T.R.

GEBZE TECHNICAL UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

LEARNING TO INPAINT IMAGES UISNG SCENE CONSTRAINTS

MOHAMED ABBAS HEDJAZİ

A THESIS SUBMITTED FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER ENGINEERING

GEBZE

2021

T.R.

GEBZE TECHNICAL UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

LEARNING TO INPAINT IMAGES USING SCENE CONSTRAINTS

MOHAMED ABBAS HEDJAZİ

A THESIS SUBMITTED FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER ENGINEERING

THESIS SUPERVISOR

ASSIST. PROF. DR. YAKUP GENÇ

GEBZE

2021

T.R.
GEBZE TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

SAHNE KISITLAMALARINI KULLANARAK GÖRÜNTÜLER TAMAMLAMA

MOHAMED ABBAS HEDJAZİ

DOKTORA TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

DANIŞMANI
DR. ÖĞR. ÜYESİ. YAKUP GENÇ

GEBZE
2021

GEBZE TEKNİK ÜNİVERSİTESİ	DOKTORA JÜRİ ONAY FORMU

GTÜ Fen Bilimleri Enstitüsü Yönetim Kurulu'nun ……/……./...…… tarih ve ...…. /…… sayılı kararıyla oluşturulan jüri tarafından ……/……./…… tarihinde tez savunma sınavı yapılan ……'ın tez çalışması …… Anabilim Dalında DOKTORA tezi olarak kabul edilmiştir.

JÜRİ

ÜYE (TEZ DANIŞMANI)	:	Dr. Öğr. Üyesi Yakup Genç
ÜYE	:	Prof. Dr. Yusuf Sinan Akgül
ÜYE	:	Dr. Öğr. Üyesi Ayşe Betül Oktay
ÜYE	:	Doç. Dr. Behçet Uğur Töreyin
ÜYE	:	Prof. Dr. Erchan APTOULA

ONAY

Gebze Teknik Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun ……/……./…… tarih ve ……/…….. sayılı kararı.

İMZA/MÜHÜR

SUMMARY

Image inpainting fills in the corrupted regions with plausible alternative contents. Recent GAN-based (Generative Adversarial Networks) inpainting methods show remarkable improvements over traditional ones. However, they assume the models implicitly learn the image structure and texture without enforcing constraints about the scene. Consequently, these models fail to capture object semantics, synthesize blurry texture details and produce significant artifacts on large masked regions due to GAN stability problems. Also, they employ large models requiring high computation time. This thesis proposes four methods to tackle these problems and complete the missing regions with the correct structure and fine-grained textures. Our first model exploits segmentation labels and edges to constrain image inpainting and reconstruct the object boundaries and the image structure. The second method stabilizes GAN training using four progressive generators and discriminators. To restore fine-grained detail, we use a texture-based loss function. The third method proposes a curriculum-style training approach to complete large regions. It increases the masked region size progressively in training time to stabilize GANs. Our final approach uses multi-resolution deep network paths to enlarge receptive fields and ensure low and high-level feature learning. It employs an adaptative weighting mechanism in the loss functions to focus on images exhibiting large masks and complex textures in the corrupted regions. We conduct our experiments on public datasets to validate our proposed methods. Results show that the proposed methods outperform state-of-the-art algorithms and speed up the inference time. We extend the experiments to other tasks, such as image outpainting and image blind inpainting.

Keywords: Image inpainting, Constrained learning, Deep learning, Generative adversarial networks.

ÖZET

Görüntü tamamlama teknikleri imgelerin bozuk kısımlarını uyumlu alternatif içeriklerle doldurur. Son zamanlardaki çekişmeli üretken ağlar (GAN) tabanlı görüntü tamamlama yöntemleri, geleneksel görüntü işleme yaklaşımlarına göre kayda değer gelişmeler göstermektedir. Bu teknikler, eğitimi yönlendirmek için girdi imge sahne hakkında herhangi bir kısıtlama kullanmaz. Böylece elde edilen modeller sahnedeki nesnelerin semantiğini yakalamada zorluk çekmekte ve özellikle bulanık doku ve yapısal bozukluklar içeren imgeler oluşturmaktadır. Aynı zamanda modellerin yer ve zaman karmaşıklığı yüksektir.. Bu tez, bahsedilen sorunları çözen kademeli dört yöntem önererek eksik bölgeleri yüksek sadakatli yapısal ve dokusal imgeler ile tamamlamaktadır. ilk modelimiz, görüntü tamamlamada kısıt için segmentasyon etiketlerinden ve kenarlardan yararlanır. Bu model nesne sınırlarını ve görüntü yapısını yeniden oluşturmaya yardımcı olmaktadır. İkinci model, dört aşamalı jeneratörleri ve diskriminatörleri kullanarak GAN eğitimi stabilize etmektedir. Görüntü ayrıntılarını yakalabilmek için bir yeni bir doku uyum fonksiyonu tasarlanmıştır. Üçüncü model, büyük bölgeleri tamamlamak için bir müfredat eğitimi yaklaşımı önermektedir. GAN'ları stabilize etmek için eğitim süresinde maskelenmiş bölge boyutunu aşamalı olarak artırır. Son modelimiz, lokal odağı genişletmek için farklı çözünürlüklü derin sinir ağlarını kullanır ve aynı anda düşük ve üst düzey özellik öğrenimi sağlar. Ayrıca, bozuk bölgelerde büyük maskeler ve karmaşık dokular sergileyen görüntülere odaklanmak için kayıp fonksiyonda uyarlanabilir bir ağırlık mekanizması kullanır. Geliştirilen modellerin doğrulanması için deneyler popüler veri kümeleri üzerinde yapılmıştır. Modellerin testi sonucunda en son çıkan yöntemlerden daha iyi performans göstermektedir ve daha hızıdır. İmge tamamlama işlemleri sadece iç bölgeler için değil aynı zamanda dış bölge tamamlama ve kör görüntü iç bölge tamamlama problemlerine de başarılı bir şekilde uygulanmıştır.

Anahtar Kelimeler: Görüntü tamamlama, Kısıtlı öğrenme, Derin öğrenme, Çekişmeli Üretici Ağlar.

ACKNOWLEDGEMENTS

I would like to express my sincerest gratitude to:

My advisor Dr. Yakup Genç, for his continuous support, expert advice and guidance throughout my Ph.D. study. He has supervised me with tremendous knowledge, valuable criticism and limitless patience. I will always appreciate his motivation that inspired me a lot.

The thesis jury members Prof. Dr. Yusuf Sinan Akgül and Dr. Ayşe Betül Oktay for their time, insightful comments, valuable feedback that improved the thesis quality and contributions.

The professors of the GTÜ computer engineering department. I enjoy all their lectures and benefit from their advice and experience that improved my skills and expanded my knowledge.

My research colleagues, for their collaboration, discussions, encouragement, support and social meetings.

The Turkish Scholarships organization (YTB) for the financial support, help and understanding during my Ph.D. study.

My parents and all the rest of my family, for their unconditional love and unwavering encouragement.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

Abbreviations		Descriptions
AR	:	Augmented Reality
BRISQUE	:	Blind/Reference-less Image Spatial Quality Evaluator
CA	:	Contextual Attention
CAM	:	Contextual Attention Mechanism
CL	:	Curriculum Learning
CNN	:	Convolutional Neural Network
Conv2D	:	2D Convolution
CPU	:	Central Processing Unit
CSA	:	Coherent Semantic Attention
DFNET	:	Diffusion Network
DL	:	Deep learning
DMRP	:	Deep Multi-resolution Paths
DNN	:	Deep Neural Network
DR	:	Diminished Reality
EC	:	Edge-Connect
ETMG	:	Efficient Texture-aware multi-GAN
FID	:	Fréchet Inception Distance
GAN	:	Generative Adversarial Network
GC	:	Gated Convolution
GFLOPS	:	Giga Floating-point Operations
GPU	:	Graphic Processing Unit
IoU	:	Intersection over Union
IS	:	Inception Score
LeakyReLU	:	Leaky Rectified Linear Unit
LBP	:	Local Binary Pattern
LSGAN	:	Least Squares GANs
M	:	Million
MAE	:	Mean Absolute Error
ms	:	Millisecond
MSE	:	Mean Squared Error
NIQE	:	Naturalness Image Quality Evaluator

NSGAN	:	Non-saturating GAN
PSNR	:	Peak Signal-to-Noise Ratio
ReLU	:	Rectified Linear Unit
SC	:	Scene Constraints
SF	:	Structure-Flow
SSIM	:	Structural Similarity Index Measure
TConv2D	:	Transposed 2D Convolution
VAE	:	Variational Auto-Encoder

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Diminished Reality (DR) is a computer vision field that eliminates, hides and sees through objects in the real world. In other words, it is the opposite of Augmented Reality (AR), which inserts virtual objects into the real world to keep the user with additional knowledge about the scene. In the last decade, DR problems have received a lot of attention, especially with the recent boom of deep learning methods that achieve immense success and build a new state-of-the-art in many complex computer vision problems [1].

Image completion or image inpainting is a DR technique that synthesizes plausible contents to fill in the missing regions or to remove unwanted objects/artifacts in an image (Figure 1.1). The estimated pixels should be coherent with the non-damaged parts (background) to ensure that the new contents are visually realistic and natural within the scene.

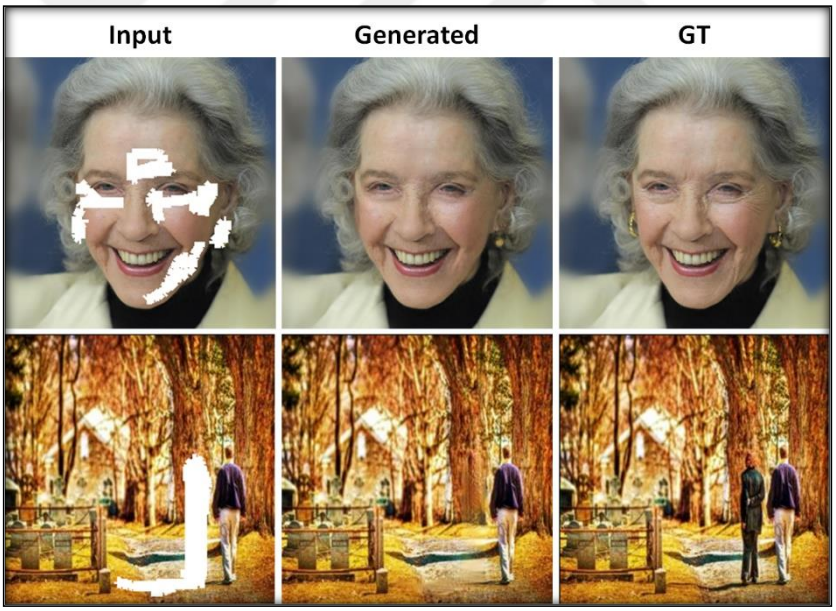

Figure 1.1: Example of image editing (first row) and object removal (second row).

Infilling is a fundamental part of human vision. Vertebrate eyes do not cover the whole visual field due to blind spots where optic nerves leave the eye. This spot does not contain any photo-receptor cells and does not contribute to the information flow of the scene. Our brains use the information from the peripheral area, such as texture, geometry and semantics, to fill in the missing parts [2]. Hence inpainting is easy for humans since they can understand the image structure representing the scene, even

when significant portions are invisible. However, this task is very challenging for a computer and involves an inverse problem.

To this end, image inpainting has attracted significant interest from computer vision and pattern recognition communities. It has a wide range of applications, including image editing [3,4], image restoration [5], object removal [6] and image compression [7].

Prior approaches in computer vision solve the inpainting problem by extracting low-level features, matching and pasting nearest-neighbor patches [8][9][10]. These methods generate promising results in repetitive textures and simple structure scenes. However, they fail to capture high-level information and usually present critical failures for images with non-repetitive patterns, such as faces and complex scenes.

Like many computer-vision tasks, image inpainting also took its share with the rapid advancements in deep learning. It exploits the fast progress of convolution neural networks (CNNs).

Current learning-based image inpainting methods [11][12][13][1] outperform traditional ones [8][9][10] both qualitatively and quantitatively using generative adversarial networks (GANs) [14]. They demonstrate a powerful tool to fill in the corrupted image with plausible alternative contents using a generator and a discriminator network. While the generator strives to synthesis images as close as possible to ground truth distribution, the discriminator distinguishes the real from generated ones.

1.1. Motivations

As mentioned previously, GANs-based image inpainting methods [11][12][15][3] learn high-level features from large-scale datasets to fill in corrupted regions. They establish a robust mapping between the corrupted and the ground truth images. However, most of the current approaches suffer from different problems, which inspire us to solve them in this thesis.

Initially, current methods do not put any constraints on the input and let the model decide what to generate. They assume that the model can implicitly acquire information from far spatial regions to synthesis new content. Consequently, most of them generate significant artifacts leading to distorted structures and non-realistic texture details [11] (Figure 1.2). These failures especially appear in the boundaries of

the objects where the model usually completes the masked regions using the predicted dominant background and does not preserve the semantics of the objects [16]. We can explain this by that most of the GAN-based approaches do not provide additional information to the model, such as image textures and semantics to constrain the inpainting problem.

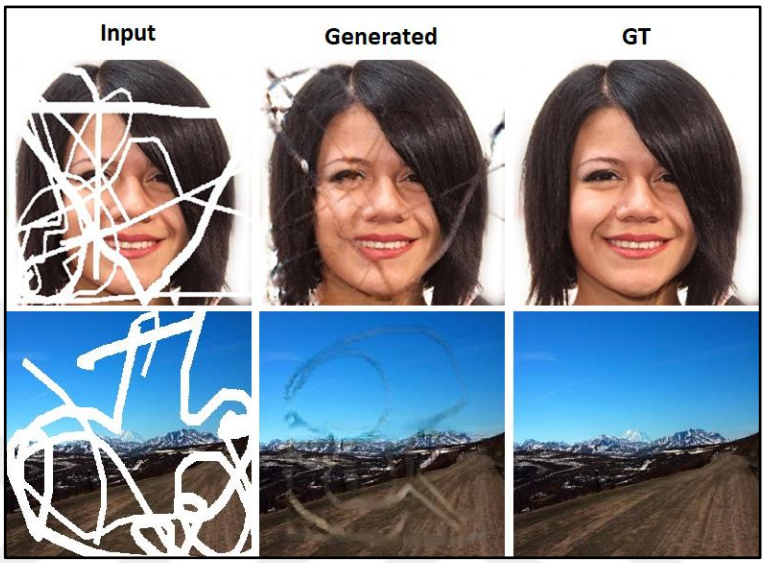

Figure 1.2: Example of artifacts produced by methods that do not enforce any input constraints.

Secondly, current GAN-based inpainting methods miss fine-grained textures in highly structured scenes [3] (Figure 1.3). They assume that image textures are learned using only reconstruction and adversarial supervision. However, this task is challenging without additional loss functions that enforce fine-grained textures. Moreover, all the mentioned methods use encoder-decoder architectures, where the encoder maps the corrupted image to a latent space, then the decoder reconstructs the ground truth image. This operation may not restore texture details due to small receptive fields.

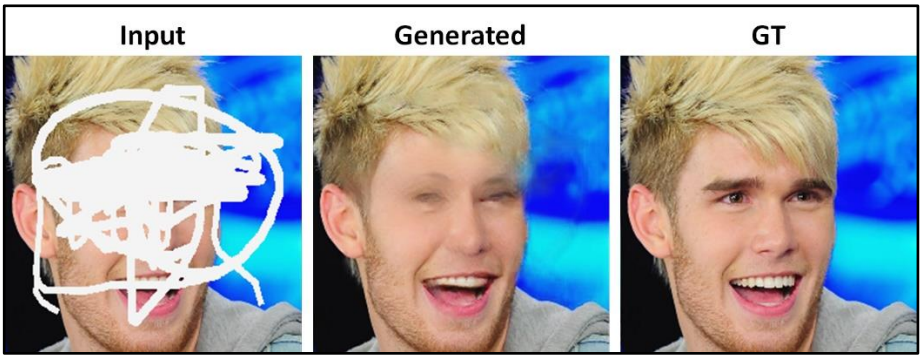

Figure 1.3: Example of a synthesized image with blurry textures.

Third, current methods encounter difficulties generating correct structures and colors when the image resolution and the masked region grow large [11] (Figure 1.4). This drawback is mainly due to the training instability of GANs that lead to mode collapse and over-fitting. In other words, although GANs fit the inpainting problem very well, it is challenging to train two networks in a cyclic manner where they compete against each other for totally different objectives.

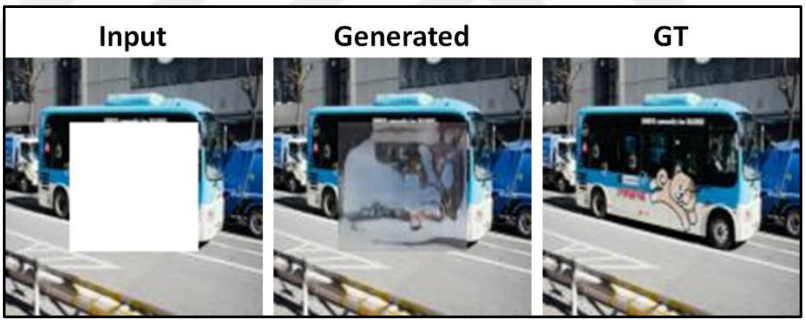

Figure 1.4: Example of structure and color inconsistency due to GAN instability.

Finally, current approaches require expensive computation hardware limiting many applications in resource-constrained environments. Their high computation cost is mainly due to the usage of large coarse-to-fine[1] models [3][17][16][18] and contextual attention mechanisms (CAM) [19][20]. The first technique applies a multi-stage training process that optimizes the parameters of two or more networks. Specifically, while the coarse stages estimate the initial image from the corrupted one,

[1] Coarse: initial prediction from the corrupted image. It contains fewer texture details. Fine: generated by the refinement network that enhances the coarse image to have global consistency and fine-grained textures.

the refinement stage uses the initially estimated image to generate a final plausible image. CAM borrows information from the surrounding parts to fill in corrupted regions. However, it still fails to ensure feature continuities and has high inference latency [21]. Also, training and inference on high-resolution images exponentially increase computation and memory resources. Consequently, the batch size becomes smaller, which decreases the performance of GANs [22].

Because of all the mentioned problems, image inpainting remains a challenging task in the computer vision field. In our Ph.D. thesis, we develop efficient image inpainting methods to synthesis semantically correct images with fine-grained textures.

1.2. Contributions

In this thesis, we conduct several studies on designing new network architectures, adding new constraints, introducing new loss functions and training approaches. Our contributions can be summarized as follows.

1.2.1. Image inpainting using scenes constraints

Most of the existing methods do not enforce any constraints to guide the image inpainting task. They either synthesis unrealistic and blurry texture details or fail to capture object semantics. Furthermore, they employ huge models with inefficient attention mechanisms. Motivated by these observations, we propose a new end-to-end generative-based architecture for image inpainting. Specifically, our model exploits the segmentation label estimations to robustly reconstruct the object boundaries and avoid blurry or semantically incorrect images. Subsequently, it estimates edges to recover the image structure details. Instead of predicting the segmentation labels/edges from the corrupted image, we exploit the coarse image that contains more valuable global structure data. To the best of our knowledge, our work is among the first to enforce both structure and texture constraints to restrict the inpainting task and generate images with realistic texture and correct semantic.

1.2.2. Efficient architecture for image inpainting

As mentioned in the motivations section, recent GAN-based inpainting methods show remarkable improvements and generate plausible images using multi-stage networks or CAM. However, these techniques increase the model complexity limiting their application in low-resource environments. Furthermore, they fail in generating high-resolution images with realistic texture details due to the GAN stability problem. Motivated by these observations, we introduce a new deep generative-based multi-resolution image inpainting framework to speed up the running time and improve performance. Our approach is composed of four successive efficient generators filling in four different resolutions. Specifically, the training starts with lower-resolution images and progressively doubles their size, such that their corresponding generators can exploit the previously inpainted regions. The latter shows a model stability improvement since training GANs on low-resolution images proves easier and converges faster. Another main problem with direct high-resolution image synthesis is that the discriminator will focus on texture details. Hence, it can easily reject synthesized images in the early training stages. Our approach drops the refinement module after the target resolution since it significantly increases the network size. We remedy the lack of this refinement stage by our proposed progressive training approach and a texture-based loss function.

1.2.3. New proposed loss functions for image inpainting

One of the challenging problems in image inpainting problems is unrealistic texture generation that usually leads to blurry and geometrically distorted results. To ensure fine-grained textures, we adopt an LBP-based (Local-binary-patterns) loss function to minimize the difference between the generated and the ground truth textures. LBP is a non-parametric texture descriptor that is widely used in many computer-vision tasks. During training, we minimize the distance between the ground truth LBP and the predicted one. In another study of this thesis, we observe that the images may include different texture complexity and mask sizes. Hence, we propose a new function that gives different weights for each image in the reconstruction and the adversarial losses. We enforce the generator to focus on images exhibiting large masks and complex textures in the corrupted regions.

1.2.4. GAN stabilization techniques

Curriculum learning is a technique that gradually reveals training samples to the model from the easiest to the most difficult. Inspired by this idea, we propose a curriculum-style strategy to progressively train an effective generator by growing the size of the masked regions in the context of image inpainting. In particular, the proposed method increases the masked region size progressively in training time. The intuition was that the generator and the discriminator networks solve the inpainting problem starting from easy to more difficult regions. By easy, we mean small and corrupted parts with basic structures that can be locally filled without the need for global object structures. However, large mask regions are troublesome to complete since they need local and global scene understanding. Incorporating such an approach in a GAN framework stabilizes the training, provides better color consistencies and captures object continuities. During test time, the user gives variable mask sizes and multiple holes at arbitrary locations. In another GAN stabilization technique, we propose a deep multi-resolution path architecture to robustly complete masked images. Specifically, we employ a deep network for each scale to increase the receptive field and recover high-frequency information from several input resolutions. Each path contains a deep network without downsampling to keep original image details. We concatenate the feature maps of previous and current network paths to ensure local and global image consistency.

1.3. Dissertation Outline

We organize the remainder of this thesis as follows. In Chapter 2, we introduce some core concepts about GANs and their variants. Besides, we review the most important traditional and deep learning-based image inpainting. We propose the image inpainting using scene constraints in Chapter 3. Efficient texture-aware multi-GAN for image inpainting is introduced in Chapter 4. We present the learning to inpaint by progressively growing the mask regions in Chapter 5 and the image inpainting using deep multi-resolution paths and adaptative loss functions in Chapter 6. We compare our proposed approaches against each other and propose a combination of them in Chapter 7. Finally, Chapter 8 concludes this thesis, discusses the limitations of our

methods and describes future works. The Appendix includes supplementary materials for Chapter 3, Chapter 4 and Chapter 7.

2. BACKGROUND

This chapter presents an overview of several concepts relevant to GANs within the context of the image inpainting task. We explore the related GANs-based architectures and loss functions in the first section. Traditional and current learning-based state-of-the-art image inpainting methods are discussed in the second section. We note that the reader needs to have a basic understanding of computer vision problems and terminologies. Also, we assume that deep learning techniques, such as feed-forward, convolutional neural networks, activation functions, optimizers, basic loss functions and other terminologies are known. We encourage the reader to learn the fundamentals of deep learning in [23].

2.1. Generative adversarial networks

Learning-based image inpainting methods optimize a deep encoder-decoder network to reconstruct the input image. The corrupted pixels values can be either zero, one or the mean pixel value of ImageNet [24]. However, encoder-decoder architectures produce blurry results and are often proceeded by a post-processing step [12]. Recently, GANs [14] are a great data distribution modeling technique. They achieve tremendous success in image synthesis and other complex computer vision problems, including video generation [25], image-to-image translation [26] and modulation classification [27]. The remainder of this section presents the most popular loss functions (NS-GAN [14] and LS-GAN [28]), training approaches (Progressive GAN) and architectures of GANs (Image-to-image translation).

2.1.1. NS-GANs

Proposed by [14], GAN is an emerging technique for learning complex data distributions. It consists of two networks competing against each other: the generator aims to generate real-looking images from a latent distribution (Gaussian). The discriminator tries to distinguish between real and synthesized images. The networks optimize different parameters simultaneously to reach their objectives. The generator learns indirectly from samples by updating its parameters based on the feedback of the discriminator (Figure 2.1). However, training GANs is very hard due to the

simultaneous optimization of different objectives and parameters. The discriminator and the generator update their parameters based on the loss functions described in (2.1) and (2.**2),** respectively.

$$L_{discriminator} = \min_{D} E_{x\sim\mathbb{P}_r}[-\log(D(x))] + E_{\mathrm{z}\sim\mathbb{N}(0,1)}[-\log(1 - D(\mathrm{G}(\mathrm{z})))] \quad (2.1)$$

$$L_{generator} = \min_{G} E_{\mathrm{z}\sim\mathbb{N}(0,1)}[-\log(D(\mathrm{G}(\mathrm{z})))] \quad (2.2)$$

Where:

- $\mathbb{P}_{\mathrm{r}}$ is the real data distribution.
- $\mathbb{N}(0,1)$ is the unit Gaussian distribution.
- D is the discriminator network (differentiable) function.
- G is the generator network (differentiable) function.

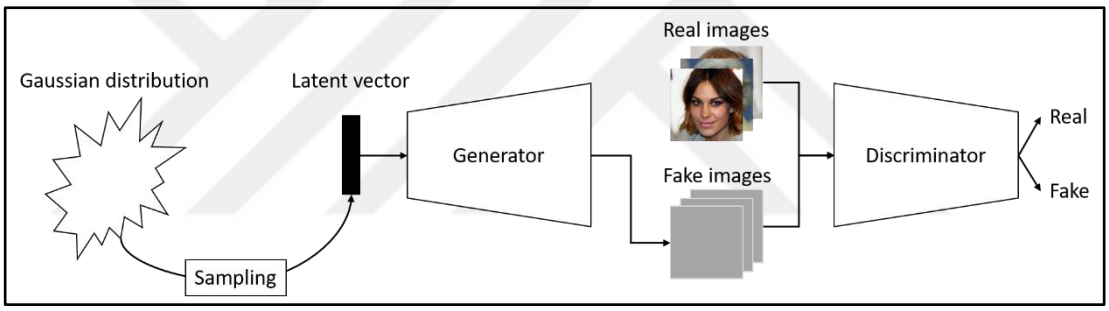

Figure 2.1: Generative adversarial network architecture.

2.1.2. LS-GAN

Despite its improvement in image synthesis over VAEs [29], generated samples of NS-GAN lack realism. This problem is due to the cross-entropy loss of the discriminator that leads to vanishing gradients. The latter occurs when the generator sees synthesized images on the correct side of the decision boundary but are still far from real data. Figure 2.2 illustrates the behaviors of the loss functions (NS-GAN [14] and LSGAN [28]), where the decision boundary needs to go across the real distribution for successful training.

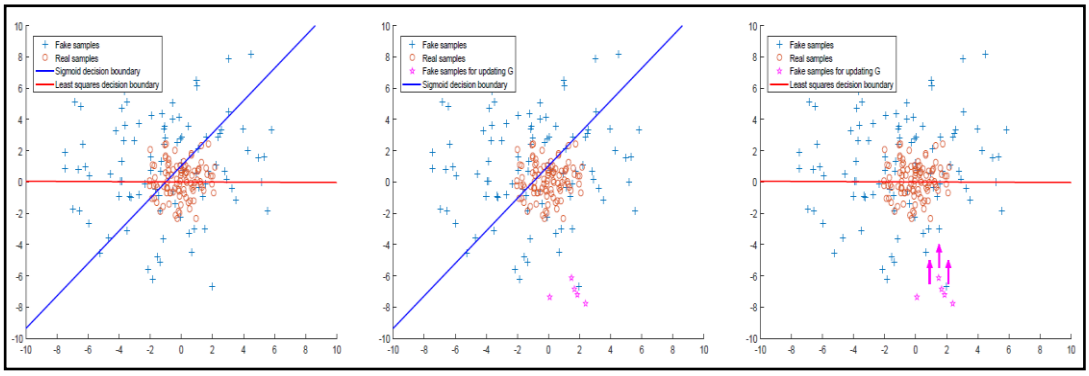

Figure 2.2: Illustration of different GAN losses.

The cross-entropy loss function gets a small error when updating the generator using fake samples (magenta-colored) because they are on the correct side of the decision boundary. In contrast, the LSGAN loss forces the generator to synthesize samples as close as possible to the decision boundary. The LSGAN loss functions for the discriminator and the generator update are defined in (2.3) and (2.4):

$$L_{discriminator} = \min_{D} E_{x\sim\mathbb{P}_r}[(D(x) - 1)^2] + E_{\mathrm{z}\sim\mathbb{N}(0,1)}[D(\mathrm{G}(\mathrm{z}))^2] \tag{2.3}$$

$$L_{generator} = \min_{G} E_{\mathrm{z}\sim\mathbb{N}(0,1)}[(D(\mathrm{G}(\mathrm{z})) - 1)^2] \tag{2.4}$$

[30] showed that NS-GAN minimizes the Jensen-Shannon divergence between real and fake data distribution, while the LSGAN minimizes the Pearson divergence. However, there is no correlation between the loss value and the training convergence, which means there is no way to know if the training is complete. Also, it suffers from mode collapse in which the generator ignores most of the variations in the data. A solution is to balance the generator and the discriminator training giving a lower bound on the loss to avoid mode collapse and apply some random noise to real images.

2.1.3. Progressive growing of GANs

[22] create a novel network architecture that generates high-resolution images of good quality with realistic textures. Furthermore, the training time is speed-up and become stable. The intuition is that synthesized images can be easily identified as fake by the discriminator. Hence the generator needs to generate samples of fine details, which is difficult in the early training stages. To approach this problem, they progressively increase image resolution by smoothly adding more layers to the

generator and the discriminator, as illustrated in Figure 2.3. This approach lets the generator learn the global structure of the image and progressively shift the distribution to finer details rather than learning everything at once.

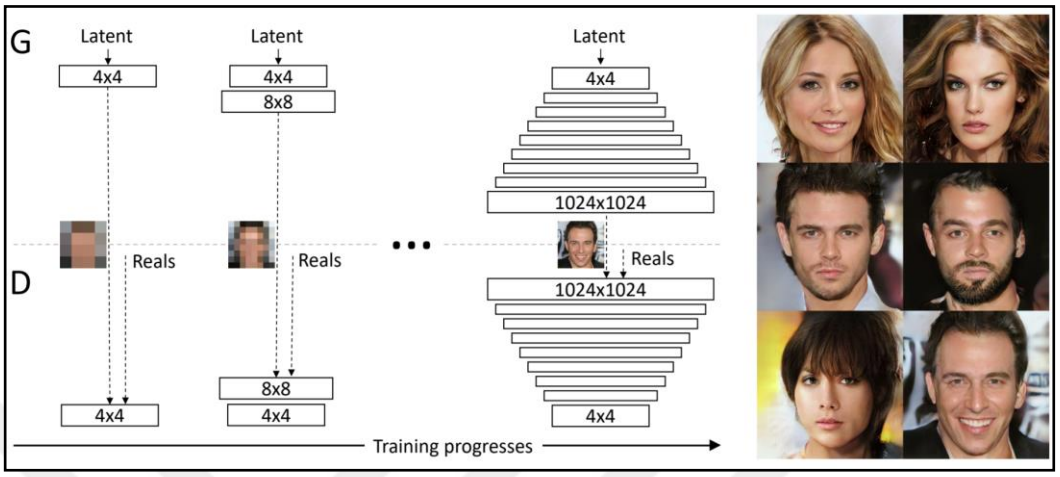

Figure 2.3: Progressive growing of GANs architecture.

2.1.4. Image-to-image translation using GANs

[26] used GANs in a conditional setting, where the input and the output are from image domains A and B, respectively. Examples include semantic maps to realistic photos, map to aerial images, sketch to faces, etc. They employ two loss functions: the first one is a reconstruction loss which minimizes the distance of the output, and the ground truth images of domain B using an L1 or L2 loss. The second one is an adversarial loss, where the discriminator distinguishes real from generated images. The latter pushes the distribution of the generator to the distribution of domain B. In other words, the discriminator acts as a customized loss function for the desired task leading to different results in different translation tasks. Figure 2.4 shows the image-to-image translation architecture, where the generator (G) takes the image from domain A and generates an image of domain B. The discriminator (D) takes real and fake images from domain B.

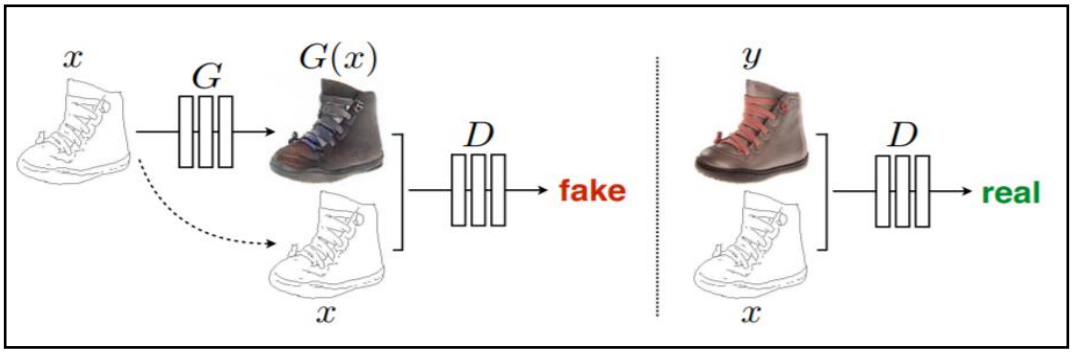

Figure 2.4: An image-to-image translation architecture.

2.2. Image inpainting methods

Image inpainting approaches in the literature can be principally divided into two major groups: conventional and learning-based. The former one includes diffusion-based and patch-based techniques that exploit the information around the masked region to fill in the corrupted image. The second group predicts the image structure and texture in an end-to-end manner using generative models.

2.2.1. Conventional image inpainting

[31] jointly interpolates the image gray levels and gradient directions to extend the isophote lines into the holes of missing data. [32] uses global image statistics like the histogram of local features to build a distribution, then inpaint the masked regions by finding the most probable image given the boundary and the distribution. These approaches manipulate narrow holes with simple textures. However, they generate visually significant artifacts and noisy patterns for large missing areas.

In contrast, patch-based methods handle relatively significant holes by extracting low-level features of the uncorrupted regions and searching the closest patches in a dataset to paste them into the missing pixels [10].

Patch-Match [8] is the first study that finds the approximate nearest neighbor matches between image patches. This approach has high memory and computation cost since it iteratively explores all the samples of the dataset. Subsequently, several methods are proposed to reduce memory usage, speed up the running time and improve the quality of the synthesized content. [9] generalizes the Patch-Match algorithm using more nearest neighbors, search across different scales and rotations, and matches many

descriptors and distance metrics. After that, image melding [33] integrates image gradients into the distance metric between patches. Despite the continuous efforts of the researcher of the computer vision community, patch-based approaches tend to produce significant artifacts [10]. They show global inconsistencies in complex scenes where the overlapped objects and non-repetitive patterns appear. We can explain this by that these methods do not understand the global semantic of the image since it does look for high-level features across patches. Furthermore, they cannot generate patches, which are not present in the image.

2.2.2. Deep learning inpainting

Recently deep learning-based (DL) methods have outperformed traditional methods and generate plausible images using GANs. These results come as no surprise since DL methods have improved solutions to many challenging computer vision tasks, including image-to-image translation [34][26] and object recognition [35]. In the context of image inpainting, learning-based methods benefit from the fast improvements of deep neural networks (DNNs) and GANs [14] to learn the image semantic from large-scale datasets [36][37][38]. These methods directly predict the missing pixel values using encoder-decoder architectures.

Context encoder [11] is one of the earliest methods that use GANs to complete rectangular masked regions in an image. However, the generated images of this approach lack global consistency and show many artifacts around the holes. To address this limitation, [12] extends it using a global and a local discriminator to ensure general image coherence and local image consistency. The drawback of this technique is the need for a post-processing step to guarantee the color coherency around square holes. It uses Poisson Image blending [39] that increases the usage of computational resources.

Another category replaces the postprocessing step with a refinement network that employs the CAM to learn features from image patches surrounding the missing region [19]. In other words, it explicitly attends to related feature patches at distant spatial locations to enhance semantic consistency. However, it exhibits blurriness and does not ensure pixel continuity around rectangular regions. This issue was addressed by [21] that can handle free form masks by adding a coherent semantic attention layer to the refinement network. However, this method is time-consuming since it performs

complex operations requiring high computational resources. [1] reduces the number of the parameters using a squeeze-and-excitation [40] residual network in both generator and discriminator. Besides, it proposes a joint context-awareness loss to generate more realistic textures. However, it misses fine-grained textures in large corrupted regions.

Other approaches handle irregular masks and address the artifacts problem without using adversarial learning. [15] uses a stack of partial convolution layers and mask updating steps to perform image inpainting using an autoencoder without adversarial learning. The intuition was that regular convolutions treat valid pixel values and masked values similarly, while partial convolutions are conditioned only on non-masked pixels. In other words, it employs an automatic mask updating mechanism of the partial convolution layers that eliminate substituting pixels and use only valid pixels. This architecture demonstrates the effectiveness of training image inpainting models on irregularly shaped holes. Yet, it exhibits artifacts leading to unrealistic textures and structures. [41] achieves competitive results using a fusion block that generates a flexible alpha composition map to combine corrupted and non-corrupted pixels. Also, it uses UNet [42] architecture embedded with the proposed fusion blocks to handle nonharmonic region boundaries. [13] employs global and local discriminators to build a fusion network that produces semantically coherent images.

Other recent methods apply curriculum learning (CL) techniques [43] to simplify the image inpainting task and stabilize the training of GANs. In another strategy, [44] divides the inpainting task into multiple phases under which each one fills in a part of the entire curriculum. Another work [38] gradually reconstructs the structure of the image (edges) with the corresponding visual features.

More recently, many-stage networks have been introduced to solve structure and texture problems. To reduce the complexity of the image inpainting and obtain realistic results, [45] utilizes multiple latent codes to describe the high dimensional distribution of the images. It divides the inpainting task into content inference and style imitation. [46] preserves the foreground objects in overlapped scenes using three stages: contour detection, contour completion and image completion. [17] divides the inpainting problem into two phases. The first one recovers the edges, and the second one uses that structure information to help the completion network to estimate the final image. [47][18] use the segmentation labels to guide the structure prediction of the missing region. [3] uses gated convolution layers in the coarse network and a CAM in the refinement network to generate the final image. GC is a hard-gating single-channel un-

learnable layer multiplied to input feature maps and has a dynamic feature selection mechanism for each channel and each spatial location. Also, they add edges as an additional input (a sketch) to the coarse network to give the user the ability to interactive editing.

We note that all the mentioned methods estimate the additional information from the corrupted image, which lacks global structure and may include semantically inconsistent content. Furthermore, they aim to recover the high-frequency information in the image or preserve the generated object boundaries without input constraints. [20] is also a coarse-to-fine architecture that uses a refinement generator with multi-scale discriminators to synthesize smooth images. However, attention layers significantly increase the computational complexity and speed down the inference time of [20][3]. Reducing the model size without affecting the quality of the generated images is desirable. In our thesis, we tackle this problem and build effective image inpainting methods that ensure fine-grained textures and correct structures. Meanwhile, the inference time should be reduced by eliminating complex operations such as CAM and perceptual losses.

2.3. Quality assessment techniques

In this section, we investigate several image quality assessment metrics: Fréchet Inception Distance (FID), Inception Score (IS), Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM).

2.3.1. Inception Score (IS)

The IS [48] metric measures the quality of the generated images based on their diversity and quality. The intuition is that when a human examines a photo, he/she can confidently determine what is in there (saliency). Also, when a human looks at a set of images would say that it has lots of objects (diversity). The overall equation is defined in (2.6), where the saliency is expressed by $p(y|x)$, meaning that the distribution of classes for any individual image should have low entropy. The diversity is expressed by $p(y)$, meaning that the overall distribution of classes across the generated images should have high entropy (absence of dominating classes). The KL divergence (2.5) is a measure of, on average, how different is the score distribution

for a generated image from the overall class balance. Higher scores are better, corresponding to a larger KL-divergence between the two distributions [49].

$$KL(P||Q) = \sum_i P(i) log \frac{P(i)}{Q(i)} \quad (2.5)$$

$$IS(X) = exp(E_x[KL(p(y|x)||p(y))]) \quad (2.6)$$

2.3.2. Fréchet Inception Distance (FID)

The FID [50] metric improves image assessment over the IS score by comparing the statistics of synthesized to real images. It uses the Fréchet distance between two multivariate Gaussians as defined in 2.7):

$$FID = \|\mu_r - \mu_g\|^2 + Tr(\sum_r + \sum_g - 2\sqrt{\sum_r \sum_g}) \quad (2.7)$$

Where: $X_r \sim \mathbb{N}(\mu_r, \sum_r)$ and $X_g \sim \mathbb{N}(\mu_g, \sum_g)$ are the 2048-dimensional activations of the Inception-v3 'pool3' layer for real and generated samples, respectively. Lower FID is better, corresponding to more similar real and generated samples as measured by the distance between the distributions of their activations [49].

2.3.3. PSNR

Peak signal-to-noise ratio (PSNR) is one of the widely used image quality metrics to evaluate compression and image inpainting methods. It calculates the ratio between the maximum possible powers of a signal and the power of distorted noise (2.8). If the reconstructed and the original image match, the score should be high and vice-versa.

$$PSNR(x, y) = 10 \times \log_{10}(\frac{255^2}{MSE(x, y)}) \quad (2.8)$$

2.3.4. SSIM

The Structural Similarity Index (SSIM) quantifies the image quality of the proceed image based on the ground truth by measuring the perceptual difference between them. Unlike MSE and PSNR that estimate pixel by pixel errors, SSIM considers the

dependencies between pixels based on their spatial locality, which expresses valuable information about the structure of the objects in scenes. Thus, it becomes one of the fundamental metrics for many applications, such as compression, deblurring and inpainting. SSIM is defined in (2.9).

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x + \mu_y + c_1)(\sigma_x + \sigma_y + c_2)} \tag{2.9}$$

2.4. Summary

We devote this chapter to a brief overview of deep learning techniques for image inpainting. In Section 2.1, we have mainly focused on the GANs architecture and loss functions, which are essential to understand the rest of the thesis. Extensive presentation and more details of the field can be found in this excellent book [24]. In Section 2.2, we have introduced deep learning-based image inpainting, and we explain the difference between them and traditional methods such as diffusion-based and patch-based. We discuss the advantages and drawbacks of the presented approaches and the remaining challenges in the field that will be addressed in Chapter 3, Chapter 4, Chapter 5 and Chapter 6. Although GANs fit the inpainting problem, they still have difficulties synthesizing coherent structures and fine-grained textures. In Section 2.3, we have presented the fundamental metrics to quantify the performance of image inpainting methods.

3. IMAGE INPAINTING USING SCENE CONSTRAINTS

3.1. Introduction

A robust image inpainting approach should generate images with correct structure and realistic texture details. Traditional methods extract low-level features from the valid pixels to match and paste patches [8][9][10]. However, these methods do not synthesize plausible content in complex scenes where non-repetitive patterns appear. Furthermore, they have a high computational overhead due to their iterative nature. On the other hand, early deep convolution learning-based methods capture the image semantic using GANs that map the corrupted image to a visually realistic one in an end-to-end manner.

Although learning-based methods achieve plausible results, they suffer from texture artifacts and structure preservation problems. These failures especially appear in the boundaries of the objects where the model usually completes the masked regions using the predicted dominant background and does not preserve the semantics of the objects [16]. We can explain this by that most of the current methods assume that the image texture and semantic can be learned implicitly by CNNs without any further supervision. Therefore, they do not provide any additional information to the model.

Recent GAN-based approaches address these limitations by breaking down the inpainting problem into two or more stages, and they provide additional information to the model. [17][3][16] divides the inpainting problem into two tasks. The first one recovers the edges from the corrupted image, and the second stage estimates the fine image. However, the edges are not enough to ensure realistic colors and boundaries. [47] uses the segmentation labels to guide the structure prediction of the missing regions. Using only this information is not sufficient since one segmentation label can represent several textures. [46] employs an image contour network to preserve the foreground objects in overlapped scenes. These approaches still suffer from a lack of combined high-frequency information in the missing regions.

While the previously mentioned methods focus on using a single aspect of the image, we predict both the segmentation labels and the edges to use them as additional constraints to help the network to generate the correct structure and texture in the final image. In other words, we use the segmentation labels to guide the generator to

complete the boundaries and understand the object classes. Furthermore, we use edges to provide high-frequency information in structured scenes. Unlike [18][47], we predict segmentation labels from the coarse image that contains more accurate structure information than the corrupted image. Furthermore, we use both the coarse image and segmentation labels to estimate the edges, which is not the case in [17] that predicts it directly from the corrupted image. Finally, we merge the coarse image, the estimated segmentation labels, and edges information to generate the fine-grained image. Experiments demonstrate that dividing the inpainting problem into multiple stages helps the networks to generate competitive results. To keep a small number of parameters, we use small networks for all the predictors. To stabilize our model and reduce the training time, we adopt the curriculum resizing technique [51] to image inpainting. The training process starts with small resolution images, then gradually increases the problem difficulty by feeding images with higher resolutions. To the best of our knowledge, our work is among the first to enforce both structure and texture constraints to restrict the inpainting task and generate images with realistic texture and correct semantic. We can summarize our contributions as follows:

- We combine the segmentation labels and the edges to explicitly constrain the image inpainting task and preserve the object structure and textures.
- We adopt the curriculum resizing technique to the image inpainting task to reduce the training time.
- We evaluate the proposed method on the MSCOCO [52] and the CelebHQ [53] datasets. The experiments prove that our model is computationally efficient than several state-of-the-art methods. Meanwhile, it can generate realistic images and obtain competitive performance.

3.2. Approach

We propose multi-stage modeling of image inpainting for a specified domain. Our approach successfully fills in missing parts of an image while estimating the underlying image constraints such as edges and segmentation labels. These constraints help in describing the scene semantics.

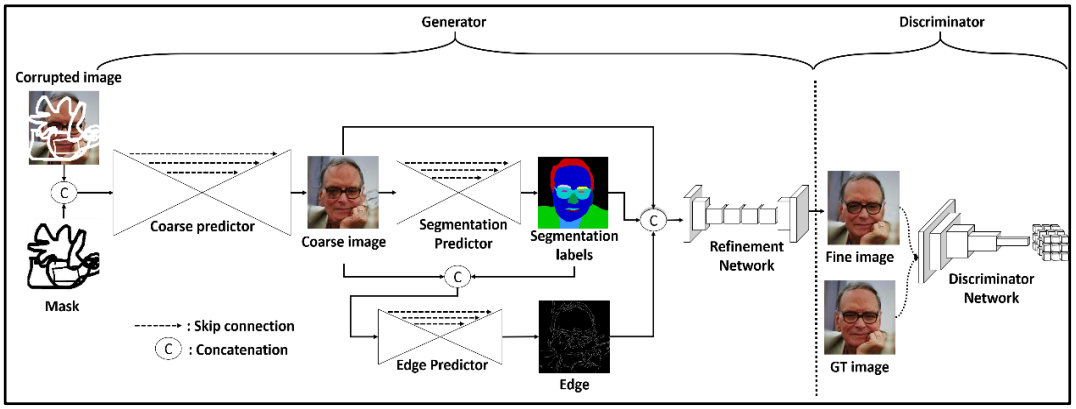

Figure 3.1: The overall architecture of the SC model.

3.2.1. Architecture

GAN-based image inpainting methods use a generator and a discriminator to fill in the missing regions in an image [14]. The generator is composed of four stages (Figure 3.1). The first one takes as input the corrupted image and the mask, then outputs a coarse image. The second estimator generates the image segmentation labels using the coarse image. The latter depicts the visual structure of the missing regions. Note that the ground truth segmentation labels are required to supervise the task. Alternatively, if not available, we can use a pretrained segmentation labels estimator. The third network estimates the edges in the image using the coarse image and the segmentation labels. Using the latter ones at this stage is intended to simplify the task for the edge estimator since segment boundaries tend to overlap with image edges. In the last phase, the refinement network employs the coarse image, the segmentation labels and edges to generate the final image.

We adopt the well-known UNet [42] architecture in the first three stages. We divide the number of parameters in each layer by 2 for the coarse image estimator and by 4 in the segmentation labels and edges estimators. We use a modified refinement network of [54][3] in the last stage. We set the dilation factors in the middle block to 1, 2, 3 and 4 to avoid the gridding problem. This problem is due to the use of the same dilation factor for all the successive convolution layers. The latter may cause discontinuous convolution kernels and inconsistency of local information [54]. We use dense connections between all the layers to reuse the previous features.

We remove the CAM layer since it increases the floating-point operations due to the high-resolution tensor matrix multiplications and SoftMax activation functions.

Furthermore, it can lead to semantic irrelevance resulting in pixel discontinuity of the generated regions. We investigate different architectures in the refinement network to find the best design that reduces the number of parameters without affecting the inpainting performance. We focus on this step to improve the model efficiency and decrease the memory resource requirements. We describe the architecture in Table 3.1. We use the PatchGAN discriminator [26] that outputs a tensor of real and fake values to criticize different patches in the image.

Table 3.1: Architecture of the SC refinement network.

Block	Layer	Dim	Kernel	Stride	Dilation
Encoder	Conv2D	32	3×3	1	1
	Conv2D	64	4×4	2	1
	Conv2D	128	4×4	2	1
	Conv2D	128	3×3	1	1
	Conv2D	128	3×3	1	1
Middle	Conv2D	128	3×3	1	1
	Conv2D	128	3×3	1	2
	Conv2D	128	3×3	1	3
	Conv2D	128	3×3	1	4
Decoder	Conv2D	128	3×3	1	1
	Conv2D	128	3×3	1	1
	Nearest2D	-	-	-	-
	Conv2D	64	3×3	1	1
	Nearest2D	-	-	-	-
	Conv2D	32	3×3	1	1
	Conv2D	3	3×3	1	1

3.2.2. Semantic maps and edges as constraints

Most of the GAN-based inpainting methods assume that the network can implicitly learn the image structure and texture, which is a challenging optimization problem for neural networks. Recent approaches provide additional information such as the segmentation labels [47][18] and the edges [3][17] to explicitly guide the network to generate structurally plausible content with realistic textures. However, these methods estimate the segmentation labels or edges from the corrupted image, which seem to wash away most of the information since several details and structures

disappear in the missing regions. Different from [47][18], we estimate the segmentation labels from the coarse image, which contains more semantically correct content. We estimate the edges from the segmentation labels and the coarse image. This composition helps edge estimation since the coarse image contains better global information structures than the corrupted image. Meanwhile, the segmentation labels identify many fundamental edges in the scenes. Our network exploits all the previous predictions to synthesis high-quality images with global semantic consistency, smooth boundaries, and realistic texture details.

3.2.3. Curriculum resizing

Recent inpainting methods use CL techniques to stabilize the GAN training and reduce the learning difficulty in large holes by progressively growing the mask regions [55], the image structure [38] and the fine image [56]. However, all the mentioned methods train the models on high-resolution images, which drastically increases the training time. Furthermore, training GANs on low-resolution images proves to be stable [53]. Inspired by this observation, we adopt the curriculum resizing technique to the image inpainting task to decrease the training time without much effect on the inpainting performance. Most of the image inpainting training methods employ only high-resolution images to learn the mapping function between the corrupted image and ground truth image. In our work, we propose to divide the training into several steps, such that each one trains the model on a specific resolution. As seen in Figure 3.2, we assign most of the training time to smaller images to speed up training and reduce the usage of computational resources. We effectively estimate missing regions by training on low-resolution images to increase the receptive field of convolution filters. However, on high-resolution images, the discriminator network can easily distinguish high-frequency information of ground truth samples from the generated ones leading to an easy rejection and unstable training. In contrast, low-resolution images do not contain many details avoiding GAN failure and mode collapse. Since our network is fully convolutional, we do not have any problem feeding different image sizes.

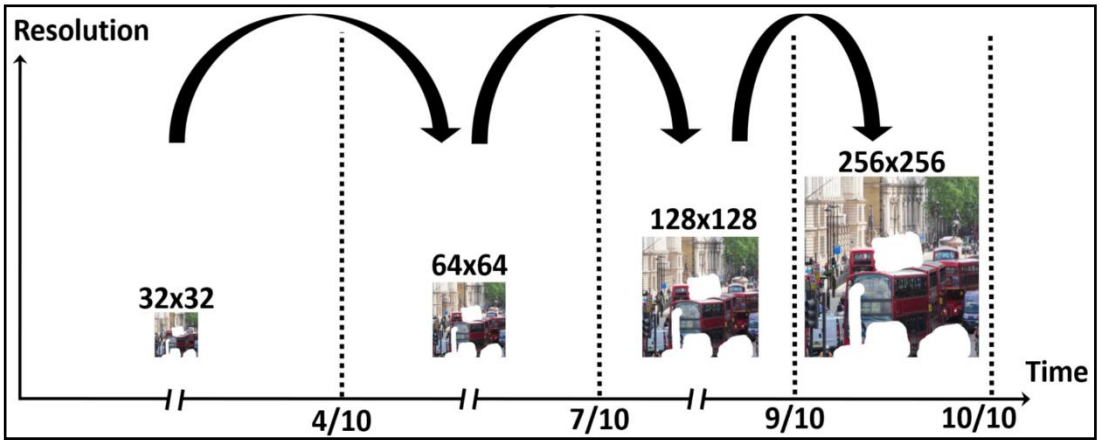

Figure 3.2: Illustration of the progressive resizing approach.

3.2.4. Loss functions

Let $I_g, I_{m,}$ and $I_{e,}$ be the ground truths for the image, the mask, the segmentation labels, and the edge, respectively. Also, let $G_c(.), G_s(.), G_e(.)\ and\ G_f(.)$ be the coarse image estimator, the segmentation labeler, the edge estimator, and the refinement network, respectively. The coarse image O_c, the segmentation labels O_s, the edges O_e and the fine image O_f are predicted using 3.1), 3.2), 3.3), and 3.4), respectively.

$$O_c = G_c\left(I_g \times I_{m,} I_{m,}\right) \quad 3.1)$$

$$O_s = G_s(O_c) \quad 3.2)$$

$$O_e = G_e(O_c\ , O_s) \quad 3.3)$$

$$O_f = G_f(O_c\ , O_s, O_e) \quad 3.4)$$

L1 loss: we measure the error between the ground truth image and the predictions in (3.5) :

$$L_{rec} = \left\|O_s - I_g\right\|_1 + \|O_s - I_z\|_1 + \|O_e - I_e\|_1 + \left\|O_f - I_g\right\|_1 \quad (3.5)$$

Adversarial loss: we optimize the LSGAN [28] adversarial loss for each resolution as defined in (3.**6**).

$$L_{dis} = \mathrm{E}\,[(\mathrm{D}(I_g) - 1)^2] + \mathrm{E}[\mathrm{D}(O_f)^2]$$
$$L_{adv} = \mathrm{E}\,[(\mathrm{D}(O_f) - 1)^2] \quad (3.6)$$

Overall loss: we use a weighted sum of the reconstruction and the adversarial. The choice of hyper-parameters in the loss function plays a significant role in the improvement of the image inpainting performance. We select the weights empirically by conducting a set of experiments and changing the hyper-parameters for each loss component. We give a weight γ_{adv}=0.1, γ_{rec}==1 for the adversarial loss and the reconstruction loss, respectively. The overall loss is defined in (3.7):

$$L_{overall} = \gamma_{adv} \times L_{adv} + \gamma_{rec} \times L_{rec} \quad (3.7)$$

3.3. Experimental evaluation

3.3.1. Datasets and masks

We experiment on two datasets that contain images with their corresponding segmentation labels. The first one is CelebAMask-HQ [53][57] which has 30K highly quality faces of size 1024×1024 with a large variation in facial characteristics such as the shape and the color. Its segmentation labels are 19 classes, including facial associates and components such as skin, nose, eyes, etc. The second dataset is the MSCOCO [52] that has 118K training images and 5K test images with 80 different categories. This dataset is very suitable for our case since it contains crowded scenes with a high change in texture and color. Most of the users of image inpainting applications interact using arbitrary shapes. Hence, recent deep learning-based inpainting methods [3][17][47][46] utilize irregular mask sizes to train their models. We use [58] to train our model, which contains random variable masks covering diverse parts of the image. In evaluation time, our masks are divided into four categories covering $10-20\%$, $20-30\%$, $30-40\%$ and $40-50\%$ of the image.

3.3.2. Implementation details

In this part, we describe in detail our training procedure and hyperparameters settings. We use Pytorch to implement the proposed method using CUDA v10.1 and

cuDNN v7.6.4. We train the model in an end-to-end manner for 50 epochs using a batch size of 8. It takes three days to converge on a single NVIDIA TITAN X GPU. For updating weights, we use Adam optimizer [59] with hyperparameters $\alpha = 0.5$ and $\beta = 0.99$, respectively. We set the learning rate of the first three stages to 10^{-3} and $10^{-4}$for the refinement stage. Since GANs are very sensitive to high learning rates, we detach the coarse image, segmentation labels and edge predictions from their networks. Hence, the backpropagation from the adversarial loss does not affect the weights in their corresponding networks. After observing an improvement in the segmentation labels and the edge predictions, we decrease the learning rate of the first three stages to 10^{-4}, and we attach the coarse, segmentation labels and edge predictions. [17] shows that the σ hyper-parameters in the Canny edge detector have an important impact on the inpainting performance. We use $\sigma = 2$ based on their ablation studies.

3.4. Results and discussion

We qualitatively and quantitatively compare our approach against five recent image inpainting methods. Contextual Attention (CA) [19] uses a local-patch attention mechanism in the refinement network to improve the coarse image. Edge Connect (EC) [17] employs edges as additional information to guide the model to generate plausible structures. Gated Convolutions (GC) [3] produce promising results since it uses both the gated convolutions layers and edge information. Structure-flow (SF) [16] preserves robust structure and texture using edges predictor and appearance flow generator, respectively. Coherent Semantic Attention (CSA) [21] representing the semantic relevance between the missing region features. In the CelebAMask-HQ dataset [53][57], we use the pre-trained models of compared state-of-the-arts, which is another reason for selecting these approaches. For a fair comparison, we train all the models for 50 epochs on the MSCOCO [52] dataset. Furthermore, we use the same masks and test splits in all the experiments.

3.4.1. Qualitative results

To evaluate our method, we report qualitative comparison results on the MSCOCO dataset [52]. As seen in Figure 3.3, CA [19] generates significant artifacts

since it does not predict any additional information to guide the network. GC [3] and EC [17] can recover the global structures of the image, but they exhibit inconsistencies between the corrupted and the valid pixels. Despite their promising results, SF [16] and CSA [21] present color discontinuities in some cases. Our model generates both meaningful structures and realistic textures for missing regions. We can explain this by that the estimated segmentation labels contribute significantly to the generation of correct boundaries. In particular, estimating edges from the coarse image preserves more structures than estimating them from the corrupted image. Also, the combination of the two auxiliary information guides the model to synthesis plausible contents. To further evaluate our approach, we present Figure 3.4 that shows visual comparisons with the other state-of-the-art methods on the CelebAMask-HQ dataset [53][57]. The images generated by CA [19] suffer from artifacts and distortions. The performance of GC [3] is much better since it can synthesis smoother images. However, artifacts still exist around the borders of the face components (i.e., eyes, mouth, and nose). CSA [21] and our method show very competitive results and generate more natural images with fewer artifacts around the boundaries.

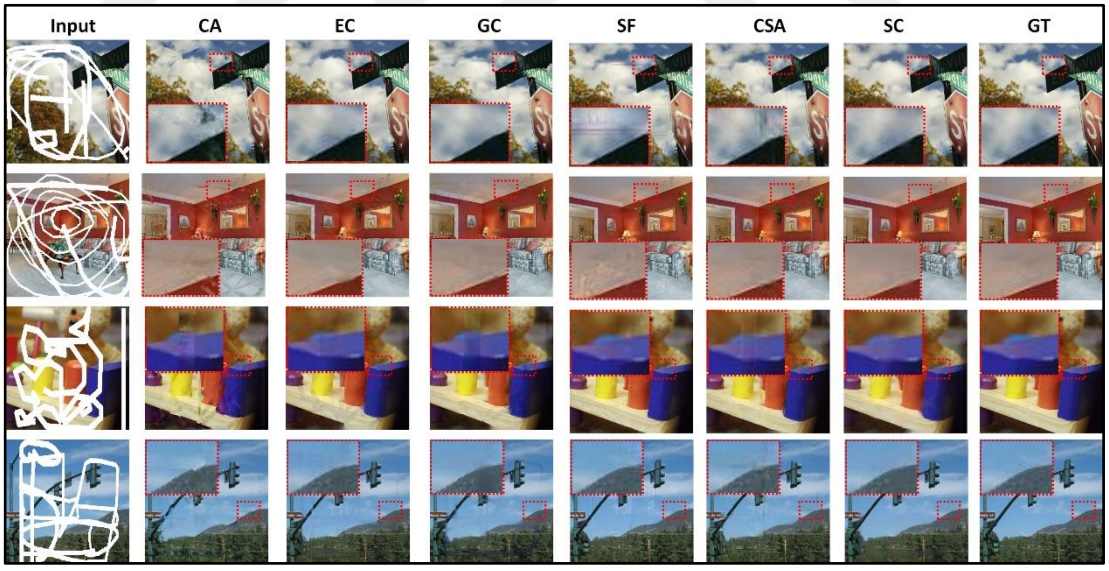

Figure 3.3: Qualitative comparison of our SC model on MSCOCO.

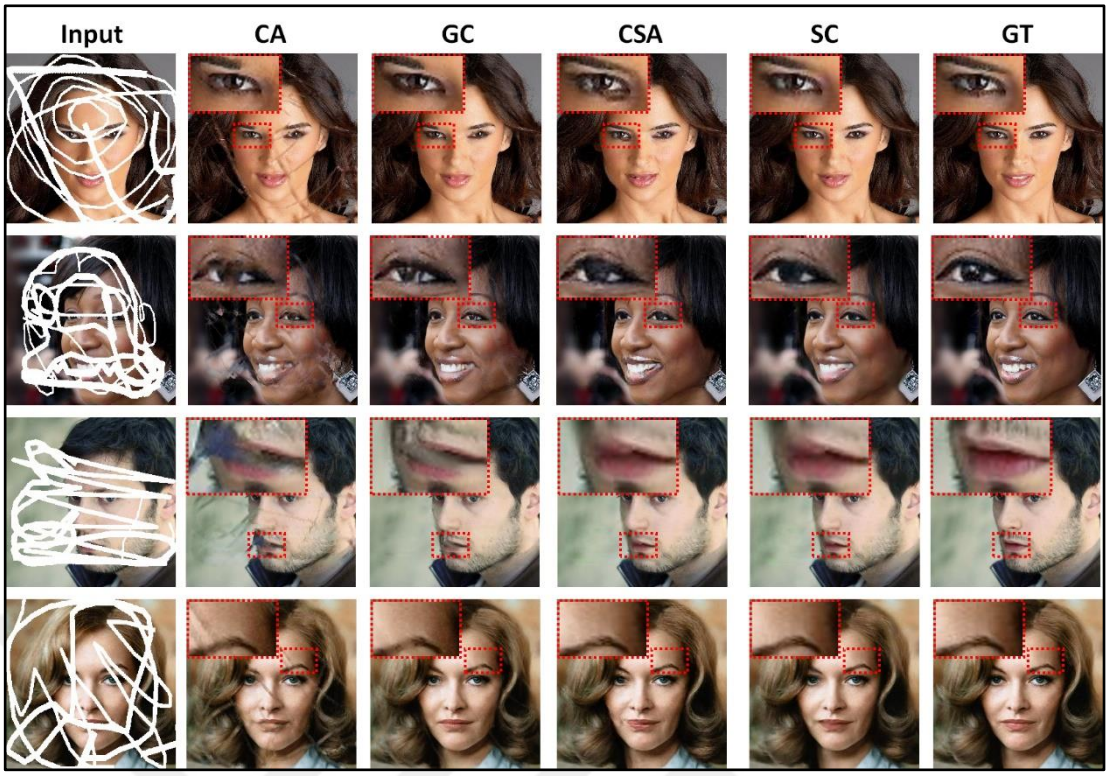

Figure 3.4: Qualitative comparison of our SC model on CelebHQ.

3.4.2. Quantitative results

It is well known that image inpainting tasks lack accurate evaluation metrics [19]. Nevertheless, to quantify the performance of the proposed approach, we use commonly used inpainting metrics, including MAE, SSIM and PSNR following [17][16]. Furthermore, we investigate some representative blind image quality metrics that evaluate the generated image without the ground truth. Precisely, they calculate the no-reference image quality score for an image using the Naturalness Image Quality Evaluator (NIQE) [60], the distortion Type Classification and Label Transfer (TCLT) [61] and the Blind/Reference-less Image Spatial Quality Evaluator (BRISQUE) [62]. As seen from Table 3.2, CA [19] performs the worst among the six methods. EC [17] exhibits better performance than CA in all the metrics. GC [3] and SF [16] achieve competitive results and very close scores. CSA [21] reports the best score in the compared state-of-the-art methods. Our approach outperforms all the state-of-the-art methods on all the metrics, and with different mask sizes ($10-20\%$, $20-30\%$, $30-40\%$ and $40-50\%$). We can explain this by the fact that combining segmentation labels and edges supervise the model to achieve high quantitative performance. Furthermore, predicting them from the coarse image tends to get a better

estimate as opposed to doing the estimation directly from the corrupted image. To further evaluate our approach, we report a quantitative comparison of the CelebAMask-HQ dataset [53][57]. Table 3.3 lists the evaluation results for several mask sizes. Our method compares very well against three existing methods. Similar to the MSCOCO dataset [52], the lower performance for CA [19] is expected since it does not use any additional constraints such as boundary edges or object segmentation labels. The proposed method outperforms GC [3] in all the metrics and has very close scores to CSA [21] that uses a huge model with complex mechanisms.

Table 3.2: Quantitative comparison of the SC model on MSCOCO.

	Mask size	CA	EC	GC	SF	CSA	SC
MAE^-	10-20%	0.014	0.013	0.012	0.013	0.012	**0.011**
	20-30%	0.030	0022	0.020	0.020	0.019	**0.018**
	30-40%	0.042	0.032	0.031	0.030	0.028	**0.025**
	40-50%	0.068	0.052	0.050	0.048	0.045	**0.041**
$SSIM^+$	10-20%	0.953	0.956	0.961	0.960	0.963	**0.971**
	20-30%	0.880	0.925	0.930	0.933	0.937	**0.948**
	30-40%	0.819	0.888	0.893	0.895	0.901	**0.920**
	40-50%	0.679	0.774	0.784	0.788	0.796	**0.825**
$PSNR^+$	10-20%	26.63	28.72	28.82	28.90	29.12	**30.38**
	20-30%	23.48	26.08	26.17	26.24	26.43	**27.61**
	30-40%	21.48	24.23	24.29	24.37	24.68	**25.76**
	40-50%	18.62	21.12	21.15	21.18	21.71	**22.67**
$NIQE^-$	10-20%	5.83	5.36	4.90	4.87	4.56	**4.04**
	20-30%	6.05	5.54	5.21	5.17	4.89	**4.31**
	30-40%	6.44	5.76	5.31	5.33	5.06	**4.84**
	40-50%	6.89	6.15	5.42	5.49	5.34	**5.11**
$TCLT^-$	10-20%	24.52	22.94	22.43	21.87	21.36	**20.81**
	20-30%	25.86	24.09	23.69	22.93	22.47	**21.71**
	30-40%	27.03	25.37	26.96	24.04	23.66	**22.81**
	40-50%	28.61	26.82	26.33	25.87	25.19	**24.31**
$BRISQUE^-$	10-20%	25.05	24.03	23.29	23.74	23.16	**22.03**
	20-30%	25.35	24.66	23.78	23.87	23.07	**22.81**
	30-40%	25.87	25.05	24.34	24.25	23.62	**23.36**
	40-50%	26.50	25.66	25.16	25.12	24.36	**24.15**

Table 3.3: Quantitative comparison of the SC model on CelebHQ.

	Mask size	CA	GC	CSA	SC
MAE^{-}	10-20%	0.014	0.009	**0.005**	0.007
	20-30%	0.024	0014	**0.011**	0.012
	30-40%	0.033	0.021	0.018	**0.017**
	40-50%	0.052	0.031	0.030	**0.029**
$SSIM^{+}$	10-20%	0.953	0.982	0.983	**0.986**
	20-30%	0.918	0.968	0.973	**0.975**
	30-40%	0.881	0.950	0.958	**0.959**
	40-50%	0.796	0.899	**0.903**	0.900
$PSNR^{+}$	10-20%	28.55	32.53	33.33	**33.81**
	20-30%	25.54	29.73	30.56	**30.96**
	30-40%	23.58	27.80	28.67	**28.95**
	40-50%	21.03	25.05	25.41	**25.85**
$NIQE^{-}$	10-20%	4.93	4.76	4.47	**4.23**
	20-30%	5.19	4.98	4.65	**4.47**
	30-40%	5.37	5.02	4.81	**4.74**
	40-50%	5.61	5.29	4.98	**4.94**
$TCLT^{-}$	10-20%	28.46	27.34	**26.62**	26.71
	20-30%	30.15	28.63	27.92	**27.39**
	30-40%	32..22	30.42	29.36	**28.10**
	40-50%	34.68	32.36	31.77	**29.62**
$BRISQUE^{-}$	10-20%	23.59	22.96	**21.07**	21.13
	20-30%	23.79	23.17	**21.66**	21.70
	30-40%	23.90	23.75	22.21	**22.18**
	40-50%	24.62	24.03	22.46	**22.38**

3.4.3. Computation time analysis

In addition to the qualitative and the quantitative index (MAE, SSIM, PSNR, NIQE, TCLT and BRISQUE) analysis, we compare the proposed approach against the state-of-the-art methods in terms of the number of floating-point operations in GFLOPS, the model parameters in millions and the computation time both on CPU and GPU in milliseconds. Note that the CPU is an Intel(R) Core (TM) i7-2600K CPU @ 3.40 GHz, and the GPU is NVidia Titan X. We can see from Table 3.4 that CA [19] has the least number of network parameters. However, it shows significantly inferior performance compared to other state-of-the-art methods (see Table 3.2 and Table 3.3). Furthermore, our approach presents the least number of floating points and CPU/GPU time. We can explain this by that our networks do not employ complex

mechanisms such as CAM. Although EC [17] inpainting performances are better than CA, it reports higher network parameters, GFLOPS and running time. Besides, it presents inferior inpainting performance compared to the other methods. GC [3] and our model are very competitive in terms of performance and the number of network parameters. However, our proposed approach reports the shortest running time and GFLOPS due to the non-use of expensive gated operations. SF [16] has comparable results, but the proposed method presents a better computational speed than SF, which has the largest GFLOPS. Despite its promising results, CSA [21] has the highest network parameters and running time among all the compared models.

Table 3.4: Computational time analysis of the SC model.

Model	GFLOPS	PARAMS (M)	CPU (ms)	GPU (ms)
CA	22.4	**2.9M**	383	18
EC	122.4	21.5M	704	32
GC	39.6	4.1M	490	27
SF	262.4	92.5M	810	36
CSA	55.16	132M	972	180
SC	**16.4**	4M	**374**	**16**

3.4.4. Ablation studies

We analyze the impact of each component of our model individually. We conduct several experiments on the CelebAMask-HQ dataset [53][57] and show quantitative and qualitative results for each case.

3.4.4.1.Intermediate outputs of the proposed method

We conduct a set of experiments to confirm that the proposed end-to-end deep generative model can robustly generate the coarse image, the segmentation labels, the edges and the final (fine) inpainted image for the given corrupted image. We show examples of all the mentioned predictions in Figure 3.5. We can see that the proposed

approach achieves good performance in the CelebAMask-HQ [53][57] and MSCOCO [52] datasets. Furthermore, the estimated segmentation labels and the edges improve the coarse image and help to synthesis semantically correct objects with realistic texture details.

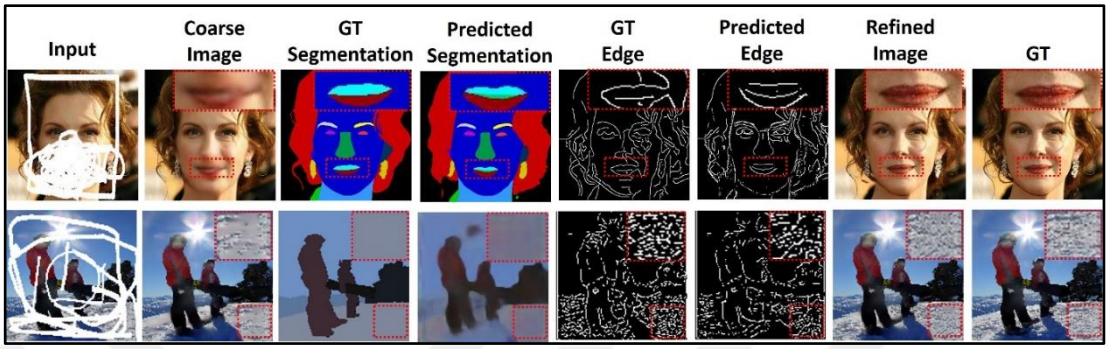

Figure 3.5: Intermediate outputs of the SC approach on the CelebHQ and MSCOCO datasets.

3.4.4.2. The effect of segmentation labels

We perform two different experiments to investigate the impact of segmentation labels. We compare our full model with another version that employs only the predicted edges to guide the model. Table 3.5 shows that the first configuration yields higher performance proving that the two constraints explicitly lead the network to learn better structures and textures.

3.4.4.3. The effect of the edges

We analyze how our predicted edges contribute to the final image. We believe that the edges help to represent the image structure. To verify this, we also compare our full model with another version that employs only the segmentation labels to guide the model. The quantitative comparison in Table 3.5 indicates that the estimated edges considerably improve the performance of our network. Furthermore, this experiment proves that the edges are more crucial than the segmentation labels in our inpainting framework.

Table 3.5: Ablation studies of the SC model on the CelebHQ dataset.

Methods	MAE^{-}	$SSIM^{+}$	$PSNR^{+}$
Full	**0.0165**	**0.9554**	**29.82**
Full w/o segmentation	0.0179	0.9511	29.46
Full w/o edge	0.0173	0.9513	29.37
Full w/ CL	0.0169	0.9543	29.64
Full from corrupted	0.0176	0.8835	27.58

3.4.4.4. Our method vs. prediction from corrupted images

Using segmentation labels and edges to guide inpainting models has been used in [17][47][18]. However, no approach combines them to supervise the image inpainting task. Also, all the mentioned methods estimate edges and segmentation labels from the corrupted image. To prove the limitation of this approach, we compare it against our proposed method that estimates segmentation labels from the coarse image. Besides, it predicts the edges from the segmentation labels and the coarse image. As seen from Table 3.5, estimating edges and segmentation labels from a corrupted image decrease the performance that proves the effectiveness of the proposed methods. Also, Figure 3.6 illustrates that the proposed approach can identify segmentation labels and edges more effectively than estimating them directly from the corrupted image. Note that columns from left to right represent: the input image, the prediction from the corrupted image, our method, and the ground truths. Also, rows from top to down represent the segmentation labels, the edges, and the images.

3.4.4.5. The accuracy of segmentation labels and edges predictions

Since segmentation labels and edge estimations play a significant role in the proposed method, we report the quantitative results of this part, including accuracy, precision, recall, F1 score and Intersection over Union (IoU). We can see from Table 3.6 and Table **3.7** that the two networks achieve plausible results. It can identify the edges of the corrupted regions and predict correct segmentation labels in most cases. Furthermore, as we did previously, we compare the proposed method with the

conventional version that estimates the segmentation labels and the edges from the corrupted images. We can see that this version reports inferior performances in all the metrics that prove the effectiveness of our prediction approach.

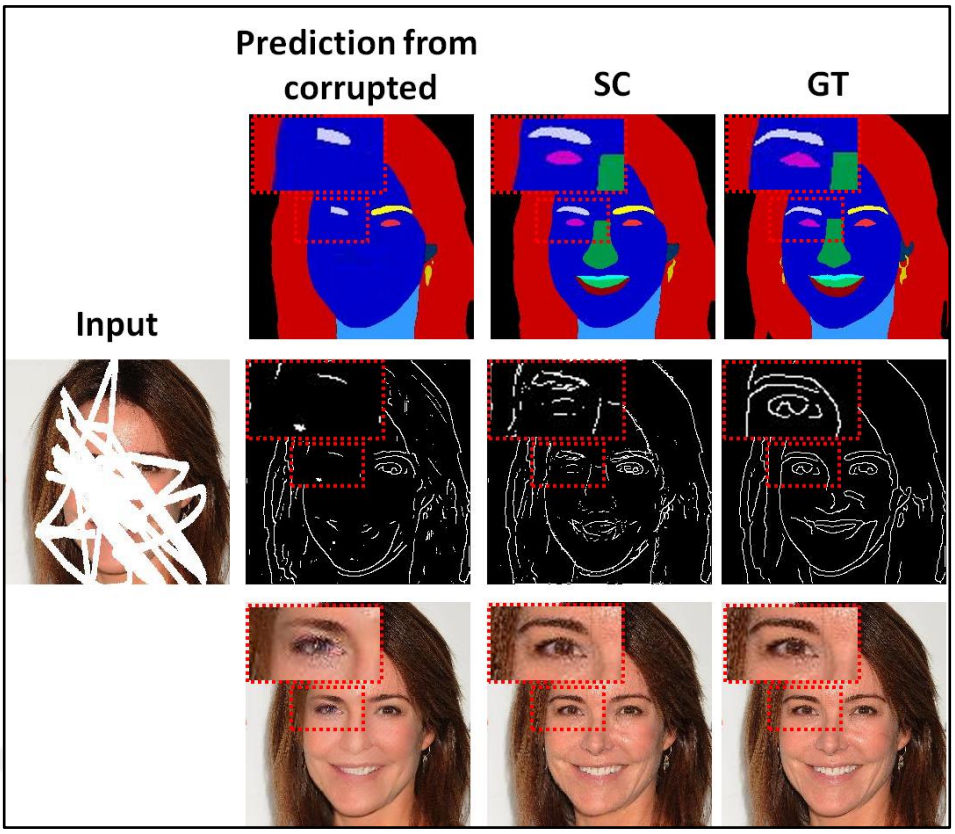

Figure 3.6: Predictions from corrupted images vs. predictions from the SC model.

Table 3.6: Edge prediction metrics of the SC model.

	Metric	From corrupted	SC
CelebHQ	Accuracy	91.46	**97.02**
	Precision	67.06	**88.49**
	Recall	79.38	**81.93**
	F1	72.70	**85.08**
MSCOCO	Accuracy	86.53	**94.89**
	Precision	62.96	**84.62**
	Recall	77.89	**79.04**
	F1	69.63	**81.73**

Table 3.7: Segmentation labels prediction metrics of the SC model.

	Metric	From corrupted	SC
CelebHQ	Accuracy	84.99	**85.09**
	Precision	71.58	**72.10**
	Recall	86.39	**87.55**
	F1	78.29	**79.07**
	IoU	62.72	**66.48**
MSCOCO	Accuracy	78.03	**80.21**
	Precision	68.12	**69.87**
	Recall	83.04	**84.39**
	F1	74.84	**76.44**
	IoU	61.72	**63.16**

3.4.4.6.Failure cases

Figure 3.7 shows that the performance of our method is highly dependent on the segmentation labels and edge predictors. We show our successful/failure predictions in the upper/lower figures, respectively. Specifically, each row represents the masked image, the ground truth image and the estimated segmentation labels, edges, and the final image. Note that robust segmentation labels and edge predictions help the model to generate high-quality images. In contrast, wrong segmentation label predictions can degrade the performance of our method. In other words, sparse or discontinued predicted edges generate low-quality images with fewer texture details. These failure cases especially appear in non-repetitive objects, such as accessories and clothing. This issue is a potential limitation of our study. However, these unsuccessful cases rarely occur since the end-to-end optimization of the refinement network involves weights updating edges, segmentation labels and coarse estimators.

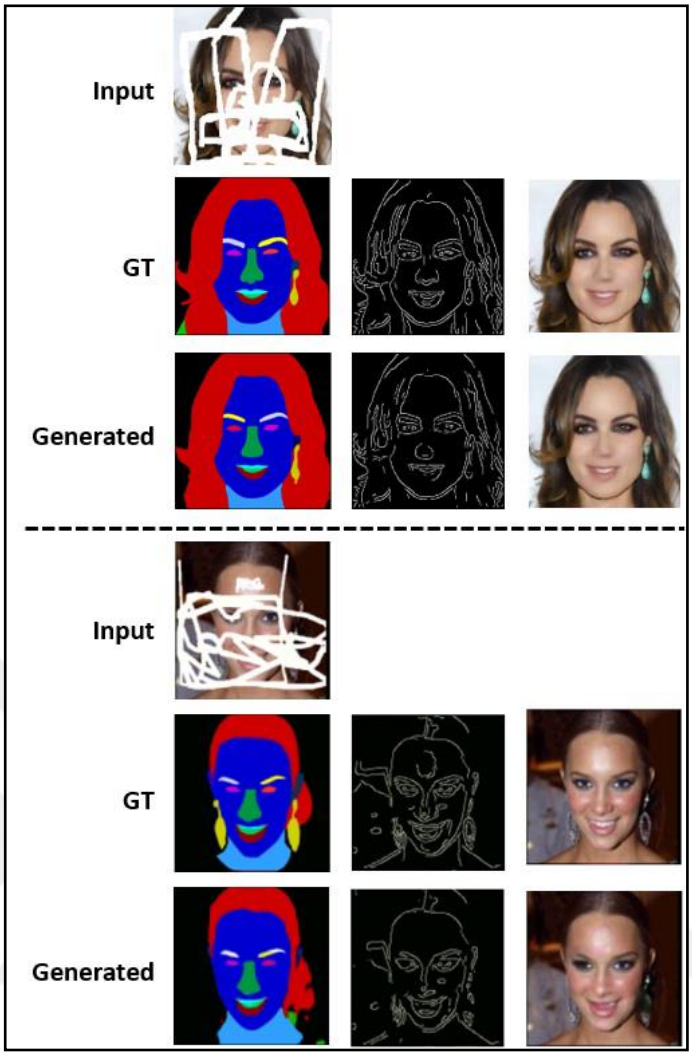

Figure 3.7: Successful/failure predictions of the SC model.

3.4.4.7.The effect of CL

To show the efficiency of the proposed curriculum resizing approach, we replace the traditional training procedure that optimizes the parameters of the full model on a single resolution (256×256) with the CL one. While the traditional one takes three days to converge on a single NVIDIA TITAN X GPU, the CL training takes only one day. Also, Table 3.5 shows that the CL version has comparable performance to the common one, which is very suitable for low computational power platforms.

3.4.5. Interactive editing

The proposed method allows interactive drawing of the input masks for object removal applications. It robustly restores the missing regions with a high relevance

with the background. We show some examples of interactive editing in Figure 3.8. We remove objects with uniform backgrounds (snow and sky) and complex backgrounds (human skin and hair). In both cases, the generated images illustrate fine-grained textures and semantic consistency.

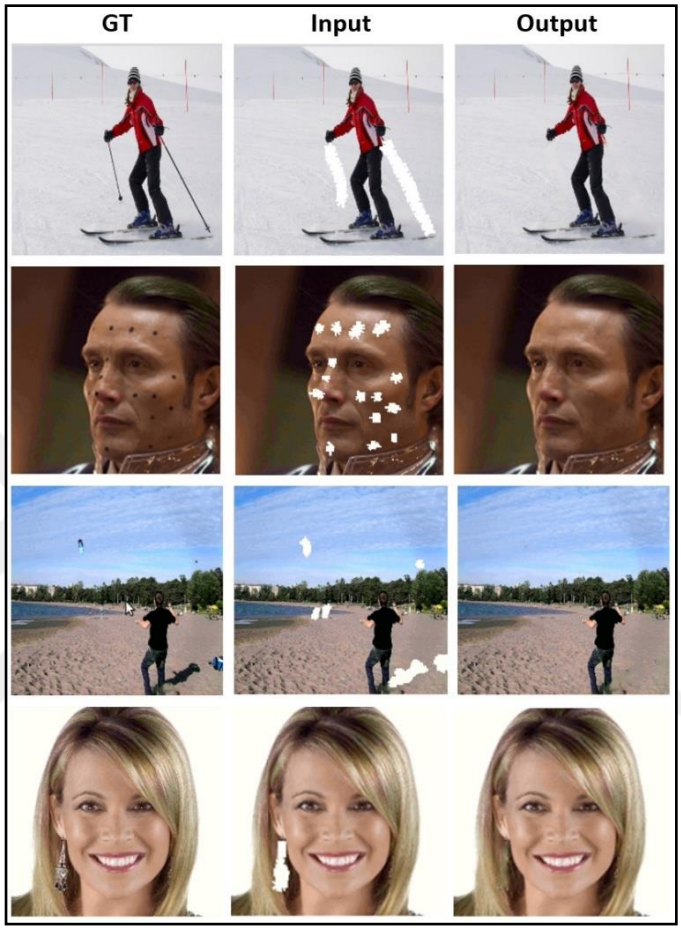

Figure 3.8: Object removal and image editing using the SC model.

3.5. Summary

This chapter proposes a GAN-based image inpainting approach that enforces input constraints to guide the model. Specifically, we estimate the segmentation labels and edges to recover consistent object boundaries and coherent structure in the corrupted regions. We demonstrate that combining these two estimated entities yields visually appealing images. Our model performs favorably against several state-of-the-art methods on public datasets. It successfully recovers damaged pixels and removes

objects from scenes. The proposed curriculum resizing technique speeds up the training time without decreasing the performance.

4. EFFICIENT TEXTURE-AWARE MULTI-GAN FOR IMAGE INPAINTING

4.1. Introduction

Most GAN-based inpainting techniques suffer from structure preservation and unrealistic texture generation problems, which leads to blurry and geometrically distorted results. To address the issues, most of the current GAN-based inpainting methods employ coarse-to-fine architectures [3][17][16][18]. Specifically, the coarse stage predicts the initial image from the corrupted [19][20] or reconstructs the image structure represented in the edge [17][16], the contour [46] and the segmentation labels [47][18]. The refinement stage generally uses the predicted coarse image or the reconstructed information to generate a final plausible image. However, the performance of the mentioned multi-stage approaches is strongly related to the contour/edge/segmentation labels prediction stages. Furthermore, they require expensive computational resources since they optimize the parameters of two or more networks. Other studies employ the contextual attention mechanism (CAM) to borrow information from the surrounding parts [19][20]. However, CAM still fails to ensure feature continuities [21] and requires expensive computational resources. In addition to coarse-to-fine architecture and CAM, there exists another bottleneck that drastically increases the model capacity. Namely, training on high-resolution images, which involves big models with many parameters. Consequently, the training becomes slower and enforces smaller batch sizes due to computational and memory resource constraints, which decreases the performances [22].

Motivated by these observations, we introduce a new deep generative-based multi-resolution image inpainting framework to speed up the running time and restore fine-grained textures. Our approach is composed of four successive efficient generators filling in four different resolutions. Specifically, the training starts with lower-resolution images and progressively doubles their size, such that their corresponding generators can exploit the previously inpainted images (Figure 4.1). This technique improves the model stability since training GANs on low-resolution images proves easier and converges faster [22]. Another main problem with direct high-resolution image synthesis is that the discriminator will focus on texture details. Hence, it can easily reject synthesized images in the early training stages. Our

approach drops the refinement module after the target resolution since it significantly increases the network size. We remedy the lack of this refinement stage by our proposed progressive training approach and a texture-based loss function. The latter adopts Local-binary-patterns (LBP) [63] to the image inpainting task. LBP is a non-parametric texture descriptor that is widely used in computer vision tasks [64]. We minimize the distance between the ground truth LBP and the predicted one to enforce fine-grained textures in the corrupted regions. Hence, our approach does not require high computational resources since it neither performs complex operations (CAM) nor uses the refinement networks. We conduct our qualitative and quantitative experiments on conventional inpainting datasets Places2 and CelebHQ [53]. The results show that our efficient model can generate visually appealing images and outperforms current state-of-the-art methods. Our main contributions are as follows:

- We present a new GAN-based image inpainting architecture that employs efficient progressive GANs to improve the performance and speed up the inference time.
- We adopt an LBP-based loss function to constrain the inpainting task and ensure realistic texture details.
- The experiments on Places2 [22] and CelebHQ [53] datasets exhibit competitive qualitative and quantitative results against current state-of-the-art methods. We also show the scalability of the proposed approach to other applications, such as blind image inpainting and image outpainting.

4.2. Approach

4.2.1. Multi-resolution based inpainting

Training GANs on high-resolution images is a challenging optimization problem that involves millions of parameters. [22] produces low-resolution images from a latent Gaussian vector in the first stage. During training, it progressively adds layers to the generator and the discriminator to increase the image resolution. However, this framework is unsuitable for image-to-image translation applications since they require a high-resolution image as input. We introduce a GAN-based architecture for image inpainting that includes four progressive generators and discriminators. We train an encoder-decoder generator on a low-resolution image for many epochs to robustly

produce samples with a very close distribution to the original one. As the training progresses, we use the pretrained generators as the starting point for the successive higher-resolution generator. The latter exploits the filled-in regions of the previous lower-resolution images to learn the global image consistency and inpaint correct structures. In contrast, training GANs on high-resolution images is hard to stabilize, which affects the model performance. We can explain this by that, during training, the discriminator keeps rejecting most of the generated images since the ground truth image contains fine-grained texture details, which are very difficult for the generator to produce [22][65].

To the best of our knowledge, the proposed architecture is one of the first studies that apply progressive generators and discriminators for image inpainting. [20] is a coarse-to-fine architecture that predicts a high-resolution coarse image and enhances it using multi-scale discriminators in the refinement stage. The discriminator of each scale criticizes the output of that resolution size and gives high gradient feedback to early convolution layers. However, it does not directly exploit the refined images in lower scales, which may still be a bottleneck for the high-resolution discriminator. The latter reject the generated samples easily due to high-frequency information. In contrast, we build our high-resolution prediction on already filled-in predictions in lower resolutions (Figure 4.1). In this way, the discriminator criticizes reasonable synthesized images that are close to ground truth samples. The proposed approach neither uses coarse-to-fine architecture nor an attention mechanism that significantly increases the model complexity in [20]. Another approach in [41] uses a UNet [42] architecture embedded with fusion blocks in a multi-scale manner. However, they drop the adversarial learning and use perceptual and style losses [66] to enforce texture details. Using only the former losses without adversarial learning can result in checkerboard artifacts since it is hard to find the best loss weights [15]. In our approach, we use the adversarial, reconstruction and the proposed LBP loss functions to enhance the image texture (see Section 4.2.3).

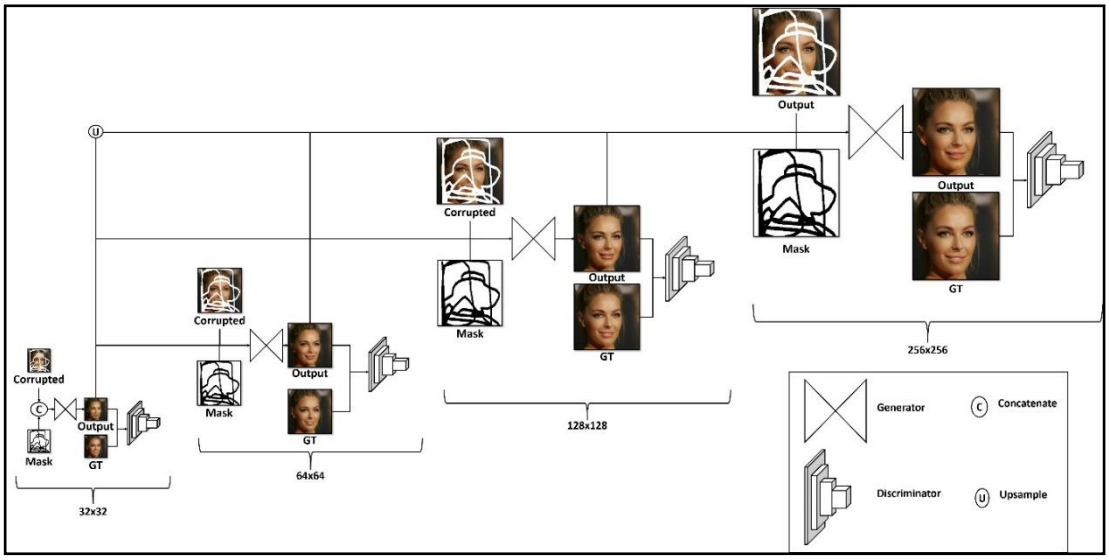

Figure 4.1: Architecture of the ETMG model.

4.2.2. Architecture

Using different resolutions with multiple receptive fields helps the network to learn the global structure of the image. As illustrated in Figure 4.1, the training starts with 32×32 resolution images. We channel-wise concatenate the corrupted image and the mask to feed them to their specific-resolution generator. We give the output of the latter and the ground truth images to the PatchGAN [26] discriminator that shifts the generator distribution to the real one. We use the generator network defined in Figure 4.2 (a). We train the generator and the discriminator until convergence. During training, we visualize different quality metrics and loss values. We stop the training by monitoring the image quality metrics and when the visual quality of the synthesized images is good enough as input for the succeeding resolution. We use the pre-trained generator of the 32×32 resolution to train the next resolution network (64×64). The network in Figure 4.2 (b) contains three sub-networks where each one takes a specific input. We feed the concatenation of the corrupted image and the mask to the first sub-network. We feed the concatenation of the corrupted image and the mask to the first sub-network. We feed the estimated image of resolution 32×32 to the second sub-network. The last sub-network takes the concatenation of the feature maps produces by the previous sub-networks to produce the final image (64×64). Similarly, we follow the same approach for the last two resolutions (128×128 and 256×256),

where each generator exploits the previously inpainted images as described in Figure 4.2 (c) and Figure 4.2 (d).

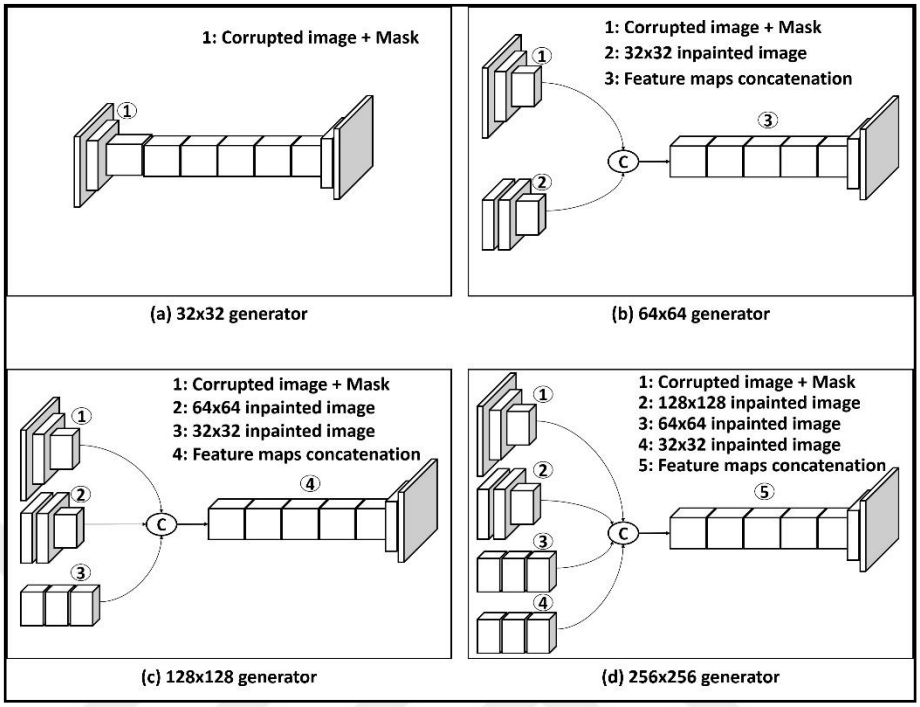

Figure 4.2: Input and output of the progressive generators.

4.2.3. LBP operator for texture preservation

LBP [63] is a nonparametric image operator that transforms an image into an array representing the local structure of the image by comparing each pixel with its adjacent pixels. LBP is a robust descriptor that can summarize the most important texture information in an image. Also, it shows computational simplicity and good performance in many computer vision and image processing applications [64]. An example of a 3×3 LBP operator is shown in Figure 4.3. LBP iterates over each pixel in a grayscale image to check the values of the surrounding 3×3 patch, whether they are smaller than the center pixel or not. The resulting binary number is converted to a decimal number and placed in the corresponding position in the LBP image. Early deep learning-based inpainting methods assume that the image texture and semantic can be learned automatically by CNNs without further supervision. Recent GAN-based approaches demonstrate that this task is challenging and require additional information. [67] employs discriminative modules and class supervision to enforce fine-grained features. Other GAN-based inpainting approaches add [3] or predict

[17][16] edges to ensure realistic textures. However, choosing the correct threshold for the Canny edge detector [68] that can preserve the image texture for both highly structured and simple images is difficult in practice. Furthermore, the edges cannot provide sufficient texture details in many cases, such as the face skin and uniform backgrounds. Motivated by these observations, we investigate hand-crafted features. Specifically, we adopt the famous texture operator LBP [63] as a new loss function for image inpainting to ensure better texture learning as used in [69] for infrared and visible image fusion and [70] for face recognition. In particular, we minimize the loss between the LBP of the ground truth and the predicted images using the LBP layer defined in the algorithm of Figure 4.4. We select the LBP operator since it is robust to illumination variations and invariance to gray-scale changes. Furthermore, it does not add parameters to the network, and it is computationally inexpensive. However, LBP is a non-differentiable iterative function that cannot be optimized using backpropagation. To address that, we transform the problem into matrix multiplication operations using a fixed weight convolution layer. Thus, it does not add learnable parameters to our full model. We base our implementation on [71]. Note that we only use the LBP loss in the last resolution (256×256), which speeds up the inference time.

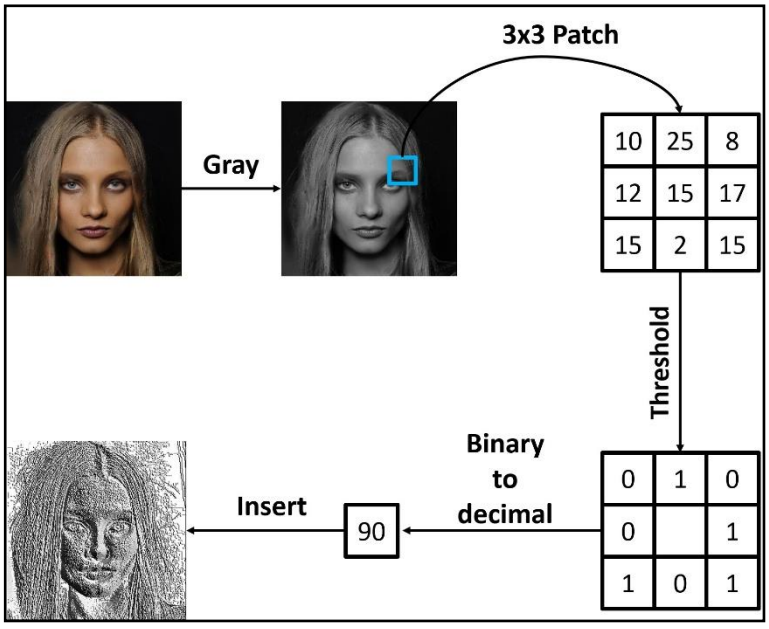

Figure 4.3: Example of 3×3 LBP operator applied on a grayscale image.

Algorithm 1: The LBP layer pseudo-code

Input: Gray-scale image
Output: LBP image
Function `LBPLayer`:
- Conv = 2D convolution layer.
- Initialize the parameters to: $in_{channels} = 1$, $out_{channels} = 8$, $kernel = 3$, $stride = 1$, $dilation = 1$, $bias = False$.
- Initialize the kernels to zeros.
- Initialize the center of the kernels to -1.
- Initialize the remaining values to 1 in position: 0, 1, 2, 3, 4, 5, 6 and 7 for each kernel, respectively.
- codes = list of 8 values initialized to 1, 2, 4, 8, 16, 32, 64, 128.
- ReLU = Rectified Linear Unit activation function.
- $result = Conv(input)$.
- $result = ReLU(result)$.
- $result = result * codes$.
- $result = result.sum(dim = 1)$.
- $result = result/255$.

return result

Figure 4.4: The algorithm of the differentiable LBP layer.

4.2.4. Loss functions

Let $I_{n\times n}$ and $M_{n\times n}$be the ground truth image and the mask where n is the size of the square image. Also, let $G_{n\times n}()$ be a generator network that generates an image $O_{n\times n}$ Also, let $LBP(.)$ be a differentiable LBP layer. that takes a grayscale image and outputs the LBP image. The output image for various resolutions can be obtained using (4.1), (4.2), (4.3) and (4.4):

$$O_{32\times32} = G_{32\times32}(I_{32\times32} \times M_{32\times32}, M_{32\times32}) \tag{4.1}$$

$$O_{64\times64} = G_{64\times64}(I_{64\times64} \times M_{64\times64}, M_{64\times64}, O_{32\times32}) \tag{4.2}$$

$$O_{128\times128} = G_{128\times128}(I_{128\times128} \times M_{128\times128}, M_{128\times128}, O_{32\times32}, O_{64\times64}) \tag{4.3}$$

$$O_{256\times256} = G_{256\times256}(I_{256\times256} \times M_{256\times256}, M_{256\times256}, O_{32\times32}, O_{64\times64}, O_{128\times128}) \tag{4.4}$$

L1 loss: we measure the error between the ground truth image and the predicted image for each resolution as defined in (4.5).

$$L_{rec} = \|O_{n\times n} - I_{n\times n}\|_1 \tag{4.5}$$

Adversarial loss: we optimize the LSGAN [28] adversarial loss for each resolution as defined in (4.6).

$$\begin{aligned} L_{dis} &= \mathrm{E}\,[(\mathrm{D}(I_{n\times n}) - 1)^2] + \mathrm{E}[\mathrm{D}(O_{n\times n})^2] \\ L_{adv} &= \mathrm{E}\,[(\mathrm{O}(I_{n\times n}) - 1)^2] \end{aligned} \tag{4.6}$$

Texture loss: we use the LBP differentiable layer to calculate the loss between the ground truth texture and the generated 256×256 image texture as defined in (4.7).

$$L_{texture} = \left\|\mathrm{LBP}(\mathrm{Gray}(O_{fine})) - \mathrm{LBP}(\mathrm{Gray}(I_g)\right\|_1 \tag{4.7}$$

Overall loss: we use a weighted sum of the reconstruction, the adversarial and the texture loss. We give a weight γ_{adv}=0.1, γ_{rec}==1 and $\gamma_{texture}$==10 for the adversarial loss, the reconstruction loss and the texture loss, respectively. The overall loss is defined in (4.8):

$$L_{overall} = \gamma_{adv} \times L_{adv} + \gamma_{rec} \times L_{rec} + \gamma_{texture} \times L_{texture} \tag{4.8}$$

4.3. Experimental evaluation

4.3.1. Datasets and masks

We conduct our experiments using two conventional image inpainting datasets. The first one is Places2 [72] that has more than 1.8M images and 400 scene categories, such as bedrooms, streets, etc. Places2 dataset became a popular image inpainting dataset since it has a vast natural scene variation. We use the original train and test split for the Places2 dataset. To further enrich our experiments, we evaluate our method on CelebHQ [53], which has 30K cropped face images with a large pose and background variations selected from the CelebA dataset [73]. We use the original training and test split. Since users of image inpainting applications usually want to edit

or remove arbitrary shapes in the scenes, we use irregular mask sizes [58] during training.

4.3.2. Implementation details

In this part, we describe our training procedure and the hyper-parameter settings. We use Pytorch [74] to implement the proposed method using CUDA v10.1 and cuDNN v7.6.4. We use Adam optimizer [59] with hyper-parameters $\alpha = 0.5$ and $\beta = 0.99$, respectively. We set the batch size to 32, and we fix the learning rates to 10^{-4} for the generators and the discriminators. We use spectral normalization [75] in all the convolution layers of the discriminator. The details of the architectures are described in Table 8.1, Table 8.2, Table 8.3, Table 8.4 and Table 8.5 of the Appendix. We freeze the weights of the previous networks when training the generator and the discriminator of the current resolution.

4.4. Results and discussion

We compare qualitatively and quantitively our full model against current state-of-the-art methods, including Contextual Attention (CA) [19], Edge Connect (EC) [17], Deep Fusion Network (DFNet) [41], Gated Convolution (GC) [3] and Structure Flow (SF) [16]. We select these approaches for two main reasons: the availability of the pretrained models that ensure a fair comparison and save both time and computational resources. The second reason is that they achieve very competitive results using different approaches. We use the original train and test splits for Places2 [72] and CelebHQ [53] datasets.

4.4.1. Qualitative results

We qualitatively compare our approach with the selected state-of-the-art methods on two datasets. We zoom in on different parts of the image to show the difference between the generated images. As seen from Figure 4.5, CA [19] produces significant artifacts leading to misrepresented structures. EC [17] produces better results since it estimates edges to recover the global structure of the image, but obvious visual artifacts still appear in the masked regions. While DFNet [41] generates plausible and smooth images with global image consistency using their introduced

fusion blocks, it still exhibits observable color discrepancies. GC [3] produces realistic images due to the gated convolution layers and the refinement network, but it can miss some texture details. SF [16] generates plausible images with fine-grained textures since it employs two effective stages to preserve both structures and textures, respectively. However, it suffers from remarkable inconsistencies near the boundaries. Our method presents competitive results and shows very realistic textures in all the missing regions.

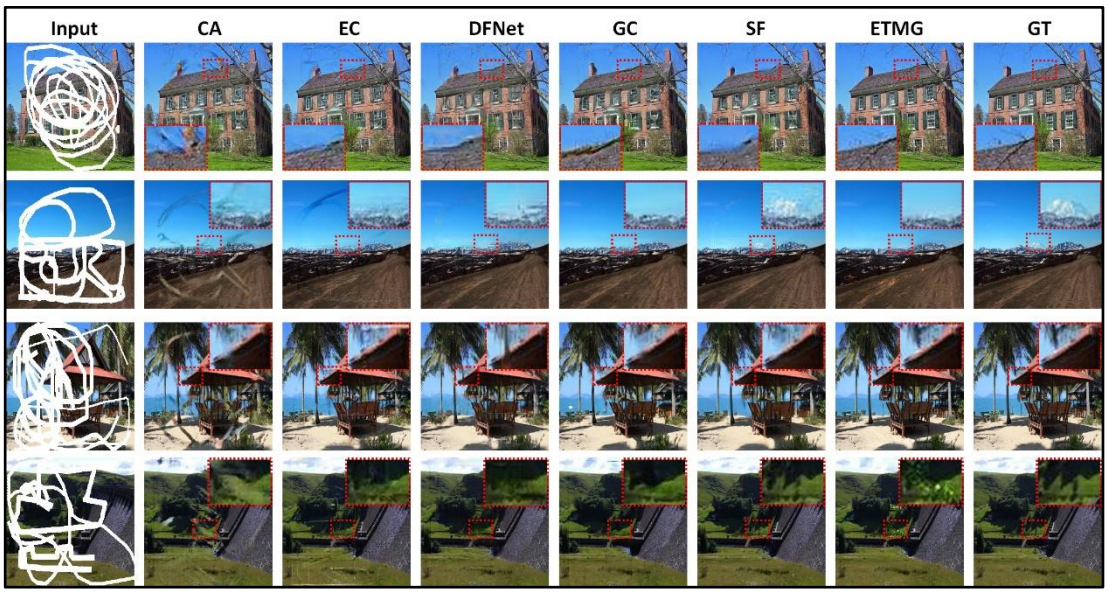

Figure 4.5: Qualitative comparison of the ETMG model on Places2.

To further demonstrate the effectiveness of the proposed method, we report qualitative results on the CelebHQ dataset [53]. We can see from Figure 4.6 that the images produced by CA show visually poor performance. GC generates realistic images but still shows discordance between the background and the parts of the corrupted region. SF [16] synthesis smooth faces with realistic textures. However, it sometimes exhibits color and row discontinuities in the predicted pixels. Our method presents the most natural faces without using large models or complex mechanisms such as CAM. The results can be explained by that our approach looks at different image scales using multiple GANs to ensure visually realistic images with both local and global structure consistencies. Besides, the proposed LBP-based loss function both improves and sharpens the texture of the generated parts. Additional results of our proposed approach in the Appendix section show that our model may synthesize plausible new contents due to adversarial learning. Also, our qualitative results show

that our model enforces close LBP features if the generated contents have resembling structures to the ground truth or only the color is changed. In contrast, the produced LBP features are distinct for modified image structures. Note that in both cases, the LBP-based loss function ensures fine-grained textures.

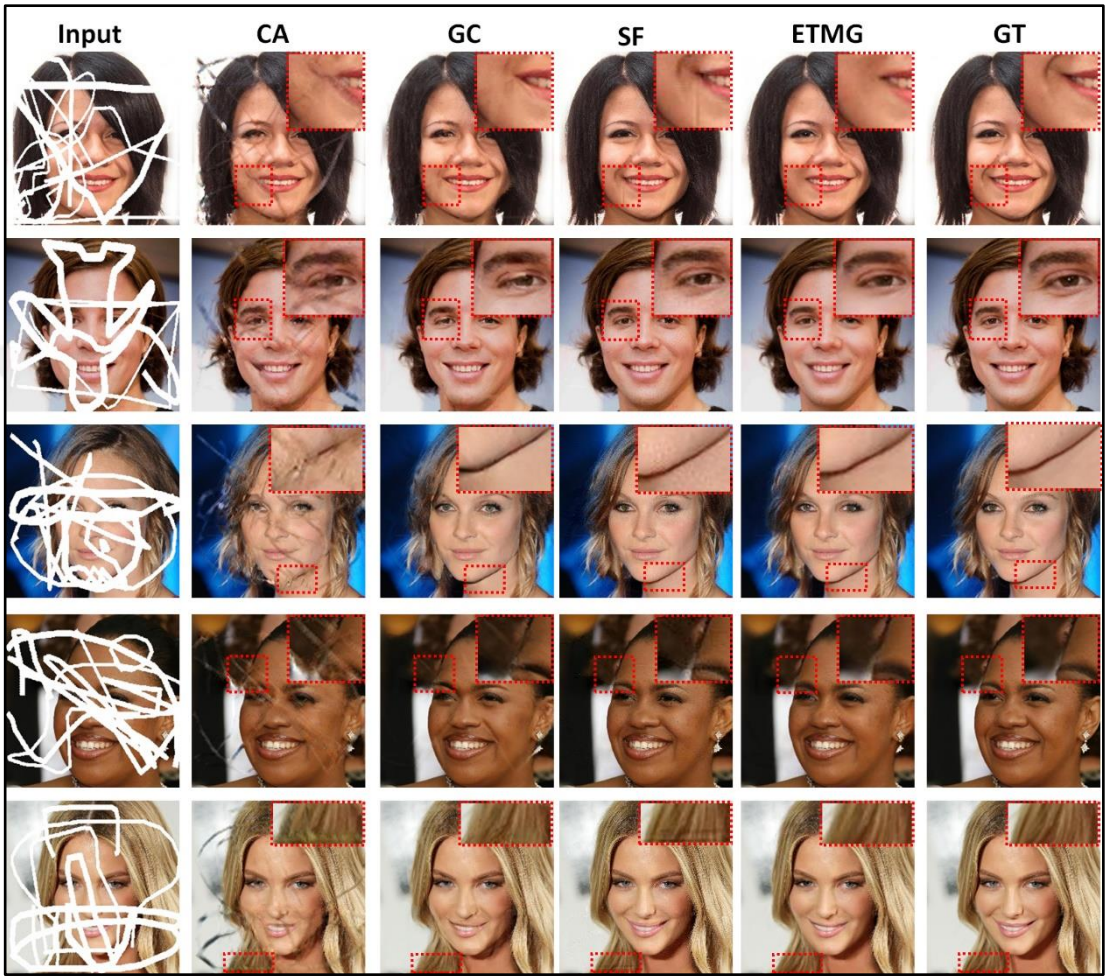

Figure 4.6: Qualitative comparison of the ETMG model on CelebHQ.

4.4.2. Quantitative results

To quantify the performance of the proposed approach, we use three well-known assessment metrics, including MAE, PSNR and SSIM following works of [17][16]. To achieve a fair comparison, we use the same masks and test splits of the two datasets. Table 4.1 lists the evaluation results on the Places2 dataset [72]. We can see that CA [19] shows the worst performances in the three metrics on different mask sizes. EC [17] exhibits the best results since it predicts the edges to supervise the image structure generation. The scores of DFNet [41] and GC [3] are better and very close to each

other. SF [16] shows higher performance in SSIM and PSNR scores in large mask sizes. Our approach achieves competitive results compared to the mentioned state-of-the-art methods without using lightweight generators. Table 4.2 reports the quantitative comparison of CelebHQ [53]. Our proposed method outperforms CA, which shows significantly lower performance. Also, it achieves very comparable results to GC and SF that have a bigger model.

Table 4.1: Quantitative comparison of the ETMG on Places2.

	Mask size	CA	EC	DFNet	GC	SF	ETMG
MAE^-	10-20%	0.019	0.013	0.010	0.011	0.012	**0.009**
	20-30%	0.033	0022	0.019	0.018	0.019	**0.016**
	30-40%	0.048	0.031	0.028	0.026	0.026	**0.024**
	40-50%	0.075	0.053	0.045	0.045	0.044	**0.042**
$SSIM^+$	10-20%	0.922	0.947	0.965	0.969	0.966	**0.971**
	20-30%	0.861	0.913	0.936	0.942	0.944	**0.946**
	30-40%	0.795	0.879	0.901	0.909	0.912	**0.916**
	40-50%	0.660	0.762	0.803	0.810	0.812	**0.816**
$PSNR^+$	10-20%	26.31	27.88	29.51	30.10	30.23	**30.62**
	20-30%	22.07	25.51	26.73	27.13	27.32	**27.71**
	30-40%	20.91	23.96	24.87	25.07	25.38	**25.74**
	40-50%	18.27	20.80	22.03	21.78	21.97	**22.55**

Table 4.2: Quantitative comparison of the ETMG model on CelebHQ.

	Mask size	CA	GC	SF	ETMG
MAE^-	10-20%	0.014	0.009	0.011	**0.006**
	20-30%	0.024	0014	0.015	**0.010**
	30-40%	0.033	0.021	0.018	**0.015**
	40-50%	0.052	0.031	0.028	**0.024**
$SSIM^+$	10-20%	0.953	0.982	0.984	**0.988**
	20-30%	0.918	0.968	0.971	**0.979**
	30-40%	0.881	0.950	0.950	**0.967**
	40-50%	0.796	0.899	0.912	**0.924**
$PSNR^+$	10-20%	28.55	32.53	33.26	**34.64**
	20-30%	25.54	29.73	30.42	**31.79**
	30-40%	23.58	27.80	28.74	**29.81**
	40-50%	21.03	25.05	25.63	**26.64**

4.4.3. Model efficiency

Table 4.3 shows the number of floating-point operations in GFLOPS, model parameters in millions and the CPU/GPU runtime in milliseconds. For a fair comparison, we test all the models on the same hardware for 100 iterations to find the mean inference time. We use an Intel(R) Core (TM) i7-2600K CPU @ 3.40GHz and an NVIDIA TITAN XP GPU. We can see from Table 4.1 and Table 4.2 that our proposed method performances are very competitive to SF [16] and GC [3]. But our model has only 3M parameters and 9.5 (GFLOPS), while SF involves many parameters (92.5M) and 262.4 (GFLOPS) due to the use of two large networks for the smooth and refined image prediction. Also, GC has 4.1M parameters and 39.6 (GFLOPS). This result is due to the use of costly CAM layers and gated operations (sigmoid activation functions) in the refinement network of GC, which augment the number of network parameters and GFLOPS. Our full model is computationally efficient than DFNet [41] that has 9.7 (GFLOPS) and 3.3M parameters. EC [17] has 21.5M parameters and a computation cost of 122.5 (GFLOPS) due to two large models for edge detection and the refinement network for the final prediction. CA [19] has the smallest number of parameters (2.9M). However, it has a high computation cost (22.4 GFLOPS) than our model since it involves many attention layers. Besides, it shows the worst performance both in the quantitative and the qualitative comparison. Concerning the inference time, our model yields the best results, highlighting the efficiency of the proposed approach.

Table 4.3: Computational time comparison of the ETMG model.

Model	GFLOPS	PARAMS (M)	CPU (ms)	GPU (ms)
CA	22.4	2.9M	383	18
EC	122.4	21.5M	704	32
GC	39.6	4.1M	490	27
SF	262.4	92.5M	810	36
ETMG	9.5	3M	334	11

4.4.4. Ablation study

To further demonstrate the effectiveness of the proposed method and show the contribution of each part to the entire approach, we conduct a set of additional experiments. We investigate the effect of LBP loss function and the LBP operator shape. Also, we compare the proposed LBP-loss function against perceptual loss [66]. We analyze the performances of the four generators, and we evaluate the quality of the generated textures. Finally, we show the scalability of the proposed approach to other applications, including image outpainting and blind image inpainting.

4.4.4.1.Effect of the LBP loss

To analyze the contribution of our proposed LBP loss function to the entire approach, we implement two settings of the model, and we show qualitative and quantitative results for each version on the CelebHQ dataset [53]. The first employs only the proposed architecture, while the second adds the LBP loss function to constrain the prediction. We believe that the LBP can robustly describe image textures since the filter comparison operations keep the most meaningful pixels. Table 4.4 indicates that the LBP loss improves the performance and correlates very well with the metrics. Also, we can see from Figure 4.7 that our additional LBP layer restores the image texture and provide realistic images. Note that images of the first version are plausible and have semantic consistency, which proves the effectiveness of our proposed multi-resolution generators.

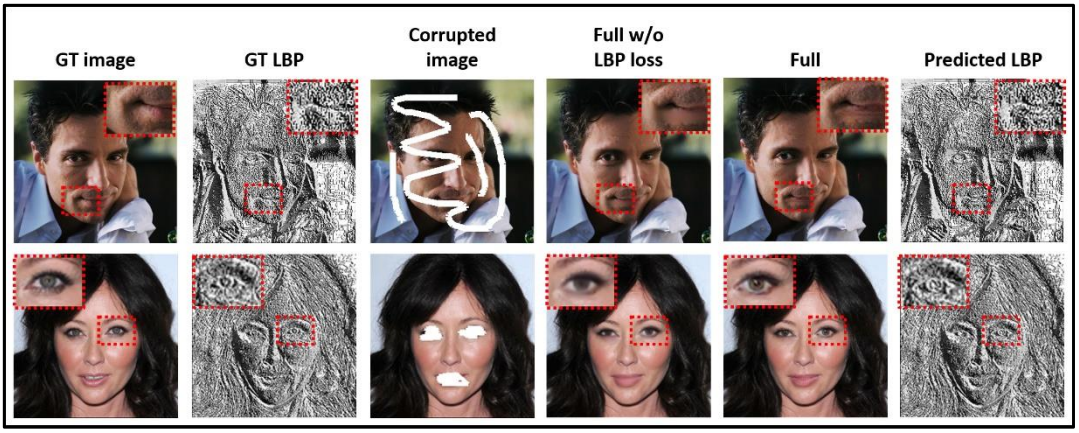

Figure 4.7: Qualitative ablation studies of the LBP loss on CelebHQ.

Table 4.4: Quantitative ablation studies of the LBP loss on CelebHQ.

Methods	MAE^{-}	$SSIM^{+}$	$PSNR^{+}$
Full	**0.014**	**0.964**	**30.72**
Full w/o LBP loss	0.015	0.957	29.89
Full w/ LBP dilation 4	0.014	0.959	30.15
Full w/ LBP dilation 1	0.015	0.964	30.72
Full w/ Perceptual loss	0.014	0.960	30.17

4.4.4.2.Effect of the LBP operator shape

It is well-known that large convolution filters lead to blurriness when applied in the last deconvolution layer. Thus, we fix the filter size to 3×3, and we investigate two different LBP dilation factors to show whether they affect the results or not. We can see from Table 4.4 that using a filter with a dilation factor of 1 achieves better results than a filter with a dilation of 4 since the latter looks for distant pixels from the desired region, which causes blurriness as seen in Figure 4.8.

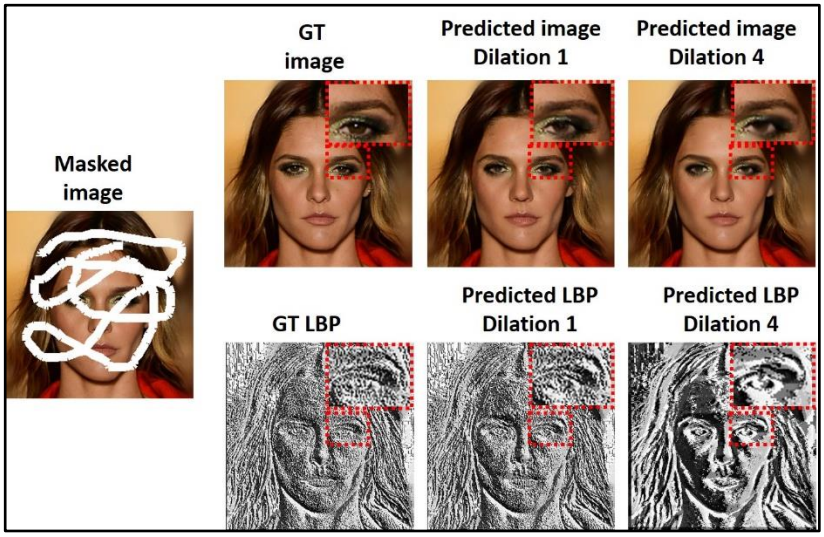

Figure 4.8: Final prediction of the ETMG model using different dilation factors of the LBP operator (1 and 4).

4.4.4.3. Perceptual Loss vs. LBP-based loss

To further analyze the impact of the LBP-based loss function, we compare the proposed approach to a high-level feature loss baseline. Specifically, we drop the LBP loss and use the same multi-GAN architecture with the perceptual loss [66]. During training, generated images and ground truth images are fed to a VGG network to produce the intermediate feature maps in different layers. We observe that the perceptual loss drastically increases the training time since it compares high-dimensional feature maps. On the other hand, our approach compares a single LBP feature map, which speeds up the training. Table 4.6 demonstrates the superiority of our full model in all the quantitative metrics.

4.4.4.4. Analysis of the four generators

Our approach investigates different receptive fields by optimizing the parameters of four progressive generators. In particular, the generator of the higher resolution benefits from the previously inpainted images by the lower ones to learn the global image structure. To show the image structure improvement, we analyze the input of the four generators in the two datasets Places2 [72] and CelebHQ [53]. As seen from Figure 4.9, as the training advances, the quality of the image is improved, and more meaningful structures (edges and boundaries) appear. Although the images of lower resolutions are blurry and do not provide sharp texture details, they recover the global structure of the image, which aids the estimation of damaged pixels of the next resolution. We can see from Figure 4.9 that 32×32 resolutions recover the global structure of the nose and the eyes. However, the images are still blurry and lack texture details. As the resolution increases, the network synthesis visually appealing nose and eyes.

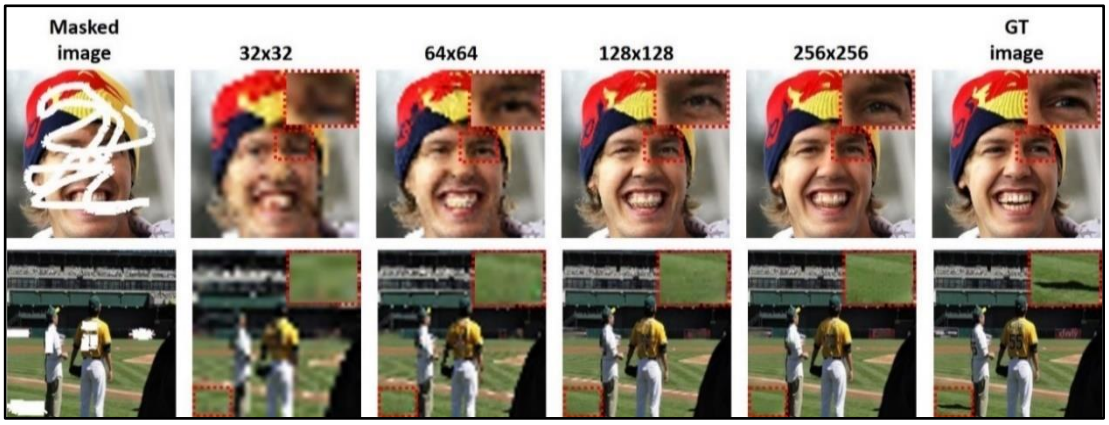

Figure 4.9: Intermediate predictions of the ETMG model on different resolutions.

4.4.4.5.Analysis of the generated texture quality

To further evaluate our approach performances, we measure the accuracy of the edge in the corrupted regions for the Places2 [72] and the CelebHQ [53] datasets since edges robustly express the image structure. We use Canny [68] since it is one of the famous edge detectors to find edges in the generated and the ground truth images. We calculate different metrics on the predicted edge of the corrupted regions to show the percentage of the recovered edges. Table 4.5 shows that our approach can restore most of the texture details since it achieves high precision, recall, accuracy, and F1 measure scores.

Table 4.5: Edge prediction metrics of the ETMG model over CelebHQ and Places2.

Methods	Accuracy (%)	Precision (%)	Recall (%)	F1(%)
CelebHQ	99.9	86.2	85.2	85.7
Places2	93.4	84.7	84.0	84.3

4.4.4.6.Scalability of the proposed method

To confirm the scalability of the proposed method, we conduct four completing experiments. Quantitive and qualitative results are shown in Table 4.6, Figure 4.10 and Figure 4.11. In the first experiment, we train and test it using block-wise masks. Specifically, we put a single hole region with a rectangular shape at different locations. Although this experiment is more challenging than the free-form masks, our method

shows remarkable relevance between the squared hole and the background on CelebHQ [53]. Also, it still exhibits visually plausible results on both uniform and non-uniform backgrounds on the Places2 dataset [72].

Table 4.6: Quantitative evaluation of the ETMG approach on different tasks, including block-wise masks, blind image inpainting and image out-painting.

Task	Metric	Places2	CelebHQ
Block-wise masks	MAE^{-} $SSIM^{+}$ $PSNR^{+}$	0.043 0.813 22.16	0.025 0.907 26.31
Blind inpainting (Free-form mask)	MAE^{-} $SSIM^{+}$ $PSNR^{+}$	0.031 0.889 24.68	0.013 0.952 28.72
Out-painting	MAE^{-} $SSIM^{+}$ $PSNR^{+}$	0.046 0.802 20.87	0.027 0.886 24.66

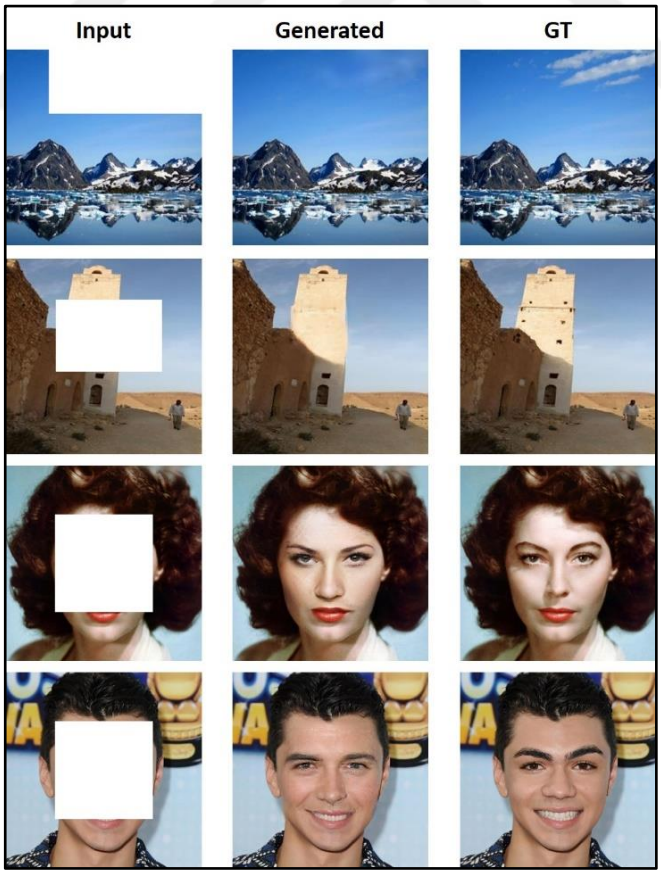

Figure 4.10: Qualitative results of the ETMG model using a rectangular mask.

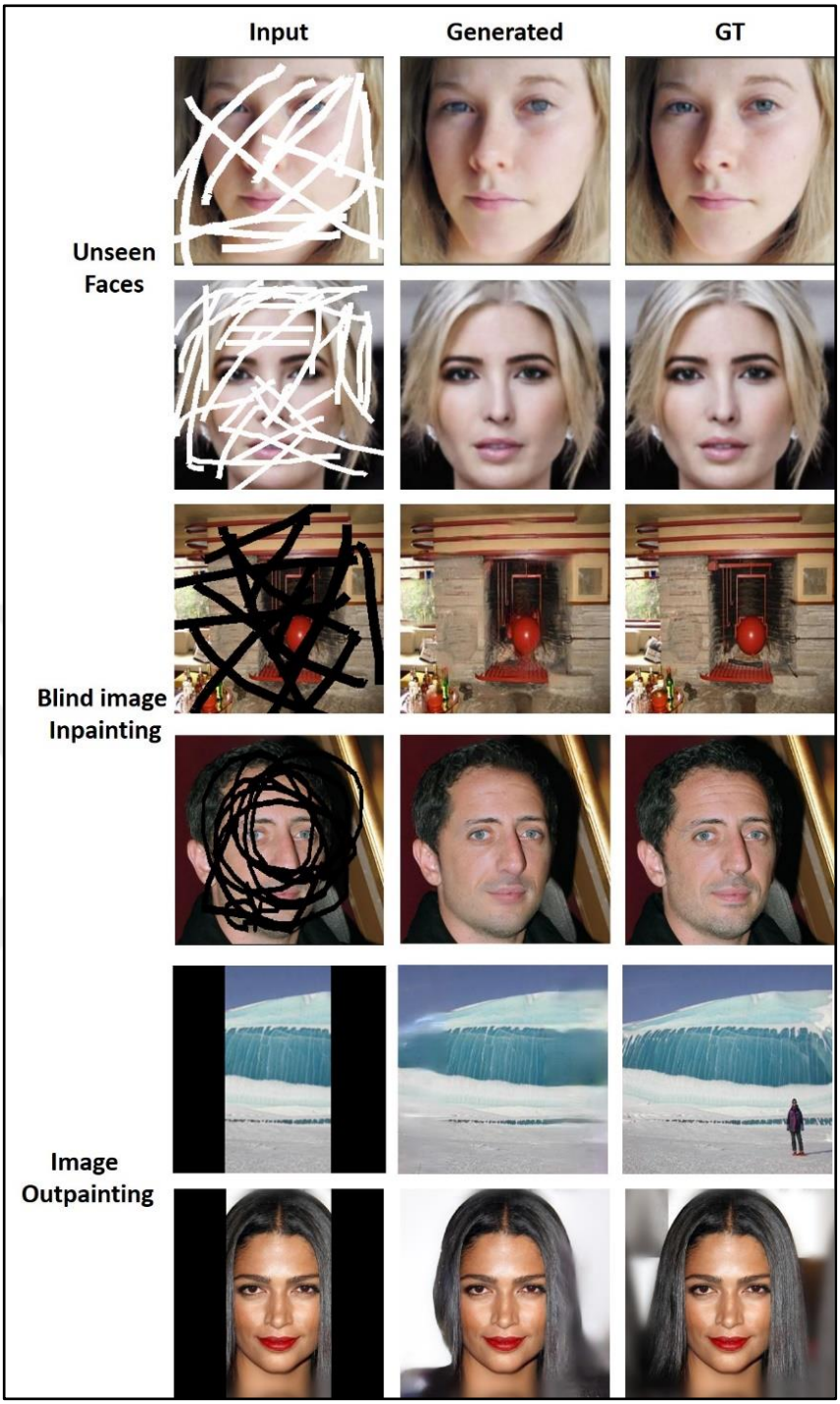

Figure 4.11: Scalability of the ETMG approach on several tasks, namely, unseen faces, blind image inpainting and image out-painting.

To verify the generalization capability of the proposed method, we test our pretrained model on images from the internet [76]. Although we run the model on unseen faces, it performs well in generating visually appealing results with realistic textures. The third experiment evaluates the approach to blind image inpainting. During training, we give only the corrupted image without the mask. We obtain

promising results, confirming that the proposed approach can be applied to other real-world applications. Note that the proposed method achieves higher performances in the image inpainting task since the mask guides the model to distinguish between valid and missing pixels. In the last experiment, we investigate the image outpainting task [77]. We mask 1/4 in the left and the right of the image, and we retrain and test our model on CelebHQ [53] and Places2 [72] datasets. This experiment reports lower performances compared to all tasks. We can explain this by that image outpainting includes two challenges: the missing mask channel and having two large separate corrupted regions.

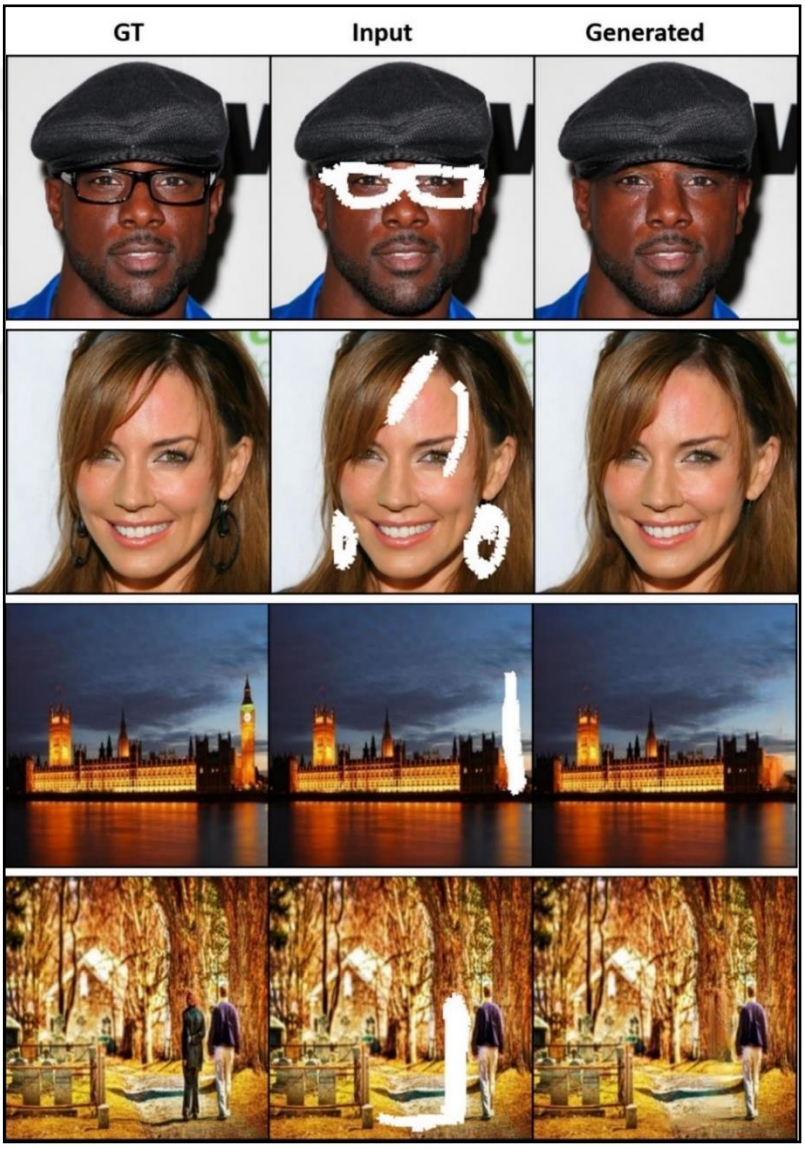

Figure 4.12: Object removal and image editing using the ETMG model.

4.4.5. Interactive editing

Our method allows users to remove unwanted objects by interactively drawing the input masks. At the same time, it can robustly recover the corrupted parts without artifacts. In both cases, the generated images have realistic texture and global semantic consistency. We provide some results of the interactive inpainting in Figure 4.12. Our approach robustly removes the glasses and face accessories around complex textured objects such as eyes and hair in the CelebHQ [53] dataset. Furthermore, it provides plausible images on the Places2 [72] dataset that includes crowded scenes.

4.5. Summary

This chapter presents a GAN-based image inpainting approach that addresses two main problems in the image inpainting field. Namely, decreasing the inference time and ensuring fine-grained textures in the corrupted regions. To achieve the first goal, we propose efficient progressive generators and discriminators to stabilize the training and improve performances. Filling in low-resolution images is less challenging for GANs due to the small dimensional space. Meanwhile, it guides higher resolution generators to learn the global structure consistency of the image. To ensure fine-grained textures, we employ an LBP-based loss function in the final resolution. Extensive experiments show that our model outperforms state-of-the-art and speeds up the inference time. Also, we apply the proposed approach to other related tasks, such as image outpainting and blind inpainting. Results demonstrate the scalability of our model even with large mask regions.

5. LEARNING TO INPAINT BY PROGRESSIVELY GROWING THE MASK REGIONS

5.1. Introduction

Recently, generative-based image inpainting methods produce visually plausible images. However, they still have difficulties generating the correct structures and colors as the masked region grows large. This drawback is mainly due to the training stability issues of GANs. In this study, we address this limitation using a new curriculum-style training approach. The proposed method increases the masked region size progressively in training time. During inference, the user gives variable-size masks at arbitrary locations. Incorporating this technique in GANs stabilizes the training and provides better inpainting performance.

Deep learning methods [11][12][15][3] applied GANs [14] to fill in masked regions by learning from large image datasets. They outperform the traditional inpainting methods [8][9][33] both qualitatively and quantitatively. However, some of these methods [11] fill in the center of the image, which may fail to inpaint variable size regions. Furthermore, they suffer from artifacts and need post-processing steps to correct the resulted image [12]. Therefore, understanding the structure and different objects in the scene helps to achieve high-quality image completion.

Although GANs fit the inpainting problem very well, they suffer from stability problems that lead to mode collapse and over-fitting. To address these limitations, [78] provides architectural guidelines and optimization hyper-parameters that lead to better synthesis results. Moreover, a multi-stage generation approach introduced in [22] creates high-quality images by progressively adding layers to the generator and the discriminator. Furthermore, [53] improves [22] by controlling the visual features of the image in different scales through the adaptive normalization layer [66]. Some works addressed the loss functions improving the training stability including, Wasserstein distance [30], Least Squares [28] and Energy-based GANs [79].

Another attempt to stabilize the training of GANs is to employ a curriculum learning (CL) approach [43]. It achieved a lot of success in many tasks, including natural language processing [80][81], image recognition [82] and generation [44]. CL is a setting that gradually reveals training samples to the model from the easiest to the

most difficult. Inspired by this idea, we propose a curriculum-style strategy to progressively train an effective generator by growing the size of the masked regions in the context of image inpainting. The intuition was that the generator and the discriminator networks solve the inpainting problem starting from simple to much harder inpainting regions. By simple, we mean small mask regions with basic structures that can be inpainted without global object structures. On the other hand, hard means large mask regions that need a local and global understanding of the scene.

We validate our approach using several models of different architectures and loss functions. The first one is a customized model that contains two networks: a deep residual convolutional generator [83] and a multi-scale discriminator. The latter one criticizes the quality and the relevance of the completed image on different scales. In the generator, we replace the vanilla convolutions with the gated convolutions introduced in [3]. They proved that it is a good replacement for vanilla convolutions in the context of image inpainting. The other methods are two of the state-of-the-art models [11] and [3]. We conduct two experiments: fixed versus progressively growing masked regions on the previously stated models. Furthermore, to show the effectiveness of our approach, we check if a simple reconstruction loss is sufficient to stabilize the generator for the first training iterations. In another setup, we use a fixed masked region then gradually increase the adversarial loss weight. We report qualitative and quantitative results on the MSCOCO [52] and CelebA [73] datasets. The quantitative metrics include MAE, PSNR, Inception score (IS) [48] and Fréchet Inception Distance (FID) [50] quality metrics. Our contributions are as follow:

- We propose the progressively growing of the masked regions as a GAN stabilization technique for image inpainting tasks.
- We compare the usage of fixed versus progressively growing mask regions using different architectures and loss functions, and we report the qualitative and quantitative results on two challenging datasets.
- We investigate other training stabilization setups and compare them against our approach.

5.2. Approach

5.2.1. Curriculum learning

Curriculum Learning (CL) is a training approach that gradually reveals data to a neural network starting from the easiest to the most difficult [43]. It achieved great success in many tasks, including natural language processing, image recognition and generation. Unlike the traditional training approach of CNNs that uniformly samples mini-batches from the data distribution, [84] used CL to order the training samples by difficulty and creates mini-batches. This approach improves network accuracy and learning speed. [85] improves the generalization ability by increasing the dropout rate throughout training. [85] employed CL on GANs such that the discriminator criticizes difficult samples as training progresses. They augment the dimensionality of the sample space with additional random variables. This approach makes the task much difficult for the discriminator and prevents it from being over-confident. In the context of image inpainting, [86] utilizes a progressive generative network to fill in images with squared masks. The approach splits the task into different stages, where each one aims to do a part of the entire curriculum. After that, an LSTM framework chains all of them. In this study, we propose a curriculum-style strategy to progressively train an effective generator by growing the size of the masked regions in the context of image inpainting. The intuition was that the generator and the discriminator networks solve the inpainting problem from basic to large inpainting regions.

5.2.2. Progressive growing of the mask as a GAN stabilization technique

GANs are hard to train due to their nature, which depends on two networks having two sets of parameters optimized separately. That leads to many problems, including mode collapse, non-convergence, and vanishing gradients. The inpainting task is strongly affected by robust adversarial loss functions, stable architectures, and GAN stabilization techniques. We focus on the last point and propose a simple yet effective training technique to stabilize the training of GANs in the context of image inpainting. The process is as follows: the generator starts by solving a simple problem. After each k iterations, the corrupted region grows till the region size reaches the half size of the image, as illustrated in Figure 5.1. By simple, we mean that the mask region

contains basic structures (textures) while the hard refers to the mask region has complicated structures and objects. We claim that, in the beginning, the generator easily inpaints narrow parts since the adversarial loss is responsible for an easy problem that is simply a reconstruction in this case. Then, the difficulty of the problem increases as we grow the width of the mask. Thus, the generator can fill in the half size of the masked region without much difficulty. That makes the adversarial loss stable in the successive k training iterations. The training process continues this way till a specified maximum width. We will investigate this claim by reporting the quantitative results of each k iterations using different mask sizes.

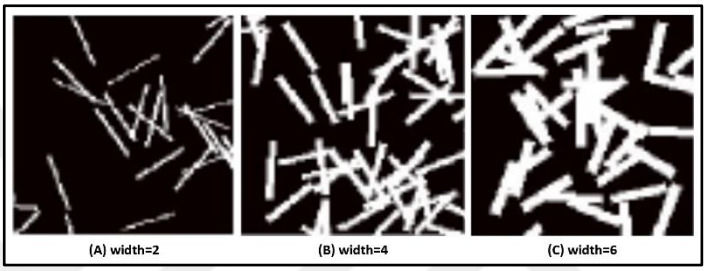

Figure 5.1: Illustration of the PGM approach.

5.2.3. Architectures and Training

To validate our approach, we use different models: our customized model illustrated in Figure 5.2, CA [19] and the GC [3].

Our customized model**:** the generator has two subnetworks, an encoder network that down-samples the size of the input to $1/4$ the original size followed by two residual blocks. We duplicate the number of filters after each gated convolution and residual block. The decoder network is the reverse order of the encoder. Instead of using transposed convolutions as generally done in decoders, we use bilinear interpolation before applying gated convolutions. The last convolution layer outputs an RGB image. In the discriminator network, we use a multi-scale architecture that contains five convolution layers. The latter down-samples the feature map size and increases the number of filters, and the last two convolution layers have the same number of filters. The discriminator outputs an array of tensors for each image scale. Instead of using Batch Normalization [87] that causes inference issues when the batch size is small, we use the Instance Normalization [88] that normalizes each sample independently across

spatial locations. Also, it provides visual and appearance invariance, and it is agnostic to the contrast of the image. The loss functions include the LSGAN loss [28], L1 loss between the generated and the ground truth image. Finally, we use the Perceptual loss using a pre-trained VGG network.

The Context-encoder model (CE) optimizes an autoencoder network to produce a rectangular hole in the center of the image. The discriminator considers the latter as fake and the center of the ground truth image as real. The training requires two loss functions: a pixel-wise reconstruction loss and an adversarial loss.

The Free-form inpainting model (GC): the generator has the same architecture as [19] followed by a refinement network without residual connections. The discriminator is a Patch-GAN that classifies image patches of size 70×70 as real or fake. Thus, there is no need for a global and local discriminator as in [12]. Furthermore, the networks do not add any normalization layer. It computes two loss functions: A Hinge loss and a reconstruction loss. It does not include any perceptual or style loss.

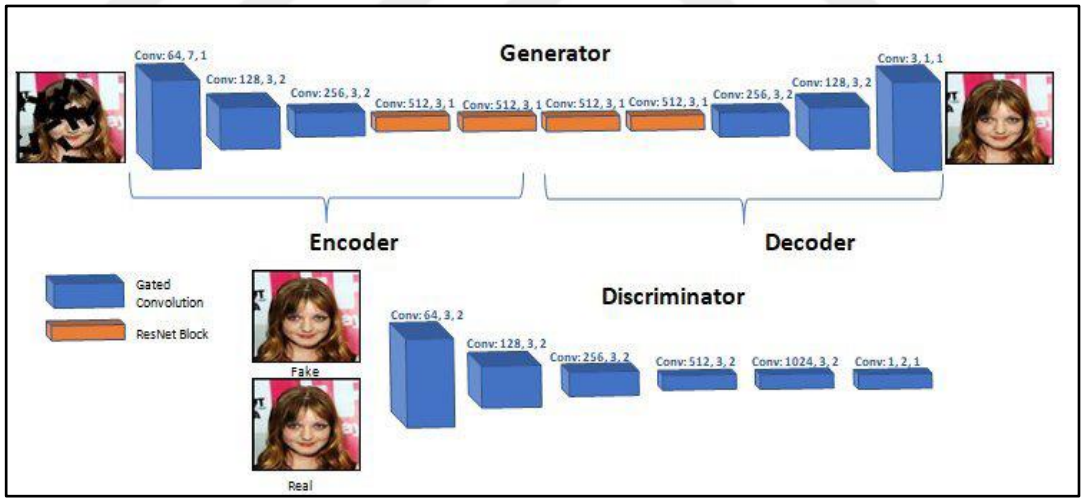

Figure 5.2: Overall architecture of the Generator and the Discriminator of the PGM model.

5.3. Experimental evaluation

5.3.1. Datasets

We experiment on challenging image datasets used in several computer vision tasks. The MSCOCO [52] dataset contains cluttered scenes with color and structure variations. The CelebA [73] dataset contains cropped faces that have fewer structure changes. We train on 200k and 82k training images defined in CelebA and MSCOCO, respectively. We test the performance on 10000 random validation images for the CelebA dataset and 5000 test images for the MSCOCO dataset.

5.3.2. Implementation details

We show the impact of our proposed training approach on the customized and state-of-the-art models [11][3]. As mentioned previously, we compare the fixed versus progressive mask size approach for all the models on the MSCOCO [52] and CelebA [73] datasets. Furthermore, to prove/disprove the correctness of the proposed method, we compare it with two other training strategies, as shown in Figure 5.2. The generator and the discriminator networks of the customized model use the Adam optimizer with a learning rate of 0.0002 and a batch size of 4. For [11] and [3], we keep the same hyper-parameters used in the original work. We train all the models for 1M iterations. Furthermore, we increase the mask size and the adversarial weight after 100k iterations. For a fair comparison, we fix the randomness seed while training the models to make sure that we give the same input (same masked regions) and the same order of the images to the models.

5.3.3. Experimental setups

Our main experimental setup is to investigate the fixed size masks versus the PGM approach. We use constant weights for both the reconstruction and the adversarial loss. To prove/disprove our claim, we explore the setups illustrated in Figure 5.3.

- We use the reconstruction and the adversarial losses during the whole training (first row).

- We train the model for half of the training time using the reconstruction loss. We use the adversarial loss and the reconstruction loss for the rest of the training, (second row).
- We use the reconstruction loss using a fixed weight. We linearly increase the weight of the adversarial loss after each k iterations (third row).
- Our PGM method progressively increases the mask size after each k iterations. The reconstruction and the adversarial losses remain fixed (fourth row).

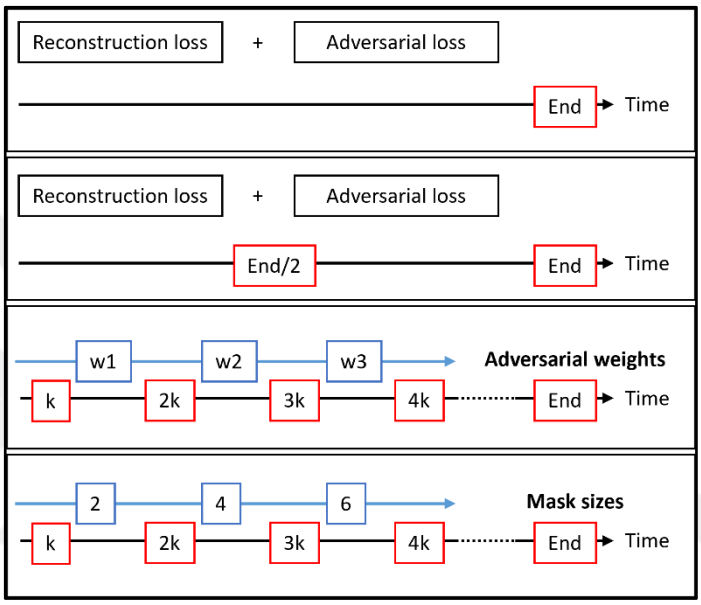

Figure 5.3: Different training setups on the PGM model.

5.3.4. Comparison plan

Unlike the state-of-the-art models that show their superiority, we aim to confirm the impact of our proposed training scheme (Progressive growing) and the two setups described above. We compare our customized model against CE [11] and GC [3]. To adapt our training approach to the CE model, we train using a small rectangle in the middle of the image. Then progressively increase the rectangle size to reach the half size of the image. [3] adds the sketch as an additional input to the model. To ensure a fair comparison, we only input the image and the mask. We test on the MSCOCO [52] and CelebA [73] datasets for the different setups on our customized model and [3]. We report the quantitative comparison using MAE, PSNR, IS, and FID. Furthermore, we show the output of our customized model versus [11] on different training schemes in the qualitative comparison. Since the CE model input is a fixed central mask in the

middle of the image, we do not compare it against the other models. Thus, we only report the qualitative and quantitative results of the different setups against each other. We do not perform any post-processing step for all the models. Due to hardware restrictions, we use images of resolution 128×128 in both datasets. We implement the models using Pytorch v1.1.0, CUDA v10.0, CUDNN v7.5.1, and the hardware GPU is NVIDIA GTX 1080 Ti. The training takes roughly five days per experiment.

5.4. Results and discussion

5.4.1. Quantitative evaluation

We experiment using image inpainting evaluation metrics including MAE, PSNR, IS and FID to quantify the performance of the training approaches. We calculate the MAE and PSNR using the output of the model and the ground truth images. We use the InceptionV3 model pretrained on the ImageNet dataset [24] to calculate the IS and FID scores. We see that our progressive growing approach improves the performance (MAE, PSNR, and FID) of all the models in the MSCOCO (Figure 5.4, Figure 5.5, Figure 5.6 and Figure 5.7) [52] and CelebA (Figure 5.8, Figure 5.9, Figure 5.10 and Figure 5.11) [73] datasets. Meanwhile, the results of the other three setups are not deterministic since they outperform each other depending on the model and the dataset. The IS metric is based on the classification probabilities. Therefore, it does not give a stable performance quantification on the CelebA dataset since the latter one contains only one class (faces). As claimed previously, to prove the effectiveness of our training approach, we experiment using different mask sizes. Our method outperforms the other setups on the two datasets in most cases. To apply our curriculum learning training approach to free-form mask models, we must control the width, height, orientation and number of masks in the images. On the other hand, applying it on [11] is easier since we can control the size of the masked regions (rectangular mask shapes). However, the performance of this model is still low compared to the other models due to its local consistency nature and the use of the standard convolution layers. Although our customized model has a larger number of parameters than [3], the latter outperforms it in all the training approaches in the MSCOCO and CelebA datasets (Table 5.1**)**. This result can be explained by the usage of a refinement network in [3].

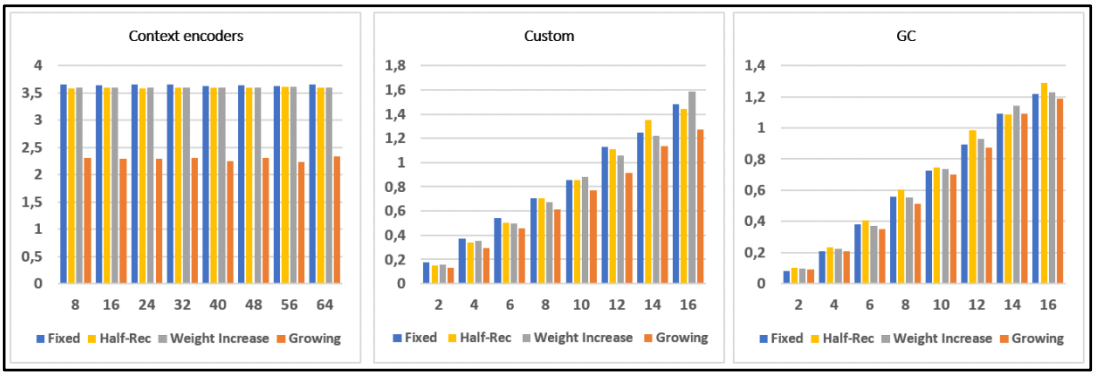

Figure 5.4: Quantitative comparison using MAE of the different training setups of the PGM model on the MSCOCO.

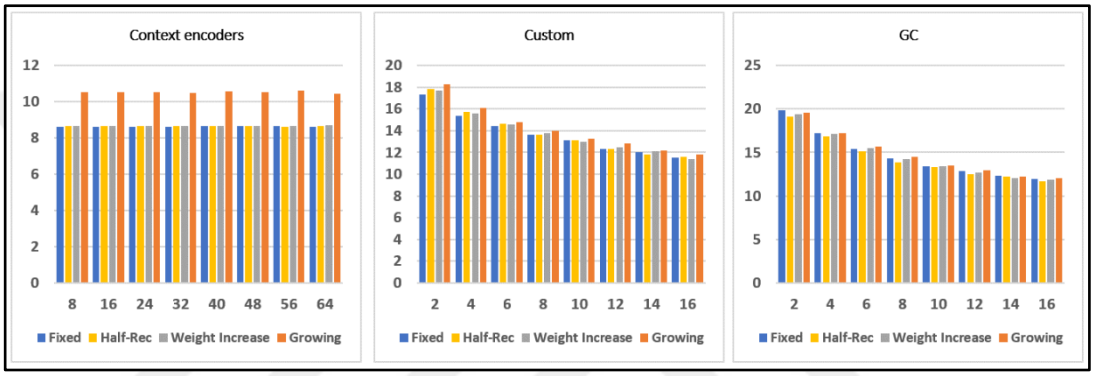

Figure 5.5: Quantitative comparison using PSNR of the different training setups of the PGM model on the MSCOCO.

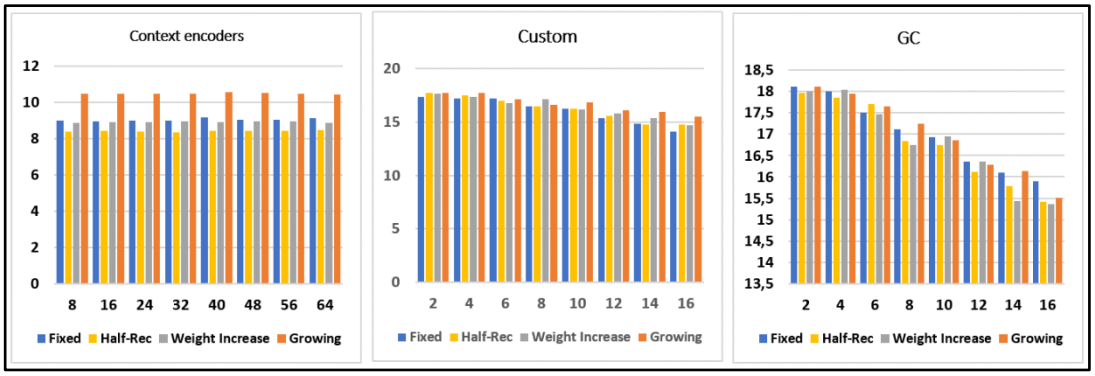

Figure 5.6: Quantitative comparison using IS of the different training setups of the PGM model on the MSCOCO.

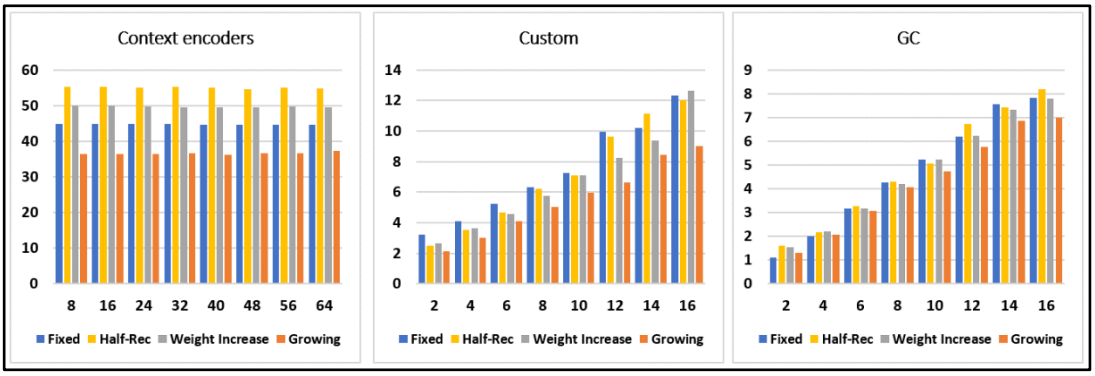

Figure 5.7: Quantitative comparison using FID of the different training setups of the PGM model on the MSCOCO.

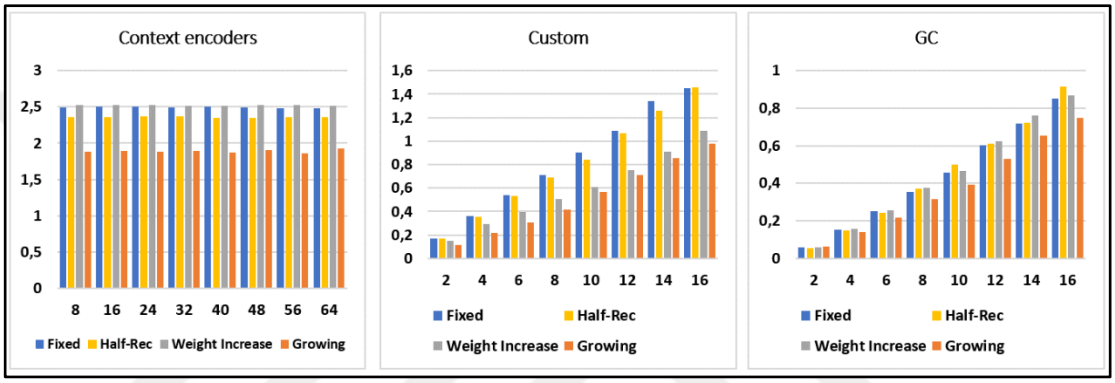

Figure 5.8: Quantitative comparison using MAE of the different training setups of the PGM model on the CelebA.

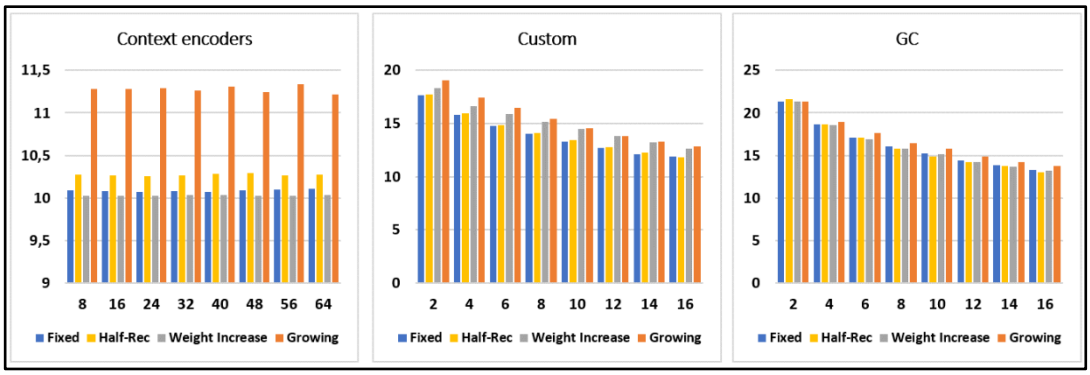

Figure 5.9: Quantitative comparison using PSNR of the different training setups of the PGM model on the CelebA.

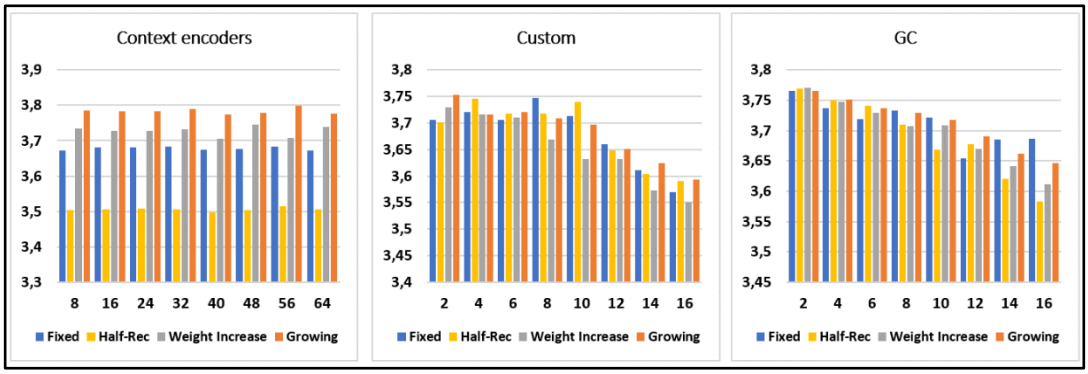

Figure 5.10: Quantitative comparison using IS of the different training setups of the PGM model on the CelebA.

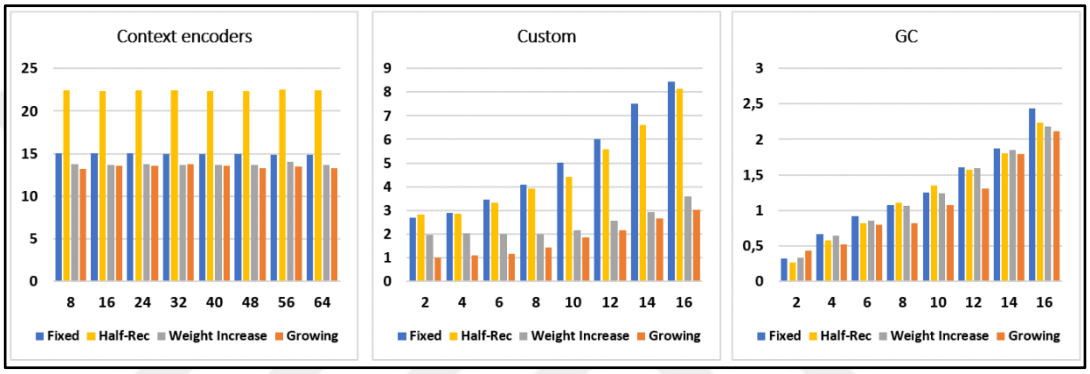

Figure 5.11: Quantitative comparison using FID of the different training setups of the PGM model on the CelebA.

Table 5.1: Quantitative comparison of conventional vs. PGM approach on CelebA.

Metric	Fixed	Growing
MAE (CE)	2.479	**1.92**
MAE (Custom)	1.450	**0.98**
MAE (GC)	0.849	**0.74**
PSNR (CE)	10.10	**11.21**
PSNR (Custom)	11.90	**12.85**
PSNR (GC)	13.31	**13.78**
IS (CE)	3.67	**3.77**
IS (Custom)	3.57	**3.59**
IS (GC)	**3.68**	3.64
FID (CE)	14.88	**13.26**
FID (Custom)	8.44	**3.03**
FID (GC)	2.44	**2.11**

5.4.2. Qualitative evaluation

We compare the fixed and the progressive growing training approach using our customized model and GC [3] on the MSCOCO [52] and CelebA [73] datasets. As seen from Figure 5.12, the custom model does not generate visually realistic images on the fixed setup. Our proposed training approach improves it to complete the missing parts more robustly, but it still generates artifacts compared with [3]. The latter can generate smooth and plausible images without our training approach. However, blurriness appears when we increase the mask size. On the other hand, applying the progressive growing approach to [3] composes a stable model that produces fewer artifacts. CE [11] uses a rectangular shape mask in the center of the image. For this reason, we compare only the fixed versus growing training approaches. Figure 5.13 shows that although applying our approach to that model does not give plausible and natural images, it improves the results of the original model by removing the artifacts around the rectangular mask.

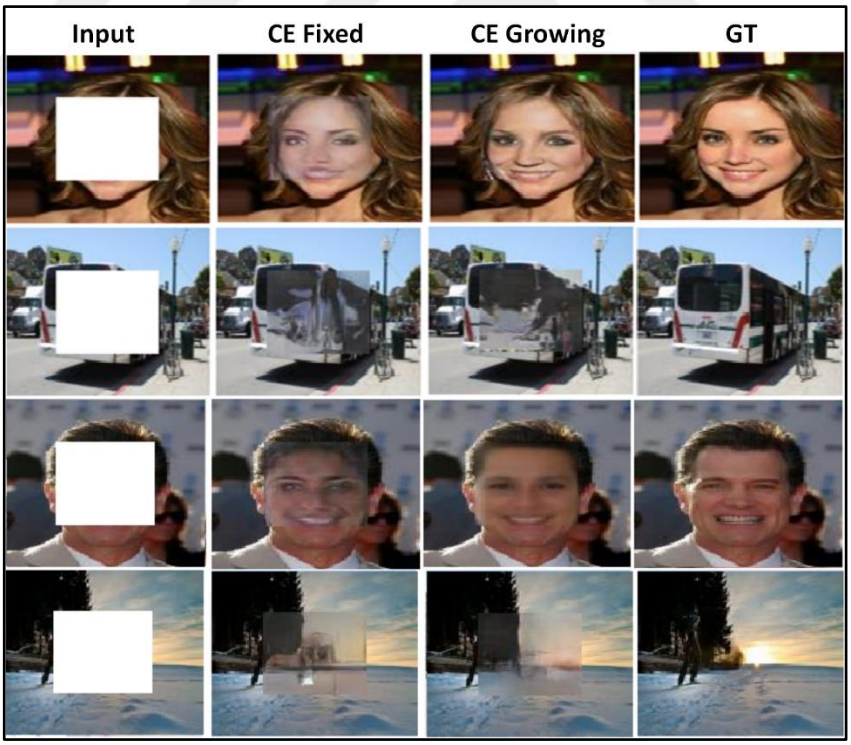

Figure 5.12: Qualitative comparison of the conventional vs. PGM approach using block mask.

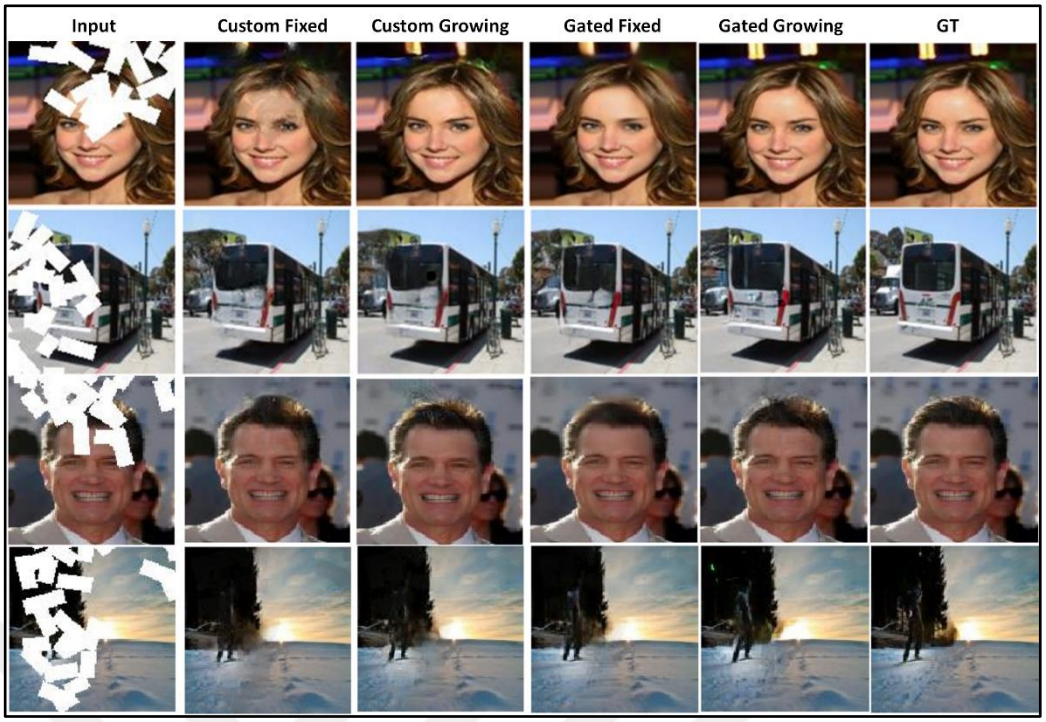

Figure 5.13: Qualitative comparison of the conventional training vs. PGM using free-form mask.

5.5. Summary

This chapter proposes a GAN-stabilization technique in the context of image inpainting. To generate the correct structures and colors in large holes, we employ curriculum-style training by progressively growing the masked regions. Incorporating such an approach in GANs stabilizes the training since the generators and discriminators gradually solve the problem. Results show that this technique provides better color consistencies and captures object continuities. Also, it improves the performance of several state-of-the-art methods on public datasets.

6. IMAGE INPAINTING USING DEEP MULTI-RESOLUTION PATHS AND ADAPTATIVE LOSS FUNCTIONS

6.1. Introduction

Learning-based approaches exploit the fast progress of CNNs and GANs [14] to learn high-level features from large-scale datasets. They establish a robust mapping between the corrupted and the ground truth images. Early GAN-based methods assume that the model can implicitly acquire information from far spatial regions. However, most of these methods generate visually significant artifacts leading to distorted structures and non-realistic texture details [11]. Recent studies work around these problems by dividing the inpainting problem into multiple stages, such that the first one predicts the image structure, including edges [17], segmentation maps [18] and contour [46]. The last stage exploits the reconstructed information to guide the completion task and obtain more realistic images. However, these architectures significantly increase model size, complexity and inference time. Furthermore, the performance is related to the prediction stage that requires extra supervision (ground truth data). Other approaches exploit information near corrupted regions to predict the missing pixels using the context attention layer [19]. All the mentioned methods use encoder-decoder architectures, such that the encoder maps the corrupted image to a latent space, and the decoder reconstructs the ground truth image. This operation may not restore texture details due to small receptive fields. [12] addresses this problem using dilated convolution layers. However, it results in gridding problems [54] since it leaves pixels near the center pixel non-exploited by the convolution filters. [41][15][89] employ a UNet-based [42] hierarchical network to avoid information loss. However, each scale is given to few convolution layers, which limits the learning of high and low-level features. In this work, we propose a deep multi-resolution path architecture to robustly complete masked images. Specifically, we employ a deep network for each scale to increase the receptive field and recover high-frequency information from several input resolutions. Each path contains many ResNet layers [83] without downsampling to keep original image details. We concatenate the feature maps of previous and current network paths to ensure local and global image consistency. Moreover, we observe that the images may include different texture

complexity and mask sizes. Hence we propose a new function that gives different weights for each image in the reconstruction and the adversarial losses to recover fine-grained textures. In particular, we eforce the generator to focus on images exhibiting large masks and complex textures in the corrupted regions. To validate the proposed approach, we conduct our experiments on two standard datasets Places2 [72] and CelebHQ [53]. Quantitative and qualitative results demonstrate that our method generates realistic images with coherent global semantic structure and fine-grained textures. Furthermore, it shows superior performance against the state-of-the-art. We summarize our contributions as follow:

- We introduce a new GAN-based image inpainting framework that exploits multi-resolution paths to enlarge receptive fields and improve performance.
- We propose a new training approach that employs a weighted loss function to enforce fine-grained texture details.
- The proposed method generates visually appealing images and outperforms state-of-the-art without using additional information or post-processing.

6.2. Approach

Previous image inpainting methods employ an encoder-decoder architecture where the input and the output are high-resolution images. The encoder down-samples the feature maps to $1/4$ of the image resolution then applies a series of residual convolution layers [83]. However, due to the small receptive field at this resolution, convolution layers cannot capture information about distant patches in the image. We overcome this problem using different corrupted image resolutions fed to deep network paths (Figure 6.1). Specifically, having a deep network path on the 32×32 image increases the receptive field to cover all the patches in the image without the need for down-sampling or computationally expensive attention layers [19][3][89]. In other words, we do not miss high and low-frequency information of the original image since we do not use any stride-convolution layers neither on low-resolution images nor on high-resolution ones.

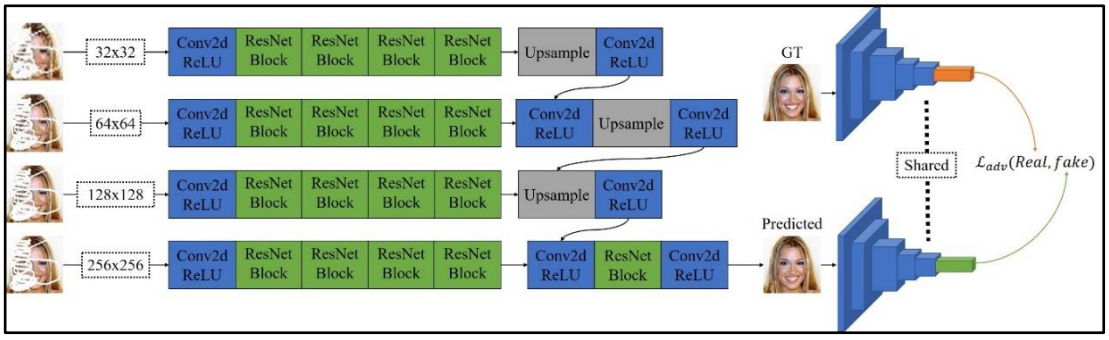

Figure 6.1: The overall architecture of the DMRP model.

6.2.1. Architecture

Our generator has four network paths where each one takes a corrupted image of size $n \times n$ and feeds it to a sequence of convolution and ResNet [83] blocks. We use the nearest neighbor up-sampling to increase the resolution of the 32×32 network path to match the output of the 64×64 resolution path. These two feature maps are concatenated and fed to convolution and upsampling layers to learn high-level features. We repeat the same procedure for the 128×128 and 256×256 resolutions to output the inpainted image. We give the latter and the ground truth image to a PatchGAN discriminator [26] that criticizes the different image patches.

6.2.2. Adaptative weighted loss

Loss functions of the previous image inpainting approaches give the same importance for all images regardless of texture complexity and the mask size. Thus, during back-propagation, the parameters of the generator are updated equivalently for all the samples. Giving the same weight for all the images makes the network biased towards easy examples. By easy, we mean images with small holes and simple texture details. In our method, we assign high weights to corrupted images with complex textures and large masks. We extract the edges from the ground truth image using the Canny edge detector [68]. We define the texture complexity by calculating the ratio between the number of edges and the image size (ones and zeros in the corrupted region). Finally, we normalize the weights in the [0, 1] interval based on the largest mask in the dataset (training masks). We refer to our adaptative weighting function as $AdaW(.)$.

6.2.3. Loss functions

Let T, M, I and O be the ground truth, mask, corrupted and generated images, respectively. Let N be the number of images in the batch. We resize T and M to match different input sizes $n \times n$, namely: 32×32, 64×64, 128×128 and 256×256. The input $C_{n \times n}$ is the channel-wise concatenation of $I_{n \times n}$ and $M_{n \times n}$, we define the generator in (6.1):

$$O = G(C_{32\times32}, C_{64\times64}\, C_{128\times128}, C_{256\times256}) \tag{6.1}$$

Reconstruction loss: we use the L1 loss between T and O to reconstruct the image using (6.2):

$$L_{rec} = \sum_{i=1}^{N} AdaW(T_i, M_i)\, \|T_i - O_i\|_1 \tag{6.2}$$

Adversarial loss: we use the RaLSGAN loss as defined in [90] using (6.3) and (6.**4**) for the discriminator and the generator, respectively:

$$\begin{aligned} L_d = E_{x_r \sim P, m \sim S}\Big[&(AdaW(x_r, m) \times D(x_r) \\ &- E_{x_f \sim Q}[AdaW(x_f, m) \times D(x_f) - 1)^2\Big] \\ &+ E_{x_f \sim Q, m \sim S}\big[(AdaW(x_f, m) \times D(x_f) \\ &- E_{x_r \sim P}[AdaW(x_r, m) \times D(x_r) + 1)^2\big] \end{aligned} \tag{6.3}$$

$$\begin{aligned} L_{adv} = E_{x_r \sim P, m \sim S}\Big[&(AdaW(x_r, m) \times D(x_r) \\ &- E_{x_f \sim Q}[AdaW(x_f, m) \times D(x_f) + 1)^2\Big] \\ &+ E_{x_f \sim Q, m \sim S}\big[(AdaW(x_f, m) \times D(x_f) \\ &- E_{x_r \sim P}[AdaW(x_r, m) \times D(x_r) - 1)^2\big] \end{aligned} \tag{6.4}$$

Where: $AdaW(.)$ is the proposed weighting function. P, Q and S are the distributions of real, generated and mask images, respectively. $D(.)$ is the discriminator network.

Overall loss: as defined in (6.**5**), we update the weights of the generator by summing up the losses in (6.2) and (6.3).

$$L_{overall} = 0.01 \times L_{adv} + L_{rec} \tag{6.5}$$

6.3. Experimental evaluation

We conduct our experiments using two public datasets Places2 [72] and CelebHQ [53]. Places2 dataset is commonly used for image inpainting since it contains 1.8M images and over 400 different scenes. The CelebHQ dataset includes 30K highly structured faces with high pose and background variations. We use the original train and test split for the two datasets. We employ free-form masks from [58], which automatically generate multiple holes with random shapes and sizes at different locations. In evaluation time, we divide the masks into four categories covering $10-20\%$, $20-30\%$, $30-40\%$ and $40-50$ of the image. We use Pytorch v1.6 [74] to implement the proposed method using CUDA v10.1 and cuDNN v7.6.4. We use Adam [59] optimizer with hyperparameters $\beta 1 = 0.5\ and\ \beta 2 = 0.99$, respectively. We set the batch size to 32, and we fix the learning rates to 10^{-4} the generator and the discriminator.

6.4. Results and discussions

6.4.1. Qualitative results

We qualitatively compare the proposed method against the mentioned state-of-the-art methods, namely Contextual Attention (CA) [19], Deep Fusion Network (DFNet) [41], Gated Convolution (GC) [3] and Structure Flow (SF) [16] in Figure 6.2. CA shows poor performances in the two datasets since significant visual artifacts appear, leading to mispresented structures and wrong boundaries. DFNet can capture a coherent global semantic, but artifacts still exist, especially on the CelebHQ dataset [53]. GC and SF generate smooth images with realistic textures. However, they still exhibit color discrepancies and row discontinuities in the predicted pixels. Our approach generates visually appealing images with meaningful structures and realistic

textures in the Places2 dataset [72]. Besides, it shows the most natural faces in the CelebHQ dataset with a high relevance between the holes and the background.

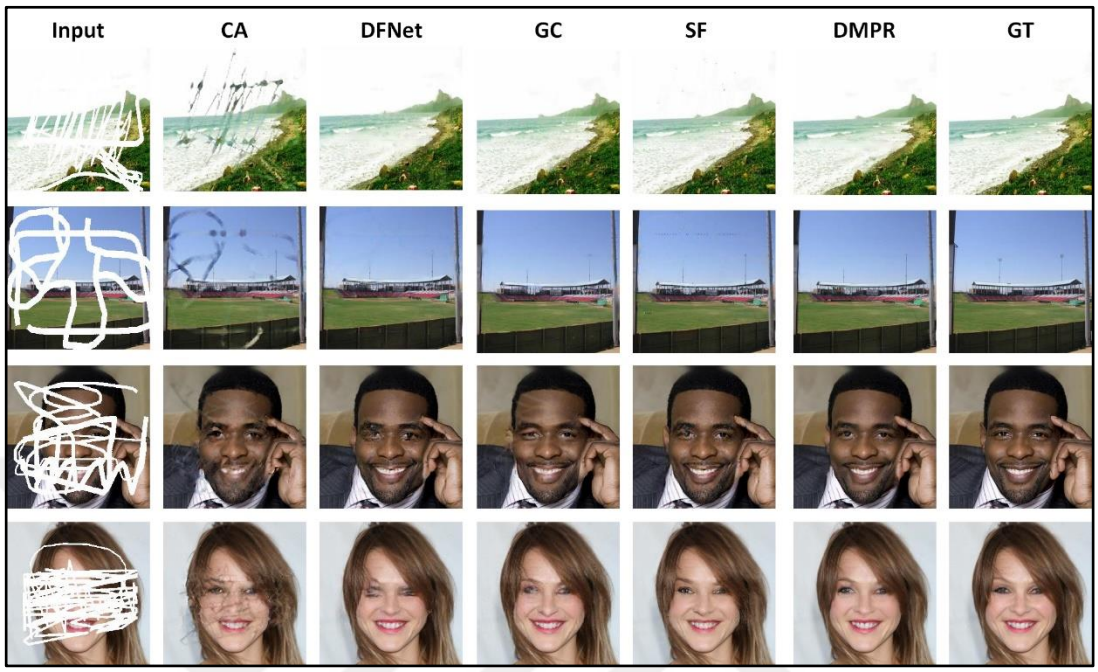

Figure 6.2: Qualitative comparison of the DMPR model on Places2 and CelebHQ.

6.4.2. Quantitative results

To quantify the performance of the proposed approach, we use three well-known assessment metrics following [16][17], including MAE, PSNR and SSIM. We report the evaluation results on Places2 [72] and CelebHQ [53] datasets in Table 6.1 and Table 6.2, respectively. We can see that our model performs favorably against all the compared state-of-the-art algorithms in all the metrics.

Table 6.1: Quantitative comparison of the DMPR model on Places2.

	Mask size	CA	EC	DFNet	GC	SF	DMPR
MAE^-	10-20%	0.019	0.013	0.010	0.011	0.012	**0.010**
	20-30%	0.033	0022	0.019	0.018	0.019	**0.017**
	30-40%	0.048	0.031	0.028	0.026	0.026	**0.025**
	40-50%	0.075	0.053	0.045	0.045	0.044	**0.042**
$SSIM^+$	10-20%	0.922	0.947	0.965	0.969	0.966	**0.970**
	20-30%	0.861	0.913	0.936	0.942	0.944	**0.946**
	30-40%	0.795	0.879	0.901	0.909	0.912	**0.914**
	40-50%	0.660	0.762	0.803	0.810	0.812	**0.814**
$PSNR^+$	10-20%	26.31	27.88	29.51	30.10	30.23	**30.55**
	20-30%	22.07	25.51	26.73	27.13	27.32	**27.68**
	30-40%	20.91	23.96	24.87	25.07	25.38	**25.49**
	40-50%	18.27	20.80	22.03	21.78	21.97	**22.42**

Table 6.2: Quantitative comparison of the DMPR model on CelebHQ.

	Mask size	CA	GC	SF	DMPR
MAE^-	10-20%	0.014	0.009	0.011	**0.008**
	20-30%	0.024	0014	0.015	**0.012**
	30-40%	0.033	0.021	0.018	**0.016**
	40-50%	0.052	0.031	0.028	**0.026**
$SSIM^+$	10-20%	0.953	0.982	0.984	**0.985**
	20-30%	0.918	0.968	0.971	**0.976**
	30-40%	0.881	0.950	0.950	**0.962**
	40-50%	0.796	0.899	0.912	**0.919**
$PSNR^+$	10-20%	28.55	32.53	33.26	**34.32**
	20-30%	25.54	29.73	30.42	**31.19**
	30-40%	23.58	27.80	28.74	**29.56**
	40-50%	21.03	25.05	25.63	**26.48**

6.4.3. Ablation study

We conduct a set of experiments on the CelebHQ [53] dataset to investigate the impact of each component. To evaluate our deep multi-resolution path architecture, we compare it against a UNet-based generator. In other experiments, we progressively remove one path from the entire framework. Table 6.3 shows that the proposed architecture outperforms the UNet-based generators since it involves larger receptive fields. Furthermore, combining several learning paths improves quantitative results

and generates visually appealing images Figure 6.3. To investigate the impact of the proposed weighting function, we remove it from the loss function and retrain the model using the traditional reconstruction and adversarial losses. Results validate the proposed weighting function that presents more realistic texture details and high-quality images in Figure 6.2 and higher scores in Table 6.3.

Table 6.3: Quantitative ablation studies of the DMPR model on CelebHQ.

Methods	MAE^{-}	$SSIM^{+}$	$PSNR^{+}$
Full	**0.015**	**0.960**	**30.38**
Full w/o 32 path	0.021	0.945	28.85
Full w/o 32 and.64 paths	0.025	0.918	26.44
Full w/o 32,64 and 128 paths	0.032	0.881	24.17
Full w/o AdaW	0.016	0.955	30.15
UNet-based generator	0.022	0.939	28.67

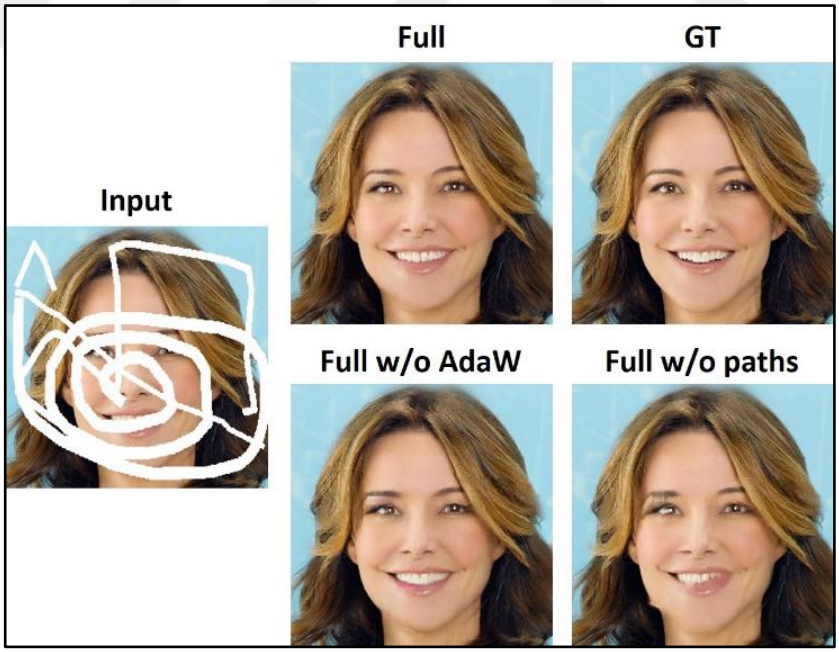

Figure 6.3: Qualitative ablation studies of the DMPR model on the CelebHQ.

6.4.4. Model efficiency

We show the efficiency of our DMPR model against CA, GC, and SF in terms of the number of floating points, the number of parameters, and the runtime on both GPU and CPU (Table 6.4). We can see that our model is very comparable to CA. However, the latter reports inferior performance and exhibits high floating points due to the CAM. GC and SF are less efficient and show a large inference time.

Table 6.4: Computational time comparison of the DMPR model.

Model	GFLOPS	PARAMS (M)	CPU (ms)	GPU (ms)
CA	22.4	2.9M	383	18
GC	39.6	4.1M	490	27
SF	262.4	92.5M	810	36
DMPR	18.2	4.6M	390	18

6.5. Summary

This chapter presents GAN-based image inpainting that fills in the corrupted regions with fine-grained textures and coherent structures. To achieve this aim, we use multi-resolution deep network paths to enlarge receptive fields and ensure low and high-level feature learning. Besides, we employ an adaptative weighting mechanism in the loss functions to focus on images exhibiting large masks and complex textures in the corrupted regions. Experimental results show the superiority of the proposed approach compared against several state-of-the-art methods.

7. COMPARISON OF THE PROPOSED APPROACHES

7.1. Introduction

In this chapter, we conduct extensive experiments to compare our three proposed GAN-based image inpainting approaches SC, ETMG and DMP. We report qualitative and qualitative results on two public datasets, namely CelebHQ and Places2. Additionally, we use the proposed PGM technique as a GAN stabilization technique to improve the performance of the mentioned methods. Note that we use the same data splits, and we train the models for the same number of epochs to ensure a fair comparison.

7.2. Places2 dataset

In the first experiment, we report qualitative and quantitative comparisons of our proposed image inpainting approaches on the Places2 dataset. Furthermore, we employ our PGM as a GAN stabilization technique on DMPR and ETMG models. Specifically, we progressively increase the masked regions during training time from simple masks to much harder ones. We can see from Figure 7.1 that all the models generate smooth images with global image consistency and realistic texture details. Besides, they report competitive quantitative results without using large models or complex mechanisms such as CAM and perceptual losses. Also, Table 7.1 shows that the PGM technique improves the performances in DMPR and ETMG models in all the metrics that prove the effectiveness of curriculum learning in the context of image inpainting. Note that we did not report the experimental results of the SC model on the Places2 dataset because the latter does not include the segmentation labels, which are necessary to supervise the training.

Table 7.1: Quantitative evaluation of the proposed image inpainting approaches on the Places2 dataset.

Metric	DMPR	DMPR+PGM	ETMG	ETMG+PGM
MAE^{-}	0.023	0.021	0.019	**0.018**
$SSIM^{+}$	0.908	0.912	0.917	**0.919**
$PSNR^{+}$	26.13	26.62	26.67	**26.74**

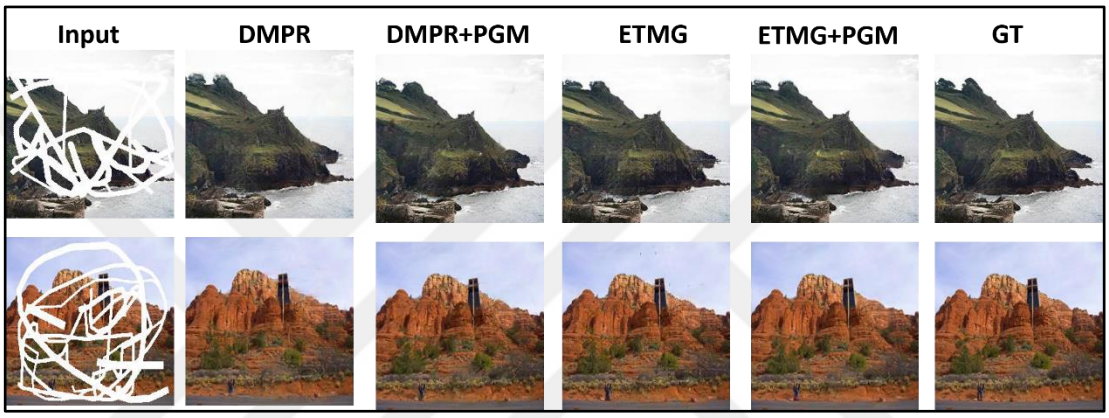

Figure 7.1: Qualitative evaluation of the proposed image inpainting approaches on the Places2 dataset.

7.3. CelebHQ dataset

In the second experiment, we report qualitative comparisons of our proposed image inpainting approaches on the CelebHQ dataset. We can see from Figure 7.2 and Table 7.2 that the ETMG model performs favorably compared with SC and DMPR. It generates the most natural faces with a high relevance between the masked regions and the background. Meanwhile, it reports superior quantitative performance in all the metrics. We can explain this by that the ETMG approach looks at different image scales using multiple efficient generators to ensure visually realistic images with both local and global structure consistencies. Also, the proposed LBP-based loss constrains better than segmentation labels and edges that cannot provide sufficient texture details in many cases, such as the face skin and uniform background. In other words, minimizing the loss between the LBP of the ground truth and the estimated images

ensures fine-grained textures. Note that SC and DMPR models also exhibit competitive performance.

Table 7.2: Quantitative evaluation of the proposed image inpainting approaches on the CelebHQ dataset.

Metric	DMPR	SC	ETMG	ETMG+SC
MAE^{-}	0.015	0.016	0.013	**0.010**
$SSIM^{+}$	0.960	0.955	0.964	**0.971**
$PSNR^{+}$	29.88	26.89	30.72	**31.19**

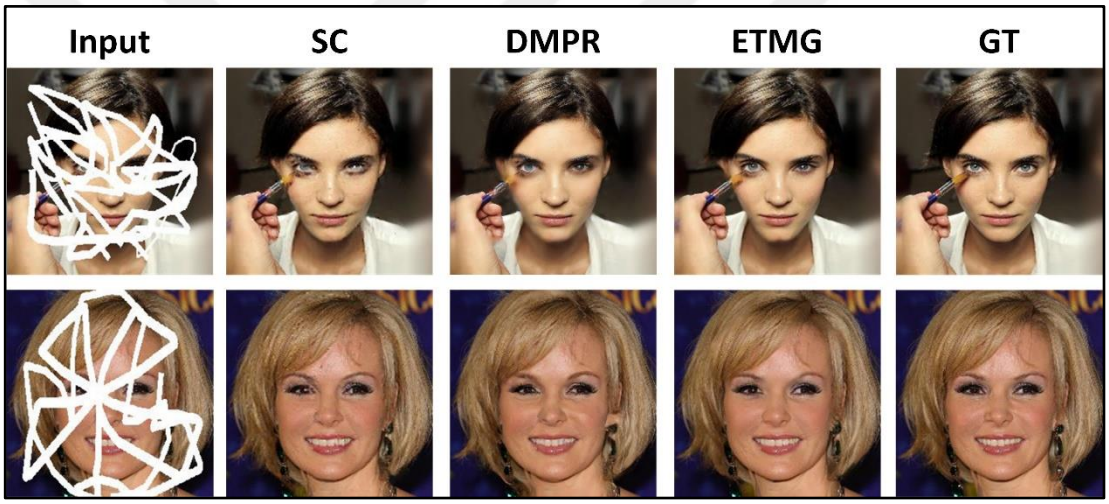

Figure 7.2: Qualitative evaluation of the proposed image inpainting approaches on the CelebHQ dataset.

In the last experiment, we combine the ETMG and the SC approaches on the CelebHQ dataset. Specifically, we estimate the coarse image and the segmentation labels for all the resolutions of the ETMG architecture. The progressive growing generators and discriminators stabilize GANs training and recover coherent structures. Meanwhile, the segmentation labels guide the generator of each resolution to estimate semantically meaningful images. Unlike the SC approach, we do not estimate edges, and we compensate them with the proposed LBP-based loss function that recovers more fine-grained textures. We can see from Figure 7.2 that our new composed approach yields more favorable results compared to all the individual methods. Also, the quantitative results in Table 7.2 show remarkable improvement over SC and

ETMG models. We note that we show the intermediate results of all the resolutions by providing the estimated images and their corresponding segmentation labels (Figure 7.3 and Figure 7.4).

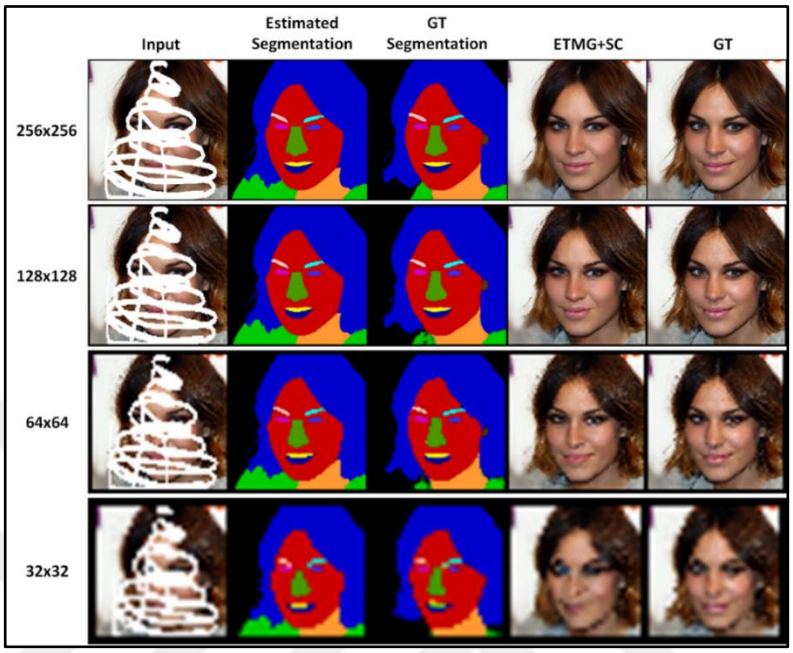

Figure 7.3: Example 1 of ETMG + SC intermediate results.

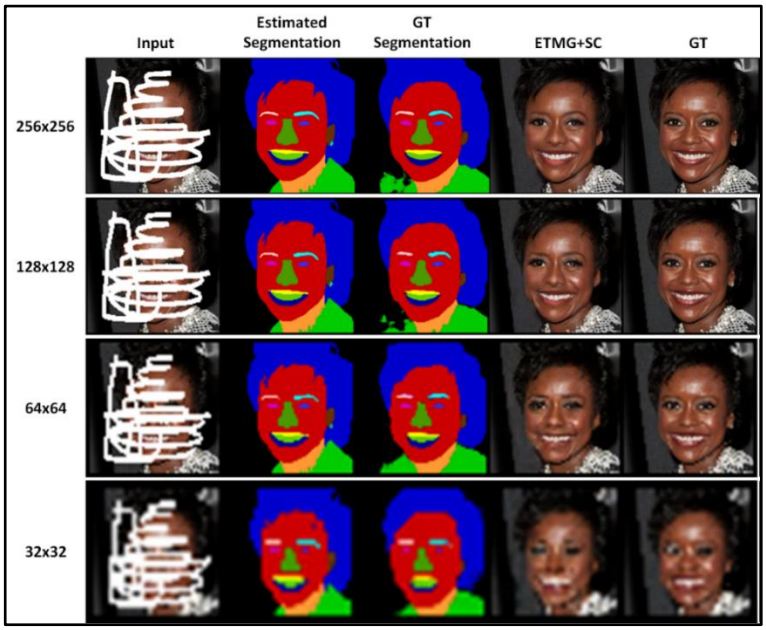

Figure 7.4: Example 2 of ETMG + SC intermediate results.

7.4. Computational time comparison

To compare the efficiency of our proposed image inpainting approaches, we report the number of floating-point operations, parameters and the runtime on CPU and GPU of our models on an Intel(R) Core (TM) i7-2600K CPU @ 3.40GHz and an NVidia Titan XP GPU. Table 7.3 shows that the ETMG model yields the lowest inference time. The reason is that for low-resolution images, a small generator is sufficient to model the data distribution. As the image resolution increase, we need fewer parameters since the global structure of the image is already filled in, and the next generator only focuses on restoring the details. Hence, the total number of floating-point operations remains smaller. In contrast, SC and DMPR directly handle high-resolution images. Consequently, their models need to have enough parameters to synthesis large regions in a nonprogressive way. We note that the combined approach is also efficient. However, it exhibits a higher number of parameters due to the additional segmentation label estimation decoders.

Table 7.3: Computational time comparison.

Model	GFLOPS	PARAMS (M)	CPU (ms)	GPU (ms)
SC	16.4	4M	374	16
DMPR	18.2	4.6M	390	18
ETMG	**9.5**	**3M**	**334**	**11**
ETMG+SC	10.2	3.8M	362	14

7.5. Summary and discussion

This chapter presents extensive experiments to compare the proposed image inpainting architectures, loss functions, and GANs stabilization techniques. Qualitative and quantitative results show that the combination of the SC and the ETMG yields the most effective model that synthesis visually appealing images with realistic texture details. Meanwhile, the ETMG is computationally efficient than all the proposed approaches. Also, the PGM technique improves the performance of the DMPR and the ETMG methods. Note that our four methods synthesize visually

appealing images with fine-grained textures on large corrupted regions. Meanwhile, all of them are computationally efficient than most of the state-of-the-art approaches. Also, we show the generalization of the ETMG method on images, which does not exist in the training or the validation set. Besides, we demonstrate the scalability of our approach on other related tasks such as image outpainting and blind image inpainting. In these tasks, we do not provide the mask that helps the model to focus on corrupted regions. Although these tasks are more challenging than the conventional image inpainting task, our ETMG approach still synthesizes realistic results. We test this method on block-wise masks, which are very difficult to complete, especially in the mask boundaries. Results show that the generated images have global and local consistencies and do not exhibit artifacts around the mask.

8. Conclusions

This chapter concludes the thesis and summarizes our main contributions by recalling the principal solved problems, the proposed approaches, and the obtained results. Then, it discusses the limitations of the proposed methods and suggests the possibilities for future research.

8.1. Summary of the contributions

In this thesis, we present four image inpainting methods that have been proposed to solve the following problems:

- Enforcing constraints about the input to explicitly guide the model to synthesize plausible contents.
- Reconstruct object boundaries and avoid blurry or semantically incorrect images.
- Enforce fine-grained textures for complex scenes and non-repetitive patterns.
- Speed up the inference time by removing inefficient attention mechanisms and replace them with efficient architectures.
- Generate visually appealing structures and texture as the masked region grows large.

To achieve the mentioned goals, we have started our research by designing several approaches such that each one focuses on addressing a specific problem.

The first study proposes a new end-to-end deep generative model for recovering the corrupted parts of a given image. Our multi-stage image inpainting method jointly estimates segmentation labels and object boundary edges from the coarse image. We demonstrate that combining these two estimated entities can represent the image structure and semantic information yielding realistic textures in the recovered image. Besides, we show that predicting this auxiliary information from the corrupted image decreases the inpainting performance.

The second method introduces an effective and efficient end-to-end GAN-based framework for image inpainting. Our approach employs progressive efficient generators to stabilize the training and improve the performances. We fill in different image sizes, such that the generators of higher-resolution profit from the previously inpainted regions. Moreover, we demonstrate that the proposed LBP-based loss

function constrains image inpainting and enforces texture details. We report quantitative and qualitative comparisons on public datasets. Experimental results show that the proposed approach generates realistic images with global structure consistency and fine-grained textures. Also, it outperforms state-of-the-art methods and significantly speeds up the computational time. Furthermore, it shows promising results for other related applications, such as image outpainting and blind inpainting.

The third method proposes a new curriculum-style training for image inpainting by progressively growing the masked regions. Experiments show that our model generates realistic and plausible images, even with large mask regions. Furthermore, it improves several inpainting models quantitatively, including the state-of-the-art for a wide variety of regular and irregular masks on several datasets.

The last study proposes a new generative-based approach for image inpainting. Combining the features from different scales and using deep network paths enlarge receptive fields and capture more relevant information. Our proposed weighting mechanism in loss functions improves the performance by focusing on complex textures in corrupted regions. Experiments show that our model restores fine-grained textures and achieves competitive performances against the state-of-the-art.

8.2. Limitation and future work

The results obtained during this thesis are globally promising and encouraging. However, this section discusses the limitation of the designed approaches and gives a clue about the possible solutions to overcome them. Also, we suggest directions for future research that are not explored in our study.

Our approaches do not add any uncertainty to the output, meaning that the model is fully deterministic and works as a one-to-one mapping function. Instead, a future model can add a latent variable from a pre-defined distribution to produce a one-to-many mapping. The latter can give a user control over the output to choose different results based on his/her preferences.

Another alternative is to learn from examplar images in the dataset, where the model is given many samples to decide the output in test time (Figure 8.1). Specifically, the first phase trains a shared encoder-decoder model to reconstruct unmasked images in the training set. The second phase involves another encoder-decoder architecture, which takes the concatenation (through a pooling operation) of

the masked image and the latent vectors of the pretrained shared encoder then a decoder network reconstructs the original image. During test time, we extract all the latent vectors of the training set before loading the model. Therefore, inference time remains efficient since the latent vectors reside in the memory.

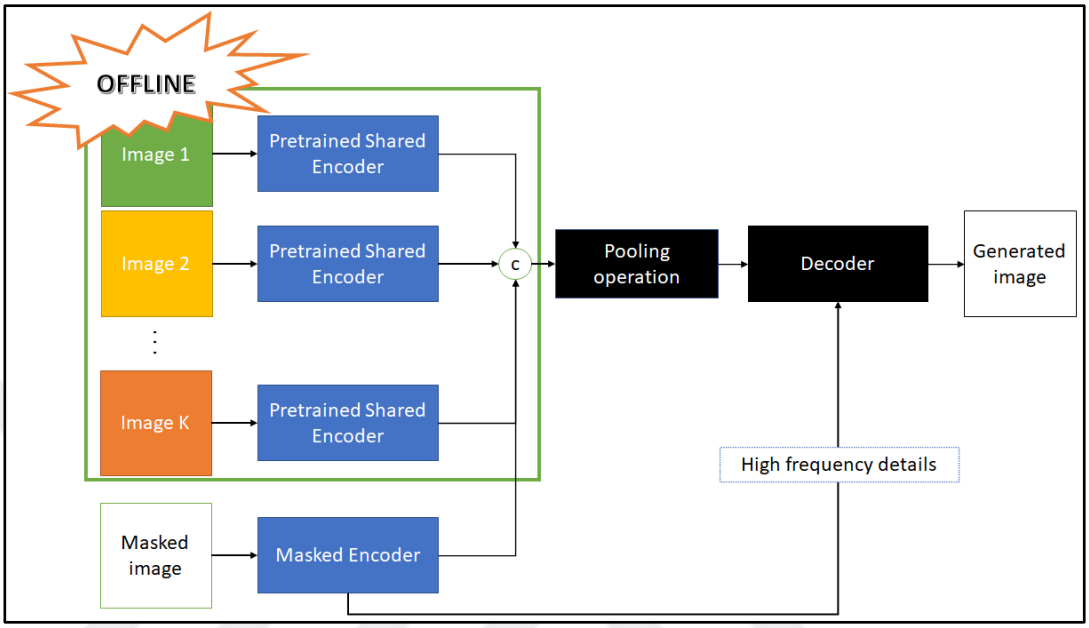

Figure 8.1: Exemplar-based image inpainting.

As future work, we also suggest employing the ETMG architecture to other image-to-image translation tasks, such as image denoising and image deblurring. The model can be extended to learn distributions in lower-resolution images and then proceed to higher resolutions.

REFERENCES

[1] Chen Y., Liu L., Tao J., Xia R., Zhang Q., Yang K., Xiong J., Chen X., (2021), "The improved image inpainting algorithm via encoder and similarity constraint", *Visual Computer, 37* (7), 1691–1705.

[2] Anstis S., (2010), "Visual filling-in", *Current Biology, 20* (16), 664-666.

[3] Yu J., Lin Z., Yang J., Shen X., Lu X., Huang T., (2019), "Free-form image inpainting with gated convolution", *International Conference on Computer Vision*, 4470–4479, *Seoul, Korea, 27 October-2 November*.

[4] Gui Y., Zeng G., (2020), "Joint learning of visual and spatial features for edit propagation from a single image", *Visual Computer, 36* (3), 469–482.

[5] Yamauchi H., Haber J., Seidel H. P., (2003), "Image restoration using multiresolution texture synthesis and image inpainting", *International Conference on Computer Graphics*, 120–125, *Tokyo, Japan, 9-11 July*.

[6] Kawai N., Sato T., Yokoya N., (2016), "Diminished Reality Based on Image Inpainting Considering Background Geometry", *IEEE Transactions on Visualization and Computer Graphics, 22* (3), 1236–1247.

[7] Gao Y., Cheung G., Maugey T., Frossard P., Liang J., (2016), "Encoder-driven inpainting strategy in multiview video compression", *IEEE Transactions on Image Processing, 25* (1), 134–149.

[8] Barnes C., Shechtman E., Finkelstein A., Goldman D. B., (2009), "PatchMatch", *ACM SIGGRAPH, 28* (3), 1.

[9] Barnes C., Shechtman E., Goldman D. B., Finkelstein A., (2010), "The generalized PatchMatch correspondence algorithm", *European Conference on Computer Vision*, 29-43, Crete, Greece, 5-11 September.

[10] He K., Sun J., (2014), "Image completion approaches using the statistics of similar patches", *IEEE Transactions on Pattern Analysis and Machine Intelligence, 36* (12), 2423–2435.

[11] Pathak D., Krahenbuhl P., Donahue J., Darrell T., Efros A. A., (2016), "Context Encoders: Feature Learning by Inpainting", *International Conference on Computer Vision and Pattern Recognition*, 2536–2544, Las Vegas, NV, USA, 26 June-1 July.

[12] Iizuka S., Simo-Serra E., Ishikawa H., (2017), "Globally and locally consistent image completion", *ACM Transactions on Graphics, 36* (4), 1–14.

[13] Chen Y., Zhang H., Liu L., Chen X., Zhang Q., Yang K., Xia R., Xie J., (2021), "Research on image Inpainting algorithm of improved GAN based on two-discriminations networks", *Applied Intelligence*, *51* (6), 3460–3474.

[14] Goodfellow I. J., Pouget-Abadie J., Mirza M., Xu B., Warde-Farley D., Ozair S., Courville A., Bengio Y., (2014), "Generative adversarial nets", *International Conference on Neural Information Processing Systems*, 2672–2680, Montreal, Canada, 8-13 December.

[15] Liu G., Reda F. A., Shih K. J., Wang T. C., Tao A., Catanzaro B., (2018), "Image Inpainting for Irregular Holes Using Partial Convolutions", *European Conference on Computer Vision*, 85-100, Munich, Germany, 8-14 September.

[16] Ren Y., Yu X., Zhang R., Li T. H., Liu S., Li G., (2019), "Structureflow: Image inpainting via structure-aware appearance flow", *International Conference on Computer Vision*, 181–190, *Seoul, Korea, 27 October-2 November*.

[17] Nazeri K., Ng E., Joseph T., Qureshi F., Ebrahimi M., (2019), "Edgeconnect: Structure-guided image inpainting using edge prediction", *International Conference on Computer Vision Workshops*, 3265–3274, *Seoul, Korea, 27 October-2 November*.

[18] Li A., Qi J., Zhang R., Kotagiri R., (2019), "Boosted GAN with Semantically Interpretable Information for Image Inpainting", *International Joint Conference on Neural Networks, Budapest, Hungary*, 1–8, *14-19 July*.

[19] Yu J., Lin Z., Yang J., Shen X., Lu X., Huang T. S., (2018), "Generative Image Inpainting with Contextual Attention", *International Conference on Computer Vision and Pattern Recognition*, 5505–5514, Salt Lake City, Utah, USA, 19 - 21 June.

[20] Shao M., Zhang W., Zuo W., Meng D., (2020), "Multi-scale generative adversarial inpainting network based on cross-layer attention transfer mechanism", *Knowledge-Based Systems, 196*, 105778.

[21] Liu H., Jiang B., Xiao Y., Yang C., (2019), "Coherent semantic attention for image inpainting", *International Conference on Computer Vision*, 4169–4178, *Seoul, Korea, 27 October-2 November*.

[22] Karras T., Aila T., Laine S., Lehtinen J., (2018), "Progressive Growing of GANs for improved quality, stability, and variation", *International Conference on Learning Representations, 0–0, Vancouver, Canada, 30 Apr-3 May.*

[23] Goodfellow I., Bengio Y., Courville A., Bengio Y., (2016), *"Deep learning"*,

Issue 2, MIT Press Cambridge.

[24] Jia Deng, Wei Dong, Socher R., Li-Jia Li, Kai Li, Li Fei-Fei, (2009), "ImageNet: A large-scale hierarchical image database", *International Conference on Computer Vision and Pattern Recognition*, 248–255, Miami, FL, USA, 20-25 June.

[25] Tulyakov S., Liu M. Y., Yang X., Kautz J., (2018), "MoCoGAN: Decomposing Motion and Content for Video Generation", *International Conference on Computer Vision and Pattern Recognition*, 1526–1535, Salt Lake City, Utah, USA, 19-21 June.

[26] Isola P., Zhu J. Y., Zhou T., Efros A. A., (2017), "Image-to-image translation with conditional adversarial networks", *International Conference on Computer Vision and Pattern Recognition*, 5967–5976, *Honolulu, Hawaii, USA, 22-25 July*.

[27] Tu Y., Lin Y., Wang J., Kim J. U., (2018), "Semi-supervised learning with generative adversarial networks on digital signal modulation classification", *Computers, Materials and Continua, 55* (2), 243–254.

[28] Mao X., Li Q., Xie H., Lau R. Y. K., Wang Z., Smolley S. P., (2017), "Least Squares Generative Adversarial Networks", *International Conference on Computer Vision*, 2794–2802, *Venice, Italy, 22-29 October*.

[29] Kingma D. P., Welling M., (2014), "Auto-encoding variational Bayes", *International Conference on Learning Representations*, 2014.

[30] Arjovsky M., Chintala S., Bottou L., (2017), "Wasserstein GAN", *International Conference on Machine Learning*, 214-223, Sydney, Australia, 7-9 August.

[31] Bertalmio M., Sapiro G., Caselles V., Ballester C., (2000), "Image inpainting", *International Conference on Computer Graphics and Interactive Techniques, 417–424, New Orleans, Louisiana, USA, July*.

[32] Ballester C., Bertalmio M., Caselles V., Sapiro G., Verdera J., (2001), "Filling-in by joint interpolation of vector fields and gray levels", *IEEE Transactions on Image Processing, 10* (8), 1200–1211.

[33] Darabi S., Shechtman E., Barnes C., Dan B Goldman, Sen P., (2012), "Image melding: Combining inconsistent images using patch-based synthesis", *ACM Transactions on Graphics, 31* (4), 1–10.

[34] Zhu J. Y., Park T., Isola P., Efros A. A., (2017), "Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks", *International Conference on Computer Vision*, 2223-2232, *Venice, Italy, 22-29 October*.

[35] Huang, Gao and Liu, Zhuang and Van Der Maaten, Laurens and Weinberger K. Q., (2017), "Densely connected convolutional networks", *International Conference on Computer Vision and Pattern Recognition*, 4700-4708, *Honolulu, Hawaii, USA, 22-25 July*.

[36] Zeng Y., Fu J., Chao H., Guo B., (2019), "Learning pyramid-context encoder network for high-quality image inpainting", *International Conference on Computer Vision and Pattern Recognition*, 1486-1494, Long Beach, CA, USA, 16-20 June.

[37] Li J., Wang N., Zhang L., Du B., Tao D., (2020), "Recurrent feature reasoning for image inpainting", *International Conference on Computer Vision and Pattern Recognition*, 7760-7768, Virtual, 16-18 June.

[38] Li J., He F., Zhang L., Du B., Tao D., (2019), "Progressive reconstruction of visual structure for image inpainting", *International Conference on Computer Vision*, 5962-5971, *Seoul, Korea, 27 October-2 November*.

[39] Di Martino J. M., Facciolo G., Meinhardt-Llopis E., Pérez P., Gangnet M., Blake A., (2003), "Poisson Image Editing", *Image Processing On Line*, *5* (3), 313–318.

[40] Hu J., Shen L., Sun G., (2018), "Squeeze-and-excitation networks", *International Conference on Computer Vision and Pattern Recognition*, 7132-7141, Salt Lake City, Utah, USA, 19 - 21 June.

[41] Hong X., Xiong P., Ji R., Fan H., (2019), "Deep fusion network for image completion", *ACM International Conference on Multimedia*, 2033–2042, Nice, France, 21-25 October.

[42] Ronneberger O., Fischer P., Brox T., (2015), "U-net: Convolutional networks for biomedical image segmentation", *International Conference on Medical Image Computing and Computer-Assisted Intervention, 234-241, Munich, Germany, 5-9 October*.

[43] Bengio Y., Louradour J., Collobert R., Weston J., (2009), "Curriculum learning", *Annual International Conference on Machine Learning*, 41–48, Montreal, Quebec, Canada, 14-18 June.

[44] Zhang H., Hu Z., Luo C., Zuo W., Wang M., (2018), "Semantic image inpainting with progressive generative networks", *ACM international conference on Multimedia*, 1939–1947, New York, NY, USA, 22-26 October.

[45] Xiao J., Liao L., Liu Q., Hu R., (2019), "CISI-net: Explicit latent content inference and imitated style rendering for image inpainting", *Proceedings of the*

AAAI Conference on Artificial Intelligence, 33(01), 354-362.

[46] Xiong W., Yu J., Lin Z., Yang J., Lu X., Barnes C., Luo J., (2019), "Foreground-aware image inpainting", *International Conference on Computer Vision and Pattern Recognition*, 5840-5848, Long Beach, CA, USA, 16-20 June.

[47] Song Y., Yang C., Shen Y., Wang P., Huang Q., Jay Kuo C. C., (2019), "SPG-Net: Segmentation prediction and guidance network for image inpainting", *British Machine Vision Conference, 0-0, Newcastle, UK, 3-6 September*.

[48] Gulrajani I., Ahmed F., Arjovsky M., Dumoulin V., Courville A., (2017), "Improved training of Wasserstein gan", *Advances in Neural Information Processing Systems*, 5768–5778, *Long Beach, CA, USA, 4-9 December*.

[49] Web 1, (2018), *https://nealjean.com/ml/frechet-inception-distance/, (access time: 20/02/2020)*.

[50] Heusel M., Ramsauer H., Unterthiner T., Nessler B., Hochreiter S., (2017), "GANs trained by a two time-scale update rule converge to a local Nash equilibrium", *Advances in Neural Information Processing Systems*, 6627–6638, *Long Beach, CA, USA, 4-9 December*.

[51] Web 2, (2020), https://cedrickchee.gitbook.io/knowledge/courses/fast.ai/deep-learning-part-2-cutting-edge-deep-learning-for-coders/2018-edition/lesson-14-image-segmentation, *(access time: 20/02/2020)*.

[52] Lin T. Y., Maire M., Belongie S., Hays J., Perona P., Ramanan D., Dollár P., Zitnick C. L., (2014), "Microsoft COCO: Common objects in context", *European Conference on Computer Vision*, 740-755, Zurich, Switzerland, 6-12 September.

[53] Karras T., Laine S., Aila T., (2019), "A style-based generator architecture for generative adversarial networks", *International Conference on Computer Vision and Pattern Recognition*, 4401-4410, *Long Beach, CA, USA, 16-20 June*.

[54] Fang Y., Li Y., Tu X., Tan T., Wang X., (2020), "Face completion with Hybrid Dilated Convolution", *Signal Processing: Image Communication*, *80*, 115664.

[55] Hedjazi M. A., Genc Y., (2019), "Learning to inpaint by progressively growing the mask regions", *International Conference on Computer Vision Workshops*, 0–0, *Seoul, Korea, 27 October-2 November*.

[56] Kwon Y., Kim S., Yoo D., Yoon S. E., (2019), "Coarse-to-fine clothing image generation with progressively constructed conditional GaN", *International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, 83–90, Prague, Czech Republic, 25–27 February.

[57] Lee C. H., Liu Z., Wu L., Luo P., (2020), "MaskGAN: Towards Diverse and Interactive Facial Image Manipulation", *International Conference on Computer Vision and Pattern Recognition*, 5549-5558, Virtual, 16-18 June.

[58] Web 3, (2018), https://github.com/karfly/qd-imd, *(access time: 29/10/2020)*.

[59] Kingma D. P., Ba J. L., (2015), "Adam: A method for stochastic optimization", *International Conference for Learning Representations, 0-0, Banff, Canada, 14-16 April.*

[60] Mittal A., Soundararajan R., Bovik A. C., (2013), "Making a "completely blind" image quality analyzer", *IEEE Signal Processing Letters*, *20* (3), 209–212.

[61] Wu Q., Li H., Meng F., Ngan K. N., Luo B., Huang C., Zeng B., (2016), "Blind image quality assessment based on multichannel feature fusion and label transfer", *IEEE Transactions on Circuits and Systems for Video Technology*, *26* (3), 425–440.

[62] Sun T., Zhu X., Pan J. S., Wen J., Meng F., (2015), "No-reference image quality assessment in spatial domain", *Advances in Intelligent Systems and Computing*, *329* (12), 381–388.

[63] Chen L., Wang Y. H., Wang Y. D., Huang D., (2009), "Face recognition with statistical local binary patterns", *International Conference on Machine Learning and Cybernetics*, 2433–2439, Baoding, China, 12-15 July.

[64] Ahonen T., Hadid A., Pietikäinen M., (2006), "Face description with local binary patterns: Application to face recognition", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *28* (12), 2037–2041.

[65] Odena A., Olah C., Shlens J., (2017), "Conditional image synthesis with auxiliary classifier gans", *International Conference on Machine Learning*, 2642-2651, Sydney, Australia, 6-11 August.

[66] Johnson J., Alahi A., Fei-Fei L., (2016), "Perceptual losses for real-time style transfer and super-resolution", *European Conference on Computer Vision*, 694–711, Amsterdam, Netherlands, 11–14 October.

[67] Zheng Z., Yang X., Yu Z., Zheng L., Yang Y., Kautz J., (2019), "Joint discriminative and generative learning for person re-identification", *International Conference on Computer Vision and Pattern Recognition*, 2138-2147, *Long Beach, CA, USA, 16-20 June*.

[68] Canny J., (1986), "A Computational Approach to Edge Detection", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *PAMI-8* (6), 679–

698.

[69] Xu D., Wang Y., Xu S., Zhu K., Zhang N., Zhang X., (2020), "Infrared and visible image fusion with a generative adversarial network and a residual network", *Applied Sciences (Switzerland), 10* (2), 103144.

[70] Juefei-Xu F., Savvides M., (2014), "Weight-Optimal Local Binary Patterns", *European Conference on Computer Vision Workshops, 148-159, Zurich, Switzerland, 6-12 September*.

[71] Web 4, (2016), https://devaginiblog.wordpress.com/2016/06/03/local-binary-patterns-lbp/, *(access time 10/03/2020)*.

[72] Zhou B., Lapedriza A., Khosla A., Oliva A., Torralba A., (2018), "Places: A 10 Million Image Database for Scene Recognition", *IEEE Transactions on Pattern Analysis and Machine Intelligence, 40* (6), 1452–1464.

[73] Liu Z., Luo P., Wang X., Tang X., (2018), "Deep Learning Face Attributes in the Wild", *International Conference on Computer Vision*, 3730-3738, Santiago, Chile, 13-16 December.

[74] Paszke A., Gross S., Chintala S., Chanan G., Yang E., (2017), *Automatic differentiation in PyTorch, Advances in Neural Information Processing Systems, 0-0, Long Beach, CA, USA, 4-9 December*.

[75] Miyato T., Kataoka T., Koyama M., Yoshida Y., (2018), "Spectral normalization for generative adversarial networks", *ArXiv Preprint ArXiv: 1802.05957*.

[76] Huang Z., Zheng Z., Yan C., Xie H., Sun Y., Wang J., Zhang J., (2020), "Real-world automatic makeup via identity preservation makeup net", *International Joint Conference on Artificial Intelligence*, 652–658, *Yokohama, Japan, 7-15 January*.

[77] Yang Z., Dong J., Liu P., Yang Y., Yan S., (2019), "Very long natural scenery image prediction by outpainting", *International Conference on Computer Vision Workshops*, 10561-10570, *Seoul, Korea, 27 October-2 November*.

[78] Radford A., Metz L., Chintala S., (2016), "Unsupervised representation learning with deep convolutional generative adversarial networks", *International Conference on Learning Representations, 97-108, Shanghai, China, 13-15 September*.

[79] Zhao J., Mathieu M., LeCun Y., (2016), "Energy-based generative adversarial network", *ArXiv Preprint ArXiv:1609.03126*.

[80] Kocmi T., Bojar O., (2017), "Curriculum Learning and Minibatch Bucketing in

Neural Machine Translation", *ArXiv Preprint ArXiv:*1707.09533.

[81] Platanios E. A., Stretcu O., Neubig G., Poczos B., Mitchell T. M., (2019), "Competence-based curriculum learning for neural machine translation", *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 1162–1172, Minneapolis, MN, USA, 2-7 June.

[82] Sarafianos N., Giannakopoulos T., Nikou C., Kakadiaris I. A., (2017), "Curriculum learning for multi-task classification of visual attributes", *International Conference on Computer Vision Workshops*, 2608-2615, *Venice, Italy, 22-29 October*.

[83] He K., Zhang X., Ren S., Sun J., (2016), "Deep residual learning for image recognition", *International Conference on Computer Vision and Pattern Recognition*, 770-778, Las Vegas, NV, USA, 26 June-1 July.

[84] Weinshall D., Cohen G., Amir D., (2018), "Curriculum learning by transfer learning: Theory and experiments with deep networks", *International Conference on Machine Learning*, 5238-5246, Stockholm, Sweden, 18-24 July.

[85] Sharma R., Barratt S., Ermon S., Pande V., (2018), "Improved Training with Curriculum GANs", *ArXiv Preprint ArXiv:1807.09295*.

[86] Yeh R. A., Chen C., Yian Lim T., Schwing A. G., Hasegawa-Johnson M., Do M. N., (2017), "Semantic image inpainting with deep generative models", *International Conference on Computer Vision and Pattern Recognition*, 5485-5493, *Honolulu, Hawaii, USA, 22-25 July*.

[87] Ioffe S., Szegedy C., (2015), "Batch normalization: Accelerating deep network training by reducing internal covariate shift", *International Conference on Machine Learning*, 448-456, Lille, France, 6-11 July.

[88] Ulyanov D., Vedaldi A., Lempitsky V., (2016), "Instance Normalization: The Missing Ingredient for Fast Stylization", *ArXiv Preprint ArXiv:1607.08022*.

[89] Zhang J., Tang S., Zhang X., Li Y., Zhang R., (2020), "Ahff-Net: Adaptive Hierarchical Feature Fusion Network for Image Inpainting", *International Conference on Image Processing*, 478–482, UAE, 25-28 October.

[90] Jolicoeur-Martineau A., (2019), "The relativistic discriminator: A key element missing from standard GaN", *arXiv preprint ArXiv:1807.00734*.

BIOGRAPHY

Mohamed Abbas Hedjazi received his B.Sc. and M.s degrees in Computer Engineering from the University of Batna in 2013 and 2015, respectively. He is currently pursuing his Ph.D. degree in Gebze Technical University, Institute of Natural and Applied Sciences, the program of Computer Engineering. His research interests are in the areas of image processing, computer vision, and deep learning.

APPENDICES

Appendix A: publications within the scope of the thesis study

- Learning to Inpaint by Progressively Growing the Mask Regions, in The IEEE International Conference on Computer Vision Preregistration workshop (ICCV 2019).
- Image Inpainting using Scene Constraints, in the Journal of Signal Processing Image Communication (2020).
- Efficient Texture-aware Multi-GAN for Image Inpainting, in the Journal of Knowledge-based Systems (2021).
- Image Inpainting using Deep Multi-resolution Paths and Adaptative Loss Functions, Accepted in The IEEE International Conference on Image Processing (ICIP 2021).

Appendix B: additional results of SC

We show additional results of the SC model on the CelebHQ dataset (Figure 8.2). We show the corrupted, the estimated and the ground truth segmentation labels/edges, the output and the ground truth images.

Appendix C: architecture and additional results of ETMG

The architecture of discriminators: Table 8.1 shows the architecture of the PatchGAN discriminator [26] where: $n = 24$ for the 32×32 and the 64×64 discriminators, and $n = 48$ for the 128×128 and the 256×256 discriminators. We use a slope of 0.2 in LeakyReLU. We use Spectral Normalization [75] in the convolution layers where: *bias=False*. We initialize the weights using a Gaussian distribution with $gain = 0.02$.

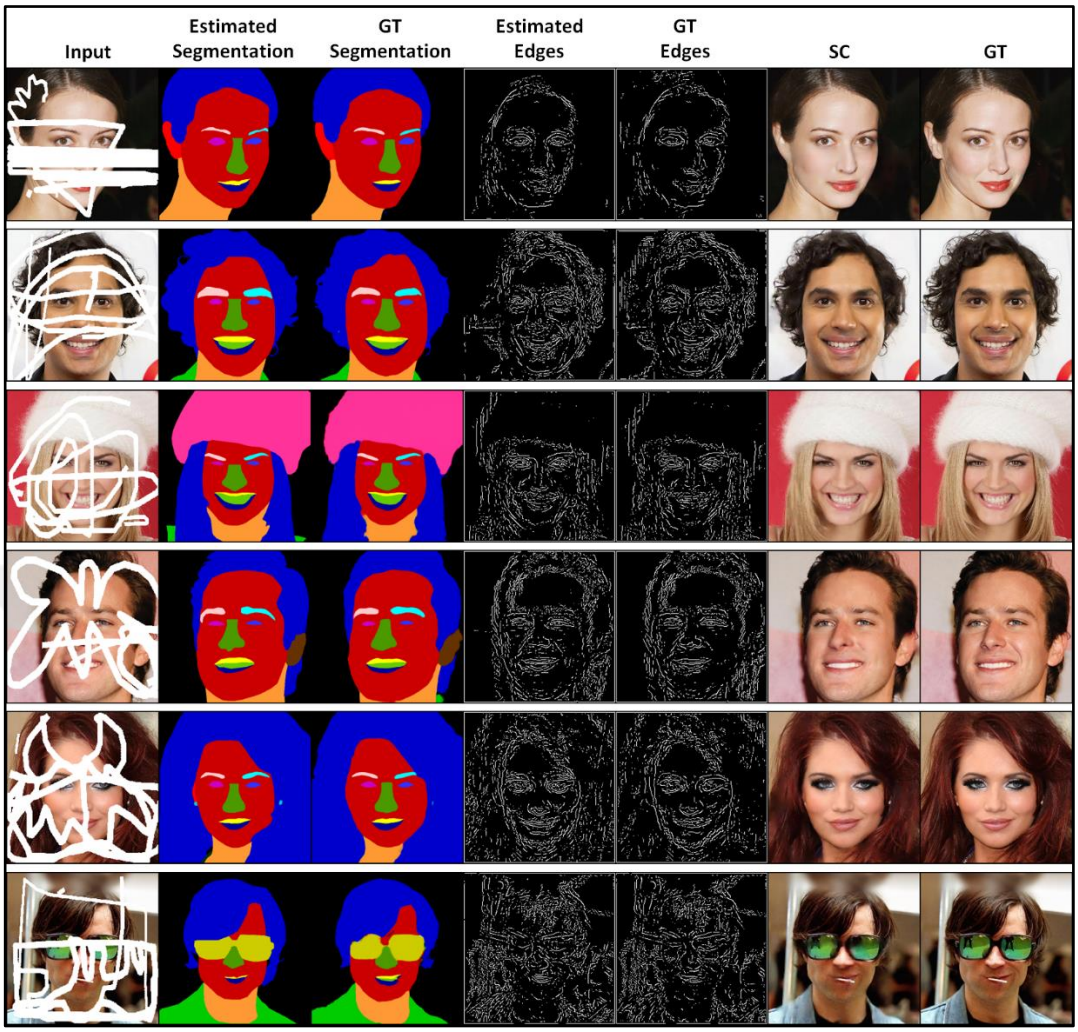

Figure 8.2: Additional results of the SC model.

The architecture of generators: for all the generators defined in Table 8.2, Table 8.3, Table 8.4 and Table 8.5 we use the same weight initialization method used in the discriminator. *TConv2D* refers to the *ConvTranspose2d* layer in Pytorch [74]. The Gray function is defined as follows: $Gray(r, g, b) = 0.299 \times r + 0.587 \times g + 0.110 \times b$ where r, g, and b are the red, green and blue colors, respectively.

Table 8.1: Architecture of the ETMG discriminator network.

Layer	Dim	Kernel	Stride	Padding	Activation
Conv2D	n	4×4	2	1	LeakyReLU
Conv2D	$n \times 2$	4×4	2	1	LeakyReLU
Conv2D	$n \times 4$	4×4	2	1	LeakyReLU
Conv2D	1	4×4	1	1	LeakyReLU

Table 8.2: Architecture of the 32×32 ETMG generator network.

Layer	Dim	Kernel	Stride	Padding	Activation
Conv2D	24	3×3	1	1	ReLU
Conv2D	48	4×4	2	1	ReLU
Conv2D	48	4×4	2	1	ReLU
Conv2D	96	3×3	1	1	ReLU
Conv2D	96	3×3	1	1	ReLU
Conv2D	96	3×3	1	1	ReLU
Conv2D	96	3×3	1	1	ReLU
Conv2D	96	3×3	1	1	ReLU
TConvt2D	48	4×4	2	1	ReLU
TConv2D	24	4×4	2	1	ReLU
Conv2D	3	3×3	1	1	Tanh

Table 8.3: Architecture of the 64×64 ETMG generator network.

Block	Layer	Dim	Kernel	Stride	Padding	Activation
1	Conv2D	24	3×3	1	1	ReLU
	Conv2D	48	4×4	2	1	ReLU
	Conv2D	48	4×4	2	1	ReLU
2	Conv2D	24	3×3	1	1	ReLU
	Conv2D	48	4×4	2	1	ReLU
	Conv2D	48	3×3	1	1	ReLU
3	Conv2D	96	3×3	1	1	ReLU
	Conv2D	96	3×3	1	1	ReLU
	Conv2D	96	3×3	1	1	ReLU
	Conv2D	96	3×3	1	1	ReLU
	Conv2D	96	3×3	1	1	ReLU
	TConvt2D	48	4×4	2	1	ReLU
	TConv2D	24	4×4	2	1	ReLU
	Conv2D	3	3×3	1	1	Tanh

Table 8.4: Architecture of the 128×128 ETMG generator network.

Block	Layer	Dim	Kernel	Stride	Padding	Activation
1	Conv2D	24	3×3	1	1	ReLU
	Conv2D	48	4×4	2	1	ReLU
	Conv2D	48	4×4	2	1	ReLU
2	Conv2D	24	3×3	1	1	ReLU
	Conv2D	48	4×4	2	1	ReLU
	Conv2D	48	3×3	1	1	ReLU
3	Conv2D	24	3×3	1	1	ReLU
	Conv2D	48	3×3	2	1	ReLU
	Conv2D	48	3×3	1	1	ReLU
4	Conv2D	96	3×3	1	1	ReLU
	Conv2D	96	3×3	1	1	ReLU
	Conv2D	96	3×3	1	1	ReLU
	Conv2D	96	3×3	1	1	ReLU
	Conv2D	96	3×3	1	1	ReLU
	TConvt2D	48	4×4	2	1	ReLU
	TConv2D	24	4×4	2	1	ReLU
	Conv2D	3	3×3	1	1	Tanh

Table 8.5: Architecture of the 256×256 ETMG generator network.

Block	Layer	Dim	Kernel	Stride	Padding	Activation
1	Conv2D	24	3×3	1	1	ReLU
	Conv2D	48	4×4	2	1	ReLU
	Conv2D	48	4×4	2	1	ReLU
2	Conv2D	24	3×3	1	1	ReLU
	Conv2D	48	4×4	2	1	ReLU
	Conv2D	48	3×3	1	1	ReLU
3	Conv2D	24	3×3	1	1	ReLU
	Conv2D	48	3×3	2	1	ReLU
	Conv2D	48	3×3	1	1	ReLU
4	Conv2D	24	3×3	1	1	ReLU
	Conv2D	48	3×3	2	1	ReLU
	Conv2D	48	3×3	1	1	ReLU
5	Conv2D	96	3×3	1	1	ReLU
	Conv2D	96	3×3	1	1	ReLU
	Conv2D	96	3×3	1	1	ReLU
	Conv2D	96	3×3	1	1	ReLU
	Conv2D	96	3×3	1	1	ReLU
	TConvt2D	48	4×4	2	1	ReLU
	TConv2D	24	4×4	2	1	ReLU
	Conv2D	3	3×3	1	1	Tanh

Learning curves: we show the training curves of our four generators and discriminators. The loss curves show a stable training that reflects the visual quality of the generated images. Figure 8.3 shows the loss values of the generators and the discriminators. In Figure 8.4 we show the reconstruction loss values. During training, we use masks that cover $30 - 40\%$ of the image. The successful exploitation of previously inpainted low-resolution images leads to fast convergence.

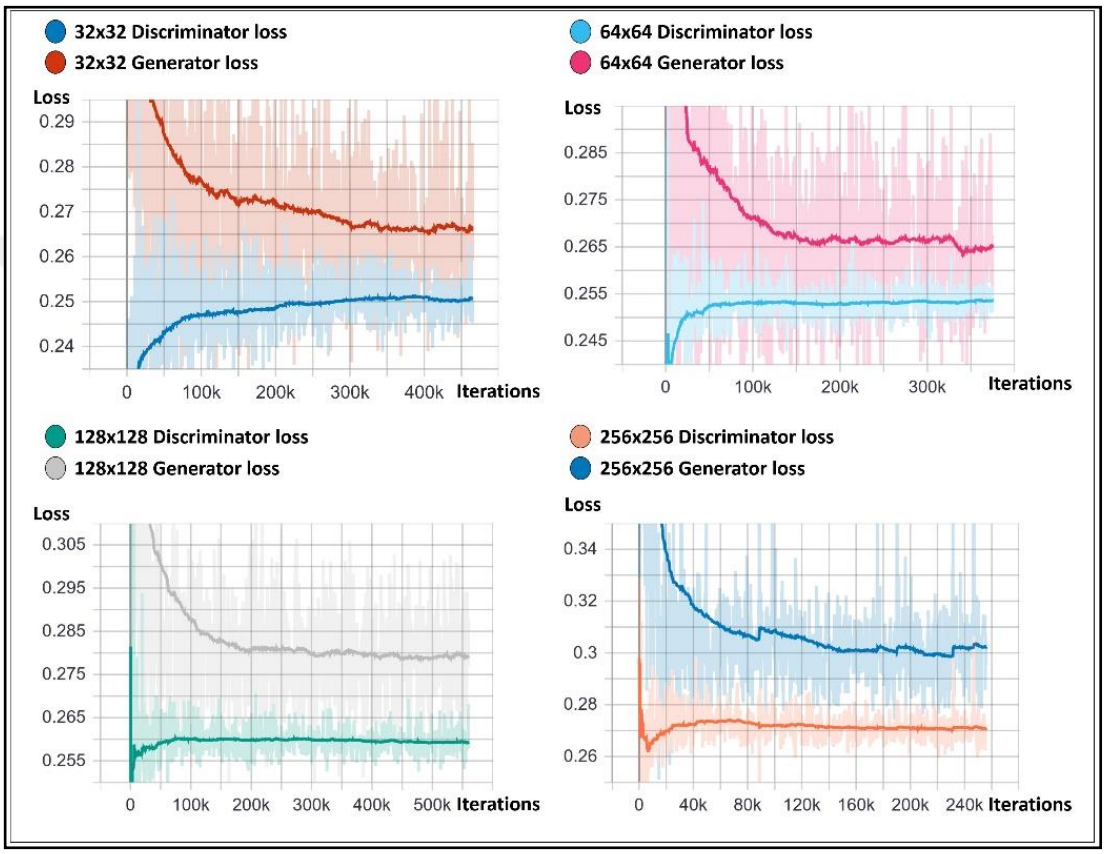

Figure 8.3: The GAN losses of the ETMG generators and discriminators.

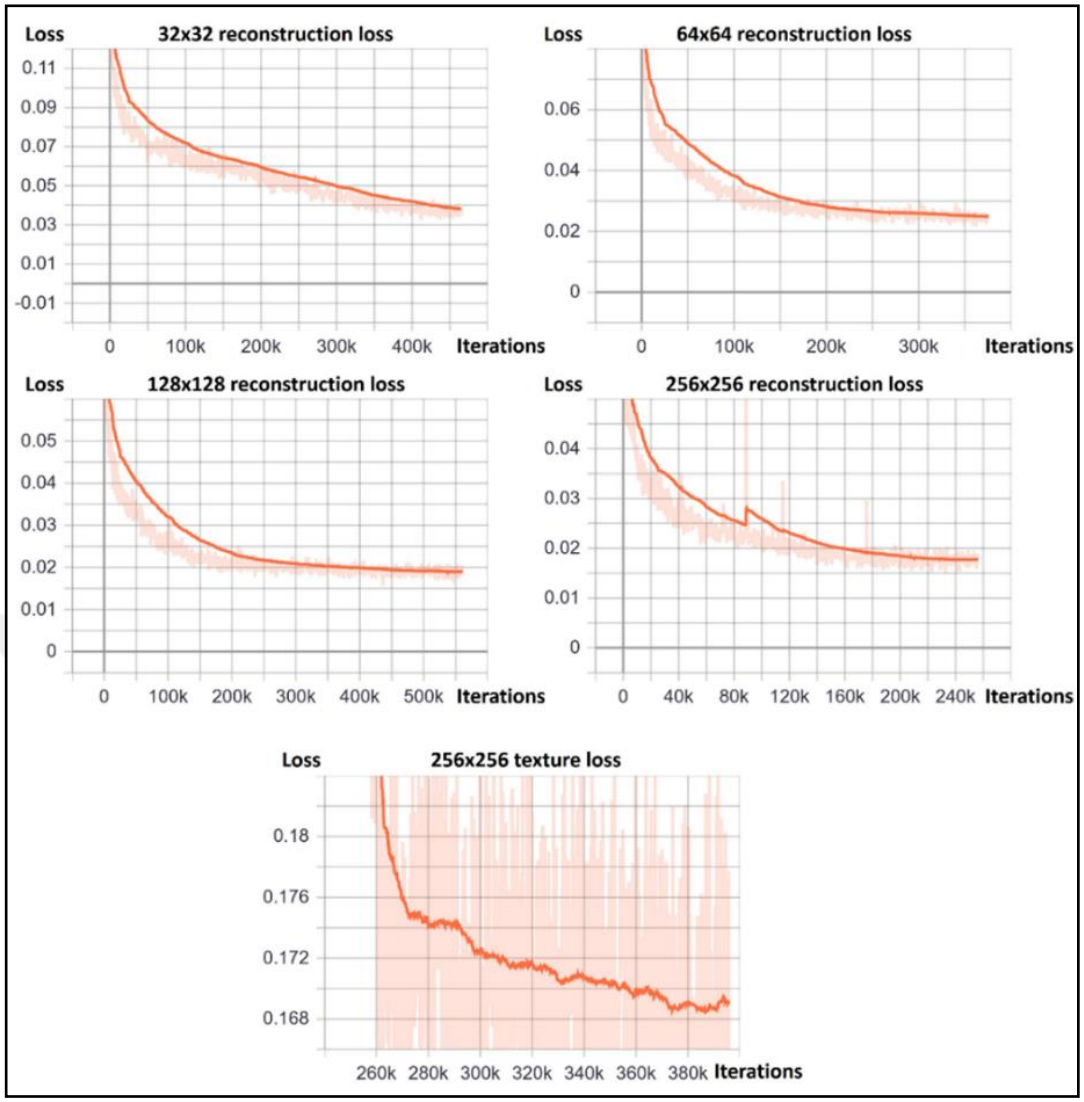

Figure 8.4: Reconstruction and LBP loss of the ETMG model.

Additional results: we show additional results of the ETMG model on Places2 (Figure 8.5) and CelebHQ (Figure 8.6). We show the corrupted, the output, the ground truth images. Also, we show the output and the ground truth LBP images.

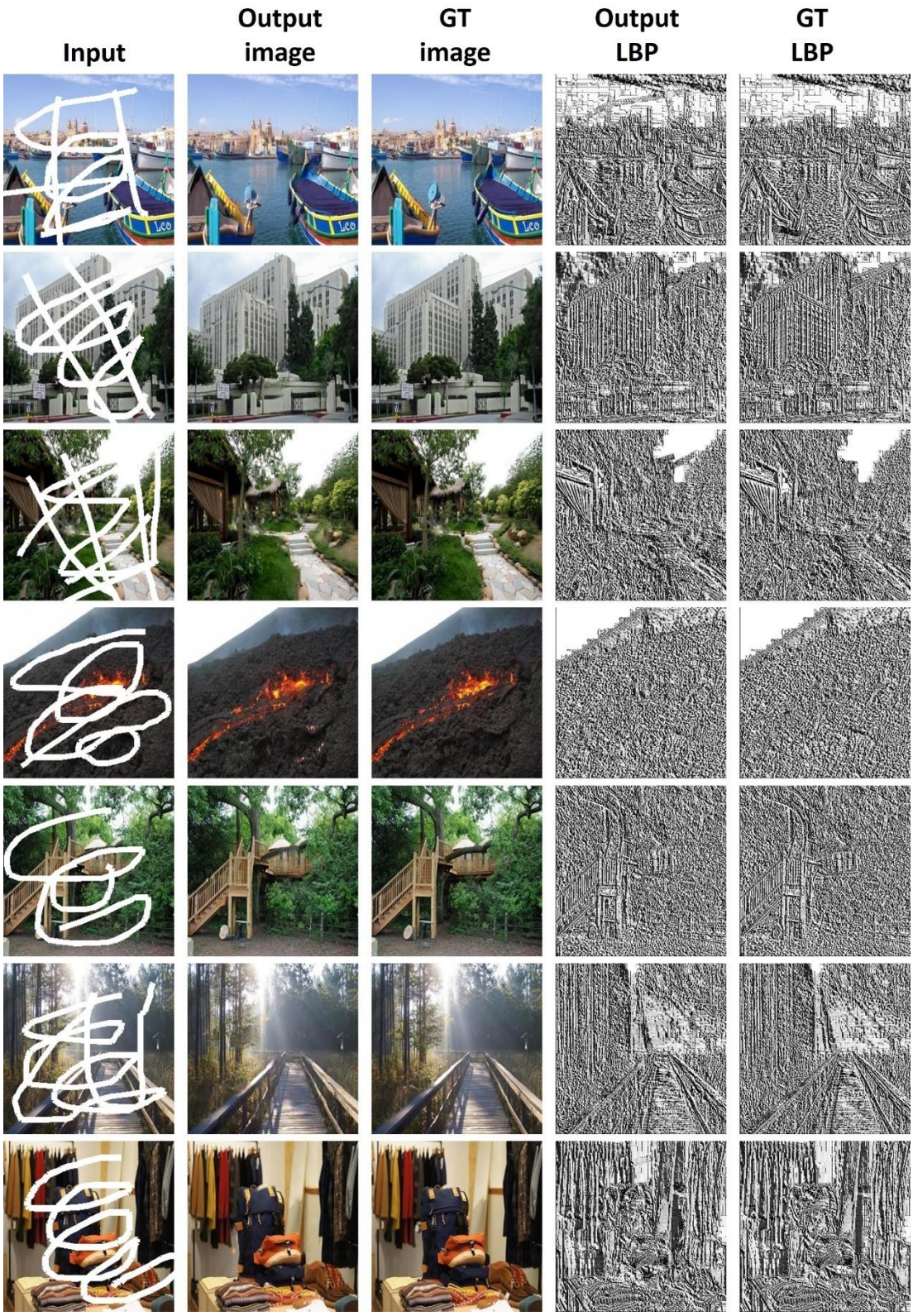

Figure 8.5: Additional qualitative results of the ETMG model on the Places2.

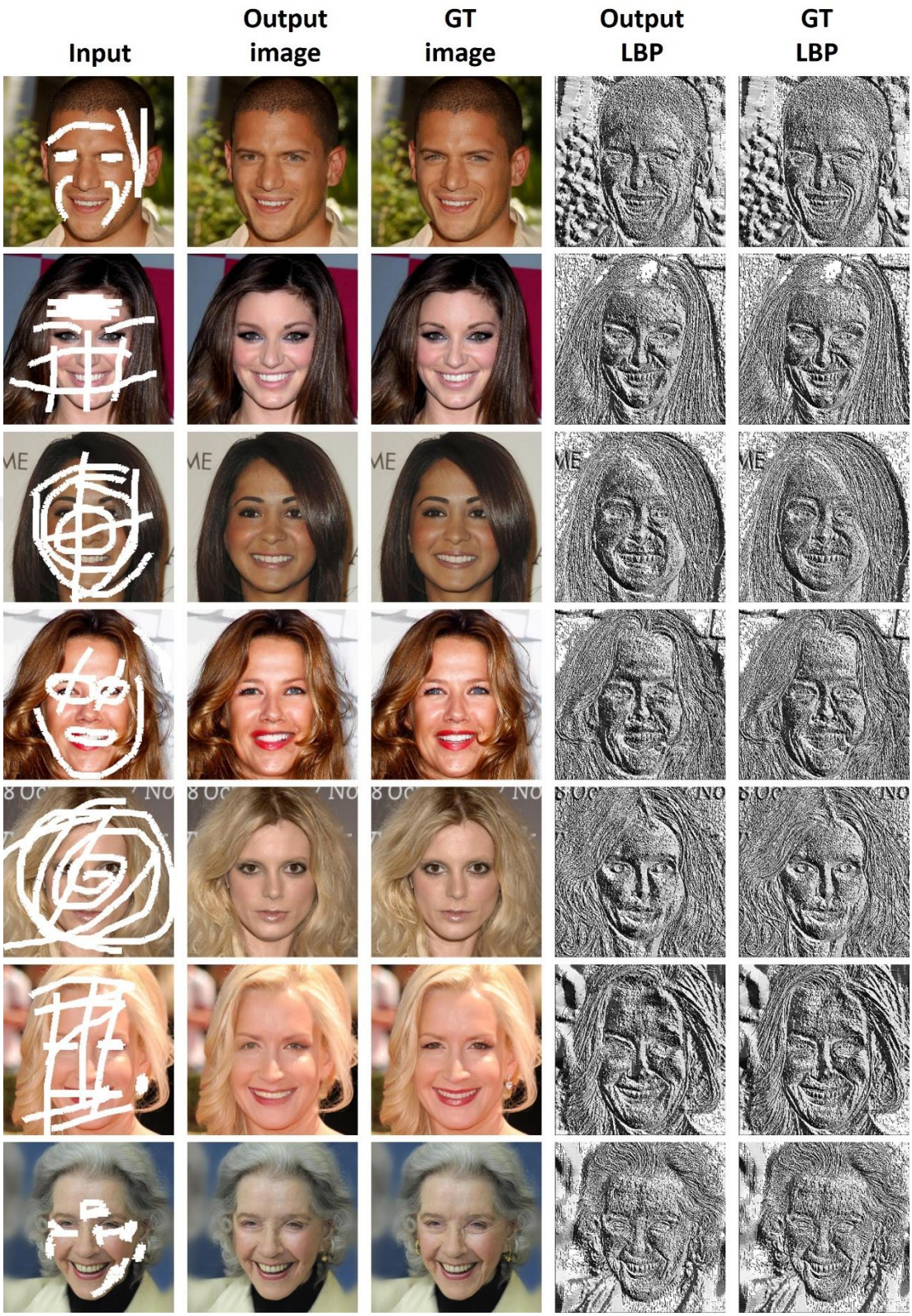

Figure 8.6: Additional qualitative results of the ETMG model on the CelebHQ.

Appendix D: additional results of ETMG + SC

We show additional results of the ETMG + SC model on CelebHQ (Figure 8.7). We show the corrupted, the estimated and the ground truth segmentation labels, the output and the ground truth images.

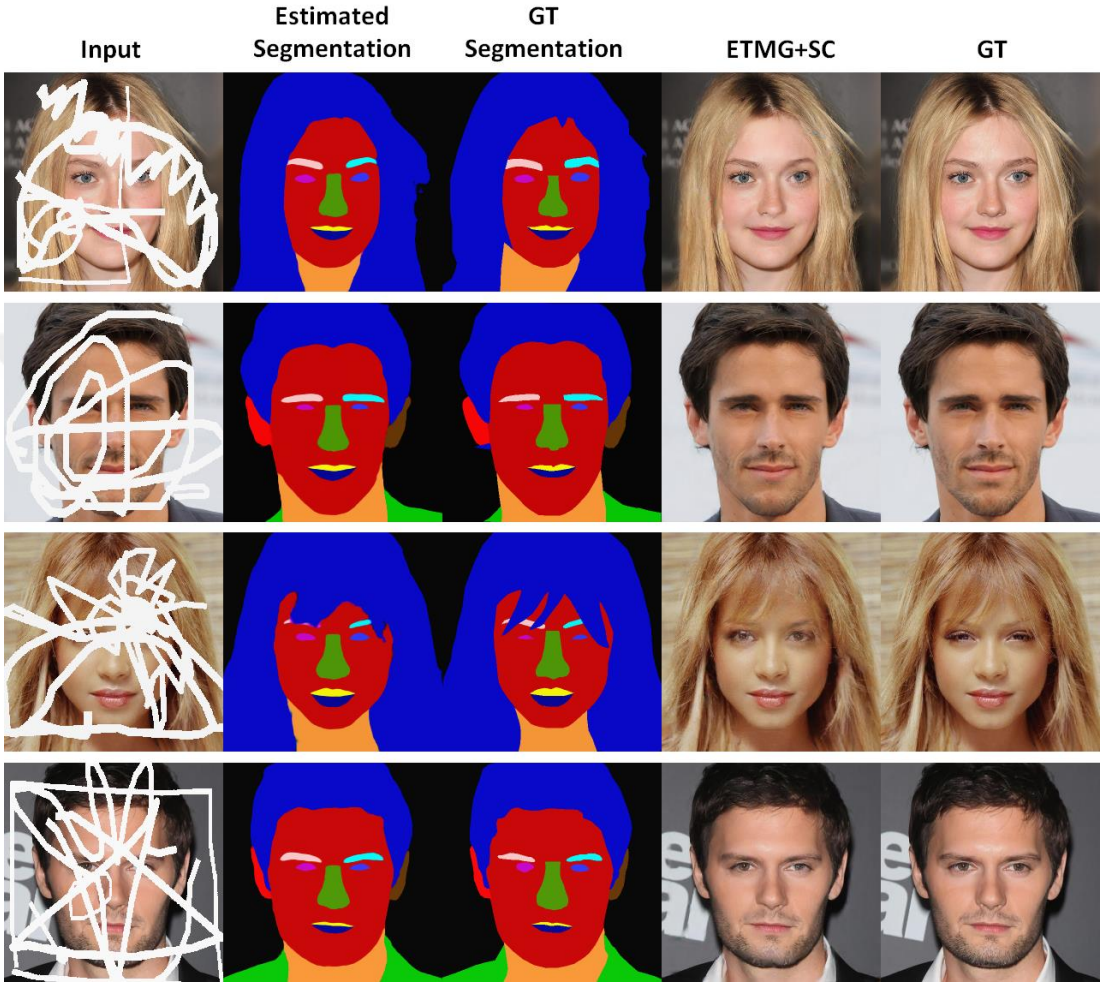

Figure 8.7: Additional results of the ETMG + SC model.