DISSERTATION

submitted

to the

Combined Faculty of Natural Sciences and Mathematics

of

Heidelberg University, Germany

for the degree of

Doctor of Natural Sciences

Put forward by

M.Sc. Ricard Durall López

Born in:	Barcelona, Spain
Oral examination:	03-11-2022

Deep Generative Models for Image-to-Image Translation

Advisor: PD. Dr. Ullrich Köthe

Abstract

The rise of artificial intelligence has significantly impacted the field of computer vision. In particular, deep learning has advanced the development of algorithms that comprehend visual data, and that can infer information about the environment, i.e., to mimic human vision. Among the wide variety of visual algorithms, in this thesis, we study and devise generative deep-learning models that enable image-to-image translation tasks, including style transfer and attribute manipulation. Such editing capacity might come in handy in those scenarios where additional data that contains certain properties is required, but is not available a priori, or it is quite restricted.

Over the last years, we have seen how data has become the new gold in many domains, as it has for deep-learning approaches. Indeed, the main Achilles' heel of these models is the ridiculous amount of labelled information that they crave. Therefore, we start this work by presenting a few-shot learning system that exploits alternative forms of supervision, successfully completing translation tasks with a very limited amount of samples. In this way, we open the door to less data-demanding image-to-image systems. A second focus of this thesis is the exploration and analysis of novel end-to-end models that incorporate inpainting modules to further improve their editing abilities. To that end, we assess different architectures and loss terms, together with semantic manipulations (label information) as well as with geometry manipulations (mask information), as input signal controls. The experimental evaluation of these scenarios allow us to gain insight into the role that the aforementioned elements might play when applying style and attribute modifications. Furthermore, we conduct a frequency spectrum analysis for both forged (deepfake) and generated images, paying attention to our image-to-image context as well. From this, we derive and discuss the effects that up-convolutional units might have on the final outcomes, such as artefacts in the high-frequency band. Last but not least, we present an image-to-image transformation system for a real-world application: identification of seismic events, such as diffraction and faults. The goal here is to combine two academic disciplines, i.e., computer vision and geophysics, into one project, drawing and integrating their knowledge to solve a given seismic problem.

Zusammenfassung

In den letzten Jahrzehnten wurde der Bereich Computer Vision erheblich durch künstlichen Intelligenz vorangetrieben. Durch den Erfolg von Deep Learning wurden Algorithmen entwickelt, die es ermöglicht, visuelle Daten automatisch zu verstehen. Diese erlaubt es dem Computer, das menschliche Sehen nachzuahmen und dadurch Rückschlüsse über ihr Umgebung zu ziehen. In dieser Arbeit untersuchen wir generative Deep-Learning-Modelle, die Bild-zu-Bild-Übersetzungsaufgaben ermöglichen. Diese beinhaltet auch die Übertragung von Stilen und der Manipulation von Attributen. Eine solche Art von Technik kann sich in Szenarien als nützlich erweisen, in denen zusätzliche Daten mit bestimmten Eigenschaften benötigt werden, die jedoch a priori nicht oder nur sehr eingeschränkt verfügbar sind.

Deep-Learning-Ansätze benötigen eine große Menge an Daten. Dadurch nimmt der Bedarf an Daten immer mehr an Bedeutung. Konkret brauchen solche Ansätze sogenannte annotierten Daten, um effektiv die Zielfunktion automatisch zu lernen und das Training dadurch überhaupt zu ermöglichen. Daher stellen wir in dieser Arbeit zunächst ein Lernsystem vor, das alternative Formen der Überwachung ausnutzt und Übersetzungsaufgaben mit einer sehr begrenzten Anzahl von Stichproben erfolgreich bewältigt. Auf diese Weise vereinfachen wir die datenintensiven Bild-zu-Bild-Systemen durch das automatische Generieren von Bilddaten. Ein zweiter Schwerpunkt dieser Arbeit ist die Erforschung und Analyse neuartiger End-to-End-Modelle, die Inpainting-Module zur weiteren Verbesserung ihrer Bearbeitungsfähigkeiten enthalten. Zu diesem Zweck werden verschiedene Architekturen und Zielfunktionen zusammen mit semantischen Manipulationen (Label-Informationen) sowie mit geometrischen Manipulationen (Masken-Informationen) als Eingangssignalsteuerungen untersucht. Unsere Experimente zeigen, welche Faktoren bei der Anwendung von Stil- und Attributmodifikationen eine wichtige Rolle spielen und wie diese beeinflusst werden können. Darüber hinaus führen wir eine Frequenzspektrumsanalyse durch, welche sowohl gefälschte (Deepfake) als auch für generierte Bilder beinhalten werden. Daraus leiten wir die Auswirkungen ab, die Aufwärtsfaltungseinheiten auf das Endergebnis haben können, wie z. B. Artefakte im Hochfrequenzbereich. Anschließend stellen wir ein Bild-zu-Bild-Transformationssystem für eine konkrete Anwendung vor: das Identifizieren von seismische Ereignisse wie Beugung und Verwerfung. Ziel ist es, zwei akademische Disziplinen, Computer Vision und Geophysik, in einem Projekt zu vereinen.

Dadurch können in Zukunft seismische Probleme mit Computer Vision Ansatz gelöst werden.

Publications

This dissertation has led to the following scientific publications:

- Ricard Durall, Franz-Josef Pfreundt, Janis Keuper. Semi few-shot attribute translation. International Conference on Image and Vision Computing New Zealand (IVCNZ), 2019.
- Ricard Durall, Franz-Josef Pfreundt, Ullrich Köthe, Janis Keuper. Object segmentation using pixel-wise adversarial loss. German Conference on Pattern Recognition (GCPR), 2019.
- Ricard Durall, Margret Keuper, Franz-Josef Pfreundt, Janis Keuper. Unmasking deepfakes with simple features. 2019.
- Ricard Durall, Franz-Josef Pfreundt, Janis Keuper. Local facial attribute transfer through inpainting. International Conference on Pattern Recognition (ICPR), 2020.
- Ricard Durall, Margret Keuper, Janis Keuper. Watch your up-convolution: Cnn based generative deep neural networks are failing to reproduce spectral distributions. Conference on Computer Vision and Pattern Recognition (CVPR), 2020.
- Ricard Durall, Valentin Tschannen, Franz-Josef Pfreundt, Janis Keuper. Synthesizing seismic diffractions using a generative adversarial network. Society of Exploration Geophysicists Technical Program Expanded Abstracts (SEG), 2020.
- Ricard Durall, Avraam Chatzimichailidis, Peter Labus, Janis Keuper. Combating Mode Collapse in GAN training: An Empirical Analysis using Hessian Eigenvalues. International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISGRAAP), 2021.
- Ricard Durall, Valentin Tschannen, Norman Ettrich, Janis Keuper. Generative models for the transfer of knowledge in seismic interpretation with deep learning. The Leading Edge (LTE), 2021.
- Ricard Durall, Jireh Jam, Dominik Strassel, Moi Hoon Yap, Janis Keuper. FacialGAN: Style Transfer and Attribute Manipulation on Synthetic Faces. British Machine Vision Conference (BMVC), 2021.

Acknowledgements

I would like to thank my advisor Prof. Dr. Ullrich Köthe and my co-advisor Prof. Dr.-Ing. Janis Keuper for their support and tutelage during the course of my Ph.D., and for giving me the opportunity to work on my passion over the last few years.

I am deeply grateful to my Fraunhofer colleagues, who provided an entertaining and stimulating environment, and made me feel at home. In particular, I would like to thank Kalun, Avraam, Dominik L. and Valentin for their friendship, and for those memorable light-hearted "Kicker" moments. A huge thank also goes to Dominik S. for his big help, spending countless hours keeping my greedy jobs alive.

To my family, my mother Mª Carmen, my father Esteban, my brother Daniel and all my friends, thank you for a lifetime of care and support. Keeping the best for last, to Laura, thank you for your unconditional support throughout the ups and downs of my time as a Ph.D. student.

Contents

Chapter 1

Introduction

1.1 Computer Vision

Computer vision is a field of artificial intelligence that enables machines, such as computers, mobile phones and cameras, to extract meaningful information from images, videos and other visual digital content, and to take consequent actions. In other words, if artificial intelligence enables systems to think, computer vision enables them to see. Therefore, its main goal is to develop algorithms that help artificial entities to understand the content of visual data, and to infer information about the environment, i.e., to mimic human vision.

When talking about human vision, the eyes are probably the first thing that comes to our mind. In fact, they are the sensing organs responsible for detecting light, and sending signals with the visual information along the optic nerve to the brain, which ultimately analyses and memorizes the input content. Note, however, that this is a lifetime process with a slow but steady learning curve. Only when we have been exposed to enough contexts and environments, is it then possible to understand and explore unseen surroundings, finding connections within these new scenarios. Computer vision works much the same as human vision does, except it has a far shorter head start. Hence, the artificial solutions will be time-constrained. To deal with this limitation, computer vision systems are equipped with cameras, data and algorithms, instead of retinas, optic nerves and a visual cortex, which allow the engines to speed up the learning process by several orders of magnitude. Figure 1.1 illustrates the schematic pipeline of human vision and its artificial equivalent for computer vision.

New technologies are constantly emerging and evolving to cope with the current, upcoming challenges. Machine learning, in particular deep learning, has become a dominant player for most computer vision tasks, such as image classification, object detection, image generation and semantic segmentation. The reason for its success is that deep-learning approaches involve stand-alone iterative learning to recognize specific patters of the input, rather than traditional hard-coded programming. As a result, automatic feature extraction is, nowadays, leading to

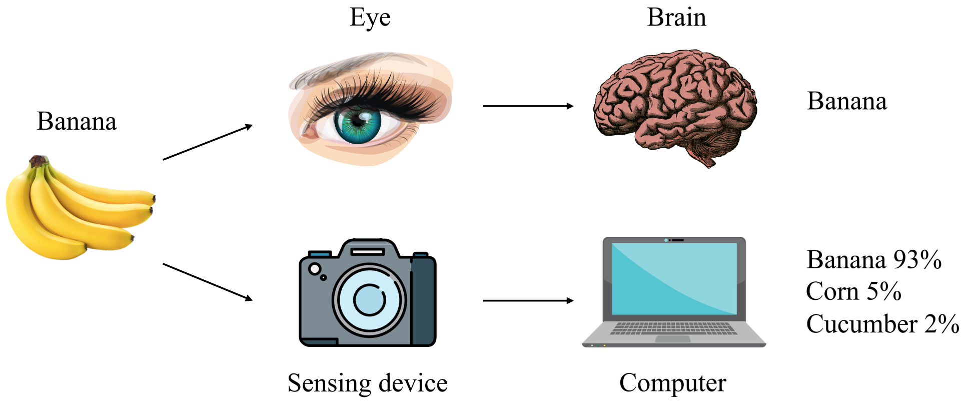


Figure 1.1. Schematic pipeline of human vision and of computer vision. Nature has always been a source of inspiration for human being. In the computer vision community, scientists are inspired by the different elements of human vision to build an artificial one.

major breakthroughs, not only in the vision community, but in many other domains. By feature extraction, we refer to the skill performed by an artificial neural network, where it has to derive task-specific indicators from given data, under the constraints of a specific problem. Historically, finding these indicators or characteristics has been a very time-consuming procedure, due to machine-learning practitioners needing to do it manually. While deep-learning approaches have eclipsed previous work in this regard, they do also have flaws that one might need to consider when using them. For example, they often suffer from data-hungriness due to the necessity of vast amounts of labelled data. Data augmentation, less data-demanding algorithms or structures with inductive biases are some techniques to try to circumvent this drawback. Nonetheless, this is still an ongoing research problem that keeps the researchers busy.

As mentioned before, deep learning has surpassed those traditional manually designed computer vision algorithms in global performances for almost every data type, in this way allowing data science teams to invest their efforts in more meaningful tasks. In fact, its success has been so profound that it has become an indispensable player on Industry 4.0. A real-use case scenario could be to train a system to recognize vehicle tyres, so that if there were any defect or tyre-related issues, they would be automatically spotted and discarded.

1.2 Generative Deep Learning

Generative modelling is a machine-learning task, where a model learns the data distribution of the training set $\boldsymbol{x}$, so that it can describe it in terms of a probabilistic model $p(\boldsymbol{x})$. The model essentially addresses the density estimation, which is a core problem in unsupervised learning. However, it is possible to extend generative models to a supervised setting, where labelled information $\boldsymbol{y}$ is also employed. To do so, the model now attempts to estimate the joint probability distribution $p(\boldsymbol{x}, \boldsymbol{y})$ by creating new plausibly instances, given the observation and the desired label. Often, generative algorithms go hand in hand with discriminative ones. Unlike generative approaches, discriminative modelling tries to estimate a conditional probability distribution $p(\boldsymbol{y}|\boldsymbol{x})$, which in practice means to model a decision boundary between the classes, ignoring the actual distribution of each class.

Suppose we have a dataset with two classes of animals, dogs and cats, and our goal is to classify them. A discriminative approach will try to find a straight line, i.e., a decision boundary that separates the two animals. In contrast, a generative approach, however, will build a model of what dogs look like, and a second model of what cats look like. Then, it will classify new instances after checking whether the new animal matches better with the dog model or with the cat model.

Learning generative models is usually a non-trivial task—especially when dealing with exact distributions in the computer vision domain. This is mainly due to the complex, high-dimensional probability distribution of the input data, i.e., images. Nevertheless, convolutional neural networks (CNNs) have been proven to be a reliable partner for generative modelling for vision, as they can automatically discover features and patterns in the input data distribution. Over the past years, thanks to the rise in neural-based approaches, deep-generative models have started to flourish, and have quickly become a very relevant technology. Current state-of-the-art modelling methods show remarkable capacity at estimating the likelihood of each observation, and at generating samples that reliably follow the real underlying distribution.

The principle of maximum likelihood defines a model that provides an estimate of a probability distribution, which is parameterized by $\boldsymbol{\theta}$. Note that we refer to the likelihood as the probability that the model assigns to the training data $\prod_{i=1}^{m} p(x^{(i)}, \boldsymbol{\theta})$, given a dataset containing m training examples. Therefore, the principle simply says to choose those parameters that maximize the likelihood of the training data

$$\boldsymbol{\theta}^* = \text{arg} \max_{\boldsymbol{\theta}} \prod_{i=1}^{m} p(x^{(i)}, \boldsymbol{\theta}). \tag{1.1}$$

If we only stick to deep-generative models that work via this principle, we can construct a simple taxonomy scheme, as shown in Figure 1.2. From top to bottom, we first split maximum likelihood into two classification nodes: explicit density models and implicit density models. Explicit density models provide an explicit parametric specification of the density function $p(\boldsymbol{x}, \boldsymbol{\theta})$, whereas implicit density models define a stochastic procedure that can directly generate data. Additionally, explicit models can be classified as a tractable or an approximate group. If the model is able to capture all the complexity of the data while still maintaining computational tractability, it will fall into the tractable group, otherwise the approximate group.

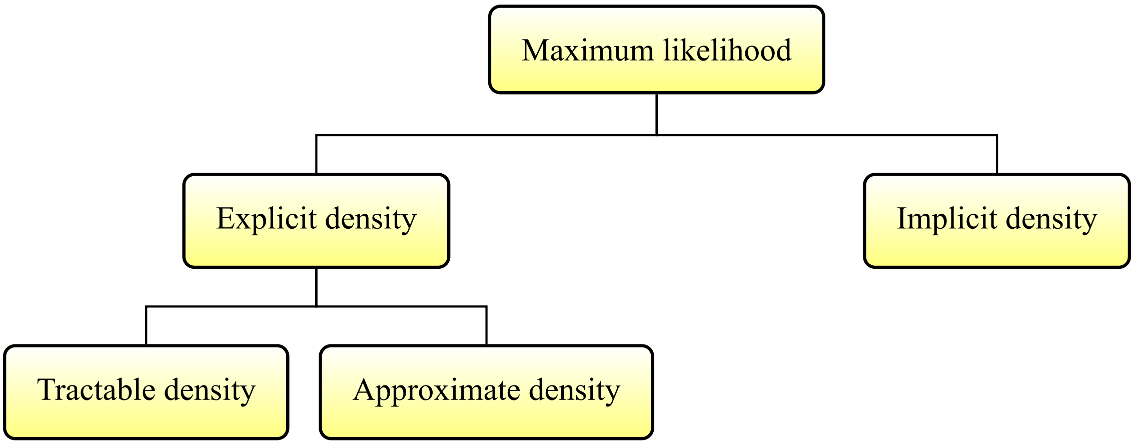


Figure 1.2. Taxonomy of deep generative models based on the principle of maximum likelihood. On the left branch of this taxonomic tree, models build an explicit density that can either be computationally tractable, e.g., flow-based generative models [1], or an approximation, e.g., variational autoencoders [2]. On the right branch of the tree, the models implicitly represent the probability distribution over the space where the data lies. To do so, these models work with an indirect interaction with the probability distribution that allows samples to be drawn from it, e.g., generative adversarial networks [3].

1.3 Generative Adversarial Networks

Generative adversarial networks (GANs) [3] are deep generative models composed of two networks: the generator G and the discriminator D (see Figure 1.3). GANs have an implicit density function that allows us to sample from the input model distribution without explicitly defining it. In order to achieve this, GANs approximate the real data distribution p_r with a surrogate (fake) data distribution p_f, by minimizing the "distance" between them. More specifically, the generator, as originally introduced by Goodfellow *et al.* [3], optimizes the Jensen-Shannon divergence between p_r and p_f, by using the feedback of the discriminator. From a game theory perspective, this optimization problem may be seen as a zero-sum game between two players: the discriminator model and the generator model. On the one hand, D tries to maximize the probability of correctly classifying the input, either as real or as fake, by updating its loss function

$$\mathcal{L}_D = \mathbb{E}_{\boldsymbol{x}\sim p_\mathrm{r}}\left[\log\left(D(\boldsymbol{x})\right)\right] + \mathbb{E}_{\boldsymbol{z}\sim p_\mathrm{z}}[\log(1 - D(G(\boldsymbol{z}))], \tag{1.2}$$

through stochastic gradient ascent. On the other hand, G tries to minimize the probability of the discriminator to correctly classify the generated data, by updating its loss function

$$\mathcal{L}_G = \mathbb{E}_{\boldsymbol{z}\sim p_\mathrm{z}}[\log(1 - D(G(\boldsymbol{z}))] \tag{1.3}$$

via stochastic gradient descent. Note that $\boldsymbol{x}$ are data samples and $\boldsymbol{z}$ is random noise.

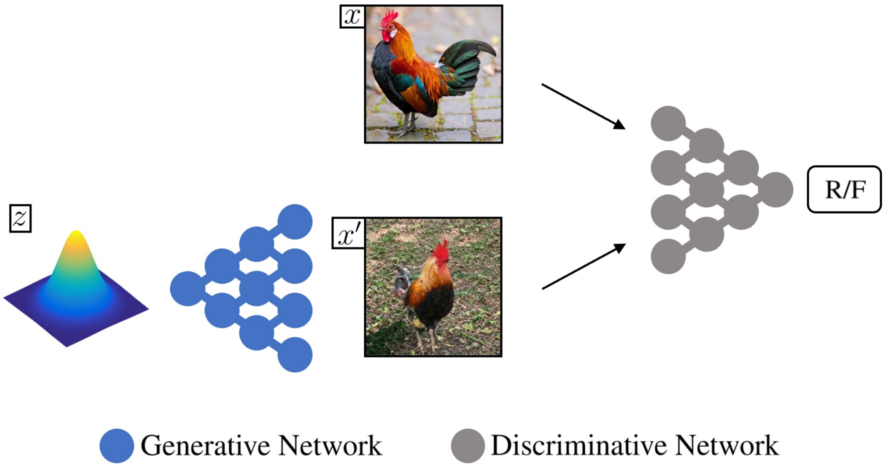


Figure 1.3. Schematic pipeline of a GAN. A GAN consists of two neural networks: the generator and the discriminator. The generator is fed with random noise z, and its goal is to transform it into a sample x' that follows the real data distribution. To achieve that, x' is sent to the discriminator, and through back-propagation, the discriminator provides gradient information to the generator, which the latter will use to improve its outputs. Simultaneously, the discriminator also learns to distinguish between real x (R) and generated sample x' (F), making the whole framework an adversarial setup.

The convergence of this minimax game is intrinsically different from standard cases with a single objective function, since GANs may converge to a Nash-equilibrium. This will occur when the generator synthesizes samples that look as if they were drawn from the p_r. As a result, the discriminator is left indecisive, no longer able to consistently determine the nature of its input. In practice, since the objectives are non-convex, by using local gradient information we can only guarantee to find local optima, i.e., a local Nash-equilibrium [4].

GANs have had and still have a tremendous impact on the computer vision community; their frameworks have been adopted in numerous deep-learning fields. Semi-supervised learning [5], transfer learning [6], reinforcement learning [7], and feature selection are just a few examples where GANs have shown remarkable results in recent years. In terms of final applications, the most important work lays in the image-processing field, where novel contributions are continuously appearing in many diverse subfields, such as image synthesis [8–10], image super-resolution [11–13], image-to-image translation [14–16], inpainting [17–19] and semantic segmentation [20–22].

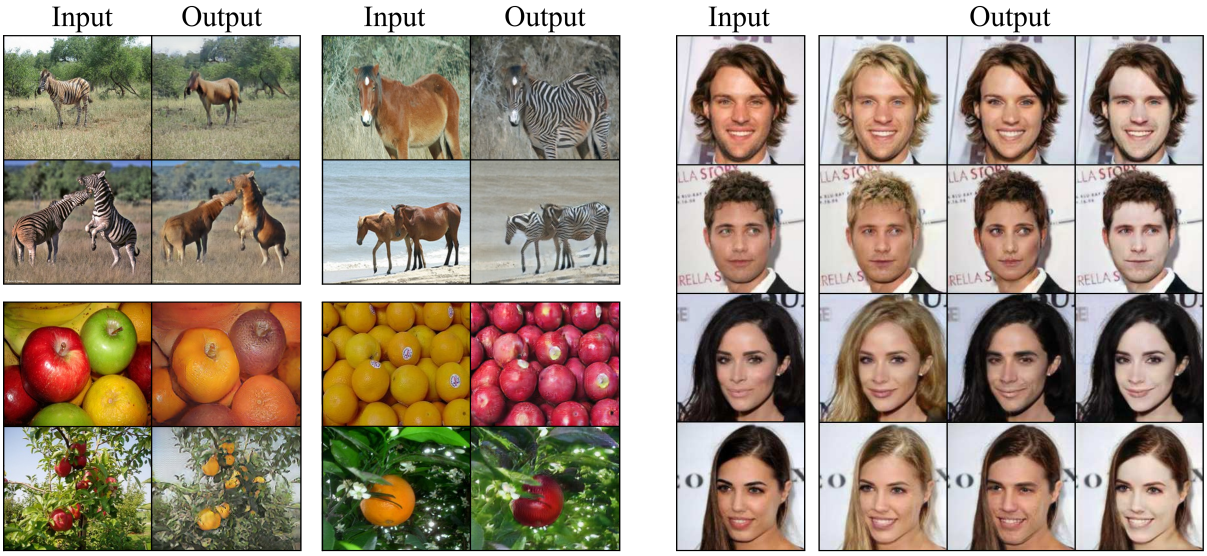


Figure 1.4. Examples of image-to-image translation [15, 16]. (Left) Style transfer from *zebra* to *horse*, and from *apple* to *orange*, and vice versa. (Right) Attribute transfer where the outputs contain the target attribute: *blond hair*, *gender* and *pale skin*.

1.4 Image-to-Image Translation

Image manipulation is a challenging task in computer vision due to the cumbersomeness of generating images, or parts of them, whilst preserving the subtle texture and patterns of the source input. Image-to-image translation is a class of image manipulation, where the goal is to convert an input image x_A from a source domain A to a target domain B with the intrinsic source content preserved and the extrinsic target properties transferred. To that end, we need to train a mapping network $G_{A\to B}$ that generates an image x_{AB} indistinguishable from the target domain image $x_B \in B$ given the input source image $x_A \in A$. Mathematically, we can formulate this translation process as

$$x_{AB} \in B : x_{AB} = G_{A\to B}(x_A). \tag{1.4}$$

Solving the problem of image-to-image translation is attracting increasing attention because of its wide practical application value. For example, it can be used in the medical domain, where the data available is often scarce and very sensitive. Using image-to-image translation could help to create synthetic datasets. However, many of the translation approaches still require supervised learning settings in which pairs of corresponding source and target images need to be available—and this is not always possible. As a consequence, there is an increasing motivation towards unsupervised solutions, where the source and target image sets are not paired examples between the two domains. In recent years, GAN-based models for image-to-image translation have achieved great success producing lifelike images with a high degree of variability, i.e., providing diverse geometry, textures and colours. As a result, many computer vision tasks have benefited from it, by exploiting GANs for common editing techniques, such as image segmentation [23, 24] colourization [25, 26], style transfer [14, 27] and attribute manipulation [16, 28].

Figure 1.4 illustrates a few image-to-image translations. On the left side, there are examples of style transfer, where for example, given a set of images of *zebras* as the source domain, the system translates them to a *horse* style domain, and vice versa. On the right side, there are examples of attribute transfer, where, given a set of input faces, the system modifies the target attribute, such as *hair colour*, *gender* and *skin colour*.

1.5 Contribution

This thesis provides the following main contributions:

- We highlight the important weakness of attribute transfer regarding its label dependency, since almost all existent methods are based on vast amounts of annotated data. A few-shot meta-learning strategy is proposed to cope with such a limitation, by allowing our model to transfer when scarce, new attributes appear.

- The robustness of attribute transfer approaches can be further improved by employing additional inpainting techniques. More specifically, we suggest incorporating an inpainting block that focuses on the region of interest—where the attributes are located—while keeping the background unmodified yet consistent with the modifications.

- More advanced models are those that not only can deal with attribute transfer, but also with style transfer. To that end, we introduce a novel model that enables rich, simultaneous style transfers and interactive attribute manipulation, while maintaining the featured person's identity.

- Currently, there are many techniques that can seamlessly stitch anyone in the world into a photo they never actually participated in. Such altered content might become very dangerous, as they can be widely propagated over the Internet. We address this problem by detecting artificial image content using a simple yet effective method based on classical frequency domain analysis.

- We present a use case application, where the system benefits from transferring the style domain of real image so that there is no need for labelled real data. To achieve that, we propose an end-to-end system for seismic applications that reduces the generalization gap between synthetic training data and real testing data.

1.6 Thesis Organization

This thesis consists of five main parts, not counting the introduction and conclusion. A brief overview of the chapters is given hereunder.

Chapter 2. In the last few years, researchers have worked on improving few-shot learning in many different domains. Nonetheless, for image-generation tasks it is still underexplored. Therefore, before discussing advanced image-to-image translation systems, we study the possibility of few-shot meta-learning approaches when facing attribute-translation tasks. In this way, we gain some insight into understanding to what extent the labelled data is required for such applications. Specifically, we train a model on certain attributes and test it on others that were not seen during training; in this context, the attributes are treated as classes. This work was previously published at IVCNZ 2019.

Chapter 3. We devise a method to learn attribute translation based on semantic-manipulation (label information). To do so, we build an end-to-end approach that incorporates an inpainting network that helps to improve the translation results. In particular, we take advantage of the fact that most facial attributes are induced by local structures, e.g., relative position between the eyes and ears, resulting in more realistic synthesis and more manageable attribute control. The quality of generated results is assessed on experiments using different target attributes. We believe that merging techniques, i.e., attribute transfer and inpainting, can help the field to progress at a faster pace. This work was previously published at ICPR 2020.

Chapter 4. Besides attribute translation, this thesis addresses the style-transfer problem as well. To that end, we design a novel architecture, composed of a generative network, a style network, a segmentation network, and a discriminative network, which sequentially combines each of these blocks. It uses both semantic- and geometry-manipulation (label and mask information) as input signals, to deal with style transfers as well as interactive facial-attribute manipulation. Moreover, we introduce a local segmentation loss that guarantees a pixel-wise attribute control. Finally, we evaluate the results, both qualitatively and quantitatively, and report state-of-the-art scores on reference-guided generation. This work was previously published at BMVC 2021.

Chapter 5. The increasing sophistication of smartphones and the growth of social networks have led to a gigantic amount of new digital content. This tremendous use of digital images together with GANs' improvement have opened the door to a new vein of AI-based synthetic image generation, leading to a fast dissemination of high-quality modified content. While significant developments have been made in image forgery detection, it still remains an ongoing research task. In this chapter, we tackle the problem of detecting artificial image content, paying attention to the image-to-image cases too. This work was previously published at CVPR 2020.

Chapter 6. We present an image-to-image transformation system for a real-world application: identification of seismic events, such as diffractions and faults. In particular, an end-to-end model that allows synthetic patterns to be transferred (style transfer) into generated data, as well as the identification of seismic events to be automated. We formulate the problem as a supervised semantic-segmentation task, where the combination of synthetic data and its realistic transformation is the key to provide a larger, more realistic variety of samples, with their ground-truth, at minimal manual labour cost. We show that our proposed model outperforms standard methods, which traditionally have only been trained on purely synthetic data. This work was previously published at TLE 2021.

Chapter 7. Last but not least, we conclude this thesis with a final discussion summarizing the most interesting findings regarding attribute and style translation. We also draw some conclusions on what might be promising future directions.

Chapter 2

Few-shot Learning for Attribute Transfer

Deep-learning models for computer vision have experienced tremendous growth in recent years, achieving state-of-the-art results in a large range of tasks, from image classification [29, 30], to semantic segmentation [31, 32], and generative models [3, 10]. Some factors fuelling this dramatic expansion include the increase in the sophistication of learning algorithms and the growing computing capability of machines. However, most of these seminal outcomes rely heavily on deep-learning models, which need to be trained in both abundant labelled data and rich sample diversity. The high annotation cost, as well as the scarcity of data in some domains, significantly limit the applicability of some of these advanced vision systems. In contrast, humans tend to be highly effective at recognizing new instances with extremely few annotated examples, or even without any. This is possibly due to our deep understanding of our surroundings, our innate capacity to establish connections between new concepts, and our own prior knowledge and experience. It is thus of great interest to develop models that can mimic such behaviour, and consequently, to generalize when new cases appear. Few-shot learning tasks have been defined for this purpose.

Few-shot learning is a powerful paradigm of limited supervision, where a model can be trained on a set of classes with a very small amount of training data, contrary to the normal practice involving large quantities. For example, if we have a task of categorizing beetle species from photos where some rare specimens lack pictures, we can use a few-shot classifier to cope with this limitation. In the case of only having one image of a particular species, it would be a one-shot classifier. In the extreme case where no image is present in a certain category, it would be a zero-shot machine learning problem. Among few-shot learning methods, two main approaches exist: a data-level approach and parameter-level approach (see Figure 2.1). The data-level approach is based on the concept that, whenever there is an insufficient amount of data to build a model without underfitting or overfitting, more data samples need to be added. Data augmentation is the classic technique to deal with this issue. This involves increasing the dataset by creating slightly modified copies of already existing samples. A feasible alternative is to

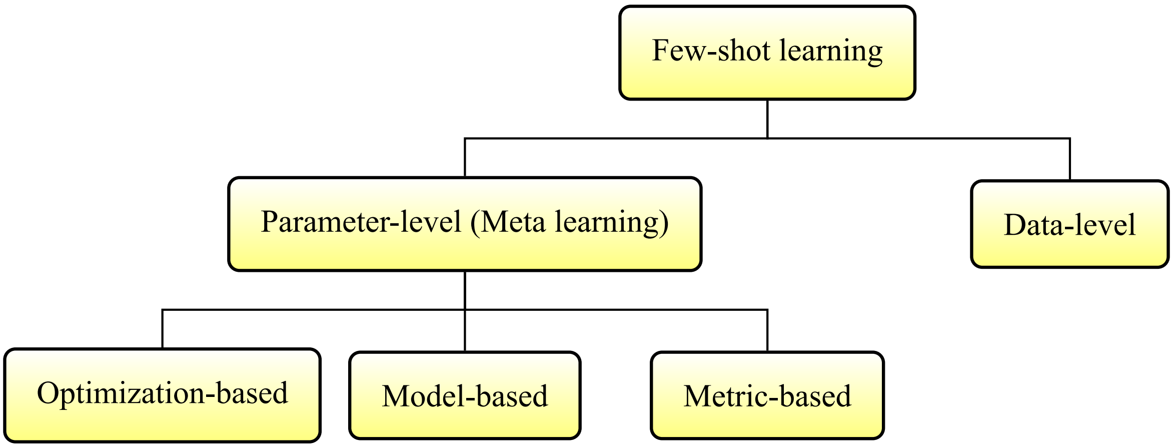


Figure 2.1. Taxonomy of few-shot learning. This work is based on the meta-learning branch, more specifically, on an optimization-based approach.

generate new data using generative models like GANs. On the other hand, instead of generating more data, we can enhance the performance of the model by limiting its parameter space using regularizations and customized loss functions. In this manner, parameter-level algorithms can help to generalize, even with a limited number of samples. They manage to become generalizing engines thanks to their capacity to find the best route in the parameter space, and to give optimal predictions with little data. This second approach is also called meta learning.

One promising direction of few-shot meta learning is image classification. In fact, it has already attracted considerable attention [33–37] given its success in several practical applications. The main idea behind this work is the accumulation of experience from learning multiple training tasks so that the performance holds when dealing with potentially unseen tasks. Besides classification tasks, meta learning has also been proven to be suitable for other types of applications. Recent work on generative modelling has implemented meta-learning techniques, showing their viability of image generation [38–41]. These novel approaches can lead to a breakthrough in generative models, since most others still require vast amounts of data points to generate comprehensible images. In particular, meta learning could potentially be very relevant for GAN-based approaches, by contributing to continuing to push the current state of the art forward in a wide variety of generative tasks, such as style transfer or attribute transfer.

In this chapter, we address the label limitation that supervised generative methods suffer from, by proposing a new conditional GAN-based approach, trained in a meta-training fashion. This allows us to perform attribute transfer using just a few labelled samples from a new class.

2.1 Background

In this section, we define the few-shot paradigm, and provide an overview of the meta-learning algorithms. Additionally, we introduce related literature and some of its most prominent work.

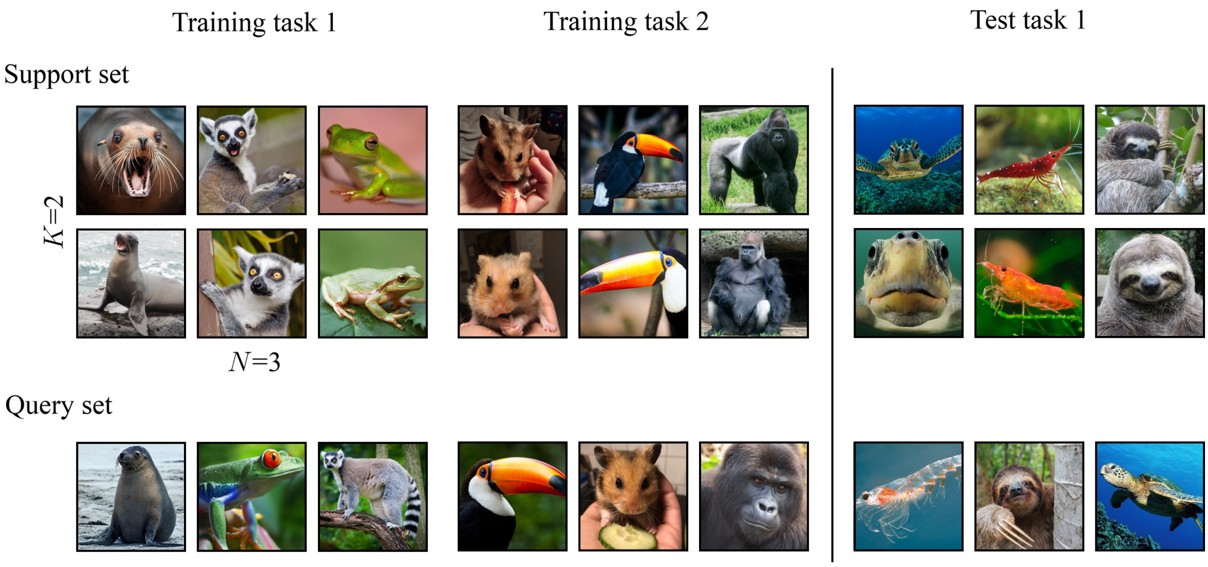


Figure 2.2. Meta-learning framework for a 2-shot, 3-class classification. At training time, for each training task, the model trains using the support set and evaluates its performance on the query set. At testing time, we use an unseen set of tasks, and reevaluate the results following the same procedure. Ideally, the model will be able to classify the new tasks correctly as well.

2.1.1 Preliminaries

While machine-learning systems have surpassed humans at many tasks, they generally need far more data to reach the same level of performance. Nonetheless, it is not completely fair to directly compare humans to algorithms, since we confront new tasks with a large amount of prior knowledge. In other words, we do not learn a new skill from scratch, but rather by fine-tuning and recombining sets of pre-existing skills. Meta learning is a machine-learning subfield, also known as learning to learn [42], which aims to design models that can learn new skills or adapt to unseen environments rapidly with just a few training examples. It can solve the data-hungriness dependency while maintaining a similar rate of performance. To achieve this goal, meta-learning paradigms are trained using a variety of learning tasks (training tasks), and optimized for the best performance on a set of several tasks, including those potentially unseen (test tasks). Each of these tasks is associated with a labelled dataset that can consist of a set of images for a classification problem; a set of state, action and reward for a reinforcement setup; or even a set of attributes for an image generation task.

Suppose we have an image classification problem in a meta-learning context. In particular, consider the typical K-shot, N-class classification, where each training task includes a support set with K examples, N classes, and a query set to evaluate the performance of the current classification task. The main idea here is that the system is repeatedly trained with different training tasks that match the structure of the testing task. At each training task, the model updates its parameters. The loss function is determined by the classification performance on the query set of the current training task, based on knowledge gained from its support set. Since the network is presented with a different task at each time step, it must learn how to discriminate data classes in general, rather than a particular subset of classes. Finally, the testing task can be quickly learnt with few samples following the same steps. Figure 2.2 depicts a 2-shot, 3-class image classification scenario for nine different animals.

2.1.2 Related Work

Research literature on few-shot meta learning exhibits a great diversity of approaches, each of which with its own flavour. In broad terms, they can be divided into three main categories: metric-based, model-based and optimization-based methods. 1) Metric-based methods [33,36, 37,43] during the training tasks learn embeddings that separate different classes. Essentially, these methods exploit prior knowledge about similarity, and learn an efficient distance metric. 2) Model-based methods [44,45] use a network with external or internal memory to learn to store “experience” during training, in order to generalize on new tasks. 3) Optimization-based methods [34,35,46–48] intend to constrain the optimization algorithm to choose parameters to speed up the learning process so that the model can perform well with few examples. In our work, we focus especially on the latter category.

Optimization-based methods, also known as initialization-based methods, tackle the meta-learning problem by “learning to fine-tune”. The idea behind this approach is to learn a good model initialization, i.e., the weight parameters. If we bring this problem into the loss landscape, the aim of this method is to find a rather flat region where the model can generalize well when new instances appear. In order to be good at generalization while fulfilling its main goal, e.g., a classification task, this kind of model is two-fold constrained. It is limited in terms of the amount of labelled data, and the gradient update steps. In general, these initialisation-based methods are capable of achieving fast and effective adaptations with scarce training examples, as long as domain shifts between base and novel classes are not too abrupt.

Although most meta-learning applications address classification tasks, new scenarios and contexts are being constantly pushed forward. For instance, it is possible to extend the few-shot meta-learning definition to fit generative modelling setups. To the best of our knowledge, Lake *et al.* [38] were some of the pioneers in successfully combining few-shot techniques with image generation, i.e., few-shot image generation. In this first approach, the model was trained on handwritten character images and their strokes, sampled from the Omniglot [42] dataset. This yielded a novel system that was able to generate binary samples. Driven by the idea of taking a more general approach, Rezende *et al.* [39] presented a sequential generative model which was only trained on purely handwritten characters—this time with no stroke information. Despite the clear advantage of having a simplified input, this second approach suffered from lengthy sequential inference as an inherent consequence of its framework. In a later publication, Bartunov and Vetrov [40] tackled this issue by suggesting matching networks, known as memory-assisted networks. This new implementation generated binary images from the Omniglot dataset, using few-shot learning with fast inference periods. Last but not least, the FIGR [41] framework proposed a radical change by integrating GANs as a fundamental block of pipeline structure. In this way, they could overcome the scaleability limitation found in the previous models. Furthermore, this latter work presented a more extensive set of experiments, including additional results using the MNIST [49] and the FIGR-8 [41] datasets.

2.2 Contributions

In this work, we focus on a challenging meta-learning problem, where we define a scenario of few-shot attribute translation. In particular, we propose a novel approach, capable of performing facial-attribute transfer, while being restricted in terms of quantity of available labelled data. In order to achieve this goal, we employ an optimization-based method, based on the Reptile [47] algorithm, where we sequentially train two independent but topologically equal neural networks. One of the major challenges comes from the lack of work on attribute translation in a meta-learning context. As we mentioned, most of the existing few-shot algorithms are applied to classification tasks. In our approach, we extract techniques from these methods [33–37] and from FIGR-8 [41], and we leverage those ideas that can be implemented in our use case. That is to say, we apply similar principles but with attributes. We name it “Semi Few-shot Generative Adversarial Network” (Semi Few-shotGAN) because it uses GANs as a main building block, and there is only one class: human faces. Notice that although we have only one kind of image, they do have different attributes. As a result, we have a scenario where we have more than one class represented by the attributes. Therefore, this is not a trivial setup, since the network still needs to learn the ability to perform image-to-image transformation for untrained target domains (attributes) with few examples. In particular, we apply our model to the CelebA dataset, and control several hair-colour attributes. Overall, our contributions are summarized as follows:

- We propose a novel generative adversarial network based on meta-learning trained for end-to-end attribute translation.
- We demonstrate how we can successfully learn to transfer hair attributes in a few-shot fashion, opening the door to further research with more complex translations.
- We provide qualitative results based on the CelebA dataset showing the effectiveness of our approach.

2.3 Method

We start this section introducing the problem definition. Then, we present the training of the model in two levels of abstraction, which train separately but in a symbiotic manner: 1) from a few-shot perspective, we give a detailed explanation of the configuration of our meta-learning framework; 2) from a generative model perspective, we define a conditional GAN framework capable of conducting the attribute-translation task. After that, we merge both perspectives so as to have a few-shot attribute-transfer system. Eventually, we close this section with the description of our end-to-end training strategy.

2.3.1 Problem Definition

Our proposed model addresses the attribute-translation task in a meta-learning context. Given a source image $x \in \mathbb{R}^{H\times W\times 3}$, and its attribute task $\tau \in \mathbb{R}^{1\times N}$, our goal is to train a model that can translate facial attributes even when the target task is very restricted in quantity of samples. Note that H and W are the height and width of the data, respectively, and N is the category number of the attribute tasks.

2.3.2 Meta-learning Model Architecture

We define a few-shot attribute-transfer problem, based on the meta-learning setup Reptile [47] and FIGR [41]. In this scenario, we have access to a set of tasks $\boldsymbol{\tau}$, where each individual task τ is an attribute-transfer problem with one target attribute and its corresponding loss $\mathcal{L}_\tau$. Ideally, this loss term assesses the ability to generate realistic samples belonging to the target domain of the attributes. We tackle this problem by building a meta-learning model that learns an initialization for the parameters of a neural network $\boldsymbol{\theta}$ within a limited k updates, such that when we optimize these parameters at testing time, learning will be fast, i.e., the model will generalize well from few examples from the test task. Such a learning algorithm can be defined as

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{\boldsymbol{\tau}}[\mathcal{L}_{\boldsymbol{\tau}}(U^k_{\boldsymbol{\tau}}(\boldsymbol{\theta}))], \tag{2.1}$$

where $U^k_{\boldsymbol{\tau}}(\boldsymbol{\theta})$ is usually the stochastic gradient descent operator from the minimization problem.

The architecture of our method consists of two independent neural blocks with the same topology: one called inner-loop structure and the other outer-loop structure. As a result, the optimization algorithm is split into two parts as well. The first part, known as meta training, requires the inner structure to be trained. To accomplish this, we sample tasks $\boldsymbol{\tau}$ from the training dataset. This dataset contains a virtually infinite amount of samples belonging to certain classes/attributes. We then use these samples to train the inner-loop, and then we update the parameter $\boldsymbol{\theta}$. After k updates, we start with the second part of the optimization: the outer structure training. To train this part, we set the gradient of the outer optimizer to be equal to $\tilde{\boldsymbol{\theta}} - \boldsymbol{\theta}$, and we take one step on the optimizer, i.e., back-propagation. Notice that this second network never runs forward-propagation. Algorithm 1 describes the pseudocode of the meta-learning model.

Algorithm 1 Reptile algorithm [47].

```
Initialize $\boldsymbol{\theta}$, the vector of initial parameters.
for iteration $= 1, 2, 3...$ do
    Sample a task $\tau$
    Compute $\tilde{\boldsymbol{\theta}} = U_\tau^k(\boldsymbol{\theta})$, denoting $k$ steps of the optimizer    {Here starts inner-loop}
    Update $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \epsilon(\tilde{\boldsymbol{\theta}} - \boldsymbol{\theta})$    {Here starts outer-loop}
end for
```

One of the main advantages of this algorithm is the high degree of flexibility it offers. With a few modifications, this framework can accommodate different types of problems. In this work, we make use of said property, and integrate a generative network inside the algorithm, more specifically, a GAN architecture in each loop.

2.3.3 Attribute-Transfer Model Architecture

We build our attribute-transfer baseline based on the recent state-of-the-art image-to-image translation model StarGAN [16], which in turn was an adaptation of CycleGAN [15]. In broad terms, the architecture resembles the vanilla GAN [3] as it also employs an adversarial framework made of a generator and a discriminator. However, in this scenario, the goal of the generator is to synthesize samples containing the target attribute, but that are also realistic enough to deceive the discriminator. The network architecture of our proposal is shown in Figure 2.3.

Discriminative Network. The discriminator D takes samples of real images $\boldsymbol{x}$ and generated images $\boldsymbol{x'}$, and tries to classify them correctly. This procedure consists of two classification steps. First, there is a classifier that evaluates whether the input image looks realistic or not. For this task, we implement an adversarial loss $\mathcal{L}_{\text{adv}}$ that employs Wasserstein distance [50]—more specifically, a variation called Wasserstein with gradient penalty [51], defined for the generator as

$$\mathcal{L}_{\text{adv,gen}} = -\,\mathbb{E}_{\boldsymbol{x'}}[D(\boldsymbol{x'}))], \tag{2.2}$$

and for the discriminator as

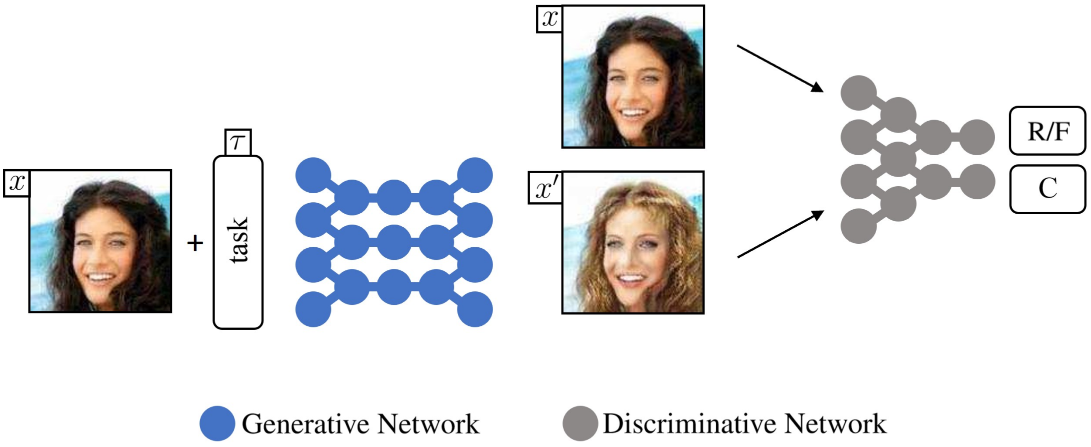


Figure 2.3. Overview of the attribute-transfer model based on GAN structure. It consists of a generator G and a discriminator D, which are trained in alternative pattern [3]. G takes the source (input) image x, and the task τ, and learns to generate an output image x', according to a target domain. On the other hand, D learns to distinguish between real x (R) and generated images x' (F), and to classify them according to their domain (C). The figure shows an attribute-transfer example, where G changes the original hair colour to a target one (*blond*), and D evaluates the results.

$$\mathcal{L}_{\text{adv,disc}} = -\,\mathbb{E}_{\boldsymbol{x}}[D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{x}'}[D(\boldsymbol{x}'))] + \lambda_{\text{gp}}\,\mathbb{E}_{\boldsymbol{x}'}[(||\nabla\, D(\alpha\boldsymbol{x} + (1-\alpha)\boldsymbol{x}')||_2 - 1)^2], \quad (2.3)$$

where α is a random variable following the discrete uniform distribution over the set $\{0, 1\}$.

Then, inspired by Triple-GAN [52], there is a second classifier that is in charge of the attribute-domain classification. This additional loss function $\mathcal{L}_{\text{class}}$ computes the binary cross-entropy, penalizing incorrect image-domain transformations. For the training of the generator, this term can be written as

$$\mathcal{L}_{\text{class,gen}} = \mathbb{E}_{\boldsymbol{x}'}[-\log D(\boldsymbol{\tau}_{\text{target}}\,|\,\boldsymbol{x}')], \quad (2.4)$$

where $\boldsymbol{\tau}_{\text{target}}$ are the target domains (attribute) of $\boldsymbol{x}'$, and for the discriminator as

$$\mathcal{L}_{\text{class,disc}} = \mathbb{E}_{\boldsymbol{x}}[-\log D(\boldsymbol{\tau}_{\text{original}}\,|\,\boldsymbol{x})], \quad (2.5)$$

where $\boldsymbol{\tau}_{\text{original}}$ are the source domains of $\boldsymbol{x}$.

The overall discriminator loss can be formulated as

$$\mathcal{L}_{\text{disc}} = \lambda_{\text{adv}}\,\mathcal{L}_{\text{adv,disc}} + \lambda_{\text{class}}\,\mathcal{L}_{\text{class,disc}}. \quad (2.6)$$

Generative Network. The aim of the generator G is to learn mappings between multiple attribute domains. To achieve this, we train G to translate source (input) images $\boldsymbol{x}$ into output images $\boldsymbol{x}'$, where $\boldsymbol{x}'$ have to belong to the target domains, defined by the target tasks $\boldsymbol{\tau}_{\text{target}}$;

$$\boldsymbol{x}' = G(\boldsymbol{x}, \boldsymbol{\tau}_{\text{target}}). \tag{2.7}$$

Similarly to the discriminator, the generator loss also uses an adversarial loss $\mathcal{L}_{\text{adv,gen}}$ (see Equation 2.2) and a domain classification loss $\mathcal{L}_{\text{class,gen}}$ (see Equation 2.6). However, when training the generator, these losses are fed exclusively with synthetic examples $\boldsymbol{x}'$ and using the target tasks $\boldsymbol{\tau}_{\text{target}}$. Additionally, there is a third loss term called cycle consistency loss $\mathcal{L}_{\text{cycle}}$ [15], whose mission is to guarantee that the translated images $\boldsymbol{x}'$ preserve the content of their input images $\boldsymbol{x}$, while changing only the domain-related parts. To achieve this, G performs an entire cyclic translation $\boldsymbol{x} \to \boldsymbol{x}' \to \boldsymbol{x}''$, forcing $\boldsymbol{\tau}$ to be crucial for moving between domains. First, it translates the original images $\boldsymbol{x}$ into images in the target domains $\boldsymbol{x}'$ (conditioned on $\boldsymbol{\tau}_{\text{target}}$), and then, it reconstructs the original images from the translated images $\boldsymbol{x}''$ (conditioned on $\boldsymbol{\tau}_{\text{orginal}}$). This cyclic procedure can be written as

$$\begin{aligned} \mathcal{L}_{\text{cycle}} &= ||\boldsymbol{x} - \boldsymbol{x}''||_1, \\ \text{with } \boldsymbol{x}'' &= G(\boldsymbol{x}', \boldsymbol{\tau}_{\text{orginal}}). \end{aligned} \tag{2.8}$$

The overall generator loss can be defined as

$$\mathcal{L}_{\text{gen}} = \lambda_{\text{adv}}\,\mathcal{L}_{\text{adv,gen}} + \lambda_{\text{class}}\,\mathcal{L}_{\text{class,gen}} + \lambda_{\text{cycle}}\,\mathcal{L}_{\text{cycle}}. \tag{2.9}$$

2.3.4 End-to-End Model Architecture

In order to merge the GAN's framework with the meta-learning algorithm, we first need to duplicate the GAN architecture. In other words, we create two models: GAN_{A} and GAN_{B}. Then, we set GAN_{A} in the inner-loop structure, with its generator G_{A} and discriminator D_{A}, and GAN_{B} in the outer-loop structure, with its generator G_{B} and discriminator D_{B}. In Figure 2.4 an overview of the scheme is depicted.

Algorithm 2 describes the adapted Reptile pseudocode for meta-training the attribute-transfer GAN system. Once the meta-training part converges to a minimum, i.e., the model generates realistic samples within the target domain, the meta-testing process can take place. Meta-testing refers to the learning part, where a new, reduced task needs to be learnt, which in our case is a certain attribute. Algorithm 3 depicts how the model can quickly adapt to incorporate the new task on its generative setup. For that purpose, we copy the final meta-learning parameters $\boldsymbol{\theta}_{\text{B}}$ to the parameters $\boldsymbol{\theta}_{\text{A}}$, and we train GAN_{A} on the new task. Note that now the k value might be set to a large value, while for the meta-training part it is 1 by default. Intuitively, the reason for doing more gradient steps during meta-testing is to prioritize learning features that are hard to reach when there is a limited amount of samples.

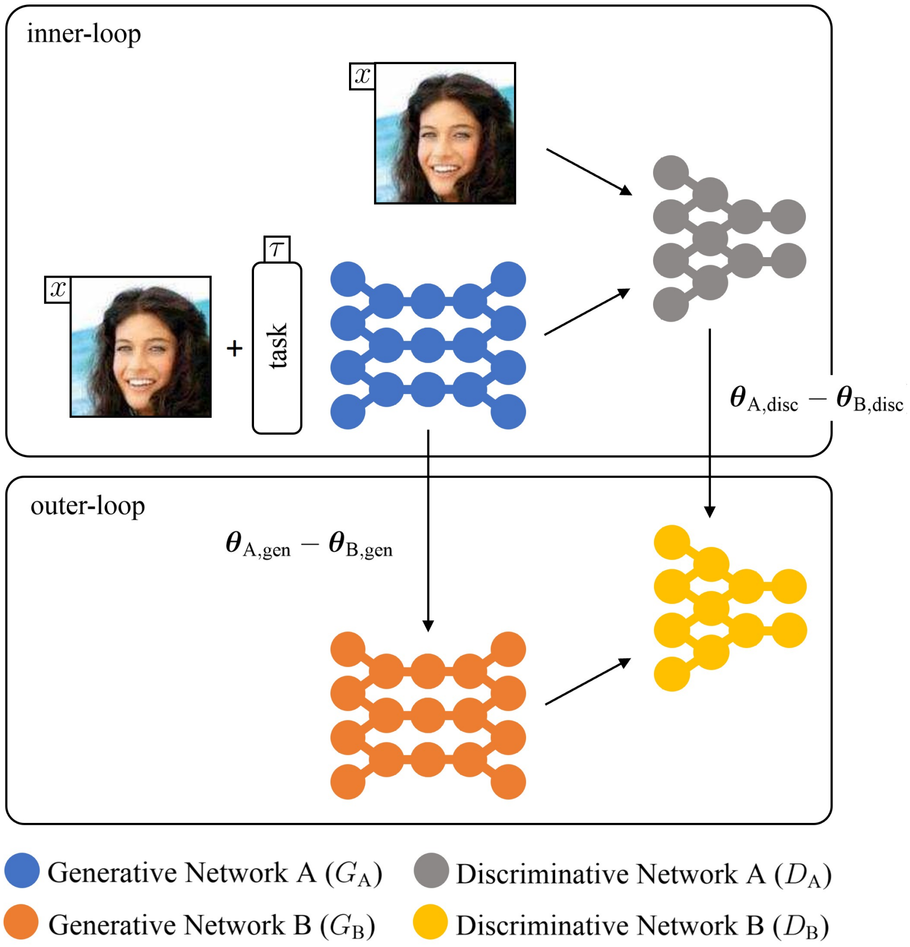

inner-loop
x
x
+
τ
task
$\boldsymbol{\theta}_{\text{A,disc}} - \boldsymbol{\theta}_{\text{B,disc}}$
outer-loop
$\boldsymbol{\theta}_{\text{A,gen}} - \boldsymbol{\theta}_{\text{B,gen}}$
Generative Network A (G_{A})
Discriminative Network A (D_{A})
Generative Network B (G_{B})
Discriminative Network B (D_{B})

Figure 2.4. Overview of our few-shot attribute-transfer architecture. It consists of an inner-loop structure and an outer-loop structure. Each of these structures contains a GAN model made of a generator G and a discriminator D. The inner-loop trains GAN_{A} by optimizing the minimax game between G_{A} and D_{A} on a task τ. The outer-loop sets its gradient to $\boldsymbol{\theta}_{\text{A}} - \boldsymbol{\theta}_{\text{B}}$, and then its optimizer takes one step, updating G_{B} and D_{B}. Note that the gradients from the generator and discriminator are independently treated.

Algorithm 2 Meta-training the attribute-transfer GAN model.

```
Require: $n_{\text{iter}}$, number of iterations. $\alpha$'s, learning rate. $m$, batch size. $k$, number of iterations inner-loop. $n_{\text{gen}}$, number of skipped iterations of the generator per discriminator iteration.
Require: $\boldsymbol{\theta}_0$, initial generator and discriminator parameters.
Initialize $\boldsymbol{\theta}_{\text{B}} = \boldsymbol{\theta}_0$
for $i < n_{\text{iter}}$ do
    Sample a batch of images $\{x^{(z)}\}_{z=0}^{m}$
    Sample a task $\tau$
    Initialize $\boldsymbol{\theta}_{\text{A}} = \boldsymbol{\theta}_{\text{B}}$
    for $j < k$ do
        Train discriminator $D_{\text{A}}$                                {Here starts inner-loop}
        $\boldsymbol{\theta}_{\text{A,disc}} \leftarrow \boldsymbol{\theta}_{\text{A,disc}} + \alpha_{\text{disc}} \nabla_{\boldsymbol{\theta}_{\text{A}}} \mathcal{L}_{\text{disc}}(\boldsymbol{x}, \tau)$
        Train generator $G_{\text{A}}$
        if $mod(i, n_{\text{gen}}) = 0$ then
            $\boldsymbol{\theta}_{\text{A,gen}} \leftarrow \boldsymbol{\theta}_{\text{A,gen}} + \alpha_{\text{gen}} \nabla_{\boldsymbol{\theta}_{\text{A}}} \mathcal{L}_{\text{gen}}(\boldsymbol{x}, \tau)$
        end if
    end for
    Train discriminator $D_{\text{B}}$                                    {Here starts outer-loop}
    $\boldsymbol{\theta}_{\text{B,disc}} \leftarrow \boldsymbol{\theta}_{\text{B,disc}} + \epsilon(\boldsymbol{\theta}_{\text{A,disc}} - \boldsymbol{\theta}_{\text{B,disc}})$
    Train generator $G_{\text{B}}$
    $\boldsymbol{\theta}_{\text{B,gen}} \leftarrow \boldsymbol{\theta}_{\text{B,gen}} + \epsilon(\boldsymbol{\theta}_{\text{A,gen}} - \boldsymbol{\theta}_{\text{B,gen}})$
end for
```

Algorithm 3 Meta-testing the attribute-transfer GAN model.

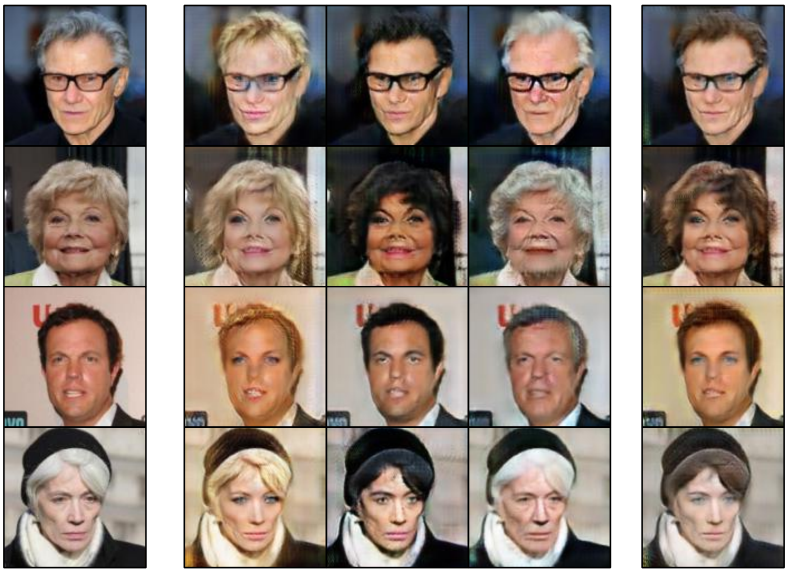
```
Require: $\alpha$'s, learning rate. $m$, number of sample. $k$, number of iterations inner-loop. $n_{\text{gen}}$, number of skipped iterations of the generator per discriminator iteration.
Require: $\boldsymbol{\theta}_{\text{B}}$, a copy of the final meta-trained parameters.
Initialize $\boldsymbol{\theta}_{\text{A}} = \boldsymbol{\theta}_{\text{B}}$
Sample of images $\{x^{(z)}\}_{z=0}^{m}$
New task $\tau$
for $i < k$ do
    Train discriminator $D_{\text{A}}$                {Here starts inner-loop}
    $\boldsymbol{\theta}_{\text{A,disc}} \leftarrow \boldsymbol{\theta}_{\text{A,disc}} + \alpha_{\text{disc}} \nabla_{\boldsymbol{\theta}_{\text{A}}} \mathcal{L}_{\text{disc}}(\boldsymbol{x}, \tau)$
    Train generator $G_{\text{A}}$
    if $mod(i, n_{\text{gen}}) = 0$ then
        $\boldsymbol{\theta}_{\text{A,gen}} \leftarrow \boldsymbol{\theta}_{\text{A,gen}} + \alpha_{\text{gen}} \nabla_{\boldsymbol{\theta}_{\text{A}}} \mathcal{L}_{\text{gen}}(\boldsymbol{x}, \tau)$
    end if
end for
```

2.4 Experiments

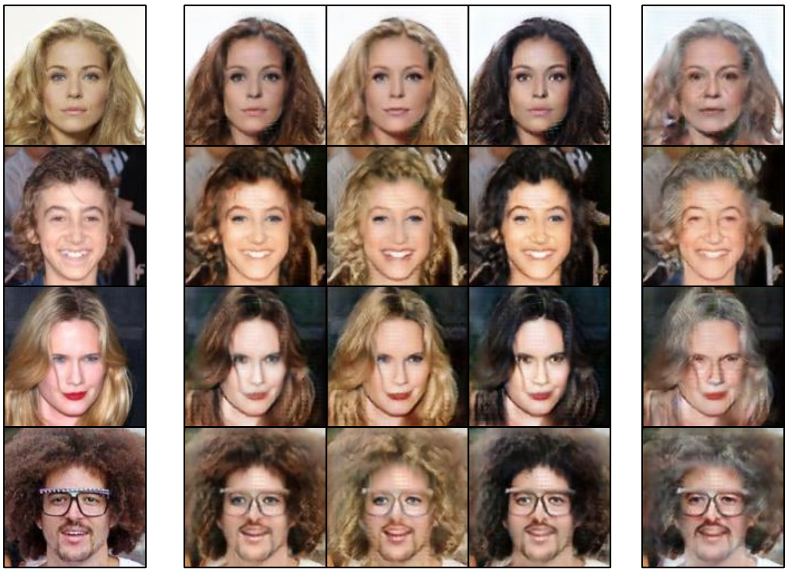
In this section, we present experimental results evaluating the effectiveness of the proposed method. We first give a detailed description of the experimental setup. Then, we present the results, and we discuss the possible interpretation. Finally, we run a hyperparameter search to estimate the limitations.

2.4.1 Experimental Setup

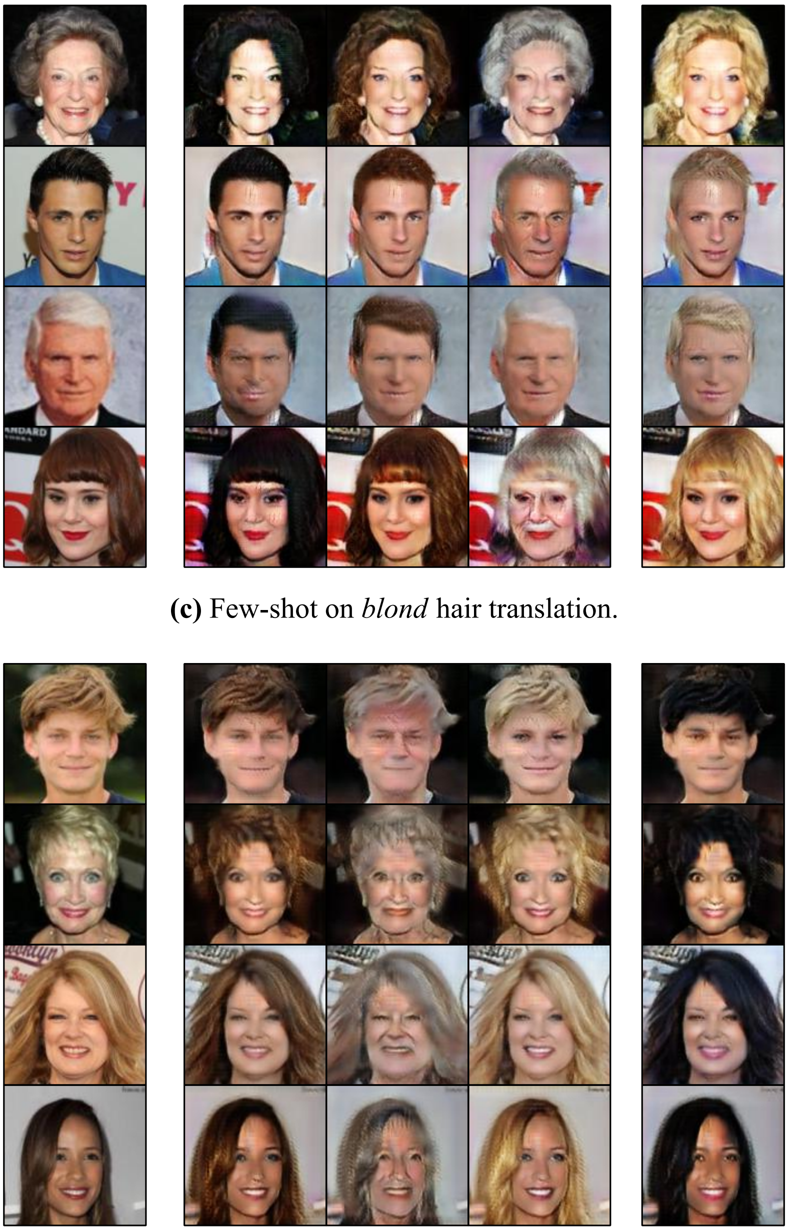
We train Semi Few-shotGAN on the CelebA [53] dataset, which consists of 202,599 images of celebrity faces with different facial attributes. In our experiments, we focus on hair attributes, in particular, hair colour: *blond*, *black*, *brown* and *grey* hair. As the dataset is heavily unbalanced, we generate a subset of CelebA with 32,500 images equally distributed between all the hair attributes. Notice that it is important to have a balanced distribution, otherwise the algorithm might fail when transferring to marginal attributes. We randomly select 2,000 images for testing, and use the remaining images as the training dataset. Finally, before starting training, we crop and resize the initially 178×218×3 pixel image to 128×128×3 pixels. All experiments presented are conducted on a single NVIDIA GeForce GTX 1080 GPU.



(a) Few-shot on *brown* hair translation.



(b) Few-shot on *grey* hair translation.



(c) Few-shot on *blond* hair translation.

(d) Few-shot on *black* hair translation.

Figure 2.5. Examples of results from few-shot hair-attribute transfer at testing time. Each subfigure is an independent experiment with a different attribute scenario, where the first column contains the input images, the following three columns shows the attribute translation outputs learnt during the meta-training. The last column displays the output results of the target attribute, which has been trained using only a few training samples at meta-testing time.

2.4.2 Training Setting

Our approach is divided into two generative models, GAN_A and GAN_B. Each of them uses an independent Adam [54] optimizer with β_1 = 0.5 and β_2 = 0.999. The batch size is set to 16, and the model is trained for 200,000 iterations. The generator is updated after every five discriminator updates, as suggested in [16, 51]. We set λ_class, λ_cycle and λ_gp to 10, and λ_adv to 1. Furthermore, we implement a learning rate scheduler, setting its initial value to 10^{-4}, which linearly decreases to zero over the last 100,000 iterations. The inner-loop for meta-learning is set to k = 1 iteration, and for the meta-testing k = 10 iterations.

2.4.3 Results of Attribute Transfer

We present a qualitative evaluation of the results of our proposal, including an empirical study for different attributes. To do so, we verify that the generated images consistently contain the target attribute, pre-defined by the task τ. As previously mentioned, we focus on hair-colour attributes: *blond*, *black*, *brown* and *grey*. Therefore, our attribute transfer involves translating from the original hair colour, to the target one. We conduct our experiments in a simulated scenario, where during meta-training 3 hair colours are used, and during meta-testing a few samples are used from the remaining colour. For example, Figure 2.5a shows the results of training the images using the tasks *blond*, *black* and *grey* during meta-training, and then applying few-shot learning, i.e., meta-testing, for the target task, in this case, *brown* hair. Independently, we repeat the same experiment for each of the hair-colour attributes; Figure 2.5 shows some testing examples.

By assessing the qualitative results, we can verify that through our meta-learning approach, the generative algorithm learns to find suitable initialization weights so that when an unseen and limited task appears, e.g., *brown* hair, with just a few samples and iterations, the algorithm is able to synthesize samples incorporating this new attribute. Additionally, the system keeps the capacity to transfer the attributes for which it is originally trained, e.g., *blond*, *black* and *grey* hair as well as the ability to produce natural-looking faces.

Finally, we notice empirically that the *grey* hair attribute incorporates more information than just the colour of the hair. In fact, we observe that this transformation often implies an ageing effect on the subject. The main reason for such an event is the inherent entanglement of some attributes. In the CelebA dataset, the attribute *grey* hair is almost always associated with older people. Therefore, when the model is to learn the attribute, it cannot untangle the *old* and *grey* hair attributes, unless we deliberately introduce a signal for that, e.g., an additional label for *old*.

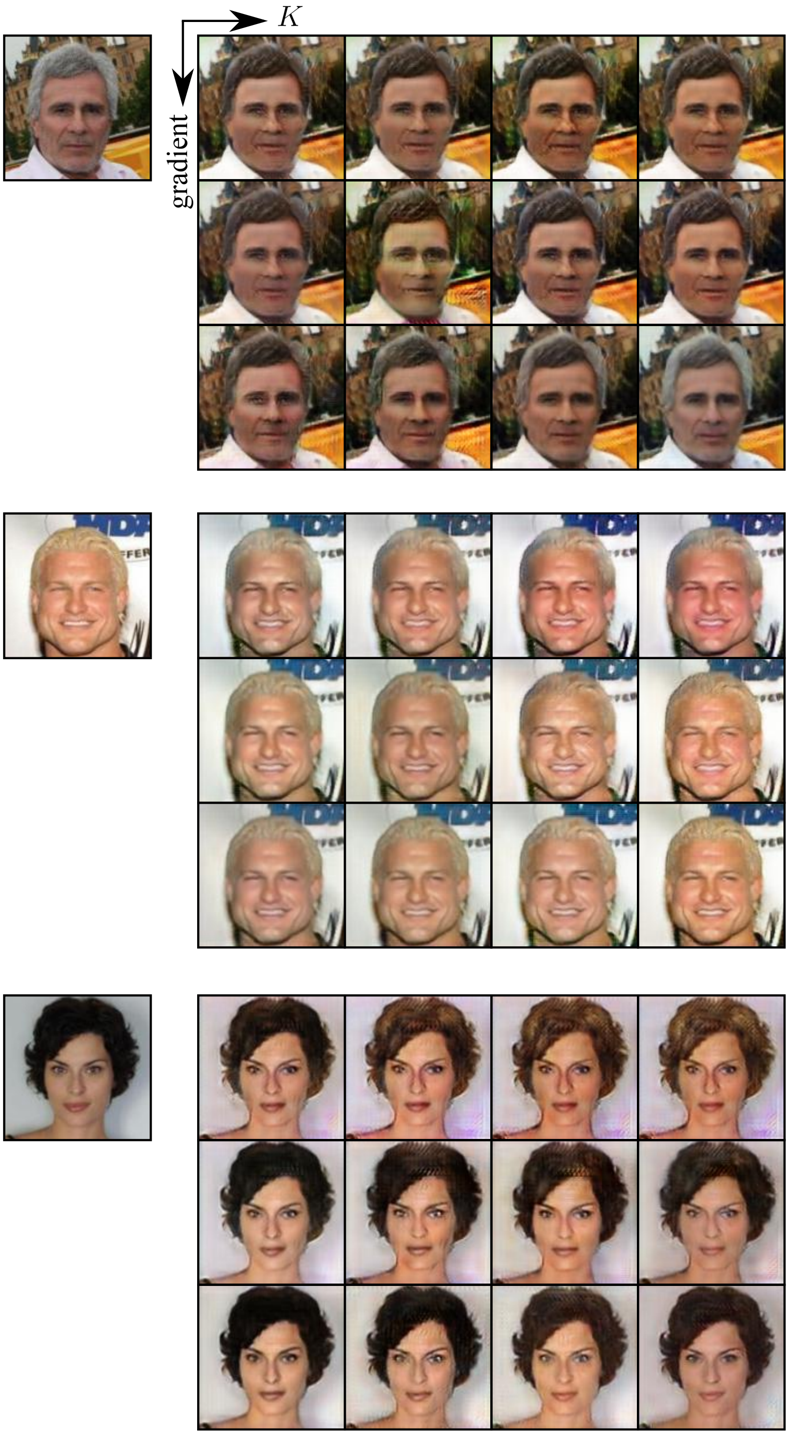


Figure 2.6. Examples of few-shot hair attribute transfer results for different hyperparameters at testing time. From top to bottom, our target attribute is *black*, *grey* and *brown* hair. Each subfigure is an independent experiment, where we run an evaluation on the grid search space from the hyperparameters. On the one hand, we modify the amount of samples of the unseen target class (different K-shots). Column-wise from left to right: We use 4, 8, 16 and 32 samples. Row-wise from top to bottom: We use 10, 100, and 1000 gradient updates. We observe best results when using 32 samples and 10 gradient step updates (upper-right corner).

2.4.4 Hyperparameter Study

In this subsection, we present additional experiments, which help to build a general idea of how sensitive our model is, and to what extent it is limited. Hence, we explore a grid search space, evaluating the impact of two hyperparameters: 1) the amount of samples of the unseen target attribute (different K-shots scenarios); and 2) the number of gradient steps at meta-testing time. More specifically, we employ 4, 8, 16 and 32 samples, and 10, 100 and 1000 for the gradients updates.

In Figure 2.6 we empirically observe the impact after applying the different configuration setups on three examples. Column-wise, the generated images are shown for the different K-shots, and row-wise, for gradient steps. As one might expect, the more samples are used during meta-testing optimization, the better the translation results. This behaviour emphasizes the high correlation between good results and a massive amount of data that we are used to in the deep-learning community. On the other hand, more surprisingly, the number of gradient updates shows, at first sight, a counter-intuitive effect. The more we optimize towards the new attribute task, the worse the results become. This phenomenon might be explained in terms of the loss surface during training, with respect to its generalization behaviour. In [55–57], it is speculated that the width of a loss optimum is critically related to its generalization properties. Consequently, training our optimizer in excess, i.e., updating the gradients too many times, might lead to them leaving the wide regions, i.e., significantly sharp optimums. In practice, this means that at testing time, the generative model yields worse results. In our experiments, we observe a decline in attribute-transfer capabilities, the more updates the optimizer makes.

2.5 Limitations

Although moving towards unsupervised or few-shot learning might be the solution for limited labelled data in the upcoming years, it is still unclear which will be the winner approach. While it is true that our method successfully generates samples from a very limited target domain, the meta-training domains are very similar—at least from a level of human understanding. Therefore, one important limitation might be the incapacity to generalize for a wider range of attribute domains, especially when those involve morphological changes, e.g., training the meta-training with attribute domains, such as *blond*, *moustache* or *eyeglasses*, and then at meta-testing time, targeting *skin* colour or *gender*. Said generalizability is the ultimate goal for any attribute-transfer model, in particular, for those with scarcity of annotated data.

2.6 Summary

In this chapter, we tackle few-shot learning for attribute transfer, a form of limited supervision for image-to-image translation tasks. By using this approach, our model learns to generate realistic images containing unseen attributes with just a few samples. While previous work trained with simple datasets, such as MNIST, we propose a new meta-learning system capable of learning and transferring attributes, given a more complex image dataset, i.e., CelebA. Furthermore, our proposal is conducted with no lengthy inference time, no external memory and no additional data. As for the amount of data, the hyperparameter study shows the potential from our model, where no more than 16 samples are needed to synthesize new attributes that coherently merge with the rest of the image. This feature may open the door to other applications that were previously restricted by the large amount of data required. We see many interesting avenues for future work, including combining different types of facial-attribute transformation, such as *hair* colour with *smiling*, *eyeglasses*, or applying it to other fields, such as medical or seismic.

Chapter 3

Inpainting for Attribute Transfer

Generative deep-learning modelling is a growing field, in which recent work has shown remarkable success in different domains. In particular, the computer-vision community has witnessed a dramatic improvement in a large variety of tasks, such as image synthesis [3, 8, 10], image inpainting [18, 19, 58, 59], and image-to-image translation [16, 60, 61]. The latter task poses the problem of translating images from one domain to another, including style transfer [9, 14, 15], and attribute transfer [16, 62, 63].

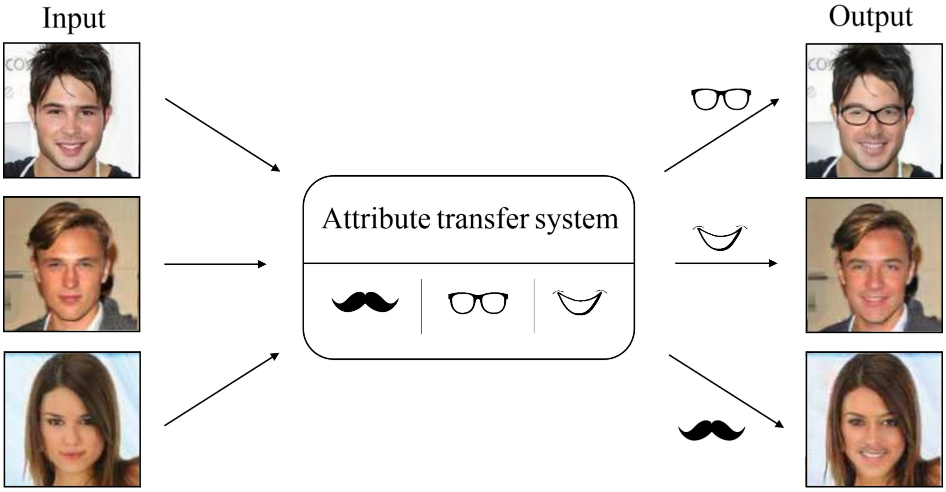


Figure 3.1. Schematic pipeline of an attribute-translation system. The input image undergoes some attribute transformations so that the output contains *smiling*, *moustache* or *eyeglasses* on the face.

The objective of an attribute-transfer system is to synthesize realistic and appealing images containing pre-defined target attributes. Figure 3.1 depicts a schematic pipeline of such a system. On the left side, we have the input images, which we feed into the model. Then, we select which attributes we want to modify, either adding or removing them. Finally, the model generates the output images with the target attributes. In this case, for example, if we look at the second input image, which is originally a *not-smiling* man, without a *moustache* and without *eyeglasses*, after the attribute-transfer operation, the man is *smiling* (his target), while keeping the other two attributes unmodified.

Attribute-transfer methods have achieved rapid progress with different implementations of GANs [16, 63], leading to state-of-the-art results in the field. Most of these approaches are based on the global manipulation of the latent space; these often end up with complex training procedures. For example, unlike invertible systems [64], it is fairly common for GAN-based models to require additional mapping paths between any two attributes, which tends to make the approaches less stable.

Image inpainting or reconstruction refers to the task of inferring locally missing or damaged parts of an image. Its main challenge is to synthesize realistic pixels that are coherent with existing ones. These techniques have been quite extensively applied to applications like photo editing [65], restoration of damaged paintings [66], and image-based rendering [67]. Figure 3.2 shows a simplified pipeline of an image inpainting system.

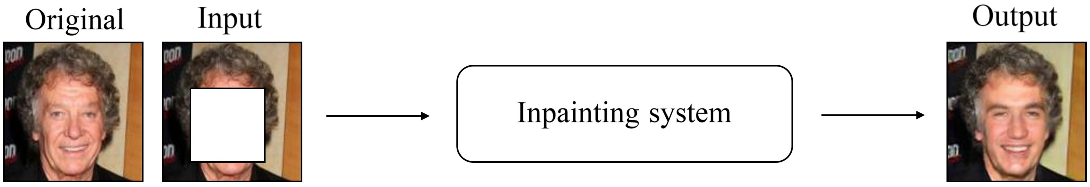


Figure 3.2. Schematic pipeline of an inpainting system. Given a source image, we crop a square patch from the centre of it to generate the input image. The output image must have the missing area inpainted, taking the rest of the image into consideration.

Image inpainting techniques are mostly classified according to their underlying approach. On the one hand, we have local methods, based on low-level feature information, such as colour or texture. On the other hand, we have methods that rely on recognizing global patterns in images, such as small shapes or objects, to predict pixels from the missing regions. This second type is often implemented with deep CNNs. In recent years, CNN-based models have been combined with generative models, resulting in approaches that can deal with both local and global features while inpainting. In particular, the introduction of GANs has been very fruitful, inspiring numerous recent projects [17, 18, 58, 60]. Most of this work has formulated the inpainting task as a conditional image generation problem, exploiting the generator network for inpainting, and the discriminator network for validating the results.

In this chapter, we present a novel image-to-image method for the common sub-task of local attribute transfers. In particular, we introduce a model that targets specific face attributes and performs semantic alterations, e.g., removing a *moustache*. In contrast to previous methods, where local changes were conducted by generating new global images, we propose formulating local attribute transfers as an inpainting problem. By removing and then regenerating specific parts of the images, “Attribute Transfer Inpainting Generative Adversarial Network”

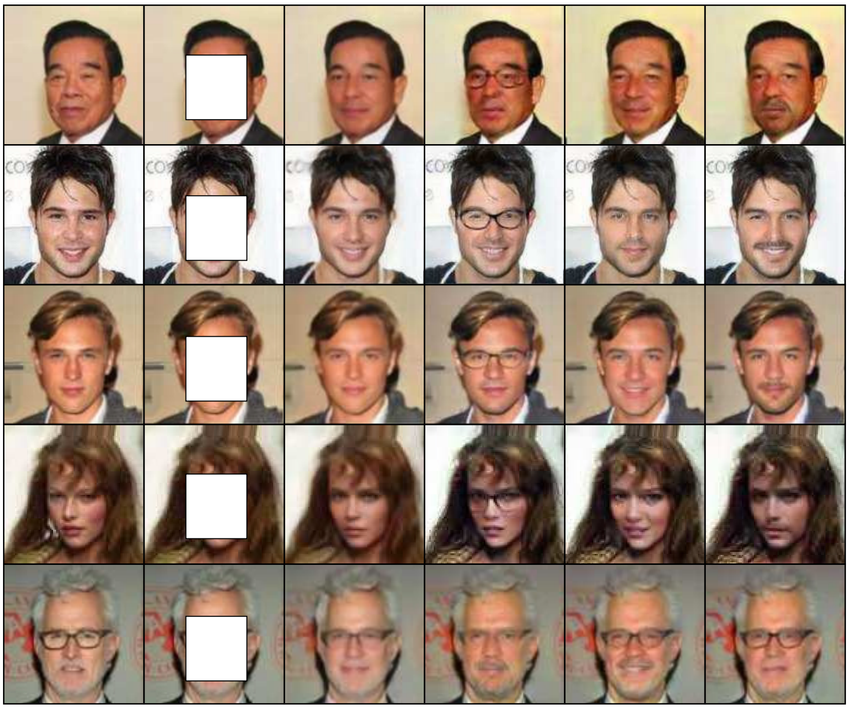


Figure 3.3. ATI-GAN image-to-image attribute-translation results. The first column shows the source images, the second the input masked images, the third the inpainted results and the remaining columns are the attribute-transfer results (*eyeglasses*, *smiling* and *moustache*). Note that the inpainted results are an intermediate step in the translation system, and the attribute results contain or exclude the target attribute with respect to the source images.

(ATI-GAN) is able to exploit local contextual information, while keeping the background unmodified, resulting in visually sound results. Figure 3.3 displays a few samples of different attribute translations from ATI-GAN.

3.1 Background

In this section, we formally define the image inpainting problem, and provide an overview of the related work, paying special attention to GAN-based approaches.

3.1.1 Preliminaries

Assuming $\bar{\boldsymbol{x}}$ are the observed images that are missing some parts of their content, and $\boldsymbol{x}$ are the source (original) uncorrupted images, we can formulate the image corruption process as

$$\bar{\boldsymbol{x}} = \eta(\boldsymbol{x}), \tag{3.1}$$

where η is an arbitrary stochastic damaging process that corrupts the source images. Given this scenario, the inpainting task's learning objective can be formulated as

$$\min_{f} \mathbb{E}_{\boldsymbol{x}} || f(\bar{\boldsymbol{x}}) - \boldsymbol{x} ||_2^2. \tag{3.2}$$

From this formulation, we can see that the goal is to find a function f that best approximates η^{-1}. Therefore, we can treat the image-inpainting problems in a unified framework, and choose an appropriate η for each case. In a deep-learning context, the η function is modelled by neural networks, where GAN-based approaches have gained popularity.

3.1.2 Related Work

Classic inpainting methods are often based on either local or non-local information algorithms to reconstruct the missing pieces of the images, known as holes or patches. Local methods [68,69] attempt to solve the problem using only contextual information, such as colour or texture. In other words, they match, copy and merge background areas into the holes by propagating the information from the holes' boundaries. These approaches need very little training or prior knowledge to provide accurate results, especially in backgrounds and regular structures. However, these inpainting methods usually fail when dealing with large patches due to their inability to generate novel image content. More powerful methods are global, also known as non-local approaches. Even though these techniques require more expensive patch computations, they can successfully handle larger and denser patches. In many of these methods, CNNs have become the *de facto* core implementation. The reason for that is their capability to learn to recognize patterns, features and irregular structures in images, and then to employ them to fill holes or corrupted areas. Among the large collection of CNN frameworks, GANs have emerged as a promising paradigm for inpainting tasks [17,18,58,60]. They have even shown significant potential to produce inpainted results on high image resolution [19,59]. Nevertheless, to reach this peak performance, GANs have gone through a constant evolution over the past few years. For instance, the inpainting task was initially formulated as a simple conditional image generation problem, with one generator and one discriminator [17,58]. However, Iizuka *et al.* [60] and Yu *et al.* [19] introduced the concept of global discriminators (for coarse structures) and local ones (for fine-grained structures), as a substitute for the one-discriminator baseline topology.

GAN-based architectures have also been employed on other computer-vision tasks. Recently, researchers have focused their attention on transferring visual attributes, such as colour [25], texture [14, 15] or facial features [16, 62, 63]. Although most of these approaches successfully synthesize images belonging to the target domain, developing models that generalize between completely different problems is still an open question. Pix2pix [14] was one of the first proposed models to consistently learn image-domain transformations with some generalizability. They accomplished this task by employing pairs of images. The model could convert samples from their original domain to the target domain, e.g., segmentation labels to the source (original) image. Unfortunately, this system required both source and target images to exist as pairs in the training dataset. Posterior work, CycleGAN [15] and StarGAN [16], addressed this drawback by introducing alternatives that did not need a paired structure, e.g., via a virtual result in the target domain. In this way, if it is inverted again, the output must match the source image. IcGAN [70] combined a conditional GAN with an encoder, which allowed multiple attributes to be manipulated at once. VAE/GAN [71] joined a variational autoencoder with a GAN method to learn embeddings in which high-level abstract visual features, e.g., wearing *eyeglasses*, could be modified using simple arithmetic. Furthermore, alternative loss terms haven been adopted, reporting better stability, efficiency and efficacy properties. WGAN [50] was the first GAN to use Earth-Mover distance loss, also known as Wasserstein loss, outperforming the vanilla GAN [3]. This new loss was quickly and successfully employed on image-to-image translation [14, 15, 72]. Not long after that, Gulrajani *et al.* [51] showed that there were improvements in image generation when a gradient penalty was added to the Wasserstein loss, to restrict the discriminative function 1-Lipschitz. They named the approach WGAN-GP. As a result, a second wave of publications [16, 19], proposed to use the latest adversarial loss.

3.2 Contributions

In this work, we introduce a novel attribute-transfer approach that modifies source images in such a way that the output images incorporate the pre-defined target attributes. To accomplish this task, the proposed GAN-based architecture integrates an inpainting block. This block allows all attention to be focused on the region of interest—where the attributes are located—while keeping the background unmodified yet consistent with the modifications. In particular, we take advantage of the fact that most facial attributes are induced by local structures, e.g., relative position between eyes and ears. Hence, it is sufficient to change only a few parts of the face, and to force the generator to synthesize them into realistic outputs.

The ATI-GAN model achieves local attribute transfer in a single end-to-end architecture with three main building blocks. First, we have an inpainting network that takes masked images as the input, and outputs fully restored images. Second, we have a network that takes these inpainted images as the input, including their encoded attributes in a one-hot vector. Then, it generates the final images combining both pieces of information. Therefore, the network learns how to separate attribute information from the rest of the image representation. Finally, a third network acts as a discriminator; its goal is to judge if the output images look realistic, and if they comply with their target attributes. We apply our model on the CelebA dataset, and demonstrate the capacity of the system to produce high-quality outputs. Quantitative and qualitative results show superior inpainting and attribute-transfer performance compared with similar work. Overall, our contributions are summarized as follows:

- We propose ATI-GAN, a novel generative adversarial network for local attribute transfer.
- We demonstrate that our model can successfully learn to fill holes while learning attribute translation through a collaborative embedded system.
- We introduce an image-to-image translation system that benefits from a dual cooperation, resulting in an overall improvement. Such cooperation opens the door to further combinations within the field.
- We provide both quantitative and qualitative results for the intermediate task of inpainting, as well as for the attribute-transfer task. We assess the effectiveness of the approach on the CelebA dataset.

3.3 Method

We start this section by introducing the problem definition. Then, we provide a detailed presentation of the architecture of our model, which is split into three parts. After that, we analyse the role of each of these blocks and their corresponding objective optimizations via their loss formulation. Eventually, we close this section with the description of how our model trains in an end-to-end fashion, embedding different tasks, i.e., inpainting and attribute transfer, to produce realistic and controllable sample results.

3.3.1 Problem Definition

Our proposed model addresses the attribute-translation task in an inpainting context. Given a source image $x \in \mathbb{R}^{H \times W \times 3}$, its masked pair $\bar{x} \in \mathbb{R}^{H \times W \times 3}$, and its attribute-domain label $c \in \mathbb{R}^{1 \times N}$, our goal is to train a model that can exploit the inpainting paradigm to control facial-attribute transfer. Note that H and W are the height and width of the data, respectively, and N is the category number of the attribute-domain label.

3.3.2 Model Architecture

The pipeline of our proposal is depicted in Figure 3.4. It is composed of a reconstructive network, a generative network and a discriminative network. By combining each of these blocks sequentially, the ensemble model successfully learns to transfer attributes.

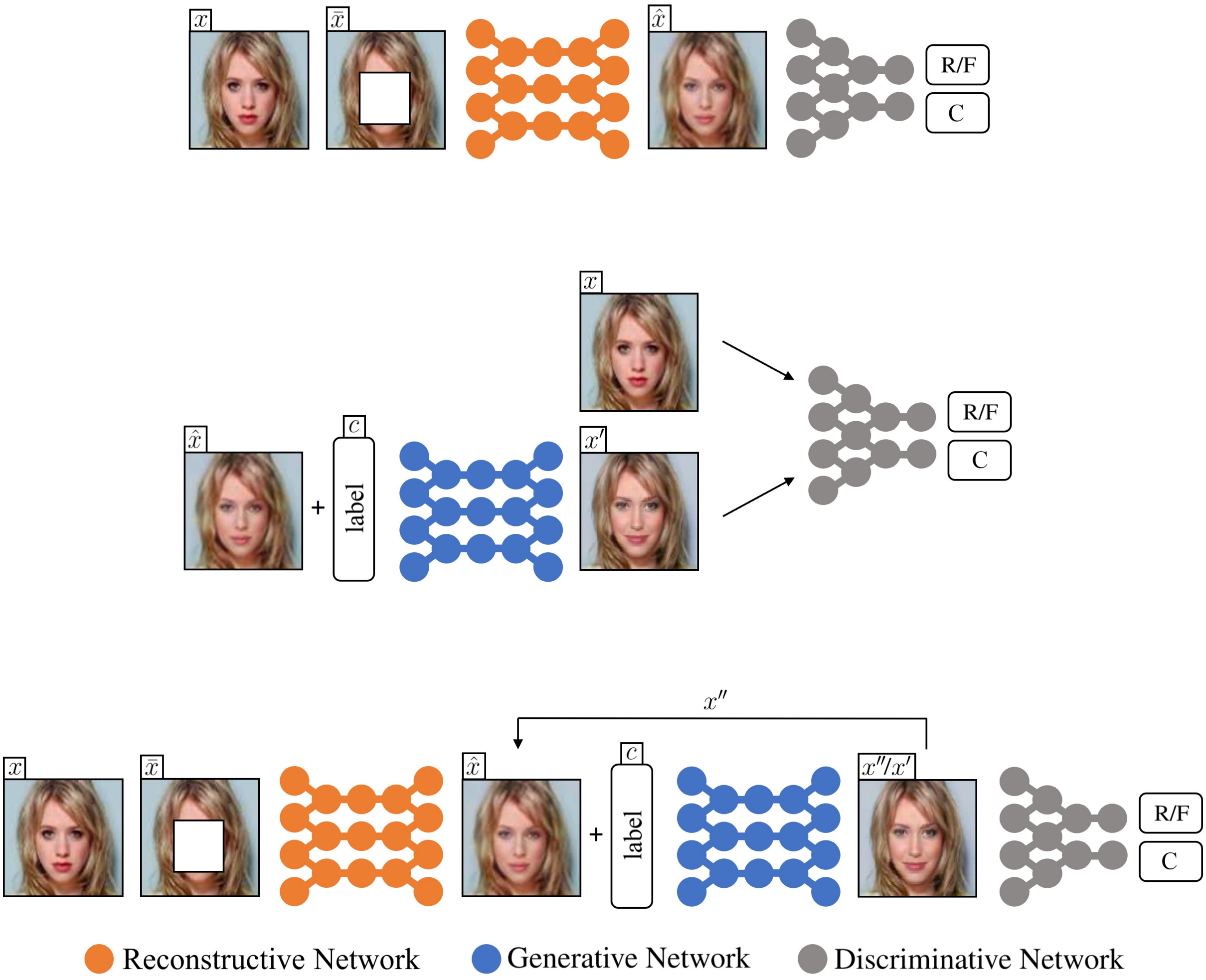


Figure 3.4. Pipeline of the ATI-GAN framework at training. Each network has its own separate training process. (Top) The reconstructor R takes the masked image $\bar{x}$ and inpaints a realistic filling $\hat{x}$. Then, the discriminator D judges the outcome. (Centre) D takes the real image x (R) and the fake image x' (F), and learns to distinguish them. Furthermore, it also classifies them according to their target domain (C). (Bottom) The generator G takes the output image $\hat{x}$ and its domain label c, and synthesizes the attribute-transfer image x'. Next, the synthesized image x' and the original domain label are fed into G to generate x''—creating a loop to learn attribute-inversion paths. After that, both x' and x'' are passed to D to be evaluated.

Reconstructive Network. Given a set of source (real) images $\boldsymbol{x}$ and their paired masked images $\bar{\boldsymbol{x}}$, the goal of the reconstructor R is to fill the large missing region of $\bar{\boldsymbol{x}}$, known as a patch or hole, with plausible content so that the inpainted results $\dot{\boldsymbol{x}}$ looks realistic. To achieve appealing inpaintings, the reconstructor R relies on different loss terms. On the one hand, we have an autoencoder loss $\mathcal{L}_{\text{ae}}$ which essentially constrains the reconstructed images by minimizing the absolute differences between these images and their source at both contour and patch level. This optimization task is defined as

$$\mathcal{L}_{\text{ae}} = ||\boldsymbol{x}_{\text{c}} - \dot{\boldsymbol{x}}_{\text{c}}||_1 + \lambda_{\text{p}}\,||\boldsymbol{x}_{\text{p}} - \dot{\boldsymbol{x}}_{\text{p}}||_1, \tag{3.3}$$

where the subindexes c and p refer to contour and patch, respectively. Figure 3.5 illustrates the different components that intervene.

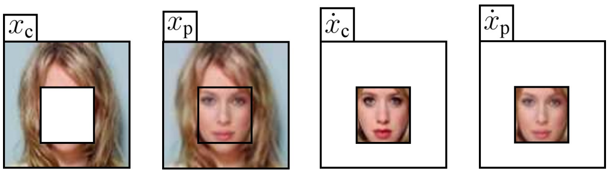


Figure 3.5. Components of $\mathcal{L}_{\text{ae}}$: contour from the source image x_{c}, contour from the reconstructed image $\dot{x}_{\text{c}}$, patch from the source image x_{p}, and patch from the reconstructed image $\dot{x}_{\text{p}}$.

We apply separate l_1 distance norms: one for the contour and one for the patch. The reason for this is that the patch loss does not have to be strictly 0 since it is possible to generate valid synthetic patch images, i.e., realistic and consistent with the contour, which are not exactly the same as the source images. Consequently, they will not reach a 0 loss value. In contrast, the contour loss has to get as close as possible to 0. Here we are not interested in alternative reconstructions, but the source one.

On the other hand, we have $\mathcal{L}_{\text{adv,rec}}$ (see Equation 3.7 and 3.8), which penalizes unrealistic images, and $\mathcal{L}_{\text{class,rec}}$ (see Equation 3.11), which takes care of incorrect image-domain transformations. These two loss terms are assessed by the discriminator D. The whole inpainting training process is therefore formulated as a minimization problem of the following terms

$$\mathcal{L}_{\text{rec}} = \lambda_{\text{ae}}\mathcal{L}_{\text{ae}} + \lambda_{\text{adv}}\mathcal{L}_{\text{adv,rec}} + \lambda_{\text{class}}\,\mathcal{L}_{\text{class,rec}}. \tag{3.4}$$

Once the inpainting task is complete, the reconstructed result $\dot{x}$ undergoes a contour swapping process, creating $\hat{x}$. We refer to it as a “modification” step; a visual example is displayed in Figure 3.6. The reason for this manipulation is that we only want to modify the attributes, i.e., the inpainted patch. Thus, the contour can be replaced by the original (source) one. While it is true that it is useful for the reconstructor to enhance its understanding by learning to reconstruct

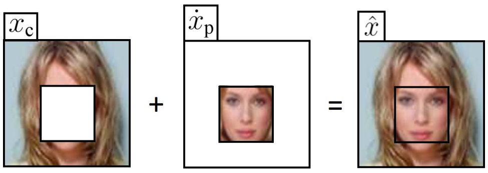


Figure 3.6. The "modification" step takes place in between the reconstructor and the generator. It consists of combining the contour from the source image x_c with the patch from the reconstructed image $\dot{x}_p$, which produces $\hat{x}$.

the contour during the inpainting process, the result has no further use in the rest of the system. Consequently, it can be dismissed.

Finally, in terms of topology implementation, we adopt the coarse network architecture introduced by Yu *et al.* [19]. As the size of the receptive fields are a decisive factor in inpainting tasks, we use dilated convolutions to guarantee a sufficiently large size. Additionally, we apply mirror padding for all convolution layers and exponential linear units [73] as activation functions. Figure 3.7 depicts the scheme of the pipeline of the reconstructive model, together with the "modification" step.

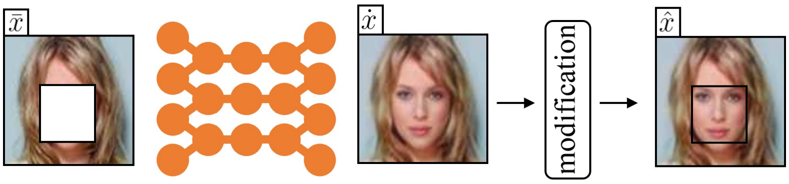


Figure 3.7. Pipeline of the reconstructor structure. Given a masked input $\bar{x}$, the network inpaints the mask, producing $\dot{x}$. The "modification" step is also depicted, where $\dot{x}$ becomes $\hat{x}$ after the contour swap.

Generative Network. The role of the generator G is to learn the mappings between multiple attribute domains. To achieve this goal, we use the modified outputs from the reconstructor $\hat{\boldsymbol{x}}$ as the inputs. Then, we train G, to translate $\hat{\boldsymbol{x}}$ into output images $\boldsymbol{x}'$ that belong to the target domains. We control the domains by conditioning the $\hat{\boldsymbol{x}}$ via domain labels $\boldsymbol{c}_{\text{target}}$. The generator relies on several loss terms to fulfil the translation task. In the same manner as in the inpainting block, G also employs an adversarial loss $\mathcal{L}_{\text{adv,gen}}$ (see Equation 3.7 and 3.9) and a classification loss $\mathcal{L}_{\text{class,gen}}$ (see Equation 3.12). Moreover, we have an extra term called cycle consistency loss $\mathcal{L}_{\text{cycle}}$ (see Equation 3.5), which helps create strong inversion paths between latent space and target domains [15, 16, 62]. In our case, such paths guarantee that the translated images $\boldsymbol{x}'$ preserve the content of their input images $\hat{\boldsymbol{x}}$, while changing only the domain-related parts through the latent space; namely, it creates attribute inversion paths. This cyclic term is defined as

$$\begin{aligned} \mathcal{L}_{\text{cycle}} &= ||\hat{\boldsymbol{x}} - \boldsymbol{x}''||_1 \\ \text{with } \ \boldsymbol{x}'' &= G(\boldsymbol{x}', \boldsymbol{c}_{\text{original}}), \\ \text{and with } \ \boldsymbol{x}' &= G(\hat{\boldsymbol{x}}, \boldsymbol{c}_{\text{target}}). \end{aligned} \tag{3.5}$$

Note that the generator performs the entire cyclic translation $\hat{\boldsymbol{x}} \rightarrow \boldsymbol{x}' \rightarrow \boldsymbol{x}''$, forcing the domain labels to be crucial for moving between domains. First, the source images $\hat{\boldsymbol{x}}$ are translated to the target domains $\boldsymbol{x}'$ (conditioned on $\boldsymbol{c}_{\text{target}}$) and then, $\boldsymbol{x}'$ are translated back to their original domain $\boldsymbol{x}''$ (conditioned on $\boldsymbol{c}_{\text{original}}$).

Finally, arranging all the generator terms together leads to the following objective loss

$$\mathcal{L}_{\text{gen}} = \lambda_{\text{adv}}\mathcal{L}_{\text{adv,gen}} + \lambda_{\text{class}}\,\mathcal{L}_{\text{class,gen}} + \lambda_{\text{cycle}}\,\mathcal{L}_{\text{cycle}}. \tag{3.6}$$

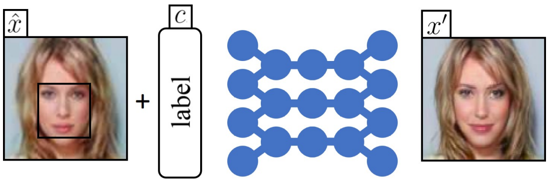


Figure 3.8. Pipeline of the generator structure. Given an input $\hat{x}$ and the domain label c, the network synthesizes an output x', containing the target attribute.

Regarding the topology, we build our attribute-transfer generator, based on the recent state-of-the-art image-to-image translation model StarGAN [16], which in turn was an adaptation of CycleGAN [15]. On top of this baseline, we incorporate dilated convolutional layers, to increase the reception field, which empirically has proven to benefit our system. We use instance normalization in this network. Figure 3.8 depicts the scheme of the pipeline of the generator model.

Discriminative Network. Our discriminator D behaves slightly differently from the vanilla GAN [3]. As one might expect, it classifies samples as real or fake, according to their appearance. However, it has an additional in-built classifier that tries to determine the domain from each sample, as in AC-GAN [74]. As a result, the discriminator needs to have one adversarial loss $\mathcal{L}_{\text{adv}}$, judging the appearance of the images, and one classification loss $\mathcal{L}_{\text{class}}$, classifying the attributes. Concerning the inner structure of our discriminator D, it also differs from standard discriminators. In particular, D is split into two fully convolutional topologies: the global discriminator D_{g} and the patch discriminator D_{p}, along with one final convolutional block to combine both discriminators' outputs to determine the domain (see Figure 3.9). The idea behind this double discriminator structure is that while D_{g} deals with the global semantic consistency of the whole image, D_{p} deals with the local semantic consistency of the generated patch. In

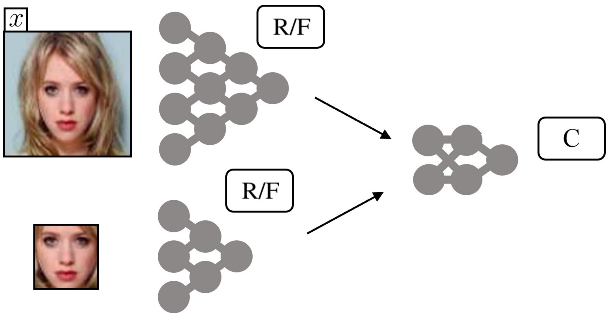


Figure 3.9. Pipeline of the inner structure of the discriminator. It has two main blocks: 1) the global discriminator D_{g}, which assesses the consistency of the entire image via its global loss $\mathcal{L}_{\text{g}}$, and 2) the patch discriminator D_{p}, which assesses the consistency of the inpainting via its patch loss $\mathcal{L}_{\text{p}}$. Finally, there is also the class error $\mathcal{L}_{\text{class}}$, which uses the whole architecture of D to determine the class of the label.

order to guarantee both consistencies, every image is evaluated simultaneously by the discriminator on two independent loss functions: a patch loss $\mathcal{L}_{\text{p}}$ and a global loss $\mathcal{L}_{\text{g}}$. Together, they form the adversarial loss formulated as

$$\mathcal{L}_{\text{adv}} = \mathcal{L}_{\text{g}} + \mathcal{L}_{\text{p}}. \tag{3.7}$$

Following recent work from Choi *et al.* [16] and Yu [19], we employ Wasserstein loss with gradient penalty as our adversarial loss $\mathcal{L}_{\text{adv}}$. Thus, we can write the global loss $\mathcal{L}_{\text{g}}$ for the reconstructor as

$$\mathcal{L}_{\text{g,rec}} = -\mathbb{E}_{\hat{\boldsymbol{x}}}[D_{\text{g}}(\hat{\boldsymbol{x}}))], \tag{3.8}$$

for the generator as

$$\mathcal{L}_{\text{g,gen}} = -\mathbb{E}_{\boldsymbol{x}'}[D_{\text{g}}(\boldsymbol{x}'))], \tag{3.9}$$

and for the discriminator as

$$\mathcal{L}_{\text{g,disc}} = -\mathbb{E}_{\boldsymbol{x}}[D_{\text{g}}(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{x}'}[D_{\text{g}}(\boldsymbol{x}')) + \lambda_{\text{gp}}\,\mathbb{E}_{\boldsymbol{x}'}[(||\nabla D_{\text{g}}(\alpha\boldsymbol{x} + (1-\alpha)\boldsymbol{x}')||_2 - 1)^2], \tag{3.10}$$

where α is a random variable following the discrete uniform distribution over the set $\{0, 1\}$. Regarding the patch loss $\mathcal{L}_{\text{p}}$, notice that it can be calculated after replacing D_{g} with D_{p}.

As we mentioned above, our discriminator D has a final convolutional block to compute the $\mathcal{L}_{\text{class}}$, which accounts for domain classification. In particular, we implement a binary cross-entropy function between the outputs from our domain classifier and the domain labels, either $\boldsymbol{c}_{\text{target}}$ or $\boldsymbol{c}_{\text{original}}$. Thus, we can write the classification loss for the reconstructor as

$$\mathcal{L}_{\text{class,rec}} = \mathbb{E}_{\hat{\boldsymbol{x}}}[-\log D(\boldsymbol{c}_{\text{original}}|\hat{\boldsymbol{x}})], \tag{3.11}$$

for the generator as

$$\mathcal{L}_{\text{class,gen}} = \mathbb{E}_{\boldsymbol{x}'}[-\log D(\boldsymbol{c}_{\text{target}}|\boldsymbol{x}')], \tag{3.12}$$

and for the discriminator as

$$\mathcal{L}_{\text{class,disc}} = \mathbb{E}_{\boldsymbol{x}}[-\log D(\boldsymbol{c}_{\text{original}}|\boldsymbol{x})]. \tag{3.13}$$

Finally, the optimization problem of the training process of the discriminator is described as

$$\mathcal{L}_{\text{disc}} = \lambda_{\text{adv}}\mathcal{L}_{\text{adv,disc}} + \lambda_{\text{class}}\,\mathcal{L}_{\text{class,disc}}. \tag{3.14}$$

3.3.3 End-to-End Model Architecture

In order to successfully merge the three components within the same framework, it is important to find the right balance between their interactions and to weigh their contributions, in particular, that of the discriminator, since it plays a fundamental role in our image-to-image system. On the one hand, the discriminator indirectly forces the generator to produce correct image transformations by penalizing those outputs that do not belong to the target domain, and/or look artificial. On the other hand, the discriminator also contributes to enhancing the inpainting results, by teaching the reconstructor to follow the source images. Algorithm 4 describes the pseudocode of our proposal.

3.4 Experiments

In this section, we present results for a series of experiments evaluating the proposed method quantitatively and qualitatively. We first give a detailed description of the experimental setup and the evaluation metrics. Then, we discuss the inpainting outcomes. Finally, we assess the attribute-transfer capability of our proposal.

3.4.1 Experimental Setup

We train ATI-GAN on the CelebA [53] dataset, which consists of 202,599 images of celebrity faces with different facial attributes. We randomly select 2,000 images for testing, and use all the remaining images as the training dataset. Before training, we crop and resize the initially $178\times218\times3$ pixel images to $128\times128\times3$ pixels, and mask them with $52\times52\times3$ size patches. These masked regions are centred around the tip of the nose, which in most cases, obstructs a large part of the face. All experiments presented are conducted on a single NVIDIA GeForce GTX 1080 GPU.

3.4.2 Training Setting

Our approach is divided into three blocks: the reconstructor (for inpainting), the generator and the discriminator. Each of them uses an independent Adam [54] optimizer with β_1 = 0.5 and β_2

Algorithm 4 Training of the ATI-GAN model.

```
Require: $n_{\text{iter}}$, number of iterations. $\alpha$'s, learning rates. $m$, batch size. $n_{\text{gen}}$, number of skipped iterations of the generator per discriminator iteration. $n_{\text{rec}}$, number of iterations before $G$ starts to train with $\hat{\boldsymbol{x}}$.
Require: $\boldsymbol{\theta}$, initial reconstructor, generator and discriminator parameters.
Note: $\hat{\boldsymbol{x}}$ are the modified outputs of the reconstructor.
for $i < n_{\text{iter}}$ do
    Sample a batch of images $\{x^{(z)}\}_{z=0}^{m}$
    Mask the batch of images $\{\bar{x}^{(z)}\}_{z=0}^{m}$
    Sample domain labels $\boldsymbol{c}$
    Train discriminator $D$
    $\boldsymbol{\theta}_{\text{disc}} \leftarrow \boldsymbol{\theta}_{\text{disc}} - \alpha_{\text{disc}} \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\text{disc}}(\boldsymbol{x}, \boldsymbol{c})$
    Train reconstructor $R$
    $\boldsymbol{\theta}_{\text{rec}} \leftarrow \boldsymbol{\theta}_{\text{rec}} - \alpha_{\text{rec}} \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\text{rec}}(\boldsymbol{x}, \bar{\boldsymbol{x}}, \boldsymbol{c})$
    Train generator $G$
    if $mod(i, n_{\text{gen}}) = 0$ then
        if $i < n_{\text{rec}}$ then
            $\boldsymbol{\theta}_{\text{gen}} \leftarrow \boldsymbol{\theta}_{\text{gen}} - \alpha_{\text{gen}} \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\text{gen}}(\boldsymbol{x}, \boldsymbol{c})$
        else
            $\boldsymbol{\theta}_{\text{gen}} \leftarrow \boldsymbol{\theta}_{\text{gen}} - \alpha_{\text{gen}} \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\text{gen}}(\hat{\boldsymbol{x}}, \boldsymbol{c})$
        end if
    end if
end for
```

= 0.999. The batch size is set to 16, and the model is trained for 200,000 iterations. We employ the output of the reconstructor as the input for the generator after 50,000 iterations. This sort of delay can be seen as a warm-up to ensure that the reconstructor provides informative inputs for the generator. The generator is updated after every five discriminator updates, as suggested in [16, 51]. We set λ_{ae}, λ_{cycle} and λ_{gp} to 10, λ_{class} and λ_{adv} to 1, and λ_{gp} to 5. Furthermore, we implement a learning rate scheduler, setting its initial value to 10^{-4}, linearly decreasing to zero over the last 100,000 iterations.

3.4.3 Evaluation Metrics

Reconstruction-level Evaluation. The term peak signal-to-noise ratio (PSNR) is an expression for the ratio, in decibels, between two images. It is used as a quality measurement between the maximum possible value (power) of a source image and the power of distorting noise. Its mathematical representation is

$$\text{PSNR} = 20 \log \frac{\text{MAX}_x}{\text{MSE}_{x,y}}, \tag{3.15}$$

where MAX_x is the maximum possible pixel value of the source image x, and $\text{MSE}_{x,y}$ stands for mean square error between x and its modified output image y.

Inpainting approaches often use PSNR as a measurement of quality. The higher the value of this metric, the better the quality of the output image. However, PSNR might oversimplify the comparison since it directly measures the difference in pixel values, estimating absolute errors. Therefore, this metric is usually combined with a second fidelity measure called the structural similarity index [75] (SSIM). Its formula is defined as

$$\text{SSIM} = \frac{(2\mu_x\mu_y + \text{C}_1)(2\sigma_{xy} + \text{C}_2)}{(\mu_x^2 + \mu_y^2 + \text{C}_1)(\sigma_x^2 + \sigma_y^2 + \text{C}_2)}, \tag{3.16}$$

where μ's represent the means, σ's the standard deviations, and C's the constants to ensure stability. Last but not least, the term σ_{xy} is calculated from

$$\sigma_{xy} = \frac{1}{\text{N} - 1} \sum_{i=1}^{\text{N}} (x_i - \mu_x)(y_i - \mu_y), \tag{3.17}$$

where N is the amount of pixels.

The SSIM is a perceptual metric that quantifies image quality degradation as the perceived change in structural information (strong interdependencies). It requires two images from the same image capture—a reference image and a manipulated image. The range of values for the SSIM is [-1, 1]. A value of 1 indicates that the two images are the same or very similar. A value of -1 indicates the opposite. Frequently, these values are adjusted to be in the range [0, 1], while holding the same meaning. In this work, we employ the latter range.

Attribute-level Evaluation. For a qualitative evaluation, in order to assess the attribute-transfer capacity of our model, we perform a user study in a survey format, where users have to label an image with a 1 when the attribute is recognized, and a 0 otherwise. For each target attribute and its antonymous counterpart, e.g., *eyeglasses* and *not-eyeglasses*, we conduct a separate test that randomly evaluates 10% of the testing data, and we calculate the average accuracy.

3.4.4 Results of Image Inpainting

The image inpainting problem has a number of different scenarios. The context of our interest is human faces. Given a face, where a significant part of the pixels are masked, our objective is to restore them so that the final outcome is a plausible and realistic human face. In order to achieve an appealing inpainting result, the synthesized face must fit into the mask/hole, taking into account both the quality of the reconstructed area, and the adaptation within the unmasked regions of the input image.

As mentioned by Yeh *et al.* [58] and Yu *et al.* [19], the perfect numerical metric for semantic inpainting does not exist. The reason for that is the existence of an infinite amount of valid solutions. In other words, image inpainting algorithms do not aim to reconstruct the ground-truth image, but to fill the masked area with content that is in line with the image. Hence, the ground-truth solution is only one out of many possibilities.

As for our inpainting task, we crop square patches from the centre of the images. This procedure leads to a standard scenario for inpainting on the CelebA dataset, since most relevant information is located in the centre. Table 3.1 shows the comparison of PSNR and SSIM metrics, where the other inpainting solutions have also followed the same cropping procedure. We can observe how our proposal achieves very competitive results, obtaining substantially higher values of PSNR. Such an improvement mainly comes from a good balance between the inpainting block and the discriminator block. While the reconstructor learns to produce the coarse features from faces (natural-looking structures) based on the autoencoder loss, the discriminator provides signals, in the form of gradients, to drive the results to be smooth with fine-grained details (detailed structures). Figure 3.10 illustrates inpainting results for different types of facial features such *gender*, *age* or *skin* colour. Note that since the reconstructor never sees the source image directly, but its masked versions, the inpainted solution tends to be a slightly different face. Besides that, due to the intrinsic limitations of our reconstruction model, we can observe that some complex attributes containing high-frequency components might turn out blurry, such as the inner structure of the eyes. Although this is an undesired effect that could cause severe problems for inpainting applications, it is not an issue for our approach, as the inpainted outputs are an intermediate step in our architecture, and eventually the final outputs become sharper after the generator transfers the target attributes.

Method	PSNR (dB) ↑	SSIM ↑
SIIWGAN [76]	19.20	0.920
SIIDGM [58]	19.40	0.907
CE [17]	21.30	0.923
GL [60]	23.19	0.936
GntInp [19]	23.80	0.940
GMCNN [77]	24.46	0.944
GL+LID [78]	25.56	**0.953**
Ours	**31.80**	0.946

Table 3.1. Reconstruction-level evaluation on inpainting. Quantitative comparison between our inpainted result and other inpainting approaches.

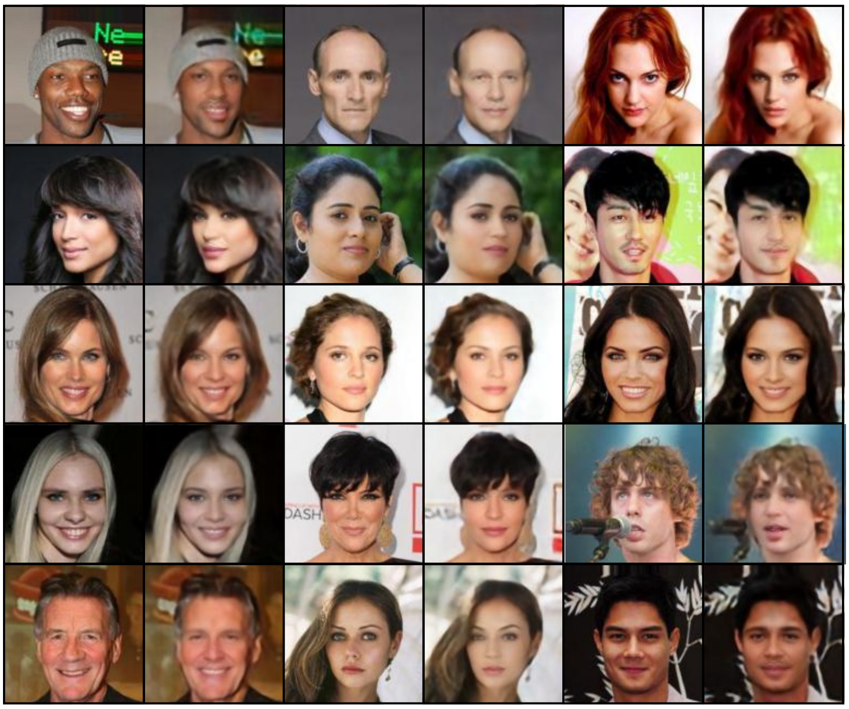


Figure 3.10. Example of inpainted results at testing times. Each pair consists of an original image (left), and its reconstruction (right). Note that even though some results may be blurry, they are just an intermediate step, and the final outputs will be sharp again.

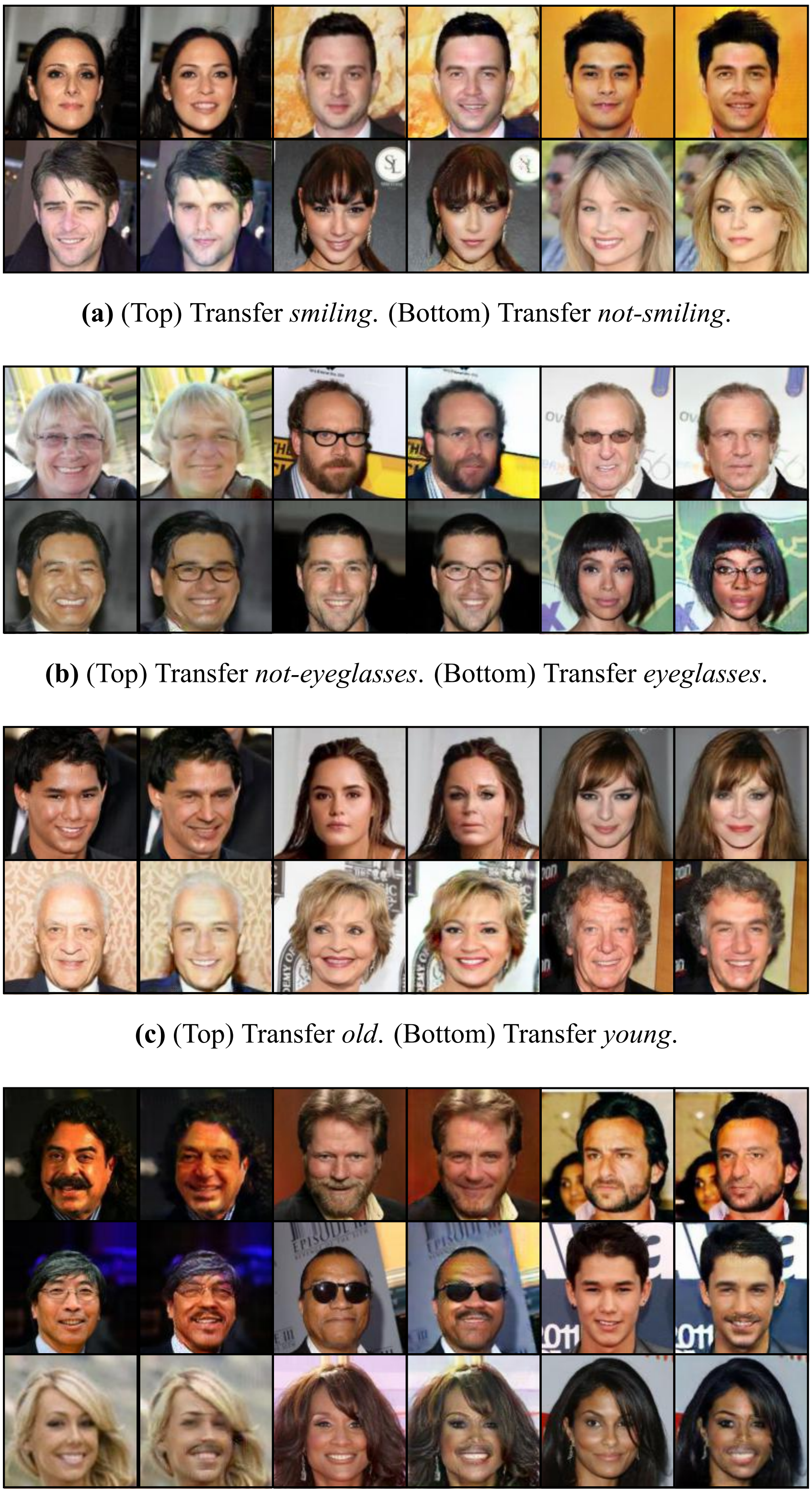
Our inpainting results demonstrate that the proposed approach is able to exploit the benefits of an end-to-end GAN-based architecture, propagating informative gradients throughout the whole system, achieving significant scores on reconstruction evaluation. Nevertheless, notice that we do not aim to outperform state-of-the-art image inpainting techniques, but rather we use the inpainting algorithm as a key element of our attribute-transfer system.

3.4.5 Results of Attribute Transfer

As we mentioned before, our ultimate objective is to build a system that successfully performs attribute transfer. In this subsection, we focus on the assessment of such a task. In particular, we validate the manipulation of the following facial attributes: *eyeglasses*, *moustache*, *smiling* and *young*, and their opposites: *not-eyeglasses*, *not-moustache*, *not-smiling* and *old*.

We start running a qualitative evaluation of the results. Figure 3.11 shows a few generated examples, where inpainting and attribute-transfer networks have been employed in an end-to-end system to synthesize the target attribute. We can observe how ATI-GAN clearly fulfils its goal, synthesizing natural-looking faces that include the pre-defined specifications. Furthermore, the results contain detailed structures, removing the blurriness that the intermediate inpainted images suffer from.

We conduct a user study to evaluate the attribute-transfer capability of ATI-GAN. We focus on the aforementioned list of facial attributes. Results in Table 3.2 demonstrate that the majority of our translations achieve high rates of success. More interesting, however, is to understand the meaning behind these percentages. The first relevant event happens when moving from *eyeglasses* $\rightarrow$ *not-eyeglasses* and vice versa. The perceptual evaluation shows a substantial dissymmetry upon transferring this attribute. The main reason for this is the non-invertible nature of our approach, meaning that moving from domain A to B involves one path, whereas moving from B to A involves another. Consequently, in order to build these attribute paths, our model needs to be exposed to translations from both directions. We hypothesize that an unbalanced exposure to certain attributes, in this case *eyeglasses*, might have a significant effect on the model's final performance. In fact, it is well-known that the CelebA dataset suffers from unbalanced attribute distributions. An even more prominent example of this kind is the *moustache* attribute. Following the same line of thought, it is expected that *not-moustache* $\rightarrow$ *moustache* has a lower success rate for women than for men, since presumably, there are no examples of women with a *moustache*. Finally, Figure 3.12 shows a comparison of attribute-transfer accuracy between different baselines and our proposal. Together with StarGAN, ATI-GAN provides the most stable transfer results across the different target attributes.



(a) (Top) Transfer *smiling*. (Bottom) Transfer *not-smiling*.

(b) (Top) Transfer *not-eyeglasses*. (Bottom) Transfer *eyeglasses*.

(c) (Top) Transfer *old*. (Bottom) Transfer *young*.

(d) (Top) Transfer *not-moustache*. (Centre) Transfer *moustache* for men. (Bottom) Transfer *moustache* for women.

Figure 3.11. Example of attribute-transfer results at testing times. Each pair consists of a source image (left), and its transfer image (right). The variety of faces displayed in the examples demonstrates the capacity of our model to deal with attribute transfer, independently of *gender*, *age*, *skin* colour among other facial features.

Attribute transfer	Accuracy (%)
smiling → *not-smiling*	86
not-smiling → *smiling*	88
eyeglasses → *not-eyeglasses*	50
not-eyeglasses → *eyeglasses*	82
young → *old*	85
old → *young*	70
moustache → *not-moustache* (men)	33
not-moustache → *moustache* (men)	98
not-moustache → *moustache* (women)	68

Table 3.2. Attribute-level evaluation on transformed images for each target attribute. Note that adding and removing attributes are not symmetric operations, and neither are their final scores.

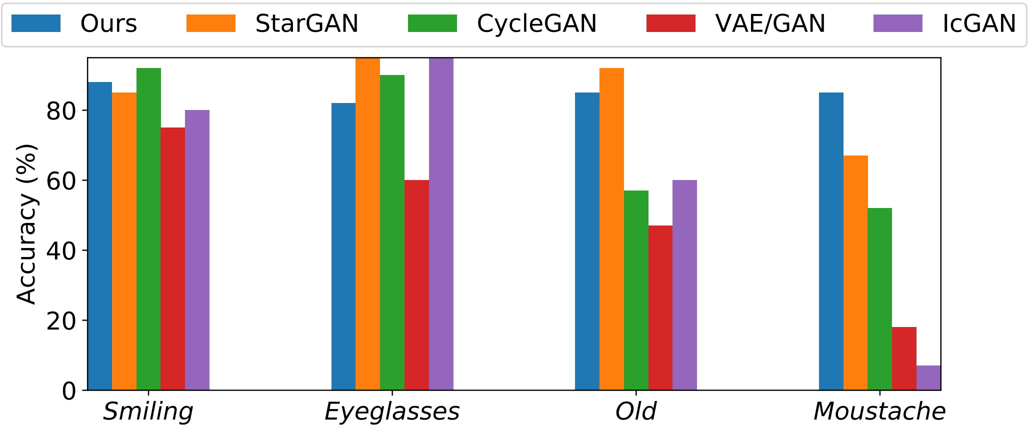


Figure 3.12. Attribute-level evaluation on transformed images for *smiling*, *eyeglasses*, *old* and *moustache* attributes. Quantitative comparison between the baseline models StarGAN [16], CycleGAN [15],VAE/GAN [71], IcGAN [70] and our approach.

3.5 Ablation Study

We present further results that support the importance of the inpainting block and its contribution to the ATI-GAN framework. In this ablation study, we evaluate the outcomes quantitatively, applying the reconstruction-level metrics. Despite their name, these are also valid scores for assessing our final attribute-transfer results in terns of image quality. We first compare our approach to StarGAN, as both methods offer the best results on average across the attributes (see Figure 3.12). In Table 3.3 we can observe how ATI-GAN obtains superior scores, having important gains in both metrics. Thus, one can confirm the superiority of ATI-GAN in the presented scenario. Last but not least, we assess the role of our reconstruction block. To conduct this experiment, we simply remove the inpainting network, and train the model again. Qualitative results show that the implementation of the reconstructor, leads to a notable improvement, consistently boosting both metric scores—by around 2dB on PSNR, and 5% on SSIM.

Method	PSNR (dB) ↑	SSIM ↑
StarGAN [16]	22.80	0.819
Ours (w/o inpainting)	29.84	0.901
Ours	**31.80**	**0.946**

Table 3.3. Reconstruction-level evaluation on attribute transfer. Quantitative comparison on final synthetic results between StarGAN [16], our ablated model and our end-to-end model.

3.6 Limitations

The ATI-GAN framework exploits the local structures of the face, e.g., the relative position between eyes and ears, to inpaint and then to transfer attributes with a high degree of freedom. While the symmetrical property of local structures allows synthetic faces to be successfully rendered containing the target attributes, it also constrains this framework's usage for other applications. This is due to the method having a strong dependency on the structural properties of the input data. As a result, in order to explore different applications, one needs either to employ images with structural similarities, or to implement a stronger reconstructor. A second drawback of ATI-GAN arises from the inpainting strategy. As we explained before, the reconstructor provides one of many possible solutions. As a consequence, the identity of the original face might not be preserved, or it might be slightly modified. In our experiments, we observe that this is not the usual case, however, sometimes small alterations can be found. On the other hand, such small discrepancies can be viewed as a source of valid permutations, leading to a wider range of output possibilities.

3.7 Summary

In this chapter, we introduce a novel image-to-image translation model that performs accurate local attribute transfers. While previous work was mostly based on attribute manipulations through GAN's latent space, we propose a complementary approach that allows the inpainting techniques to be exploited in attribute-transfer scenarios. ATI-GAN is a functional end-to-end attribute-transfer model that utilizes the inpainting paradigm to try to change only the targeted parts of the image, i.e., the pre-defined attributes, while the remaining parts are used to reinforce the generator's learning to produce realistic outputs. Results of our experiment show that our proposal synthesizes high-quality human faces, containing the target attributes. We do believe the method is generalizable to other objects and domains, since it is able to produce synthetic images that contain certain specifications on demand. Interesting avenues of future work could include the exploration of multi-attribute transfer, high-resolution synthesis, and new application domains, among others.

Chapter 4

Style Transfer and Attribute Manipulation

Facial image manipulation is a challenging task that involves generating faces whilst preserving the subtle texture of their relevant features. When editing, different levels of structural changes are imposed on key characteristics so that the system steers in the target direction, aiming to synthesize realistic facial images. Generative neural networks have been very successful in this task due to their ability to produce lifelike faces with a high degree of variability, i.e., providing diverse geometry, textures and colours. Therefore, they have been utilized in different editing scenarios. Common editing techniques, such as image-to-image translation, include style transfer [14, 27] and attribute manipulation [16, 28].

Style-transfer approaches define two or more visual domains, where the goal is to translate images from one domain to another. Usually, these domains represent distinguishable properties, like *gender*, *hairstyle*, and *skin* colour, among others. Ideally, while training these models, one needs to pre-define the boundaries of the target domains/styles, as they can be arbitrarily large. To address this problem, StarGANv2 [27] introduces an approach where the styles are controlled either by domain-specific encoders or by semantic labels. This structure allows image-to-image translation models to handle a wide diversity of styles. Despite StarGANv2's remarkable results, it has two important limiting factors. First, the scaleability over multiple domains, as it has a direct impact on the size of the architecture and consequently, the model is not appropriate for certain scenarios. Additionally, it might require pre-classification of the data, according to the target domain, which is not always a simple task since some domains cannot be binarised. The second limitation comes from its intrinsic semantic-manipulation property, i.e., it translates whole images, not allowing local nor pixel-wise manipulations. As a result, the model cannot have full control over the faces, generating less diverse attributes than other attribute-manipulation techniques.

Thanks to the semantic reasoning of facial-attribute manipulations, one can achieve a rich generation of attributes. These manipulations consist of an image-generation process, closely bound to the style transfer, with supervision from well-defined features (target attributes), which

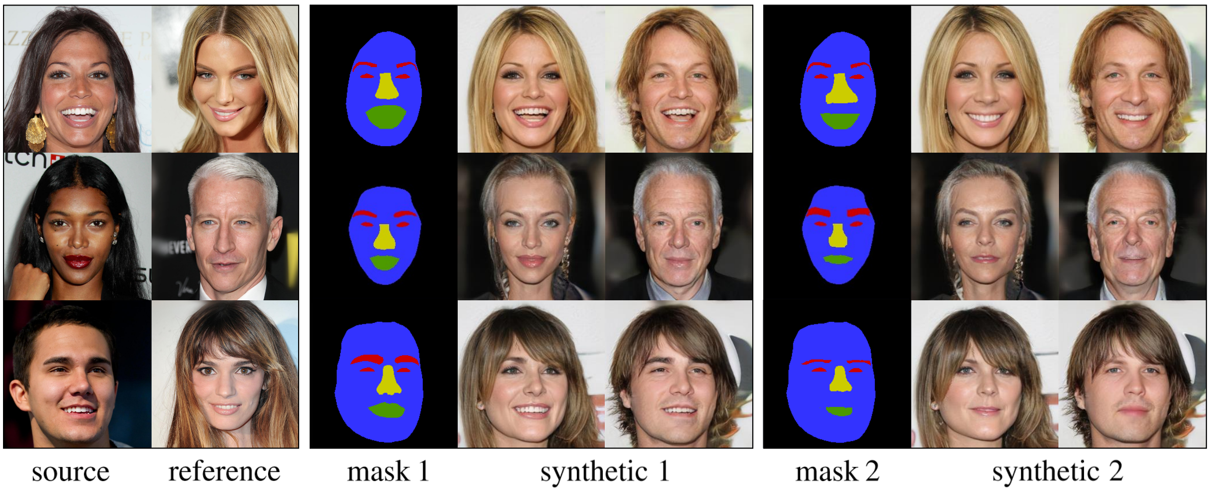


Figure 4.1. Given a source image, its semantic-segmentation mask and a reference image, users can change the mask so that the synthetic result follows the modified mask, while retaining the identity from the source and the style from the reference. Additionally, the *gender* can also be chosen. In this figure we show three cases (rows) each with two segmentation masks and their synthetic results. Note that the mask 1 column is not modified, while mask 2 contains modifications of the *eyes*, *nose* or *mouth* attributes.

are modified with consistency, preserving realism compared with the rest of the face. We can categorize attribute-manipulation techniques into semantic-level manipulations [16, 79, 80] and geometry-level manipulations [81–83]. The former is precise and easy to train, but it does not allow users to interactively alter the face images. On the other hand, the latter achieves notable results using segmentation masks as intermediate representations of facial features, granting the users more freedom to manipulate the attributes at will. However, these geometry approaches still struggle to generate high-diversity outputs. MaskGAN [28] tries to overcome this drawback by mimicking the user's manipulation via a mask manifold. In this way, the model trains with a wide variety of masks, resulting in a richer output diversity when synthesizing. Nevertheless, this method is limited in terms of style transfer, since it lacks the ability to conduct morphological changes when applying the transfer. A solution to mitigate this issue is to describe the style via the available attributes, but it is quite restrictive and impractical, all things considered.

In this chapter, to address these limitations, we propose FacialGAN, a model that learns both style transfer and attribute manipulation, thereby ensuring that it can deal with all the aforementioned tasks at once. Similarly to StarGANv2, we design a network that extracts and applies diverse styles to generate realistic facial images. Furthermore, inspired by MaskGAN, we incorporate geometry information via semantic-segmentation masks to achieve a fine-grained manipulation of facial attributes, leading to a rich diversity of outputs. Figure 4.1 displays a few samples of different style and attribute transformations from FacialGAN.

4.1 Background

In this section, we formally define the semantic-segmentation problem, and provide an overview of the related work, paying special attention to GAN-based approaches.

4.1.1 Preliminaries

Semantic segmentation can be defined as the task of classifying each pixel in an input image from a pre-defined set of classes. Therefore, it can be viewed as a classification problem at pixel level. Due to their multi-domain versatility, segmentation models play a central role in a broad range of applications, including medical image analysis, autonomous driving, video surveillance, and augmented reality to count a few. Over the past few years, deep-learning based methods [31, 84–86] have yielded a new generation of image-segmentation models with remarkable performance improvements—often achieving the highest accuracy rates in popular benchmarks—resulting in a paradigm shift in the field. Figure 4.2 presents a few examples of image segmentation outputs.

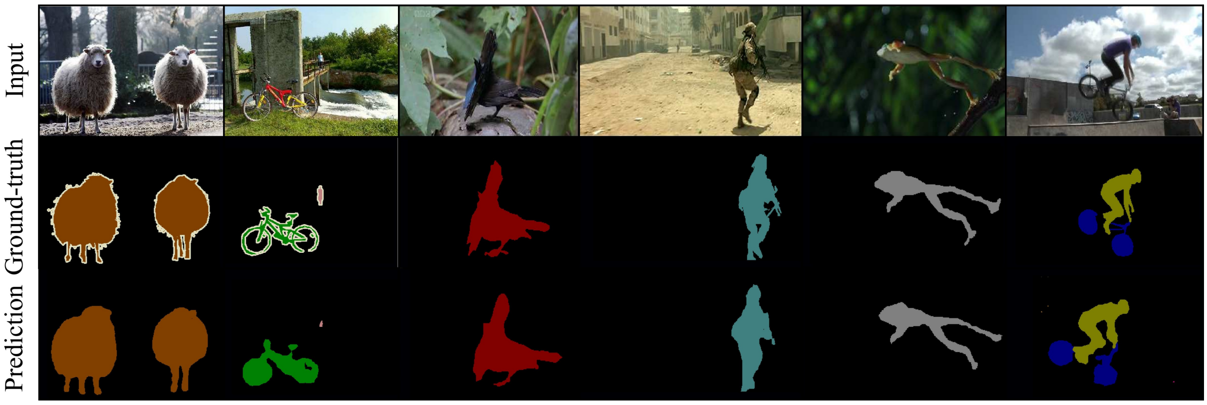


Figure 4.2. Semantic-segmentation results of Durall *et al.* [21].

4.1.2 Related Work

GANs [3] have shown impressive results in various computer vision tasks like image generation [8, 10], image-to-image translation [14, 87], inpainting [59, 88] and image segmentation [89, 90]. Among them, facial manipulation tasks have continuously gained attention in recent years due to the high demand of facial editing applications. Facial manipulation can be seen as a multi-domain image-to-image translation problem, where the model works with a unique domain (features) from the face. From a style-transfer point of view [14, 15, 27, 79, 91–94], each domain consists of styles, e.g., *hairstyle*, *makeup* and *skin* colour. From an attribute-translation perspective [16, 28, 70, 82, 83, 95–99], each domain consists of attributes, e.g., *smiling*, *eye-*

glasses and *moustache*. Note, however, that these domain definitions are rather loose, and some styles can fall into attribute categories and vice versa.

Pix2pix [14] was one of the pioneers in style translation, and suggested learning to map from a source to a target domain, using paired images in a supervised manner. Shortly after, new techniques, such as cycle consistency loss [15] or shared latent space [79], were introduced to remove the need for pairs, reducing the dataset complexity in this way. Concurrently, more advanced topologies were developed, e.g., cascaded refinement [91] or multiscale [92]. Follow-up work proposed using disentangled representations: DRIT [93], based on a domain-invariant content space and a domain-specific attribute space; MSGAN [94], based on a maximization of the ratio of the distance between generated images and their latent codes; and StarGANv2 [27], based on a framework that tackled both diversity of generated images and scaleability over multiple domains.

Attribute-translation tasks have also undergone major improvement, including semantic-level and geometry-level manipulations. Under the semantic-level umbrella, DNA-GAN [95] tried to approach the attribute-translation task by generating swappable attribute-related blocks in the latent space between two images. IcGAN [70] combined a conditional GAN with an encoder, which allowed multiple attributes to be manipulated at once. StarGAN [16] introduced an important breakthrough by employing a single generator to perform multi-domain image translations. AttGAN [96] also achieved remarkable results with an encoder-decoder architecture, where the attribute information had been treated as part of the latent representation. ModularGAN [100] proposed a modular architecture consisting of several reusable and composable modules. GANimation [101] trained its model on facial images with real-valued attribute labels, and thus could achieve impressive results in facial expression interpolation. For a finer control, RelGAN [97] primarily took advantage of relative attributes, which described the desired change on selected attributes. As for geometry-level attribute approaches, ELEGANT [98] introduced a new model that used two images of opposite attributes as inputs, to transfer exactly the same type of attribute from one image to the other by exchanging certain parts of their encodings. A different approach was implemented by SPADE [82], where they employed the input layout for modulating the activations in normalization layers. Chen *et al.* [99] suggested splitting facial attributes into multiple semantic components, each of which corresponding to a specific facial region. More recent approaches have incorporated semantic-segmentation masks to grant pixel-wise control of the synthetic outputs. Gu *et al.* [81] introduced a mask-guided portrait-editing framework, leveraging conditional GANs, guided by provided face masks. Similarly, MagGAN [83] developed a segmentation mask-guided conditioning strategy that incorporated into the generator the influence region of each attribute, which was combined with a multi-level patch-wise discriminator. Finally, MaskGAN [28] also enabled face manipulation via semantic-segmentation masks, which served as intermediate representations for flexible modifications with fidelity preservation. However, it learnt the face-manipulation process via a mask manifold, instead of directly transforming images in the pixel space.

4.2 Contributions

In this work, we focus on the challenging task of facial image editing. Given a source face image, a target-style face image, and a guide segmentation label mask, our novel model is able to synthesize an output image that first, shares a similar style with the target-style image while preserving the input face identity, and second, accurately follows the semantic mask. To the best of our knowledge, FacialGAN is a pioneer of incorporating both techniques under the same framework, leading to flexible and fine-grained editing control. Moreover, our facial system achieves state-of-the-art scores in reference-guided synthesis, improving seminal work, such as StarGANv2 and MaskGAN. To achieve this, we propose a multi-objective training process that is able to balance the different components of the architecture. In particular, we introduce a new local segmentation loss to encourage the network to follow the geometry specified in the guidance face mask. Unlike MaskGAN, where a complex training strategy generates supervision signals, our segmentation loss back-propagates informative gradients thanks to its locality characteristics. In other words, it exploits the region of interest (the target attributes) at pixel level. Overall, our contributions are summarized as follows:

- We propose FacialGAN, a novel model enabling rich simultaneous style transfers and interactive facial attribute manipulations, while maintaining the identity.
- We introduce an intuitive local segmentation loss that guarantees the pixel-wise attribute control, simplifying the complex global pipelines of MaskGAN.
- We assess results, both qualitatively and quantitatively, on the CelebA-HQ dataset. We report state-of-the-art scores on reference-guided generation, surpassing StarGANv2 and MaskGAN.

4.3 Method

We start this section by introducing the problem definition. Then, we provide a detailed presentation of the architecture of our model that splits into four parts. After that, we explain the loss function used in our training process, including its various terms and the reason for them. Eventually, we close this section with the description of our end-to-end training strategy.

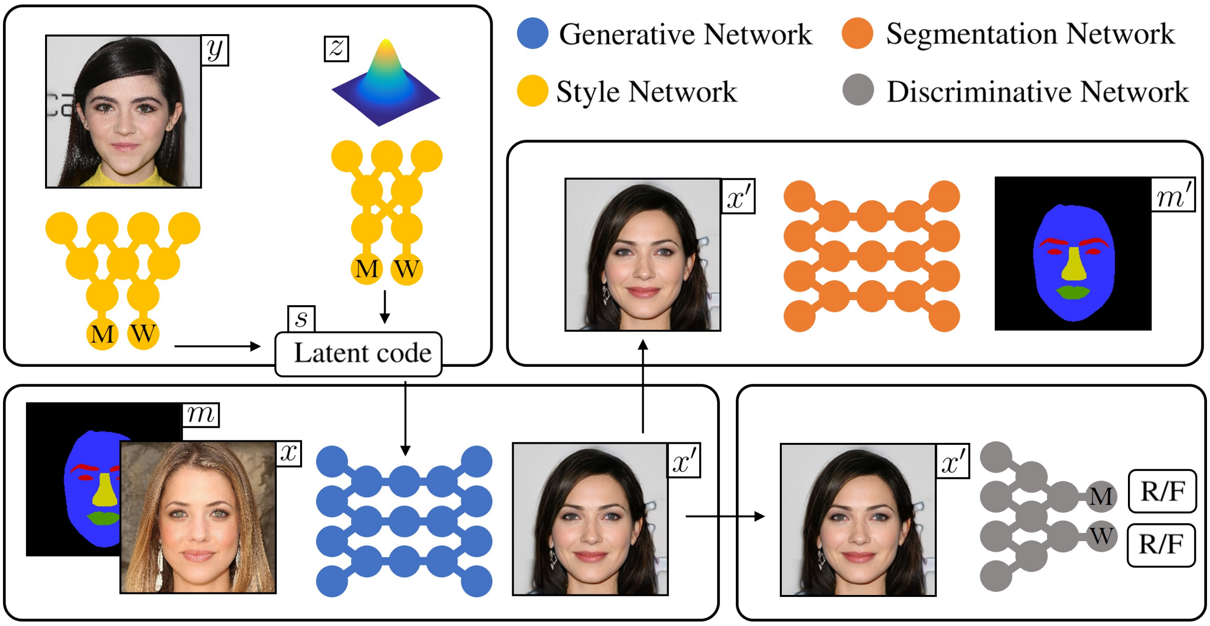


Figure 4.3. Pipeline of FacialGAN framework at training. First, the style network extracts the latent code s containing the style. It can be defined either from a reference image y, or from random noise z. It is in this step that the *gender* is specified: *man* (M) or *woman* (W). Then, the latent code s together with the source image x and its semantic-segmentation mask m are fed into the generator to synthesize x'. To ensure that the generated face x' is consistent with the mask m, we employ a segmentation network. Finally, the discriminator classifies the output as real (R) or as fake (F) according to its *gender*.

4.3.1 Problem Definition

Our proposed model addresses the attribute-translation task in a semantic-segmentation context, while dealing simultaneously with the style-transfer task. Given a source image $x \in \mathbb{R}^{H \times W \times 3}$, its segmentation label mask $m \in \mathbb{R}^{H \times W \times N}$, an arbitrary reference image $y \in \mathbb{R}^{H \times W \times 3}$, and random noise $z \in \mathbb{R}^{1 \times 16}$, our goal is to train a model that can synthesize new faces. These faces would be conditioned on the style from y and z, including the *gender* domain, which are consistent with the geometry constraint of m, and that maintain the identity from x. Note that H and W are the height and width of the data, respectively, and N is the category number of the semantic label.

4.3.2 Model Architecture

The pipeline of our proposal is depicted in Figure 5.3. It is composed of a generative network, a style network, a segmentation network and a discriminative network. By combining each of these blocks sequentially, the ensemble model successfully transfers styles and attributes.

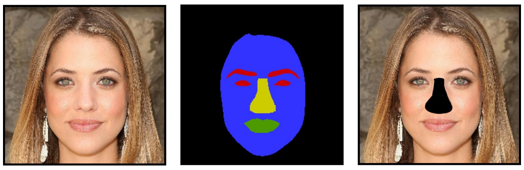


Figure 4.4. From left to right, a source image, its segmentation mask, and its masked input. In this example, the random masked attribute is the *nose*.

Generative Network. The task of the generator G is to translate source images $\boldsymbol{x}$ into output images $\boldsymbol{x'}$, which retain the identity from the source, following the label masks $\boldsymbol{m}$, while reflecting the style codes $\boldsymbol{s}$. Inspired by ATI-GAN [87], we use an encoder-decoder topology and randomly mask one of the attributes of $\boldsymbol{x}$, so that the network learns to inpaint coherent attributes (see Figure 4.4). Then, we concatenate the masked images with the segmentation masks $\boldsymbol{m}$, and feed them into the encoder. For the style transfer, we inject the latent codes $\boldsymbol{s}$ into the decoder using adaptive instance normalization, introduced by Hang *et al.* [102].

Style Network. The aim of this network is to generate valid style codes $\boldsymbol{s}$. To do so, the network is split into two subnetworks: a mapping network F and an encoder network E. While F generates style codes from random noise $\boldsymbol{z}$, E extracts the style from reference images $\boldsymbol{y}$. We adopt the architecture of StarGANv2 [27] as a backbone and simplify it for a binary domain to control the *gender* information. The remaining attributes are manipulated through the mask.

Segmentation Network. To guarantee a diverse and interactive face manipulation, we need to ensure that G follows the geometry of the mask. Therefore, the moment that the outputs $\boldsymbol{x'}$ and $\boldsymbol{m}$ are not aligned, the segmentation network S generates a control signal that penalizes the generator. To achieve this, we feed $\boldsymbol{x'}$ into S and compare the generated output masks $\boldsymbol{m'}$ with the label masks $\boldsymbol{m}$, which serve as ground-truth. Furthermore, in order to improve results, we apply random modifications on the input mask m, such as erosion and dilation, at training time. In this way, the model is exposed to a more diverse mask scenario.

Discriminative Network. The last component is a convolutional discriminator D. However, it behaves slightly differently from the vanilla implementation [3], as it takes samples of both real and generated faces and tries to correctly classify them as real and fake, based on the *gender* domain. This discrimination procedure is called "multitask classification" and it has been successfully employed in previous work [27, 103].

4.3.3 Multi-Objective Learning

Learning to synthesize realistic and diverse images while transferring styles and manipulating attributes is a complex task; it requires different regulariser terms that focus on specific tasks. In this work, we mainly use five independent losses during training to achieve our goal.

$$\mathcal{L}_{\text{adv}} = \mathbb{E}_{\boldsymbol{x}}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{x},\boldsymbol{m},\boldsymbol{z}}[\log(1 - D(G(\boldsymbol{x}, \boldsymbol{m}, F(\boldsymbol{z}))))] \tag{4.1}$$

The adversarial loss $\mathcal{L}_{\text{adv}}$ [3] is the core element in any GAN-based model. Essentially, it makes the generated images more realistic and assess the control over the *gender* domain.

$$\mathcal{L}_{\text{sty}} = \mathbb{E}_{\boldsymbol{x},\boldsymbol{m},\boldsymbol{z}}[||\boldsymbol{s} - E(G(\boldsymbol{x}, \boldsymbol{m}, \boldsymbol{s}))||_1] \tag{4.2}$$

The style loss $\mathcal{L}_{\text{sty}}$ [27] is vital to achieving reliable style transfers. It is responsible for forcing the generator to utilize the style codes $\boldsymbol{s}$, which are extracted from $F(\boldsymbol{z})$. This is possible by minimizing the distance between them and those extracted from E when fed with generated images $\boldsymbol{x'} = G(\boldsymbol{x}, \boldsymbol{m}, \boldsymbol{s})$.

$$\mathcal{L}_{\text{ds}} = -\mathbb{E}_{\boldsymbol{x},\boldsymbol{m},\boldsymbol{z_1},\boldsymbol{z_2}}[||G(\boldsymbol{x}, \boldsymbol{m}, F(\boldsymbol{z_1})) - G(\boldsymbol{x}, \boldsymbol{m}, F(\boldsymbol{z_2}))||_1] \tag{4.3}$$

By maximizing the distance between two generated images with respect to their corresponding latent codes $\boldsymbol{z_1}$ and $\boldsymbol{z_2}$, the diverse sensitivity loss $\mathcal{L}_{\text{ds}}$ [94] forces the generator to explore more minor modes, and therefore, to produce more diversity.

$$\begin{aligned} \mathcal{L}_{\text{cycle}} = &\mathbb{E}_{\boldsymbol{x},\boldsymbol{m},\boldsymbol{z}}[||\boldsymbol{x} - G(\boldsymbol{x'}, \boldsymbol{m}, E(\boldsymbol{x}))||_1] \\ &\text{with } \ \boldsymbol{x'} = G(\boldsymbol{x}, \boldsymbol{m}, F(\boldsymbol{z})) \end{aligned} \tag{4.4}$$

The cyclic consistency loss $\mathcal{L}_{\text{cycle}}$ [15] guarantees the preservation of the domain's invariant characteristics, such as pose, while changing the styles faithfully.

$$\mathcal{L}_{\text{seg}} = -\sum_{h,w} \mathbb{E}_{\boldsymbol{x}} \left[\boldsymbol{m}^{h,w,c} \log S(\boldsymbol{x}^{h,w,c}) + (1 - \boldsymbol{m}^{h,w,c}) \log(1 - S(\boldsymbol{x}^{h,w,c}))\right] \tag{4.5}$$

Finally, the local segmentation loss $\mathcal{L}_{\text{seg}}$ is based on binary cross-entropy formulation with the singularity that it works locally. Depending on the manipulated attribute c, $\mathcal{L}_{\text{seg}}$ will evaluate a certain region (h, w) of the image. The goal of this loss is to ensure that the mask rules the attribute geometry of output images.

Combining all the aforementioned loss terms leads to the final objective, which can be formulated as

$$\min_{G,F,E,S} \max_{D} \mathcal{L}_{\text{final}} = \lambda_{\text{adv}}\mathcal{L}_{\text{adv}} + \lambda_{\text{sty}}\mathcal{L}_{\text{sty}} + \lambda_{\text{ds}}\mathcal{L}_{\text{ds}} + \lambda_{\text{cycle}}\mathcal{L}_{\text{cycle}} + \lambda_{\text{seg}}\mathcal{L}_{\text{seg}}, \tag{4.6}$$

where λ_{adv}, λ_{sty}, λ_{ds}, λ_{cycle} and λ_{seg} are the hyperparameters for each term.

4.3.4 End-to-End Model Architecture

To obtain images with diverse styles and flexible attribute manipulation, we train our approach to perform a twofold task: 1) generate domain-specific style vectors from arbitrary images and random noise, and 2) synthesize realistic faces following the geometry dictated by pseudo-random masks. By enforcing such a behaviour, the model learns to reflect the style vectors and the changes to the masks' attributes, producing images with diversity and scaleability over multiple domains. Algorithm 5 describes the pseudocode of our proposal.

Algorithm 5 Training of the FacialGAN model.

```
Require: $n_{\text{iter}}$, number of iterations. $\alpha$'s, learning rates. $m$, batch size.
Require: $\boldsymbol{\theta}$, initial generator, style and discriminator parameters.
Require: $\boldsymbol{\theta}_{\text{seg}}$, pre-trained segmentation parameters.
Note: $\boldsymbol{\theta}_{\text{rest}}$ represents the generator and style parameters.
for $i < n_{\text{iter}}$ do
    Sample a batch of images $\{x^{(j)}\}_{j=0}^{m}$
    Mask of the batch of images $\{m^{(j)}\}_{j=0}^{m}$
    Sample a batch of reference images $\{y^{(j)}\}_{j=0}^{m}$
    Sample a batch of random noise $\{z^{(j)}\}_{j=0}^{m}$
    Train discriminator $D$
    $\boldsymbol{\theta}_{\text{disc}} \leftarrow \boldsymbol{\theta}_{\text{disc}} - \alpha_{\text{disc}} \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\text{adv}}(\boldsymbol{x}, \boldsymbol{m}, \boldsymbol{z})$
    Train generator $G$ and style networks $E$ and $F$
    $\boldsymbol{\theta}_{\text{rest}} \leftarrow \boldsymbol{\theta}_{\text{rest}} - \alpha_{\text{rest}} \nabla_{\boldsymbol{\theta}} (\mathcal{L}_{\text{adv}} + \mathcal{L}_{\text{cycle}} + \mathcal{L}_{\text{seg}} + \mathcal{L}_{\text{sty}} + \mathcal{L}_{\text{ds}}(\boldsymbol{x}, \boldsymbol{m}, \boldsymbol{y}, \boldsymbol{z}))$
end for
```

4.4 Experiments

In this section, we benchmark our approach by showing quantitative and qualitative results. We first give a detailed description of the experimental setup and the evaluation metrics. Then, we comprehensively analyse different style-transfer setups. Finally, we assess the attribute-transfer capability of our proposal.

4.4.1 Experimental Setup

We train FacialGAN on the CelebA-HQ [104] and CelebAMask-HQ [28] datasets. While CelebA-HQ consists of 30,000 high-quality facial images picked from the CelebA [53] dataset, CelebAMask-HQ consists of the corresponding semantic-segmentation labels, separated into 19 classes. For our experiments we resize all images and label masks to the size of $256\times256\times3$ pixels, and we create and employ four customized classes: *eyes*, *nose*, *mouth* and *skin*. We choose the state-of-the-art DRIT [93], MSGAN [94], SPADE [82], StarGANv2 [27] and MaskGAN [28] models as our baseline models for comparison. DRIT, MSGAN and StarGANv2 perform latent-guided and reference-guided style transfer, whereas SPADE and MaskGAN perform geometry-level facial-attribute manipulation. All the experiments have been conducted in a single NVIDIA Tesla A100 GPU.

4.4.2 Training Setting

Our approach is divided into two blocks. First, we train our segmentation model, based on U-Net [31], with the default Adam [54] optimizer with learning rate set to 10^{-2}. The batch size is set to 32, and the model is trained for 50 epochs. Then, we train our generative model for 200,000 iterations using a batch size of 8, but in this training we have four Adam optimizers, where we set $\beta_1 = 0$ and $\beta_2 = 0.99$. As for the learning rates, we set 10^{-4} for G, D and E, and 10^{-6} for F. The losses are all equally weighted, except for the segmentation with $\lambda_{\text{seg}} = 2$, and the style diversification, where λ_{ds} is linearly decayed to zero over training. We implement the non-saturating adversarial loss with R1 regularization [105], and we set $\gamma = 1$.

4.4.3 Evaluation Metrics

Distribution-level Evaluation. To evaluate both the diversity and the visual quality from different models, we use the Fréchet inception distance (FID) [106] and the learnt perceptual image patch similarity (LPIPS) [107] metrics. FID measures the discrepancy between generated and real images, by using the Fréchet distance between features extracted from the last average pooling layer of Inception-V3 [108], pre-trained on ImageNet; these features are then fitted to a multivariate Gaussian distribution. LPIPS measures the diversity between image patches of generated images, using the l_1 distance norm between features extracted from AlexNet [29], pre-trained on ImageNet.

Attribute-level Evaluation. To evaluate the ability of our system to manipulate target attributes, we pre-train binary facial classifiers for the specific attributes on CelebA so that later we can test our attribute manipulation. In particular, we use a ResNet-18 [109] architecture.

Segmentation-level Evaluation. To evaluate the capacity of our proposal to generate synthetic images conditioned on the input mask, we pre-train a facial semantic-segmentation network on CelebA-HQ. In particular, we use a U-Net [31] architecture to measure the pixel-wise accuracy between the input layout and the predicted results.

Method	FID ↓	LPIPS ↑
DRIT [93]	52.1	0.178
MSGAN [94]	33.1	0.389
StarGANv2 [27]	**13.7**	**0.452**
Ours	15.8	0.426

Method	FID ↓	LPIPS ↑
DRIT [93]	53.3	0.311
SPADE [82]	46.2	-
MSGAN [94]	39.6	0.312
MaskGAN [28]	37.1	-
StarGANv2 [27]	23.8	0.388
Ours	**22.8**	**0.415**

Table 4.1. Distribution-level evaluation on style transfer. (Left) Quantitative comparison on latent-guided synthesis. (Right) Quantitative comparison on reference-guided synthesis.

Identity-level Evaluation. In the context of style and attribute facial manipulation, it can be relevant to preserve the identity of the person featured in the image. Therefore, we employ a pre-trained face-verification classifier on LFW [110] dataset. In particular, we use the ArcFace [111] model with an accuracy of 99.5%.

4.4.4 Results of Style Transfer

We start our experimental evaluation by assessing the style-transfer ability of our model from two perspectives: latent-guided synthesis and reference-guided synthesis.

Latent-guided refers to the fact that the system learns to model random noise into valid latent code representations, which account for specific styles. Table 4.1 (left) provides a quantitative comparison of the baseline methods. Our approach provides very competitive results, outperforming most of the models on both FID and LPIPS scores, and coming very close to StarGANv2. The main reason for these scores is the ability to morphologically change the attributes, resulting in a wide variety of synthetic faces. In other words, our model produces highly diverse results with a balanced image quality. Furthermore, we conduct a visual inspection of a few samples. In Figure 4.5 (top), a qualitative comparison between the different baselines is illustrated. Each column contains the style-transfer result from different models when using random noise as an input. The top two rows correspond to the results of converting *man* to *woman* and vice versa in the bottom two rows. We observe that both StarGANv2 and our model generate images with a higher visual quality compared to the DRIT and MSGAN models. While most of the time DRIT synthesizes plausible outcomes, they do not contain morphological changes, leading to poorer style transfers. On the other hand, MSGAN generates results containing more substantial modifications; nonetheless, the method seems to fail to synthesize realistic images.

The second perspective, reference-guided, refers to the fact that the system learns to extract high-level semantics, such as *hairstyle*, *makeup* or *skin* colour, from the reference images,

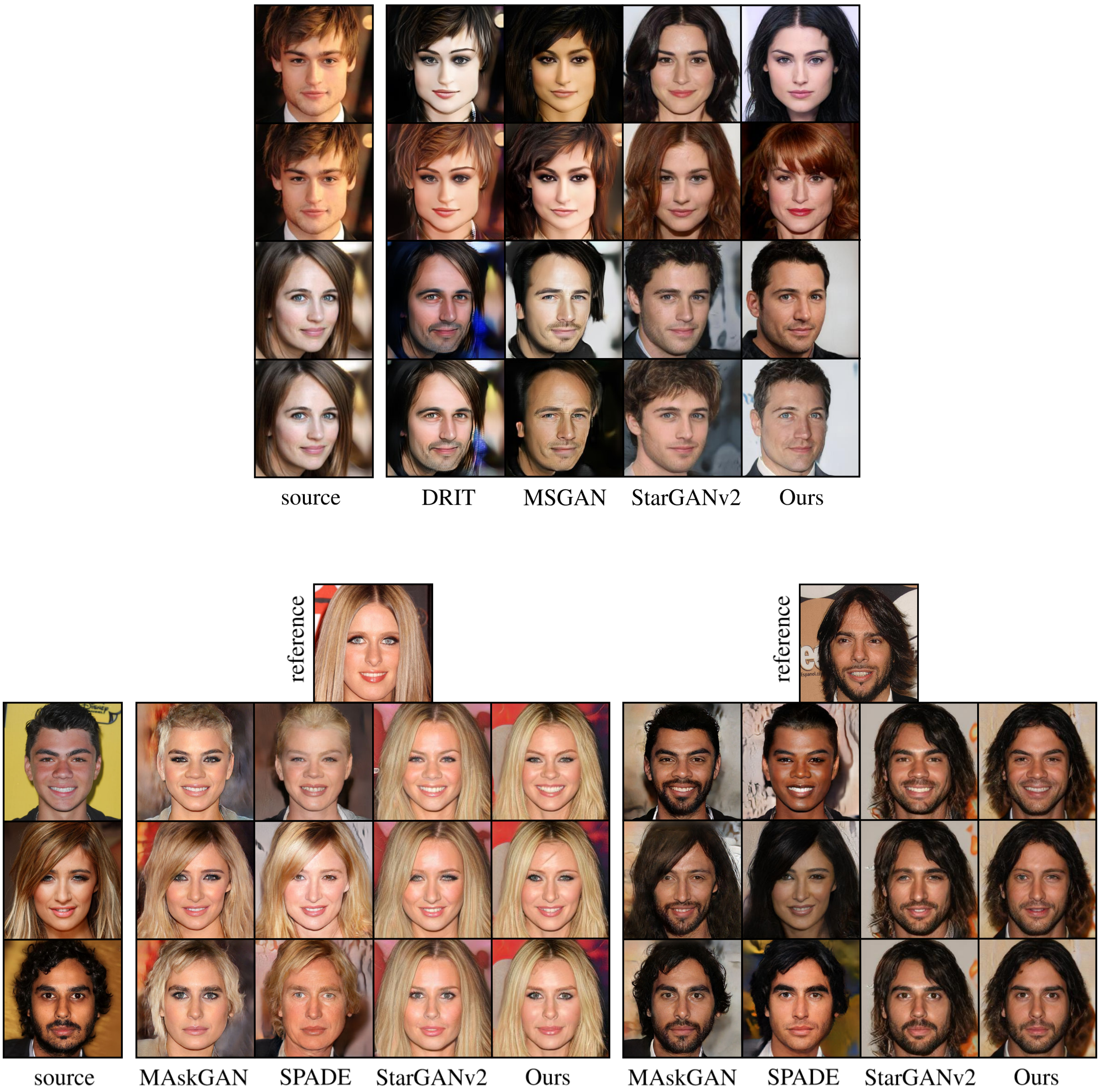


Figure 4.5. Qualitative comparison of style transfer on image synthesis. (Top) Latent-guided generation using random latent codes. (Bottom) Reference-guided generation using input images.

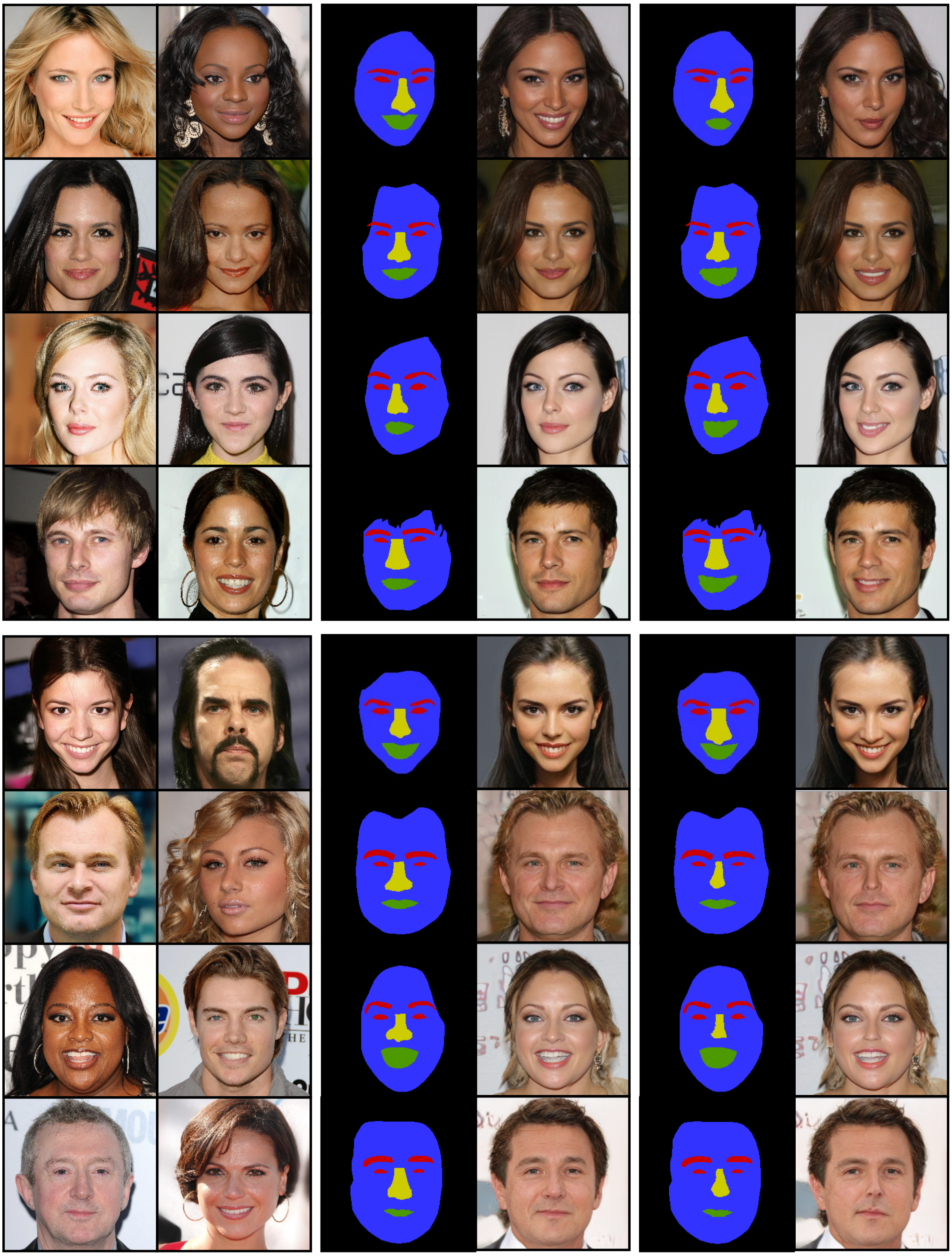
and to represent it in a latent code. The pose and identity of the source images are preserved. Table 4.1 (right) shows the quantitative comparison of our method and the baseline methods for reference-guided synthesis. Additionally, we also benchmark the models from SPADE and MaskGAN. However, note that these methods use reference labels on top of the source images. In this second experiment, our model achieves superior scores in both FID and LPIPS metrics compared to the competing baselines. This implies that our approach produces the most diverse and realistic results, while taking the styles of reference images into consideration. Figure 4.5 (bottom) compares the appearances of FacialGAN with the baseline methods. We observe that our approach and StarGANv2 have successfully rendered distinctive styles like *hairstyle* or *skin* colour, while MaskGAN fails at *hairstyle* translation, and SPADE mostly matches only the colours of reference images.

4.4.5 Results of Attribute Transfer

On top of the style-transfer capability, our model also offers a fine-grained attribute manipulation, based on supervised signals. These signals come from a one-hot encoded vector that determines the *gender*, and, from the semantic facial mask that controls the *eyes*, *eyebrows*, *nose* and *mouth*. Figure 4.6 shows synthesized images with style and attribute modifications.

We start our analysis by investigating the control over attributes, in particular, over the *gender*. One important difference between some baseline models and ours is how the *gender* information is encoded into the generative system. On the one hand, we have SPADE and MaskGAN that employ the reference image to determine the *gender*; in other words, they treat *gender* as a part of the style information. On the other hand, we have our model and StarGANv2 that use a label to set the *gender*, and therefore, treat it independently of the style. The first column of Table 4.2 shows the classification accuracy that each baseline achieves on a *gender* classifier when targeting only *man* outputs. As one can expect, there is a clear difference between those approaches that use a *gender*-specific signifier, and those that do not. Note that the pre-trained classifier has an accuracy of 96.1%, which serves as a ground-truth.

Besides *gender*, our model allows the *eyes*, *eyebrows*, *nose* and *mouth* to be manipulated independently, synthesizing images accordingly. As mentioned above, such control comes from the information of the masks that enable our approach to react to modifications on these masks at pixel-level. Hence, it is possible to scale the size of a specific attribute, or even to redraw it completely from scratch. The main limitation in terms of manipulation may arise from the need for realistic customized masks so that they can be translated into realistic faces. Thus, to validate our model on an attribute controlled by the mask, we conduct an experiment on the *smiling* attribute. We compare with previous work: SPADE and MaskGAN. Drawing smiles is a challenging task since it not only affects the mouth attribute, but also influences the whole



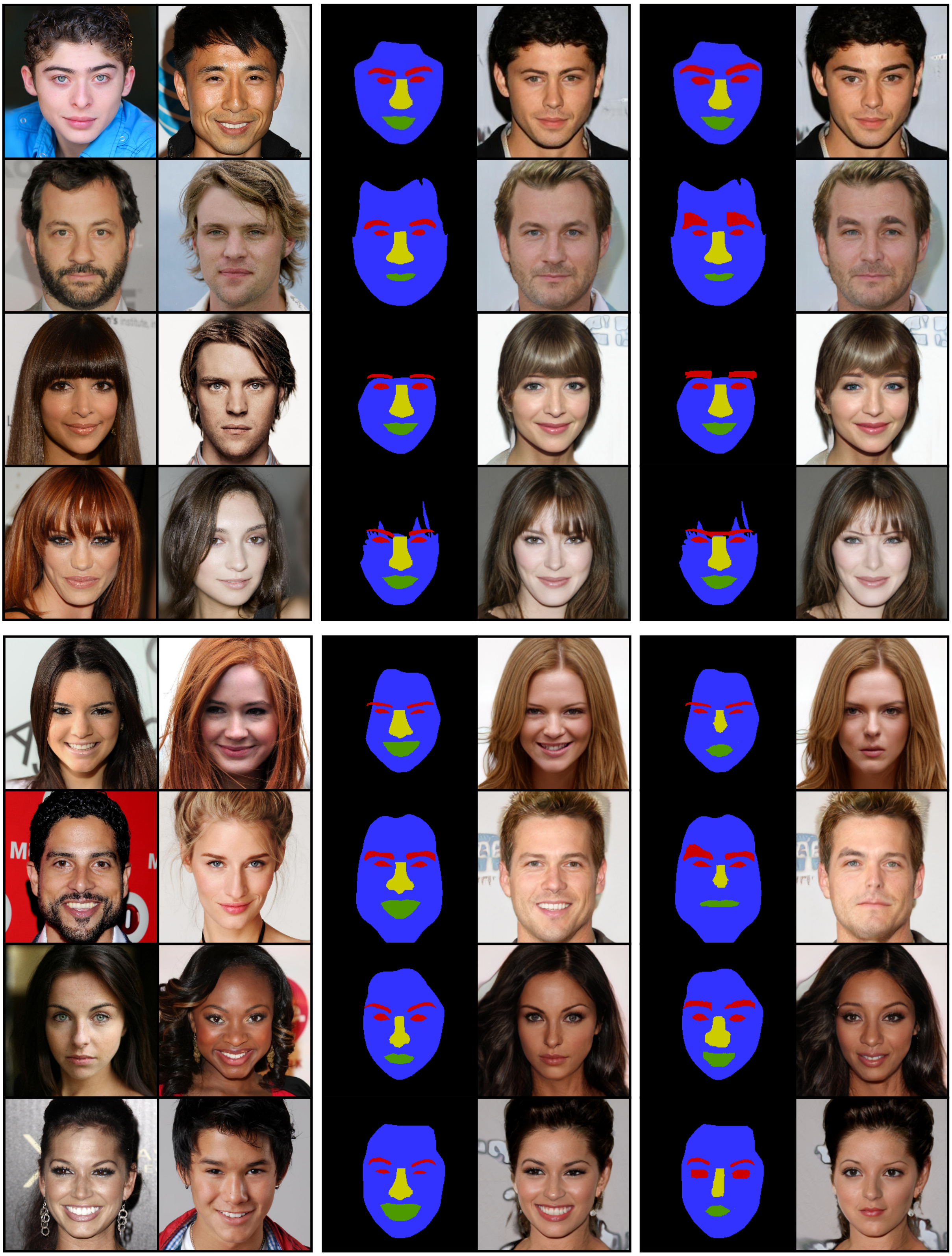


Figure 4.6. In these pictures, we show that our model is able to learn to transform a source image to reflect the style of a given reference image while being consistent with the semantic mask. The source and style-reference images appear in the first two columns, whereas the respective segmentation masks are given in columns 3 and 5. Columns 4 and 6 show the generated images. Each block of four rows displays a modified attribute. From top to bottom: *mouth*, *nose*, *eyes* (*eyebrows*) and mixed.

Method	*Man* accuracy (%)	*Smiling* accuracy (%)	Identity accuracy (%)
SPADE [82]	54.5	73.8	70.7
MaskGAN [28]	71.7	77.3	76.4
StarGANv2 [27]	**100**	-	-
Ours	**100**	**81.4**	**89.8***
Ground-truth	96.1	92.3	99.5

Table 4.2. The second and third columns show the attribute-level evaluation on *man* and *smiling* transfer synthesis accuracy. The fourth shows the identity-level evaluation after drawing smiles. *Note that our synthetic images also contain style modifications, which make identity preservation more challenging.

expression of the face, resulting in large geometry variations. To run this evaluation, we manipulate *not-smiling* masks to become *smiling* ones, and then, we generate a new set of images. The second column of Table 4.2 shows how our method achieves very competitive results, outperforming the baselines.

In addition to the attribute-transfer assessment, we study identity preservation. We first analyse the effect of style transfer on face recognition. To that end, we measure the accuracy of face recognition on the source images and the generated images with style transfer; we reach an identity accuracy of 89.8%. Then, we apply attribute modification, more specifically the *smiling* attribute, and we measure again the accuracy of face recognition between the source images and the generated images. This time, the generated images contain both attribute modifications and style transfer; we reach the same identity accuracy of 89.8%. Table 4.2 shows that even with additional style transfer, our method is able to preserve identity better than the baselines, obtaining a significant improvement.

Last but not least, having high attribute-transfer and identity accuracy are important steps towards our goal. Nevertheless, these metric might be incomplete since it does not evaluate the precision of the mask. Hence, we conduct a segmentation study per attribute (see Table 4.3), where we assess the consistency between the input layouts m and the predicted results m' in terms of pixel-wise accuracy.

4.5 Limitations

We run an empirical study to determine under which circumstances our proposal starts to behave erroneously, producing inconsistent outputs. To find such limitations, we constantly increase the size of the mask of the target attributes and evaluate the results. We repeat the same procedure but decreasing the mask size. Figure 4.7 displays a few examples, where we can see how our

Attribute	Ours	Ground-truth
Eyes accuracy (%)	98.39	98.81
Nose accuracy (%)	99.30	99.45
Mouth accuracy (%)	98.78	99.06
All accuracy (%)	96.40	98.75

Table 4.3. Segmentation-level evaluation to measure the consistency between the input masks and the predicted parsing results in terms of pixel-wise accuracy.

model follows the mask if it can be translated into a plausible, realistic face. Once the modified mask contains unnatural structures, such as no *eyes*, the network starts to ignore the mask input. The main reason for this behaviour is the effect that the discriminator has over the generator during training, forbidding the generator to learn to synthesize unrealistic faces.

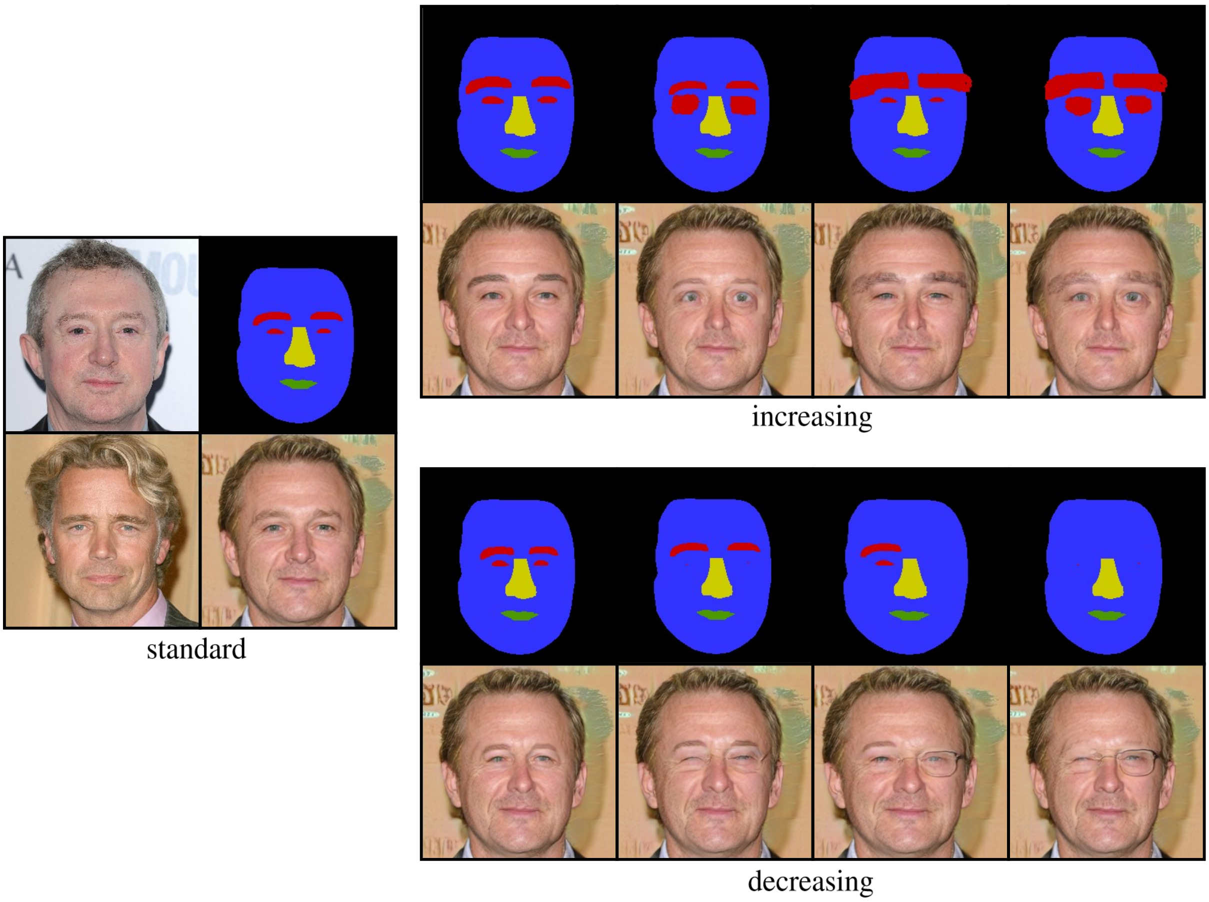


Figure 4.7. Results on extreme manipulation of *eyes* (*eyebrows*) mask.

A second limitation factor arises from our segmentation loss. In order to generate informative gradients, this loss needs to work locally. Therefore, we need training data with pre-defined areas, i.e., the masks of the target attributes. Otherwise, the regions of no interest weaken the learning signal, leading to a possible loss of control of attribute editing.

4.6 Summary

In this chapter, we propose a novel interactive attribute-manipulation and style-transfers framework for facial image editing, named FacialGAN. It learns to extract and to apply diverse styles from reference images, while preserving the input face identity, and to incorporate the geometry information of a guidance segmentation mask. A multi-objective strategy guarantees the generation of high-quality faces, and a fine-grained manipulation of facial attributes. Experimental results demonstrate that our FacialGAN outperforms state-of-the-art approaches. We see many interesting applications for future work, including simulation for virtual try-on *eyeglasses*, *earrings*, *makeup* or even some cosmetic surgeries such as rhinoplasty, eyelid lift, or cheek enhancement to mention a few.

Chapter 5

Perils of Image Manipulation

In the last years, the increasing sophistication of mobile devices and the growth of social networks have led to a gigantic amount of new digital content. This tremendous use of digital images has been followed by a rise in techniques for altering them. Until recently, such techniques were beyond the reach of most users, since they were complex and time-consuming, requiring a high domain expertise in computer vision. Nevertheless, thanks to the latest advances in machine learning and the accessibility to large volumes of training data, those limitations have gradually faded away. As a consequence, the complexity and time needed for generating and manipulating the digital content has significantly decreased, resulting in a democratization of such modification techniques.

Deep generative models have lately been extensively used to produce artificial images with a realistic appearance. These models are based on deep neural networks, which are able to approximate the true-data distribution from a given training set. Thus, it is possible to sample from and manipulate the learnt distribution at will. Two of the most popular approaches are variational autoencoders [2] and GANs [3]. The latter approach, especially, has been pushing forward the limits of state-of-the-art results, improving the control and the quality of synthetic images [8,9,104]. As a result, GANs are opening the door to a new vein of artificial-intelligence-based fake image generation, leading to a fast dissemination of high-quality synthetic content. While significant developments have been made for image forgery detection, it still remains an ongoing research task, since most current methods rely on deep-learning systems, i.e., they are strongly dependent on large amounts of labelled training data.

In this chapter, we address the problem of detecting artificial image content, more specifically, fake faces. Despite the fact that many face-editing and generative algorithms seem to produce flawless realistic human faces [87, 112, 113], upon closer examination, they do exhibit artefacts in certain domains, which are often hidden to the naked eye. In order to spot such irregularities, we present a simple yet effective method, based on classical frequency domain analysis. Compared to previous systems [114–116], which demand large amounts of labelled

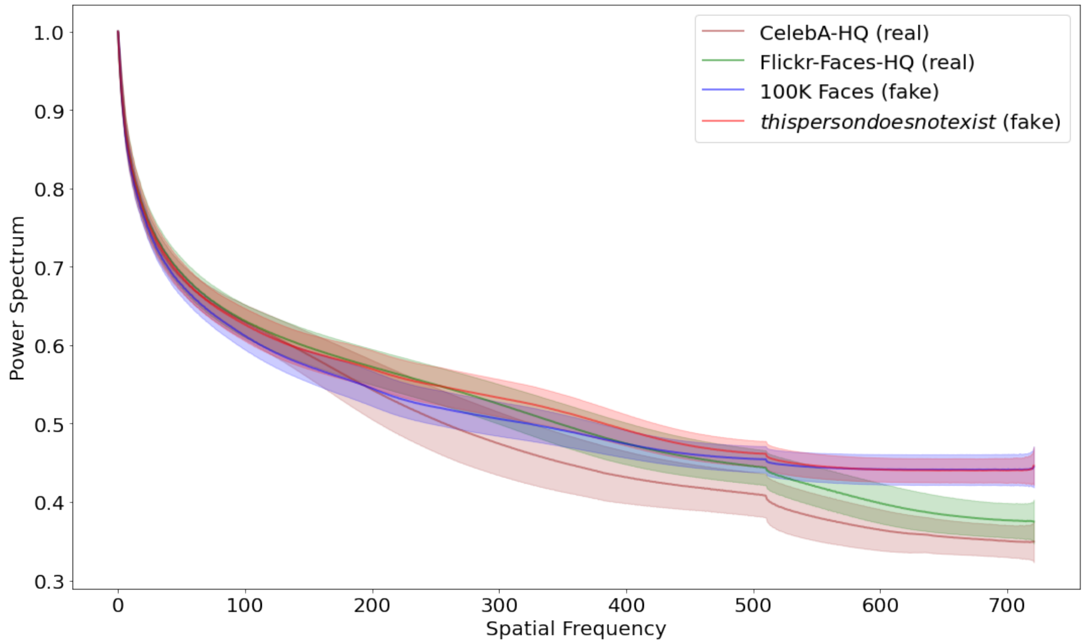


Figure 5.1. Azimuthal integration statistics (mean and variance) of the sub-datasets from the Faces-HQ dataset. The higher the frequency, the larger the difference is between real and fake data.

data, our approach achieves cutting-edge results using only a few annotated training samples, and remarkable accuracy in fully unsupervised scenarios. Figure 5.1 shows the frequency components of real and synthetic data, where one can notice how differently they behave at high-frequency bands.

5.1 Background

In this section, we formally define the forensic analysis of deepfakes, and provide an overview of the related work, paying special attention to GAN-based approaches.

5.1.1 Preliminaries

Nowadays, there are many deep-learning techniques that are able to seamlessly stitch anyone in the world into a photo in which she/he never actually participated. While forged media has been in existence for decades, recent progress has enabled automatic, large-scale creation of realistic-looking fake content. These fake photos (images) and videos are known as deepfakes.

Deepfakes can become very dangerous, since they can be widely propagated on social media, spreading propaganda, disinformation, and fake news in a matter of minutes. For example, the comedian Jordan Peele produced a fake video of President Obama criticizing President Trump by modifying the lip movements and Obama's voice in a real video. This comedy video illustrates the huge potential perils that deepfakes might have, as they essentially cast doubt on the veracity of all online information.

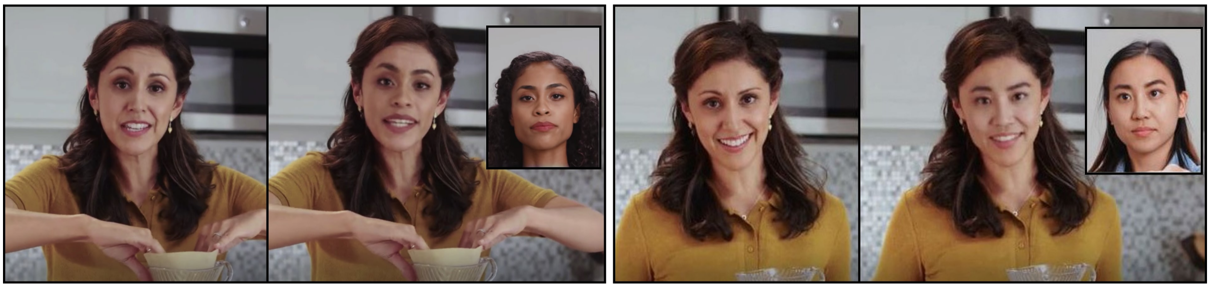


Figure 5.2. Two examples of deepfakes from [117, 118]. Both show the original image (left) and its deepfake result (right), which contains the face of the target person (top right corner), resulting in the concealment of the original person's identity.

5.1.2 Related Work

Traditional image-forensic methods for tampering detection can be classified according to the image features that they work with, such as local noise estimation [119], pattern analysis [120], illumination model [121] and steganalysis feature classification [122]. However, with the deep-learning breakthrough, the computer vision community has radically steered towards neural network techniques. For example, Zhou *et al.* [123] and Cozzolino and Verdoliva [124] employed CNN-based models to try to capture some of the aforementioned image features, but in an implicit manner. In other words, their neural networks learnt to spot altered content, without the need for an explicit definition of the features.

Brundage *et al.* [125] forecast the impact of new deep-learning technologies in a detailed report, and the necessity to mitigate the harmful effects of malicious uses of artificial intelligence. In particular, the introduction of GANs [3] has marked a milestone in generative models, but also in fake-data detection, known as forgery detection. GANs have shown great potential to synthesize and manipulate images and videos, which might sometimes result in deepfake content. In fact, this sort of fraudulent content has been emerging and establishing itself on renowned online portals over recent years. As a result, the image-forensic community has redoubled its efforts to detect such content, paying special attention to deep-learning models.

Li *et al.* [126] noticed that when videos were artificially created, the people featured did not blink. This happened due to the scarcity of training images where the subjects' eyes were closed. Nevertheless, this flaw was rapidly circumvented by adding training pictures displaying people with their eyes closed. Yang *et al.* [127] suggested focusing on finding unnatural head poses to detect modified digital content. However, shortly after, the pose information became a parameter that tampering techniques started taking into consideration, solving this other flaw. On the other hand, Closkey and Albright [128] and Li *et al.* [129] proposed analysing the colour-space features from generated and real images, and exploiting their disparity in order to classify them. In the context of GANs, other approaches [114–116], rather than leveraging explicit lacks or failures, they relied on CNNs to distinguish synthetic from real images. In the same vein, Hsu *et al.* [130] introduced a deep forgery discriminator with a contrastive loss function. Finally, work including temporal domain information by employing recurrent neural networks upon CNNs has also been proposed [131].

Summarizing, although deep-learning methods for forgery detection show promising performance when spotting any kind of anomaly, e.g., not blinking and head pose, most of these flaws can be easily incorporated into the counter-forensic method. For example, by integrating the missing feature into a GAN's discriminator, the generator can be fine-tuned to learn a countermeasure for that specific weakness. Thus, the forgery detection can be viewed as some sort of endless race.

5.2 Contributions

In this work, we address the problem of detection of manipulated face images, including partially modified, fully generated and image-to-image transformations. In order to determine the nature of these images, we introduce a machine-learning pipeline, which relies on a classical frequency analysis, followed by a traditional machine-learning classifier. The core of our approach is the significant information that high-frequency components implicitly contain, since they allow us, most of the time, to unequivocally differentiate real and artificial images. To that end, we exploit such differences/artefacts by analysing and classifying the frequency components with supervised and unsupervised classifiers. Note that the proposed pipeline does not involve nor require vast quantities of data, making it a very convenient approach for those scenarios that suffer from data scarcity. Furthermore, we study the relationship between spectral properties of generated images and their up-sampling operations, linking in this way the artefacts with the up-convolutional units. In addition, we provide a new dataset, called Faces-HQ, which we use to complement the CelebA and FaceForensics++ datasets, for our experimental evaluation. Overall, our contributions are summarized as follows:

- We introduce a novel classification pipeline for artificial-face detection based on the frequency components.
- Our theoretical analysis and further experiments reveal that commonly used up-sampling technique, i.e., up-convolutions, might cause the observed artefacts.
- We present a new dataset of images (Faces-HQ) with high-quality real and fake faces from a collection of different public databases.

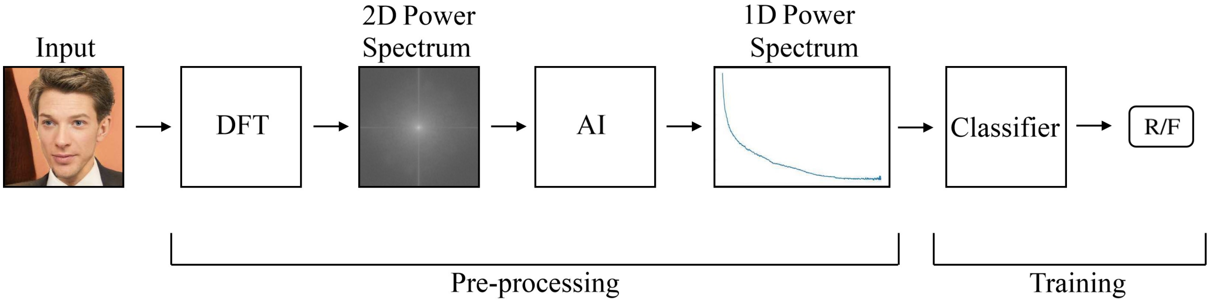


Figure 5.3. Pipeline of our forgery-detector approach. Given an input image, the pre-processing block computes the DFT and AI, and outputs the frequency features that the classifier uses to determine whether the face is real (R) or fake (F). Notice that input images are grey-scaled.

- We exploit the spectral distortions to achieve highly accurate detection rates that reach up to 100% accuracy on public benchmarks.

5.3 Method

We start this section by introducing the problem definition. Then, we provide a detailed presentation of the architecture of our model that is split into two parts. After that, we analyse the spectral effects of up-convolutions, from a theoretical perspective. Finally, we close this section with the description of our end-to-end training strategy.

5.3.1 Problem Definition

Our proposed system addresses the task of detecting artificial faces, by exploiting the artefacts on certain frequency components. Given a synthetic image $x \in \mathbb{R}^{H\times W\times 3}$ and real image $y \in \mathbb{R}^{H\times W\times 3}$, our goal is to train a model that can accurately classify them according to their nature, i.e., fake or real. Note that H and W are the height and width of the data, respectively.

5.3.2 Model Architecture

The pipeline of our proposal is depicted in Figure 5.3. It is composed of a frequency domain transformation (pre-processing part), followed by a basic classification method (training part). By combining both blocks, our approach successfully detects modified and fully synthetic images.

Frequency Domain Analysis. The analysis is of utmost importance in signal processing theory and applications. In particular, in the computer vision domain, the repetitive nature or the frequency characteristics of images can be analysed on a space defined by the Fourier trans-

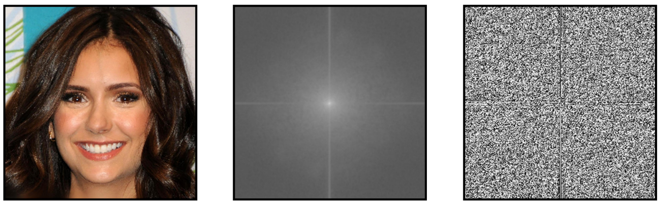


Figure 5.4. Example of a DFT applied to a sample. (Left) Input image. (Centre) Power spectrum. (Right) Phase spectrum. Notice that we convert the input image to greyscale before applying DFT.

form. Such a transformation consists of a spectral decomposition of the input data, indicating how the signal's energy is distributed over a range of frequencies. Methods based on frequency domain analysis have shown wide applications in image processing, such as image analysis, image filtering, image reconstruction and image compression.

The discrete Fourier transform (DFT) is the discrete analogue of the (continuous) Fourier transform for signals sampled on equidistant points. More specifically, the DFT ($\mathcal{F}$) is a mathematical technique to decompose a discrete signal into sinusoidal components of various frequencies, ranging from 0 (the constant frequency corresponding to the image mean value), up to the maximum representable frequency, given the spatial resolution. In our scenario, before computing the DFT, we first grey-scale the input images, $\boldsymbol{x}$ and $\boldsymbol{y}$, to have 2D image data I of size $H \times W$. Then, we can compute the DFT as

$$\mathcal{F}(I)(\ell, k) = \sum_{h=0}^{H-1} \sum_{w=0}^{W-1} e^{-2\pi i \cdot \frac{j\ell}{H}} e^{-2\pi i \cdot \frac{jk}{W}} \cdot I(h, w), \tag{5.1}$$
$$\text{for} \quad \ell = 0, \dots, H-1, \quad k = 0, \dots, W-1.$$

The frequency-domain representation of a signal $\mathcal{F}(I)$ carries information about the signal's amplitude and phase at each frequency. Figure 5.4 depicts the complex output information, i.e., power and phase. Note that the amplitude spectrum is the square root of the power spectrum.

After applying a DFT to an input sample, the information is represented in a new domain but within the same dimensionality. In our case, as we work with images, the output transformations still contain 2D information. In order to analyse the effects on spectral distributions, we count on a simple but characteristic 1D representation of the Fourier power spectrum, called azimuthal integration (AI). It can be seen as a compression, gathering and averaging of similar frequency components into a feature vector. In this way, we can reduce the amount of feature points without losing relevant information. We compute the azimuthal integration over radial frequencies ϕ of $\mathcal{F}(I)$ as

$$\mathrm{AI}(\omega_k) = \int_0^{2\pi} \|\mathcal{F}(I)\left(\omega_k \cdot \cos(\phi), \omega_k \cdot \sin(\phi)\right)\|^2 \mathrm{d}\phi \tag{5.2}$$
$$\text{for} \quad k = 0, \dots, W/2 - 1\ ,$$

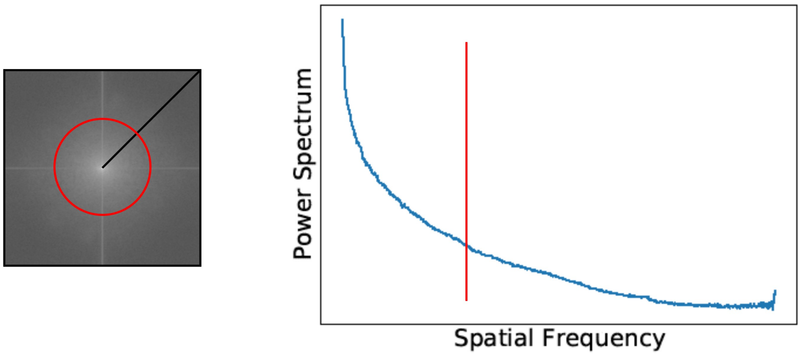


Figure 5.5. From an input image we calculate its 2D power spectrum (left), and its azimuthal integration (right), also known as 1D power spectrum. Each frequency component is the radial integral over the 2D spectrum. Red lines depict an example.

assuming square images ($H = W$). Figure 5.5 gives a schematic impression of this processing step.

Classifying AI Components. Once we have computed the AI on the input images, we can start their classification. One of the technically simplest (linear) classification algorithms is the logistic regression. This is a statistical model that employs a logistic function to characterize a binary dependent variable. Given an input $z \in R^n$, the output from the hypothesis $h_{\boldsymbol{\theta}}(z)$ is the estimated probability, which is used to infer how reliable a predicted value can be. Logistic regression is formulated as

$$h_{\boldsymbol{\theta}}(z) = \frac{1}{1 + e^{-\boldsymbol{\theta}^T z}}. \tag{5.3}$$

Via maximum likelihood estimation, the regression coefficients $\boldsymbol{\theta}$ can be determined that best fit the probability model of our classification problem. The algorithm stops when the convergence criterion is met or the maximum number of iterations is reached.

Support vector machines (SVMs) [132, 133] are among the most widely employed learning algorithms for (non-linear) data classification. The goal of the SVM formulation is to produce a model, based on the training data, that identifies an optimal separating hyperplane. In other words, given a training set of instance-label pairs $(\boldsymbol{z}, \boldsymbol{t})$ where $z \in R^n$ and $t \in \{+1, -1\}$, the SVM maps the instances $\boldsymbol{z}$ into a higher dimensional space via the function ϕ. By doing so, the algorithm tries to find an optimal linear separating hyperplane, with the maximal margin, that correctly assigns the class labels $\boldsymbol{t}$. This can be achieved by minimizing the following optimization problem

$$\begin{aligned} \min_{\boldsymbol{\theta}, b, \boldsymbol{\xi}} \quad & \frac{1}{2}||\boldsymbol{\theta}||^2 + C\sum_{i=0}^{l-1} \xi^{(i)}, \\ \text{s.t.} \quad & t^{(i)}(\boldsymbol{\theta}^T \phi(z^{(i)}) + b) \geq 1 - \xi^{(i)}, \\ & \xi^{(i)} \geq 0, \end{aligned} \tag{5.4}$$

where $\boldsymbol{\theta}$ and b are the parameters of the classifier, $\boldsymbol{\xi}$ the slack variable, $C > 0$ the penalty parameter of the error term, and l is the number of samples.

While supervised classification algorithms like logistic regression and SVM rely on labelled training examples to learn a classification, detecting tampering is also desired in the absence of annotated data; k-means is a suitable candidate for this. The algorithm k-means is an unsupervised clustering technique, which, given a training set $\boldsymbol{x}$, finds similarities among the elements in order to group them together, by minimizing the within-cluster sum of squares, i.e., the variance. A common approach to heuristically approximate a solution consists of three steps: 1) initialize cluster centroids $\boldsymbol{\mu} \in R^n$, 2) assign each training example x to the closest cluster centroid μ, and 3) move each cluster centroid μ to the mean of the points assigned to it. Then, steps 2 and 3 are iteratively repeated until the convergence criterion is fulfilled. Formally, the k-means objective function is to find

$$\arg\min_{j} \sum_{j=0}^{k-1} \sum_{i=0}^{l-1} ||x^{(i)} - \mu^{(j)}||^2, \tag{5.5}$$

where the hyperparameter k is usually set to be equal to the number of classes from the classification problem, and l is the number of samples.

5.3.3 Theoretical Analysis

The frequency-domain analysis allows us to study the spectral effects of up-convolutions in deep neural networks. This can come in particularly handy to study generative neural architectures like GANs, as they produce high-dimensional outputs, e.g., images, from very low-dimensional latent spaces, using some kinds of up-scaling mechanisms while propagating data through their networks. The two most commonly used up-scaling techniques in literature and in popular frameworks, like TensorFlow [134] and PyTorch [135], are up-convolution by interpolation (*up+conv*) and transposed convolution (*transconv*). Figure 5.6 illustrates the two methods.

For the theoretical analysis, we consider, without loss of generality, the case of a one-dimensional signal a and its discrete Fourier transform $\hat{a}$

$$\hat{a}_k = \sum_{j=0}^{N-1} e^{-2\pi i \cdot \frac{jk}{N}} \cdot a_j \tag{5.6}$$

$$\text{for} \quad k = 0, \ldots, N-1.$$

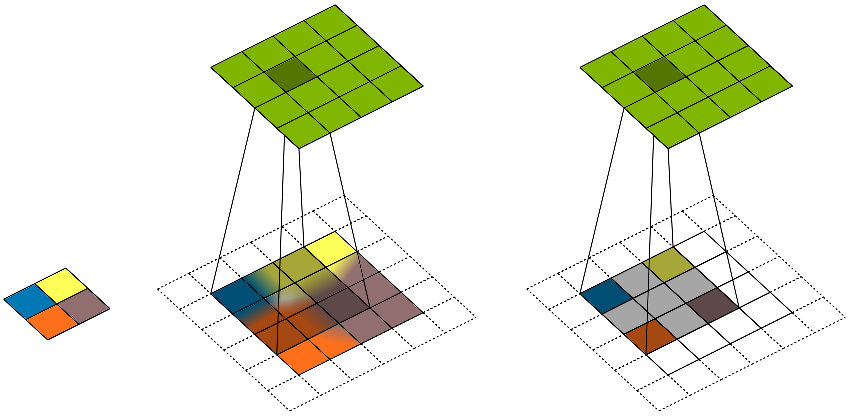


Figure 5.6. Schematic overview of the two most common up-convolutional units. (Left) Low-resolution input image (here 2×2). (Centre) Up-convolution by interpolation (*up+conv*): the input is scaled via interpolation (bilinear or nearest neighbour), and then convolved with a standard learnable filter kernel of size 3×3 to form the 4×4 output (green). (Right) Transposed convolution (*transconv*): the input is padded with a "bed of nails" scheme, and then convolved with a standard filter kernel to form the 4×4 output (green).

If we want to increase a's spatial resolution by factor 2, $\hat{a}^{up}$, we get

$$\begin{aligned}\hat{a}^{up}_{\bar{k}} &= \sum_{j=0}^{2\cdot N-1} e^{-2\pi i\cdot\frac{j\bar{k}}{2\cdot N}} \cdot a^{up}_j \\ &= \sum_{j=0}^{N-1} e^{-2\pi i\cdot\frac{2\cdot j\bar{k}}{2\cdot N}} \cdot a_j + \sum_{j=0}^{N-1} e^{-2\pi i\cdot\frac{2\cdot(j+1)\bar{k}}{2\cdot N}} \cdot b_j \\ &\text{for}\quad \bar{k} = 0,\ldots,2N-1,\end{aligned} \tag{5.7}$$

where $b_j = 0$ for "bed of nails" interpolation, as used by *transconv*, and $b_j = \frac{a_{j-1}+a_j}{2}$ for bilinear interpolation, as used by *up+conv*.

Let us first consider the case of $b_j = 0$, "bed of nails" interpolation. For this case, the second term in 5.7 is zero. The first term is similar to the original Fourier transform, yet with the parameter k being replaced by $\bar{k}$. Thus, the spatial resolution is increased by a factor of 2, leading to a scaling of the frequency axes by a factor of $\frac{1}{2}$. Let us now consider the effect from a sampling theory based viewpoint. That is

$$\hat{a}^{up}_{\bar{k}} = \sum_{j=0}^{2\cdot N-1} e^{-2\pi i\cdot\frac{j\bar{k}}{2\cdot N}} \cdot a^{up}_j \tag{5.8}$$

$$= \sum_{j=0}^{2\cdot N-1} e^{-2\pi i\cdot\frac{j\bar{k}}{2\cdot N}} \cdot \sum_{t=-\infty}^{\infty} a^{up}_j \cdot \delta(j-2t), \tag{5.9}$$

since the point-wise multiplication with the Dirac impulse comb only removes values for which

$a^{up} = 0$. Assuming a periodic signal and applying the convolution theorem [136], we get

$$(5.9) = \frac{1}{2} \cdot \sum_{t=-\infty}^{\infty} \left(\sum_{j=-\infty}^{\infty} e^{-2\pi i \cdot \frac{j\bar{k}}{2 \cdot N}} a_j^{up} \right) \left(\bar{k} - \frac{t}{2} \right) , \tag{5.10}$$

which is equal to

$$\frac{1}{2} \cdot \sum_{t=-\infty}^{\infty} \left(\sum_{j=-\infty}^{\infty} e^{-2\pi i \cdot \frac{j\bar{k}}{N}} \cdot a_j \right) \left(\bar{k} - \frac{t}{2} \right) \tag{5.11}$$

by Equation 5.7. Thus, the "bed of nails" up-sampling will create a high-frequency replica of the signal in $\hat{a}^{up}$. More precisely, all observed spatial frequencies beyond $\frac{N}{2}$ are potential upsampling artefacts, which can only be removed if the up-sampled signal is smoothed appropriately.

In the case of bilinear interpolation, we have $b_j = \frac{a_{j-1}+a_j}{2}$ in Equation 5.7, which corresponds to an average filtering of the values of a adjacent to b_j. This is equivalent to a point-wise multiplication of a^{up} spectrum $\hat{a}^{up}$ with a sinc function by their duality and the convolution theorem, which suppresses artificial high frequencies. Yet, the resulting spectrum is expected to be overly low in the high-frequency domain.

5.3.4 End-to-End Model Architecture

To successfully combine the frequency features with the classifier, we design a sequential pipeline. First, we apply the discrete Fourier transform, then the azimuthal integration, and finally, we train a detector using these extracted features. Algorithm 6 describes the pseudocode of our proposal.

Algorithm 6 Training of the detection model using the logistic regression classifier.

```
Require: $n_{\text{iter}}$, number of iterations. $\alpha$, learning rate. $m$, batch size.
Require: $\boldsymbol{\theta}$, initial parameters.
for $i < n_{\text{iter}}$ do
    Sample a batch of synthetic images $\{x^{(j)}\}_{j=0}^{m}$
    Sample a batch of real images $\{y^{(j)}\}_{j=0}^{m}$
    Create input set and greyscale it
    $S = (\boldsymbol{x}, \boldsymbol{y})$
    Apply DFT and AI transformations
    $\boldsymbol{z} = \text{AI}(\mathcal{F}(S))$
    Train logistic regression classifier
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} h_{\boldsymbol{\theta}}(\boldsymbol{z})$
end for
```

5.4 Experiments

In this section, we evaluate the forgery detection pipeline in three scenarios, where the altered input faces have different natures, i.e., partially modified, fully generated and image-to-image transformations. To that end, we first give a detailed description of the experimental setup. Then, we comprehensively analyse the distortions of the power spectrum caused by the upsampling techniques, as explained in the theoretical analysis. Finally, we assess the classification capability of our proposal.

5.4.1 Experimental Setup

We train our face detector with the FaceForensics++ [137], CelebA [53] and Faces-HQ datasets. FaceForensics++ contains a deepfake collection with 363 original video sequences in 16 different scenes, and over 3,000 video sequences with partial face manipulations and their corresponding binary masks. All videos contain a trackable, mostly frontal face without occlusions, which enables automated tampering methods to generate realistic forgeries. The dimensions of the extracted face images may vary, but they are around 80×80×3 pixels (low-resolution). CelebA consists of 202,599 celebrity face images with different facial attributes. The dimensions of the images are 178×218×3, which can be considered a medium-resolution in our context. Finally, to the best of our knowledge, no public dataset currently provides high-resolution images with annotated fake and real faces. Therefore, we create our own face dataset, called Faces-HQ. In order to have a sufficient variety of faces, we download and label the images available from the CelebA-HQ [104] dataset, Flickr-Faces-HQ [9] dataset, 100K Faces project [138] and *www.thispersondoesnotexist.com*. In total, we collect 40,000 high-quality face images, half of which real and the other half fake. The dimensions of the images are 1024×1024×3 pixels. Table 5.1 contains the summary. All experiments presented are conducted on a single NVIDIA GeForce GTX 1080 GPU.

Dataset	# samples	Category	Label
CelebA-HQ [104]	10,000	Real	0
Flickr-Faces-HQ [9]	10,000	Real	0
100K Faces project [138]	10,000	Fake	1
www.thispersondoesnotexist.com	10,000	Fake	1

Table 5.1. Inner structure of Faces-HQ dataset. It consists of a balanced collection of real and fake high-resolution face images, with a total amount of 40,000 samples.

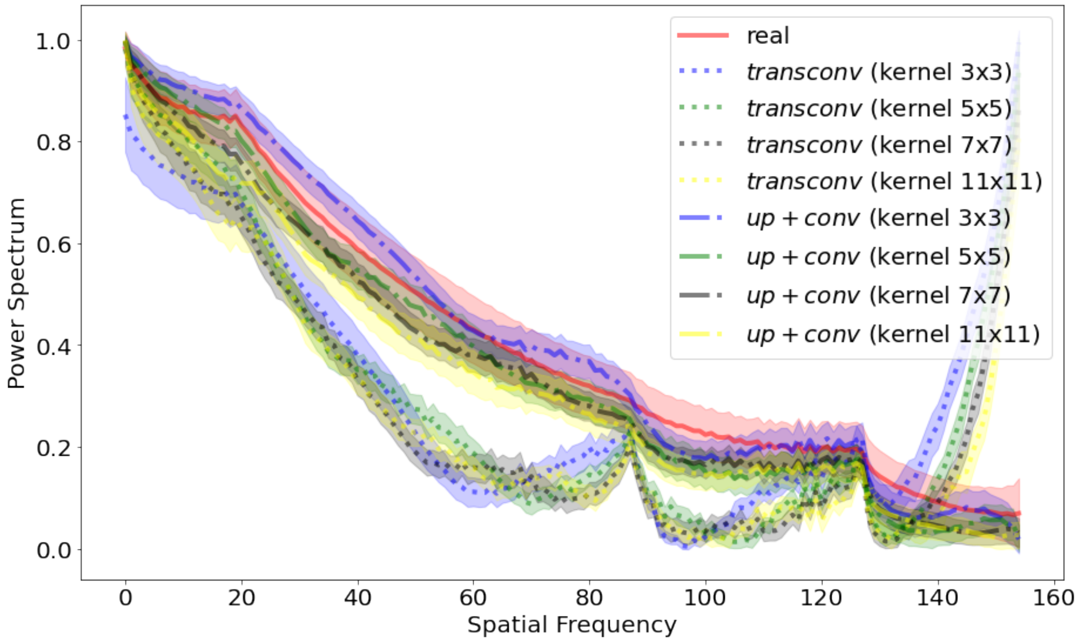


Figure 5.7. Azimuthal integration statistics (mean and variance) from different up-sampling techniques and filter sizes. All scenarios have important effects on the spectral distributions of the outputs. Transposed convolutions (*transconv*) add large amounts of high-frequency noise, while interpolation-based methods (*up+conv*) are lacking high frequencies.

5.4.2 Analysis of Spectral Effects of Up-Convolutions

We use a very simple autoencoder setup on the CelebA dataset for an initial investigation of spectral properties of images after undergoing up-sampling operations. It consists in down-scaling the input image by a factor of 2, and then, using the different up-convolution methods to reconstruct the original image size. This simple, yet representative scheme allows us to study the differences between the two most popular up-sampling techniques: up-convolution by interpolation (*up+conv*) and transposed convolution (*transconv*).

We evaluate the frequency spectrum for both up-sampling techniques, and the impact of the filter size. In particular, we consider varying the decoder filter size from 3×3 to 11×11. As we explained in the theoretical analysis, *transconv* units create high-frequency replicas due to their "bed of nails" up-sampling approach; this effect is present and visible in our autoencoder setup, see Figure 5.7. Additionally, we can see the impact of different filter sizes, although none can successfully remove the spectral distortions. On the other hand, when using *up+conv* units, the spectrum properties are also altered, however they are more similar to the real frequency components, resulting in better performance than *transconv* units. Finally, Figure 5.8 provides some qualitative result, where the artefacts in the frequency spectrum are shown to be relevant for the visual appearance.

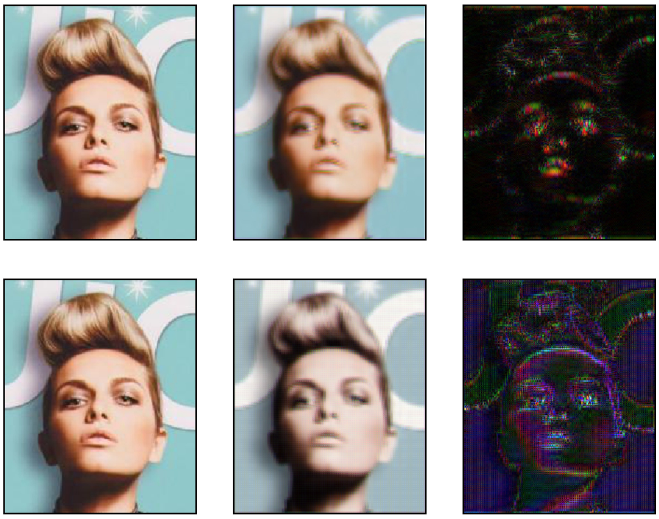
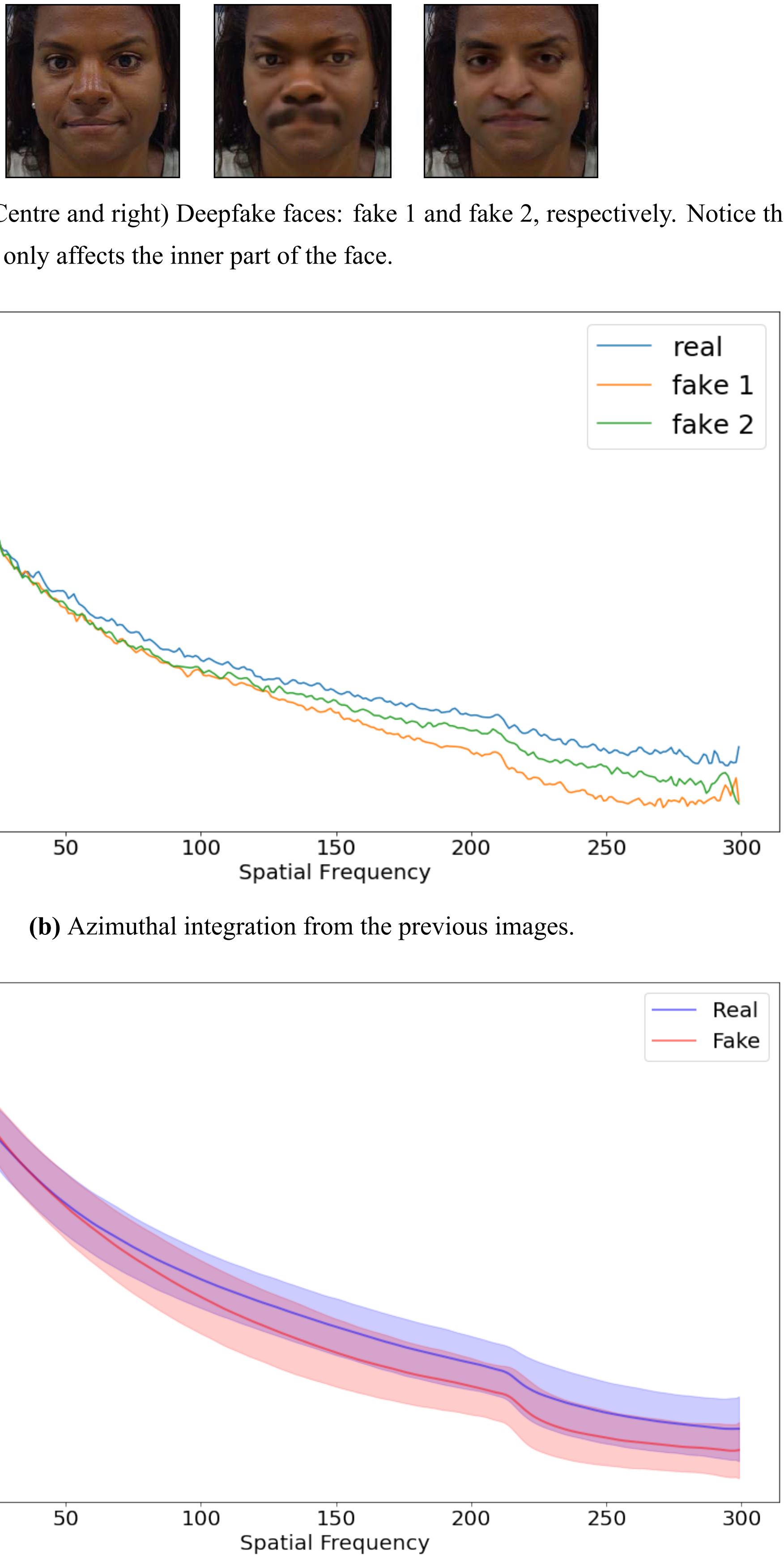


Figure 5.8. Examples of the effects of the spectral distortions on the output images of the autoencoder setup. (Left) Input images. (Centre) Output images. (Right) Filtered difference images. (Top) Blurring effect caused by the missing high frequencies in *up+conv* units. (Bottom) High-frequency artefacts induced by *transconv* units.

5.4.3 Results of Deepfake Detection

Since FaceForensics++ is composed of videos, we first need to extract the frames, and then crop the inner parts of the faces so that we can use them as input images. These inputs contain partial face modifications (deepfakes). Note that due to the different content of the scenes of the videos, these cropped images may have slightly different sizes. While the pre-processing part from the proposed pipeline is size-independent, i.e., no changes in size are required, the classifier is not independent, and hence, it expects a fixed amount of features. Therefore, we add an extra processing step, just before the classifier and after the pre-processing, that interpolates the azimuthal integration to a fixed size (300 features), and normalizes it by dividing it by the 0^{th} frequency coefficient. Once the interpolation step is finished, we start training the classifier engine. To do so, first of all, we divide the interpolated data, where 80% is for the training stage and the remaining 20% is for the testing. After that, we train the classifier, and assess its accuracy on real and deepfake faces.

Figure 5.9 illustrates the comparison between real and deepfake images. While it is futile to look at the image domain to spot the deepfakes (see Figure 5.9a), if we look at the frequency domain, it is not (see Figure 5.9b). However, the frequency analysis is not always sufficient. Figure 5.9c shows how the azimuthal integrations from the real and deepfake images partially overlap with each other, meaning that some samples will be misclassified. To try to overcome this issue, it is recommendable to employ more samples for training, as they have a direct impact on the detection performance. In fact, Table 5.2 depicts how the classification accuracy is bound to the number of training samples. These results support frequency components' potential as



(a) (Left) Real face. (Centre and right) Deepfake faces: fake 1 and fake 2, respectively. Notice that the deepfake modification only affects the inner part of the face.

(b) Azimuthal integration from the previous images.

(c) Azimuthal integration statistics (mean and variance) from FaceForensics++ dataset.

Figure 5.9. Example of cropped samples from the FaceForensics++ dataset and their corresponding azimuthal integration curves.

key features for forgery detection, independently of the classifier, since both supervised methods (logistic regression and SVM) and the unsupervised k-means achieve high scores. On the other hand, it is also true that none of the approaches reach a perfect score. The main reason for this is that the input images are low-resolution and only partially modified. Consequently, the frequency analysis is limited, i.e., these images' frequency components are not rich enough to distinguish unequivocally between real and deepfake content.

Finally, in order to further improve the accuracy scores, we employ an additional technique that consists in averaging the classification rate per video, applying a simple majority vote over the single frame classifications. As a result, we boost the accuracy scores by 3% on both supervised methods.

# samples	logistic reg.	SVM	k-means
200	73%	77%	65%
1,000	76%	82%	67%
2,000	78%	87%	67%
2,000*	81%	90%	-

Table 5.2. Test accuracy for forgery detection on the FaceForensics++ dataset. We evaluate the results on logistic regressions, SVM and k-means, under different data settings. *Accuracy on full video sequences via majority vote of single frame detections.

# samples	logistic reg.	SVM	k-means
200	100%	100%	81%
1,000	100%	100%	82%
2,000	100%	100%	82%

Table 5.3. Test accuracy for forgery detection on the Face-HQ dataset. We assess the results on logistic regressions, SVM and k-means, under different data settings.

5.4.4 Results of Synthetic Data Detection

In this subsection, we show that the spectral distortions in state-of-the-art GANs that are caused by the up-convolutions can be employed to identify fully synthetic data. Using only a small amount of annotated training data, or even none at all, our approach is able to detect generated faces with almost perfect accuracy.

High-Resolution Synthetics. For the high-resolution synthetic data detection experiment, we use the Face-HQ dataset. We start transforming every sample from the spatial domain to the azimuthal integration domain, i.e., reducing high-quality 1024×1024×3 colour images to their 722 features (1D power spectrum). Unlike the deepfake detection scenario, now there is no interpolation step since all the data samples have the same dimensions. Therefore, the preprocessing step involves only the discrete Fourier transform, followed by azimuthal integration. Regarding the classifier engine, we stick to the same approaches, i.e., logistic regression, SVM and k-means.

In Figure 5.1 we can observe a pattern on the azimuthal integration for real images, and a different one for generated images. This clear difference in the spectrum properties leads to perfect accuracy when using supervised approaches, and significantly high accuracy when using the unsupervised. Table 5.3 contains the classification scores when we change the quantity of samples. Accuracy results are stable on all the setups, and the number of samples does not play an important role anymore.

Medium-Resolution Synthetics. In this second experiment of synthetics forgery detection, we follow the same procedure as for high-resolution, but this time we work with medium-resolution. More specifically, we employ 128×128×3 images (88 features), and generate our own synthetic data by training a few GAN models. In this way, we can have a better insight into which frequencies these generative models fail to mimic from the input data distribution. In particular, we evaluate the following well-known GANs: DCGAN [139], DRAGAN [140], LSGAN [141], and WGAN-GP [51]. For each of these approaches, we run an individual training process using the CelebA dataset.

Similar to the high-resolution case, the effects of up-convolutional units are also quite strong on the medium-resolution setup. Therefore, as one might expect, the classification scores for different amounts of samples reflect the same tendency, achieving 100% accuracy for the supervised setting, and 93% for the unsupervised one (see Table 5.4). Finally, Figure 5.10 shows how straight-forward the differentiation is between real and medium-resolution generated data, when looking at the azimuthal integrations.

# samples	logistic reg.	SVM	k-means
200	100%	100%	91%
1,000	100%	100%	93%
2,000	100%	100%	93%

Table 5.4. Test accuracy for forgery detection on GAN-generated data and the CelebA dataset. We assess the results of logistic regressions, SVM and k-means, under different data settings.

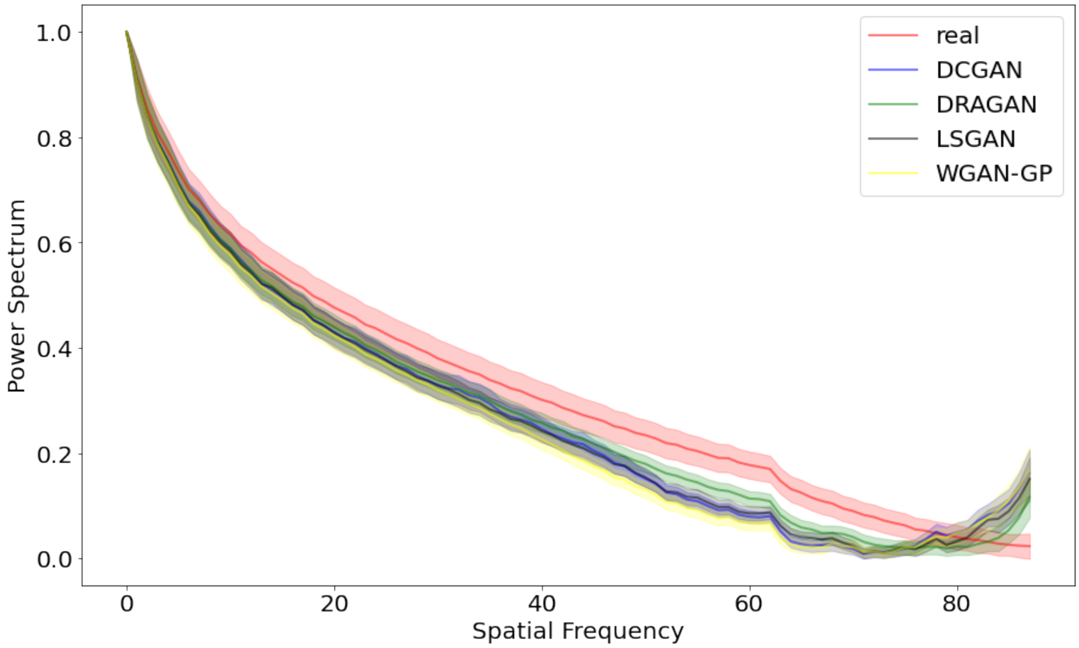


Figure 5.10. Azimuthal integration statistics (mean and variance) from the CelebA dataset and GAN generated images. Common up-convolution methods induce significant spectral distortions into generated images.

5.4.5 Results of Domain Transfer Data Detection

Last but not least, we assess the performance of our detection pipeline for images that have undergone an image-to-image transformation, either style or attribute or both transformations. To that end, we work with the outputs from Semi Few-shotGAN [112], ATI-GAN [87], and FacialGAN [113], and their corresponding source input images. Table 5.5 contains the results of the tampering data detection for the aforementioned models, by different classifiers.

	# samples	logistic reg.	SVM	*k*-means
Semi Few-shotGAN [112]	200	93%	93%	83%
ATI-GAN [87]	200	83%	81%	75%
FacialGAN [113]	200	100%	100%	96%

Table 5.5. Test accuracy for forgery detection on three image-to-image translation approaches: Semi Few-shotGAN, ATI-GAN, and FacialGAN. We assess the results on logistic regressions, SVM, and *k*-means under different data settings.

Even though these domain-transfer approaches do not have to synthesize images from scratch, e.g., from random noise, they still cannot match the frequency components of real data accurately enough. As a result, a cursory investigation of their azimuthal integration reveals the synthetic nature of these domain-transform images. However, there are noticeable differences in their detection accuracy. For example, if we look at ATI-GAN, we see that it has the lowest accuracy results, therefore its frequencies are closer to the real ones. One reason for that happening might be that the contour of the images remains mostly unmodified, allowing frequencies from the real images to be retained. Regarding the Semi Few-shotGAN setup, the accuracy is quite high, yet not perfect. In this case, we hypothesize that the model is able to partially mimic some frequency components, since its main task is rather simple, i.e., it does not involve morphological changes, but it is still failing at correctly reproducing all the components. Finally, FacialGAN gets almost perfect scores for all the classifiers, making it the worst approach in terms of frequency reproducibility. This can be explained by the following: 1) it synthesizes high-resolution images, and 2) it applies considerable morphological changes. As a consequence, FacialGAN has a more complex generation process, leading to larger artefacts in its spectrum.

5.5 Limitations

Our tampering detector exploits the frequency anomalies/artefacts that synthetic data suffers from. While this analysis of frequency components allows cheap and powerful features to be computed, reaching almost perfect accuracy, it is not always sufficient. Furthermore, it can be easily neutralized by counter-forensic methods. For example, if an image is compressed, it is likely that its frequency content has been altered, resulting in a non-standard spectrum. Another case when frequencies are modified, is when there are some sorts of image processing, such as image filtering or pixel averaging. In Table 5.6, we assess the robustness of our detector when applying image filtering. More specifically, we compute the average of the pixel values inside a 5×5 moving window, and we set the result at the centre pixel. As we can see, all the accuracy scores show a clear deterioration, regardless of the input images and classifiers. Finally, another solution to deceive our detector is to incorporate frequency information when training the generative models [142, 143]. In this manner, the generator is aware of this constraint, and can correct the output spectrum so as not to differ from the real one.

5.6 Summary

In this chapter, we describe and evaluate the accuracy of our method to detect partially modified, fully generated, and image-to-image transformation images. To do so, we exploit the information from the frequency domain. Additionally, we theoretically justify why synthetic content might contain frequency artefacts when up-convolutional units are applied. Experimental results demonstrate the robustness of our pipeline, providing consistent scores when facing limited training data and different classifier engine. As for future work, we believe that there

	# samples	logistic reg.	SVM	k-means
FaceForensics++ [137]	200	93% | 59%	93% | 69%	83% | 58%
Face-HQ	200	83% | 50%	81% | 50%	75% | 50%
CelebA [53]	200	100% | 81%	100% | 96%	91% | 66%
Semi Few-shotGAN [112]	200	93% | 59%	93% | 69%	83% | 58%
ATI-GAN [87]	200	83% | 50%	81% | 50%	75% | 50%
FacialGAN [113]	200	100% | 68%	100% | 75%	96% | 66%

Table 5.6. Test accuracy for forgery detection when image filtering is applied. We assess the results on all the previous experimental setups. Each classifier column consists of (left) the standard accuracy, and (right) the altered accuracy. Note that the altered accuracy refers to those experiments that have applied the image filtering before the discrete Fourier transform.

are many possibilities to further improve our approach to make it more robust against counter-forensic attacks. One of which could be to try to find a frequency fingerprint that cannot be easily altered when compressing or filtering the images. In this way, we would keep using few samples while providing reliable detection.

Chapter 6

Style Transfer for Seismic Applications

Interpreting seismic data requires characterizing a number of key elements, e.g., the position of the faults and main reflections, the presence of structural bodies, such as salt, and, the clustering of areas exhibiting a similar amplitude-versus-angle response. Manual interpretation of geophysical data is often a difficult and time-consuming task, complicated by the lack of resolution and the presence of noise. Additionally, it usually requires field expertise to understand the acquisition and processing workflows, as well as general knowledge about the geology of the Earth's subsurface. Traditional seismic tasks, such as interpreting horizons and faults, or delineating channels or salt domes involve detecting the amplitudes in large seismic volumes. When performed manually, this can be a tedious process, requiring the efforts of skilled interpreters for a large period of time. For all these reasons, a number of algorithms have been developed to partially automate this process [144–149]. As an example, most interpretation platforms offer horizon auto-tracking tools that use the sparse hints provided by the interpreter to follow a chosen reflection across an entire volume. Unfortunately, writing efficient and robust algorithms that can yield accurate results in areas of poor signal-to-noise ratio is very challenging. The computation cost of those algorithms may be prohibitive, and manual editing by experts is usually required in the most difficult areas of the data [144, 150].

In recent years, approaches based on deep leaning, in particular on convolutional neural networks, have shown remarkable results in automating certain interpretative tasks [147, 152]. However, these state-of-the-art systems usually need to be trained in a supervised manner and suffer from a generalization problem. Hence, it is highly challenging to train a model that can provide accurate results on new real data, obtained with an acquisition, processing and geology that are different to those of the data used for training. Nonetheless, generalization is neither new nor exclusive to seismic tasks. In fact, it is a well-studied area [153, 154] with many still open questions that need to be tackled in order to achieve models that can guarantee generalization properties in all sorts of scenarios.

In this chapter, to address the generalization limitation, we introduce a novel method that

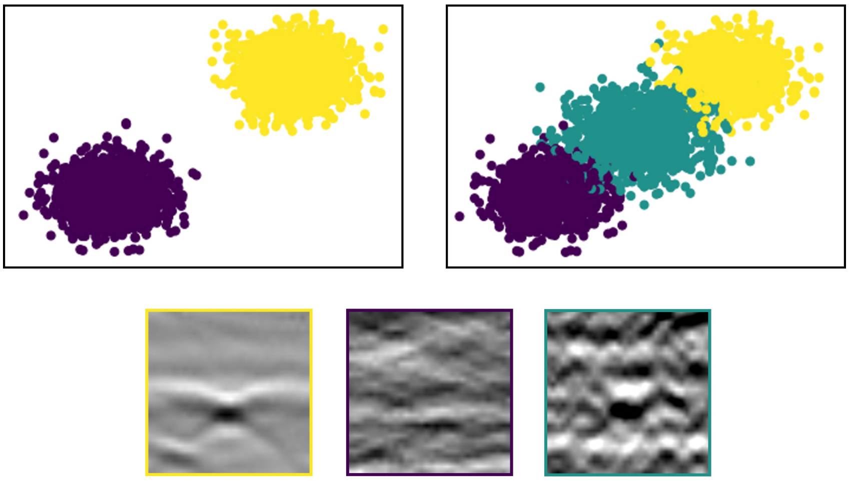


Figure 6.1. Illustrative example of the gap existing between annotated synthetic training data and uninterpreted real target data. (Top-left) Initial setup: real data (purple) and synthetic data (yellow) have different distributions. (Top-right) Durall *et al.* [151] approach, where they generate semi-synthetic samples (green) that fill in the gap between the distributions. (Bottom) Example of samples belonging to the aforementioned distributions.

employs generative neural networks to decrease the gap existing between annotated synthetic training data and uninterpreted real target data in deep-learning based methods. Figure 6.2 illustrates the generalization-gap scenario that seismic tasks might face. Inspired by Durall *et al.* [151], we explore the possibilities to create and use a more realistic dataset by exploiting the generative abilities of GANs to transfer features between different style domains. In particular, we present a model that integrates a GAN-based image-to-image translation network together with a semantic-segmentation network in an end-to-end fashion. In this way, given a set of synthetic training data, we employ the generative network to morph the data such that it better resembles the real target style domains. Simultaneously, the segmentation network learns from the modified data and eventually yields better segmentation results on the real testing data.

6.1 Background

In this section, we formally define the seismic-interpretation task, and provide an overview of the related work, paying special attention to deep-learning approaches, including GAN-based methods.

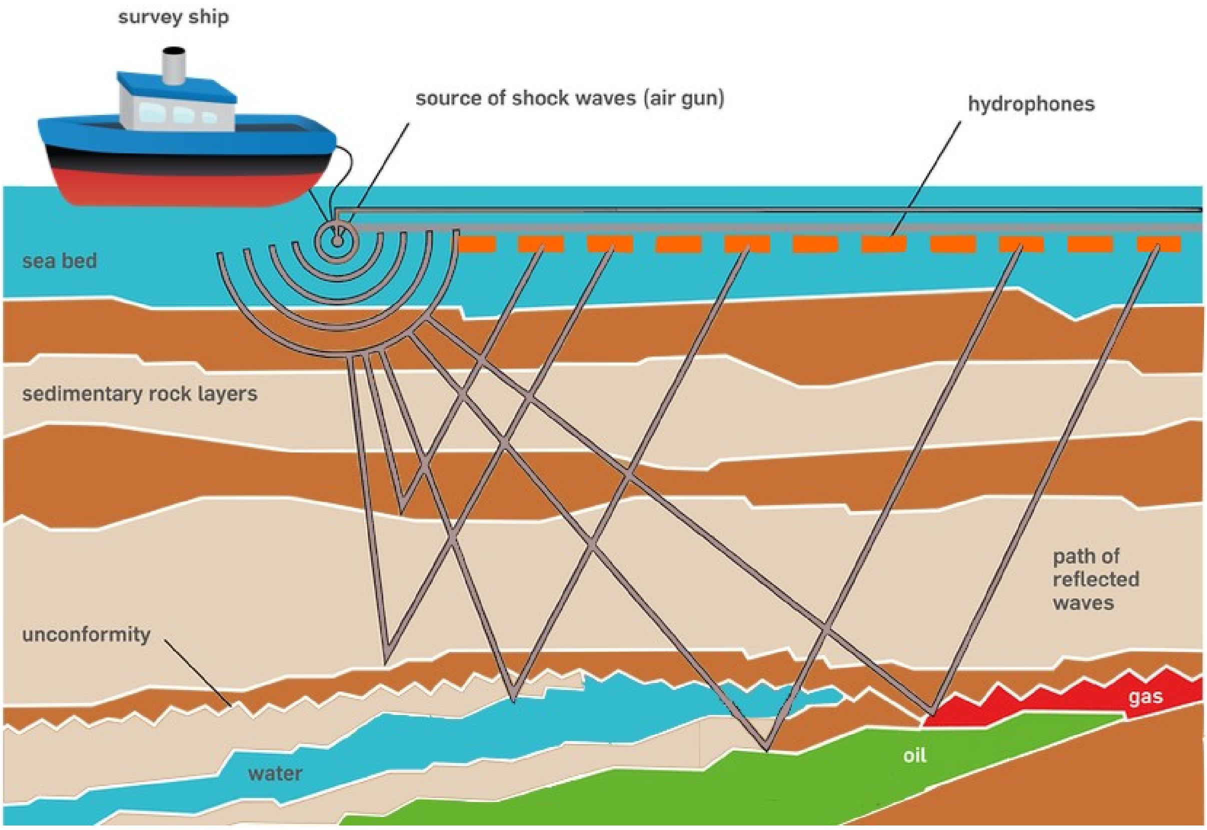


Figure 6.2. Illustrative example of marine data acquisition. The vessel tows an air gun (the seismic source) and a cable with regularly spaced hydrophones (the seismic receiver). After producing the shock waves, the reflected sound is recorded at different points away from the source, and later combined to create the underground images.

6.1.1 Preliminaries

Seismic data is the physical observation, measurement, or estimation of seismic sources, seismic waves, and their propagating media. It is the principal geophysical information used to image the subsurface of the Earth in both land and marine environments.

In the last few decades, methods using seismic data have had significant developments in the theory and practice of data acquisition and processing methodologies. This is due to the fundamental role that they currently play in oil and gas exploration, and consequently, exploitation. The aim of seismic data acquisition and processing, also known as seismic exploration, is to learn about the Earth's interior. To that end, it is necessary to identify some specific intersections between the intended targets and the measurable parameters. This process of discovery consists of three distinct stages: 1) data acquisition to gather and record continuous seismic signals from seismic stations, 2) data processing to identify and enhance the desired signal, and 3) data interpretations to make predictions based on the processed data. Seismic acquisition requires the use of a seismic source at specified locations for a seismic survey. After the source has emitted sound waves, the energy of these waves travels within the subsurface of the Earth as seismic waves. These are recorded at specified locations on the surface by receivers, i.e., geophones or hydrophones. Figure 6.2 depicts an example of marine seismic acquisition. Then, before data processing, it is important to conduct the process of data quality control. This involves checking the survey geometry, data format, and consistency between different components of the dataset, as well as assuring that the quality and quantity of the dataset are satisfactory for the study objectives. After that, the treatment of the seismic data starts—commonly referred to as data processing. In order to create an accurate image of the subsurface, artefacts, outliers and noise in the data must be removed or at least minimized. Finally, data interpretation takes place. This mainly involves asking a set of questions about the data that relate to one's study goals. After the process of data interpretation, seismic modelling is often conducted, using the interpreted model and the real data geometry. This is to generate predictions to compare with the real measurements and hence further verify the interpretation.

6.1.2 Related Work

Machine learning, and especially deep learning, has been employed to replace traditional handcrafted algorithms. Deep-learning models set the new state of the art in many applications across different fields [155]. Therefore, a number of authors propose using deep learning to automate various seismic-interpretation tasks, including fault detection [146–148], salt delineation [145, 149], horizon tracking [148, 156], channel identification [157], interpolation [158, 159], detecting diffraction [160], or litho-facies classification [161]. However, despite the excellent results obtained on these tasks, there are still some issues that limit the deployment of deep-learning models for everyday tasks. Indeed, most state-of-the-art approaches for seismic interpretation employ labelled data for their ground-truth evaluations, hence, they are supervised learning methods. That is, their success strongly depends on the quality and quantity of annotated training examples. As it is well-known, preparing the training data may prove very challenging. Manual interpretation is tedious and time consuming, sometimes making the label process too expensive, or even infeasible when scaling up problems. As a result, in many cases there are only small training datasets available, restricting the models to perform poorly and fail to generalize beyond the training and validation data. Alternative approaches [160,162,163] make use of synthetic data for training so that they can have larger datasets. Despite being a promising avenue, it also has disadvantages; among them, the oversimplification of data stands out. Seismology data is characterized by its complicated structures, and non-accurate synthetic data might cause an incorrect interpretation on complex regions of field-recorded data [147,160]. To deal with this issue, work on generative deep-learning modelling has been explored, in particular, with GANs [151, 164, 165], that is, to create and use a more realistic dataset. This is done by exploiting the model's generative abilities to transfer features between different domains, including the styles.

6.2 Contributions

In this work, we introduce a novel workflow to reduce the generalization gap that exists between oversimplified synthetic training data and real application data. Rather than relying on extensive manual labelling of additional examples to improve the performance, we instead resort to a self-supervised algorithm. This transforms existing training samples so that they become more similar to the target data domain, by taking into account their style. Specifically, we build a framework that integrates an image-to-image translation network, combining a GAN's framework with a semantic-segmentation network. We validate our proposed model on two challenging problems: the detection of faults in 2D stacked data, and the detection of diffracting events in 2D prestack data. In both applications, we use wave physics to create synthetic training examples, and we show that our method improves the prediction of the network on real field data. We demostrate that, when transitioning from synthetic training data to real validation data, our workflow yields superior results compared to its counterpart without the generative network. Overall, our contributions are summarized as follows:

- We propose a novel end-to-end system for systemic applications that reduces the generalization gap between synthetic training data and real testing data.
- We integrate an image-to-image network that allows us to transfer the style domain of target images, leading to better accuracy, while removing the need for labelled real data.
- We assess the results in two real case applications: the detection of diffraction events, and the detection of faults on the F3-Netherlands dataset.

6.3 Method

We start this section by introducing the problem definition. Then, we provide a detailed presentation of the architecture of our model, which is split into two parts. After that, we explain the loss function used in our training process, including its various parts and the reason we have them. Eventually, we close this section with the description of our end-to-end training strategy.

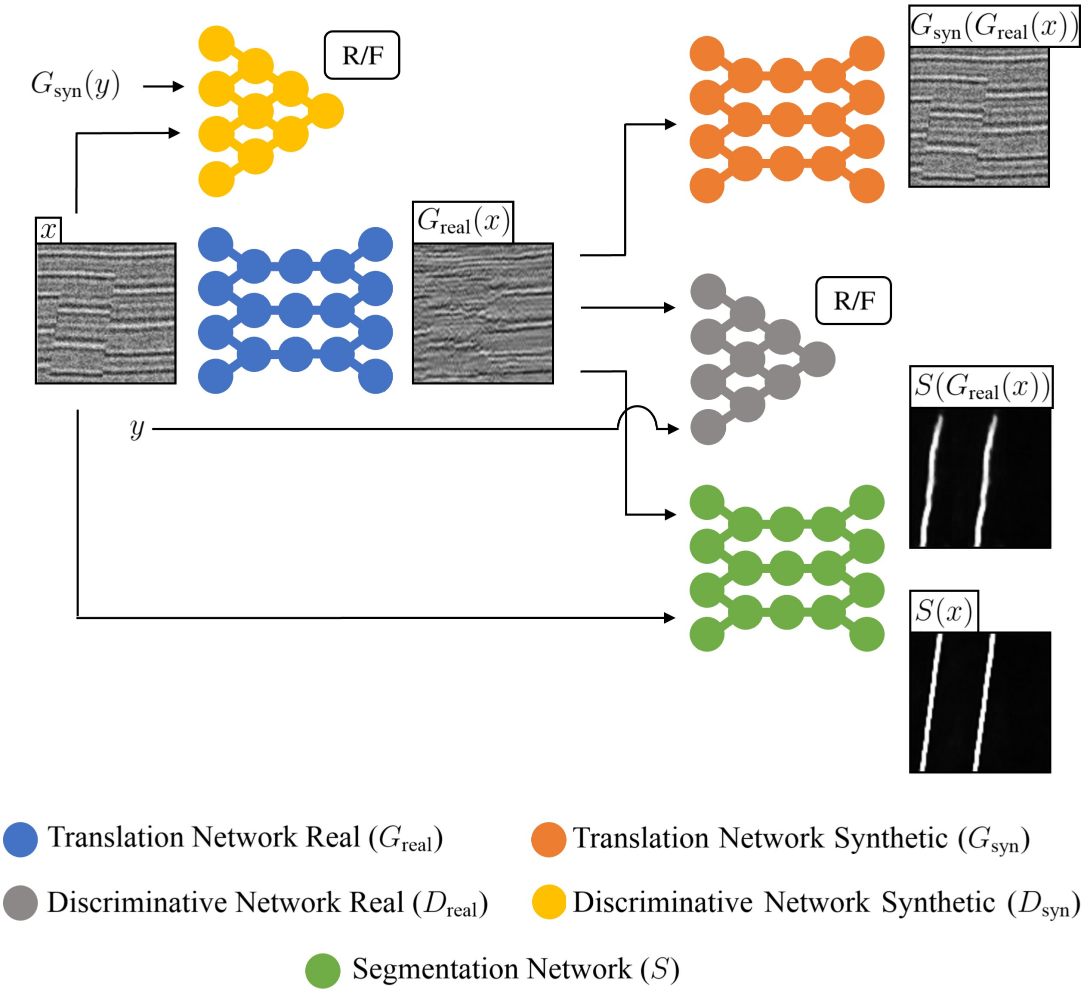

$G_{\text{syn}}(y)$
R/F
x
$G_{\text{real}}(x)$
$G_{\text{syn}}(G_{\text{real}}(x))$
R/F
$S(G_{\text{real}}(x))$
y
$S(x)$
● Translation Network Real (G_{real})
● Translation Network Synthetic (G_{syn})
● Discriminative Network Real (D_{real})
● Discriminative Network Synthetic (D_{syn})
● Segmentation Network (S)

Figure 6.3. Overview of the end-to-end pipeline for the translation of data from the synthetic to the real domain. Given a synthetic sample x, the model transforms it into a realistic sample from the real domain $G_{\text{real}}(x)$. Then this is fed into the following blocks: 1) the discriminator synthetic D_{syn} to assess the style transformation, classifying the images as real (R) or fake (F); 2) the segmentor S to improve the segmentation results; and 3) the translator synthetic G_{syn} to close the cyclic loop transformation.

6.3.1 Problem Definition

Our proposed model addresses the semantic-segmentation task by exploiting style-domain transfers on the training data. Given a synthetic image $x \in \mathbb{R}^{H \times W \times N}$, its segmentation label mask $m \in \mathbb{R}^{H \times W \times 1}$, and real reference image $y \in \mathbb{R}^{H \times W \times N}$, our goal is to train a model that can transfer the style from y to x, while being consistent with the geometry constraint of m so that the segmentation network can learn from it. Note that H, W and N are the height, width and depth of the data, respectively.

6.3.2 Model Architecture

The pipeline of our proposal is depicted in Figure 6.3. It is composed of an image-to-image translation network and a segmentation network. By combining both blocks sequentially, the ensemble model successfully transfers style domains, thereby improving the segmentation outcomes.

Segmentation Network. In seismic interpretation, when the segmentation task is solved by training CNNs, it is typically based on seismic patches, extracted from a global volume, and their corresponding interpretation. Notice that this interpretation indicates the location of the objects of interest. In this manner, the pairs of data-label examples with their pixel-wise annotations are created. To perform the semantic segmentation, we design a segmentation network S, based on the U-Net [166] architecture. Given input data samples $\boldsymbol{x}$, the outputs of the network are scaled to pseudo-probabilities $S(\boldsymbol{x}) = \boldsymbol{m'}$, using a normalized exponential function, which ideally should be equal to the ground-truth masks $\boldsymbol{m}$. Figure 6.4 depicts the scheme of the pipeline of the segmentation network.

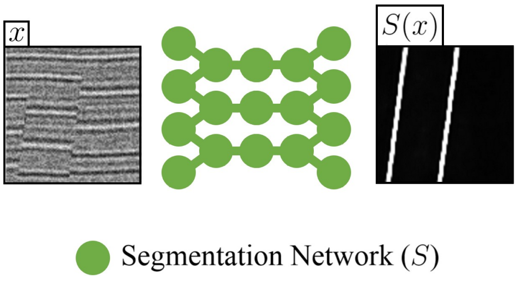


Figure 6.4. Pipeline of the segmentation structure. Given an input image x, the network links each pixel in the image to a class label, producing an output mask $S(x) = m'$. Ideally, it should be equal to the ground-truth m.

Translation Network. CycleGAN [15] is one of the most successful GAN-based approaches that exploits the intrinsic generative capacity of GANs to train an image-to-image translation

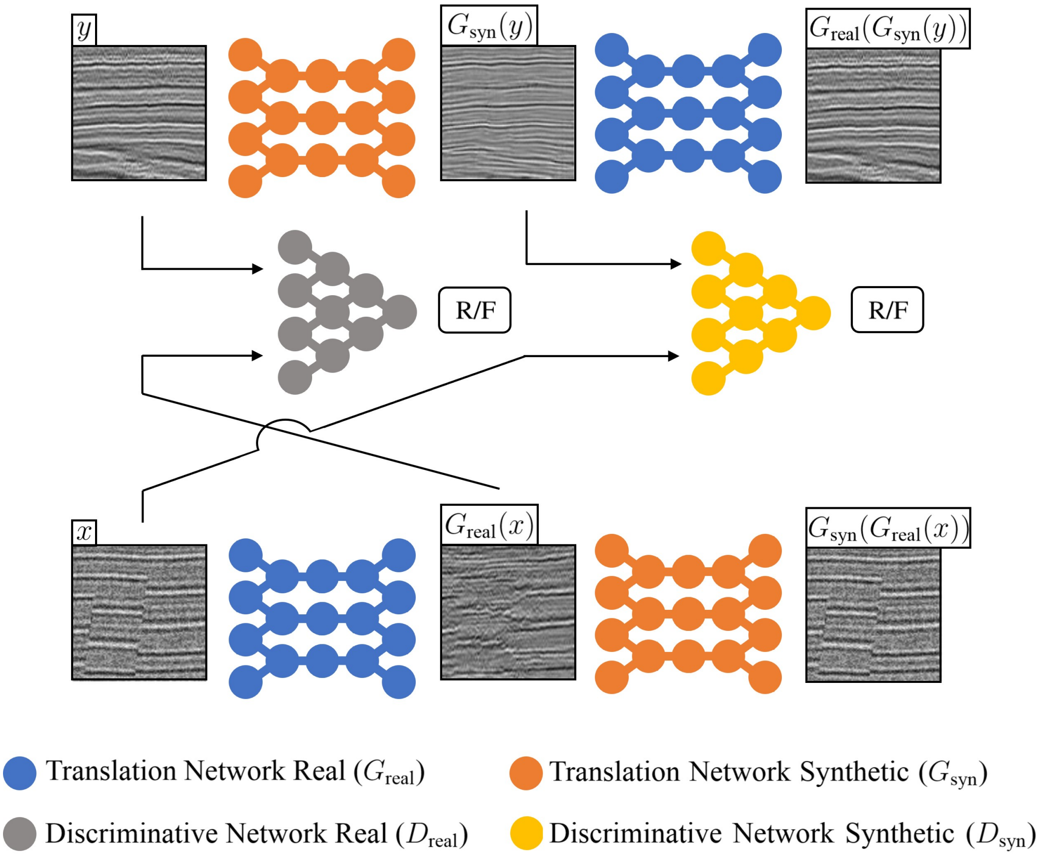

y
$G_{\text{syn}}(y)$
$G_{\text{real}}(G_{\text{syn}}(y))$
R/F
R/F
x
$G_{\text{real}}(x)$
$G_{\text{syn}}(G_{\text{real}}(x))$
Translation Network Real (G_{real})
Translation Network Synthetic (G_{syn})
Discriminative Network Real (D_{real})
Discriminative Network Synthetic (D_{syn})

Figure 6.5. Pipeline of the image-to-image translation structure. Given two images from different style domains, real y and synthetic x, the model learns to translate them between the two domains. (Top) Cyclic transformation from real y to synthetic style domain $G_{\text{syn}}(y)$, evaluated by D_{syn}. (Bottom) Cyclic transformation from synthetic x to real style domain $G_{\text{real}}(x)$, evaluated by D_{gen}. Note that we feed both transformed results, $G_{\text{syn}}(y)$ and $G_{\text{real}}(x)$, to a second translation network, to get back to the original domain, either real or synthetic, to close the cyclic loop.

system. In particular, this method can learn to capture special characteristics of one dataset (styles), and then to translate them into the target dataset, despite the absence of paired training examples between datasets. Motivated by this possibility, we propose to build a model that can take advantage of CycleGAN's properties to improve segmentation results in a seismological scenario.

Our image-to-image translation approach consists of four CNN elements: translator real G_{real}, translator synthetic G_{syn}, discriminator real D_{real}, and discriminator synthetic D_{syn}. Note that the translators are responsible for transforming the images, therefore, acting as generators. On the one hand, G_{real} translates images from the synthetic to the real style domain. Although these generated images remain synthetic, now they contain characteristics from the real domain. On the other hand, G_{syn} does the opposite translation, i.e., it translates images from the real to the synthetic style domain. Even though we are not interested in this latter translation, it is important to have the full cycle (symmetry), as it brings stability in the training of the translation system. Finally, the discriminators determine whether the generated images look like the input data or not, i.e., whether they share the same style domain. Since there are two style domains, there is one discriminator to assess the translation to the real domain, D_{real}, and one to the synthetic domain, D_{syn}. Figure 6.5 shows the scheme of the pipeline of the aforementioned elements. We can see how the system transfers images from a synthetic, to a real, and back to a synthetic domain (or the other way around), and how these modified images are evaluated by the discriminators.

6.3.3 Multi-Objective Learning

We can control the interaction between the segmentation network and the translation network through the combination of different loss terms that forms the overall optimization function. In this work, we use four independent losses in training to achieve our goal.

$$\mathcal{L}_{\text{adv}} = \mathcal{L}_{\text{adv,real}} + \mathcal{L}_{\text{adv,syn}}, \tag{6.1}$$

where the adversarial loss responsible for translating to the real domain is

$$\mathcal{L}_{\text{adv,real}} = \mathbb{E}_{\boldsymbol{y}}\left[\log\left(D_{\text{real}}(\boldsymbol{y})\right)\right] + \mathbb{E}_{\boldsymbol{x}}[\log(1 - D_{\text{real}}(G_{\text{real}}(\boldsymbol{x})))], \tag{6.2}$$

and to the synthetic style domain is

$$\mathcal{L}_{\text{adv,syn}} = \mathbb{E}_{\boldsymbol{x}}\left[\log\left(D_{\text{syn}}(\boldsymbol{x})\right)\right] + \mathbb{E}_{\boldsymbol{y}}[\log(1 - D_{\text{syn}}(G_{\text{syn}}(\boldsymbol{y})))]. \tag{6.3}$$

The adversarial loss $\mathcal{L}_{\text{adv}}$ [3] is the core element in any GAN-based model. Essentially, it makes the generated images more realistic and evaluates the control over the style domain.

$$\mathcal{L}_{\text{cycle}} = \mathbb{E}_{\boldsymbol{x}}[||G_{\text{syn}}(G_{\text{real}}(\boldsymbol{x})) - \boldsymbol{x}||_1] + \mathbb{E}_{\mathbf{y}}[||G_{\text{real}}(G_{\text{syn}}(\boldsymbol{y})) - \boldsymbol{y}||_1] \tag{6.4}$$

The cyclic consistency loss $\mathcal{L}_{\text{cycle}}$ [15] guarantees the preservation of the domain's invariant characteristics, e.g., geological structures, while changing its styles faithfully. Its goal is to ensure that mappings between domains are the reverse of each other. For instance, for the real data we have $\boldsymbol{y} \rightarrow G_{\text{syn}}(\boldsymbol{y}) \rightarrow G_{\text{real}}(G_{\text{syn}}(\boldsymbol{y})) \approx \boldsymbol{y}$. Due to forward and backward operations, cycle consistency loss can reduce the space of possible mapping functions, leading to a unique map between input and output, and therefore, a more stable algorithm.

$$\mathcal{L}_{\text{id}} = \mathbb{E}_{\boldsymbol{x}}[||G_{\text{syn}}(\boldsymbol{x}) - \boldsymbol{x}||_1] + \mathbb{E}_{\boldsymbol{y}}[||G_{\text{real}}(\boldsymbol{y}) - \boldsymbol{y}||_1] \tag{6.5}$$

The identity loss $\mathcal{L}_{\text{id}}$ [15] helps to regularize the generator to work along the lines of an identity mapping when samples from the target domain are provided. In other words, given an input that already looks like it is from the target domain, the system should not map it into a different image.

$$\mathcal{L}_{\text{seg}} = -\sum_{h,w} \mathbb{E}_{\mathbf{x}}[\boldsymbol{m}^{h,w} \log S(\boldsymbol{x}^{h,w}) + (1 - \boldsymbol{m}^{h,w}) \log(1 - S(\boldsymbol{x}^{h,w}))] \tag{6.6}$$

The segmentation loss $\mathcal{L}_{\text{seg}}$ is set to be the average binary cross-entropy distance between the distributions of the labels and the distribution of the output of the network, over the training dataset. The distance is minimized when the network learns to better approximate the solution given in the training examples.

Combining all the aforementioned loss terms leads to the final objective, which can be formulated as

$$\min_{G,S} \max_{D} \mathcal{L}_{\text{final}} = \lambda_{\text{adv}}\mathcal{L}_{\text{adv}} + \lambda_{\text{cycle}}\mathcal{L}_{\text{cycle}} + \lambda_{\text{id}}\mathcal{L}_{\text{id}} + \lambda_{\text{seg}}\mathcal{L}_{\text{seg}}, \tag{6.7}$$

where λ_{adv}, λ_{cycle}, λ_{id}, and λ_{seg} are the hyperparameters for each term.

6.3.4 End-to-End Model Architecture

To successfully combine the different networks and their corresponding loss terms, it is essential to train them in a synchronized, end-to-end fashion. In particular, we integrate the image-to-image translation network on top of a semantic-segmentation network, resulting in an improvement when testing with real data. Unlike other approaches trained only on synthetic data, our method also allows real data information to be incorporated, which boosts the results at testing time. In order to do so, we feed the segmentation network with both synthetic and generated data. Note, however, that the latter has been produced by the generative network, by mimicking real data features, hence, it comes with labels since it was originally a synthetic sample. Algorithm 7 describes the pseudocode of our proposal.

Algorithm 7 Training of the proposed model.

```
Require: $n_{\text{iter}}$, number of iterations. $\alpha$'s, learning rates. $m$, batch size. $n_{\text{gen}}$, number of skipped iterations of the generator per discriminator iteration.
Require: $\boldsymbol{\theta}$, initial segmentor, generators and discriminators parameters.
for $i < n_{\text{iter}}$ do
    Sample a batch of synthetic images $\{x^{(j)}\}_{j=0}^{m}$
    Mask of the batch of images $\{m^{(j)}\}_{j=0}^{m}$
    Sample a batch of real images $\{y^{(j)}\}_{j=0}^{m}$
    Train discriminators $D$
    $\boldsymbol{\theta}_{\text{disc}} \leftarrow \boldsymbol{\theta}_{\text{disc}} - \alpha_{\text{disc}} \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\text{adv}}(\boldsymbol{x}, \boldsymbol{y})$
    Train generators $G$
    if $mod(i, n_{\text{gen}}) = 0$ then
        $\boldsymbol{\theta}_{\text{gen}} \leftarrow \boldsymbol{\theta}_{\text{gen}} - \alpha_{\text{gen}} \nabla_{\boldsymbol{\theta}} (\mathcal{L}_{\text{adv}} + \mathcal{L}_{\text{cycle}} + \mathcal{L}_{\text{id}} + \mathcal{L}_{\text{seg}})(\boldsymbol{x}, \boldsymbol{m}, \boldsymbol{y})$
    end if
    Train segmentor $S$
    $\boldsymbol{\theta}_{\text{seg}} \leftarrow \boldsymbol{\theta}_{\text{seg}} - \alpha_{\text{seg}} \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\text{seg}}(\boldsymbol{x}, \boldsymbol{m})$
end for
```

6.4 Experiments

In this section, we evaluate the proposed system for the detection of diffraction and fault scenarios. For each task, we train our model on a synthetic dataset, specifically created to contain the geological features that we aim to highlight in the real field data. We first give a detailed description of the experimental setup of each case separately, and then, we discuss the results and their geophysical interpretation.

6.4.1 Experimental Setup

Detection of diffraction. In this first scenario, our goal is to detect diffracted waves in prestack seismic data, migrated to the dip-angle domain [167,168]. In this domain, the angles relate to the different directions of illumination of the subsurface. We work with a field dataset, recorded as a 2D line in shallow waters with a high-resolution, low-penetration source. Interpreting this type of data is challenging, since its high-frequency and shallow nature induce a lot of noise. It is likely that most scattering events in this dataset are caused by small boulders with

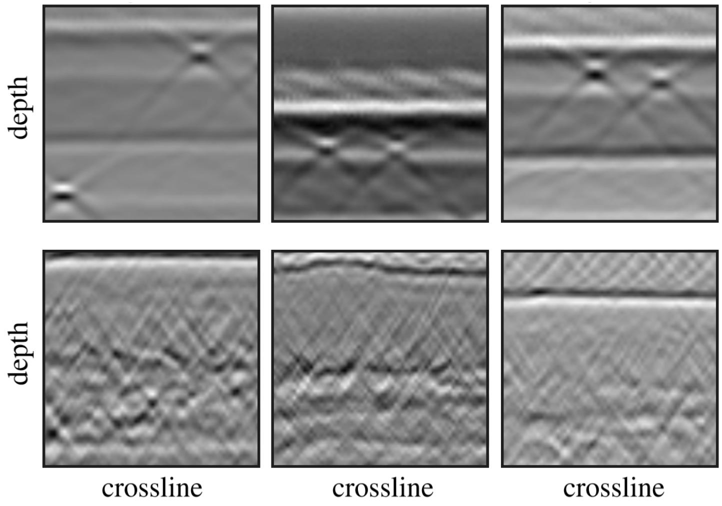


Figure 6.6. Stacked training samples from the detection of diffraction experiment. (Top) Synthetic images containing diffractions, i.e., boulders. (Bottom) Real images which may contain scattering points. The vertical sampling and the horizontal sampling are both 0.5 m. The size of each image is 64×64 pixels = 32×32 m.

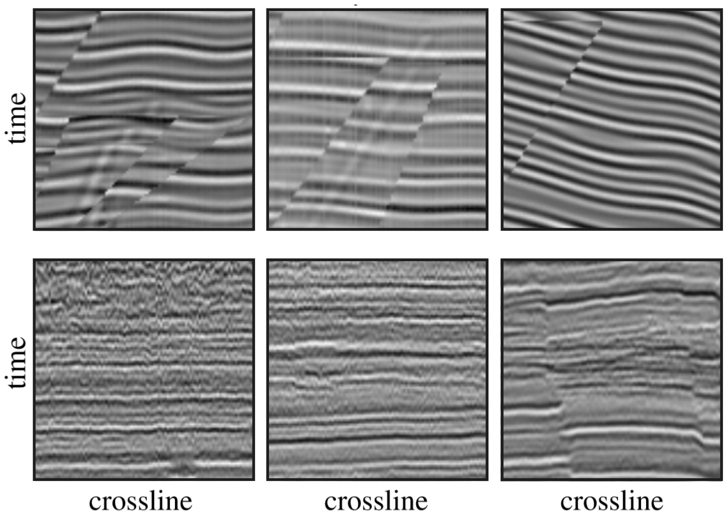


Figure 6.7. Training samples from the detection of faults experiment. (Top) Synthetic images containing faults. (Bottom) Real images which may contain faults. The vertical sampling is 4 ms and the horizontal sampling is 12.5 m. The size of each image is 96×96 pixels = 0.384 s × 1200 m.

a high acoustic impedance. Additionally, we create a synthetic dataset with matching geometries, by generating a number of random subsurface models that contain punctual, high acoustic impedance perturbations. The labels to train our segmentation network are only available for the synthetic dataset. Both datasets contain three dimensions: two spatial dimensions (crossline and depth), as well as one prestack dimension (dip-angle). We work with a 2D CNN, where the convolutions happen in the 2D spatial domain for every dip-angle slice. This is analogous to the way CNNs process 2D colour images [29]. From the real dataset we extract 150 prestack data patches of $64\times64\times71$ pixels in the crossline×depth×dip-angle directions. We also extract 150 patches of the same shape from the synthetic dataset, with their matching label patches. Figure 6.6 shows a few training samples.

Detection of faults. For this second scenario, we aim to automatically highlight the main faults present in a seismic volume. The goal is to train a neural network that computes a fault-delineation attribute, which can yield good results on field data where little to no manual annotation is available. We test our workflow with the open F3-Netherlands [169] dataset, a small 3D seismic survey from offshore Netherlands, containing both long and well-defined faults as well as smaller and more chaotic fault networks. We generate a synthetic faulted seismic dataset, similarly to [147]. From the real dataset we extract 250 data patches of 96×96 pixels, in the crossline×time directions, and 150 patches of the same shape from the synthetic dataset, with their matching label patches. Figure 6.7 shows a few training samples.

6.4.2 Training Setting

Our approach is divided into two blocks: the segmentor and the translator. Both of them use independent Adam [54] optimizers with $\beta_1 = 0.5$, $\beta_2 = 0.999$. The batch size is set to 8, and the model is trained for 200 epochs. The generator is updated after every two discriminator updates. For the detection of the diffraction scenario, we set λ_{adv}, λ_{cycle}, λ_{id}, and λ_{seg} to 2, 10, 2 and 1, respectively. And for the detection faults we set λ_{adv}, λ_{cycle}, λ_{id}, and λ_{seg} to 1, 0.1, 5 and 0.1, respectively. Furthermore, we implement a learning-rate scheduler, setting its initial value to $2\cdot10^{-4}$, and linearly decreased to zero over the last 100 epochs.

6.4.3 Results of Detecting Diffraction

In order to start the evaluation, we first analyse the domain transformations of our generative model. Notice that we are mostly interested in the transformation of synthetic to real data, since its output is used to train the segmentor. This is possible because the generated data is more similar to the real data, and it also inherits the masks from the purely (original) synthetic data. Figure 6.8 depicts an example of this transformation in the stacked domain, where an original

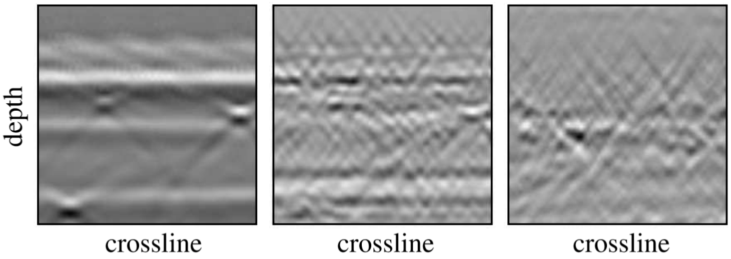


Figure 6.8. From left to right: a stacked synthetic training sample, its stacked transformation and one random stacked real sample. The vertical sampling and the horizontal sampling are both 0.5 m. The size of each image is 64×64 pixels = 32×32 m.

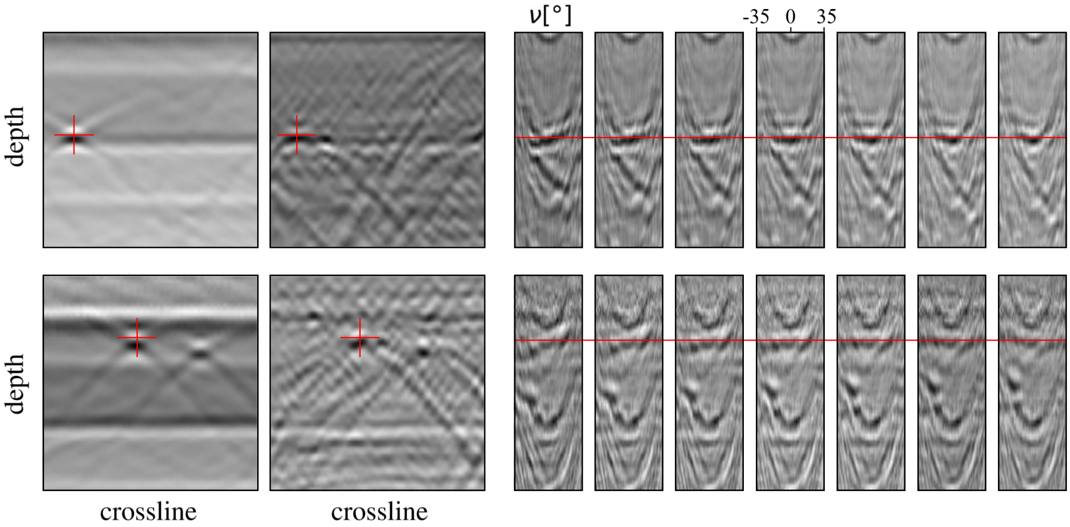


Figure 6.9. Each row displays a transformation example in the prestack domain for the detection of diffraction (boulders). From left to right: a stacked synthetic training sample, its stacked transformation and its transformation before stacking. The red lines highlight synthetic scattered events. For the spatial images, the vertical sampling and the horizontal sampling are both 0.5 m. The size of each image is 64×64 pixels = 32×32 m. For the gathers (prestack), the vertical sampling is 0.5 m and horizontal sampling is 1°. The size of each gather is 64×71 pixels = 32 m × 71°.

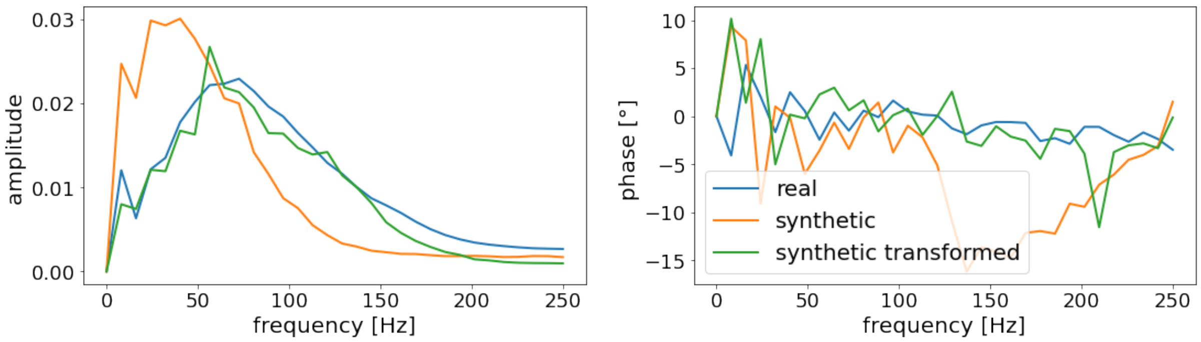


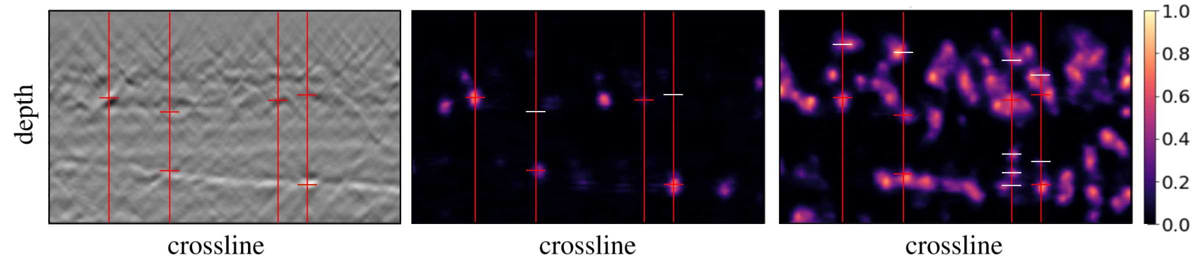
Figure 6.10. Average frequency spectrum computed on real, synthetic and synthetic transformed domains for the diffraction datasets.

image is shown (left), its transformed image (centre), and one real image (right). We can see the style similarities between the real and transformed data, and structure similarities between the original and transformed data. However, it is not sufficient to analyse the stacked domain. It is imperative that the generated images are coherent in the prestack domain too. Thus, we conduct a second validation inspection, this time by looking at the prestack domain along the dip-angles direction. Figure 6.9 shows how our system can produce synthetic seismic images in the dip-angle migration domain, where the diffraction curves produce the expected patterns. We also display in Figure 6.10 the average frequency spectrum. We observe that both the amplitude and the phase of the synthetic data are much closer to the real data once they have been modified by the generative network.

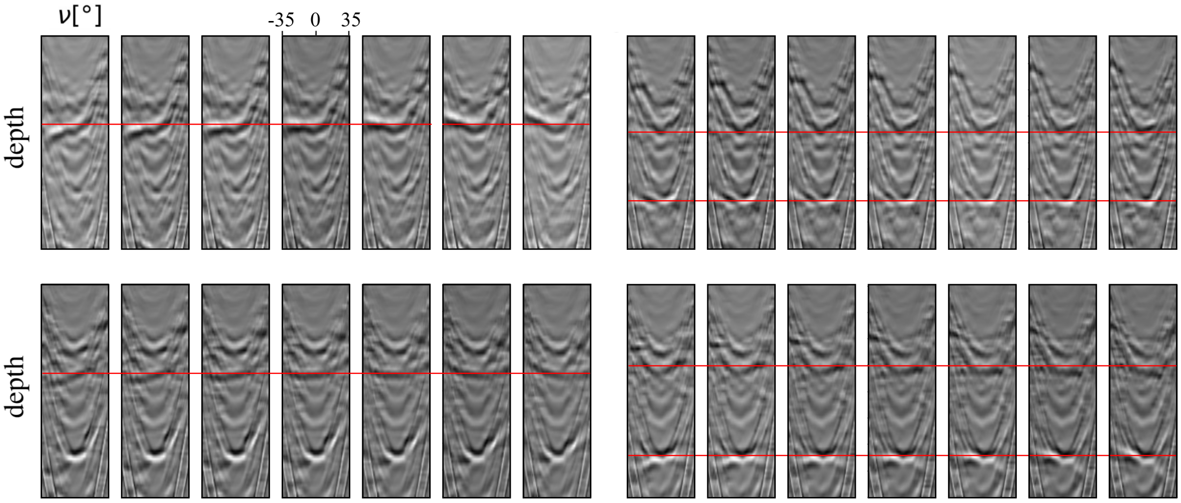
In Figure 6.11 we can observe the result given by the segmentation network obtained at testing time. Since we do not know the true number or the location of the scattering objects, there is no label information nor ground-truth, so we cannot easily give a quantitative measure of the performance of the system. Nevertheless, to assess the results, we investigate the data manually, and we also conduct an ablation study. This study involves running the same experiment but using only the segmentation network, trained on purely synthetic data. We remove the generative model, but we keep the same hyperparameters for the segmentor for a fair comparison. We argue that, in this way, this test helps to reveal the impact of our approach. Figure 6.11a shows the real data (left), the objects found by our segmentation network (centre), and the objects found by the ablation study (right). We manually verify that most of the time, our method correctly detects diffraction (boulders), leading to a high true-positive rate. However, it does find fewer potential boulders than the ablation case, which might imply that there are several false negatives. On the other hand, the ablation system seems to over-detect and consequently, to display an unrealistic amount of diffraction, where many of them are just false positives. Overall, we are satisfied with the rate of true positives of our method. Most areas highlighted by the attribute seem to correspond to actual diffracted events, leading to a clean prediction attribute.

6.4.4 Results of Detecting Faults

In this second set of experiments, we also start the evaluation by visually inspecting the transformations of synthetic data into real data. In this case, the generative task is simpler since the data is already in the stacked domain, i.e., no depth dimension. Figure 6.12 shows a few pairs of this transformation, where we can assess the efficacy of our approach. Although the data generated do not look completely realistic, in particular the fault planes, they do seem visually closer to the real data. We also display in Figure 6.13 the average frequency spectrum. We observe again that both the amplitude and the phase of the synthetic data are much closer to the



(a) From left to right: real data, prediction of our approach, prediction of the ablation. The red horizontal lines highlight the true positive findings, while the white horizontal lines highlight both false positives and false negatives. The red vertical lines indicate the slices where we look at the prestack data (see b).



(b) Clockwise from top-left: the prestack data corresponds to the red vertical lines from real data (see a left image). The red horizontal lines on the prestack data match with the ones on the real data.

Figure 6.11. Prediction of diffraction (boulders) of our approach on the real data at testing time, and its ablation counterpart.

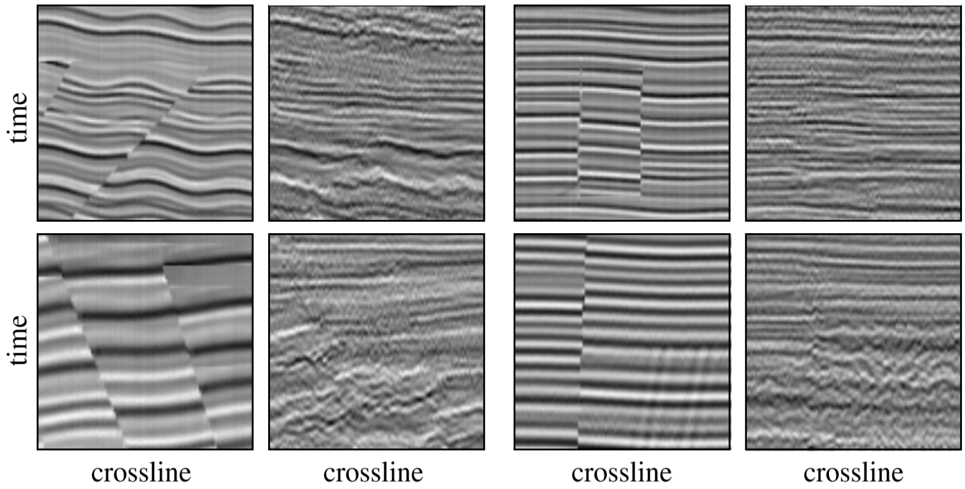


Figure 6.12. Pairs of synthetic training samples (1st and 3rd columns) and their transformations into real data (2nd and 4th columns). The vertical sampling is 4 ms and the horizontal sampling is 12.5 m. The size of each image is 96×96 pixels = 0.384 s × 1200 m.

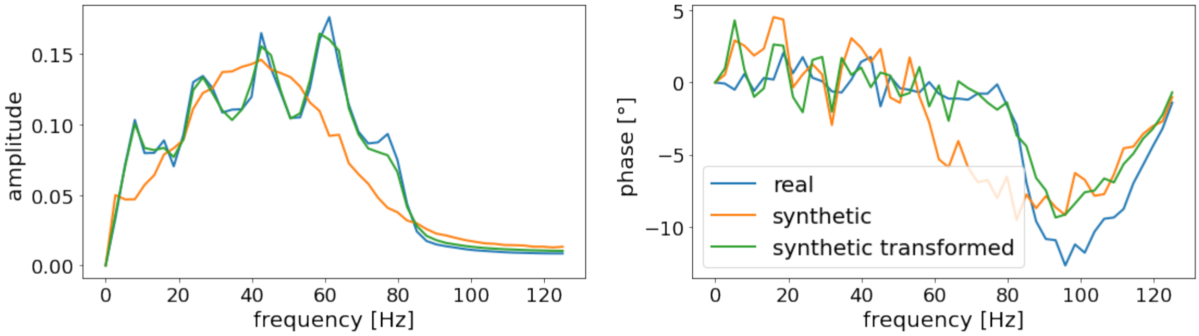


Figure 6.13. Average frequency spectrum computed on real, synthetic and synthetic transformed domains for the faults datasets.

real data once they have been modified by the generative network.

As in the diffraction case, we now also need to run an ablation study to corroborate that our proposed method outperforms the plain segmentation network on the fault scenario. Again, we do not know the true number nor the location of the faults. Therefore, a quantitative measurement of the performance of the method is not easy to conduct. Nevertheless, due to the nature of this data, it is likely, to a certain extent, to visually determine the effectiveness of the approach. Figure 6.14 displays the result given by the segmentation network, trained on both synthetic data and generated data (left), and on only synthetic data (right), the latter of which being the results of the ablation study. The original synthetic training data only contained perfect fault planes, hence, the ablation approach has troubles with predicting the non-ideal fault planes in the real data. On the other hand, our method trained on the modified synthetic data, shown in Figure 6.12, is better at recognizing real faults as well as leading to more true positives and longer, more realistic fault-plane predictions.

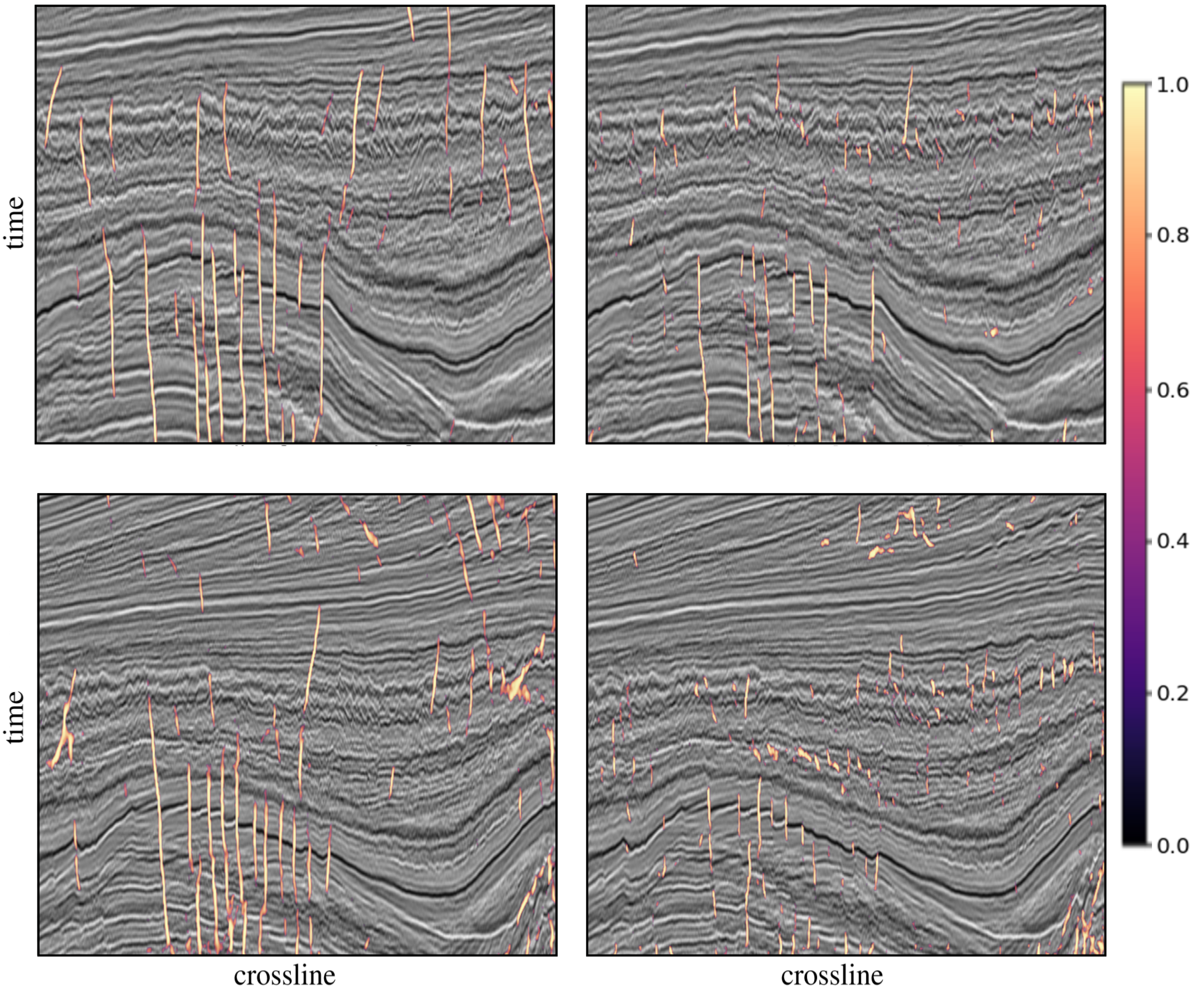


Figure 6.14. Prediction of faults on the real data at testing time. (Left) Predictions on testing data of our approach. (Right) Predictions on testing data of the ablation study. The vertical sampling is 4 ms and the horizontal sampling is 12.5 m. The size of each image is 311×600 pixels = 1.244 s × 7500 m.

6.5 Limitations

The main success of our method is helping to reduce the generalization gap between the training (synthetic) and testing (real) data. Indeed, while it is relatively trivial to train a neural network to converge on a small, well-controlled dataset, it is highly challenging to train a neural model that can consistently yield accurate results on unseen data, with potentially very different acquisition, geology, and processing settings. We know, from other domains, that the best way to reduce this gap is to gather a profusion of high-quality, high-diversity data examples, often annotated by hand [155]. However, this method does not scale well to geosciences, where data requires expertise and continued efforts to be interpreted. Within our framework, we diminish the need for manual annotations to a large extent. In fact, in both experiments we use only synthetic labels, obtained from simple modelling. Thanks to the generative model, we refine the synthetic images to shrink the generalization gap, achieving superior results than when the network is trained on unmodified synthetic data (ablation study). While promising, the method also suffers from some limitations. Firstly, it requires training several networks in a unified fashion. In addition to being computationally more demanding, it also becomes more challenging for the practitioner to tune the hyperparameters, in order to properly balance all the loss terms together and bring the system to a stable equilibrium. Moreover, the task is complicated by the fact that we do not have access to a robust quantitative measure of the final accuracy of the model. Indeed, our goal is to yield an improved prediction on the unlabelled dataset, and since it is not interpreted, one needs to resort to a visual inspection of the results and to rely on a more qualitative assessment. This way of proceeding may be time consuming and does not allow for the use of automated hyperparameter-selection methods. Finally, in both of the evaluated applications, we observe several remaining false negatives. Namely, it appears that despite the use of the generative model to match the target data, the segmentation network is not able to recognize some of the most challenging objects in the data. This, for instance, is the case in Figure 6.11, where the noisiest diffraction patterns are not detected. In Figure 6.14, we also see that many of the small faults, present in the relatively chaotic area at the centre of the data, are not highlighted. For these examples, it is likely that additional manual annotations are required to be fed into the network, enabling it to learn to recognize the rarest of events obscured by the highest noise levels. This could be achieved by fine-training the model, using a selection of additional data chosen by the interpreter, in a manner similar to that described in [148].

6.6 Summary

In this chapter, we propose an end-to-end model that allows synthetic patterns to be transferred into generated data, as well as the identification of seismic events to be automated, such as diffraction and faults. We formulate the problem as a semantic-segmentation task, where the positions of the events are expressed as probability distributions, and the approach is based on a self-supervised deep-learning technique. In order to boost the segmentation results, our translation network transfers the characteristics from the real domain to the synthetic domain, while keeping the original synthetic structure. In this way, we can guarantee that the generated images will be valid for training the segmentor, i.e., they match with the labels, and also that their geophysical properties will then be more realistic. This highlights the attractiveness of the proposed system, in particular, helping in common scenarios where the scarcity of annotated data is the main limiting factor. Our approach is also versatile. We demonstrate the use of our framework for the detection of faults and diffraction and the method can be applied in a similar fashion to interpret other elements, such as salt bodies or channel networks. Additionally, while we show its use in helping the transition from synthetic data to real data, the framework may be applied equivalently to other transfer problems. For instance, one can employ a legacy field dataset, manually interpreted by geoscientists, as the base training data, and choose a new non-interpreted dataset as the target. The method might also help in reducing the decrease in accuracy that may be observed on a dataset after a pre-processing step changes the amplitudes, or between two vintages of time-lapse datasets. On the other hand, we believe that human experts still play an essential role, since uncertainties and non-uniqueness of the interpretation must be considered when evaluating the performance of a network. All in all, we foresee great potential in our framework, and hope to prove it valuable in further work for other applications. We think that it is a promising approach that can help to fill the gap between the need for a vast amount of training data, and the need to obtain good results on real data, with minimal effort using synthetic modelling.

Chapter 7

Conclusion

This thesis proposes several approaches to learn image-to-image translation—including attribute and style transforms—together with the dangers that such technology can pose and a seismic use case. In this final chapter, we summarize the main findings, and propose some interesting directions for future work.

7.1 Summary

We start the thesis introducing deep learning as a core component of current approaches for most computer vision tasks. The reason for such broad usage of this technology is its intrinsic ability to automatically learn specific features from a training dataset, under the constraints of a given problem, such as image classification. In our work, we focus on the vision task of image generation and manipulation, where we employ generative adversarial networks as a baseline. Originally, Goodfellow *et al.* conceived GANs for pure image generation purposes, where the input was random noise and the output a synthetic image. However, ever since then, GANs have been developed in many other scenarios, resulting in a wide range of applications. In this thesis, we employ them for image-to-image translation, more specifically for domain transformation, including attribute and style transformation.

Although image-to-image translation has drawn increasing attention and made tremendous progress, most approaches are still heavily data-dependent, being very sensitive to the amount of annotated training data. Therefore, when one wants to extrapolate the problem to new scenarios, where labelled data is not abundant, the systems might perform poorly. Driven by the motivation to solve this problem, in our second chapter, we propose an approach that can overcome a labelled-limited scenario. More specifically, we tackle few-shot learning for attribute transfer, which is a form of restricted supervision, where we train a model on certain attributes and test on unseen ones that appear during testing. To that end, we build a GAN-based approach that trains in a meta-training fashion, allowing us to perform attribute transfer using just a few labelled samples from a limited class. This sort of approach might become a powerful paradigm, solving the label limitation that supervised generative methods suffer from. Furthermore, it is conducted with no lengthy inference time, no external memory and no additional data. On the other hand, while it is true that our method successfully generates samples from a new target domain, all the involved domains are very similar—at least from a level of human understanding. Therefore, one important limitation might arise when trying to generalize for a wider range of attribute domains, especially when those involve morphological changes, e.g., training with attribute domains such as *blond*, *moustache* or *eyeglasses*, and then at meta-testing time, targeting *skin colour* or *gender*. Said generalizable property is the ultimate goal for any attribute transfer model, in particular, for those with scarcity of labelled data.

We then study the attribute-transfer problem from a more common perspective. That is, we train with annotated data, where the labels describe all the different attributes that appear at training as well as testing time. A priori, this is a simpler scenario since the algorithm is trained with all the possible transformations. However, it is an intrinsically complex problem as it involves not only colour transformations, but also morphological changes. To achieve this goal, we introduce a novel attribute-transfer approach that integrates an inpainting block so that the input image is modified in such a way that the output incorporates the pre-defined target attributes. This block allows all attention to be focused on the region of interest—where the attributes are located—keeping it consistent with the unmodified background. In particular, we take advantage of the fact that most facial attributes are induced by local structures, e.g., relative position between the eyes and ears. Hence, it is sufficient to change only a few parts of the face, and to teach the generator to synthesize realistic outputs. While this symmetrical property of local structures allows synthetic faces to be successfully rendered that contain the target attributes, it also constrains the framework's usage for other applications. The main reason for this limitation is the strong dependency on the structural properties of the input data that our method relies on. As a result, in order to explore different applications, one needs either to employ images with structural similarities, or to implement a stronger reconstructor. A second drawback of our approach arises from the inpainting strategy. This might compromise the identity of the original face, by slightly modifying it or by completely removing it.

Besides the attribute-transfer ability of our approach, it might be useful to have a system that can steer synthetic images between different style domains as well. Usually, these domains represent distinguishable properties that are part of the image and cannot be described by single attributes. Despite the importance of having such systems, little work has unified both attribute and style transformations under the same pipeline. In the fourth chapter, we build a framework that incorporates successfully this pipeline into an end-to-end system. In particular, given an input face image, a target-style face image, and a guide segmentation label mask, our novel framework is able to synthesize an output image that shares a similar style with the target-style image, while preserving the input face's identity and accurately follows the semantic mask. To achieve this, we propose a multi-objective training process that balances the different components of the architecture. Furthermore, we introduce a customized segmentation loss to encourage the network to follow the geometry specified in the guide face mask, back-propagating informative gradients thanks to its locality characteristics. However, if the modified mask contains unnatural structures, e.g., no eyes, the network starts to ignore the mask input in order to avoid generating unrealistic faces that are not consistent with the geometry information. A second limiting factor comes from the need for plentiful segmentation masks for training, resulting in a rather expensive dataset.

In recent years, the increasing sophistication of smartphones and the growth of social networks have led to a gigantic amount of new digital content. This tremendous use of digital images has been followed by a rise in techniques to alter them. Deep generative models, especially GANs, have lately been extensively used to produce artificial images with a realistic appearance. As a result, a new vein of AI-based fake image generation have emerged, leading to a fast dissemination of high-quality forged content. While significant developments have been made for image forgery detection, it still remains an ongoing research task. In the fifth chapter, we address the problem of detecting artificial image content, more specifically, fake faces. Despite the fact that many face-editing, image-to-image translation and generating algorithms seem to produce realistic human faces, upon closer examination, they do exhibit artefacts in certain domains, which are often hidden to the naked eye. To spot such irregularities (artefacts), we present a simple yet effective method, based on classical frequency domain analysis. Using only a small amount of annotated training data, or even none at all, our approach is able to detect generated faces with almost perfect accuracy. Besides the accuracy results, what makes our approach even more appealing than deep-learning methods is the capacity of learning with few examples, leading to a larger field of applications. Nonetheless, the problem is far from being solved, since simple image processing techniques, like compression or image processing, can alter the original frequency content and break our classification system.

Finally, we introduce a novel framework to reduce the generalization gap that exists between oversimplified synthetic training data and real application data. Our proposed model addresses the semantic-segmentation task by exploiting domain-style transfers on the training data. In particular, we build a style-transfer system that generates data, which preserves the synthetic patterns and contains the style features from real data samples. The combination of synthetic data and its realistic transformation is the key to providing a larger, more realistic variety of samples, with their ground-truth, at minimal manual labour cost. As a result, our system outperforms standard methods, which are traditionally trained only on purely synthetic data. On the other hand, the problem still remains unsolved. Although it is relatively trivial to train a neural network to converge on a small, well-controlled dataset, it is highly challenging to train a neural model that can consistently yield accurate results on unseen data, with potentially very different acquisition, geology, and processing settings. Consequently, human experts still play an essential role, since uncertainties and non-uniqueness of the interpretation must be considered when evaluating the performance of a network.

7.2 Future Work

Throughout this thesis, we have stressed the importance of labelled data, and the limitations that the absence of which might have in deep-learning approaches. We believe that image manipulation, in particular, image-to-image translation, can be used to synthesize data on demand, and to mitigate in this way these label limitations. This is done by trying to generate new samples from particular domains/classes to match pre-defined scenarios. We think that the repercussion of such systems could not only impact the computer vision community, but also other deep-learning communities, which also suffer from scarce annotated datasets. However, most of the current approaches rely on supervised setups, making the problem a vicious circle—a chicken and egg situation. Therefore, we believe that the future of image-to-image translation must involve going through new solutions that require as little annotated data as possible. Eventually, the ideal algorithms should become unsupervised, minimizing the need for human involvement. In fact, such a necessity is not unique for image-to-image problems, but it is also for other vision domains, such as image classification, semantic segmentation, object detection, to name a few.

On the other hand, besides the technical improvements, e.g., semi-supervised setups, more accurate attribute transfer, and incorporation of geometry manipulations, the image-to-image community has to make an effort to bring its technology closer to real-world applications. For example, we foresee a wide range of applications in the medical field, where the scarce amount of public data and its notorious confidentiality still restrict many deep-learning solutions nowadays. Another interesting field, with a lot of potential applications, is the seismic domain. As discussed in the previous chapter, the amount of work employing deep learning, including image-to-image transformation, has exponentially increased over the past years. Apart from mitigating the problem of the amount of annotated real data, another main reason for employing image-to-image transformations is to help to correct geological data that contains artefacts. Far from being a rare event, this is a usual problem introduced by the migration techniques that the data undergoes before visualization. As one can imagine, similar to medical application, the amount of seismic practical possibilities is huge. Nonetheless, we think that this is only possible by promoting interdisciplinary studies, where the whole chain of participants understands the bigger picture. To that end, it is important to establish new avenues and spaces to allow these collaborations to happen. Finally, it is essential to create and open new datasets that allow researchers to benchmark the methods with more real-case scenarios, rather than toy examples.

Bibliography

[1] Aaron van den Oord, Nal Kalchbrenner, Oriol Vinyals, Lasse Espeholt, Alex Graves, and Koray Kavukcuoglu. Conditional image generation with pixelcnn decoders. *arXiv preprint arXiv:1606.05328*, 2016.

[2] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[3] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[4] Leonard Adolphs, Hadi Daneshmand, Aurelien Lucchi, and Thomas Hofmann. Local saddle point optimization: A curvature exploitation approach. *arXiv preprint arXiv:1805.05751*, 2018.

[5] Abhishek Kumar, Prasanna Sattigeri, and Tom Fletcher. Semi-supervised learning with gans: Manifold invariance with improved inference. *Advances in Neural Information Processing Systems*, 30, 2017.

[6] Ricard Durall, Valentin Tschannen, Norman Ettrich, and Janis Keuper. Generative models for the transfer of knowledge in seismic interpretation with deep learning. *The Leading Edge*, 40(7):534–542, 2021.

[7] Ali Taleb Zadeh Kasgari, Walid Saad, Mohammad Mozaffari, and H Vincent Poor. Experienced deep reinforcement learning with generative adversarial networks (gans) for model-free ultra reliable low latency communication. *IEEE Transactions on Communications*, 69(2):884–899, 2020.

[8] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

[9] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4401–4410, 2019.

[10] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8110–8119, 2020.

[11] Yuan Yuan, Siyuan Liu, Jiawei Zhang, Yongbing Zhang, Chao Dong, and Liang Lin. Unsupervised image super-resolution using cycle-in-cycle generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 701–710, 2018.

[12] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *Proceedings of the European conference on computer vision (ECCV) workshops*, pages 0–0, 2018.

[13] Kelvin CK Chan, Xintao Wang, Xiangyu Xu, Jinwei Gu, and Chen Change Loy. Glean: Generative latent bank for large-factor image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14245–14254, 2021.

[14] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[15] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2223–2232, 2017.

[16] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8789–8797, 2018.

[17] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544, 2016.

[18] Yijun Li, Sifei Liu, Jimei Yang, and Ming-Hsuan Yang. Generative face completion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3911–3919, 2017.

[19] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5505–5514, 2018.

[20] Nasim Souly, Concetto Spampinato, and Mubarak Shah. Semi and weakly supervised semantic segmentation using generative adversarial network. *arXiv preprint arXiv:1703.09695*, 2017.

[21] Ricard Durall, Franz-Josef Pfreundt, Ullrich Köthe, and Janis Keuper. Object segmentation using pixel-wise adversarial loss. In *German Conference on Pattern Recognition*, pages 303–316. Springer, 2019.

[22] Daiqing Li, Junlin Yang, Karsten Kreis, Antonio Torralba, and Sanja Fidler. Semantic segmentation with generative models: Semi-supervised learning and strong out-of-domain generalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8300–8311, 2021.

[23] Richard Zhang, Jun-Yan Zhu, Phillip Isola, Xinyang Geng, Angela S Lin, Tianhe Yu, and Alexei A Efros. Real-time user-guided image colorization with learned deep priors. *arXiv preprint arXiv:1705.02999*, 2017.

[24] Rui Li, Wenming Cao, Qianfen Jiao, Si Wu, and Hau-San Wong. Simplified unsupervised image translation for semantic segmentation adaptation. *Pattern Recognition*, 105:107343, 2020.

[25] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European conference on computer vision*, pages 649–666. Springer, 2016.

[26] Kamyar Nazeri, Eric Ng, and Mehran Ebrahimi. Image colorization using generative adversarial networks. In *International conference on articulated motion and deformable objects*, pages 85–94. Springer, 2018.

[27] Yunjey Choi, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. Stargan v2: Diverse image synthesis for multiple domains. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8188–8197, 2020.

[28] Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. Maskgan: Towards diverse and interactive facial image manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5549–5558, 2020.

[29] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

[30] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. *arXiv preprint arXiv:2106.04560*, 2021.

[31] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[32] Andrew Tao, Karan Sapra, and Bryan Catanzaro. Hierarchical multi-scale attention for semantic segmentation. *arXiv preprint arXiv:2005.10821*, 2020.

[33] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *Advances in neural information processing systems*, pages 3630–3638, 2016.

[34] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. 2016.

[35] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1126–1135. JMLR. org, 2017.

[36] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *Advances in Neural Information Processing Systems*, pages 4077–4087, 2017.

[37] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip HS Torr, and Timothy M Hospedales. Learning to compare: Relation network for few-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1199–1208, 2018.

[38] Brenden Lake, Ruslan Salakhutdinov, Jason Gross, and Joshua Tenenbaum. One shot learning of simple visual concepts. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 33, 2011.

[39] Danilo Jimenez Rezende, Shakir Mohamed, Ivo Danihelka, Karol Gregor, and Daan Wierstra. One-shot generalization in deep generative models. *arXiv preprint arXiv:1603.05106*, 2016.

[40] Sergey Bartunov and Dmitry Vetrov. Few-shot generative modelling with generative matching networks. In *International Conference on Artificial Intelligence and Statistics*, pages 670–678, 2018.

[41] Louis Clouâtre and Marc Demers. Figr: Few-shot image generation with reptile. *arXiv preprint arXiv:1901.02199*, 2019.

[42] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

[43] Gregory Koch, Richard Zemel, and Ruslan Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2, 2015.

[44] Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pages 1842–1850, 2016.

[45] Tsendsuren Munkhdalai and Hong Yu. Meta networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2554–2563. JMLR. org, 2017.

[46] Antreas Antoniou, Harrison Edwards, and Amos Storkey. How to train your maml. *arXiv preprint arXiv:1810.09502*, 2018.

[47] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

[48] Chelsea Finn, Kelvin Xu, and Sergey Levine. Probabilistic model-agnostic meta-learning. In *Advances in Neural Information Processing Systems*, pages 9516–9527, 2018.

[49] Yann LeCun, Corinna Cortes, and CJ Burges. Mnist handwritten digit database. *AT&T Labs [Online]. Available: http://yann. lecun. com/exdb/mnist*, 2:18, 2010.

[50] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

[51] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. In *Advances in Neural Information Processing Systems*, pages 5767–5777, 2017.

[52] LI Chongxuan, Taufik Xu, Jun Zhu, and Bo Zhang. Triple generative adversarial nets. In *Advances in neural information processing systems*, pages 4088–4098, 2017.

[53] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision*, pages 3730–3738, 2015.

[54] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[55] Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural Computation*, 9(1):1–42, 1997.

[56] Pavel Izmailov, Dmitrii Podoprikhin, Timur Garipov, Dmitry Vetrov, and Andrew Gordon Wilson. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.

[57] Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-sgd: Biasing gradient descent into wide valleys. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124018, 2019.

[58] Raymond A Yeh, Chen Chen, Teck Yian Lim, Alexander G Schwing, Mark Hasegawa-Johnson, and Minh N Do. Semantic image inpainting with deep generative models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5485–5493, 2017.

[59] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Free-form image inpainting with gated convolution. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4471–4480, 2019.

[60] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Globally and locally consistent image completion. *ACM Transactions on Graphics (ToG)*, 36(4):107, 2017.

[61] Sangwoo Mo, Minsu Cho, and Jinwoo Shin. Instance-aware image-to-image translation. In *International Conference on Learning Representations*, 2019.

[62] Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jung Kwon Lee, and Jiwon Kim. Learning to discover cross-domain relations with generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1857–1865. JMLR. org, 2017.

[63] Antonia Creswell, Yumnah Mohamied, Biswa Sengupta, and Anil A Bharath. Adversarial information factorization. *arXiv preprint arXiv:1711.05175*, 2017.

[64] Lynton Ardizzone, Jakob Kruse, Carsten Lüth, Niels Bracher, Carsten Rother, and Ullrich Köthe. Conditional invertible neural networks for diverse image-to-image translation. In *DAGM German Conference on Pattern Recognition*, pages 373–387. Springer, 2020.

[65] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan++: How to edit the embedded images? In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8296–8305, 2020.

[66] Eunhye Lee, Jeongmu Kim, Jisu Kim, and Tae Hyun Kim. Restore from restored: Single-image inpainting. *arXiv preprint arXiv:2102.08078*, 2021.

[67] Zinovi Tauber, Ze-Nian Li, and Mark S Drew. Review and preview: Disocclusion by inpainting for image-based rendering. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 37(4):527–540, 2007.

[68] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. In *ACM Transactions on Graphics (ToG)*, volume 28, page 24. ACM, 2009.

[69] Yao Hu, Debing Zhang, Jieping Ye, Xuelong Li, and Xiaofei He. Fast and accurate matrix completion via truncated nuclear norm regularization. *IEEE transactions on pattern analysis and machine intelligence*, 35(9):2117–2130, 2013.

[70] Guim Perarnau, Joost Van De Weijer, Bogdan Raducanu, and Jose M Álvarez. Invertible conditional gans for image editing. *arXiv preprint arXiv:1611.06355*, 2016.

[71] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. Autoencoding beyond pixels using a learned similarity metric. In *International conference on machine learning*, pages 1558–1566, 2016.

[72] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[73] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*, 2015.

[74] Augustus Odena, Christopher Olah, and Jonathon Shlens. Conditional image synthesis with auxiliary classifier gans. In *International conference on machine learning*, pages 2642–2651. PMLR, 2017.

[75] Zhou Wang, Alan C Bovik, Hamid R Sheikh, Eero P Simoncelli, et al. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[76] Patricia Vitoria, Joan Sintes, and Coloma Ballester. Semantic image inpainting through improved wasserstein generative adversarial networks. *arXiv preprint arXiv:1812.01071*, 2018.

[77] Yi Wang, Xin Tao, Xiaojuan Qi, Xiaoyong Shen, and Jiaya Jia. Image inpainting via generative multi-column convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 331–340, 2018.

[78] Ang Li, Jianzhong Qi, Rui Zhang, Xingjun Ma, and Kotagiri Ramamohanarao. Generative image inpainting with submanifold alignment. *arXiv preprint arXiv:1908.00211*, 2019.

[79] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *NIPS*, 2017.

[80] Guillaume Lample, Neil Zeghidour, Nicolas Usunier, Antoine Bordes, Ludovic Denoyer, and Marc'Aurelio Ranzato. Fader networks: Manipulating images by sliding attributes. In *NIPS*, 2017.

[81] Shuyang Gu, Jianmin Bao, Hao Yang, Dong Chen, Fang Wen, and Lu Yuan. Mask-guided portrait editing with conditional gans. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3436–3445, 2019.

[82] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2337–2346, 2019.

[83] Yi Wei, Zhe Gan, Wenbo Li, Siwei Lyu, Ming-Ching Chang, Lei Zhang, Jianfeng Gao, and Pengchuan Zhang. Maggan: High-resolution face attribute editing with mask-guided generative adversarial network. In *Proceedings of the Asian Conference on Computer Vision*, 2020.

[84] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[85] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[86] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017.

[87] Ricard Durall, Franz-Josef Pfreundt, and Janis Keuper. Local facial attribute transfer through inpainting. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 95–102. IEEE, 2021.

[88] Jireh Jam, Connah Kendrick, Vincent Drouard, Kevin Walker, Gee-Sern Hsu, and Moi Hoon Yap. R-mnet: A perceptual adversarial network for image inpainting. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2714–2723, 2021.

[89] Mahdi M. Kalayeh, Boqing Gong, and Mubarak Shah. Improving facial attribute prediction using semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[90] Sergio Benini, Khalil Khan, Riccardo Leonardi, Massimo Mauro, and Pierangelo Migliorati. Fasseg: A face semantic segmentation repository for face image analysis. *Data in brief*, 24, 2019.

[91] Qifeng Chen and Vladlen Koltun. Photographic image synthesis with cascaded refinement networks. In *Proceedings of the IEEE international conference on computer vision*, pages 1511–1520, 2017.

[92] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8798–8807, 2018.

[93] Hsin-Ying Lee, Hung-Yu Tseng, Jia-Bin Huang, Maneesh Singh, and Ming-Hsuan Yang. Diverse image-to-image translation via disentangled representations. In *Proceedings of the European conference on computer vision (ECCV)*, pages 35–51, 2018.

[94] Qi Mao, Hsin-Ying Lee, Hung-Yu Tseng, Siwei Ma, and Ming-Hsuan Yang. Mode seeking generative adversarial networks for diverse image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1429–1437, 2019.

[95] Taihong Xiao, Jiapeng Hong, and Jinwen Ma. Dna-gan: Learning disentangled representations from multi-attribute images. *arXiv preprint arXiv:1711.05415*, 2017.

[96] Zhenliang He, Wangmeng Zuo, Meina Kan, Shiguang Shan, and Xilin Chen. Attgan: Facial attribute editing by only changing what you want. *IEEE Transactions on Image Processing*, 28(11):5464–5478, 2019.

[97] Po-Wei Wu, Yu-Jing Lin, Che-Han Chang, Edward Y Chang, and Shih-Wei Liao. Relgan: Multi-domain image-to-image translation via relative attributes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5914–5922, 2019.

[98] Taihong Xiao, Jiapeng Hong, and Jinwen Ma. Elegant: Exchanging latent encodings with gan for transferring multiple face attributes. In *Proceedings of the European conference on computer vision (ECCV)*, pages 168–184, 2018.

[99] Ying-Cong Chen, Xiaohui Shen, Zhe Lin, Xin Lu, I Pao, Jiaya Jia, et al. Semantic component decomposition for face attribute manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9859–9867, 2019.

[100] Bo Zhao, Bo Chang, Zequn Jie, and Leonid Sigal. Modular generative adversarial networks. In *Proceedings of the European conference on computer vision (ECCV)*, pages 150–165, 2018.

[101] Albert Pumarola, Antonio Agudo, Aleix M Martinez, Alberto Sanfeliu, and Francesc Moreno-Noguer. Ganimation: Anatomically-aware facial animation from a single image. In *Proceedings of the European conference on computer vision (ECCV)*, pages 818–833, 2018.

[102] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1501–1510, 2017.

[103] Ming-Yu Liu, Xun Huang, Arun Mallya, Tero Karras, Timo Aila, Jaakko Lehtinen, and Jan Kautz. Few-shot unsupervised image-to-image translation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10551–10560, 2019.

[104] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

[105] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? In *International conference on machine learning*, pages 3481–3490. PMLR, 2018.

[106] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *arXiv preprint arXiv:1706.08500*, 2017.

[107] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018.

[108] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[109] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[110] Gary B Huang, Marwan Mattar, Tamara Berg, and Eric Learned-Miller. Labeled faces in the wild: A database forstudying face recognition in unconstrained environments. In *Workshop on faces in'Real-Life'Images: detection, alignment, and recognition*, 2008.

[111] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4690–4699, 2019.

[112] Ricard Durall, Franz-Josef Pfreundt, and Janis Keuper. Semi few-shot attribute translation. *arXiv preprint arXiv:1910.03240*, 2019.

[113] Ricard Durall, Jireh Jam, Dominik Strassel, Moi Hoon Yap, and Janis Keuper. Facialgan: Style transfer and attribute manipulation on synthetic faces. *arXiv preprint arXiv:2110.09425*, 2021.

[114] Francesco Marra, Diego Gragnaniello, Davide Cozzolino, and Luisa Verdoliva. Detection of gan-generated fake images over social networks. In *2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*, pages 384–389. IEEE, 2018.

[115] Darius Afchar, Vincent Nozick, Junichi Yamagishi, and Isao Echizen. Mesonet: a compact facial video forgery detection network. In *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 1–7. IEEE, 2018.

[116] Ning Yu, Larry S Davis, and Mario Fritz. Attributing fake images to gans: Learning and analyzing gan fingerprints. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7556–7566, 2019.

[117] Brian Dolhansky, Russ Howes, Ben Pflaum, Nicole Baram, and Cristian Canton Ferrer. The deepfake detection challenge (dfdc) preview dataset. *arXiv preprint arXiv:1910.08854*, 2019.

[118] Brian Dolhansky, Joanna Bitton, Ben Pflaum, Jikuo Lu, Russ Howes, Menglin Wang, and Cristian Canton Ferrer. The deepfake detection challenge (dfdc) dataset. *arXiv preprint arXiv:2006.07397*, 2020.

[119] Xunyu Pan, Xing Zhang, and Siwei Lyu. Exposing image splicing with inconsistent local noise variances. In *2012 IEEE International Conference on Computational Photography (ICCP)*, pages 1–10. IEEE, 2012.

[120] Pasquale Ferrara, Tiziano Bianchi, Alessia De Rosa, and Alessandro Piva. Image forgery localization via fine-grained analysis of cfa artifacts. *IEEE Transactions on Information Forensics and Security*, 7(5):1566–1577, 2012.

[121] Tiago José De Carvalho, Christian Riess, Elli Angelopoulou, Helio Pedrini, and Anderson de Rezende Rocha. Exposing digital image forgeries by illumination color classification. *IEEE Transactions on Information Forensics and Security*, 8(7):1182–1194, 2013.

[122] Davide Cozzolino, Diego Gragnaniello, and Luisa Verdoliva. Image forgery localization through the fusion of camera-based, feature-based and pixel-based techniques. In *2014 IEEE International Conference on Image Processing (ICIP)*, pages 5302–5306. IEEE, 2014.

[123] Peng Zhou, Xintong Han, Vlad I Morariu, and Larry S Davis. Two-stream neural networks for tampered face detection. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1831–1839. IEEE, 2017.

[124] Davide Cozzolino and Luisa Verdoliva. Noiseprint: a cnn-based camera model fingerprint. *IEEE Transactions on Information Forensics and Security*, 2019.

[125] Miles Brundage, Shahar Avin, Jack Clark, Helen Toner, Peter Eckersley, Ben Garfinkel, Allan Dafoe, Paul Scharre, Thomas Zeitzoff, Bobby Filar, et al. The malicious use of artificial intelligence: Forecasting, prevention, and mitigation. *arXiv preprint arXiv:1802.07228*, 2018.

[126] Yuezun Li, Ming-Ching Chang, and Siwei Lyu. In ictu oculi: Exposing ai generated fake face videos by detecting eye blinking. *arXiv preprint arXiv:1806.02877*, 2018.

[127] Xin Yang, Yuezun Li, and Siwei Lyu. Exposing deep fakes using inconsistent head poses. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8261–8265. IEEE, 2019.

[128] Scott McCloskey and Michael Albright. Detecting gan-generated imagery using color cues. *arXiv preprint arXiv:1812.08247*, 2018.

[129] Haodong Li, Bin Li, Shunquan Tan, and Jiwu Huang. Detection of deep network generated images using disparities in color components. *arXiv preprint arXiv:1808.07276*, 2018.

[130] Chih-Chung Hsu, Chia-Yen Lee, and Yi-Xiu Zhuang. Learning to detect fake face images in the wild. In *2018 International Symposium on Computer, Consumer and Control (IS3C)*, pages 388–391. IEEE, 2018.

[131] David Güera and Edward J Delp. Deepfake video detection using recurrent neural networks. In *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6. IEEE, 2018.

[132] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152. ACM, 1992.

[133] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[134] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[135] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[136] Yitzhak Katznelson. *An introduction to harmonic analysis*. Cambridge University Press, 2004.

[137] Andreas Rössler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. FaceForensics++: Learning to detect manipulated facial images. In *International Conference on Computer Vision (ICCV)*, 2019.

[138] 100,000 faces generated. `https://generated.photos/`.

[139] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

[140] Naveen Kodali, Jacob Abernethy, James Hays, and Zsolt Kira. On convergence and stability of gans. *arXiv preprint arXiv:1705.07215*, 2017.

[141] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2794–2802, 2017.

[142] Ricard Durall, Margret Keuper, and Janis Keuper. Watch your up-convolution: Cnn based generative deep neural networks are failing to reproduce spectral distributions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7890–7899, 2020.

[143] Steffen Jung and Margret Keuper. Spectral distribution aware image generation. *arXiv preprint arXiv:2012.03110*, 2020.

[144] Michael Bacon, Robert Simm, and Terence Redshaw. *3-D seismic interpretation*. Cambridge University Press, 2007.

[145] Pablo Guillen, German Larrazabal*, Gladys González, Dainis Boumber, and Ricardo Vilalta. Supervised learning to detect salt body. In *SEG Technical Program Expanded Abstracts 2015*, pages 1826–1829. Society of Exploration Geophysicists, 2015.

[146] Wei Xiong, Xu Ji, Yue Ma, Yuxiang Wang, Nasher M AlBinHassan, Mustafa N Ali, and Yi Luo. Seismic fault detection with convolutional neural network. *Geophysics*, 83(5):O97–O103, 2018.

[147] Xinming Wu, Luming Liang, Yunzhi Shi, and Sergey Fomel. FaultSeg3D: using synthetic datasets to train an end-to-end convolutional neural network for 3D seismic fault segmentation. *GEOPHYSICS*, 84(3):IM35–IM45, 2019.

[148] Valentin Tschannen, Matthias Delescluse, Norman Ettrich, and Janis Keuper. Extracting horizon surfaces from 3d seismic data using deep learning. *GEOPHYSICS*, 85(3):N17–N26, 2020.

[149] Yunzhi Shi, Xinming Wu, and Sergey Fomel. Saltseg: Automatic 3d salt segmentation using a deep convolutional neural network. *Interpretation*, 7(3):SE113–SE122, 2019.

[150] Jack Hoyes and Thibaut Cheret. A review of “global” interpretation methods for automated 3d horizon picking. *The Leading Edge*, 30(1):38–47, 2011.

[151] Ricard Durall, Valentin Tschannen, Franz-Josef Pfreundt, and Janis Keuper. Synthesizing seismic diffractions using a generative adversarial network. In *SEG Technical Program Expanded Abstracts 2020*, pages 1491–1495. Society of Exploration Geophysicists, 2020.

[152] Xintao Chai, Genyang Tang, Shangxu Wang, Kai Lin, and Ronghua Peng. Deep learning for irregularly and regularly missing 3-d data reconstruction. *IEEE Transactions on Geoscience and Remote Sensing*, 2020.

[153] Yiding Jiang, Dilip Krishnan, Hossein Mobahi, and Samy Bengio. Predicting the generalization gap in deep networks with margin distributions. *arXiv preprint arXiv:1810.00113*, 2018.

[154] Daniel Jakubovitz, Raja Giryes, and Miguel RD Rodrigues. Generalization error in deep learning. In *Compressed Sensing and Its Applications*, pages 153–193. Springer, 2019.

[155] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[156] Yinling Guo, Suping Peng, Wenfeng Du, and Dong Li. Fault and horizon automatic interpretation by CNN: a case study of coalfield. *Journal of Geophysics and Engineering*, 17(6):1016–1025, 12 2020.

[157] Nam Pham, Sergey Fomel, and Dallas Dunlap. Automatic channel detection using deep learning. *Interpretation*, 7(3):SE43–SE50, 2019.

[158] Qun Liu, Lihua Fu, and Meng Zhang. Deep-seismic-prior-based reconstruction of seismic data using convolutional neural networks. *Geophysics*, 86(2):V131–V142, 2021.

[159] Sara Mandelli, Federico Borra, Vincenzo Lipari, Paolo Bestagini, Augusto Sarti, and Stefano Tubaro. Seismic data interpolation through convolutional autoencoder. In *SEG Technical Program Expanded Abstracts 2018*, pages 4101–4105. Society of Exploration Geophysicists, 2018.

[160] Valentin Tschannen, Norman Ettrich, Matthias Delescluse, and Janis Keuper. Detection of point scatterers using diffraction imaging and deep learning. *Geophysical Prospecting*, 68(3):830–844, 2020.

[161] Zhege Liu, Junxing Cao, Yujia Lu, Shuna Chen, and Jianli Liu. A seismic facies classification method based on the convolutional neural network and the probabilistic framework for seismic attributes and spatial classification. *Interpretation*, 7(3):SE225–SE236, 2019.

[162] Yang Xue, Mariela Araujo, Jorge Lopez, Kanglin Wang, and Gautam Kumar. Machine learning to reduce cycle time for time-lapse seismic data assimilation into reservoir management. *Interpretation*, 7(3):SE123–SE130, 2019.

[163] Gustavo Côrte, Jesper Dramsch, Hamed Amini, and Colin MacBeth. Deep neural network application for 4d seismic inversion to changes in pressure and saturation: Optimizing the use of synthetic training datasets. *Geophysical Prospecting*, 68(7):2164–2185, 2020.

[164] Harpreet Kaur, Nam Pham, and Sergey Fomel. Seismic data interpolation using cyclegan. In *SEG Technical Program Expanded Abstracts 2019*, pages 2202–2206. Society of Exploration Geophysicists, 2019.

[165] Yuanming Li, Bonhwa Ku, Shou Zhang, Jae-Kwang Ahn, and Hanseok Ko. Seismic data augmentation based on conditional generative adversarial networks. *Sensors*, 20(23):6850, 2020.

[166] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

[167] Evgeny Landa, V Shtivelman, and B Gelchinsky. A method for detection of diffracted waves on common-offset sections. *Geophysical prospecting*, 35(4):359–373, 1987.

[168] François Audebert, Pascal Froidevaux, Hery Rakotoarisoa, and Julie Svay-Lucas. Insights into migration in the angle domain. In *SEG Technical Program Expanded Abstracts 2002*, pages 1188–1191. Society of Exploration Geophysicists, 2002.

[169] Jan C Sørensen, Ulrik Gregersen, Morten Breiner, and Olaf Michelsen. High-frequency sequence stratigraphy of upper cenozoic deposits in the central and southeastern north sea areas. *Marine and Petroleum Geology*, 14(2):99–123, 1997.