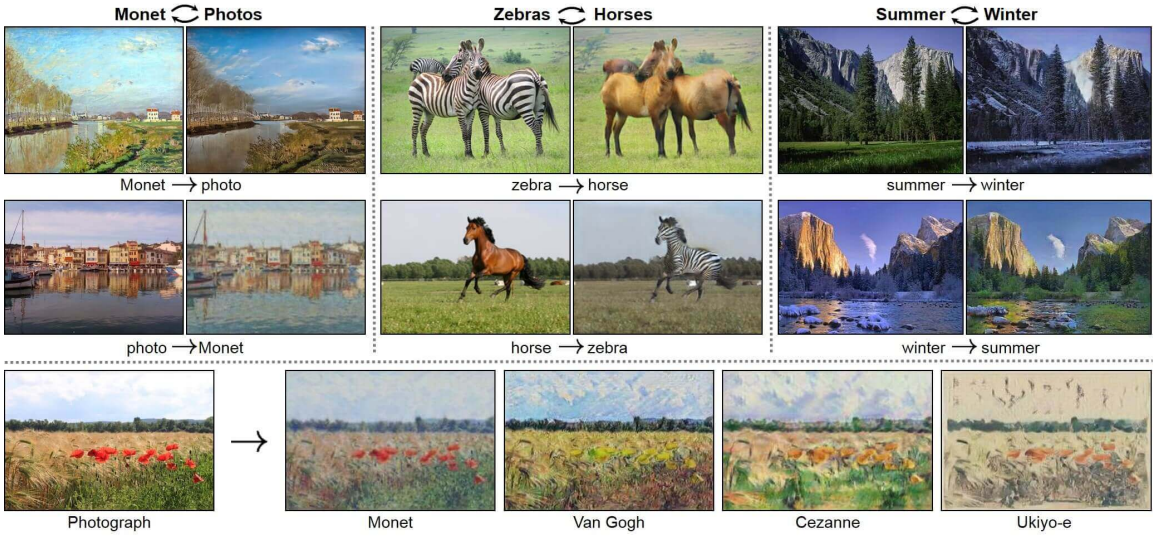

Figure 2.3: Image-to-image translations performed by CycleGAN [86].

In DualGAN, a function that accounts for reconstruction errors is used for training the generators. Assuming that X and Y are domains where $x \in X$ and $y \in Y$, λ_X and λ_Y are constant weights that influence the impact of the partial recovery errors, the loss functions of DualGAN are defined like:

$$\mathcal{L}_{D_Y}(X, Y) = D_Y(G_{XY}(x)) - D_Y(x) \tag{2.7}$$

$$\mathcal{L}_{D_X}(X, Y) = D_X(G_{YX}(y)) - D_X(y) \tag{2.8}$$

$$\begin{aligned}\mathcal{L}_G(X, Y) = \lambda_X \|x - G_{YX}(G_{XY}(x))\|_1 + \lambda_Y \|y - G_{XY}(G_{YX}(y))\|_1 \\ - D_Y(G_{XY}(x)) - D_X(G_{YX}(y))\end{aligned} \tag{2.9}$$

DualGAN establishes an alternative way of achieving the results comparable to those in CycleGAN. The reconstruction error embedded into the generators' loss is more or less identical to the cycle-consistency loss. According to Omdal [57], who evaluated the performance of CycleGAN and DualGAN in the matter of artistic image-to-image translation tasks, it was possible to train DualGAN for the same number of epochs in a quarter of the time.

The architecture of DualGAN is designed in the following way. The generators are built with an equal number of downsampling and upsampling layers. Together, these layers form a U-shaped net [60]. The discriminators adopt the PatchGAN architecture that was used in CycleGAN as well.

2.2.3 DiscoGAN

DiscoGAN is another GAN that attempts to solve image-to-image translation problems in a way akin to CycleGAN or DualGAN. Instead of the cycle-consistency loss, in DiscoGAN, there is employed reconstruction loss along with standard generator loss. For the reconstruction loss, one can use various forms of distance functions d, e.g., cosine distance, hinge-loss, or mean squared error. For an input image x from a domain X, the reconstruction loss is:

$$\mathcal{L}_{CONST_X} = d(G_{YX}(G_{XY}(x)), x) \tag{2.10}$$

In the paper [40], it is stated that the addition of one reconstruction loss does not prevent the mode collapse problem. Mode collapse occurs when two or more samples from a domain X are mapped into a single sample from a domain Y. However, DiscoGAN uses two reconstruction losses. Each loss for a different domain. The total discriminator loss is a sum of standard GAN losses, like in Equation 2.1. On the other hand, the total generator loss (Equation 2.11) is a sum of the reconstruction loss and the GAN loss. L_{CONST_X} and L_{CONST_Y} are closely related to the partial recovery errors present in Equation 2.9.

$$\mathcal{L}_G = -\mathcal{L}_{GAN_Y} + \mathcal{L}_{CONST_X} - \mathcal{L}_{GAN_X} + \mathcal{L}_{CONST_Y} \tag{2.11}$$

Each generator feeds its input through an encoder-decoder pair composed of convolution and transposed convolution layers (also called upsampling layers). The discriminators' architecture is relatively similar to the encoders used within the generators. The only difference is that one more convolution layer is added at the end of the network, together with the Sigmoid activation function. The complexity of the architecture is analogous to DualGAN and CycleGAN [40].

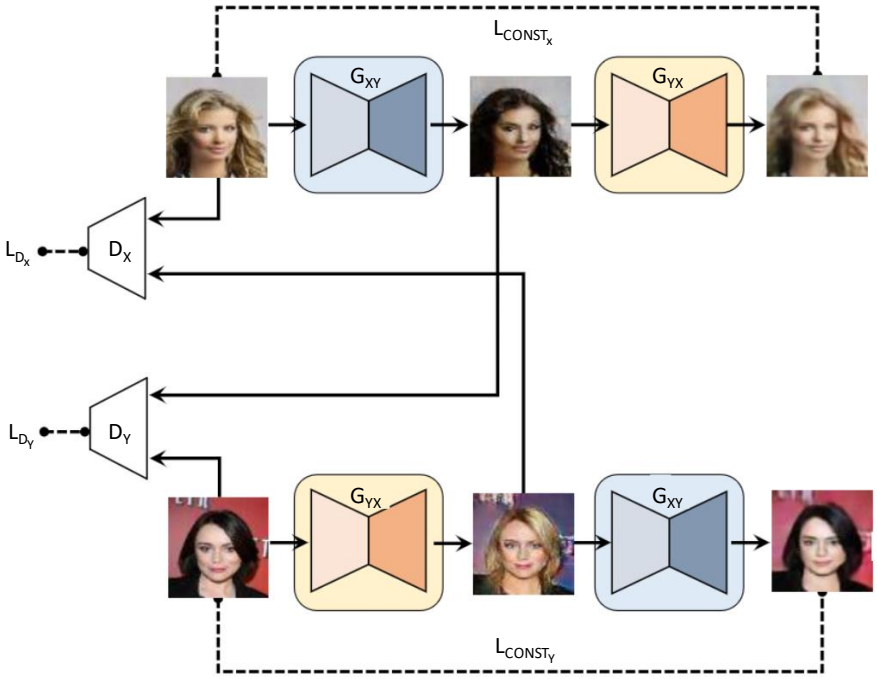

Figure 2.4: The framework of DiscoGAN [40].

2.2.4 GANimorph

In this section, we briefly describe the GANimorph framework which was invented by Gokaslan et al. [25]. The rationale behind designing GANimorph came from the fact that GANs leveraging the cycle-consistency loss and reconstruction loss cope with larger shape

deformations hardly. Furthermore, patch-based discriminators allow the network to examine only spatially local content. Due to this, the discriminators reduce the amount of information that generators should obtain.

To allow the patch-based discriminators to perceive images more as a whole, the authors of GANimorph use dilated convolutions. For the same number of parameters, the dilated convolutions increase the generators' information flow while incorporating image data from surrounding regions. This is achieved by the expansion of a receptive field (the kernel) without losing the coverage or resolution [81]. In many machine learning frameworks, the dilation rate is configurable and indicates how much is the kernel widened.[2]

Since the discriminators can better determine where the segments belong, the issue of localizing the regions within the translated images is mitigated. In Figure 2.5, there is a comparison of outputs of CycleGAN, DiscoGAN, and GANimorph.

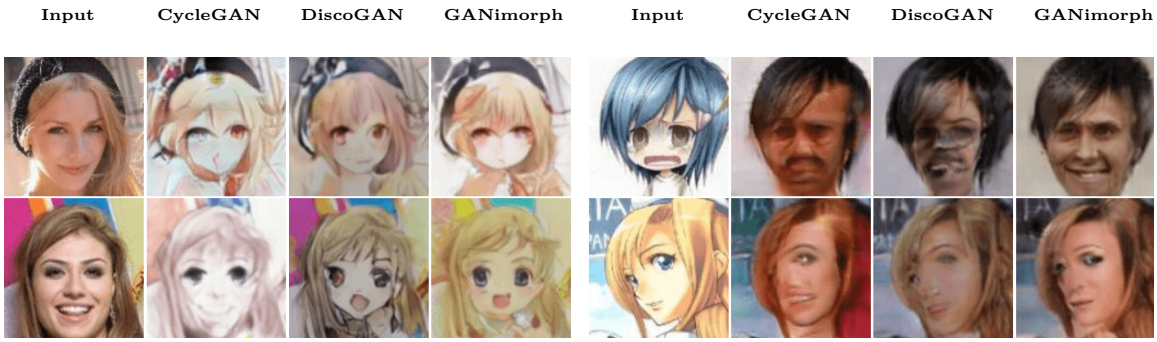

Figure 2.5: Human-to-anime and anime-to-human transfers [25].

The GANimorph's final loss function consists of the standard GAN loss, two cyclic reconstruction losses, and one feature matching loss. All these losses are normalized via so-called scheduled loss normalization. The cyclic reconstruction losses are represented by two separately computed losses based on multi-scale structure similarity loss and the L1 norm. According to Wang et al. [74], the multi-scale structure similarity loss better preserves features that are more visible to humans and allows the network to cope with shape changes more effectively. Simply put, it advocates real and fake samples to activate similar neurons within a single layer in the discriminator. This is achieved by monitoring the activation potentials during training. The feature matching loss has demonstrated that it increases the stability of the framework.

2.2.5 GcGAN

The authors of GcGAN introduced a concept of a geometry-consistent constraint into their framework. Suppose that $\widetilde{X}$ and $\widetilde{Y}$ are domains retrieved from X and Y by applying a given transformation $f(\cdot)$. The motivation behind the geometric constraint is that the transformation between two input images should be always preserved by analogous translators, i.e., G_{XY} and $G_{\widetilde{X}\widetilde{Y}}$. The geometry-consistency may be articulated as $f(G_{XY}(x)) \approx G_{\widetilde{X}\widetilde{Y}}(f(x))$ and $f^{-1}(G_{\widetilde{X}\widetilde{Y}}(f(x))) \approx G_{XY}(x)$. The function $f^{-1}(\cdot)$ is the inverse of $f(\cdot)$. In GcGAN, there was either a vertical flipping or 90°clockwise rotation used as the predefined geometric transformation function $f(\cdot)$ during the training. The consistency loss has the following form:

[2] `https://www.tensorflow.org/api_docs/python/tf/keras/layers/Conv2D`

$$\mathcal{L}_{geo}(G_{XY}, G_{\widetilde{X}\widetilde{Y}}, X, Y) = \mathbb{E}_{x\sim p_X}[\|G_{XY} - f^{-1}(G_{\widetilde{X}\widetilde{Y}}(f(x)))\|_1] + \mathbb{E}_{x\sim p_X}[\|G_{\widetilde{X}\widetilde{Y}}(f(x)) - f(G_{XY}(x))\|_1] \tag{2.12}$$

The full objective of GcGAN is a sum of the geometry-consistency loss from Equation 2.12 and the LSGAN losses for G_{XY}, D_Y, and $G_{\widetilde{X}\widetilde{Y}}$, D_X, respectively.

Regarding the implementation details, G_{XY} and $G_{\widetilde{X}\widetilde{Y}}$ have the same architecture and share all the parameters. The GcGAN framework follows the configuration of CycleGAN (using PatchGAN in the discriminators and encoder-decoder layers with residual blocks). In Figure 2.6, the reader can see how CycleGAN and GcGAN perform on regular image-to-image translation tasks.

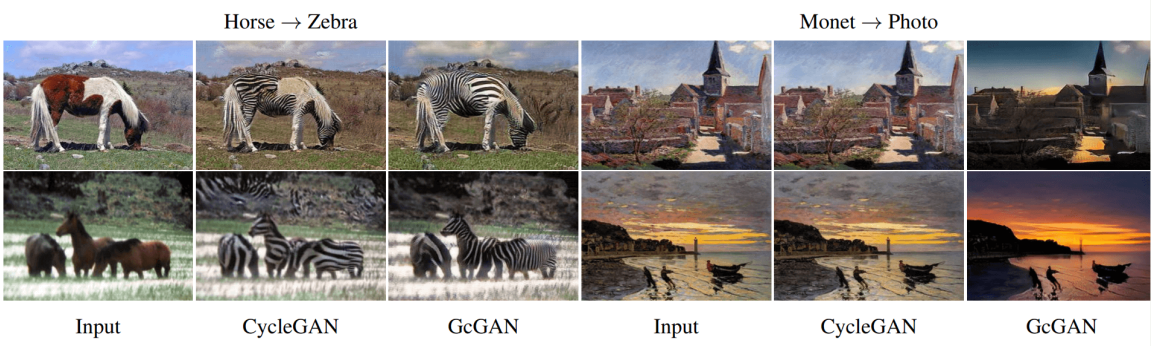

Figure 2.6: A comparison of images generated by CycleGAN and GcGAN. It is obvious that the images produced by GcGAN seem to be more realistic, yet preserving the transferred styles [24].

It is essential to mention that GcGAN is a network aimed for one-sided domain mapping. This means that we cannot use the same GcGAN model for translation tasks such as zebra-to-horse and vice-versa like it was possible in CycleGAN. Having the cycle-consistency requirement, both G_{XY} and G_{YX} are trained simultaneously. In terms of GcGAN, it is necessary to train a new model to learn inverse mapping from scratch.

Similarly, the one-sided domain mapping problem was resolved in DistanceGAN by maintaining the distances between multiple images within domains. DistanceGAN tackles the mapping problem via distance-consistency loss [8]. Still, GcGAN claims better results, concerning classification accuracies, than DistanceGAN and CycleGAN by a notable margin (6%–7%) [24].

2.2.6 StarGAN

All previous frameworks have one thing in common. They learn the mapping between two domains only. This is a tremendous limitation because if there is a demand for mappings between three and more domains, a GAN needs to be retrained and built from the very beginning. StarGAN addresses this limitation by allowing concurrent training on multiple datasets with different domains.

If we do not consider the fact that StarGAN has just one pair of a generator and discriminator, the architecture of StarGAN remains relatively unchanged with respect to CycleGAN. The adjustments were made mainly to the loss functions and the training strategy. The objective functions to optimize the generator and discriminator are:

$$\mathcal{L}_D = -\mathcal{L}_{WGAN-GP} + \lambda_{cls}\mathcal{L}^r_{cls} \tag{2.13}$$

$$\mathcal{L}_G = \mathcal{L}_{WGAN-GP} + \lambda_{cls}\mathcal{L}^f_{cls} + \lambda_{rec}\mathcal{L}_{rec} \tag{2.14}$$

Where $\mathcal{L}_{WGAN-GP}$ stands for the improved Wasserstein GAN loss with gradient penalty [27]. $\mathcal{L}_{rec}$ is the reconstruction loss, same as in Equation 2.3, albeit without the loss connected to the second generator since the framework has one generator ($\mathbb{E}_{x,c,c'}[\|x - G_{XY}(G_{XY}(x,c),c')\|_1]$, where c' is the original domain label). $\mathcal{L}_{cls}$ is domain classification loss. The superscripts denote whether the loss corresponds to domain classification loss of real or fake images. The classification is accomplished by an auxiliary classifier[3] which is placed on top of the discriminator. The auxiliary classifier enables the discriminator to control a set of domains. Finally, the generator's objective is to minimize the number of correctly classified images in a particular domain. λ_{cls} and λ_{rec} are parameters that control the importance of the computed loss functions.

To learn from multiple datasets, StarGAN incorporates the knowledge of different types of class labels. The issue is that the label information is known just partially. The complete information is however needed by the generator during the training. To alleviate such a problem, a one-hot[4] mask vector is used. It allows StarGAN to neglect unspecified labels and focus on labels that come from a real dataset. When training on 2 datasets, the mask vector is represented by a concatenated list of 2 class labels [15].

StarGANv2

In 2020, a new version of StarGAN was published. StarGANv2 attempts to resolve the main issue from which StarGAN suffered. That is, learning a deterministic mapping of each domain. This did not allow StarGAN to capture the multi-modal nature of real data distribution since every domain had to be explicitly labelled by a predetermined label.

StarGANv2 uses extra two modules to mitigate the problem mentioned above. First, the mapping network learns how to convert Gaussian noise into a style code. Second, the style encoder network studies how to extract the style code from an image. The generator, therefore, learns to generate synthetic images over various domains by utilizing these style codes [16].

MetalGAN

A relevant approach was also made by Fontanini et al. [23]. In their framework, called MetalGAN, they use meta-learning techniques to solve the problem of multi-modal training. The meta-learning suggests training a system that trains other learning subsystems. Furthermore, traditional training settings require retraining the model when a new domain is added to the target outputs. StarGAN and StarGANv2 experience this difficulty as well. On the contrary, MetalGAN promises to handle this situation by seeing just a few examples from the new domain (this is referred to as few-shots learning).

The training procedure is as follows: at the beginning, for each meta-iteration, there is a single task selected. A task is a group of images that belong to one domain. The network is trained on that single task for a specific number of internal iterations. In the

[3] An auxiliary classifier network outputs a class label c for training data which forces the StarGAN's discriminator to acknowledge additional information.

[4] A one-hot vector is a representation of categorical variables as segregated binary vectors

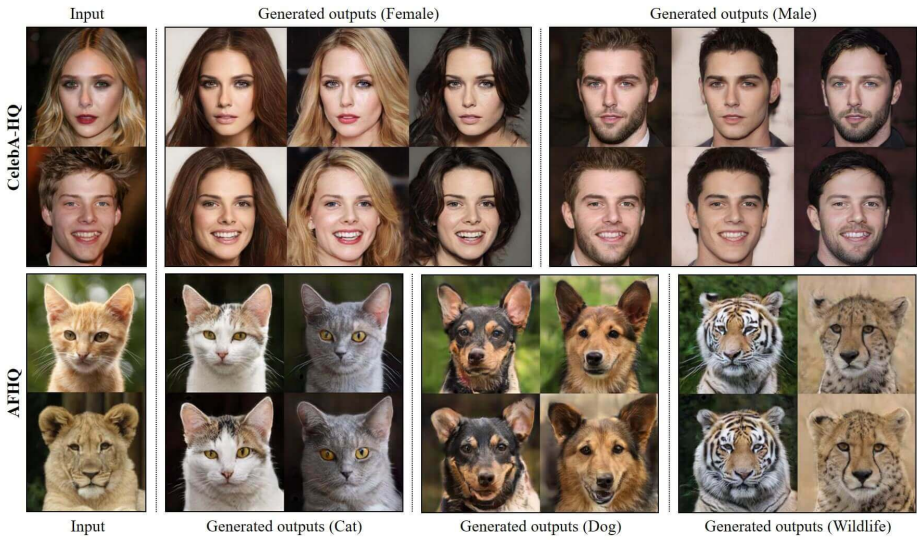

Figure 2.7: Images produced by StarGANv2. The GAN was trained on the CelebA-HQ dataset and the AFHQ dataset [42][16].

next meta-iteration, MetalGAN is trained on a different, but still related task. A final push is necessary to move the final representation in the optimal direction of the target task. This is managed by the inference part where the generator and discriminator are fed with images from new domains. As a result, MetalGAN produces images comparable to StarGAN (see Figure 2.8). Nevertheless, StarGANv2 has proven to create significantly better outputs with regards to image quality, as shown in Figure 2.7.

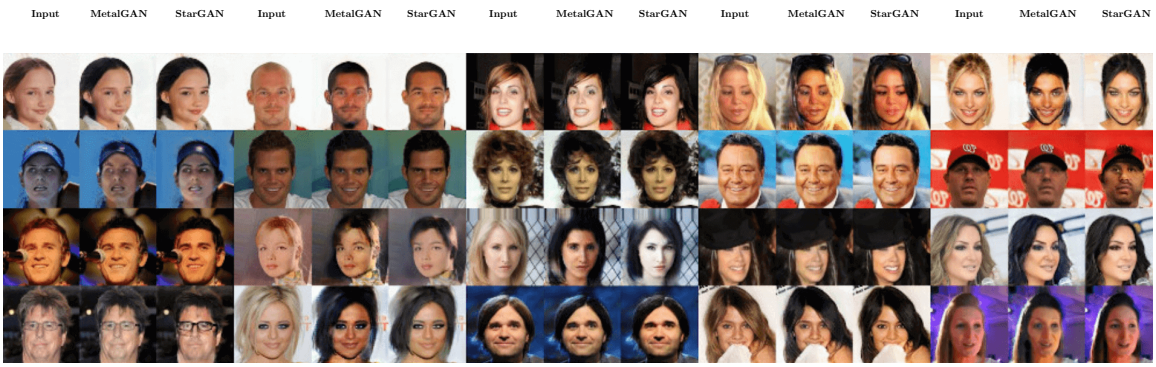

Figure 2.8: Outputs of MetalGAN and StarGAN for the black hair domain [23].

2.2.7 GANHopper

When heterogeneous domains show a significant disparity of shapes, a proper translation function is expected to be complex and may not be properly learned. We already revealed the frameworks GANimorph and GcGAN that somehow cope with this problem. Some frameworks resort to additional latent space translations. For instance, TransGaGa is a disentangle-and-translate framework that broadens awareness of the geometric properties within the scene. Instead of directly translating an image into another domain, the image is

disentangled into the Cartesian product geometry and appearance latent spaces first. After that, the translation is made based on the geometry and appearance space separately [76].

GANHopper is a framework that adopts the architecture of CycleGAN and does not rely on latent space translations. Its uniqueness lies in the fact that the output images are not translated in a single pass. GANHopper enforces gradual intermediate translations. This can be viewed as a decomposition of the overall translations which have to be made.

In general, it is assumed that a full translation is achieved by a given number of h hops. If $h = 3$, then $G(G(G(x))) = y'$ for the input image x, the generator G, and the synthetic image y'.

Regarding the training schema, GANHopper's loss is a sum of:

- $\mathcal{L}_{cyc}(G_{XY}, G_{YX}, h)$. The total cycle-consistency loss composed of cycle-consistency losses computed per individual hop.
- $\mathcal{L}_{GAN}(G_{XY}, G_{YX}, D_X, D_Y, X, Y, h)$. The standard GAN loss summed over the defined number of h hops.
- $\mathcal{L}_{dom}(G_{XY}, G_{YX}, D_H, X, Y, h)$. The hybrid loss describes the degree to which a translated image belongs to one of two domains. In GANHopper, a third discriminator (the hybrid-discriminator) assesses the correspondence to each domain. Having the constant h equal to 4, then the result of the first hop should be judged for $G_{XY}(x)$, where $x \in X$, as belonging 25% to the domain Y.
- $\mathcal{L}_{smooth}(G_{XY}, G_{YX}, h)$. The smoothness loss penalizes the difference between multiple hops regarding the values of image pixels.

Figure 2.9 displays the quality of generated images compared to CycleGAN, DiscoGAN, and GANimorph, eventually. The images produced by GANHopper perform fairly better on the dog-to-cat translation task. Also, the authors of GANHopper stress that the framework outperforms other baseline models in both quantitative analysis and human evaluation experiments [45].

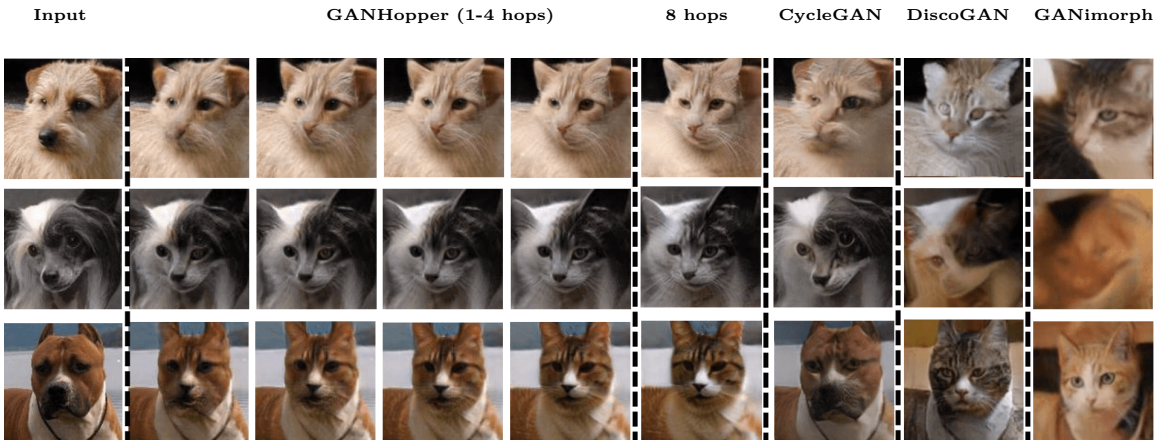

Figure 2.9: A comparison of synthetic images created by GANHopper, CycleGAN, DiscoGAN, and GANimorph [45].

2.2.8 SPA-GAN

Image-to-image translation methods need to seek for the areas which have to be transformed. Neither GcGAN nor GANimorph localizes the areas of interest. They use different procedures that often fail when input images contain numerous target instances in a cluttered background. InstaGAN addresses these issues with grace. However, it requires pixel-wise annotations to precisely differentiate between the background and instances. For applications where such information is not available during training, it poses a substantial limitation [53].

SPA-GAN tries to surpass the problem of InstaGAN. Moreover, it utilizes the attention mechanism embedded directly in the GAN architecture. The SPA-GAN architecture is almost the same as the CycleGAN architecture. In addition to that, the discriminators compute normalized attention maps that are looped back to the generators to help them focus more on the most distinctive regions.

Suppose F_i is the i-th activation map of a discriminator's layer and C indicates the number of channels. A spatial attention map is characterized by $A_D = \sum_{i=1}^{C} |F_i|$. So, A_D implies the actual effect of the neurons at each spatial location in classifying the input image.

Every attention map in the discriminator focuses on particular features. Middle layer attention maps may have higher activations on regions like eyes, while high-level attention activations can correspond to entire faces. The attention maps are supplied to the generators via element-wise product, i.e., $x' = G(x_a) = G(A_{D_X}(x) \odot x)$, for the input sample x retrieved from the domain X.

SPA-GAN's full loss is defined as a sum of two $\mathcal{L}_{GAN}$ losses (Equation 2.1), $\mathcal{L}_{cyc}$ (Equation 2.3), and $\mathcal{L}_{fm}$ which aims to preserve domain specific features between the translations.

Let G^i be the i-th feature map and C be the number of these feature maps per a generator's layer. Let y_a' be the attended generated sample formed by the generator G_{XY}. Then as well, let x_a' have the similar meaning. The feature map loss is calculated as follows [19]:

$$\begin{aligned} \mathcal{L}_{fm}(G_{XY}, G_{YX}) = \frac{1}{C} \sum_{i=1}^{C} (\|G^i_{XY}(x_a)) - G^i_{XY}(y_a')\|_1) \\ + \frac{1}{C} \sum_{i=1}^{C} (\|G^i_{YX}(y_a) - G^i_{YX}(x_a')\|_1) \end{aligned} \tag{2.15}$$

Figure 2.10 below shows the efficacy of SPA-GAN. The outputs of SPA-GAN are also compared to Attention-GAN. Attention-GAN decomposes the generator into two networks where the first network predicts the regions of interest and the second network transforms an image from one domain to another. The attention network requires supervised learning [13]. SPA-GAN is a lightweight framework that does not need additional attention networks and segmentation labels during training.

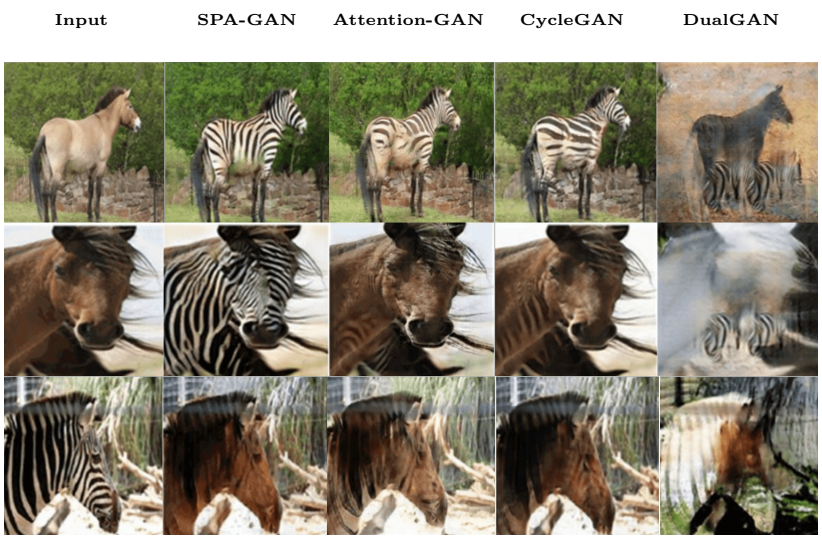

Figure 2.10: A qualitative comparison of SPA-GAN, Attention-GAN, CycleGAN, and DualGAN [19].

2.2.9 TraVeLGAN

Despite the advances in the area of style transfers, many networks are successful in mapping local texture but are typically doomed when translation tasks require large shape changes.

TraVeLGAN can better handle mappings between complex and heterogeneous domains. The framework completely eliminates the need for coupling generator weights or cycle-consistency. The translation process is tailored by a third network called *siamese* that ensures similarity between original and generated images [3].

The authors of TraVeLGAN adopted the concept of a transformation vector between two points. Similarly to language processing applications where a vector within a given space dictates how one word should be transformed to another word, here the transformation vector is used for governing the background colour, size, or shape of an image during the translation.

The approach described in the TraVeLGAN paper is related to that proposed in DistanceGAN. In DistanceGAN, pairwise distances are calculated directly on the pixel space. Eventually, this does not preserve any notion of directionality in the space between images. Therefore, it is crucial to not rely on pixel space arithmetic.

TraVeLGAN uses three kinds of losses. The standard GAN loss, siamese loss, and TraVeL loss. Formally, given a generator G, siamese network S, distance metric $Dist$, and δ signifying the minimum distance between the points in the latent space, the TraVeL and siamese losses are defined in Equations 2.16 and 2.17.

$$\begin{aligned} \mathcal{L}_{TraVeL} &= \sum_{i \neq j} \sum Dist(v_{ij}, v'_{ij}) \\ v_{ij} &= S(x_i) - S(x_j) \\ v'_{ij} &= S(G(x_i)) - S(G(x_j)) \end{aligned} \tag{2.16}$$

$$\mathcal{L}_{S_c} = \sum_{i \neq j} \sum max(0, (\delta - \|v_{ij}\|_2)) \tag{2.17}$$

G and S networks are jointly trained in the sense that each is trying to minimize $\mathcal{L}_{TraVeL}$ in addition to their specific goal. Their final objective terms are:

$$\mathcal{L}_S = \mathcal{L}_{S_c} + \mathcal{L}_{TraVeL} \tag{2.18}$$

$$\mathcal{L}_{S_c} = \mathcal{L}_{GAN} + \mathcal{L}_{TraVeL} \tag{2.19}$$

The generator implements the standard U-Net architecture with skip connections. Both the discriminator and siamese network have the same PatchGAN architecture differing only at the output layer. The siamese network has the output space of size 1000 while the discriminator network of size 1.

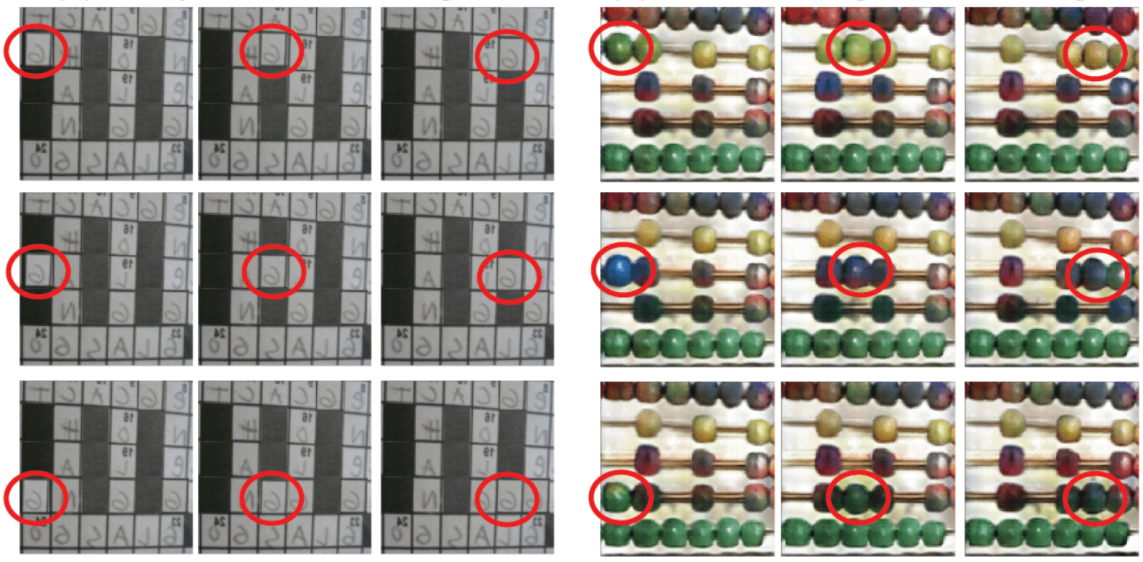

Figure 2.11: TraVeLGAN trained on images of crosswords and abacus. The learnt mapping function allows the framework to properly translate the crosswords into the seconding domain while still considering semantic changes [3].

2.2.10 U-GAT-IT

GAN frameworks deal with the image-to-image translation problem in various ways. They often tend to pile up regularization techniques and add more restrictions to the training process. For instance, UNIT enforces the shared-latent space constraint for the generators which implies the cycle-consistency constraint. Like in CycleGAN, this method does not resolve complex shape transfiguration [46]. TransGaGa deals with geometry and appearance without considering the foreground and background of images. SPA-GAN separates the foreground by leveraging the attention mechanism. Some works utilize instance normalization techniques to normalize irrelevant styles from images, like MUNIT [32]. Such normalization helps with the recognition performance of transferable regions [54]. U-GAT-IT combines the attention mechanism with improved adaptive normalization. This makes U-GAT-IT perform well on different datasets without changing hyperparameters or the model's architecture.

In U-GAT-IT, attention maps are obtained by two auxiliary classifiers which are embedded into the generator and discriminator. Thanks to that, the generator can focus

more on areas that distinguish two domains, while the discriminator can better examine fake and real images. The classifiers are trained to learn the weights of feature maps from source domains by using the global max pooling (the maximum of elements across tensor's dimensions) and global average pooling (the mean of elements across tensor's dimensions). Overall, U-GAT-IT has a specialized CAM loss function for exploiting the information from the auxiliary classifiers.

U-GAT-IT uses an identity consistency constraint which ensures that the color distributions of input and output images are similar. For a translated image $y' \in Y'$, the identity loss is defined as $\mathcal{L}_i = \mathbb{E}_{y' \sim Y'}[\|y' - G_{XY}(y')\|_1]$.

The adaptive layer-instance normalization (AdaLIN) is applied to the residual decoder blocks of the generator. It adjusts the management of layer normalization and instance normalization adaptively based on the input and output domain distributions. The layer normalization aims for better transformation to the target domain. The instance normalization is capable of preserving properties of the source domain [39].

Below, we attach examples of pictures generated by U-GAT-IT and the GANs that were introduced previously.

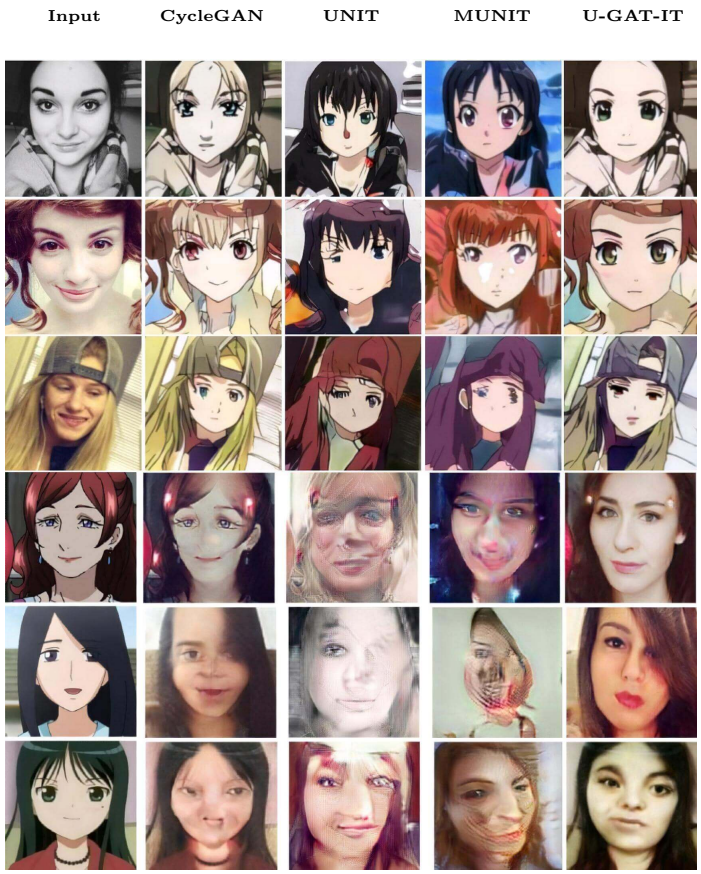

Figure 2.12: A visual comparison of the images generated in the selfie-to-anime and anime-to-selfie tasks [39].

2.3 Comparison of the Architectures

Tables 2.1 and 2.2 compares the introduced architectures in terms of loss functions, network architectures, and mapping types. Even though many architectures dispose of the same architecture, they are not identical. For example, GANimorph uses dilated convolutions in the discriminators. U-GAT-IT has the additional classifier connected directly to the encoder followed by the additional instance normalization. For the sake of simplicity, these details were omitted in the comparison.

Table 2.1: A comparison between the CycleGAN, DualGAN, DiscoGAN, GANimorph, and GcGAN architectures.

	CycleGAN	DualGAN	DiscoGAN	GANimorph	GcGAN
Year	2017	2017	2017	2018	2019
Loss function	LSGAN + CCL	WGAN + CCL	GAN + RL	GAN + CRL + FL	LSGAN + GL
Network architecture (generator & discriminator)	ResNet & PatchGAN	U-Net & PatchGAN	Encoder-decoder pair & Encoder	ResNet & PatchGAN	ResNet & PatchGAN
Mapping	Two-sided	Two-sided	Two-sided	Two-sided	One-sided

Table 2.2: A comparison between the StarGAN, TraVeLGAN, GANHopper, SPA-GAN, and U-GAT-IT architectures.

	StarGAN	TraVeLGAN	GANHopper	SPA-GAN	U-GAT-IT
Year	2018	2019	2020	2020	2020
Loss function	WGAN-GP + DCL + CCL	GAN + TL + SML	GAN + CCL + HL + SL	GAN + CCL + FML	LSGAN + CCL + CAM + IL
Network architecture (generator & discriminator)	ResNet & PatchGAN	U-Net & PatchGAN	ResNet & PatchGAN	ResNet & PatchGAN	ResNet + AdaLIN & PatchGAN
Mapping	Multi-sided	One-sided	Two-sided	Two-sided	Two-sided

The abbreviations for loss functions presented in the tables have the following significance: GAN is standard GAN loss, WGAN is Wasserstein GAN loss, WGAN-GP is Wasserstein GAN loss with gradient penalty, CCL is cycle-consistency loss, RL is reconstruction error loss, CRL is cyclic reconstruction loss, FL is feature matching loss, DCL is domain classification loss, TL is TraVeL loss, SML is siamese loss, HL is hybrid loss, SL is smoothness loss, FML is feature map loss, LSGAN is least squares GAN, CAM is class activation mapping loss, and finally, IL is identity loss.

Chapter 3

Privacy Preservation

When it comes to biometric data, they are considered to be extremely sensitive. The analysis of sensitive data is more and more demanding for various reasons. Storing and managing biometric data poses serious privacy risks. In this chapter, two famous techniques for preserving privacy are advertised. Concretely, we explain the ideas behind differential privacy (DP) and homomorphic encryption (HE).

3.1 Differential Privacy

Differential privacy (DP) allows researchers or analysts to operate with datasets that contain sensitive data while offering stronger privacy protections. Simply put, differential privacy ensures that when an item is added or removed from a database, it does not affect a query's outcome. This is done by introducing statistical noise to the query's response. With DP, the privacy of an individual is protected, yet it does not have a significant impact on the accuracy of the query's response.

3.1.1 ϵ-Differential Privacy

Assume two databases D_1 and D_2. Supposing that D_1 differs in at most one element from D_2, then D_1 is an exact subset of D_2 and the database D_2 has only one additional row. A randomized mechanism $\mathcal{M}$ gives ϵ-DP if for the databases D_1 and D_1, differing at most in one element, and all $S \subseteq Range(\mathcal{M})$, it satisfies Equation 3.1.

$$Pr[\mathcal{M}(D_1) \in S] \leq e^{\epsilon} Pr[\mathcal{M}(D_2) \in S] \tag{3.1}$$

Where the probability is taken over the randomness used by the mechanism $\mathcal{M}$. For a query function $f : \mathcal{D} \to \mathcal{R}$ and a database X, the privacy mechanism $\mathcal{M}$ corresponds to:

$$\mathcal{M}_{Lap} f(X) \triangleq f(X) + (Lap(\triangle f/\epsilon)) \tag{3.2}$$

Here, $\triangle f$ is the sensitivity of the function f and it is equal to $max_{D_1,D_2} \|f(D_1) - f(D_2)\|_1$. The Laplace mechanism is an essential ϵ-differentially private algorithm [18].

DP also depends on ϵ. The parameter ϵ in Equation 3.1 is used to quantify the privacy risk that can occur when releasing (noisy) answers computed on private data. Given a sufficiently low ϵ, an adversary's ability to identify an individual is minimal. On the other hand, the accuracy of the answers decreases by lowering ϵ too much.

One way of determining ϵ is to clarify the scenario in which DP will be held. In the model where two parties share conflicting views about how data will be used, the value of ϵ is the result of a trade-off between these conflicting objectives [30]. In the case of a face recognition system, a compromise is made between the system's accuracy and users' privacy.

ϵ-DP is closed under composition and the ϵ parameters of composed mechanisms add up [87]. This motivates the concept of a privacy budget introduced in Section 3.1.4.

3.1.2 (ϵ, δ)-Differential Privacy

(ϵ, δ)-DP is a relaxation of ϵ-DP and it is defined in the following equation:

$$Pr[\mathcal{M}(D_1) \in S] \leq e^{\epsilon} Pr[\mathcal{M}(D_2) \in S] + \delta \tag{3.3}$$

In ϵ-DP, only a small amount of information about each individual is possible to obtain because all outputs may occur with a similar probability. The additive term δ in Equation 3.3 covers two modes in which privacy could fail. In the first mode, a secret becomes completely exposed with a probability δ. In the second mode, privacy degrades gracefully when multiple queries are made. In ϵ-DP, the simultaneous release of randomized mechanisms $\mathcal{M}_1$ and $\mathcal{M}_2$ satisfies the basic composition theorem, i.e., $(\epsilon_{\mathcal{M}_1} + \epsilon_{\mathcal{M}_2})$-DP. However, the privacy decreases with the probability $\delta/2$ for ϵ_1-DP and $\delta/4$ for ϵ_2-DP in (ϵ, δ)-DP.

The value of δ is recommended to be less than $1/N$, for a specific number N of examples in the database. The notion behind this is that the notably small value avoids the worst-case scenario of violating privacy.

For (ϵ, δ)-DP, the Gaussian mechanism is a prototypical mechanism and has the next significance:

$$\mathcal{M}_{Gauss} f(X) \triangleq f(X) + N(0, \sigma^2) \tag{3.4}$$

In Equation 3.4, σ is the standard deviation of the Normal distribution. The Gaussian mechanism does not meet ϵ-DP for any ϵ if $\delta \neq 0$. Yet, it maintains (ϵ, δ)-DP for all combinations of $\epsilon < 1$ and $\sigma > \sqrt{2ln1.25/\delta}\triangle_2 f/\epsilon$. The sensitivity for the query function f is defined as $\triangle_2 f \triangleq max_{D_1,D_2} \|f(D_1) - f(D_2)\|_2$. The subscript $_2$ symbolizes the L2 norm which is known as the Euclidean norm.

There exist two reasons why researchers lean towards (ϵ, δ)-DP. One is the Gaussian mechanism. And the second is the application of the advanced composition. The goal of the advanced composition is to calculate parameters ϵ' and δ' for k-mechanisms, where each mechanism is (ϵ, δ)-differentially private, such that the composition of these mechanisms satisfies (ϵ', δ')-DP.

For any $\delta' > 0$, the advanced composite mechanism can be expressed as (ϵ', $k\delta + \delta'$)-DP, where ϵ' has the following definition:

$$\epsilon' \triangleq \sqrt{2kln(1/\delta')}\epsilon + k\epsilon(e^{\epsilon} - 1) \tag{3.5}$$

In the Gaussian mechanism, the noise comes from the same distribution as the error present in the database. Furthermore, the mechanism is closed under addition. The advanced composition and the Gaussian mechanism do not compose with each other. This means that when we take the Gaussian mechanism and apply it iteratively multiple times, we will get the Gaussian mechanism again [51].

3.1.3 (α, ϵ)-Rényi Differential Privacy

Rényi DP (RDP) is based on the concept of the Rényi divergence which is related to the Rényi entropy. RDP is a generalization of ϵ-DP, but it is conceptually weaker. (∞, ϵ)-RDP is basically ϵ-DP. On the other hand, RDP leads to stronger bounds compared to (ϵ, δ)-DP.

A randomized mechanism $\mathcal{M}$ has (α, ϵ)-RDP if for any of its distribution over two adjacent databases D_1 and D_2, it holds that [87]:

$$D_\alpha(\mathcal{M}(D_1)\|\mathcal{M}(D_2)) \triangleq \frac{1}{\alpha-1} log \mathbb{E}_{x\sim\mathcal{M}(D_2)} \left[\left(\frac{p_{\mathcal{M}(D_1)}}{p_{\mathcal{M}(D_2)}} \right)^\alpha \right] \leq \epsilon \tag{3.6}$$

Where D_α is the Rényi divergence. The optimal value for the order α ranges from 1 to $+\infty$. According to the experiments performed by Mironov in the paper [51], even a restricted set of values (i.e., 1.5, 1.75, 2, 2.5, 3, 4, 5, 6, 8, 16, 32, 64, $+\infty$) preserves the privacy guarantees.

When accessing the database numerous times via differentially private mechanisms, RDP promises to preserve privacy on the union of the answers. The composition in RDP produces more favourable privacy parameters than the advanced composition for (ϵ, δ)-DP. Simultaneous release of (α, ϵ_1)-RDP and (α, ϵ_2)-RDP is (α, $\epsilon_1 + \epsilon_2$)-RDP.

To conclude, RDP fixes the issues of (ϵ, δ)-DP while allowing easy compositions of heterogeneous mechanisms. RDP also provides a convenient and accurate way of tracking cumulative privacy loss because the privacy curves for composed mechanisms add up easily [51].

3.1.4 Privacy Accounting

A contributor who commits private data to the database needs to be compensated for the participation. Due to that, an analyst may be able to conduct only a limited study until a so-called privacy budget spent on the study exceeds. The privacy budget refers to a maximum privacy leakage allowed for all queries. Thus, the privacy budget relates to ϵ in DP. A privacy accountant implements privacy accounting. Its main task is to track privacy consumption during the analysis.

In this thesis, we exclusively focus on aspects of DP in deep learning. While training neural networks, it is inevitable to feed the networks with thousands of representative examples. Regularization techniques that aim to avoid overfitting can hide the details of the examples. However, it is still unknown whether the internal representations of the networks could encode fine details at least for some examples.

An adversary can extract training data via the model inversion attack or membership inference attack. In the model inversion attack, the adversary tries to deduce the information about the data from the prediction scores. This type of attack does not result in producing actual member's data nor gaining knowledge about her presence in the data. The membership inference attack connects to the problem of identifying the presence of the individual's data [66].

Models that employ DP during training limit the probability of successful attacks. Abadi et al. [2] proposed a differentially private stochastic gradient descent algorithm (DPSGD) to cope with the attacks mentioned earlier. Each gradient in the L2 norm is clipped by a given threshold value that should be set separately for each layer within a network. The authors stress that the clipping needs to be done before averaging. In DPSGD, the accountant accumulates the privacy cost for every access to the training data. The accumulated cost corresponds to all layers in a trained network and determines if a privacy budget exceeds.

Measuring the achieved privacy guarantee in deep learning is feasible in many machine learning frameworks. TensorFlow Privacy[1] is a framework that handles the accounting required for computing a spent privacy budget. For instance, Listing 3.1 illustrates how TensorFlow Privacy can be used.

```
from tensorflow_privacy.privacy.optimizers.dp_optimizer_keras import (
    DPKerasSGDOptimizer,
)
from tensorflow_privacy.privacy.analysis.rdp_accountant import (
    compute_rdp,
    get_privacy_spent,
)

def compute_epsilon(steps, batch_size, dataset_size, target_delta):
    orders = [1 + x / 10.0 for x in range(1, 100)]
             + list(range(12, 64))
    sampling_probability = batch_size / dataset_size
    # compute RDP of the Sampled Gaussian mechanism for each order
    rdp = compute_rdp(sampling_probability, sigma, steps, orders)
    # compute epsilon given the list of RDP values and the target delta
    return get_privacy_spent(orders, rdp, target_delta=target_delta)[0]

optimizer = DPKerasSGDOptimizer(
    l2_norm_clip=1.0,
    noise_multiplier=sigma,
    num_microbatches=batch_size,
    learning_rate=0.15,
)

# train a simple CNN model on the MNIST dataset with
# the optimizer instantiated by the class DPKerasSGDOptimizer

# the MNIST dataset has 60_000 examples (delta << 1/60_000)
delta = 1e-5

steps = epochs * dataset_size // batch_size
epsilon = compute_epsilon(steps, batch_size, dataset_size, delta)
print(f"The current epsilon for delta '{delta}' is {epsilon}")
```

Listing 3.1: Computing the expended privacy budget with TensorFlow Privacy by leveraging the RDP accountant.

GAN architectures require further modification to the existing codebase. GANobfuscator utilizes a gradient pruning strategy that improves the stability and scalability of the framework. In general, GANs are known for the issues related to instability during training. Incorporating DP in GANs worsens these issues. In GANobfuscator, the model requires access to a small amount of public data (available in many settings). The pruning bounds are tuned based on the average gradient norms computed on the public data. Regarding

[1] https://github.com/tensorflow/privacy/

privacy, the privacy accountant is updated after each access to private data. Then, the cumulative privacy loss is measured by the end of the training step and the learning stops automatically when the privacy budget exceeds [77]. In the end, the concept introduced in Listing 3.1 therefore remains almost the same.

A contrasting approach was made by Zhang et al. [83]. They proposed a GAN framework (PPGAN) which preserves privacy by leveraging a mechanism that perturbs the objective function. In particular, Laplace noise is injected into the coefficients of the objective functions. Thanks to that, a privacy budget is not accumulated in each generator step and remains unaffected by the cardinality (size) of the training set. The model is reliable and can generate real-like synthetic data while not disclosing sensitive information.

3.2 Homomorphic Encryption

In standard cryptography, information is encrypted, transmitted, and then decrypted for further manipulation. Homomorphic encryption (HE) is a form of encryption that supports computations on encrypted data without decrypting them first. Therefore, it is not required to possess a secret key to execute computations on the receiver's side.

3.2.1 Basic Concepts

A homomorphism is a structure-preserving map between two groups. Let's assume two groups $(G, \cdot)$ and $(H, *)$. The structure-preserving map between them is defined as $m : (G, \cdot) \rightarrow (H, *)$. The map is additively homomorphic if we consider addition operations and multiplicatively homomorphic if we consider multiplication operations. For an encryption function E and a set of plaintexts P, a scheme is homomorphic when it satisfies:

$$\forall p_1, p_2 \in P : E(p_1 \cdot p_2) \leftarrow E(p_1) * E(p_2) \tag{3.7}$$

Where $\leftarrow$ means that the right part can be directly computed without intermediate decryption [22]. Figure 3.1 demonstrates simple operations performed on different objects with regards to HE.

There exist three types of HE: partially HE (PHE), somewhat HE (SHE), and fully HE (FHE). In PHE, only one operation can be executed on encrypted data. The unpadded RSA cryptosystem is multiplicatively homomorphic respecting PHE. Executing more than one operation, but in a limited fashion, is allowed in SHE. FHE can compute any function without limitations. The only advantage of PHE and SHE over FHE is that these HE techniques are more efficient in their processes [56].

The standardized FHE is based on three models of computations. The models represent boolean circuits, modular (exact) arithmetic, and approximate number arithmetic. The boolean circuits model operates with encrypted bits. The modular arithmetic approach evaluates arithmetic circuits for values encrypted by modulo an integer i. Finally, the approximate number arithmetic computation produces almost accurate results. This technique is suited for a fast polynomial approximation and floating-point calculations [75].

3.2.2 Machine Learning Usage

Homomorphic methods may be applied for privacy-preserving deep learning as well. Since only certain mathematical operations, like addition and multiplication, are homomorphic, the advanced encryption standard (AES) cannot be used. Instead, discrete convolutions

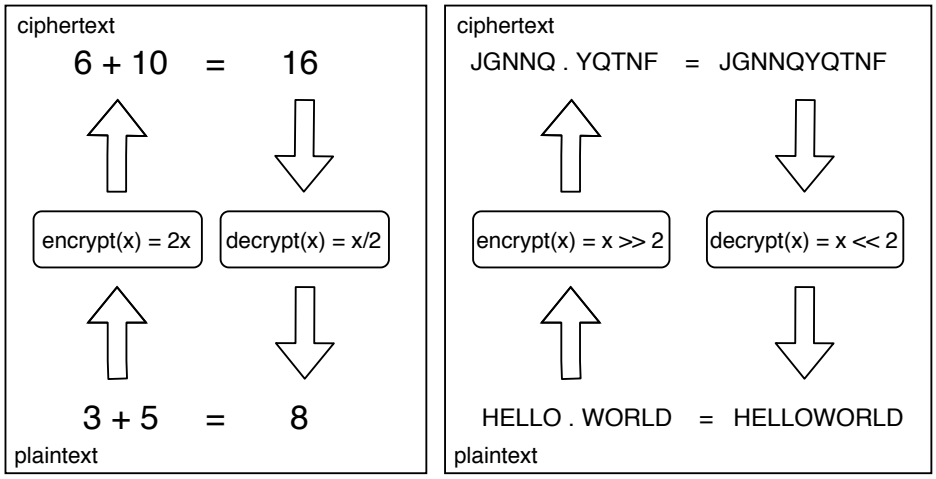

Figure 3.1: A demonstration of HE. On the left, the addition of two unencrypted integers yields the same output as the decrypted addition of two encrypted values. On the right, a concatenation of two words is performed. Following the analogy from the example on the left, the operation results in the same value after decryption.

can be computed using FHE because the convolutions are expressed as polynomials. After training, the kernel weights for every convolution layer may be encrypted. The computation is simple and relatively stable for mean pooling layers. Fully connected layers are described by polynomial functions and therefore can be evaluated using FHE too.

The activation layers introduce non-linearity to the model. In terms of activation layers, we need to select their homomorphic alternatives properly. For instance, CryptoNets,[2] which demonstrates the use of neural networks over encrypted data by leveraging SEAL,[3] replaces ReLU with the square activation function, i.e., $f(x) = x^2$. However, the authors of CryptoNets stress that networks deeper than ten layers may experience difficulties because the square activation function can yield unstable behaviour during the backpropagation. Following the CryptoNets' solution, Chabane et al. [11] incorporate a low-degree polynomial approximation of ReLU together with batch normalization.

Yet, the usage of HE in machine learning is still problematic. Noise is associated with each conversion from plaintext to ciphertext. This noise continues to expand after applying homomorphic operations. Uncontrollable accumulation of the noise leads to indecipherable ciphertext. To deal with this issue, one can use a bootstrappable SHE which decrypts and re-encrypts noisy ciphertext. The decryption algorithm is fed with the encrypted version of the decryption key. Such a procedure has shown that the new ciphertext contained less noise and was as secure as the original ciphertext. Briefly, the bootstrapping homomorphically refreshes ciphertext [56].

Although the approach mentioned in the previous paragraph (also known as the Gentry's approach, where an FHE scheme can be built from a SHE scheme) seems to be an ultimate solution to the noise problem, its underlying implementation is impractical for real-world deployment because of slow run times. The current generation of HE systems continues to increase the overall performance. Many libraries which employ specific schemes are now

[2] `https://github.com/microsoft/CryptoNets`
[3] `https://github.com/Microsoft/SEAL`

publicly available. Some of them lack bootstrapping and therefore cannot be considered as FHE, e.g., SEAL.

TF Encrypted[4] is a framework for machine learning. It enables developers to design and train models over encrypted data while taking advantage of the TensorFlow API. Similarly, TF SEAL[5] creates a bridge between TensorFlow and SEAL. The frameworks are still considered to be in the Beta stage and demand more attention from the core developers. The main disadvantage of these frameworks is that they do not support the newer versions of TensorFlow 2.0.

3.3 Protecting Biometric Data

Biometric technology has proven to be more beneficial than standard authentication and identification systems built upon passwords, keys, tokens, or access codes. The biggest issue of biometric-based systems is that nothing stops an attacker from making replay attacks once she obtains biometric templates.

In previous sections, we gave the reader a short introduction to DP and HE. Both DP and HE provide privacy guarantees. While DP is a system that allows sharing knowledge about a dataset without disclosing information of a single individual, HE operates merely on encrypted data and so protects the privacy of each user. These methods may eliminate possible attacks when it comes to biometric data.

In the upcoming two sections, a couple of works that try to resolve the problem of securing biometric data are presented.

Differential Privacy

In machine learning, the goal of DP is to give privacy protection for the training dataset. Most of the defence mechanisms are arranged against membership inference attacks. In deep learning models, DP can be applied to these three locations [84]:

- **Input layers**. Where synthetic data are generated from sensitive data.
- **Hidden layers**. Where noise is injected into gradients (like in GANobfuscator).
- **Output layers**. Where an objective function is perturbed (like in the work [83]).

Mao et al. [49] split a neural network into two successive layers and deployed these two partitions at different locations. The first (user) part ensures (ϵ, δ)-DP. The output of the user network activations is passed to the second part placed on the edge server. Such a scheme was tested to be accurate and satisfactory in terms of face recognition. Chamikara et al. [12] proposed the method named PEEP (Privacy using EigEnface[6] Perturbation) that applies perturbations on input data. As a result, the properties of local DP alleviate privacy leaks that could occur. The said two works are easy to implement and may be deployed on resource-constrained devices.

Figure 3.2 depicts face images produced by GANobfuscator. The quality of the generated images is close to the images generated by the regular WGAN. The authors of

[4] `https://github.com/tf-encrypted/tf-encrypted`

[5] `https://github.com/tf-encrypted/tf-seal`

[6] Eigenface utilizes the principal component analysis to represent the low-dimensional version of an image. It acknowledges a predefined number of the largest eigenvectors as the principal axes.

GANobfuscator also executed the inference attack on the trained model. The attacker was not able to accurately infer the distribution of the target models when the privacy budget was small. However, when the privacy budget was too low, the overall quality started to worsen. This confirms that there is always a trade-off between utility and the privacy level in methods that utilize DP [77]. Furthermore, Bagdasaryan et al. [6] pointed out that once an original non-private model exhibits lower accuracy for underrepresented and complex classes, this gap is even bigger in the corresponding private model. Thus, the underrepresented classes' size, larger gradients computed during the backpropagation, and applied final noise have a serious impact on model accuracy.

(a) (b) (c)

Figure 3.2: Images generated by GANobfuscator. The following DP configuration was used: (a) $\epsilon = 8$, $\delta = 10^{-5}$; (b) $\epsilon = 4$, $\delta = 10^{-5}$; (b) $\epsilon = 2$, $\delta = 10^{-5}$ [77].

In the work [69], a privacy-aware virtual reality interface that utilizes DP was proposed. The method prevents user re-identification and protects soft biometric traits, i.e., eye movements. Likewise, a DP privacy-preserving framework for soft biometric-based systems was designed by Sadhya and Singh [61]. The framework preserves the privacy of traits such as age or gender while retaining the original accuracy rates for the recognition. The results were evaluated in a multi-modal environment where images of fingerprints and faces were used as primary biometric characteristics.

By the end of this section, we underline that the injected statistical noise in DP often disrupts the patterns required for recognition purposes. Therefore, it is not common to have standalone systems which administer fingerprints solely. HE is more superior in this area.

Homomorphic Encryption

Biometric data cannot be reset like passwords and PINs. This kind of data should not be exposed to a third party. One solution is to encrypt the data before releasing them to the cloud. Then, specific operations can be carried out using the ciphertext by leveraging HE. Salem et al. [62] adopted transfer learning and HE which allowed them to verify features of iris and fingerprint inputs securely. They showed that combining the features made the system more rigorous as they achieved 95.47% accuracy on the test data. Similarly, Song et al. [68] implemented a system that is partially powered by SEAL and follows FHE. Here, encrypted iris templates are sent to the cloud server during the registration phase. When

a user tries to authenticate herself, the server performs the identification under the iris ciphertext and calculates a resulting authentication code. The user decrypts the code with the secret key and sends it back to the cloud server, where the final authentication is given. The scheme is shown in Figure 3.3.

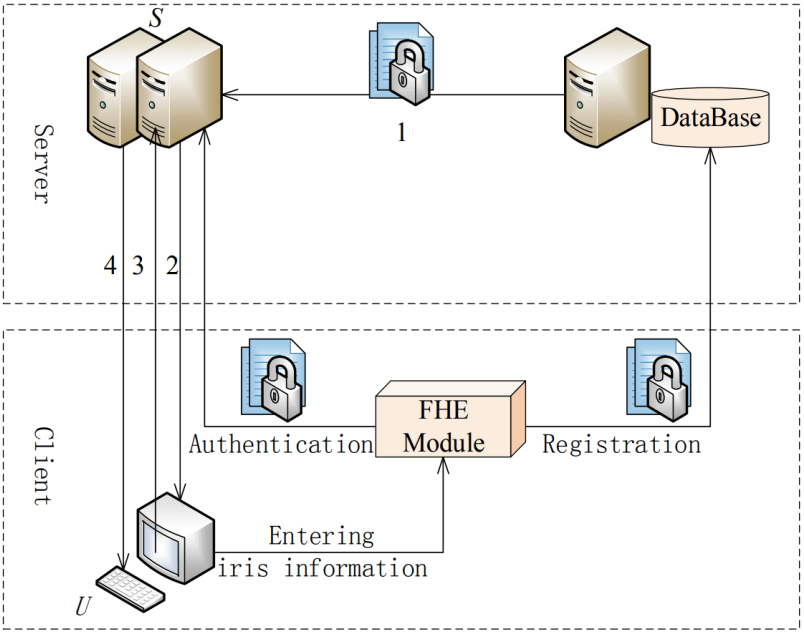

Figure 3.3: A client/server architecture arranged by Song et al. [68].

The most significant drawback of homomorphic encryption is the long execution time. By executing an algorithm in parallel, it is possible to mitigate such an issue. The parallel execution techniques for computing the Euclidean distance were applied in the work made by Catak et al. [10]. The number of parallel computations is driven by the number of available CPU cores. The solution was successfully tested on fingerprint data. The authors advocate the usage of the face and iris templates in the future.

In the previous works, only biometric data were encrypted. One could want more things to be masked. The topology of a neural network and its weights may be hidden too. To achieve this, Izabachène et al. [35] suggest replacing a classic feed-forward neural network by a Hopfield network.[7] The Hopfield neural network is better adapted to FHE. The framework recognizes faces with extraordinary accuracy in less than 0.6 seconds for fully masked and fully homomorphic classification. This is an outstanding performance in comparison to the past works.

[7] https://www.sciencedirect.com/topics/computer-science/hopfield-network

Chapter 4

Privacy Aspects of Machine Learning

All the frameworks introduced in the preceding chapters were applied in a setting where a single model is trained on a centralized dataset. However, there is also an option to incline towards techniques that do not require users to share their datasets. Such techniques take advantage of so-called collaborative learning, where training data does not leave users' devices.

In this chapter, we explore collaborative learning and compare it to centralized learning. Since both methods have their advantages and disadvantages, we also compare their performance and scalability.

4.1 Centralized Learning

Centralized learning refers to a usual way of machine learning where a neural network has access to a whole dataset. But, a collection of photos, speech, or videos gathered from multiple individuals poses tremendous privacy risks. The users from whom the data were collected can neither control how the data will be used nor delete them. Another problem presents sensitive information, such as licence plates, sounds of other people, and ambient noises that could be captured accidentally. Furthermore, in many domains related to medicine, data sharing is not permitted by law. Researchers are allowed to perform deep learning only on datasets belonging to their institutions because of that. This can result in an overfitted model that has reduced utility on other inputs [65].

Almost every mentioned issue may be resolved via DP. With DP, it is possible to synthesize input data before dispatching them to the insecure environment. In particular, this method would require training another model that generates noisy data while preserving the desired utility. Frameworks like GANobfuscator or PPGAN would be needed to be dispatched on users' devices and trained separately from scratch, which could often be unfeasible within a given time. On the contrary, all the issues can be resolved using HE. Nevertheless, HE calls for extensive refactorization of existing systems. Moreover, the high computing complexity is still an open problem even though the efficiency of HE schemes is continuously improving. Also, the length of ciphertext and computing complexity grows quickly with the growth of security strength in ordinary HE schemes. Only a few HE schemes are legitimately secure, verifiable, and thus practicable [31].

In centralized learning, privacy can be assured by combining the outputs of numerous submodels which are processed by another model afterwards. Assuming that an ensemble of teacher models trains on disjoint subsets of sensitive data and a student model attempts to mimic the ensemble based on the ensemble's aggregated outputs, privacy is ensured by the fact that the student does not learn the details characterizing the training data. In the end, training data is protected from an adversary even when she observes the student's internals.

Intuitively, the student model seems to provide reasonable privacy when trained without direct access to the training data. However, this may not suffice because even a single data point can have a negative impact on the privacy of machine learning models. Papernot et al. considered this problem and proposed the strengthened version of the teacher-user strategy. The solution is called Private Aggregation of Teacher Ensembles (PATE) [58]. See Figure 4.1 for an overview of PATE.

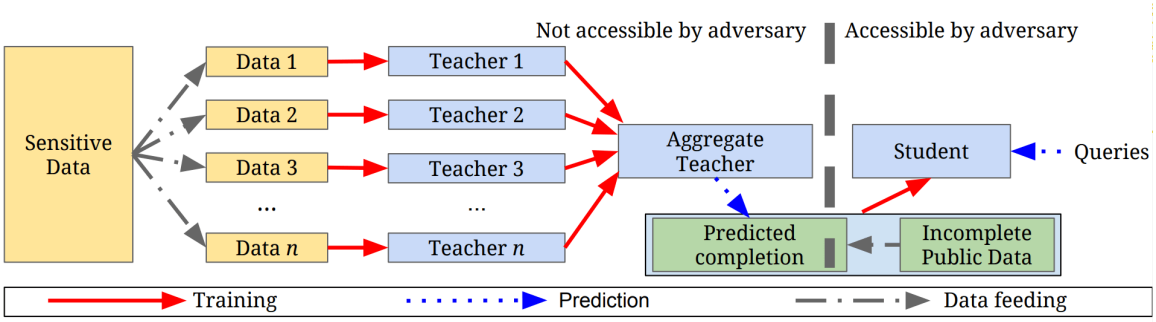

Figure 4.1: A workflow of PATE. The ensemble of teachers is trained on disjoint subsets of the sensitive data. The student model is trained on the ensemble's outputs that are associated with corresponding public data [58].

The aggregation is the part that employs DP in PATE. If an ensemble's task is to assign labels to input data, the aggregated output should not be solely affected by the prediction of one teacher. Adding random noise to the overall decision introduces ambiguity. The noise ensures that the assigned labels will be chosen randomly when the ensemble receives an equal number of votes from the teacher models. On the other hand, adding the noise to the vote counts will not change the final prediction when the label receives the most votes.

A framework consisting purely of an ensemble has two limitations. Predictions made by the aggregation increases the spent privacy budget. And the ensemble itself cannot be published because an adversary would easily extract the internal parameters. Due to that, the student model is the final product of PATE. It may respond to any query from end-users without jeopardizing the privacy because the knowledge acquired by the ensemble is transferred in a privacy-preserving manner.

Papernot et al. improved the original PATE mechanism by leveraging new techniques that are more selective and add less noise to the aggregated teachers' predictions. During the evaluation, the mechanisms performed better than the original PATE [59]. Moreover, the improved PATE mechanism was utilized in the GAN framework named PATE-GAN, which guarantees privacy for synthetic data that are based on real data [38].

Since teacher models are always trained independently, this might be useful in scenarios where datasets are owned by a group of people who do not want to share sensitive information but still wish to contribute to the learning process. Such scenarios are evaluated in Section 4.2.

4.2 Collaborative Learning

The main goal of collaborative learning (also known as federated learning) is to build machine learning models trained on datasets distributed across multiple devices. Such a configuration focuses on preventing data leakage. However, in centralized learning, only the service provider can violate users' privacy. In collaborative learning, any user may intentionally compromise other users.

In this thesis, we pay attention exclusively to horizontal federated learning. Horizontal federated learning follows the scenarios where datasets share the same feature space but different space in samples. For instance, two regional banks could have distinct users concerning the regions. However, their business is closely related. Typically, in horizontal federated learning, we assume honest participants and an honest-but-curious central server [78]. Later in this section, we address settings in which the participants are honest-but-curious too.

4.2.1 Deep Neural Networks

To handle the aforesaid privacy issues in deep learning, Shokri and Shmatikov designed a system that allows multiple parties to learn local neural network models jointly. They exploited the fact that stochastic gradient descent (SGD) can be parallelized and executed asynchronously.

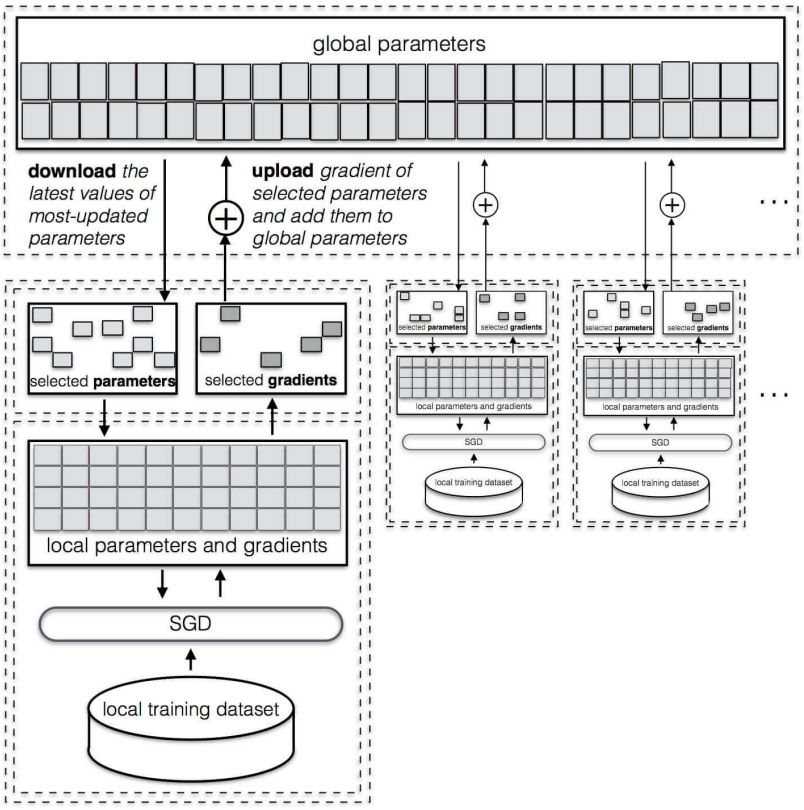

Figure 4.2: Architecture of the system designed by Shokri and Shmatikov. In the picture, there is shown a configuration where parameters are uploaded to one trusted server. The architecture itself supports direct exchange of parameters between parties as well [65].

Figure 4.2 displays how the proposed system works. Enrolled parties train their local models concurrently and independently while selectively sharing some of the model parameters, namely gradients. The parameter sharing approach enables the parties to benefit from each other because the parameters obtained from different users avoid the local models being stuck in local minima. The shared parameters are summed before they are applied in local models. Furthermore, the authors utilize DP to ensure that parameter updates do not leak too much information [65].

Although DP provides protection, it is crucial to use it properly. Hitaj et al. made an excellent research [29] on investigating the drawbacks of the collaborative learning system designed by Shokri and Shmatikov. In this system, the level of granularity was not defined correctly. DP was applied to the parameters immediately and it was not robust enough against active adversaries.

The adversary developed by Hitaj et al. pretends to be an honest participant, but in the background, it manipulates the parameters which genuine participants share. Since all nodes need to agree on a common learning objective in advance, the adversary has extra knowledge of the model's structure and other participants' data labels. The adversary can then influence the learning process which results in more leaked data from the genuine participants. In Listing 4.1, there is a pseudo-code illustrating how the adversary's training looks.

```
for e in epochs do
    DOWNLOAD parameters from the server
    REPLACE respective local parameters with the newly downloaded ones
    CREATE a replica of a local model that will represent
        a discriminator D within the GAN framework
    RUN a generator G on D targeting an unknown class a
    UPDATE the G parameters according to the answers from D
    ASSIGN a label b to the generated samples of a
    MERGE the generated data with the local dataset
    RUN SGD on the local dataset and update the local model D
    COMPUTE the gradient vectors and UPLOAD them to the server
end for
return a GAN capable of generating prototypical examples of the class a
```

Listing 4.1: The adversary training adopted by Hitaj et al. [29].

User or device-level DP would be efficient against the GAN-based attack devised earlier. Regarding the user or device-level DP, it is necessary to have a curator that aggregates and randomly selects a group of participants who train the global model. Selected nodes train their local models by leveraging DPSGD and dispatch the corresponding parameters back to the curator. A malicious participant cannot extract information about other participants since it is unclear who has participated in a single training round [44]. On the other hand, the authors of the paper [73] were able to recover data even when the learning scheme used the user-level DP. The proposed framework mGAN-AI resolves many issues that the GAN framework designed by Hitaj et al. had. Their attack could be easily detectable because of the significant adversarial influence on the learning process and it could infer only class-wised representatives which are generic samples identifying class properties rather than the exact samples of users.

4.2.2 Generative Adversarial Networks

In the work [71], Triastcyn and Faltings introduced FedGP, a collaborative training model in which participants cooperate to train GANs on their devices. The generators aim to replace real data with artificial data while producing samples from the cross-user distribution.

The result of the traditional collaborative learning is a single trained model which does not provide much flexibility when requirements change. Moreover, further aggregations from different sources are considered to be troublesome when trying to combine multiple federated models. The approach made in FedGP allows releasing entire datasets. This was not feasible before, and it has many advantages compared to the model release. FedGP uses the FedAvg algorithm, which is a generalization of FedSGD.[1] Listing 4.2 demonstrates the idea behind the FedAvg algorithm.

```
1  INITIALIZE models on every node randomly
2  for t in training steps do
3      run Server in parallel
4          SELECT a random subset S of P participants
5          SEND current weights w^t to S
6          RECEIVE updated weights w^{t+1} from S
7          AGGREGATE the updated weights w^{t+1} as (1/P) * sum_{s in S} w_s^{t+1}
8          UPDATE w^t with the aggregated weights
9      end parallel
10     run Node in parallel
11         DOWNLOAD the weights w^t
12         UPDATE the parameters for E epochs with a given learning rate
13         SEND w^{t+1} back to the server
14     end parallel
15 end for
```

Listing 4.2: The FedAvg algorithm designed by McMahan et al. [50].

Concretely, the training process in FedGP consists of communication rounds where selected nodes update their respective models in each round. At the beginning of the round, the server supplies the updated generator to nodes. Discriminators remain private and are not shared between the nodes or the server.

During the evaluation of FedGP, the authors studied the accuracy of the proposed method in two settings: with independent and identically distributed (i.i.d.) data and with non-i.i.d. data. They noticed that non-i.d.d. was beneficial for FedGP. Likely, because the discriminators are more easily trained with less diverse data. Jeong et al. [36] inspected the behaviour of non-i.i.d. data and they developed a federated augmentation method that empowered nodes to reproduce all samples across the infrastructure locally. As a result, the final synthetic dataset should be i.i.d. This might be convenient in some scenarios.

The privacy guarantee of FedGP is not as strong as (ϵ, δ)-DP. In FedGP, there is used differential average-case privacy. The gained protection proved to be sufficient against passive adversaries, but the differential average-case privacy is still not accepted by the privacy community [5]. In comparison, Triastcyn and Faltings utilized the notion of Bayesian DP (BDP) in their subsequent work [72]. The method provides user-level privacy and datasets

[1] FedSGD corresponds to the SGD algorithm applied in the federated setting. On the server, gradients are averaged proportionally to the number of training samples on nodes and used to make one gradient descent step.

of individuals are protected from the central server and other participating users. BDP allows non-uniform failure probability for all data points, while the standard DP has just uniform probability. Thanks to that, DP guarantees hold for out-of-distribution samples (e.g., for data points that are generally hard to hide). BDP enabled the GANs to generate data of higher fidelity. The experiments showed that the generated data could be used for both labelling and training. The models which were trained on the synthetic data achieved remarkable accuracy compared to prior state-of-the-art methods. Also, switching from real to synthetic data did not deteriorate the accuracy of the models model significantly.

4.2.3 Problems in Collaborative Learning

Despite the recent progress, collaborative learning still faces many challenges. The learning environment is not always as hospitable as anticipated. The following problems need to be addressed by researchers [43]:

- **Expensive communication**. A serious bottleneck in federated learning is communication. In general, it is required to implement communication-efficient methods that iteratively send small amounts of data through the network. For example, the federated augmentation method proposed in the paper [36] also aims for smaller communication overhead. However, implementing such a method seems unrealistic because sending local data to the server little by little violates the key privacy assumption of federated learning.

- **Systems heterogeneity**. Every participant within a federated network may have different computational and communication capabilities. Here, network connectivity, variability in hardware, and system-related constraints can negatively impact the learning process. Therefore, the developers of federated networks need to take care of heterogeneous hardware and design robust systems that are fault tolerant (e.g., in cases where a few participants drop out during training iteration).

- **Statistical heterogeneity**. Data points across devices may vary significantly. In a distributed setting, the basic assumption of i.i.d. data is therefore often violated. The artificial augmentation of local datasets alleviates this problem. Having datasets of faces, it is possible to utilize the GAN frameworks, such as StarGANv2, to augment the datasets.

- **Privacy concerns**. Finally, the idea behind collaborative learning is to protect the sensitive data of individuals. Current methods aim to enhance privacy by employing secure multiparty computation (SMC),[2] HE, or DP. This all comes at the cost of system efficiency or model performance.

4.2.4 Attacks and Defensive Measures

In centralized learning, only the server is treated as an unsafe spot. Standard configurations in collaborative learning pose threats from both a dishonest server and honest-but-curious participants. In this section, we provide an overview of attacks and defensive measures in collaborative learning. The presented information is taken from the paper [20].

[2] In SMC, multiple parties cooperatively compute a function over their inputs while keeping those inputs private.

An adversary has two main goals when attacking a model: to extract private data from victims and to force the model to behave differently than intended. The abilities to regulate participating clients, to differentiate the model updates before the aggregation, to shape the data on which the clients train their models, to manipulate gradient updates, or to influence the impact of clients' respective models during the aggregation are all considered to be exploitable points in federated learning. Attacks performed by the adversary can be partitioned into specific types with respect to these abilities.

Sample Reconstruction Attacks

The usage of ReLU activation functions on each layer enables an adversary to compute what relation has the input to a given loss at the output. This is done via a mathematical framework that assumes complete knowledge of the system. The input-output relation is described by several polynomials, which are used to determine training samples consistently with the loss function. However, the application of this white-box attack is limited to linear models which incorporate only the ReLU activation functions.

Another attack aims for gradients calculated within the first dense layer that may reveal more information about the training data. The gradients themselves are accumulated in a neuron and linearly scaled to the activation functions. When a model is trained on a single sample, the sample is going to be fully reconstructed by applying this attack.

Deep Leakage from Gradients (DLG) is a type of attack where an adversary tries to retrieve training samples by iteratively optimizing the inputs that produce correct gradients in a client's model.

Information Inference Attacks

The model inversion attack (MIA) is built on the assumption that a trained linear model is available in a black-box setting. Here, an adversary guesses the input training data based on the output of the model.

One can also perform model inference attacks by utilizing GANs. The multitask GAN for Auxiliary Identification (mGAN-AI) attack aims to reconstruct training samples from clients' model updates and an auxiliary dataset. The generator produces fake input images from the model updates, and the discriminator strives to figure out to which client or category the fake images belong. This attack is considered passive, but an active variant allows an adversary to target a specific victim.

Similarly, the standard GAN attack actively tries to persuade a victim to release private data by poisoning the shared model. In this case, an adversary uses a GAN to generate images for labels used by the victim. The adversary mislabels the images to make them look like they do not belong to the victim. This results in steeper gradients submitted by the victim, allowing the adversary to be identified as the victim with a higher probability.

Model Corruption Attacks

Model poisoning can be used to insert a backdoor into a model. For instance, one way of poisoning the model is to declare that a client has a significantly large amount of training samples. This forces the model to become biased towards a concrete class. Furthermore, other attacks focus on reducing the influence of singular clients.

In a Sybil-based attack, an adversary returns a multitude of poisoned models with pseudonymous identities to the server after each iteration. A more serious problem poses

attacks that are coordinated by a larger group of users. This kind of problem was observed in the Twitter chatbot developed by Microsoft. Even though it was rather an attack coordinated unwittingly, such a problem should be considered as well.

Runtime Misclassification Attacks

A misclassification attack attempts to introduce samples that are purposefully misclassified by the shared model. Federated learning is vulnerable to this type of attack because adversaries can craft fake examples without any restrictions once the model is deployed.

Defensive Measures

The attacks above cannot be performed in all situations regardless of the system configuration. For example, the attack on models which use ReLU activation functions is weak against any form of added noise. Moreover, as already stated, this type of attack is limited to linear models only, like the MIA attack.

In convolutional networks, the attack targeting gradients' values within the first dense layer requires extra algorithms to be useful. Furthermore, the method is very unreliable when local datasets are large enough. The greater size of the datasets results in high computation cost in both the DLG and MIA attacks. In addition to that, the standard GAN attack is infeasible when clients are selected randomly during one training step. Also, once clients do not share labels within a federated system, the GAN attack fails too.

Nevertheless, the following defensive measures should be applied in collaborative learning schemes:

- **Gradient selection**. A selection method for dispatching only the most important gradients. Such a method reduces the information that a server has about clients.
- **Gradient compression**. The compression has two advantages. First, it increases the communication efficiency between clients and a server. Second, encoding and decoding the gradients results in losing some information.
- **Dropout**. Dropout introduces randomness to the gradient updates. In general, this technique is used primarily for regularizing deep neural networks to prevent overfitting.
- **DP**. The sensitivity of DP should be set on a client basis. When gradients are clipped and perturbed, a client is not able to infer other clients. Here, it is important to say that Bayesian DP provides faster convergence while retaining the same privacy bounds as the standard DP in federated learning.
- **SMC**. This procedure allows clients to hide their computations from the server. Many developed schemes assume that honest participants are in the majority. Thus, SMC can be deployed without hesitation in such settings.
- **HE**. HE ensures no performance loss in terms of model convergence since the training data remains untouched and the computations are run on their encrypted versions. One of the problems is that the server does not have direct access to the central model. This makes the usage of HE questionable because a single model is usually the result of collaborative learning.

- **Robust aggregation**. Robust aggregation methods restrict malicious clients from exploiting which weights correspond to which clients based on local datasets' size. Also, a key challenge here is to force aggregation before further manipulation occurs because the server may be able to learn the users model updates.

Table 4.1: A table describing the effectiveness of specific defensive measures against the introduced attacks. The symbol ● stands for reliably effective, ◐ for limited effectiveness, ○ for not effective, and ⊙ for context dependent effectiveness, respectively [20].

Defenses \ Attacks	Loss-function / ReLU	First dense layer	DLG	MIA	mGAN-AI	GAN	Example misclassification	Model poisoning
Gradient selection	●	◐	◐	●	◐	◐	○	○
Gradient compression	◐	○	○	◐	◐	◐	◐	○
Dropout	●	○	○	●	◐	●	●	◐
DP	●	⊙	●	●	⊙	●	●	○
SMC	○	◐	◐	◐	●	○	○	○
HE	●	●	●	●	●	○	○	○
Robust aggregation	○	○	○	◐	○	◐	○	●

4.3 Privacy Overview

In centralized learning, all data are stored in one place and are managed by a single maintainer. Therefore, only the maintainer defines how the data are protected and how they will be used. Besides the problems we outlined at the beginning of Section 4.1, accidental or intentional breach may expose biometric features of enrolled individuals when the data stays unencrypted in the database.[3] Furthermore, biometric data cannot be centrally processed unless there is given explicit consent or the processing is essential for reasons of substantial public interest.[4]

On the other hand, collaborative learning suggests resolving these issues. However, it requires shifting the workload to clients who possess the data. Adversaries can profit from that because they have more insights into the learning process. Due to this, collaborative

[3] https://us.norton.com/internetsecurity-emerging-threats-biometric-data-breach-database-exposes-fingerprints-and-facial-recognition-data.html

[4] https://europa.eu/youreurope/business/dealing-with-customers/data-protection/data-protection-gdpr/index_en.htm

learning poses privacy risks from dishonest clients as well. As we mentioned in Section 4.2.4, many countermeasures need to be implemented in order to minimize potential attacks. Sill, since sensitive data are not stored on a single node, the adversaries cannot deduce all biometric features with a successfully performed attack at once.

Chapter 5

Proposed Method

Centralized learning provides fewer opportunities for adversaries in comparison to collaborative learning. Yet, at the cost of privacy. To tackle the privacy problem, we propose replacing sensitive data from a private domain with data from a safe domain, totally unrelated to the private domain. The translation to the safe domain can be performed by a GAN. Then, the centralized classification model residing on the server is trained purely on the outputs of the GAN. Thus, the server is unaware of sensitive data points from the original distribution.

At first sight, our approach perhaps resembles the concept of GANobfuscator. However, we are not focusing on anonymizing existing input datasets. We rather try to examine the possibility of replacing the whole input domain and still assure reasonable utility.

This means that we can use the translated images for authentication purposes. For instance, we assume scenarios in which users authenticate themselves with images of flowers or shoes without revealing their actual identities. After successful authentication, the system which benefits from this method grants access to permitted resources, like regular biometric-based authentication systems. As opposed to the standard biometric-based authentication systems, here, the classifiers are learnt how to identify users concerning the features posed by images of flowers or shoes instead of real faces, as shown in Figure 5.1.

Note that through this whole work, we focus only on translating images of human faces. Hence, no other biometrics will be used for final analysis or evaluation.

5.1 Analysis

Our method is related to the works made by Chen et al. [14], Sirichotedumrong and Kiya [67], and Ito et al. [34]. Chen et al. developed a privacy protection model where a GAN generates fake faces and an additional transformation method adjusts the final image to maintain the key attributes of the face. Similarly, Sirichotedumrong and Kiya developed a scheme that protects visual information on plain images by translating the input images into a visually protected (noisy) domain. Ito et al. resolved some of the issues from the previous work and focused on reducing the loss value of a classification model deployed on the server. Their scheme exposes no visual information, as opposed to the work made by Sirichotedumrong and Kiya.

However, none of these approaches considers scenarios where both the server and clients are mutually distrustful and the communication between them is not curated by a trusted third party.

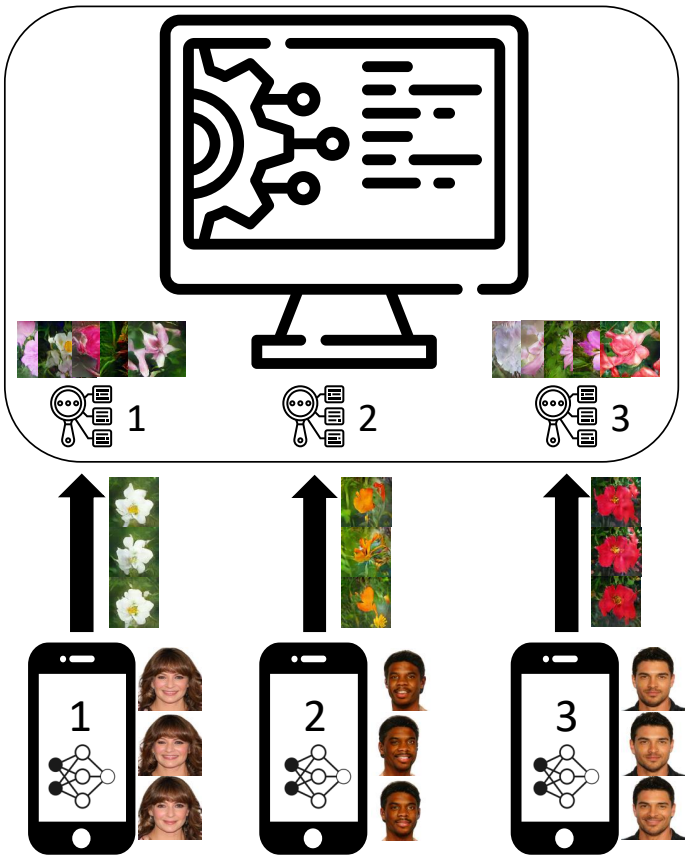

Figure 5.1: A GAN-based authentication system. Face images of users are translated to images of flowers which are then used for authentication purposes.

A possible solution to the security problem of biometric systems lies in cancelable biometrics. The concept of cancelable biometric identifiers was addressed by Bathen et al. [7]. The whole process starts by capturing the face of a user, extracting its features, and applying a set of filters. Every feature set is decomposed into blocks where each block is filtered by bloom filters. The bloom filters are useful for determining whether an element is a member of a set or not. After applying the bloom filters, the resulting data are salted before being stored. From the user's perspective, it is possible to create temporary IDs which can be changed on demand. Any system trying to use the cancelable IDs then requests from a user two things: (1) proof that the user owns the salt, (2) an ID matching the authenticated person. The only drawback of the system is that every time a new biometric is added, it is required to retrain models which identify the users from the blooms.

Our method does not require any extra transformation functions or salting and is not restricted to a concrete GAN architecture or dataset. On the other hand, some limitations are still present. First, a GAN must be capable of translating images between heterogeneous and asymmetric domains. Second, the translation mechanism must not preserve any visible features of the input images. Third, clients are required to train their models on their own because no transfer learning is employed in the process.

If one of the limitations is neglected, the system may fail. For instance, when a GAN generates images of poor quality, it means that a classifier on the server will not be able to properly learn any features that distinguish one user from the others. Also, once the

translated images preserve notable features of a human face, privacy cannot be guaranteed because it helps an attacker to identify a victim.

The privacy protection is therefore solely provided by a GAN which resides on the user's device and is trained on arbitrary datasets. The GAN model is expected to be trained on a private dataset (containing face images of the user) backed by a public dataset. When an adversary gains access to the user's device, privacy may be violated. In ideal conditions, the adversary can reconstruct the reverse mapping function by inserting a new inverse GAN into the original framework, freezing the layers of the existing GAN, and initiating the training. In the end, the adversary may have a generator capable of generating the actual faces of the users from a visually private domain. We provide more details on possible attacks on our method in Chapter 7.

5.2 Implementation Details

All the source files are written in Python. Python itself does not allow running more than one concurrent thread at once.[1] Therefore, we used TensorFlow [1], a deep learning framework developed by Google, to ensure that computationally expensive operations take advantage of multithreading during the training of GANs.

In TensorFlow, computation graphs are compiled in advance and evaluated in a custom interpreter. To achieve high performance, TensorFlow includes support for CUDA-enabled[2] GPU cards. Furthermore, TensorFlow provides a simple API for automatic differentiation. Automatic differentiation enables the framework to evaluate the derivatives of functions which can be computed automatically by repeatedly applying the chain rule on selected arithmetic operations. From the programmer's perspective, TensorFlow remembers what operations happen in what order during the forward pass and then traverses the list of the operations in reverse order to compute gradients in the backwards pass. Listing 5.1 demonstrates the way how gradients can be computed in TensorFlow.

```
with tf.GradientTape(persistent=True) as tape:
    fake_B = self.gen(real_A, training=True)
    disc_real_outputs = self.disc(real_B, training=True)
    disc_fake_outputs = self.disc(fake_B, training=True)
    disc_loss = get_disc_loss(disc_real_outputs, disc_fake_outputs)
    gen_loss = get_gen_loss(disc_fake_outputs)

disc_vars = self.disc.trainable_variables
gen_vars = self.gen.trainable_variables
disc_gradients = tape.gradient(disc_loss, disc_vars)
gen_gradients = tape.gradient(gen_loss, gen_vars)

self.disc_opt.apply_gradients(zip(disc_gradients, disc_vars))
self.gen_opt.apply_gradients(zip(gen_gradients, gen_vars))
```

Listing 5.1: A simple GAN training step implemented in TensorFlow.

[1] This is relevant only to CPython, the default and most widely used implementation of the Python language.

[2] `https://developer.nvidia.com/cuda-zone`

For our method, we selected only four out of all GAN architectures introduced earlier: CycleGAN, DiscoGAN, TraVeLGAN, and U-GAT-IT. The source codes for CycleGAN[3] and U-GAT-IT[4] are publicly available and required just minor modifications to the existing input pipelines. Other frameworks are implemented based on the recommendations from the original papers. Overall, the input pipeline is the same across the frameworks. As shown in Listing 5.2, we execute in parallel as many operations as possible to achieve peak performance for delivering input data. Also, we cache and load the data in advance before each subsequent iteration to improve the performance even more. The caching is beneficial only when there is a requirement to have the very same data during the whole training. In order to augment data, it is possible to skip the caching.

```
def read_image(image):
    image = tf.io.decode_png(tf.io.read_file(image), channels=3)
    return tf.data.Dataset.from_tensors(image)

def normalize_image(image):
    return tf.image.resize(image, [128, 128]) / 127.5 - 1

def build_input_pipeline(domain_files, batch_size):
    d = tf.data.Dataset.list_files(domain_files)
    d = d.interleave(read_image, num_parallel_calls=tf.data.AUTOTUNE)
    d = d.map(normalize_image, num_parallel_calls=tf.data.AUTOTUNE)
    d = d.batch(batch_size, drop_remainder=True)
    d = d.cache().prefetch(tf.data.AUTOTUNE)
    return d

train_A = build_input_pipeline(domain_A_files, batch_size)
train_B = build_input_pipeline(domain_B_files, batch_size)
dataset = tf.data.Dataset.zip((train_A, train_B))
```

Listing 5.2: Initialization of the input pipelines used in the implemented frameworks.

Employing DP in the training procedure was possible thanks to TensorFlow Privacy. The library provides classes for differentially private optimizers that are based on the standard TensorFlow optimizers. For instance, `DPKerasAdamOptimizer` is a differentially private variant of Adam optimizer that overrides the class `tf.keras.optimizers.Adam` (Listing 5.3). In addition to that, the library implements methods for computing the privacy budget spent to train a machine learning model. To derive achieved privacy guarantees, an RDP accountant is utilized.

```
dp_optimizer = DPKerasAdamOptimizer(
    l2_norm_clip=1.0,
    noise_multiplier=0.1,
    num_microbatches=16,
    learning_rate=5e-5, beta_1=0.5, beta_2=0.9
)
```

Listing 5.3: A definition of differentially private Adam optimizer.

[3] https://github.com/keras-team/keras-io/blob/master/examples/generative/cyclegan.py
[4] https://github.com/taki0112/UGATIT

To validate the proposed method, it was necessary to implement classifiers used for the authentication of users in a simulated environment. The classifiers are also implemented in TensorFlow. But, there is no need to explicitly compute and record gradients due to the presence of built-in methods for standard training procedures. The initialization and training can be shortened to a few lines of code (see Listing 5.4).

```
model = keras.Sequential([
    keras.layers.InputLayer(input_shape=IMAGE_SHAPE),
    hub.KerasLayer(model_handle, trainable=True),
    keras.layers.Dense(1, activation="sigmoid"),
])
model.build((None,)+IMAGE_SHAPE)
model.compile(
    optimizer=keras.optimizers.Adam(lr=1e-3),
    loss=keras.losses.BinaryCrossentropy(), metrics=['accuracy'],
)

train_dataset = build_input_pipeline(files, batch_size)
model.fit(train_dataset, epochs=EPOCHS)
```

Listing 5.4: Initialization and training of a simple classifier that benefits from the pre-trained network declared by `hub.KerasLayer`.

Chapter 6

Evaluation

To validate the proposed method, we carried out multiple experiments. First, we trained four different GANs and compared the resulting quality of generated images. The GANs were trained on five different datasets in order to give us a better overview of the utility of the frameworks. Then, we analyzed and evaluated the performance of binary classifiers which were trained on the images produced by the GANs.

6.1 Datasets

Assuming that the input data for GANs are always images of faces, one of the best available datasets is the CelebA dataset [47]. The dataset contains more than 200,000 images of celebrities and includes 10,177 distinct identities. The images cover large pose variations, different skin tones, and face shapes. Such diversity is a prerequisite for good generalization properties.

Choosing a satisfying output domain was more difficult. We have determined that many datasets are very heterogeneous and asymmetric to the domain of faces. Ultimately, the following datasets were considered for further evaluation:

- Shoes [80]. A dataset of 50,025 images collected from the catalogue of shoes from `Zappos.com`. One of the disadvantages is that the images of shoes are captured just from one angle. Therefore, further augmentations need to be employed to improve the dataset's utility.
- Textures [17]. A collection of 5,640 textural images organized into a list of 47 categories.
- Cars [41]. A dataset containing more than 16,000 images of cars. The images are captured from different angles and background conditions vary across the whole dataset.
- Flowers [55]. A collection of 8,186 flowers separated into 102 categories. The images have large colour and shape variations.
- Food [9]. A dataset consisting of 101,000 food images with substantial background differences. The images are divided into 101 food categories.

6.2 Architecture Comparison

After thoughtful consideration, we decided to implement and evaluate four GAN architectures. Out of all the architectures, TraVeLGAN worked best when trained on most of the datasets described earlier.

CycleGAN

CycleGAN performed worst in the experiments. When trained on the datasets of shoes and textures, the loss values of the discriminators rapidly converged to zero causing immediate mode collapse.

We tried to introduce the concept of image pooling, described in the CycleGAN paper, to see if the performance will improve. According to the paper, image pooling helps the discriminator to remember which images were incorrectly classified in the past. Basically, the images generated by the generator are stored in a pool (buffer) and then randomly sampled by the discriminator through the training. Still, this technique did not provide any satisfying results.

Proper hyper-parameters tuning, like setting different weights for the loss functions, could possibly resolve the problem. Yet, we found the tuning time consuming and disregarded CycleGAN from additional experiments.

DiscoGAN

The DiscoGAN framework worked better on all datasets. However, the results were still not plausible. Many times, the generators collapsed or were not able to produce sharp and crisp images.

The situation did not improve even after adding feature matching loss to the training process [63]. The inability to generate satisfying images led us to experiment with the loss established in the WGAN-GP paper. We also tried to use spectral normalization [52]. Spectral normalization stabilizes the training of GANs and avoids mode collapse by controlling the Lipschitz constant of the discriminator. While WGAN and WGAN-GP enforce the Lipschitz constraint by clipping the weights and penalizing the norm of gradients, respectively, spectral normalization constrains the spectral norm. The spectral norm is the maximum singular value of a matrix. Unfortunately, neither WGAN-GP loss nor spectral normalization did improve the final results.

On the other hand, DiscoGAN performed best only when the model was trained on the dataset of shoes. A few examples of the generated images are provided in Appendix B.1.

TraVeLGAN

TraVeLGAN has shown the most decent results out of all tested frameworks. Except for the textures and cars datasets, the model performed well on the rest of the datasets. The models which were trained on the datasets of textures and cars were not producing reasonable outputs and the generative model was jumping between modes for the same inputs through the training.

During the experiments with DiscoGAN, we noticed that changing the background of images results in very poor translations. This motivated us to examine images without

cluttered background too. Therefore, we created a new dataset of faces with removed background from the original CelebA images by utilizing image segmentation.[1]

In Figure 6.1, there are demonstrated outputs of TraVeLGAN. It is important to emphasise that the translations to flowers were only slightly affected by zooming or additional background changes. More generated samples are provided in Appendix B.2.

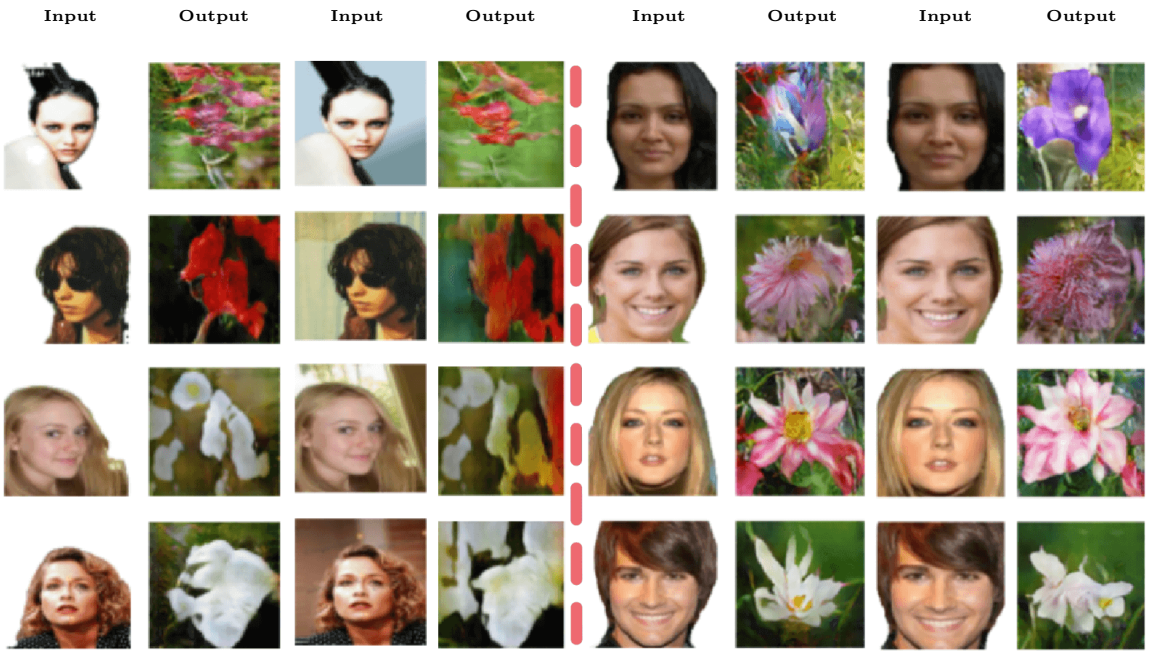

Figure 6.1: Left: outputs of TraVeLGAN trained on the dataset of flowers and CelebA images with removed background. The GAN was trained for 111 epochs with Adam optimizer. Right: outputs of TraVeLGAN trained on the dataset of flowers and cropped CelebA images with removed background. In this case, the GAN was trained for 250 epochs with Adam optimizer (FID score [28]: 51.82).

Experiments, where two or more distinct target datasets were merged, did not result in any increased utility for DiscoGAN or TraVeLGAN. The training was not prosperous and the efficiency of mapping functions was not developing. The models were consistently jumping between modes, i.e., a few data points from the target datasets.

U-GAT-IT

We tried to evaluate the performance of U-GAT-IT as well. In this case, the resulting synthetic images were fairly plausible for the dataset of shoes (Appendix B.3). However, we were not able to exploit the real performance of U-GAT-IT because the full model requires 32 GB of memory on a single GPU.

On NVIDIA Tesla T4 GPU, training the light model for 50 epochs on a dataset that consisted of 10,000 samples took more than 4 days. To reduce the training time, it is possible to do computations in parallel by splitting the model into two isolated entities. But, this would involve refactoring the source code significantly. Because of that, U-GAT-IT was not included in further experiments.

[1] `https://github.com/susheelsk/image-background-removal`

6.3 System Performance

Based on the outputs seen earlier, we believe that TraVeLGAN trained on the dataset of flowers is a sufficient method for translating images to a secure domain. The framework was able to correctly identify key features of images regardless of the background changes or magnifications.

To see whether synthetic images are suitable for authentication and identification, it is necessary to verify that the model generates undoubtedly similar images for the same individual. Unfortunately, the face images of a single individual contained within the CelebA dataset often do not resemble each other. Because of that, we had to augment the images with StarGANv2. The required precondition for the classification purposes could be afterwards verified, as shown in Figure 6.2. More images can be found in Appendix B.4.

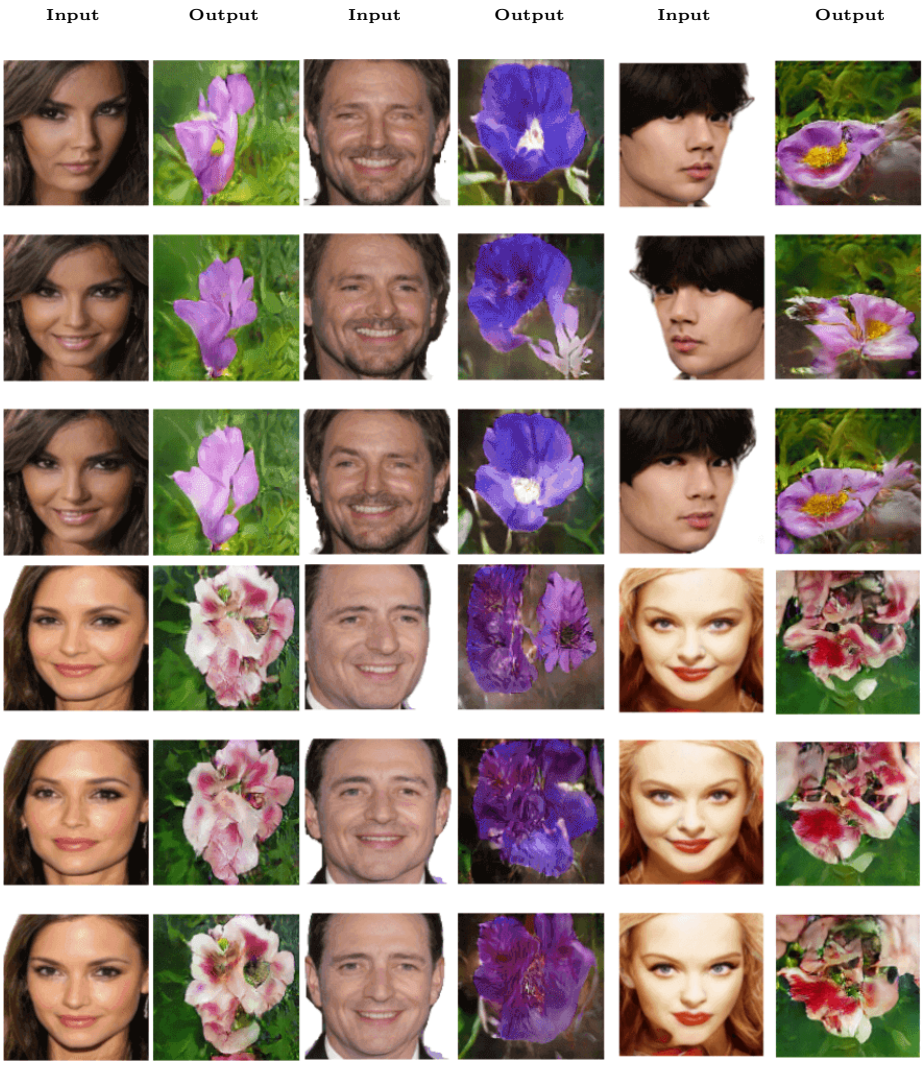

Figure 6.2: Synthetic flowers generated by TraVeLGAN for the same identities. Minor head rotations do not influence the outcome of the translation. The GAN was trained for 250 epochs with Adam optimizer.

Regarding the augmentation, we noticed that different skin tones contribute to a simple colour change of flowers. On the contrary, facial expressions, as well as hairstyles, affect

the translation process significantly, leading to generating flowers of completely different shapes.

Nevertheless, seeing the pleasant outputs produced by TraVeLGAN, the usability of our method could be further assessed on binary classification problems (i.e., authentication). To do so, we selected 93 celebrities from the CelebA dataset and augmented the images of their faces. The number of images ranged from 10 to 20 per selected individual. Then, we translated the augmented images and another random 2,000 images from the CelebA dataset to images of flowers with TraVeLGAN. Finally, we trained 93 binary classifiers on the images of flowers and 93 binary classifiers on the images of faces.

To study the performance drop between the classifiers trained on the images of flowers and classifiers trained on the images of faces, the conditions for both scenarios must be similar. Therefore, we exploited transfer learning and used the same pre-trained MobileNet V2 model in both cases. MobileNet V2 is one of the state-of-the-art computer vision models tailored for mobile devices. The model architecture consists of an initial convolution layer with 32 filters, followed by 19 residual bottleneck layers.[2] Based on the experimental evidence, the authors suggested using linear bottlenecks since employing non-linear layers in bottlenecks destroys too much information about input images [64].

Due to the fact that MobileNet V2 could not be used for our intention as it is, we used only its feature vectors of images[3] and added a final output layer, yielding 1 unit with the Sigmoid activation function to predict the probability of classes directly.

Moreover, it was necessary to change the way how training data are perceived by the classifiers. To make the classifiers pay more attention to examples from under-represented classes,[4] we adopted the concept of class weights. Such an approach allowed the models to assign higher loss values for the samples of actually classified individuals during the training. Note that the behaviour of Adam optimizer, which was used in the classifiers, is always unaffected by the scaling change.

In the end, we performed 5-fold cross-validation and used the following metrics: accuracy, precision, recall, and F_1 score to evaluate the performance of the classifiers. Precision is a proportion of positive samples that were correctly classified to all positive predicted samples. Recall is expressed as a ratio of correctly classified positive samples to the total number of all real positive samples. F_1 score represents a harmonic mean of precision and recall. Let TP, TN, FP, FN be the acronyms for true positives, true negatives, false positives, and false negatives, respectively, the mathematical notations for accuracy, precision, recall, and F_1 score are listed below [70]:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{6.1}$$

$$Precision = \frac{TP}{TP + FP} \tag{6.2}$$

$$Recall = \frac{TP}{TP + FN} \tag{6.3}$$

$$F_1 = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{6.4}$$

[2] A bottleneck layer is a layer with fewer neurons than the adjacent layers.

[3] `https://tfhub.dev/google/imagenet/mobilenet_v2_100_128/feature_vector/4`

[4] Under-represented classes are classes that describe an authenticated individual with respect to a single binary classifier. These classes have usually less training data. So, the classification is performed on imbalanced data

The time required for training one classifier for 15 epochs fluctuated from 1 to 2 minutes. The whole process of 5-fold cross-validation usually took no more than 10 minutes per classifier on NVIDIA Tesla T4 GPU. Tables 6.1 and 6.2 demonstrate scores of the classifiers. The values were averaged over distinct classes (i.e., identities) across the folds while having the acceptance threshold set to 0.7 for each class. In Table 6.1, the recall score has revealed that the classifiers trained on the synthetic flowers did not identify almost 30% out of all positive examples when the layers of MobileNet v2 were frozen. The classifiers trained on the dataset of faces correctly predicted labels for more than 91% of samples. Table 6.2 shows that the classification performance of the models trained on the dataset of flowers improved when fine-tuning was enabled. On the other hand, we have noticed that the accuracy of the classifiers trained on the dataset of faces decreased.

Table 6.1: Actual scores of the evaluated classifiers. The classifiers were trained for 15 epochs while having frozen the layers of MobileNet v2. In addition to that, a Dense layer with 512 units and 20% dropout rate was inserted before the final output layer (learning rate: 0.001, batch size: 32)

	Faces	Flowers
Metric	Average Value	Average Value
Accuracy	0.99976	0.99761
Precision	0.91385	0.83525
Recall	0.91482	0.70856
F1 Score	0.91433	0.76671

Table 6.2: Actual scores of the evaluated classifiers. The classifiers were trained for 15 epochs and their architecture was the same as declared in Listing 5.4 on lines 1-5 (learning rate: 0.0005, batch size: 54).

	Faces	Flowers
Metric	Average Value	Average Value
Accuracy	0.9992	0.9974
Precision	0.8810	0.8594
Recall	0.9117	0.8627
F1 Score	0.8961	0.8611

The experiments ensured us that our method can be still used for authentication purposes since the overall performance drop did not exceed 6%. Hyper-parameters tuning and further architectural changes should decrease the difference even more.

Chapter 7

Adversarial Attacks

In section 5.1, we outlined what ensures the protection of individuals. Thus, it is crucial to not expose the trained GAN and its weights to attackers. The attackers can easily determine which GAN framework was used for the translation and on which public datasets the framework was trained. Given the fact that the system is used for authenticating users based on the images of flowers generated from human faces, there are not so many options for training datasets out there. Identifying the used GAN framework may be harder because new GAN architectures emerge every year. Still, hiding such trivial information must not be the main security aspect. Security through obscurity should not be therefore considered as a reasonable defensive measure.

Suppose an attacker determines a used GAN framework, e.g., TraVeLGAN and public datasets. A trivial attack consisting of switching the input and output domains and straightforwardly learning the reverse mapping function is not effective. As per our observations, the generated faces looked like those depicted in Figure 7.1. Even though the TraVeLGAN model was trained multiple times with a different set of hyper-parameters, it still could not reconstruct a face that resembles a real individual. More images from the learnt mapping are attached in Appendix C.1.

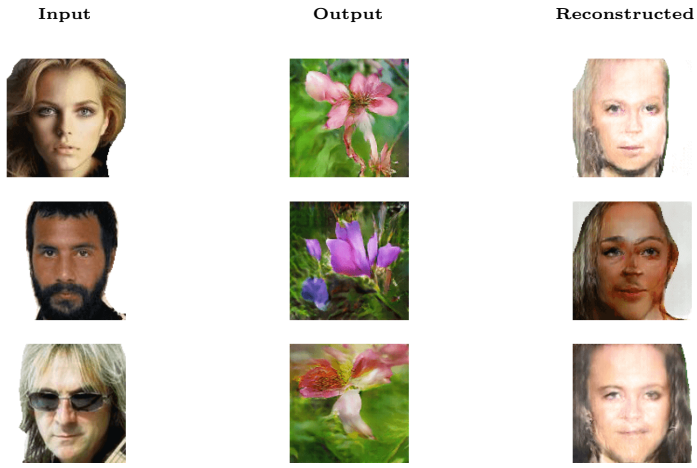

Figure 7.1: Face images reconstructed from the synthetic flowers. Reverse mapping was learnt by TraVelGAN that was trained on the datasets of faces and flowers. Compared to the previous trainings, the input and output domains were switched. The framework was trained for 240 epochs with Adam optimizer.

Similarly, an attacker will not be able to get plausible reconstructed images by exploiting cycle-consistency or reconstruction loss embedded into the framework. Based on the experiments with DiscoGAN, we observed that there is shaped an imbalance between translated and reconstructed images during the training. The generator which had been learning reverse mapping from the translated images back to the images present in the input domain either collapsed or was not able to generate images in reasonable quality (see Figure 7.2).

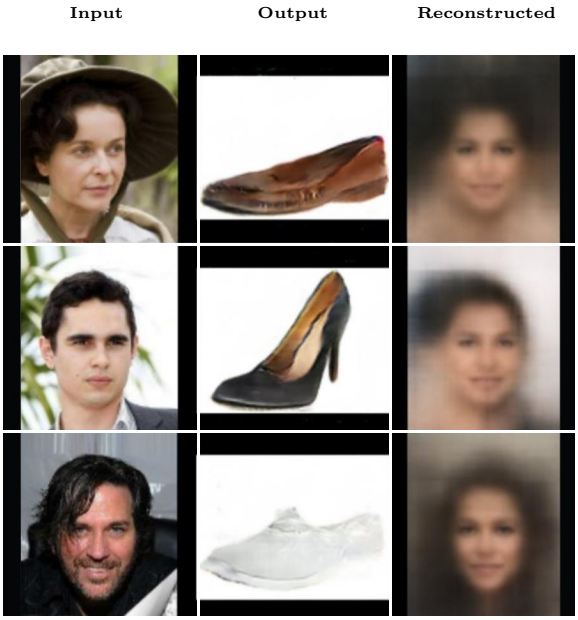

Figure 7.2: Samples produced by DiscoGAN trained on the datasets of faces and shoes. The network was trained for 200 epochs with the same hyper-parameters as presented in the original paper. Additionally, one convolution layer with 100 filters was inserted into the generators.

A more sophisticated attack suggests training a GAN the same way a victim did. The attacker then freezes the trained model and inserts a new neural network model into the framework. The new model is trained to discover reverse mapping from the visually private domain back to the domain of faces more efficiently due to the presence of a correct pair set. This attack is also referred to as an inverse transformation network attack (ITN-Attack) [34].

7.1 Inverse Transformation Network Attack

Ignoring the fact that the user's models are always initialized with random weights and that the user can govern the hyper-parameters, the attacker may create an almost identical mapping function from the domain of faces to a visually private domain. Otherwise, there is no guarantee that an ITN-Attack is going to be successful. Apart from that, the user can utilize DP during the training which results in reduced chances for the attackers as well, yet at the cost of the classification model accuracy. Furthermore, the protection is implicitly given by the properties of the chosen datasets. Transforming images between asymmetric datasets back and forth causes losing the information about initial data distribution, as shown in Figure 7.2.

If an adversary can prepare an exact pair set of input and output images, an ITN-Attack can be performed. Without having a user's model, the preparation can be considered unfeasible thanks to the characteristics of neural network models described earlier. Nevertheless, we simulated a scenario where the adversary has knowledge about everything except for a private dataset: a user's GAN, the public datasets, and the model's weights.

Assuming that an attacker somehow gains access to the user's model and extracts the model's weights, the only thing remaining to do is to train a new model that makes use of the correct pair set. In TensorFlow, this means inserting the pre-trained network into the training loop by changing a few lines of code (Listing 7.1).

```
1  def train_step(self, batch_data):
2      real_faces, real_flowers = batch_data
3
4      fake_flowers = self.pretrained_travelgan.generator(real_faces)
5
6      with tf.GradientTape(persistent=True) as tape:
7          fake_faces = self.generator(fake_flowers, training=True)
8
9          disc_real = self.discriminator(real_faces, training=True)
10         disc_fake = self.discriminator(fake_faces, training=True)
11
12         siam_real = self.siamese(fake_flowers, training=True)
13         siam_fake = self.siamese(fake_faces, training=True)
14         ...
```

Listing 7.1: One training step of the inverse transformation network (ITN), i.e., TraVeLGAN. The highlighted code demonstrate the changes required to be made in the existing training step of the ITN.

But, the change described above does not assure favourable results. Since there is a notion of valid pair sets, it is necessary to change the objective function as well. In the case of an adversary TraVeLGAN model, we introduced mean squared error loss to the training to ensure that the learnt function pays attention to pixel-level details of original images. Apart from that, we investigated the behaviour of the model with mean absolute error loss and without the presence of mean squared error loss. In both cases, the model could not properly learn the reverse mapping function.

Figure 7.3 displays a successful ITN-Attack performed via TraVeLGAN. More reconstructed images can be found in Appendix C.2. Given sufficient knowledge about correct pair sets, an attacker reconstructed the users' faces to some extent. Still, from the samples, it is possible to determine sex or hairstyle, but not the real identity. On the other hand, such reconstruction allowed the attacker to recreate new flowers that look like those generated from the real face domain. This can be treated as a security flaw because the attacker will be able to fool the classifiers.

Due to that, we performed a couple of experiments with the recreated flowers to see what is the real impact of adversarial images. Once again, we trained 93 binary classifiers and recorded the performance on the adversary images. The adversarial images were classified as valid in 45% out of all cases. The real images were correctly labelled in more than 97% of cases. Although the classification drop is significant, an attacker can be authenticated as a real user almost in one out of two trials.

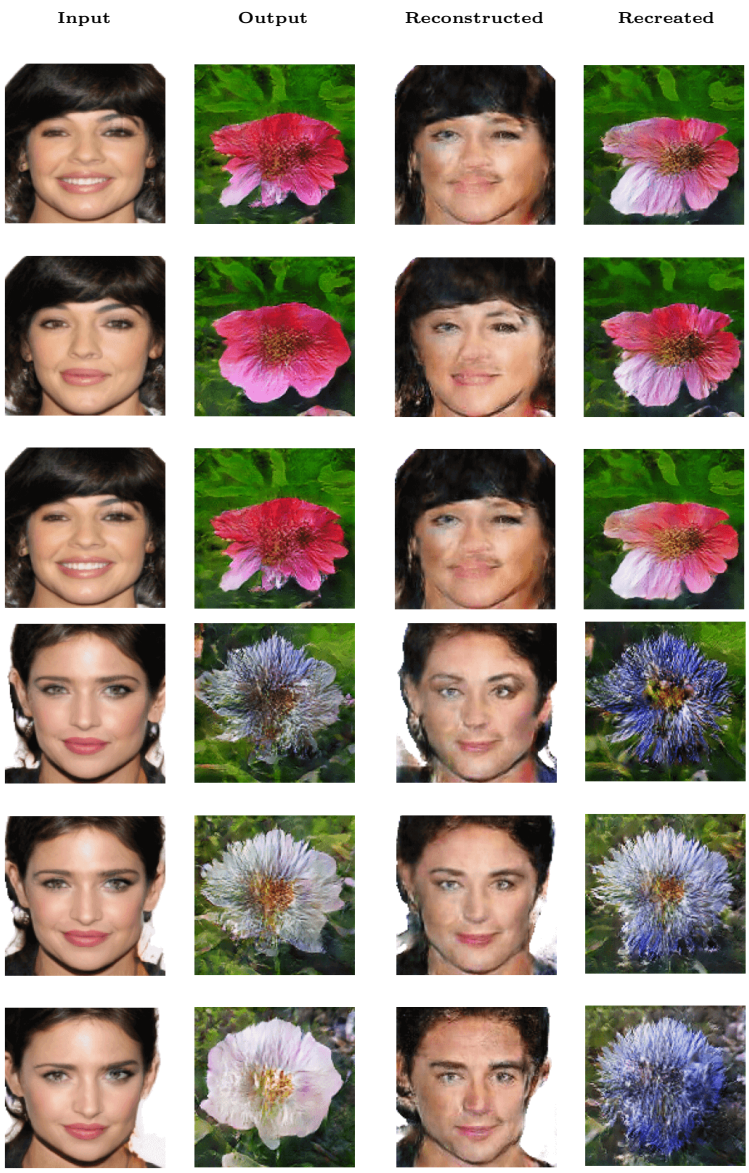

Figure 7.3: Images reconstructed after a successful ITN-Attack. An adversarial TraVeLGAN model is trained on correct pair sets of images of faces and flowers generated by a pre-trained TraVeLGAN network. From the last two columns, it is obvious that the pre-trained TraVeLGAN network pays more attention to hairstyle and face geometry. Because of that, the pre-trained network was able to recreate almost the very same images of flowers from the reconstructed faces.

7.2 Defensive Measures

Applying noise to the output images may seem to be a feasible defensive measure against adversaries. In Figure 7.4, there are displayed reconstructed images after adding Gaussian noise (with the standard deviation equal to 0.1) to normalized images of flowers. Here, the performance of the classifiers decreased subtly in terms of flowers generated by real users (see Table 7.1). However, this technique should not be considered as an acceptable

defensive measure because the adversary can retrain a model to learn which distribution need to be neglected during an ITN-Attack.

Table 7.1: A proportion of positive predicted samples for real and reconstructed flowers.

	Real		Reconstructed	
	Without Noise	With Noise	Without Noise	With Noise
Prediction	0.9718	0.9590	0.4575	0.0517

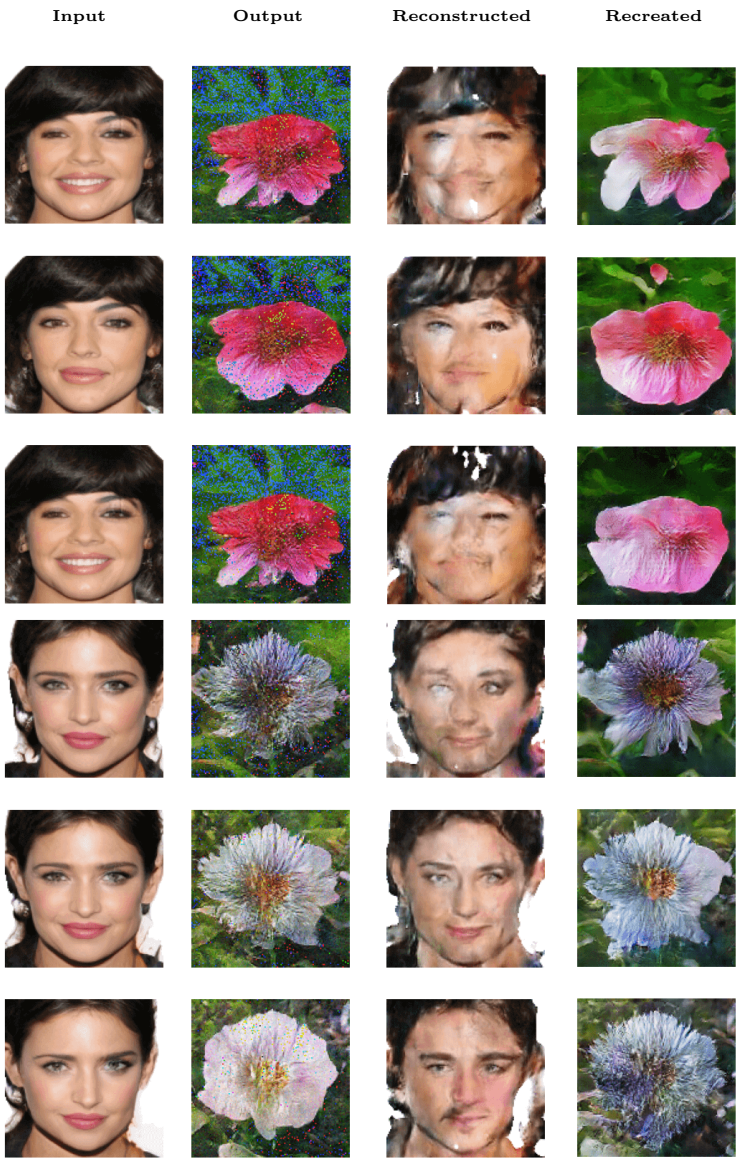

Figure 7.4: Images reconstructed after an ITN-Attack. An adversarial TraVeLGAN model is trained on correct pair sets of images of faces and flowers generated by a pre-trained TraVeLGAN network. In this case, Gaussian noise is added to the output images which results in worse quality of the reconstructed images.

Due to that, we tried to employ DP in the training of a user's TraVeLGAN model by replacing the standard Adam optimizer with its secure variant, declared in Listing 5.3. Unfortunately, the TraVeLGAN model could not learn a favourable mapping function from faces to flowers after introducing noise to the gradients. The following configurations were tested with respect to DP:

- All optimizers within the framework implemented their differentially private variants.
- DP applied only to the discriminator's optimizer.
- DP applied only to the generator's optimizer.

None of the configurations did provide decent results. The generators collapsed after tens of epochs. Because of that, we experimented with so-called warm starting, where a generative model is first trained without employing DP to boost up the convergence rate in the beginning. Then, DP is enabled only for the rest of the epochs. Such a procedure was proposed in the dp-GAN paper [82]. Again, the results were not promising and the warm starting did not alleviate mode collapse, as seen in Figure 7.5. The results did not improve even when we trained the model either for more or fewer epochs.

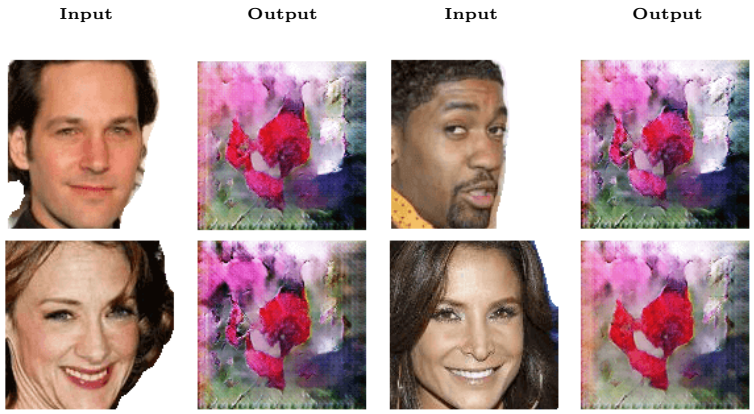

Figure 7.5: Outputs of a TraVeLGAN model trained for 220 epochs without using DP and 30 epochs with using DP.

Except for applying noise to the gradients during the training, we were not able to devise any other defensive measure. Therefore, we acknowledge that if an attacker gains access to the user's device, there is no way to prevent a successful ITN-Attack.

Still, a user is allowed to use different kinds of target domains per each authentication system. For example, it is possible to authenticate with images of flowers in one system and with images of shoes in another system. Compromising one centralized server, therefore, does not affect the other system. Without direct access to correct pair sets of faces and flowers, the attacker should not be able to reconstruct the face identifying a particular user. This assumption is built upon the fact that when we trained the same GAN multiple times, the model did not ever reach the same local optima and always generated different images of flowers for identical input data in comparison to other trained models.

Also, we admit that further transformation functions may need to be employed when protection needs to be provided against unauthorized access to the user's device. With this in mind, it is possible to encrypt the model's weights to make the model residing on the user's device tamper-resistant. The weights will be decrypted only during legitimate usage.

For example, Android implements a system that allows application developers to execute secure encryption and decryption procedures in a tamper-resistant security hardware module.[1] Speaking of Android, TensorFlow offers scripts for exporting the trained models via Tensorflow Lite converter.[2] Thus, a GAN model trained on a personal computer can be exported to the user's device and additional training and transformation will happen in a sandbox only when the model's weights are decrypted. Additionally, model compression may provide greater security guarantees because the weights of GANs will be truncated and rounded. All these improvements are advised for future work.

[1] `https://developer.android.com/training/articles/keystore`
[2] `https://www.tensorflow.org/lite/convert`

Chapter 8

Conclusion

In this thesis, we have presented a novel approach that employs a generative adversarial network (GAN) for privacy preservation in biometric-based authentication systems. In detail, the GAN is used to translate face images of individuals to a visually private domain (e.g., flowers).

The rationale behind the protection of users' privacy lies in the fact that a GAN creates a mapping function that is hard to invert since the target domain is heterogeneous to the domain of faces. Another protection is given by the implicit way of how the GAN can be trained, meaning that different starting point leads to a different optimal translation.

At the beginning of the thesis, we have introduced a couple of GAN architectures. Besides that, we compared differential privacy (DP) and homomorphic encryption (HE) with respect to their complexity, utilization, and specific usage in classification systems. Furthermore, we have assessed the privacy aspects of centralized and collaborative learning.

Based on the findings we conducted, we proposed a method that uses a GAN, which is trained in a centralized setting and still provides reasonable privacy guarantees. We have evaluated four GAN architectures (CycleGAN, DiscoGAN, TraVeLGAN, U-GAT-IT) for the proposed method and compared their performance. Based on the qualitative results of synthetic images, we found out that TraVeLGAN suits our needs best. Binary classifiers trained on the images generated by TraVeLGAN correctly classified 86% of unseen samples. Compared to the classifiers which were trained on the images of users' real faces, the overall performance drop did not exceed 6%. The comparison was made in a configuration where both types of classifiers had the same architecture and conditions. Hence, the conditions were strictly similar in all the experiments.

To validate the aforementioned privacy guarantees, we performed two types of attacks on the proposed method. First, we ran a standard GAN-based attack that is straightforwardly trained to discover a reverse mapping function from the safe domain back to the domain of faces. Second, we executed an inverse transformation network attack (ITN-Attack), where an adversary trains a network on correct pair sets of synthetic and real images. As a result, our method is resistant to the GAN-based attack. With the ITN-Attack, we were able to reconstruct the faces of users to some extent. The reconstructed faces did not resemble the real users, but hairstyle, sex, or race could be clearly determined. Still, this was possible only due to the presence of the correct pair sets what means that the attacker had knowledge about the used GAN architecture, public datasets, and model's weights. So, once the attacker gains access to the user's device and can read the model's weights, the security may be violated.

Employing DP in the training did not mitigate the said security violation. The intuition behind DP is that if a single training point does not affect the outcome of the learning, the information contained in that training point cannot be memorized and the privacy of the individual who contributed this data point to a dataset is respected. According to our experiments, adding noise to the gradients led to mode collapse almost immediately. Therefore, we are doubtful about the real prospects of employing DP in complex GAN architectures. Yet, more experiments might show the opposite result. Nevertheless, it might be appropriate to employ other GANs to perturb input data, for instance, by changing sex or age.

To conclude, our method has just two disadvantages: (1) users are required to train GANs on their own what can be time-consuming and (2) some models may end up generating almost identical images for different identities, which can result in low performance of classifiers. On the other hand, the proposed method is not restricted to a specific GAN framework or dataset. Also, the method does not require training classifiers on the server from scratch since transfer learning can be utilized. The second disadvantage may be alleviated by using another dataset or a GAN when the server detects the performance drop.

In future work, we will investigate the proposed method in a system with 500 or more users. Even though the system worked properly with 93 users in our case, experimenting with a larger user base may exhibit hidden flaws of the proposed method. Similarly, it is necessary to improve the quality of synthetic images generated by GANs. The GANs cannot generate new patterns without further augmentations or without merging distinct datasets. To improve the GAN's performance at low cost, differentiable augmentation may be employed [85]. New GAN frameworks may perform better on the same translation tasks. For instance, U-GAT-IT seemed to be a good candidate for our method too. But, due to the long training times and high computational requirements, we decided to disregard U-GAT-IT for additional experiments.

Moreover, we suggest implementing the system as a whole. In this thesis, we have only shown how GANs can be utilized for generating images that are used for authentication. Pairing users' GANs and server's classifiers need to be studied as well. Such pairing can prevent attackers to execute brute-force attacks and log in to the system as real users. Ultimately, the pairing approach needs to be further assessed because users who trust each other can share the same GAN model.

Also, we recommend developing an update procedure allowing users to submit new synthetic images after changing the visage. This could be done by caching the generated images and when the change within a given threshold is detected, the user's device will message the server to consider the new images.

Preliminary results of this thesis were presented at Excel@FIT.[1] The published paper received an expert committee award. In addition to that, the contribution was also recognized by Honeywell, a partner of the Excel@FIT conference.

[1] `http://excel.fit.vutbr.cz/`

Bibliography

[1] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z. et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems.* 2015. Available at: `https://www.tensorflow.org/`.

[2] Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I. et al. Deep Learning with Differential Privacy. In: ACM. *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security.* 2016, p. 308–318. ISBN 9781450341394.

[3] Amodio, M. and Krishnaswamy, S. Travelgan: Image-to-Image Translation by Transformation Vector Learning. In: IEEE. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.* 2019.

[4] Arjovsky, M., Chintala, S. and Bottou, L. Wasserstein GAN. In: PMLR. *Proceedings of the 34th International Conference on Machine Learning.* 2017, p. 214–223.

[5] Augenstein, S., McMahan, H. B., Ramage, D., Ramaswamy, S., Kairouz, P. et al. Generative Models for Effective ML on Private, Decentralized Datasets. *International Conference on Learning Representations (ICLR).* 2020.

[6] Bagdasaryan, E., Poursaeed, O. and Shmatikov, V. Differential Privacy has Disparate Impact on Model Accuracy. In: NeurIPS. *Advances in Neural Information Processing Systems.* 2019, p. 15479–15488.

[7] Bathen, L., Flores, G. H., Madl, G., Jadav, D., Arvanitis, A. et al. Selfis: Self-Sovereign Biometric IDs. In: IEEE. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops.* 2019.

[8] Benaim, S. and Wolf, L. One-Sided Unsupervised Domain Mapping. In: *Advances in Neural Information Processing Systems.* 2017, p. 752–762.

[9] Bossard, L., Guillaumin, M. and Van Gool, L. Food-101 – Mining Discriminative Components with Random Forests. In: *Computer Vision – ECCV 2014.* Springer International Publishing, 2014, p. 446–461. ISBN 978-3-319-10599-4.

[10] Catak, F. O., Yayilgan, S. Y. and Abomhara, M. A Privacy-Preserving Fully Homomorphic Encryption and Parallel Computation Based Biometric Data Matching. 2020.

[11] Chabanne, H., Wargny, A. de, Milgram, J., Morel, C. and Prouff, E. Privacy-Preserving Classification on Deep Neural Network. *IACR Cryptol. ePrint Arch.* 2017, vol. 2017, p. 35.

[12] Chamikara, M., Bertok, P., Khalil, I., Liu, D. and Camtepe, S. Privacy Preserving Face Recognition Utilizing Differential Privacy. *Computers & Security*. 2020, vol. 97, p. 101951. ISSN 0167-4048.

[13] Chen, X., Xu, C., Yang, X. and Tao, D. Attention-GAN for Object Transfiguration in Wild Images. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, p. 164–180.

[14] Chen, Z., Zhu, T., Wang, C., Ren, W. and Xiong, P. GAN-Based Image Privacy Preservation: Balancing Privacy and Utility. In: *Machine Learning for Cyber Security*. Springer International Publishing, 2020. ISBN 978-3-030-62223-7.

[15] Choi, Y., Choi, M., Kim, M., Ha, J.-W., Kim, S. et al. StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation. In: IEEE. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, p. 8789–8797.

[16] Choi, Y., Uh, Y., Yoo, J. and Ha, J.-W. StarGAN v2: Diverse Image Synthesis for Multiple Domains. In: IEEE. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, p. 8188–8197.

[17] Cimpoi, M., Maji, S., Kokkinos, I., Mohamed, S. and Vedaldi, A. Describing Textures in the Wild. In: IEEE. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014.

[18] Dwork, C. Differential Privacy: A Survey of Results. In: Springer. *International Conference on Theory and Applications of Models of Computation*. 2008, p. 1–19.

[19] Emami, H., Aliabadi, M. M., Dong, M. and Chinnam, R. SPA-GAN: Spatial Attention GAN for Image-to-Image Translation. *IEEE Transactions on Multimedia*. IEEE. 2020.

[20] Enthoven, D. and Al Ars, Z. An Overview of Federated Deep Learning Privacy Attacks and Defensive Strategies. *ArXiv preprint arXiv:2004.04676*. 2020.

[21] Farragher, M. *Create Any Image with C# And a Generative Adversarial Network* [online]. 2019 [cit. 2020-12-12]. Available at: `https://medium.com/machinelearningadvantage/create-any-image-with-c-and-a-generative-adversarial-network-6031a4b90dec`.

[22] Fontaine, C. and Galand, F. A Survey of Homomorphic Encryption for Nonspecialists. *EURASIP Journal on Information Security*. Springer. 2007, vol. 2007, p. 1–10.

[23] Fontanini, T., Iotti, E., Donati, L. and Prati, A. MetalGAN: Multi-Domain Label-Less Image Synthesis using cGANs and Meta-learning. *Neural Networks*. Elsevier. 2020, vol. 131, p. 185–200.

[24] Fu, H., Gong, M., Wang, C., Batmanghelich, K., Zhang, K. et al. Geometry-Consistent Generative Adversarial Networks for One-Sided Unsupervised Domain Mapping. In: IEEE. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, p. 2427–2436.

[25] Gokaslan, A., Ramanujan, V., Ritchie, D., In Kim, K. and Tompkin, J. Improving Shape Deformation in Unsupervised Image-to-Image Translation. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, p. 649–665.

[26] Goodfellow, I., Pouget Abadie, J., Mirza, M., Xu, B., Warde Farley, D. et al. Generative Adversarial Nets. In: NeurIPS. *Advances in Neural Information Processing Systems*. 2014, p. 2672–2680.

[27] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V. and Courville, A. C. Improved Training of Wasserstein GANs. In: NeurIPS. *Advances in Neural Information Processing Systems*. 2017, p. 5767–5777.

[28] Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B. and Hochreiter, S. GANs Trained by a Two Time-Scale Update Rule Converge To a Local Nash Equilibrium. *ArXiv preprint arXiv:1706.08500*. 2017.

[29] Hitaj, B., Ateniese, G. and Perez Cruz, F. Deep Models under the GAN: Information Leakage from Collaborative Deep Learning. In: ACM. *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. 2017, p. 603–618. ISBN 9781450349468.

[30] Hsu, J., Gaboardi, M., Haeberlen, A., Khanna, S., Narayan, A. et al. Differential Privacy: An Economic Method for Choosing Epsilon. In: IEEE. *2014 IEEE 27th Computer Security Foundations Symposium*. 2014, p. 398–410.

[31] Huang, R., Li, Z. and Zhao, J. A Verifiable Fully Homomorphic Encryption Scheme. In: Springer. *International Conference on Security, Privacy and Anonymity in Computation, Communication and Storage*. 2019, p. 412–426.

[32] Huang, X., Liu, M.-Y., Belongie, S. and Kautz, J. Multimodal Unsupervised Image-to-Image Translation. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, p. 172–189.

[33] Isola, P., Zhu, J.-Y., Zhou, T. and Efros, A. A. Image-to-Image Translation with Conditional Adversarial Networks. In: IEEE. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, p. 1125–1134.

[34] Ito, H., Kinoshita, Y. and Kiya, H. Image Transformation Network for Privacy-Preserving Deep Neural Networks and Its Security Evaluation. In: IEEE. *2020 IEEE 9th Global Conference on Consumer Electronics (GCCE)*. 2020.

[35] Izabachène, M., Sirdey, R. and Zuber, M. Practical Fully Homomorphic Encryption for Fully Masked Neural Networks. In: *Cryptology and Network Security*. Springer International Publishing, 2019, p. 24–36. ISBN 978-3-030-31578-8.

[36] Jeong, E., Oh, S., Kim, H., Park, J., Bennis, M. et al. Communication-Efficient On-Device Machine Learning: Federated Distillation and Augmentation under Non-IID Private Data. *ArXiv preprint arXiv:1811.11479*. 2018.

[37] Johnson, J., Alahi, A. and Fei Fei, L. Perceptual Losses for Real-time Style Transfer and Super-Resolution. In: Springer. *European Conference on Computer Vision*. 2016, p. 694–711.

[38] Jordon, J., Yoon, J. and Schaar, M. van der. PATE-GAN: Generating Synthetic Data with Differential Privacy Guarantees. In: *International Conference on Learning Representations (ICLR)*. 2018.

[39] Kim, J., Kim, M., Kang, H. and Lee, K. H. U-GAT-IT: Unsupervised Generative Attentional Networks with Adaptive Layer-Instance Normalization for Image-to-Image Translation. *International Conference on Learning Representations (ICLR)*. 2020.

[40] Kim, T., Cha, M., Kim, H., Lee, J. K. and Kim, J. Learning to Discover Cross-Domain Relations with Generative Adversarial Networks. *International Conference on Machine Learning (ICML)*. 2017.

[41] Krause, J., Stark, M., Deng, J. and Fei Fei, L. 3D Object Representations for Fine-Grained Categorization. In: IEEE. *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*. 2013.

[42] Lee, C.-H., Liu, Z., Wu, L. and Luo, P. MaskGAN: Towards Diverse and Interactive Facial Image Manipulation. In: IEEE. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020.

[43] Li, T., Sahu, A. K., Talwalkar, A. and Smith, V. Federated Learning: Challenges, Methods, and Future Directions. *IEEE Signal Processing Magazine*. IEEE. 2020, vol. 37, no. 3, p. 50–60.

[44] Lim, W. Y. B., Luong, N. C., Hoang, D. T., Jiao, Y., Liang, Y.-C. et al. Federated Learning in Mobile Edge Networks: A Comprehensive Survey. *IEEE Communications Surveys & Tutorials*. IEEE. 2020.

[45] Lira, W. P., Merz, J., Ritchie, D., Cohen-Or, D. and Zhang, H. R. GANHopper: Multi-Hop GAN for Unsupervised Image-to-Image Translation. In: *Computer Vision - ECCV 2020*. Springer, 2020, vol. 12371, p. 363–379. Lecture Notes in Computer Science.

[46] Liu, M.-Y., Breuel, T. and Kautz, J. Unsupervised Image-to-Image Translation Networks. In: NeurIPS. *Advances in Neural Information Processing Systems*. 2017, p. 700–708.

[47] Liu, Z., Luo, P., Wang, X. and Tang, X. Deep Learning Face Attributes in the Wild. In: *Proceedings of International Conference on Computer Vision (ICCV)*. December 2015.

[48] Mao, X., Li, Q., Xie, H., Lau, R. Y., Wang, Z. et al. Least Squares Generative Adversarial Networks. In: IEEE. *Proceedings of the IEEE International Conference on Computer Vision*. 2017, p. 2794–2802.

[49] Mao, Y., Yi, S., Li, Q., Feng, J., Xu, F. et al. A Privacy-Preserving Deep Learning Approach for Face Recognition with Edge Computing. In: *Proc. USENIX Workshop Hot Topics Edge Comput (HotEdge)*. 2018, p. 1–6.

[50] McMahan, B., Moore, E., Ramage, D., Hampson, S. and Arcas, B. A. y. Communication-Efficient Learning of Deep Networks from Decentralized Data. In: PMLR. *Artificial Intelligence and Statistics*. 2017, p. 1273–1282.

[51] Mironov, I. Rényi Differential Privacy. In: IEEE. *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*. 2017, p. 263–275.

[52] Miyato, T., Kataoka, T., Koyama, M. and Yoshida, Y. Spectral Normalization for Generative Adversarial Networks. *ArXiv preprint arXiv:1802.05957*. 2018.

[53] Mo, S., Cho, M. and Shin, J. InstaGAN: Instance-Aware Image-to-Image Translation. *International Conference on Learning Representations (ICLR)*. 2019.

[54] Nam, H. and Kim, H.-E. Batch-Instance Normalization for Adaptively Style-Invariant Neural Networks. In: NeurIPS. *Advances in Neural Information Processing Systems*. 2018, p. 2558–2567.

[55] Nilsback, M.-E. and Zisserman, A. Automated Flower Classification Over a Large Number of Classes. In: IEEE. *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*. 2008.

[56] Ogburn, M., Turner, C. and Dahal, P. Homomorphic Encryption. *Procedia Computer Science*. Elsevier. 2013, vol. 20, p. 502–509.

[57] Omdal, M. Final: CycleGAN and DualGAN on Artistic Image-to-Image Translation. 2019.

[58] Papernot, N., Abadi, M., Erlingsson, U., Goodfellow, I. and Talwar, K. Semi-Supervised Knowledge Transfer for Deep Learning from Private Training Data. *International Conference on Learning Representations (ICLR)*. 2017.

[59] Papernot, N., Song, S., Mironov, I., Raghunathan, A., Talwar, K. et al. Scalable Private Learning with PATE. *International Conference on Learning Representations (ICLR)*. 2018.

[60] Ronneberger, O., Fischer, P. and Brox, T. U-NET: Convolutional Networks for Biomedical Image Segmentation. In: Springer. *International Conference on Medical Image Computing and Computer-assisted Intervention*. 2015, p. 234–241. ISBN 978-3-319-24574-4.

[61] Sadhya, D. and Singh, S. K. Privacy Preservation for Soft Biometrics Based Multimodal Recognition System. *Computers & Security*. 2016, vol. 58, p. 160–179. ISSN 0167-4048.

[62] Salem, M., Taheri, S. and Yuan, J.-S. Utilizing Transfer Learning and Homomorphic Encryption in a Privacy Preserving and Secure Biometric Recognition System. *Computers*. Multidisciplinary Digital Publishing Institute. 2019, vol. 8, no. 1, p. 3.

[63] Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A. et al. Improved Techniques for Training GANs. *ArXiv preprint arXiv:1606.03498*. 2016.

[64] Sandler, M., Howard, A., Zhu, M., Zhmoginov, A. and Chen, L.-C. Mobilenetv2: Inverted Residuals and Linear Bottlenecks. In: IEEE. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, p. 4510–4520.

[65] Shokri, R. and Shmatikov, V. Privacy-Preserving Deep Learning. In: ACM. *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*. 2015, p. 1310–1321. ISBN 9781450338325.

[66] Shokri, R., Stronati, M., Song, C. and Shmatikov, V. Membership Inference Attacks against Machine Learning Models. In: IEEE. *2017 IEEE Symposium on Security and Privacy (SP)*. 2017, p. 3–18.

[67] Sirichotedumrong, W. and Kiya, H. A GAN-Based Image Transformation Scheme for Privacy-Preserving Deep Neural Networks. In: IEEE. *2020 28th European Signal Processing Conference (EUSIPCO)*. 2021, p. 745–749.

[68] Song, X., Chen, Z. and Sun, D. Iris Ciphertext Authentication System Based on Fully Homomorphic Encryption. *Journal of Information Processing Systems*. Korea Information Processing Society. 2020, vol. 16, no. 3, p. 599–611.

[69] Steil, J., Hagestedt, I., Huang, M. X. and Bulling, A. Privacy-Aware Eye Tracking using Differential Privacy. In: ACM. *Proceedings of the 11th ACM Symposium on Eye Tracking Research & Applications*. 2019, p. 1–9. ISBN 9781450367097.

[70] Tharwat, A. Classification Assessment Methods. *Applied Computing and Informatics*. Emerald Publishing Limited. 2020, no. 1.

[71] Triastcyn, A. and Faltings, B. Federated Generative Privacy. *IEEE Intelligent Systems*. IEEE. 2020.

[72] Triastcyn, A. and Faltings, B. Generating Higher-Fidelity Synthetic Datasets with Privacy Guarantees. *ArXiv preprint arXiv:2003.00997*. 2020.

[73] Wang, Z., Song, M., Zhang, Z., Song, Y., Wang, Q. et al. Beyond Inferring Class Representatives: User-Level Privacy Leakage from Federated Learning. In: IEEE. *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*. 2019, p. 2512–2520.

[74] Wang, Z., Bovik, A. C., Sheikh, H. R. and Simoncelli, E. P. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing*. IEEE. 2004, vol. 13, no. 4, p. 600–612.

[75] Wood, A., Najarian, K. and Kahrobaei, D. Homomorphic Encryption for Machine Learning in Medicine and Bioinformatics. *ACM Computing Surveys (CSUR)*. ACM New York, NY, USA. 2020, vol. 53, no. 4, p. 1–35.

[76] Wu, W., Cao, K., Li, C., Qian, C. and Loy, C. C. TransGaGa: Geometry-Aware Unsupervised Image-to-Image Translation. In: IEEE. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, p. 8012–8021.

[77] Xu, C., Ren, J., Zhang, D., Zhang, Y., Qin, Z. et al. GANobfuscator: Mitigating Information Leakage under GAN via Differential Privacy. *IEEE Transactions on Information Forensics and Security*. IEEE. 2019, vol. 14, no. 9, p. 2358–2371.

[78] Yang, Q., Liu, Y., Chen, T. and Tong, Y. Federated Machine Learning: Concept and Applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*. ACM New York, NY, USA. 2019, vol. 10, no. 2, p. 1–19.

[79] Yi, Z., Zhang, H., Tan, P. and Gong, M. DualGAN: Unsupervised Dual Learning for Image-to-Image Translation. In: IEEE. *Proceedings of the IEEE International Conference on Computer Vision*. 2017, p. 2849–2857.

[80] Yu, A. and Grauman, K. Semantic Jitter: Dense Supervision for Visual Comparisons via Synthetic Images. In: IEEE. *Proceedings of the IEEE International Conference on Computer Vision*. 2017.

[81] Yu, F. and Koltun, V. Multi-Scale Context Aggregation by Dilated Convolutions. *International Conference on Learning Representations (ICLR)*. 2016.

[82] Zhang, X., Ji, S. and Wang, T. Differentially Private Releasing via Deep Generative Model (technical report). *ArXiv preprint arXiv:1801.01594*. 2018.

[83] Zhang, X., Ding, J., Errapotu, S. M., Huang, X., Li, P. et al. Differentially Private Functional Mechanism for Generative Adversarial Networks. In: IEEE. *2019 IEEE Global Communications Conference (GLOBECOM)*. 2019, p. 1–6.

[84] Zhao, J., Chen, Y. and Zhang, W. Differential Privacy Preservation in Deep Learning: Challenges, Opportunities and Solutions. *IEEE Access*. IEEE. 2019, vol. 7, p. 48901–48911.

[85] Zhao, S., Liu, Z., Lin, J., Zhu, J.-Y. and Han, S. Differentiable Augmentation for Data-Efficient GAN Training. *ArXiv preprint arXiv:2006.10738*. 2020.

[86] Zhu, J.-Y., Park, T., Isola, P. and Efros, A. A. Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. In: IEEE. *Proceedings of the IEEE International Conference on Computer Vision*. 2017, p. 2223–2232.

[87] Zhu, Y. and Wang, Y.-X. Poission Subsampled Rényi Differential Privacy. In: PMLR. *International Conference on Machine Learning*. 2019, p. 7634–7642.

Appendix A

DVD Content

The attached DVD contains the following directories:

- **sources**. Source files of tested GAN architectures and executable scripts.
- **pretrained**. Pre-trained models' weights that can be used for further assessment.
- **datasets**. Datasets used for authentication purposes.
- **latex**. LaTeX source files.

Appendix B

Generated Images

B.1 DiscoGAN

Figure B.1: Outputs of DiscoGAN trained on the dataset of shoes. The GAN was trained for 200 epochs with Adam optimizer (learning rate: 0.0002, batch size: 200, dataset size: 20,000).

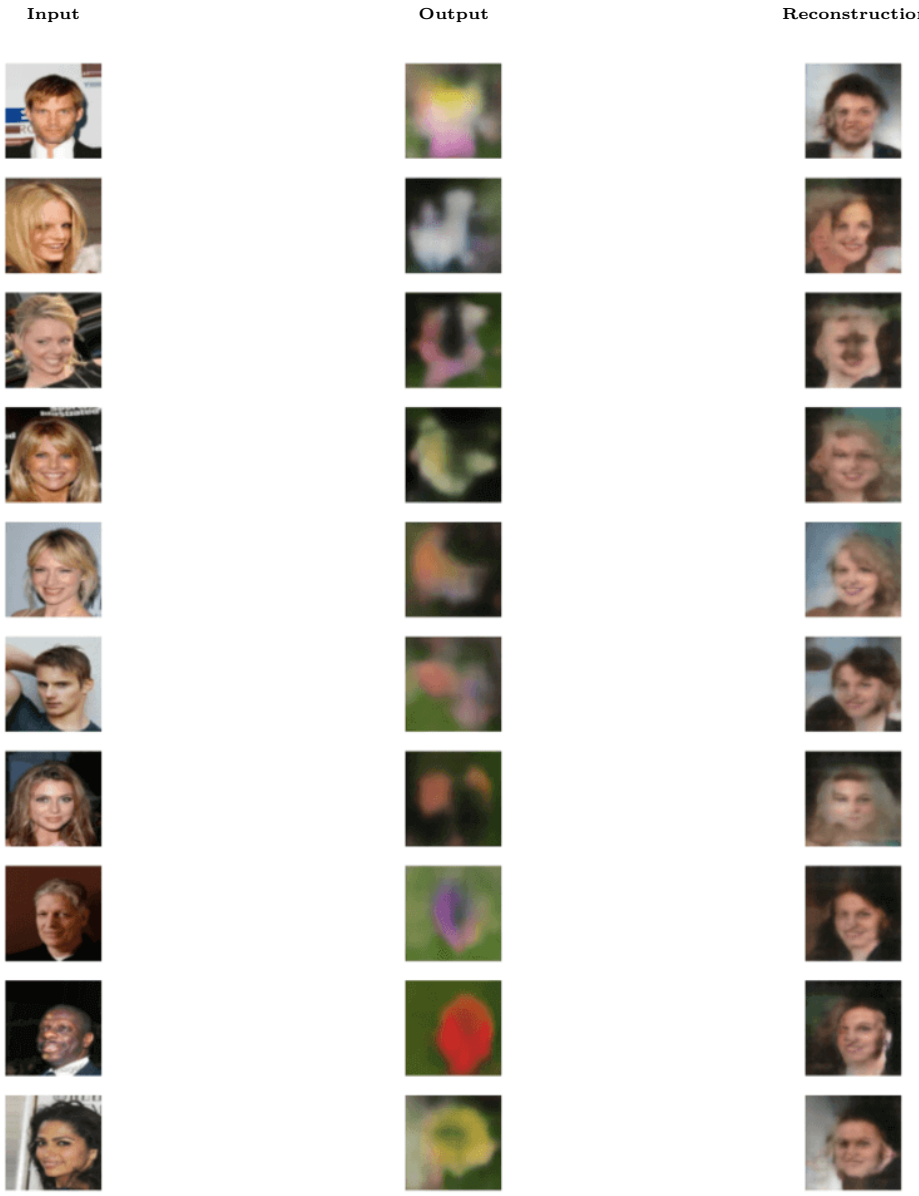

Figure B.2: Outputs of DiscoGAN trained on the dataset of flowers. The GAN was trained for 250 epochs with AdamW optimizer and added feature matching loss (learning rate: 0.0002, weight decay: 0.0001, batch size: 200, dataset size: 8,000).

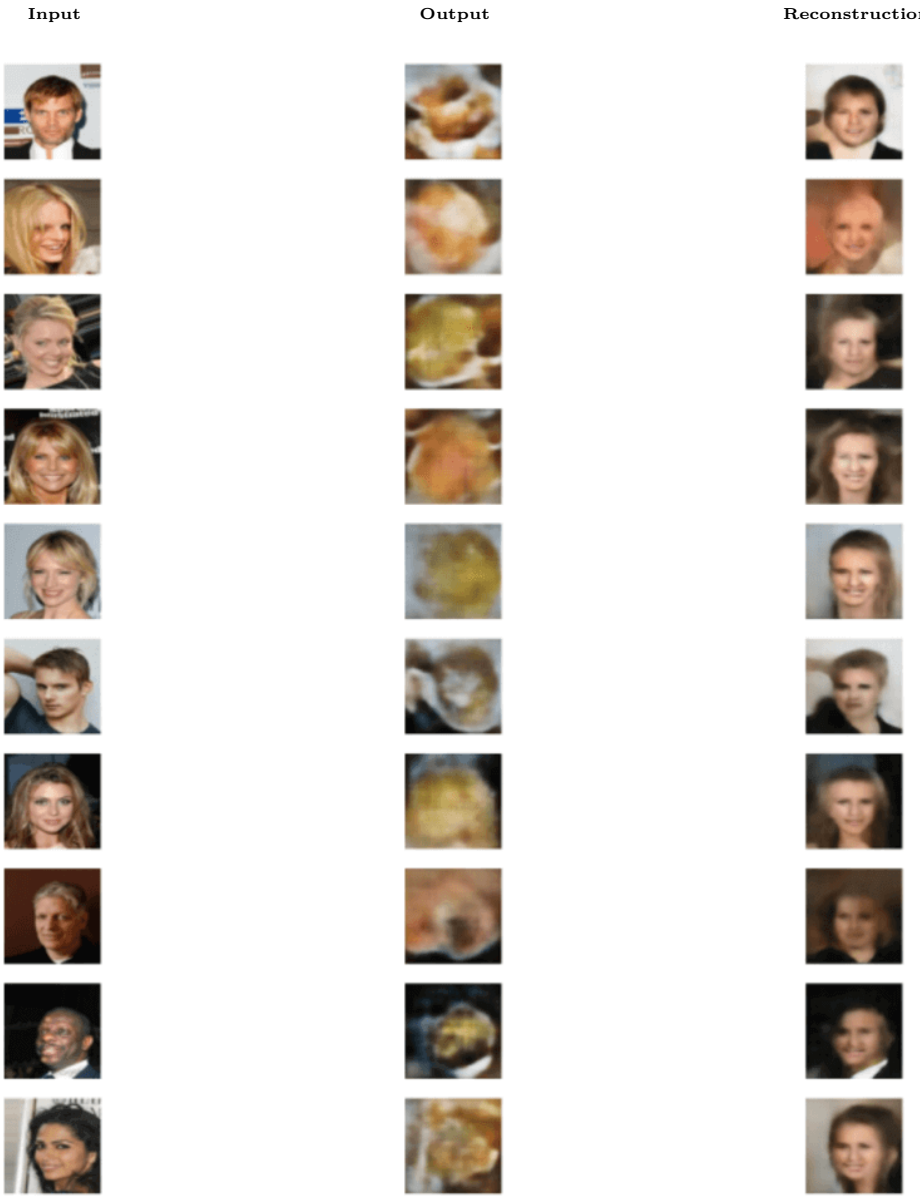

Figure B.3: Outputs of DiscoGAN trained on the food dataset. The GAN was trained for 400 epochs with AdamW optimizer and added feature matching loss (learning rate: 0.0002, weight decay: 0.0001, batch size: 200, dataset size: 5,000).

B.2 TraVeLGAN

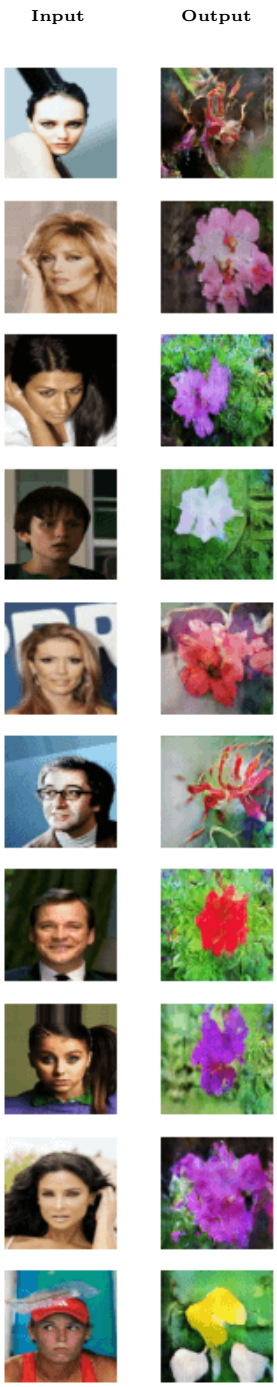

Figure B.4: Outputs of TraVeLGAN trained on the flowers dataset. The GAN was trained for 123 epochs with Adam optimizer (learning rate: 0.0002, batch size: 16, dataset size: 8,000).

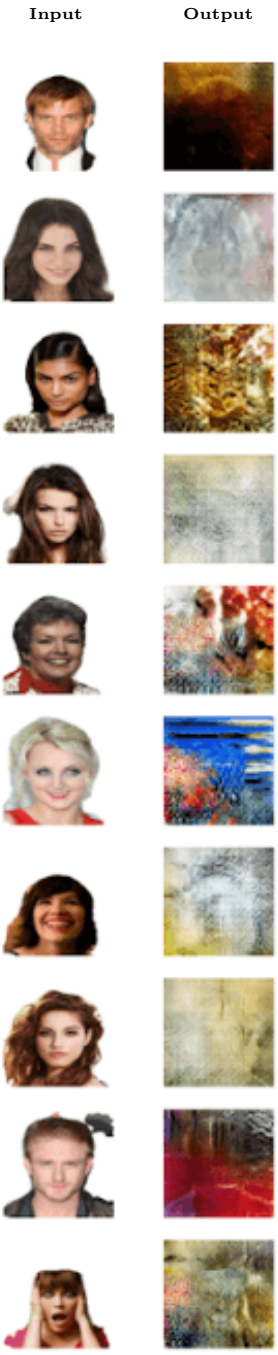

Figure B.5: Outputs of TraVeLGAN trained on the dataset of textures. The GAN was trained for 390 epochs with Adam optimizer (learning rate: 0.0002, batch size: 16, dataset size: 5,640).

Figure B.6: Outputs of TraVeLGAN trained on the dataset of cars. The GAN was trained for 390 epochs with Adam optimizer (learning rate: 0.0001, batch size: 16, dataset size: 8,100).

B.3 U-GAT-IT

Figure B.7: Outputs of U-GAT-IT trained on the shoes dataset. The GAN was trained for 50 epochs with Adam optimizer (learning rate: 0.0001, batch size: 1, dataset size: 10,000).

B.4 Classified Identities

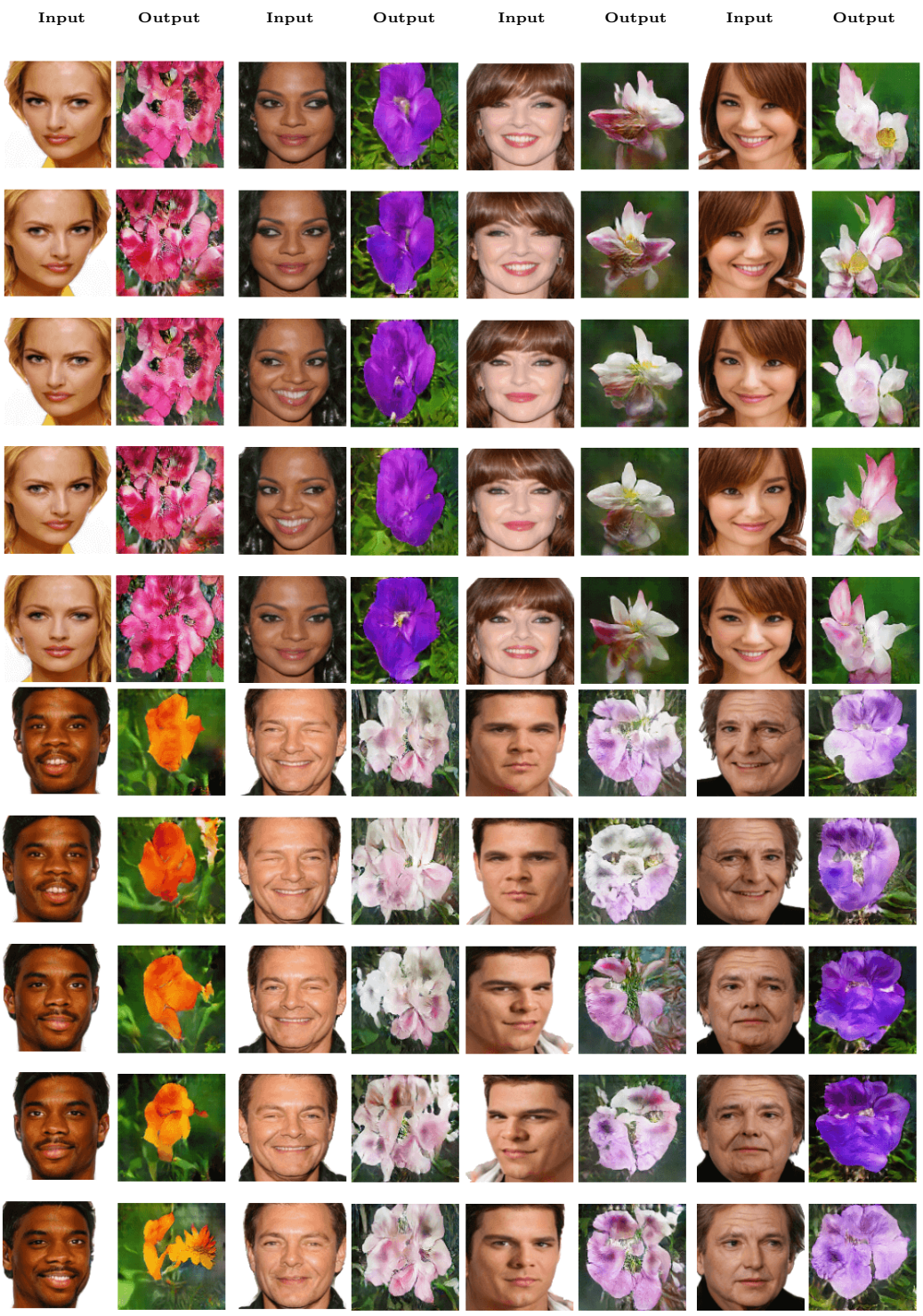

Figure B.8: Outputs of TraVeLGAN trained on the images of flowers and cropped CelebA images with removed background. The GAN was trained for 250 epochs with Adam optimizer (learning rate: 0.0002, batch size: 16, dataset size: 8,000). Given that none of the input images were seen by the GAN, it still generates satisfying images.

Appendix C

Reconstructed Images

C.1 Simple Reverse Mapping

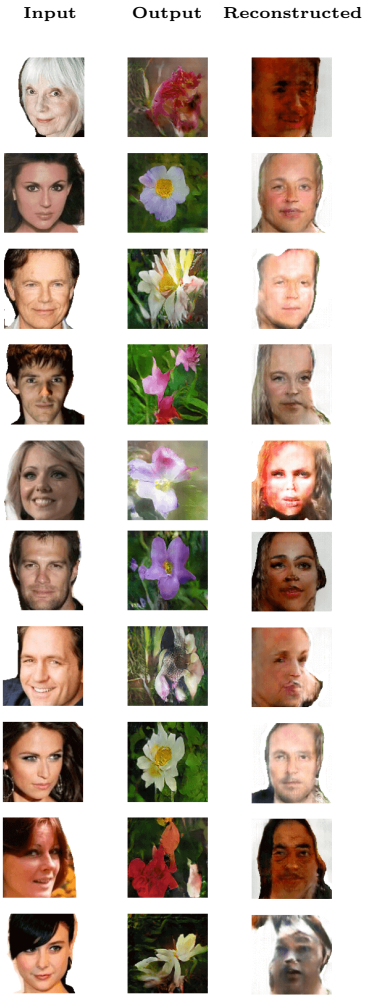

Figure C.1: Face images reconstructed by TraVeLGAN trained on the datasets of faces and flowers. The GAN was trained for 240 epochs with Adam optimizer (learning rate: 0.0002, batch size: 16, dataset size: 8,000).

C.2 Reverse Mapping with Correct Pair Sets

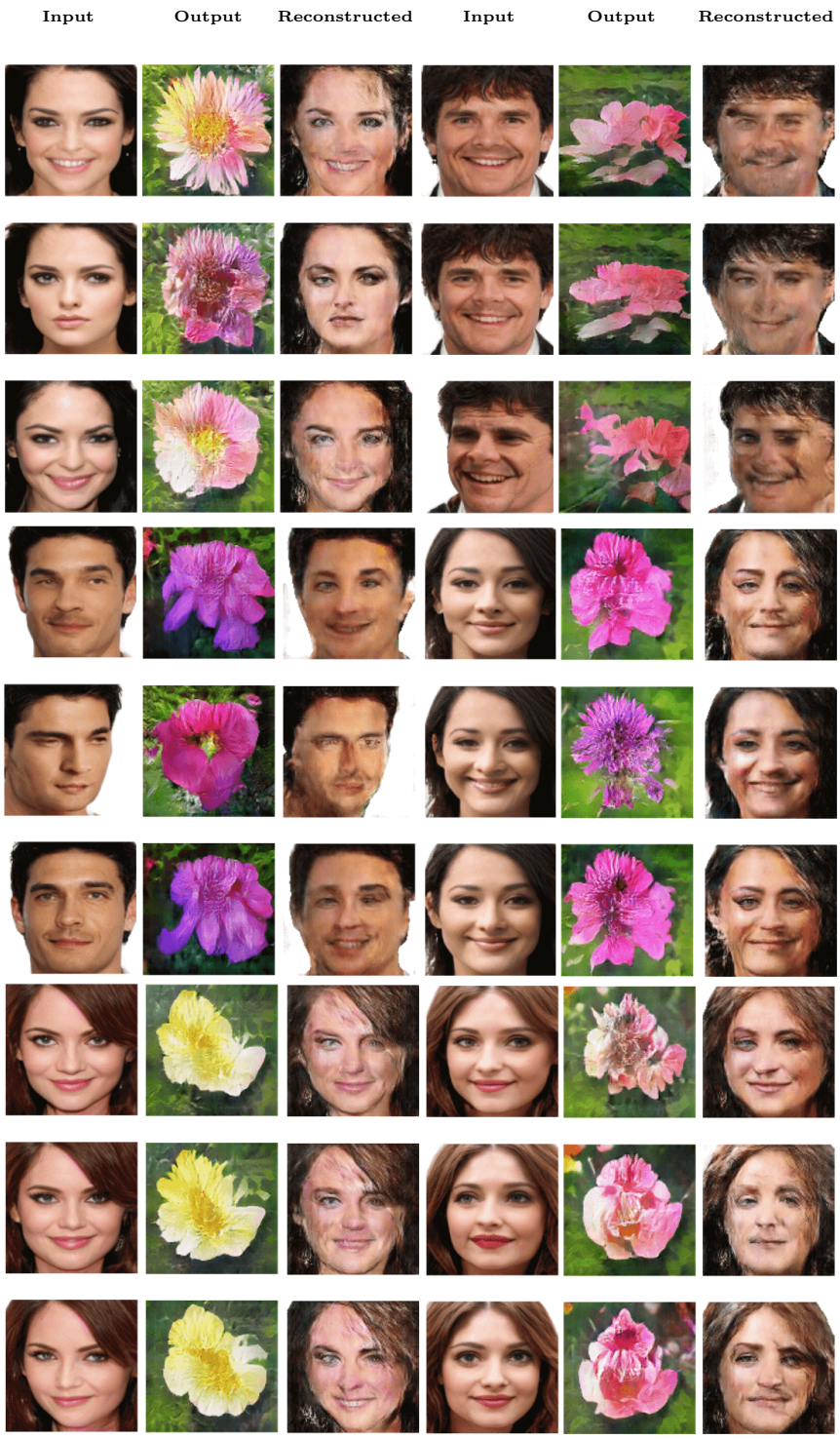

Figure C.2: The mapping learnt by a TraVeLGAN model which translates unseen images of flowers back to the images of faces. The training data for the model consisted of paired sets of faces from the CelebA dataset and flowers generated by a pre-trained TraVeLGAN network. The training of the adversary model was curated by mean squared error loss.