Towards Theoretical and Practical Image Inpainting with Deep Neural Networks

Thesis submitted in accordance with the requirements of the University of Liverpool for the degree of Doctor in Philosophy by

Shuyi Qu

Department of Electrical Engineering and Electronics
School of Electrical Engineering and Electronics and Computer Science
University of Liverpool

1st, July, 2022

Abstract

Image inpainting aims to restore deteriorated images with plausible and relevant contents, which has wide applications in many computer vision tasks. Conventional methods based on pixels propagation or patch replacement often fail in complicated scenarios with ambiguous artifacts or blurry fillings. In recent years, deep generative models constructed with autoencoder and generative adversarial networks have been exploited extensively and shown impressive results. Deep generative models are more advanced in understanding image global information through large-scale data than traditional models. However, those deep leaning-based approaches still have challenges in terms of bad generation quality under complex scenarios (i.e., artifacts and blur). Besides, they are unable to fit into real-world applications efficiently. This dissertation summarizes the proposed work on deep learning-based image inpainting to address these challenges theoretically and practically. Theoretically, this thesis analyzes and proves that multi-scale architecture can isolate the learning of structures and textures, thus generating high-quality inpainting results. The proposed model includes a pyramid of generators, each responsible for progressively capturing the image global structure and local details. Meanwhile, this thesis has studied using the feature fusion technique to address the insufficient utilization of current encoder-decoder features and further alleviate the low-quality synthesized patches. From a practical perspective, this thesis is committed to building user-oriented interactive inpainting systems for real-life uses. To this end, an efficient object removal pipeline integrated with object segmentation technique and an attention-based structural guided module is proposed. Overall, this thesis shows theoretically and practically deep generative model is a powerful tool to address these challenges. Nonetheless, the last part presents the novel contributions of this thesis and the outlook for future work.

Key Words: Image inpainting, Generative adversarial networks, Object detection, Feature fusion.

Contents

List of Figures

List of Tables

Acknowledgment

I would like to take this opportunity to thank my supervisors, colleagues, and family. My research is not possible without any help from all of them. Initially, I sincerely appreciated my primary supervisor, Professor Kaizhu Huang, for the constant support and advice throughout my research career. I learned much about research from his guidance in the past several years. I always get instant and valuable feedback from Prof. Huang for any problem I ask him. His innovative and brilliant thinking has inspired me in my research work. I would also thank my co-supervisors Dr. Qiufeng Wang and Dr. John Y. Goulermas for their earnest advice on my research.

Besides my supervisors, I would also thank all my colleagues in Pattern REcognition & Machine Intelligence Laboratory who gave me advice during my research and thesis writing. With their technical and psychological help, I have overcome many difficulties in academics and life. Meanwhile, I would thank Xi'an Jiaotong-liverpool University for providing my Ph.D. scholarship.

I would also thank my industrial advisor, Professor Zhenxing Niu, for being very helpful and supportive. He gave me a good opportunity for an internship at Alibaba DAMO Academy, and I learned a lot from his advice.

Last but not least, my parents have encouraged me and provided me with love when I encounter problems. Their endless help is my motivation to move forward.

I want to thank all the people mentioned above again for their dedicated help, and without them, none of this would have been possible.

Chapter 1

Introduction

The term image inpainting is coined initially from the ancient art world. It is a technique performed by artists to physically restore the damaged paintings and photographs with minor defects, such as scratches, dust, and cracks [1]. In the late 20th, the revolution of computers has driven the large-scale adoption of digital images [2]. Meanwhile, the prevalence of multimedia has shaped people's lives, which significantly pushes forward the development of image manipulation techniques. As one of the fundamental tasks of computer vision, the modern image inpainting technique refers to a model that synthesis pixels of the corrupt area in an incomplete image, with the capacity to restore the image with plausible textures which visually align with the rest of the pixels. These models accept inputs with unknown regions (or mask regions) surrounded by the known regions. The reconstruction is to fill in the mask regions with knowledge of the known area. Image inpainting allows users to clear up distracting objects or retouch undesired areas in images. It has essential functions in many computer vision tasks, including object removal, image manipulation, stitching, rendering, re-targeting, re-composition.

This dissertation identifies and explores the following challenges in current image inpainting research: 1) Image structure and texture are not well-modeled in a single encoder-decoder architecture; 2) Fully utilization of the extracted features is not prioritized by current methods; 3) To build an efficient user interactive image inpainting system. In order to tackle these challenges, this thesis proposes novel models and frameworks and have validated their effectiveness through extensive experimental results. Before introducing the works in detail, this chapter first presents the broad history and concept of image inpainting, associating with the emerging methods and the existing challenges.

1.1 Image Inpainting and Existing Challenges

Historically, image inpainting is a technique performed by ancient artists to restore the damaged artwork. They conduct manual restoration on physical artwork such as oil or

acrylic paintings, typically in chemical or physical measures. Fig. 1.1 has shown a manually performed inpainting. This restoration needs a trained conservator with a mature knowledge of art paintings and potential risks of treatments. Since the late 1990s, image inpainting has steeped into the digital media era. Until now, if there is no advance notice, image inpainting indicates the inpainting of the digital format of images. In computer vision and graphics, image inpainting, as a part of image editing techniques, is the task of employing the neighboring pixels to restore a deteriorated image or borrowing patches of an external reference without noticeable discrepancy to the rest of the regions. The damaged area is generally covered by a "mask" as "unknown" region, and the surrounding areas are called "known" region. Thus the inpainting task can be regarded as using the known information to solve the unknown.

Fig. 1.1: The ancient image inpainting on artwork, source from [3].

Modern image inpainting methods can be divided into traditional and deep learning-based methods. Here, I introduce a different perspective that presents the evolution of the image and mask for these inpainting methods. This perspective also reflects the progress made by modern inpainting approaches. As shown in Fig. 1.2, the radar plot shows several representative methods and their scores across different dimensions, including the image texture complexity, image resolution size, mask type (square, free-form or mixed type), and mask-to-image ratio. The traditional representative PatchMatch [4] has the most limited scores on this chart. It is an exemplar-based method that is searching for the best candidate from the known area to fill the hole, and this reference-based searching method is restricted to stationary inpainting with small masks. Another group of traditional methods is based on progressively propagating pixels from the boundary to the hole, and these approaches suffer the same limitation as PatchMatch. After the era of deep learning, deep generative models are widely adopted for image inpainting tasks. The first work using generative adversarial networks on image inpainting is Context encoder [5]. As seen in the chart, this approach could handle more complex images because the adoption of deep neural networks could bring more global context. However, Context encoder only works on low-resolution images with small-size masks, and it struggles with complicated mask

types, e.g. arbitrary masks. A more recent work, Gated [6], proposed a coarse-to-fine architecture with gated convolution. Their two-stage model has improved the inpainting quality, and the gated convolution is specially designed for free-form masks. In the radar plot, the Gated model is evaluated with higher complicity of image structure mask type and mask-to-image ratio compared with Context encoder. Meanwhile, high-resolution image inpainting remains a challenging topic, and several works are dedicated to this. One representative HiFill [7] has employed the image residual technique plus two-stage deep generative models, and it could tackle high-resolution image restoration up to 4k images. In summary, the development of inpainting methods is driven by the complexity of these standards.

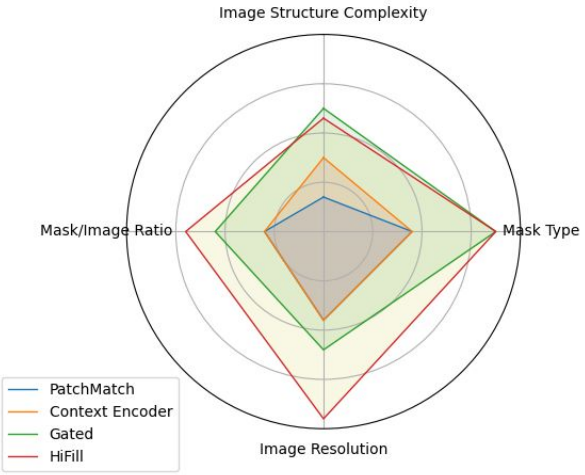


Fig. 1.2: Radar plot on image inpainting methods evaluation. There are four dimensions considered, including image structure complexity, image resolution (low to high resolution image), mask-to-image ratio (large or small mask), and mask type (square to free-form mask). Here compares four representatives including PatchMatch [4], Context encoder [5], Gated [6], and HiFill [7]. See more details from the chart.

Current deep learning-based models are often built with an autoencoder to extract image features and generate more realistic images with the help of generative adversarial networks. One of the most critical points of the image inpainting model is how to learn from the known regions. Concretely, unlike other generative tasks, where all the image information can be used to learn, image inpainting models should highlight the power of extracting image global context from the "unreasonable" incomplete input and learn the local details to predict the appropriate filling. In deep convolutional networks, the global context affects the structure of images, and missing the global information would lead to semantically meaningless patches or so-called "artifacts". While the local information affects the generated textures, the blurry patches are typically caused by inadequate

(a) Masked image (b) Current result w/ "artifact" (c) Current result w/ "blur" (d) The method in Section 3

Fig. 1.3: A failure example of current method. The masked area are colored with magenta. The current results with failed fillings are shown in (b) and (c) with artifacts and blur patches, respectively. Best view by zooming-in.

learning of local details. As shown in Fig. 1.3 is a failure case of current model. Thus, restoring an incomplete image with reasonable structure and texture under various types of masks becomes the most challenging topic in image inpainting. I break this down into the following two points: 1) the challenge of modeling the image structure and texture better instead of within a unified encoder-decoder model; 2) how to strengthen the current model, which can leverage the features sufficiently.

For the first point, as the literature study in Chapter 2, the chapter categorized a group of deep learning-based methods as progressive inpainting models. These models also focus on addressing the same challenge and adopting different architectures to better model the structure and texture. Different from the two-stage methods [6, 8], feature pyramid [7, 9], models with prior knowledge [10, 11], or models that integrate other techniques, this thesis introduces a multi-scale image pyramid to isolate the learning of structure and texture into different layers. The multi-scale architecture has been employed frequently in computer vision tasks since it could capture a different level of image statistics. For image inpainting, it can be found that the low-resolution layer focuses more on constructing low-frequency information while the high-resolution is dedicated to the high-frequency parts. After the experimental validation, I found it helpful to isolate the generation into different scales and combine them finally to obtain the results rich in structure and texture. Thus, I build a multi-scale pyramid generator with an edge-preserved image on the bottom layer to better model the image structure. Meanwhile, the pyramid uses an identical receptive field across all layers. Thus the low-resolution layer could learn more global context while learning more local details from the high-resolution layer.

Following the second point, this thesis focuses on building on top of the existing backbone with an advanced technique to enhance current features. The literature that also lies in this group adopted pyramid dilation architecture [12], new structure & texture branches, or fast Fourier convolution [13]. These methods try to build an advanced module to obtain enhanced features with better structure and texture modeling. The proposed method got

inspiration from the feature fusion perspective. First, I use the attention feature fusion module to aggregate low-level encoder features with high-level decoder features. This aggregation has compensated for the missing texture to the high-level semantics since this low-level information is weakened as the network goes deep. Unlike commonly-used skip connections, which concatenate two same-size features directly, the proposed method sends the output of attention feature fusion with the encoder features before adopting the vanilla skip connection. This strategy could alleviate the conflict between two input features due to inconsistent semantics. Second, the dilation architecture is essential for the image inpainting model to capture more global context and multi-scale information. The method enhances these features by building a dilation residual structure. This is done by applying the attention feature fusion as a residual fusion operation to enhance the features with different receptive field sizes. These two techniques could help to extract enhanced features and thus increase the efficiency of feature utilization.

As an essential technique widely applied in many downstream computer vision tasks, the interactive ability of image inpainting can not be overlooked. Current methods focus on building models with user inputs (e.g. edges or sketches) as auxiliary information for guided inpainting or creative editing [6, 14]. Otherwise, this space has not received noteworthy awareness. This thesis focuses on the efficiency of a user-oriented system. First, I build an efficient object removal system with instance segmentation integrated. The system could benefit from the segmentation algorithms as pre-processing with an automatic mask generation from user input signal only. With this unified pipeline, a user could get the distraction target to be removed with an accurate and efficient experience. Second, I also construct an attention-based module that allows the user to control the reference area of contextual attention [8]. Thus, the user can interact with current inpainting models with external input guidance. Also, the module could alleviate the foreground distraction problem. The proposed efficient user-oriented system has addressed the image inpainting challenge for practical uses.

1.2 Contributions of This Thesis

To summarize the contributions of this dissertation:

- This thesis builds a multi-scale pyramid generator to conduct image inpainting by following the strategy of “structure first detail next”.

  - The proposed pyramid generator is constructed with several sub-generators from the bottom to the top layer. The multi-scale generative architecture

isolates the learning of image global structures and local details during the restoration.

  - The pyramid generator is trained jointly with a learning scheme of progressively increasing hole size, which benefits the large hole inpainting.
  - The edge-preserved smooth image on the bottom layer is responsible for learning image structure without distracting high-frequency information.
  - The multi-scale architecture assists the proposed model for inpainting high-resolution images.

- This thesis proposes a model on top of the existing two-stage method with enhanced features to better model the image structure and texture information.

  - This method builds Skip Connection Attention Fusion (SCAF) that is constructed with skip connections and attention feature fusion technique. This module could compensate for the texture information from the encoder to the late semantics in the decoder features.
  - The proposed Dilated Convolution Residual (DCR) could better fuse the features from the multi-dilated layers to capture the image global structure and multi-scale information.
  - Both SCAF and DCR aim to strengthen the texture and structure aggregation and reduce the inconsistency of semantics during learning. I have shown quantitatively and qualitatively that the approach outperforms current methods on benchmark datasets.

- This thesis proposes two user-oriented image inpainting methods where object detection/segmentation and attention mechanism are employed.

  - The proposed efficient object removal system integrates object segmentation with image inpainting into an aggregated system. This system allows users to remove any foreground objects by only clicking on the removal target.
  - The proposed guided attention module allows users to guide the inpainting system with a guidance map input and obtain favorable results.
  - I have shown the effectiveness of the interactive system on the developed album dataset PB-101 and real-world scenarios.

1.3 Organization

Chapter 2 Background This chapter introduces the literature on image inpainting and discusses its pros and cons. I divide the image inpainting into traditional and deep learning-based groups. The traditional category includes diffusion-based and patch-based methods. I elaborate on the representative works and their limitations before introducing deep learning methods. In the deep learning section, I analyze the basic building blocks and categorize these methods into single-stage models and progressive models based on their constructions. At last, I also provide an overview of benchmark datasets on images and masks, plus the standard evaluation metrics.

Chapter 3 Image Inpainting with Pyramid Generator This chapter presents a multi-scale pyramid model for image inpainting. This model is constructed by stacking several sub-generators from multi-layers. The pyramid architecture could isolate the structure and texture learning and thus favor large-hole and high-resolution inpainting. Additionally, this method adopts an edge-preserved smooth image on the bottom layer for better structure construction. The proposed model could alleviate the conflict between image structure and texture during training and thus overcome the problem of artifacts and blurry patches.

Chapter 4 Image Inpainting with Attention Feature Fusion This chapter introduces a model based on attention feature fusion for image inpainting. This model enhances the feature learning in the feature fusion perspective to address the inefficient utilization of features on the current encoder-decoder model. The proposed skip connection attention fusion could pass the low-level textures to high-level semantics, and the dilation convolution residual is responsible for fusing multi-dilated layers. These two modules finally help construct enhanced features with accurate textures and strong semantics.

Chapter 5 Efficient User-oriented Image Inpainting In this chapter, I present two user-oriented image inpainting systems with real-world uses. The chapter first introduces a fast single-shot object detector as a pre-processing step in the interactive system. The efficient detector could reach high-speed detection without sacrificing accuracy. Next, I introduce the system, which integrates object segmentation with inpainting. This system could bring an efficient and high-quality experience for users with the requirement of interactive object removal. The proposed second system is an attention-based model that allows users to indicate the reference regions during inpainting. With the guidance map as input, users could manipulate the generated content or interactively improve the generation quality.

Chapter 6 Conclusion and Future Work The summary of contributions and future works are included in the last chapter.

Chapter 2

Background

This chapter provides a comprehensive survey of image inpainting. The works from Efros and Leung [15] and Bertalmio *et al.* [16] are two pioneer image inpainting works. Efros and Leung [15] proposed a texture synthesis based on Markov Random Field. Given a sample patch, this method synthesis a new image based on the self-similarity of pixels within the sample. This texture synthesis is a non-parametric method, and it could generate one pixel each time through querying from the exemplar image. The texture synthesis is ideal for inpainting images with simple structures and similar textures, while it cannot handle images with complicated objects. Bertalmio *et al.* [16] proposed a diffusion model dependent on the Partial Differential Equation (PDE). This work dedicates to filling the corrupt area by propagating pixels of the image from the surrounding region into the unknown region smoothly. However, this diffusion method only works for small and thin masks, and it is limited to the content from remote regions.

These pioneers have significantly pushed forward the development of image inpainting methods. Much further research inspired by the pioneer works [15, 16] is considered traditional image inpainting methods. With the rise of deep learning, many researchers have applied this leading technology to image inpainting [5]. These deep learning methods can learn from large-scale data and could generate high-quality inpainting results. Based on these findings, I categorize the overall image inpainting methods into traditional and deep learning methods. This chapter first discusses the taxonomy and then introduces each category and the related algorithms. I also provide discussions of each category with its shortcomings and applications. The organization of this chapter is: Section 2.1 reviews the traditional patch-based and diffusion-based methods; Section 2.2 introduces the deep learning approaches and the basic building model as preliminaries, including deep Autoencoders (AE) and Generative Adversarial Network (GAN); Section 2.3 summarizes the datasets used for training and evaluation in image inpainting; Section 2.4 discusses the standard evaluation metrics.

As shown in Fig. 2.1 is the taxonomy of overall image inpainting approaches. This

figure also shows the transition from traditional methods to deep learning approaches. All the methods mentioned in this dissertation are highlighted in Fig. 2.1.

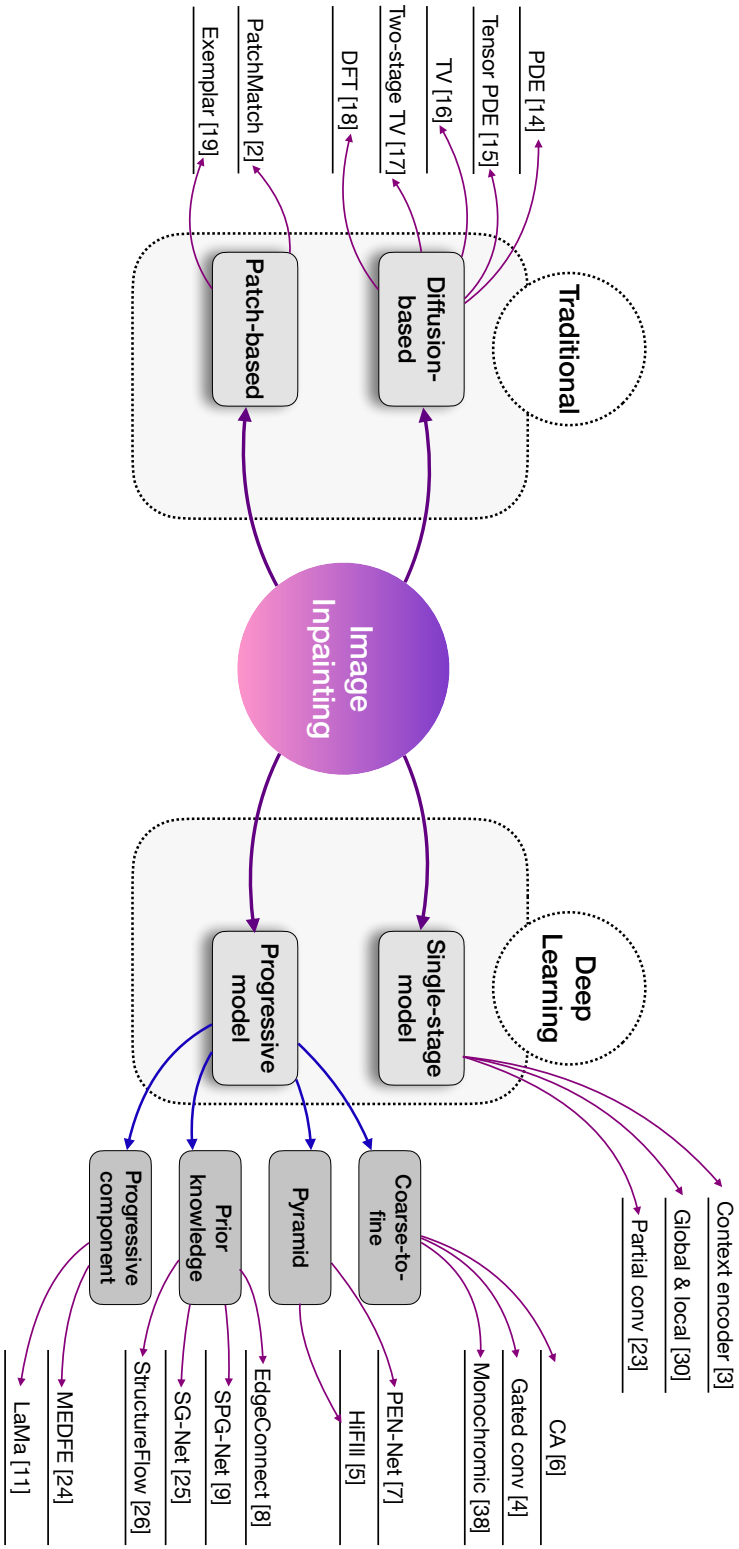


Fig. 2.1: The taxonomy of image inpainting in Chapter 2.

2.1 Traditional Image Inpainting Methods

The section divides the traditional image inpainting methods into diffusion-based and patch-based methods based on the taxonomy in other survey [3, 17].

2.1.1 Diffusion-based Methods

The diffusion-based image inpainting methods propagate the pixels gradually from the surrounding region into the hole region. Bertalmio *et al.* [16] is considered the first work with the diffusion method. This algorithm is based on Partial Differential Equation (PDE). Iteratively, it propagates the features from boundary regions to the inside regions in the isophotes (lines of equal gray values) direction. The diffusion process can be shown as follows:

$$\mathbf{I}^{n+1}(i,j) = \mathbf{I}^{n}(i,j) + \triangle t \cdot \mathbf{I}_t^n(i,j), \forall (i,j) \in \Omega. \tag{2.1}$$

Where Ω is the hole region, n represents the iteration time, (i,j) are set of pixels coordinates, $\triangle t$ indicates the rate of the change, and on the image of $\mathbf{I}^n(i,j)$, the $\mathbf{I}_t^n(i,j)$ represents the update factor. The results of this method are shown in Fig. 2.2 (from the original paper). This algorithm can be run automatically and efficiently on the image with small cracks. However, it generates blurry results under multi-masks or complicated structures due to the blurring artifact of diffusion and the lack of process on the edge of boundaries.

Fig. 2.2: Inpainting results from one of the traditional methods by [16]. Only simple and restricted mask type is supported.

Inspired by [16], Tschumperlé [18] proposed a tensor-driven PDE on color images as a regularization method. This work built heat flow constraints on integral curves to

preserve the thin structures during restoration. This technique has bought smoothness to the boundary regions while failing on large hole filling.

Chan and Shen [19] proposed a Total Variational (TV) inpainting model in 2005. This model applies the Euler-Lagrange equation and anisotropic diffusion-based on isophotes. In the work of [20], they have extended the TV model to a two-step algorithm. In the first step, this method tries to solve the non-linear Total Variation (TV)-strokes equation to find the isophote directions for the missing regions and conduct the initial inpainting. In the second step, they construct the final results based on the initial construction from the previous step.

A more recent work proposes a new mathematical model based on fractional-order derivatives with Discrete Fourier Transform (DFT) [21]. With the adoption of fractional-order derivatives, the model benefits from information with the whole image instead of neighborhoods only. The DFT is easy to implement and could help denoise without affecting the edge structure. This method is limited to manually setting the fractional order, which may lead to terrible results with lousy selection.

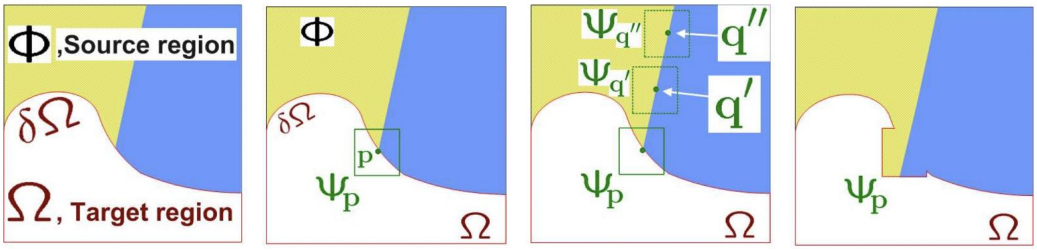
In summary, diffusion-based methods borrow the information from the boundaries to smoothly fill in the target region. These methods are suitable for restoring small cracks or scratches. For the image with significant structure change, they often construct blurry edges or artifacts due to the lack of overall understanding of the semantics within the image. Meanwhile, these methods always need an iterative process to diffuse the pixels gradually, which is time-consuming.

2.1.2 Patch-based Methods

Another branch is patch-based methods, which are also called exemplar-based approaches. These methods are derived from texture synthesis [15]. During the inpainting process, the surrounding patches of holes are regarded as a reference; then, these holes are filled-in with similar pixels within the reference patches. They usually produce better results than diffusion-based ones, especially on images with relatively larger holes. Typically, patch-batched methods can be separated into two steps: 1) Allocate the priority; 2) Choose the best candidate patch.

Criminisi *et al.* [22] proposed a method that combines texture synthesis and isophote-oriented inpainting. They adopted a priority mechanism to select the best surrounding patches, and these patches are used to fill the holes. In the first stage, the method assigns the contour pixels as priority pixels; in the second stage, as in texture synthesis, they copy the most similar content from the neighborhoods based on the priority score of each contour patch from the previous stage. The contour pixels have the highest priority compared with the inner ones, and they are filled earlier. Fig. 2.3 has shown the inpainting

process of the structure propagation. From Fig. 2.3(b), point P is the center pixel of the highest priority patch. It lays on the edge of the hole. Fig. 2.3(c) explains that the best-matched patch is selected among the neighborhood region, and this patch is used to fill the hole as in Fig. 2.3(d). It is noticeable that the patch size is arbitrary, depending on the image. This method has archived good results on images with accessible structures thanks to the isophote-driven fashion, but it could not handle the curved structures. Nonetheless, this approach generates artifacts due to the matching system's severe instability, which may select incorrect patches.

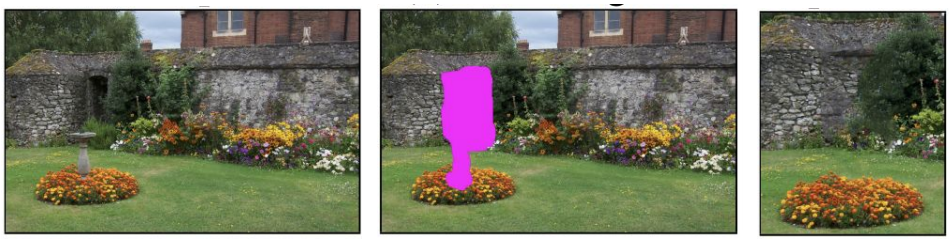


(a) (b) (c) (d)

Fig. 2.3: The whole inpainting process of the patch-based inpainting methods adopted from [22]. (a) The input image is split with target and source regions, and there is contour between them. (b) One of the filled patches is chosen based on the highest priority value. (c) The most likely candidates for filling the patch. (d) The chosen highest priority pixel is filled with patch copying from the candidate in the previous step.

To reduce the high computation cost of the best patch searching, PatchMatch [4] proposed a fast randomized algorithm that applied Nearest Neighbour Fields (NNF). This approach uses an iterative searching mechanism for finding the best-matching patch. During each iteration, the patches are selected through random sampling according to the results of NNF between two disjoint regions. Also, the pixel propagation based on natural coherence in the imagery helps disseminate good results to adjacent pixels. The combination of NNF and natural coherence pixel propagation guarantees the preservation of structure and texture simultaneously and vanishes many artifacts compared with the methods mentioned above. PatchMatch is an efficient interactive solution for image inpainting, and it has shown potential in image editing applications, such as image retargeting, completion, and reshuffling. This method cannot deal with the image with non-stationary textures due to the independence between targeted regions and known regions under complicated scenes. Fig. 2.4 shows some results of PatchMatch.

Overall, patch-based methods assume that the holes have similar content to the unbroken area. They fill the holes by searching for the best candidate patches from the known region. Patch-based approaches have shown to be more efficient and accurate than previous diffusion-based ones, especially on an image with structure replication. However, it



(a) image (b) masked image (c) results

Fig. 2.4: The inpainting results of PatchMatch [4]. From the example, the table above flowers is removed from input (a) with the mask from (b) as input.

still exists several limitations:

- The decision of the filling order has a significant impact on the final results;
- If the targeted region has an independent texture from the rest regions, these methods struggle to generate good results;
- Current models find the best candidates based on Euclidean distance between patches, which lacks a complete understanding of image information and may lead to semantically unsatisfactory inpainting.

Table 2.1: Summary of traditional image inpainting methods, the main ideas and their categories.

Reference	Main Idea	Category
Bertalmio *et al.* [16]	It propagates the features from boundary regions to the inside regions in the isophotes direction.	Diffusion
Tschumperlé [18]	This work proposed a tensor-driven PDE on color images as a regularization method.	Diffusion
Chan and Shen [19]	They have extended the TV model to a two-step image inpainting algorithm.	Diffusion
Sridevi and Srinivas [21]	It proposed a new mathematical model based on fractional-order derivatives with Discrete Fourier Transform.	Diffusion
Criminisi *et al.* [22]	It proposed a method that combines texture synthesis and isophote-oriented inpainting.	Patch
Barnes *et al.* [4]	It proposed a fast randomized algorithm that applied Nearest Neighbour Fields.	Patch

2.1.3 Discussion

This subsection summarizes the traditional image inpainting methods and discusses the transition from traditional methods to deep learning methods.

Traditional inpainting methods try to restore the deteriorated image by either diffusing the boundary smoothly or searching for the best-matching patches. Both of these methods were well-fit on:

- image with small targeted regions, such as a crack and small square;
- stationary image with uncomplicated structure;

- the content of unknown and known regions is not independent.

However, with the development of digital media, thousands of complicated images are produced by people on the Internet. Hence, the image inpainting task becomes more and more challenging. One of the most discussed problems is the inpainting of the large arbitrary mask. Unfortunately, the conventional methods tend to synthesize blur or non-realistic repetitive patches due to the diffusion and exemplar mechanisms having an inherent limitation on learning high-level image semantics. Moreover, these techniques often require a long processing time which is not suitable for real-time applications. Recently, researchers have found ways to use the tool of deep learning, especially deep generative neural networks, and have archived significant progress on building more advanced inpainting models. With training on the large-scale data, these models can learn high-level image understanding, which has proven to help generate consistent texture and coherent structure.

This chapter leaves more discussion of the deep learning methods in the following section.

2.2 Deep Learning Methods

The emergence of deep learning [23] and Generative adversarial networks (GAN) [24] has pushed forward image inpainting to a great extent. The utilization of deep generative networks has the advantage of fully unearthing the image information in terms of texture, structure, color, and internal relationships. These advantages allow deep learning-based algorithms to produce more plausible results than traditional ones. Typically, these modern methods leverage CNN-based encoder-decoder architecture to restore the deteriorated image and combine with GAN for an adversarial training fashion. The encoder is used as a feature extractor of inputs to learn the high-dimensional data abstractions. The decoder is responsible for decoding these learned features into a complete image. The use of GAN could further improve the generation results with more realistic patches.

The pioneering work [25] uses CNN as a denoise model and extend to image inpainting. However, their model is limited by computational resources and only works for greyscale images. Context encoder [5] is the first GAN-based work with an encoder-decoder model. This model is designed for restoring small resolution images with little square masks. Many further works aim to inpainting damaged images with arbitrary-shape masks [6, 26]. Some other researchers seek to separate the learning of image structure and texture with a two-stage model [6, 8], and these works are further improved to more sophisticated approaches (i.e., pyramid architecture) with finer results [9, 13, 27]. Meanwhile, many work utilize other image features as auxiliary information, e.g.

edge [10], semantic map [28], smoothed image [29]. These additional resources are often used as prior knowledge and guide more accurate inpainting. Nonetheless, other deep learning-based methods are dedicated to high-resolution image inpainting [7, 14, 30].

The section categorizes deep learning methods based on the architecture and the data/resources used during training. Overall, I broadly divide these methods into single-stage inpainting and progressive inpainting. The summary is shown in Fig. 2.1. The single-stage group could be briefly described as methods using single feed-forward networks. Secondly, the progressive group indicates the improved methods from single-stage ones. They can be divided into coarse-to-fine models, models with priors, pyramid architecture models, and single-stage models with improved components.

Before diving into these categories, this section first introduces the basic building blocks of these deep learning methods as preliminaries.

2.2.1 Preliminaries

This section introduces the basic building blocks of modern deep learning-based image inpainting models. I first discuss the autoencoder, which is responsible for image context learning and image construction, followed by the generative adversarial network (GAN) as a training technique for a more realistic prediction.

Autoencoder

Autoencoder is an unsupervised representative learning technique constructed with neural networks. For generation tasks, the autoencoder takes an image as input and aims to reconstruct it through an encoder-decoder architecture. Specifically, the encoder learns a compact feature representative, and the decoder tries to reconstruct the input from the learned features. Mathematically, for an input x, the model wants the encoder $f(\cdot)$ to learn a hidden layer h which contains the abstraction of input data x. Then, the reconstruction process on the decoder can be denoted as $y = g(h)$, where y is the reconstructed output and $g(\cdot)$ is the decoding function. See in Fig. 2.5. The process can be expressed as:

$$\min_{w_1,w_2} ||x_i - g(f(x_i; w_1); w2)||_2^2, \tag{2.2}$$

where w_1 and w_2 are weights to learn.

The idea of the autoencoder is derived from [31]. Originally, autoencoders were designed for dimensionality reduction or feature learning [32]. Due to its connections with latent variable modeling, autoencoders have been broadly adopted on deep generative models. In image inpainting, the encoder is mainly responsible for learning the context of patches. The learned features should be able to retrieve the neighbor context for producing semantically meaningful patches within the hole. As for the decoder, the function

is to construct a complete image from the compact features.

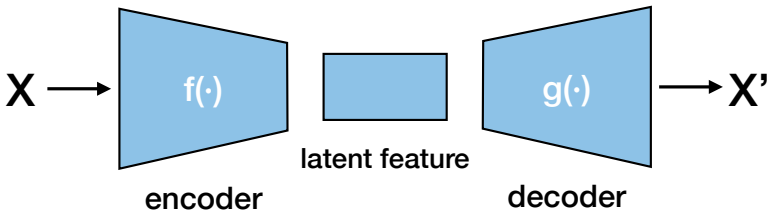


Fig. 2.5: Autoencoder architecture.

Generative adversarial network (GAN)

Generative adversarial network (GAN) [24] is a generative model constructed with a generator and a discriminator. During training, the generator always tries to compete against the adversary discriminator. Specifically, the generator produces samples $x = G(z; \theta^G)$ from the input z. This synthesized sample is paired with a real training sample and is sent to the discriminator. The discriminator output is $D(x; \theta^D)$, which denotes the probability of the sample x being a real sample. As a zero-sum game, we use a function $v(\theta^G, \theta^D)$ to represent the payoff of the discriminator, $-v(\theta^G, \theta^D)$ denotes the payoff of the generator, and the training process can be expressed as:

$$G^* = \arg\min_G \max_D v(G, D). \tag{2.3}$$

Thus, the overall loss function is:

$$\min_G \max_D v(G, D) = \mathbb{E}_{x \sim p_{data(x)}} \left[\log D\left(x\right)\right] + \mathbb{E}_{z \sim p_{z(z)}} \left[\mathbf{1} - \log D\left(G\left(z\right)\right)\right]. \tag{2.4}$$

As in the above objective function Eq. 2.4, the generator tries to "fool" the discriminator while the discriminator tries to distinguish whether the image is a generated "fake" or a real one. When reaching the final state point, the generator's output is indistinguishable from the real sample, and the discriminator should be discarded. Fig. 2.6 shows the architecture of GAN model. Inpainting approaches use GAN to generate conditional results for high-quality restoration, and an encoder-decoder network models the generator for low-level patch generation. The learned context from the generator is further strengthened with high-level consistency through adversarial training, obtaining more realistic images.

One major issue of GAN model is that the instability of training due to the nature of deep neural networks and $\max v(G, D)$ is not convex [32]. To address this problem, one of the milestones is deep convolutional GAN (DCGAN) proposed in [33] and archived sound performance on image synthesis tasks. Many works are dedicated to this problem in image inpainting, and they proposed many solutions for a stabilized training. These

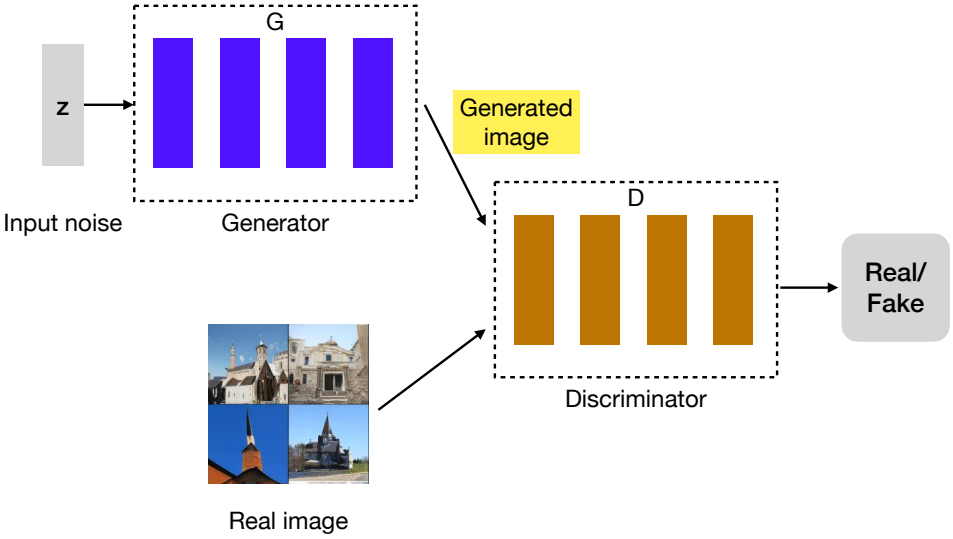


Fig. 2.6: The architecture of GAN. The input is a noise z. The generated image is paired with with a real image, and they are sent to the discriminator for distinguishing.

attempts include a modified discriminator architecture, novel loss functions, adoption of new activation functions, or new mask generation algorithms for more robust training samples.

In the following sections, I will introduce the milestones of the image inpainting model using deep learning and discuss their strengths and weaknesses.

2.2.2 Single-stage Inpainting

Context encoder [5] is the first GAN-based image inpainting work with an encoder-decoder model. As shown in Fig. 2.7 is the architecture of Context encoder. They applied an autoencoder as the generator backbone for pixel estimation and built a fully connected layer for passing information between the encoder and the decoder. Since it is impossible for the encoder feature to connect with all locations within a specific feature, they used a channel-wise fully connected layer to propagate information within activations of each feature map. During training, they utilize the $\mathcal{L}2$ loss as reconstruction loss and an adversarial loss. The reconstruction loss can capture the hole area's structure and help to fill with suitable pixels, but it will average the multiple modes of predictions. Thus, the adversarial loss would take the effect of picking a more realistic pattern from the distribution. The validation results on metrics such as Peak Signal to Noise Ratio (PSNR) have shown the outperformance of Context encoder compared with the traditional methods. However, this method is limited to low-resolution image inpainting and only works for square masks. Moreover, Context encoder generates artifacts because the discriminator solely focuses on the hole area and lacks image global information learning. Nonetheless,

Context encoder proposed an inpainting baseline model with a context learning autoencoder, and it has inspired future works enormously.

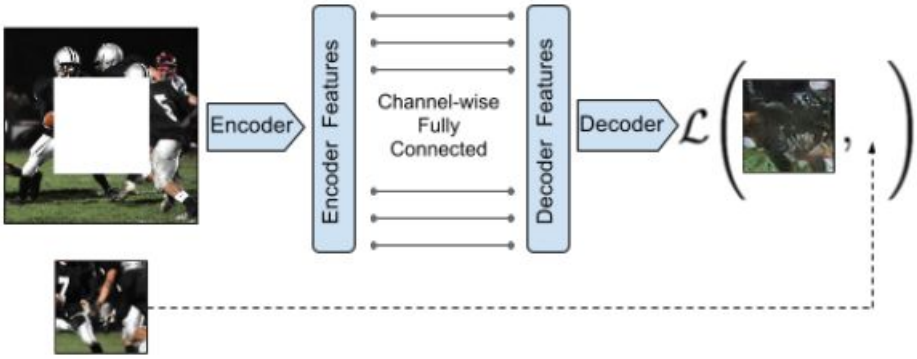


Fig. 2.7: Model of Context encoder [5].

Inspired by Context encoder, Iizuka *et al.* [34] improved the baseline model with two auxiliary context discriminators during training - global and local. The overall structure is shown in Fig. 2.8. Specifically, the global discriminator takes the entire generated image as input to capture the global semantics. In contrast, the local discriminator only takes part of the image (128×128 on the center of the filled image via the original paper) as input to maintain the local consistency. This novel design has addressed two main problems of the Context encoder. First, the independent learning of global and local context forces the network to balance global and local semantics, resulting in the generated images with semantically meaningful patches. Second, this model could work on larger resolution image inpainting. Another innovation of this method is adopting dilated convolutions into the image inpainting model to increase the receptive fields of features, which has shown promising results and has been widely used in future works. However, two primary defects of this method are incapable of free-form masked inpainting, and they heavily rely on post-processing (fast matching [35] and Poisson blending [36]).

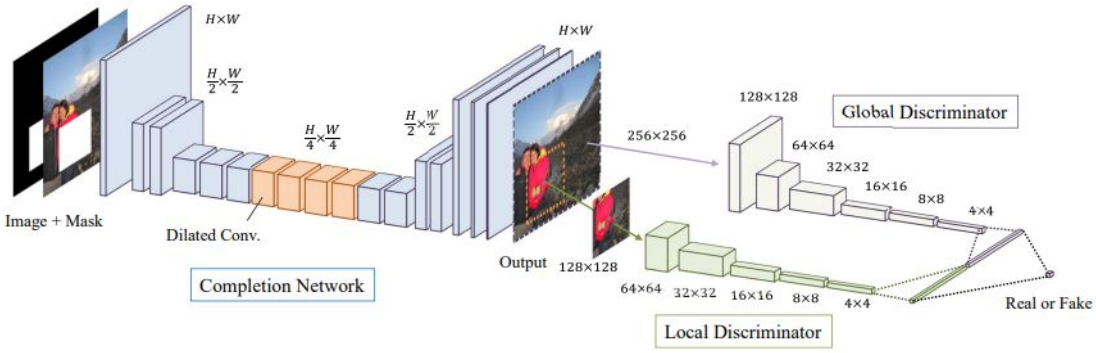


Fig. 2.8: Model of Global&Local [34].

Liu *et al.* [26] proposed that the vanilla convolution is unsuitable for inpainting tasks since the average value of the valuable area and hole area is applied during the filling. The texture differences between these two regions could inevitably introduce artifacts. This effect is particular when inpainting on large masks. To address this, partial conv is proposed with a novel mask-update scheme. Overall, the partial conv is a normalized convolution operation that only focuses on valid pixels, and the normalization process is an automatic mask updating after each layer. The mask updating is a non-learnable process, and a binary mask is updated on valid pixels where the convolution could condition its output in these areas. Thus, along with the network going deeper, this mask shrinks with all valid pixels on the final. They built the partial conv on top of the vanilla U-Net architecture. This method also works well on free-from masks inpainting thanks to its automatic mask-update mechanism. The quantitative results compared with [34] on PSNR and Structure Similarity Index Measure (SSIM) [37] have shown the effectiveness of partial convs (some results are shown in Fig. 2.9). However, this approach still struggles with large masks and sparsely structured inpainting, mainly because the rule-based mask update is limited under these scenarios. Specifically, the invalid pixels are converted into valid ones followed by the mask layer progressively, while the local and global information would be missing in the deep layers. Moreover, they have generated a publicly available mask dataset with different mask-to-image ratios with or without boundaries, which is widely used for further research.

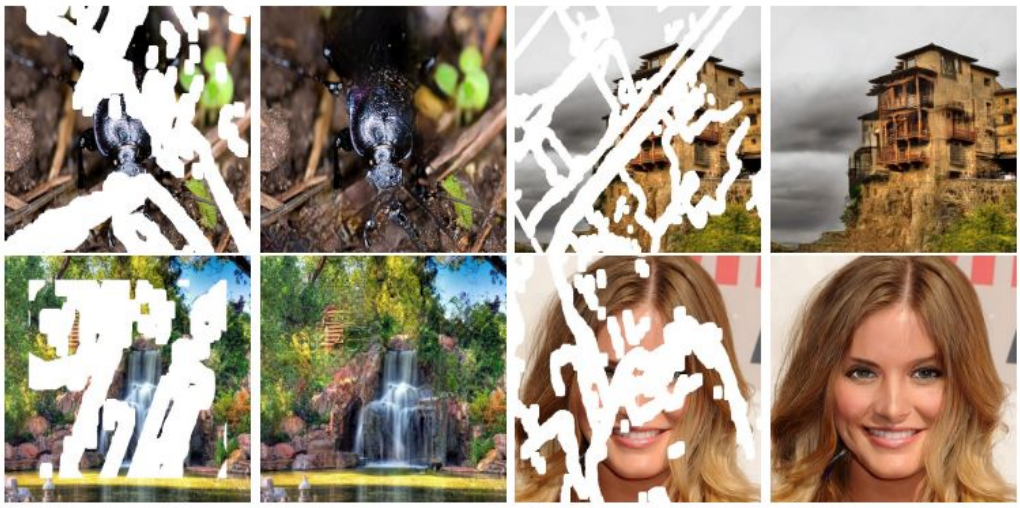


Fig. 2.9: Inpainting examples of partial conv [26]. Left are masked inputs with free-from masks, right are the inpainted resutls from partial conv.

2.2.3 Progressive Inpainting

The single-stage inpainting methods have built the fundamental backbones. Inspired by these single-stage approaches, there has been a spurt of development in deep learning-based image inpainting methods. The following methods try to build high-quality models progressively. This section discusses these methods in detail.

Coarse-to-fine model

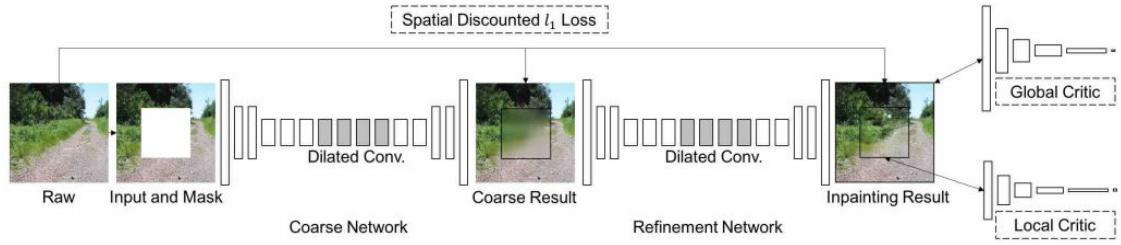


Fig. 2.10: Framework of Contextual attention [8].

One of the first works that adopted the two-stage inpainting models is Contextual attention [8]. Instead of using a single autoencoder as the generator, they proposed a coarse-to-fine model where the coarse stage is responsible for generating the overall structure within the hole. The refining stage aims to retouch the hole with fine-grained patches. This two-stage model is inspired by residual learning [38] and deep supervision [23], or more similar to the traditional art restoration process, which is to draw the structure at the initial and fill in with details later. As shown in Fig. 2.10 is the framework of Contextual attention. They built a fully-convolutional two-stage model. The inputs are downsampled twice, and the compact deep features are extracted in the "bottleneck" of the model. The output of the coarse stage is a "half-complete" image with global structure, i.e., the large patches or color consistency. This output is sent to the second stage as an initial and makes more delicate improvements on top of it. Both stages applied a similar encoder-decoder architecture, except a novel contextual attention layer is inserted in the refine stage. This attention module tries to learn the relationship between the hole and background patches based on cosine similarity within the feature level. The learned similarity guides the model to fill holes with the most similar patches. This module has further improved the spatial coherency between background and foreground patches and thus could generate more plausible results. As for the discriminator, they use the similar structure of [34] with two parallel global and local discriminators. They use two Wasserstein-GP GAN losses from [39, 40] for a stabilized training and a spatial attenuation reconstruction loss for pixels learning. This model has been regarded as a baseline for the future work, and the attention module has shown effectiveness, such as in [41], they extended the contextual attention to their coherent semantic attention module by considering the patches

relationships within the holes in addition. However, the issues with high-resolution and free-from masks inpainting still exist.

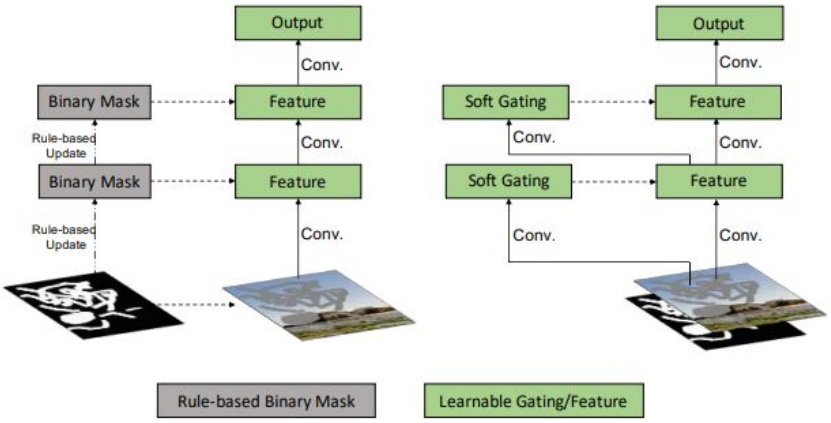


Fig. 2.11: Gated convolution [6] (right) compared with partial convolution [26] (left).

Built on top of [8], Yu *et al.* [6] improved partial conv with a novel gated conv operation, and this model is also called Gated. They proposed a learnable dynamic mask update mechanism by adding a gate operation on each vanilla convolution, as shown in Fig. 2.11. This gate process allows the network to learn a soft mask of each input feature. At each channel level, the feature is conditioned on the soft mask, which learns the valid pixels in a learnable fashion. The benefit of gated conv is that it automatically learns the useful area that contributed to the final results without introducing an external module. The generator is almost the same as [8], but with gated conv replacing all the vanilla conv layers. They also use a spectral-normalized Markovian discriminator, which is also called SN-PatchGAN as the discriminator for generating results with vivid patches and training in a more stable manner. Gated has shown promising results compared with [5, 8, 26], especially on free-form masks inpainting (see in Fig. 2.12), thanks to their gated conv. One problem of gated conv is the high computational cost since it introduces the gating operation on the vanilla convolution layer. Moreover, they introduce a user-guided mechanism that allows users to interact during the inpainting. This mechanism is implemented by adding an external channel after the inputs, and this channel is addressed as a guidance feature during the training.

To address the artifacts, which include blunt structures and abrupt colors, Wang *et al.* [42] proposed a novel two-stage model called Monochromic bottleneck (see Fig. 2.13). They use an external-internal scheme that learns the semantic information externally from a large-scale dataset while learning the internal statistics through the single test mechanism. The first external stage only inpaint on the image in the monochromic space, which reduces the learning dimension. The reconstructed structure and details are sent to

Fig. 2.12: Free-from inpainting examples of Gated convolution [6]. From left to right are original images, masked inputs and inpainted images. The black line on the second example indicates the user-guided input for interactive inpainting.

the second internal stage as a self-learning process, where the monochromic reconstruction only learns color consistency from the internal distribution itself. This way, their model could eliminate many color bleeding artifacts from the previous methods, and the external-internal scheme could produce coherent structures. One major drawback of their method is that internal time is a self-learning process that consumes much inference time.

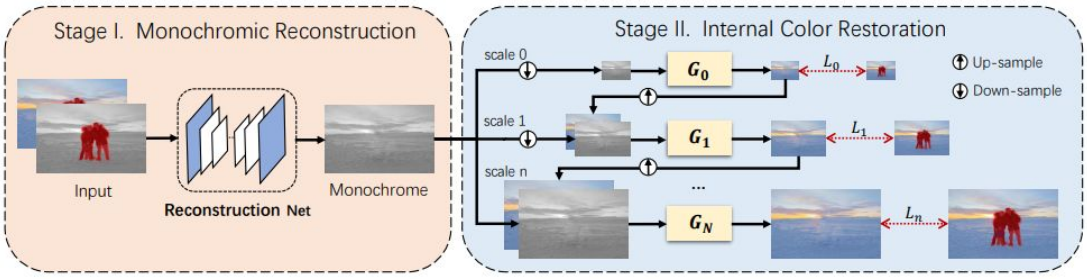


Fig. 2.13: Framework of Monochromic bottleneck [42]. This external-internal model learns the monochrome image with structure and texture from training on a large-scale dataset, and conducts the color restoration internally.

Pyramid model

In essence, the advantage of coarse-to-fine models compared with the single-stage ones are the separated learning of image global and local information. The motivation is that image global information contains more structures, while local information learns image textures. The single-stage often generates results with blurry patches or patches without coherence from the background, and this is because they could not learn the image structure and texture optimally. Apart from the above coarse-to-fine model, another group of methods tries to deal with this issue with a pyramid architecture. This section introduceS these methods detailedly.

Pyramid-context encoder [9], also called PEN-Net, is an image inpainting model constructed with a feature pyramid. Traditional methods often apply GAN model or building with stacked pooling layers to capture high-level semantics, while the results lack fidelity. PEN-Net aims to learn image semantics from the high-level features and then pass them to each low-level one.

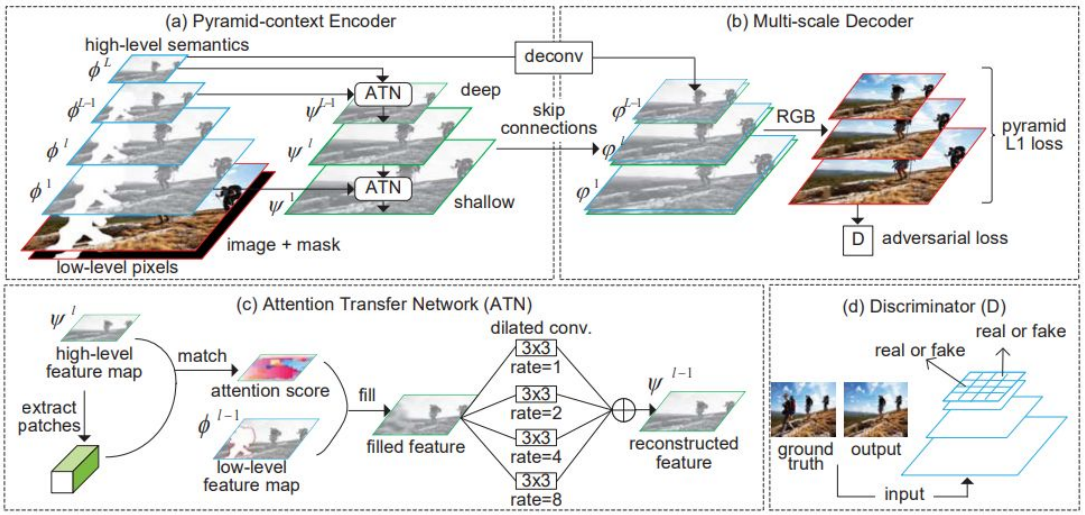


Fig. 2.14: Architecture of PEN-Net [9].

The model is shown in Fig. 2.14, and they built a feature pyramid encoder combined with a multi-scale decoder. To learn the affinity within feature space, they used an Attention Transfer Network (ATN) to fuse different level feature maps. Similar to [8], the cosine similarity is used to learn the region affinity from the high-level feature. After the softmax activation, they obtain the attention score, which guides the low-level feature map. Thus, the high-level semantics is passed to the lower level through the ATN module. Their multi-scale decoder introduced dilated convolutions in different dilation rates and aimed to refine the results further. Both visual and numeric comparison has shown the effectiveness of their pyramid architecture. However, PEN-Net's feature pyramid still synthesizes artifacts under complicated scenarios, which indicates the feature pyramid has limitations.

Previous methods focus on low-resolution image inpainting, e.g. 256×256, while the high-resolution inpainting is urgently needed as the availability of high-quality data. A recent work, HiFill [7] is proposed to address this problem with a pyramid architecture. HiFill aims for inpainting images with at least 512×512 in resolution, and it can handle 4K images through its high-frequency residuals layer. The overall model can be regarded as a pyramid structure, as shown in Fig. 2.15. The whole inpainting process can be elaborate as high-resolution input is down-sampled into 512×512 at first, and the generator could

produce a "low-resolution" complete result. At the same time, the high-frequency residuals layer is responsible for keeping the sharp details or original high-resolution input. This layer is implemented by adding learned contextual residuals through the Attention Transfer Module (ATM) to upsampled the generator output.

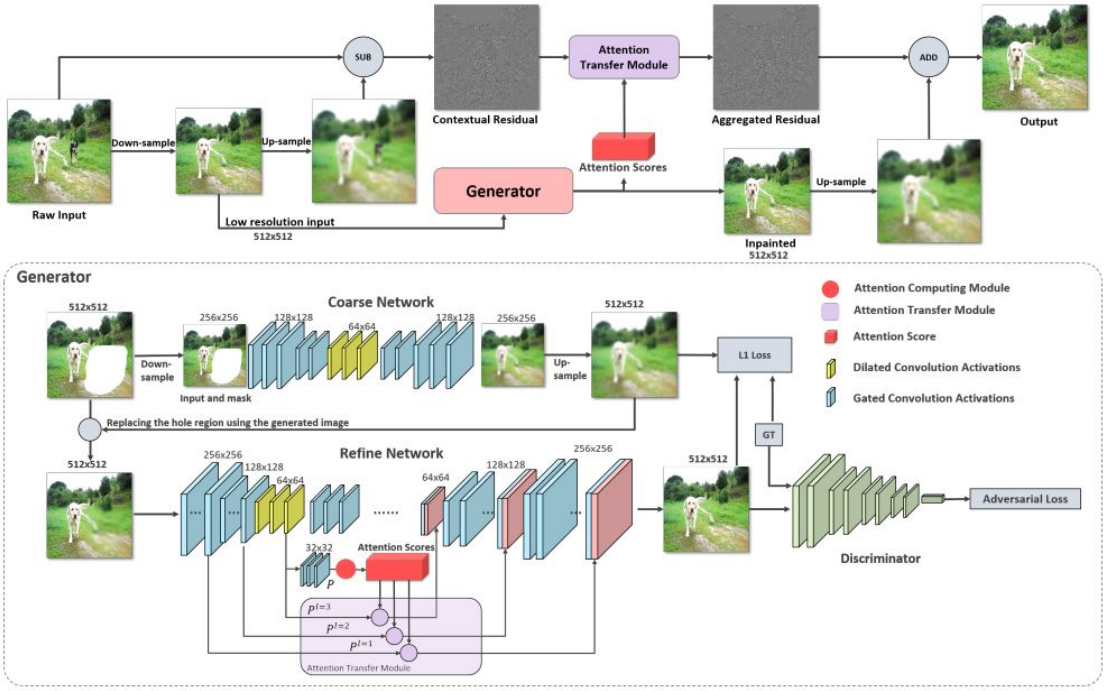


Fig. 2.15: HiFill [7] architecture.

The generator is a similar coarse-to-fine architecture as [6]. The difference is that HiFill uses 256×256 input in the coarse network while 512×512 in the refine network. Noteworthy, the proposed Contextual Residual Aggregation (CRA) firstly use the same attention mechanism as [8] to calculate the attention scores across different feature pyramid, then pass these scores into the residual layer via the ATM module. In order to reduce the networks' parameter, they use a lightweight gated convolution through depth-separable convolutions followed by 1×1 conv as a gating operation. HiFill has shed light on future works on high-resolution image inpainting. However, the performance is still limited in specific environments.

Model with prior knowledge

The methods discussed above only use the RGB images as inputs for training end-to-end inpainting models. Some researchers proposed that using other information as prior knowledge will guide the GAN network to carry out more structure learning [43]. These priors include image edges, contours, structural images, or semantic maps. Thus, as for image inpainting, it is natural to adopt these priors to restore images has more meaningful textures and precise semantics. This section discusses the methods that lie in this category.

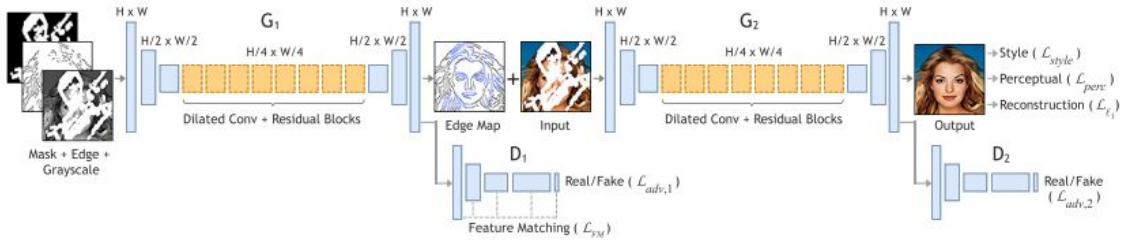


Fig. 2.16: Model summary of Edgeconnect [10]. The incomplete grayscale image plus mask plus edge are the inputs of G_1. G_1's output edge map with input color image is sent to G_2 from final inpainting.

Nazeri *et al.*. [10] proposed a two-stage image completion model called Edgeconnect. They presented that using edge as priors could eliminate previous methods' over smoothness and blur. As shown in Fig.2.16 , in the first stage, Edgeconnect uses its edge generator to complete the missing edge map with a prediction edge map. Then, in the second stage, the previous prediction edge is concatenated with the incomplete image for image inpainting. Some of the examples are shown in Fig. 2.17. The examples have shown that Edgeconnect works well on generating sharp edges of the missing area. However, further evaluations indicated that this model could not predict reasonable textures when filling large holes.

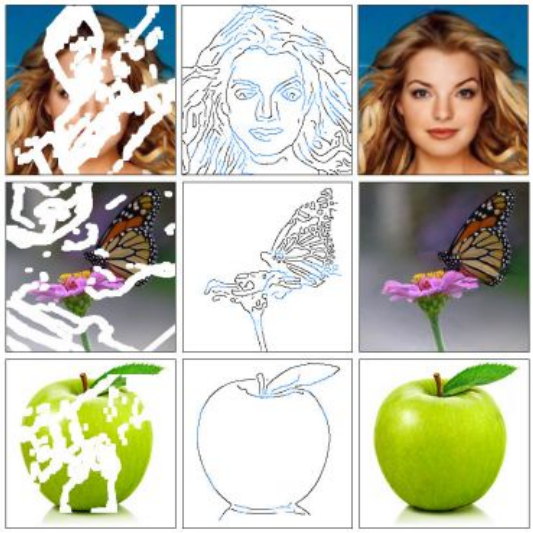


Fig. 2.17: Inpainting examples of Edgeconnect [10]. From left to right are input images, completed edges (drawn in black are detected by Canny algorithms from available regions, drawn in blue are generated), and inpainted results.

Instead of using edges as prior, SPG-Net [11] uses segmentation information to assist image inpainting in an end-to-end model. Since semantic segmentation could help disentangle the inter-class and intra-class differences of images, using the semantics could

generate images with better texture within similar semantics consistency. Similar to Edge-connect [10], they use a two-stage model constructed with a segmentation prediction network and a segmentation guidance network. As seen SPG-Net in Fig. 2.18. The first stage predicts the segmentation labels within the holes, and the following stage is responsible for generating segmentation-guided inpainting images. For the semantic segmentation initialization, they choose the Deeplabv3 [44] for inference and predict a label map with size $256\times256\times C$, where C is the number of label categories. The inpaint network is similar to [34]. One of the major drawbacks is that this model needs to train on datasets with semantic segmentation maps as labels, which is constrained by limited testing scenarios and could limit the model's use cases.

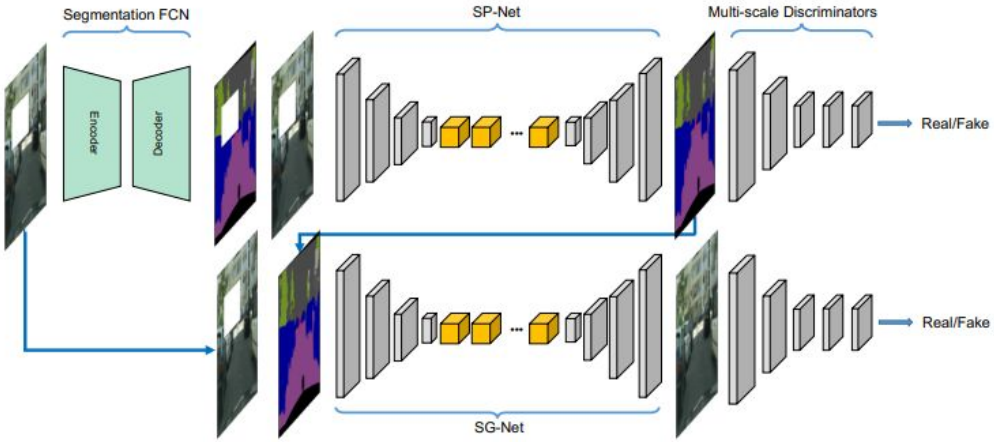


Fig. 2.18: Overview structure of SPG-Net [11]. The Deeplabv3 is used for segmentation initialization and SP-Net for segmentation map prediction. The final stage is SG-Net for generating the complete image.

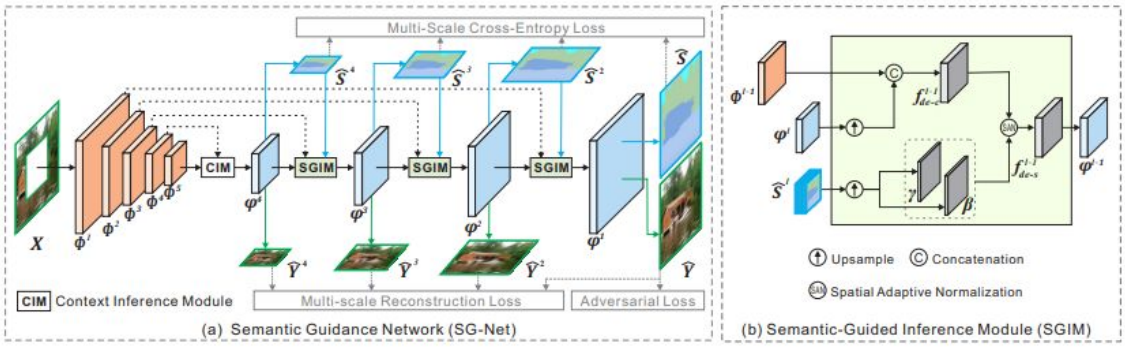


Fig. 2.19: The model of SG-Net [28]. The contextual features are updated progressively through the SGIM module. The SGIM module takes the inputs of the segmentation map and previous features.

Another approach using semantic segmentation as a prior is from Liao *et al.* [28]. In SPG-Net, the predicted segmentation map is sent to the inpainting network directly with

the masked image, which has the defect that the semantics may not be fully passed into the generation process. This defect may lead to results with unrealistic structure and ambiguous texture. The proposed method learns the structural priors and inpainted images in an iteratively updated manner. Their interplay framework (see Fig. 2.19) guarantees that the semantics can be fully learned during the inpainting. The encoder first extracts the features from the masked RGB inputs during the training. Then, the decoder is responsible for simultaneously predicting the semantic segmentation maps and the inpainted images. They adopted a multi-scale decoder with multi-scale cross-entropy and reconstruction loss for supervision to add the segmentation information to the inpainting process. Moreover, during the inference, the segmentation map is used to guide the update of the next level feature through the proposed SGIM module (see Fig. 2.19). The validation of the Outdoor Scenes [45] and Cityscapes [46] datasets shows that the progressive multi-scale refinement model could generate coherent patches within the same semantic scene. This method still has the same limitation as SPG-Net [11] regarding the training dataset. Although they did evaluations on Places2 [47] using some similar outdoor examples, the training dataset's limitation is still a problem.

Generating images with reasonable structure is always a problem in modern image inpainting models. Meanwhile, the edge-preserved smooth images [48, 49] have the feature that only the low-frequency structures are preserved. Inspired by this finding, a method called StructureFlow [29] uses the edge-preserved smooth images as image priors to assist the image inpainting task. Their model is shown in Fig. 2.20. They adopted a popular two-stages model, where the edge-preserved smooth images are used as inputs for training a structure reconstructor on the first stage. The reconstructor outputs a predicted complete structural image and is fed to the second stage for image inpainting. In the second stage, they use appearance flow to model the long-range connections between holes and known regions to synthesize coherent contexts. Overall, their method separates the structure and texture learning into two stages, and each stage leverages an encoder-decoder architecture with both $\mathcal{L}1$ and adversarial loss. Compared with previous edge-based methods (i.e., [10]), the results show that the edge-preserved images outperform the edges in terms of assisting the structure generation.

Models with progressive components

Some researchers proposed that the previous methods could not efficiently leverage their encoder-decoders features, thus missing the ability to capture image structures or textures during the generation. Instead of using a multi-stage network or a pyramid architecture, they build components to strengthen the vanilla convolutional features and seek the optimal structure & texture learning within the plain autoencoder.

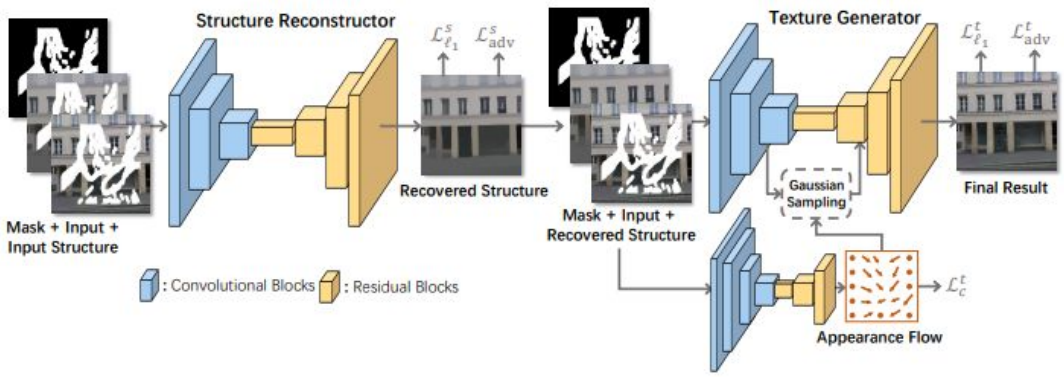


Fig. 2.20: The two-stage architecture of StructureFlow [29]. Inputs of the first stages include images, masks, and structural images. They are used to generate the recovered structure by the structure reconstructor. The texture generator in the second stage uses appearance flow to synthesize image textures.

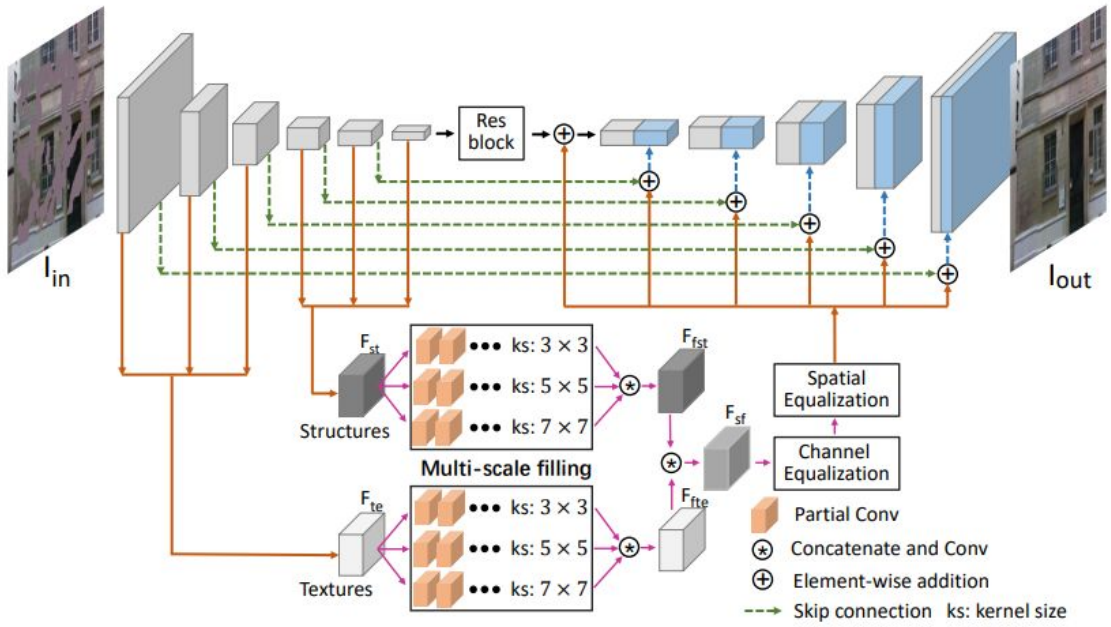


Fig. 2.21: The model of MEDFE [27]. The encoder features are regrouped into structure and texture branches. After equalizing the features in both channel and spatial levels, the obtained features are sent to the decoder through skip connections.

One of the representative works is MEDFE [27]. MEDFE argues that the previous models with structure and texture learning in different stages will generate inconsistent patches, and this is due to the independent learning on different networks. They proposed a single encoder-decoder model with a feature equalization method to strengthen the features with consistent structure and texture. They first concatenate shallow and deep features into two groups, followed by reweighing channels. Finally, they use the

bilateral propagation activation function to pass these encoder features to the decoder. Fig. 2.21 shows the overall model of MEDFE, including the feature equalization method. This method has shed light on other researchers for building components to enhance the feature consistency in a mutual model.

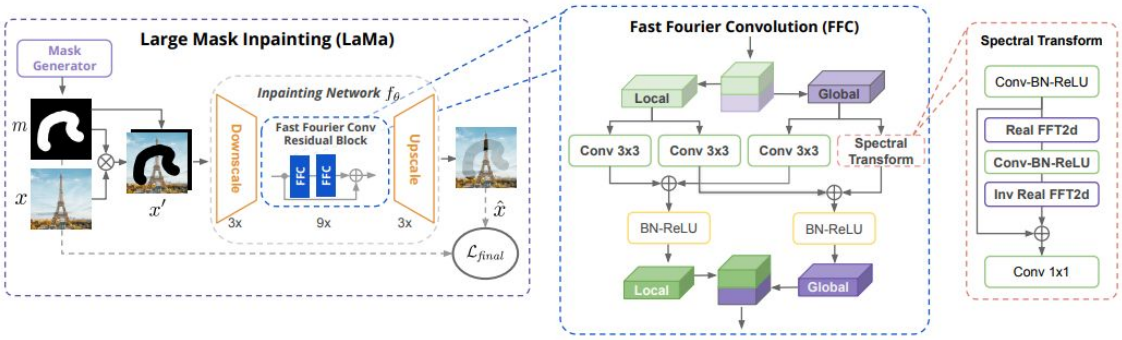


Fig. 2.22: The architecture of LaMa [13]. LaMa adopts fast Fourier convolution (FFC) in the residual block of their inpainting network. The structure of FFC is on the right.

Suvorov *et al.* [13] proposed a one-stage model called LaMa with fast Fourier convolutions (FFCs) [50]. As is shown in Fig. 2.22, the model is a straightforward encoder-decoder architecture. The use of FFC in the bottleneck is to enlarge the receptive field of features. They found that a larger receptive field helps to generate reasonable structures, and FFC could capture more global context than vanilla convolutions. The right part of Fig. 2.22 elaborates the FFC. This module is a fully differentiable operation, where inputs are halved on channel level and are sent to local and global branches. The global branch uses spectral transform to learn image global information. Finally, two branches are fused into one with the same dimension as the input feature. In addition to FFC, they also proposed a high receptive field perceptual loss to enlarge the receptive field of the model further. This novel loss is derived from perceptual loss [51], which calculates the distance between generated image and ground truth on the feature level. The network to extract features in perceptual loss is $f(\cdot)$, and they modify the $f(\cdot)$ with dilated or Fourier convolutions into a new network $f_{HRF}(\cdot)$. Thus, the final high receptive field perceptual loss can be expressed as:

$$\mathcal{L}_{HRFPL}(x, \hat{x}) = \mathcal{M}([f_{HRF}(x) - f_{HRF}(\hat{x})]^2), \tag{2.5}$$

where $\mathcal{M}$ is the sequential two-stage mean operation. From extensive evaluations of different datasets, LaMa has shown promising results on high-resolution image inpainting (especially on large masks) due to its high receptive field strategy.

Table 2.2: Summary of deep learning-based image inpainting methods, the main ideas and their categories.

Reference	Main Idea	Category
Context encoder [5]	It is the first GAN-based image inpainting work with an encoder-decoder model.	Single-stage
Iizuka *et al.* [34]	This work improved the baseline with two auxiliary context discriminators during training.	Single-stage
Liu *et al.* [26]	The proposed partial conv with a novel mask-update scheme.	Single-stage
Contextual attention [8]	They proposed a coarse-to-fine model with a contextual attention module.	Progressive: coarse-to-fine
Gated *et al.* [6]	It improved partial conv with a novel gated conv operation.	Progressive: coarse-to-fine
Wang *et al.* [42]	This work proposed a two-stage model with a monochromic bottleneck.	Progressive: coarse-to-fine
PEN-Net [9]	It is an image inpainting model constructed with a feature pyramid.	Progressive: pyramid
HiFill [7]	It is proposed to address the high-resolution inpainting problem with a pyramid architecture.	Progressive: pyramid
Edgeconnect [10]	It proposed a two-stage image completion model using edge as prior.	Progressive: prior
SPG-Net [11]	It uses segmentation information to assist image inpainting in an end-to-end model.	Progressive: prior
Liao *et al.* [28]	The proposed method learns the structural priors and inpainted images in an iteratively manner.	Progressive: prior
StructureFlow [29]	It uses the edge-preserved smooth images as image priors to assist the image inpainting task.	Progressive: prior
MEDFE [27]	It proposed a single encoder-decoder model with a feature equalization module to strengthen the features.	Progressive: components
LaMa [13]	It proposed a one-stage model with fast Fourier convolutions.	Progressive: components

2.3 Datasets

The choices of datasets in image inpainting are essential, especially for deep learning-based methods. Meanwhile, due to the characteristics of this task, the datasets of masks are also necessary for training and evaluation.

For training datasets, since it is hard to collect large-scale corrupted data in real life, researchers usually train their models on suitable image datasets and use the automatically generated masks. As the models become more powerful, the generated masks become more complex, and it has revolutionized from solely center square masks to random free-from masks plus square masks. Training with more complex masks will also generate a more robust model that can handle real-life problems. This section introduces several popular images and mask datasets, especially for modern inpainting models.

2.3.1 Places2

Places2 [47] is the most used training dataset in image inpainting. Initially, it is designed for high-level visual understanding tasks, such as scene context and object recognition. The images are collected based on principles of human visual cognition. Places2 has more than 10 million images in total, comprising over 400 scene categories (indoors and outdoors). Each class contains 5000 to 30,000 images that meet the real-world frequencies of occurrence. Based on the size of images, Places2 has two sets of data, one with 256×256 as small images set and another set of high-resolution images which are resized to have a minimum dimension of 512 while preserving the actual aspect ratio. Some of the examples from Places2 are shown in Fig. 2.23.

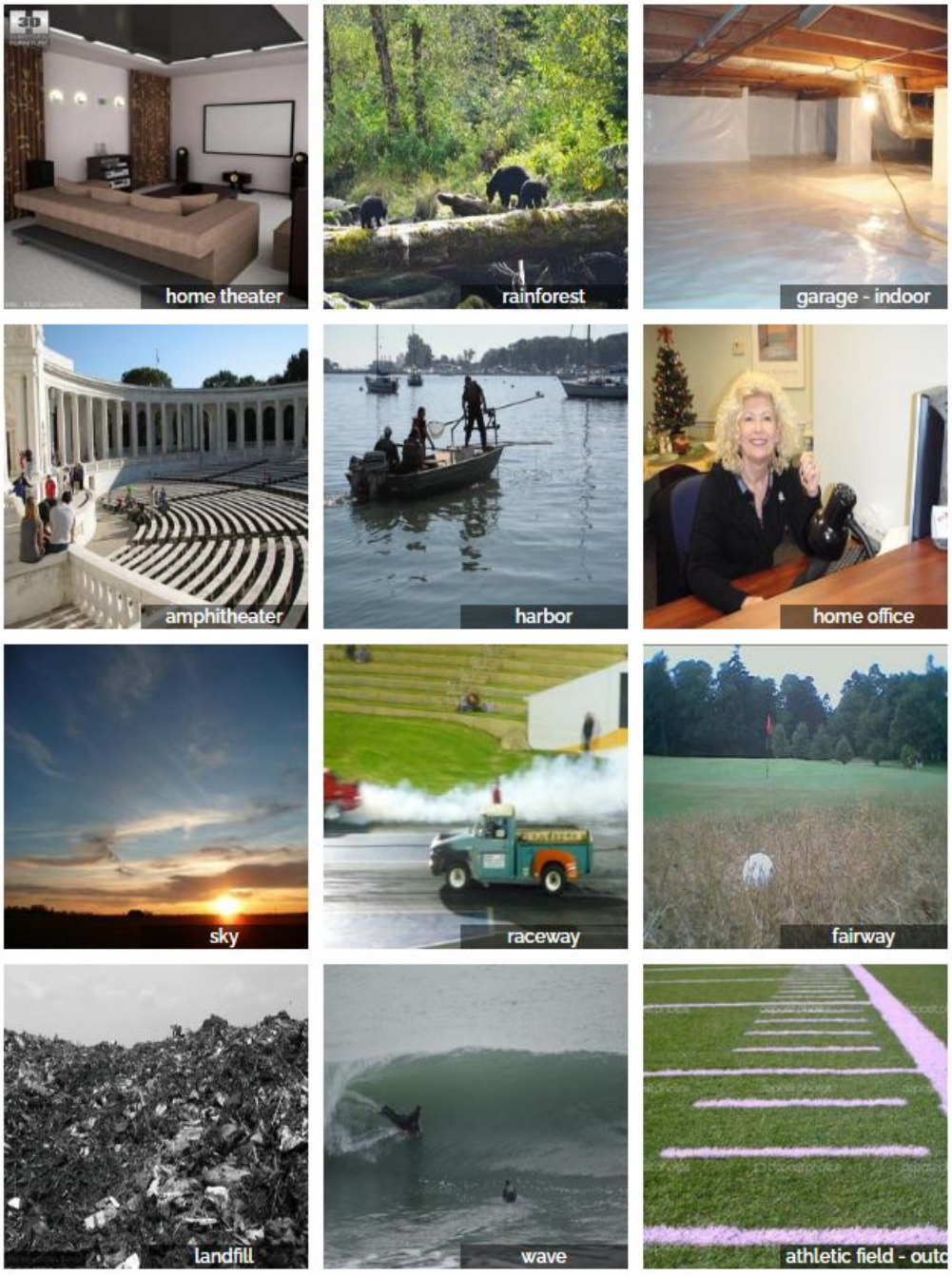


Fig. 2.23: Examples of Places2 [47] dataset.

2.3.2 CelebA

CelebFaces Attributes Dataset (CelebA) [52] is a dataset contains 202,599 facial images of celebrities. These images include 10,177 identities, five landmark locations, and each with 40 binary attribute annotations cropped to size 178×218. CelebA has been used for facial recognition and face generation tasks as a large-scale dataset for facial data.

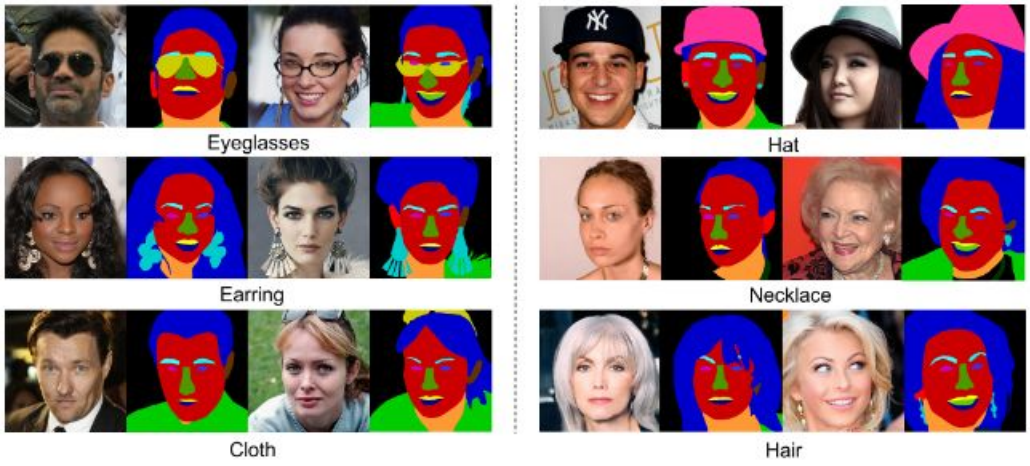


Fig. 2.24: Several examples from CelebA-HQ [53] dataset.

For more wide use cases, Karras *et al.* [53] developed the CelebA-HQ dataset which containing 30,000 high-quality images in 1024×1024, 512×512 and 128×128. They also make sure all the images in the original CelebA with various backgrounds have facials in the center of the images. This is implemented by removing the artifacts via method in [54], then by a super-resolution method trained adversarially in [55]. To extend the dimension, they applied padding and filtering techniques. After processing all the 202,599 images from CelebA, they analyzed and obtained 30,000 high-quality images, which compose the final dataset. Fig. 2.24 shows some examples of CelebA-HQ.

2.3.3 Paris Street View

Paris street view [56] is developed from Google Street View [57] to test the best algorithms works for a computational geographic task. This dataset contains street view images of Paris, with 15,000 high-quality images total. These images mainly focus on buildings in the city (see Fig. 2.25).

2.3.4 DIV2K

The DIVerse 2K (DIV2K) [58] dataset is proposed for image super-resolution task. It contains 1000 high-quality images with 2k resolution, which is considerably higher than

Fig. 2.25: Paris street view [56] examples. Source from [17].

other popular datasets. These images are originally collected via the Internet from diverse sources, and the content covers a large diversity of objects and environments. Some of the examples can be shown in Fig. 2.26.

Fig. 2.26: DIV2K [58] examples.

2.3.5 Partial conv Masks Dataset

Due to the specificity of the task image inpainting, the datasets of masks also play an important role, especially during validation. One of the most used validation masks datasets is from [26]. They developed a mask set containing 12,000 irregular greyscale images. There are six different mask-to-image ratios from 0-10% to 50%-60% with or without border constraints (holes ensure at least 50 pixels from the boundary). Some mask examples have shown in Fig. 2.27. This mask dataset has been an industry benchmark for validation due to its variety and different mask-to-image ratio settings.

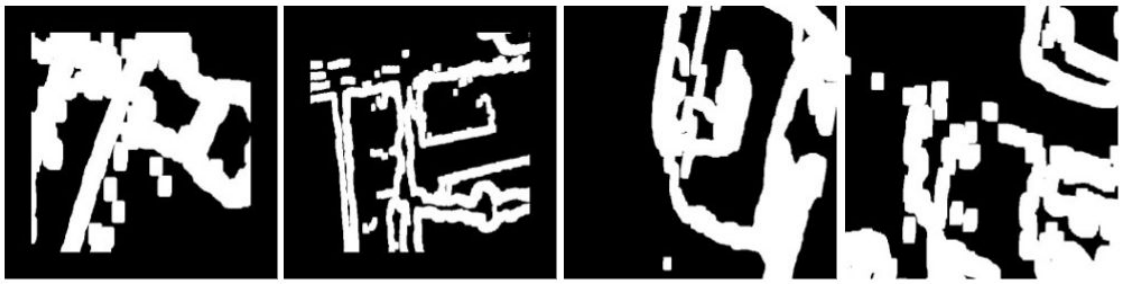


Fig. 2.27: Some examples from partial conv masks set [26]. From left to right are masks with and without border constraints.

Table 2.3: Summary of popular image inpainting datasets and their properties.

Name	Content category	Size	Resolution
Places2 [47]	Multiclass	10m	256×256 and 512×512
CelebA [52]	Face	202,599	178×218
CelebA-HQ [53]	Face	30,000	1024×1024
Paris street view [56]	Street	15,000	936×537
DIV2K [58]	Multiclass	2000	2k

2.4 Evaluation Metrics

Typically, image inpainting algorithms are evaluated quantitatively and qualitatively. The qualitative evaluation includes applying the model to popular validation datasets with different mask types or conducting a questionnaire. The questionnaire evaluation is often time-consuming and involves subjectivity. Thus, it is crucial to exploit the metrics for quantitative evaluation. The choices of these metrics are based on evaluating whether the restored image has reasonable structure and plausible textures. Some commonly used are

$\mathcal{L}1$ (Mean Absolute Error), Peak Signal to Noise Ratio (PSNR), and Structure Similarity Index Measure (SSIM) [37]. These metrics calculate the error of hole regions between restored and ground truth images. Given the ground truth image I and the generated image I', the $\mathcal{L}1$ can be denoted as:

$$\mathcal{L}1(I, I') = \frac{1}{N}\sum_{i=1}^{N} |I_i - I'_i|. \tag{2.6}$$

The Eq. 2.6 measures the average pixel error between the generated outputs and reference images.

PSNR is the ratio between the maximum possible value (power signal) and the power of distorting noise that affects the fidelity representation based on two images (restored/-ground truth). It is often used to quantify the reconstruction quality of images or videos. The calculation of PSNR is

$$PSNR = 20\log_{10}\frac{MAX_I}{\sqrt{MSE}}, \tag{2.7}$$

where MAX_I is the max possible pixel value of the image I, MSE is the mean squared error value. The higher of the PSNR value, the better of the reconstruction quality.

SSIM is another commonly used metric. It is a perception-based measurement that regards image degradation as the perceived change in structural information. There are three factors considered in SSIM: loss of correlation, luminance distortion, and contrast distortion. These factors are calculated based on the neighboring pixels of the candidate images. The calculation can be expressed as:

$$SSIM(I, I') = \frac{(2\mu_I\mu_{I'} + c_1)(2\sigma_{II'} + c_2)}{(\mu_I^2 + \mu_{I'}^2 + c_1)(\sigma_I^2 + \sigma_{I'}^2 + c_2)}, \tag{2.8}$$

where c_1 and c_1 are constants, μ_I and σ_I are average and variance of I, respectively, and $\sigma_{II'}$ denotes the covariance of I and I'.

The above three metrics $\mathcal{L}_1$, PSNR and SSIM are used in most evaluations. However, recent literature indicated that these metrics could not reflect the image with human perception. Thus, they proposed to use Frechet inception distance (FID) [59] and learned perceptual image patch similarity (LPIPS) [60] to evaluate the generated image.

FID is based on a pre-trained model (InceptionNet [61]) on ImageNet. It is calculated through computing the Frechet distance between multivariate Gaussians fitted to the embedding space of the Inception-v3 network of the restored and real images. FID is consistent with human perception and thus is widely used in GAN model evaluations. The lower scores have shown that the generated images have better quality.

LPIPS is designed to calculate the perceptual distance between two images. In LPIPS, a weighted $\mathcal{L}_2$ difference between two VGG16 [62] embeddings is used. The measure can reflect human perception, and a low LPIPS indicates the patches are perceptually similar.

Table 2.4: Summary of popular image inpainting metrics and their properties.

Name	Quantification level	Need of external resources
PSNR	Pixel	No
SSIM [37]	Pixel	No
L1	Pixel	No
FID [59]	Feature	Yes
LPIPS [60]	Feature	Yes

2.5 Summary

This chapter summarizes the literature on image inpainting based on the proposed taxonomy. The literature begins with the introduction of traditional methods and the discussion of transmission to deep learning-based approaches. The following part comprehensively studies deep learning methods categorized into single-stage and progressive models. Lastly, the popular datasets plus the evaluation metrics are presented in detail.

Chapter 3

Image Inpainting with Pyramid Generator

Current deep generative model is still hard to generate coherent and realistic image details [6, 63]. Such challenges probably stem from the neural networks' spectral bias [64], *i.e.*, neural networks are biased towards learning low-frequency components instead of high-frequency details. Most state-of-the-art methods adopt a coarse-to-fine framework to alleviate this problem. Taking the Gated [6] as an example, inputs are first restored with a coarse network, and then the details are further refined at the second stage.

Furthermore, there is a conflict in modelling image global structures and local details [63]. To tackle this problem, another two-stage model StructureFlow [29] adopt the edge-preserved smooth images as supervision in their structure reconstructor. The smoothed image discards the high-frequency details [48, 49] and help the structure reconstructor without disturbance of textures. However, modelling image structures from the smoothed image only is not enough. The global information often prefer to learn from large image regions or even the whole image, while modelling image details prefers to learn from small image patches. This dilemma requires different receptive field sizes, and it is hard to satisfy in a single neural network model.

By our understanding, image inpainting looks like artists drawing a picture for a scene. Artists usually firstly draw the global structures/sketches of the scene and then refine its local details [10, 65]. Inspired by that, we suggest explicitly separating the restoration of image structures from that of details and following the philosophy of "**structure first detail next**". Thus, we propose to engage distinct sub-generators to restore image structures and details, respectively.

In particular, we build a Pyramid Generator for image inpainting. The pyramid generator is constructed by stacking several sub-generators from the bottom to the top layer. Inputs are downsampled into different resolutions and are fed into corresponding sub-generators. The bottom sub-generators focus on restoring the image global structures,

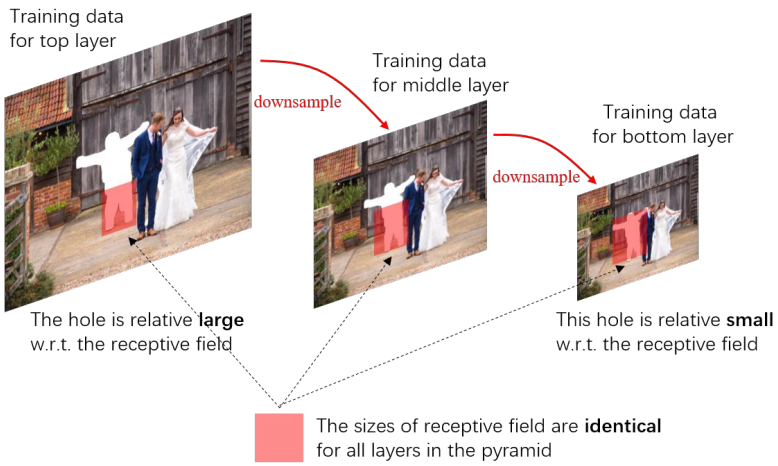


Fig. 3.1: Hole size to receptive field ratios on different layers of a pyramid. Bottom-layer inpainting task (right) looks like a small-hole inpainting task, while the top-layer inpainting (left) looks like a large-hole inpainting. It is obvious that the mask size of bottom layer is relative small with respect to the same receptive field size, while the mask size of top layer is relative large.

while the top sub-generators focus on restoring image local details. Within the pyramid, we firstly construct the image structure under the supervision of the edge-preserved smooth image. Then, we refine the reconstruction on the higher layer through the global information learning on the pixel level. At last, higher levels are responsible for learning image details.

Note that the previous *coarse-to-fine frameworks* in [6] only cascade two similar sub-generators (i.e., they are trained with inputs of the same resolution) and fail to separate the image structures restoration and image details restoration effectively. On the contrary, with the *multi-scale multi-layer stacking framework*, our pyramid generator can not only effectively isolate the structures and details modelling and indeed practise restoring structures before details.

It is well-known that image inpainting becomes much more challenging when the corrupted area is relatively large [7, 63]. Some recent work [63] illustrate that such **large-hole challenges** could be effectively alleviated by a progressive learning strategy, i.e., first learn to restore small hole and then learn to restore a large hole. Our approach precisely aligns with this strategy. As shown in Fig. 3.3, both input image and mask image are simultaneously downsampled in our pyramid. Thus the ratio of hole size to image size is identical for all layers. However, whether the hole size is large or small should be measured concerning the size of the convolutional receptive field (instead of image size). As shown in Fig. 3.1, since all sub-generators at different layers have the same architecture as well as the same receptive field size, the hole size is *relatively* small for the low-layer

sub-generator. Thus, the low-layer inpainting looks like the small-hole inpainting task. In contrast, the high-layer inpainting looks like large-hole inpainting. With our bottom-up inpainting workflow, we are aligning to the hole-increasing strategy, i.e., first conduct small-hole inpainting and then large-hole inpainting. As a result, our pyramid generator has the advantage of dealing with large holes.

Finally, our pyramid generator is more suitable for **high-resolution image inpainting**. Many previous works have shown that increasing the resolution of training images could benefit the high-resolution image inpainting [7, 9]. Our experimental results also validate such observation, as shown in Section 3.3.3. However, we find that such performance gain is still limited if we just directly train the existing models (e.g. Gated [6]) with high-resolution images (e.g., on 512×512 images), as shown in Fig. 3.7.

We argue that such observation is also related to the large-hole challenge. Since high-resolution training images often come with large holes, we have to face the large-hole challenge if we directly train a generator with high-resolution images. Nevertheless, due to the capability of handling the large-hole challenge, our approach could fully exploit the advantages of learning with high-resolution images. As a result, our approach is more suitable for high-resolution image inpainting.

We highlight our contributions of this section as follows:

- A Pyramid Generator is proposed to conduct image inpainting by following the strategy of "structure first detail next". Our dedicated multi-scale generative architecture alleviates the conflict between image global structures and local details restoration.
- Our pyramid generator has a learning scheme of progressively increasing hole size, which allows it to restore large holes.
- We further isolate the structure learning under the supervision of edge-preserved smooth image in our structure branch.
- Our approach could reap the benefits of learning with high-resolution images, and hence is suitable for inpainting high-resolution images.

3.1 Related Works

3.1.1 Image Inpainting with Pyramid Structure

Conventional image inpainting methods often utilize low-level image statistics [66]. PatchMatch [4] fills holes by searching similar patches from unfilled area. Although effective

at textured images, these methods often generate artifacts or incoherent content when inpainting complex images with global structures.

Recently, many deep learning-based methods are proposed with great improvement [5–7, 26, 34, 67]. Context encoder [5] presents the first attempt to apply the convolution neural network on image inpainting. Iizuka *et al.* improved the architecture with both global and local discriminators to keep consistency [34]. Partial convolution [26] is proposed to handle free-formed masks by using only valid pixels as conducting convolutions. Recently, Gated [6] introduced a contextual attention module and a coarse-to-fine learning framework, which has significantly improved the inpainting performance. A multistage attention module is proposed in [67] for better utilizing the non-local relationships in different feature levels. HiFill [7] focused on high-resolution inpainting tasks and proposes a contextual residual aggregation mechanism.

3.1.2 Multi-scale Mechanism

Multi-scale design is applied broadly in many computer vision tasks thanks to its progressive refinement property [68–70]. One representative work is SinGAN [70], where a pyramid of GANs is proposed to generate a realistic image sample of arbitrary size and aspect ratio. In particular, the pyramid GANs have a multi-scale framework: image is gradually generated with the increasing image scale, from the bottom layer to the top one.

The multi-scale mechanism is also applied to the image inpainting task. In [71], images are gradually refined by increasing image scales, but only $\mathcal{L}_2$ content loss, VGG textual loss, and TV loss are used for training. Since adversarial loss is not adopted for learning, it is still hard to generate realistic image details. Recently, a pyramid-context encoder PEN-Net is proposed to incorporate high-level semantics with low-level pixels in [9]. PEN-NET has only one encoder, and there is a feature pyramid within the encoder.

On the contrary, our model is similar to SinGAN, which has a stack of distinct sub-generators. All the sub-generators are trained in an adversarial manner with independent discriminators. We argue that those distinct sub-generators could better model image structures and local details, respectively.

3.2 Structure First Detail Next: Image Inpainting with Pyramid Generator

In this section, we first introduce the architecture of our pyramid generator, including its layer fusion strategy and the adaptive dilation scheme. Next, we discuss the loss function and describe how to train our pyramid generator effectively.

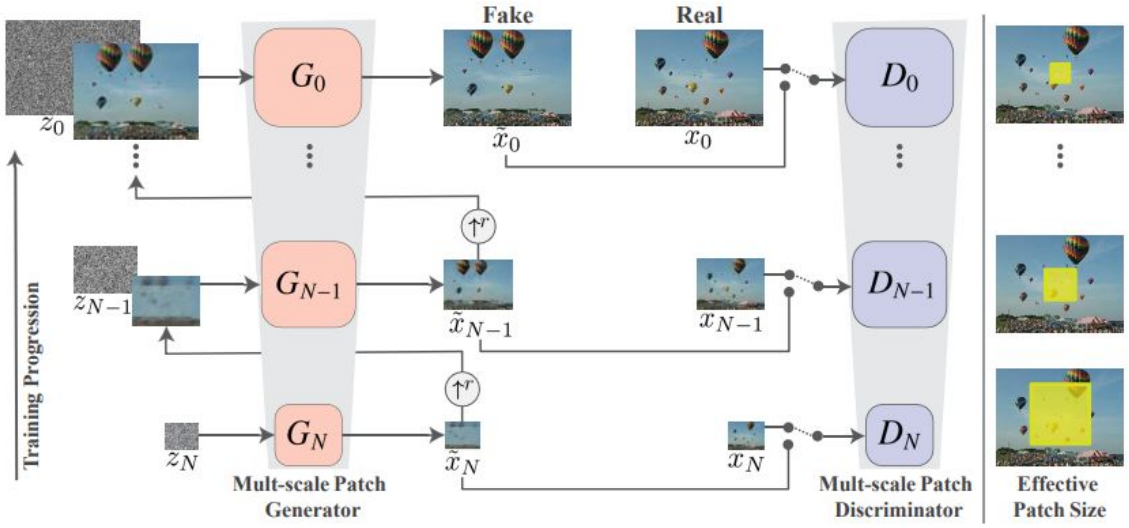


Fig. 3.2: SinGAN's model architecture [70]. The pipeline contains a pyramid of GANs. The generator on each level generates different sizes of images from random noise. The discriminator on each level has the same structure; thus, the effective patch size on generated images decreases as the pyramid goes up (yellow mask on the photos). The overall generation is a level-by-level process, passing the generated image to a higher level for a larger image generation.

3.2.1 Architecture Design

In order to fully capture the internal statistics of an image at different scales, [70] proposed a hierarchical GAN architecture to learn the distribution within each scale separately. Inspired by that, we adopt its hierarchical framework to separate the modeling of image global structures and local details at different scales.

Revisit SinGAN

We firstly revisit SinGAN in this section. SinGAN is an unconditional generative model focusing on generating images from one single input. Thus, this model is required to capture internal image statistics and be able to generate high-quality and diversified samples. To this end, they proposed a pyramid architecture with GANs on each level. These GANs are responsible for learning the different scales of patches within the single input image. The pyramid has two advantages: 1). low-level GANs could guarantee the global structure is known during training, while high-level GANs secure the fine textures; 2). the diversity of GANs with different effective patch sizes helps to generate diversified images with different aspect ratios and arbitrary sizes. As shown in Fig. 3.2 is the framework of SinGAN with a multi-scale patch generator and multi-scale patch discriminator. During training, the parameters are frozen when the low-level GAN converges, and the model starts to train the higher-level GAN. The results from the lower level are upsampled and passed to the higher level, and then it is used as the input of higher level GAN with the

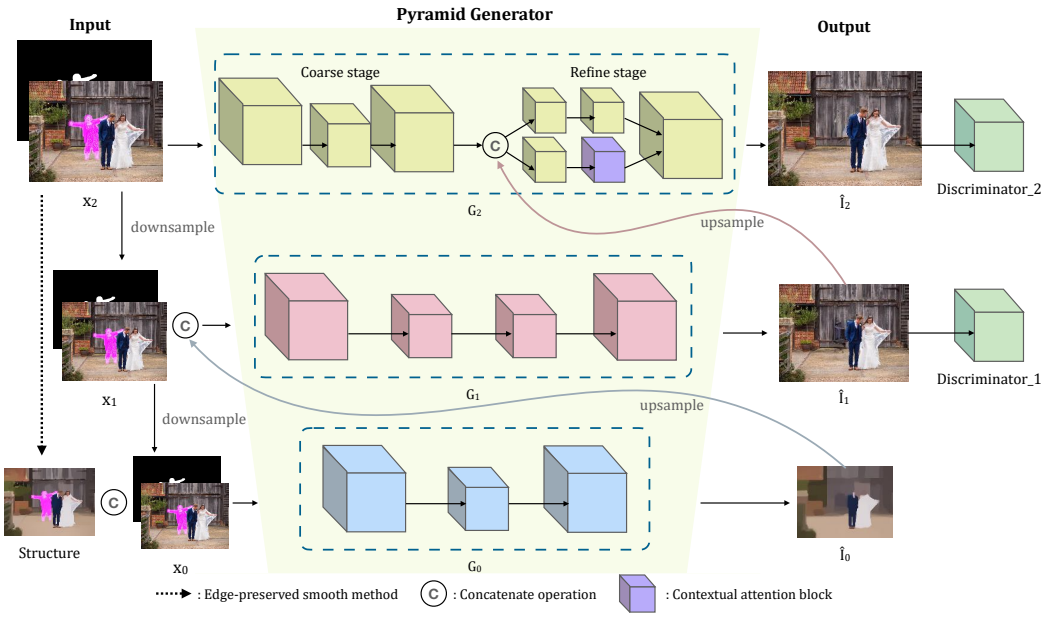


Fig. 3.3: Architecture of our pyramid generator. The input image and mask are gradually downsampled until to the bottom layer, while the structure is computed via edge-preserved smooth method [49]. Our model is trained in a bottom-up manner. The output of a lower layer will be fed to its higher layer to help refining the higher-layer's results. $\tilde{I}_2$ is regarded as the final output of our pyramid generator.

input noise. The generator adopts a residual structure where the input image is added to the generator output. The discriminator is patch-GANs with the same structure, and it is to capture the image information with different effective patch sizes, as seen on the right side of Fig. 3.2. The SinGAN model is straightforward and effective in generating images with reasonable structure and fine details with its proposed multi-scale system.

Pyramid Generator

Fig. 3.3 shows the architecture of our pyramid generator which are stacked from some sub-generators. We adopt the UNet-based structure in each sub-generators, and we replace each convolution layer with gated convolution as in Gated [6]. At different layers, the sub-generator is fed with inputs of different resolutions. To further focus on the structure reconstruction, we train the bottom layer with edge-preserved image [48, 49] instead of down-sampling of the masked image. StructureFlow [29] also adopted the same approach to train a structure reconstructor, by contrast, our bottom layer generates the "base" structure, and we pass it as a global guide to the higher layer for better structure generation. The generated image is rich in structure information with the combination of edge-preserved smooth methods and multi-layer hierarchy. To learn the image details in our highest layer, we set the original images with masks in the highest resolution as inputs. We apply the coarse-to-fine architecture from [6] in our texture generator. We

combine the coarse output with lower layer result as input of refining stage. For more discussion of our fusion strategy, please refer to 3.2.2. The contextual attention [6] is also adopted to learn long-range correlations.

We train the model with inputs in 512×512 resolution. The mask is a binary image, where 1 represents the corrupted area and 0 represents the available area. The original training image and mask are noted as I_2 and M_2, which are gradually downsampled to I_1, M_1 (with the resolution of 256×256) and I_0, M_0 (with the resolution of 128×128), respectively. During training, we generate the 128×128 structure image S via the edge-preserved smooth method [49].

The pyramid consists of $N+1$ sub-generators G_0, ..., G_N. Their inputs are x_0, ..., x_N, where each x_n is a concatenation of a masked image and a mask, except for the bottom layer $x_0 = \{I_0, S, M_0\}$. In this section, we adopt a three-layer architecture (i.e., $N = 2$) since it is enough to inpaint high-resolution images.

In the pyramid, each layer has a sub-generator G_n and a corresponding discriminator D_n. The model is trained in a bottom-up manner: 1) G_0 is trained firstly by using x_0. Let $\tilde{I_n}$ indicate the recovered image of each layer. Thus, we have the output of bottom layer,

$$\tilde{I_0} = G_0(x_0). \tag{3.1}$$

2) The output $\tilde{I_0}$ is then upsampled and fed to sub-generator G_1, meanwhile x_1 is also fed to G_1. Both of them are used to train G_1. The training of G_2 is the same as G_1. Notably, we have different fusion strategies among our pyramid, see more details in the following subsection. As a result, we have the outputs,

$$\tilde{I_n} = G_n(x_n, Upsample(\tilde{I}_{n-1})), n > 0. \tag{3.2}$$

Note that $\tilde{I_2}$ is regarded as the final output of our pyramid generator.

3.2.2 Fusion Strategy

In our pyramid generator, the output of a lower layer will be fed to a higher layer to refine the higher layer's results. For example, the output of G_0 will be fed to G_1 to refine the results of G_1. There are many possible fusion choices. The fusion could be conducted at the feature level or image level. Feature level fusion is similar to [69] that the feature maps of G_0 is upsampled and added to the corresponding feature maps of G_1. In contrast, the image-level fusion is to upsample the output image of G_0 and concatenate it with the input of G_1. On the other hand, since there are two stages in the highest G_{N-1} (i.e., coarse-stage and refine-stage), the output of the lower level could be fed to either the coarse-stage or the refine-stage of G_{N-1}.

We have conducted extensive experiments to compare many fusion options and selected the best one according to experimental results. Specifically, we adopt the refine-stage as well as image-level fusion strategy. As shown in Fig. 3.3, the output image of G_0 is upsampled and concatenated with I_1 and M_1 as the input of G_1. This process can be formulated as $\tilde{I}_1 = G_1(x_1, upsample(\tilde{I}_0))$. As for the next fusion, $\tilde{I}_2 = G_2^r(G_2^c(x_2), upsample(\tilde{I}_1))$, where G_2^c and G_2^r indicate the coarse and refine stage of G_2, respectively. We upsample the G_1 output $\tilde{I}_1$ and concatenate it with G_2 coarse stage output. The concatenation is fed to the refine stage for final inpainting.

The detailed comparisons of those options are shown in Section 3.3.4. We argue that refine-stage fusion being better than coarse-stage fusion is due to that the high-frequency details of high-resolution image (e.g. x_2) can be sufficiently exploited by the G_2^c before conducting fusion, and the validated results are shown in Section 3.3.4. By receiving both the output of G_2^c and G_1, the refine network G_2^r could properly leverage the good results of image details modeling (by G_2^c) and image structure modeling (by G_1).

Regarding to the feature-level fusion, we found that it is either inferior to image-level fusion or relatively difficult to stably achieve convergence. Besides, we also try another option that is to directly drop the coarse-stage G_2^c and only remain refine-stage G_2^r. Although it can reduce some computation, it is also difficult to stably get convergence.

3.2.3 Adaptive Dilation

Dilated convolutions are broadly used in inpainting neural networks since they can not only explicitly adjust the filter's field-of-view but also keep the resolution of feature maps [34]. Keeping the resolution of feature maps is very important to pixel-level algorithms such as image inpainting and image segmentation.

On the other hand, adjusting the filter's field-of-view is conducted by adjusting the *dilation rate* of dilated convolution layers, which will determine the receptive field of each layer. However, there is a conflict in modeling image global structures and image local details: modeling image structures often prefers to observe large image regions, while modeling image details prefers to observe small image patches. Therefore, it is tough to select a proper *dilation rate* for a single-scale architecture to balance the distinct needs of modeling image structures and details.

Our pyramid generator with several independent sub-generators has a superior advantage to designing distinct dilation layers on each layer for the dilemma. We adopt the dilation structure from [34] as our base design. Specifically, G_0 emphasizes modeling structures under a relatively small resolution. Hence we have three dilated convs with dilation rates $\{2, 4, 8\}$. In contrast, G_1 emphasize modeling image global information has dilated convs with dilation rates $\{2, 4, 8, 16\}$. At last, we have five dilated convs with the

dilation rates $\{2, 4, 8, 12, 16\}$ on G_2 for optimization of learning textures. In this way, we can effectively solve the conflict of modeling image global structures and local details within each layer, and we call this adaptive dilation scheme in this section.

3.2.4 Learning of Pyramid Generator

In this section, we describe the loss function and the training details of our pyramid generator. Our model is composed of several sub-generators, where the pyramid is trained by using GAN mechanism. Notably, for the top two layers with image-level supervision, we adopt the similar loss function as [6], which consists of a reconstruction term and an adversarial term,

$$\mathcal{L}_n = \mathcal{L}_{adv}(G_n, D_n) + \alpha \mathcal{L}_{re}(G_n), n > 0, \tag{3.3}$$

and we choose $\alpha = 1$ in our experiments.

The reconstruction term is defined as the $\mathcal{L}_1$ distance between the generated output $\tilde{I}_n$ and the ground-truth image I_n at the pixel level:

$$\mathcal{L}_{re}(G_n) = ||\tilde{I}_n - I_n||_1, n > 0. \tag{3.4}$$

For the adversarial term, we use the hinge loss. The loss for generator is

$$\mathcal{L}_G = -\mathbb{E}_{z \sim p_z, y \sim p_{data}} D\left(G\left(z\right), y\right), \tag{3.5}$$

and for discriminator, it is

$$\begin{aligned} \mathcal{L}_D = & \mathbb{E}_{x \sim p_{data(x)}} ReLU\left(1 - D\left(x\right)\right) \\ & + \mathbb{E}_{z \sim p_{z(z)}} ReLU\left(1 + D\left(G\left(z\right)\right)\right). \end{aligned} \tag{3.6}$$

The architecture of discriminators is identical to each other, as in [70]. Note that those discriminators are independently trained in our approach. While in the bottom structure reconstruction layer, we found that the adversarial term does not contribute to the overall training. Thus, we use Structure $\mathcal{L}_1$ loss to penalize the smooth image:

$$\mathcal{L}_{re}(G_0) = ||\tilde{I}_0 - S||_1, \tag{3.7}$$

where S and $\tilde{I}_0$ are input corrupted structure and output inpainted structure, respectively.

Taking the three-layer pyramid generator as an example, its total loss can be represented by

$$\mathcal{L}_{PG} = \lambda_0 \mathcal{L}_0 + \lambda_1 \mathcal{L}_1 + \lambda_2 \mathcal{L}_2, \tag{3.8}$$

where $\mathcal{L}_0$, $\mathcal{L}_1$ and $\mathcal{L}_2$ indicate $\mathcal{L}_{re}(G_0)$, the loss of G_1 and G_2, respectively. The λ_0, λ_1, λ_2 are the weights for them. In practice, we select the weight values empirically according to experimental results, i.e., we set $\lambda_{0,1,2} = 1$, which reaches the best performance.

A significant difference between our pyramid and SinGAN [70] is that SinGAN is trained layer by layer, but all sub-generators in our pyramid are *jointly* trained. We have tried the layer-by-layer training scheme for our model, i.e., we first train G_0 and then train G_1 while fixing the parameters of G_0. Experimental results show that it is relatively tricky for G_1 to achieve convergence with this layer-by-layer training scheme stably. We will try to find out the underlying reason for such phenomena in future work.

3.3 Experiments

We conduct experiments on Places2 [72], CelebA-HQ [53], and DIV2K [58] to evaluate our approach. The official train+val splits of Places2 are used to train our object-inpainting model, where each image is randomly cropped into 512×512 resolution. The $28,000$ images in CelebA-HQ are used to train our face-inpainting model, where all images are resized into size 512×512 as in [73].

Although our two inpainting models are both trained on 512×512 images, they can be used to restore an image in any resolution. Besides evaluating our approach on 512×512 testing images, high-resolution testing image are also involved into the evaluation. Particularly, we choose DIV2K dataset as our high-resolution validation dataset, which consists of 1000 images from the Internet in 2k resolution. There total data are divided into 800 train, 100 val and 100 test, respectively. These images have diverse contents in nature and are suitable for evaluations. Note that the images of DIV2K are randomly cropped into size 512×512 and 1024×1024 for evaluation, since it is memory-consuming to conduct image inpainting on 2k resolution.

We adopt the original mask generation algorithm in [6], which generates center square mask plus random free-form masks for each training sample. Our model adopted similar 3×3 gated conv in the encoder-decoder structure. In the pyramid, we downsample the input feature twice via convolution, and upsample twice via deconvolution. There is no normalization applied in our network. All of our experiments are trained with TensorFlow v1.15, CUDA v10.0. The final model needs 4 days to converge on two NVIDIA 2080Ti GPU with batch size of 2. We apply Adam as our optimizer with initial learning rate at $1e-4$.

During the training, we validated a small set of images from Places2 in the late stage to find the best checkpoint model. We applied random masks to these validations and chose the best performed on metrics (*i.e.*, PSNR, SSIM).

Table 3.1: Quantitative comparisons on the Places2 val set (with image resolution 512×512). We choose commonly used free-form mask validation settings from [26]. Up-arrow (↑) indicates higher score is better, while lower score is better for down-arrow (↓). The symbol † indicates that the model is retrained by us on 512×512 images instead of 256×256 images.

Mask ratio	10-20%			20-30%			30-40%			speed
Metrics	SSIM↑	PSNR↑	L1↓	SSIM↑	PSNR↑	L1↓	SSIM↑	PSNR↑	L1↓	time/image
EC [10]	0.9464	26.96	0.0336	0.8748	23.22	0.0483	0.8167	22.07	0.0544	178ms
PEN [9]	0.9168	25.70	0.0172	0.8390	22.47	0.0333	0.7596	20.70	0.0492	347ms
Gated [6]	0.9025	24.91	0.0123	0.8884	23.78	0.0235	0.8434	22.46	0.0362	69ms
Gated† [6]	0.9506	28.54	0.0219	0.8971	24.82	0.0242	0.8363	22.54	0.0368	69ms
HiFill [7]	0.9049	25.04	0.0219	0.8261	22.09	0.0384	0.7488	20.39	0.0543	24ms
ours	**0.9541**	**28.96**	**0.0123**	**0.9036**	**25.25**	**0.0233**	**0.8460**	**22.96**	**0.0353**	85ms

Table 3.2: Quantitative comparisons on the Div2k val set (with image resolution 512×512). We conduct experiments on square center masks and free-form masks. The free-form mask is generated by algorithms from [8].

Method	Mask type	SSIM↑	PSNR↑	L1↓
EC [10]	Center	0.7734	20.33	0.0663
PEN [9]		0.7541	19.36	0.0563
Gated [6]		0.7668	19.23	0.0535
Gated† [6]		0.7656	18.88	0.0544
HiFill [7]		0.7394	19.52	0.0563
ours		**0.7808**	**20.88**	**0.0459**
EC [10]	Free-form	0.6706	18.00	0.0937
PEN [9]		0.7185	19.66	0.0642
Gated [6]		0.7235	18.45	0.0715
Gated† [6]		0.7323	19.62	0.0633
HiFill [7]		0.6917	18.38	0.0763
ours		**0.7403**	**20.07**	**0.0611**

3.3.1 Comparison to Other Methods

In this section, we compare our model with methods EdgeConnect (EC) [10], Gated [6], PEN [9], and HiFill [7]. The original EdgeConnect and PEN are trained with 256×256 images, while HiFill is trained with 512×512 images. For more comparisons, we also re-trained Gated on 512×512 images, noted as Gated†.

For numerical comparisons, we evaluate those methods on $\mathcal{L}_1$ loss, structural similarity index measure (SSIM) [37], and peak signal-to-noise ratio (PSNR). To calculate the inference time per image, we test all the images in Places2 validation set with center masks on single NVIDIA GTX 2080Ti GPU. We compare our method with others on image with different scales, and we split the validation results into 512×512 and 1024×1024 to show the results on different resolutions.

(a) Masked (b) EC [10] (c) PEN [9] (d) Gated [6] (e) Gated† [6] (f) HiFill [7] (g) Ours

Fig. 3.4: Qualitative comparisons on Places2 val and DIV2K val set (with image resolution 512×512). † indicates methods retrained by us with 512×512 images. Best viewed by zooming-in.

Comparisons on 512×512 Images

Table 3.1 and Table 3.2 are the comparison of those methods on the datasets Places2 and DIV2K (with images in 512×512 resolution). We could find that our approach outperforms all the other methods on both the free-form mask inpainting and the center mask inpainting.

Figure 3.4 shows the visual comparison for examples of resolution 512×512 from dataset Places2 and DIV2K. From the results, we could find EdgeConnect prones to producing color inconsistent and blurry content. The original Gated could better recover con-

(a) Masked (b) PEN (c) Gated (d) Gated† (e) Ours

Fig. 3.5: Qualitative comparisons on CelebA val set (with image resolution 512×512).

tent on small holes, but it tends to produce artifacts when inpainting large holes. Moreover, PEN is not good at generating realistic high-frequency details. Probably because they are trained on 256×256 images, they cannot properly handle 512×512 inpainting task.

On the other hand, Gated†, HiFill and our model are all trained on 512×512 images, and there is an obvious performance improvement. However, the restored content from Gated† is not very coherent to their surrounding pixels, and HiFill is not good at recovering image global structures. In contrast, our approach can not only recover image global structures but also produce coherent image details.

The results on CelebA-HQ are shown in Figure 3.5. Compared to the vanilla Gated, Gated† could properly synthesize some face parts (e.g. eyes). Nevertheless, it still cannot model the global structures of a face.

Comparisons on 1024×1024 Images

Table 3.3 illustrates the comparison of high-resolution image inpainting (with images resolution 1024×1024). On the free-form mask inpainting, our approach outperforms Gated† and HiFill.

The results of these methods on high-resolution image inpainting are shown in Figure 3.6. Obviously, it is much more challenging to generate coherent and realistic image details on 1024×1024 images. Both Gated† and HiFill struggles to generate plausible border of the mountain. At the same time, our approach's inpainting results look more relevant and realistic.

Table 3.3: Quantitative comparisons on 1024×1024 images from DIV2K val set. We apply free-form masks for validation.

Method	free-form mask		
	SSIM↑	PSNR↑	L1↓
Gated† [6]	0.8622	22.39	0.0341
HiFill [7]	0.7691	19.98	0.0536
ours	**0.8676**	**22.76**	**0.0329**

(a) Masked (b) Gated† (c) HiFill (d) Ours

Fig. 3.6: Qualitative comparisons on on 1024×1024 example from DIV2K dataset with zoomed-in results.

3.3.2 Large-hole Inpainting

It is known that when the corrupted area is large, the image inpainting becomes much more challenging. As shown in Figure 3.4, when the hole size becomes large, the center of the hole tends to have color discrepancy and blurriness.

Recent work found that such issues could be alleviated by adopting the progressive learning strategy [63], without the needs of modifying the generative model itself. The progressive learning strategy is to divide the training procedure into several stages so that the small-hole training data are first used to train the model at early stages whereas large-hole training data will be used to fine-tune the model at later stages. Even if it is effective sometimes, such strategy still has some limitations. For example, it is hard to decide when to move from one stage to the next stage.

As aforementioned, we have illustrated that our pyramid generator could not only align to the progressive learning strategy but also avoid the need for explicitly separating the training procedure into several stages. In this section, we will evaluate our method by gradually increasing the hole size and show its advantage on large hole inpainting.

Specifically, the center mask setting is adopted in this experiment, since it is easy to control the ratio of hole size to the image size. We gradually increase such ratio from

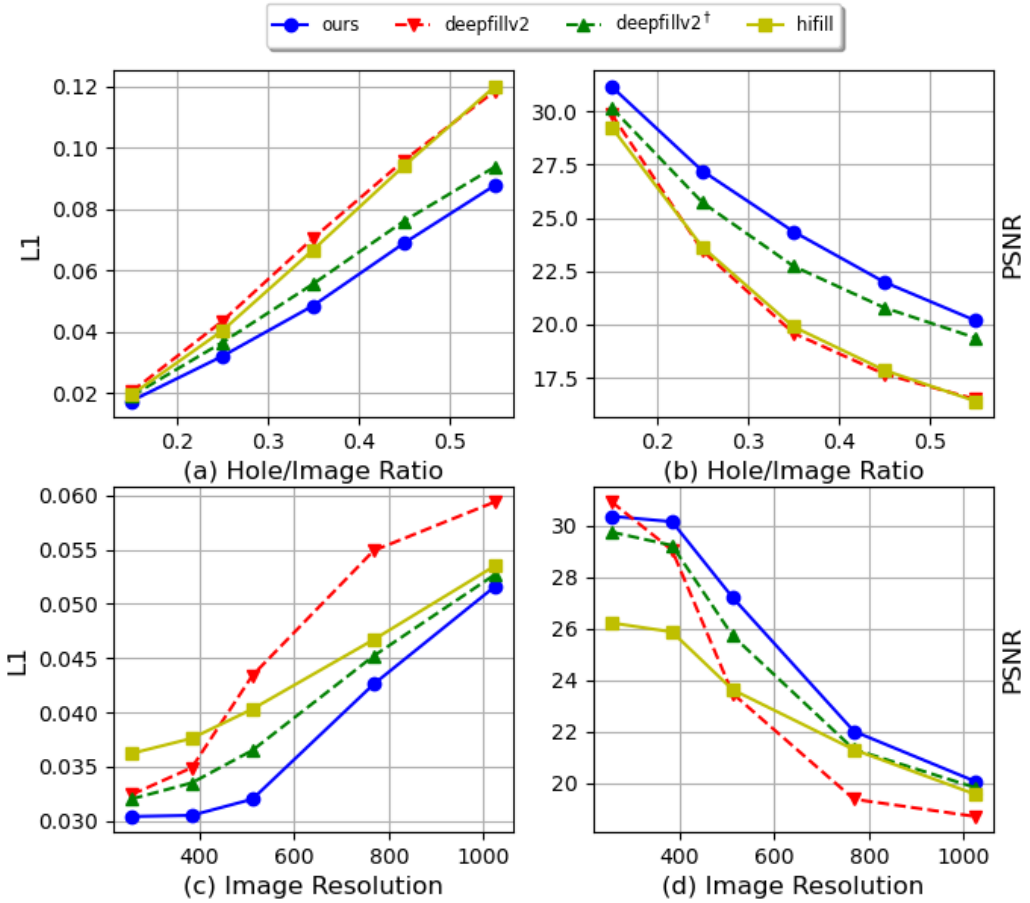


Fig. 3.7: The performance degradation with respect to the increasing of hole size (a&b) and image resolution (c&d). With the ratio of hole size increased from 15% to 55%, the degradation magnitude of our model is much less than that of Gated [6]. With the image resolution increased from 256 to 1024, our model could better cope against the performance degradation, and hence our model is more suitable for high-resolution image inpainting. All the experiments are on DIV2K val set with square shape masks.

15% to 55% with interval of 10%. As shown in Figure 3.7 (a) & (b), the red dash line is the performance of Gated [6] and the blue line indicates our model performance. There is a clear performance degradation for both methods when the ratio increases, but the degradation magnitude of our model is much less than that of Gated.

Besides, it is obvious that increasing resolution of training images also benefits large-hole inpainting task (e.g. Gated† outperforms Gated), but it is still inferior to our approach.

3.3.3 High-resolution Image Inpainting

With the rapid development of high quality camera devices, the images we meet in daily life are often high-resolution. Thus, the real-world inpainting task is often about high-resolution image inpainting.

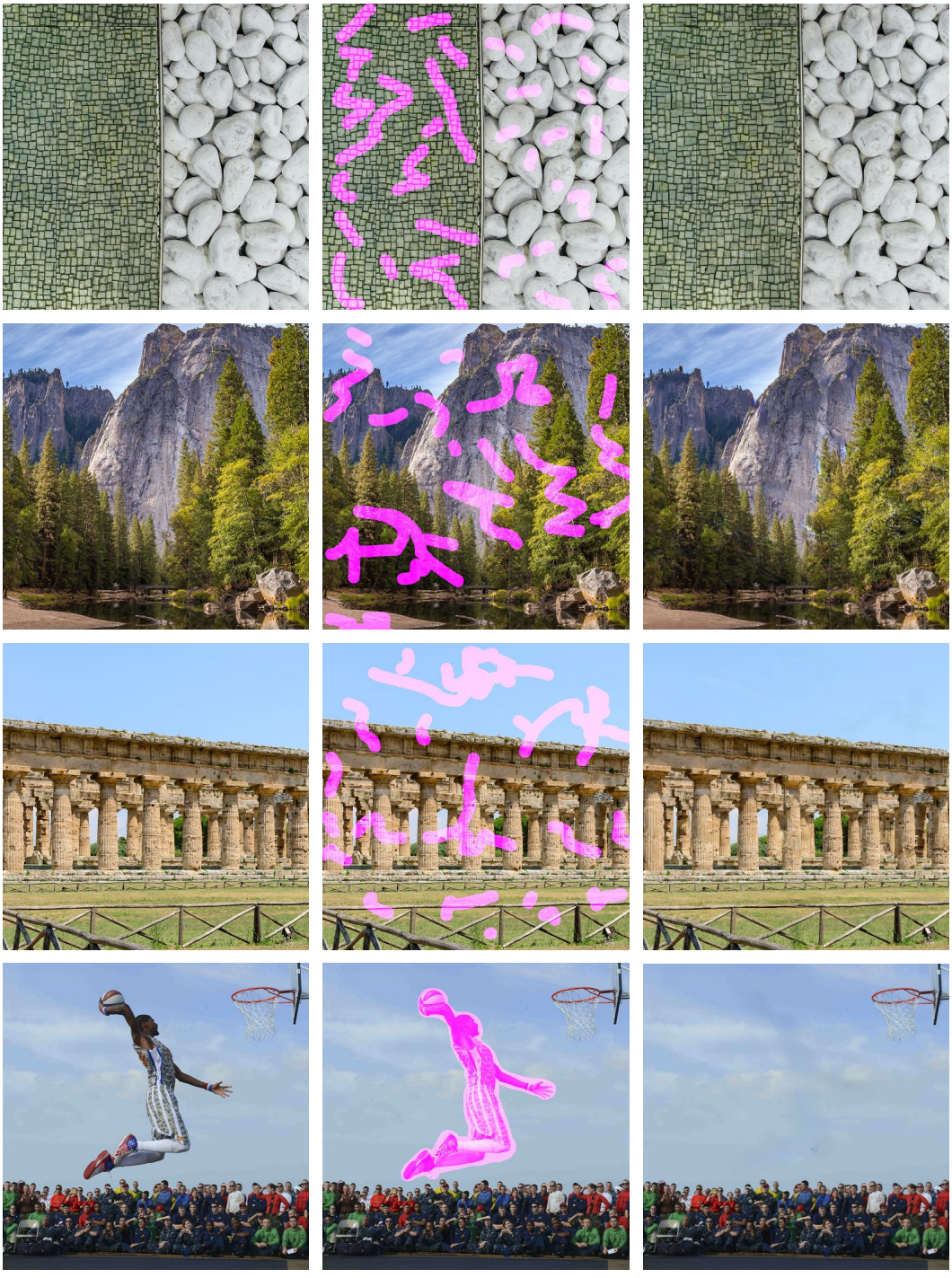
On the other hand, many previous works have demonstrated that increasing the resolution of training images is critical to the inpainting performance, especially for the high-resolution image inpainting task [7, 9].

In this section, we conduct some experiments to study such observations. Specifically, we train the same model Gated [6] with images of different resolution: the original Gated is trained with images of 256×256 resolution, while the model Gated† is trained with images of 512×512 resolution. From Figure 3.7 (c) & (d), we can see that the inpainting performance could be improved by increasing the resolution of training images, which validates the conclusion of previous work.

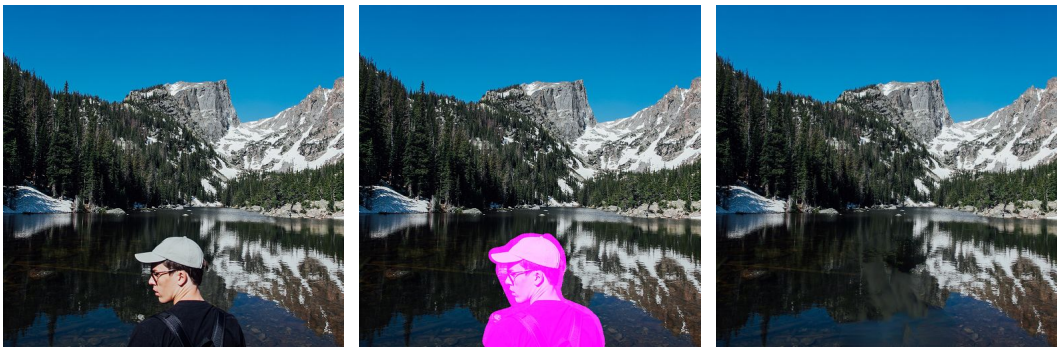
Besides, Figure 3.7 (c) & (d) tells us that the inpainting performance will decrease when the resolution of testing image is increasing. This indicates that the high-resolution image inpainting task is much more difficult than low-resolution inpainting task. Moreover, it is worth noting that such performance degradation is different between the model Gated and Gated†: the Gated† could better cope against the performance degradation than the Gated, i.e., the green line drops slower than the red line according to the metric PSNR. Therefore, it is critical to use high-resolution images as training data.

More importantly, we also evaluate our pyramid generator (noted as blue line in Figure 3.7 (c) & (d)). Compared to the model Gated† and HiFill [7], we can see that our model outperforms those methods, although they are all trained with images of same resolution (i.e., 512×512). It indicates that our model could fully exploit the benefits of learning with high-resolution images, and hence our model is good at high-resolution image inpainting.

More high-resolution inpainting results can be found in Fig. 3.8.



Original Masked Ours

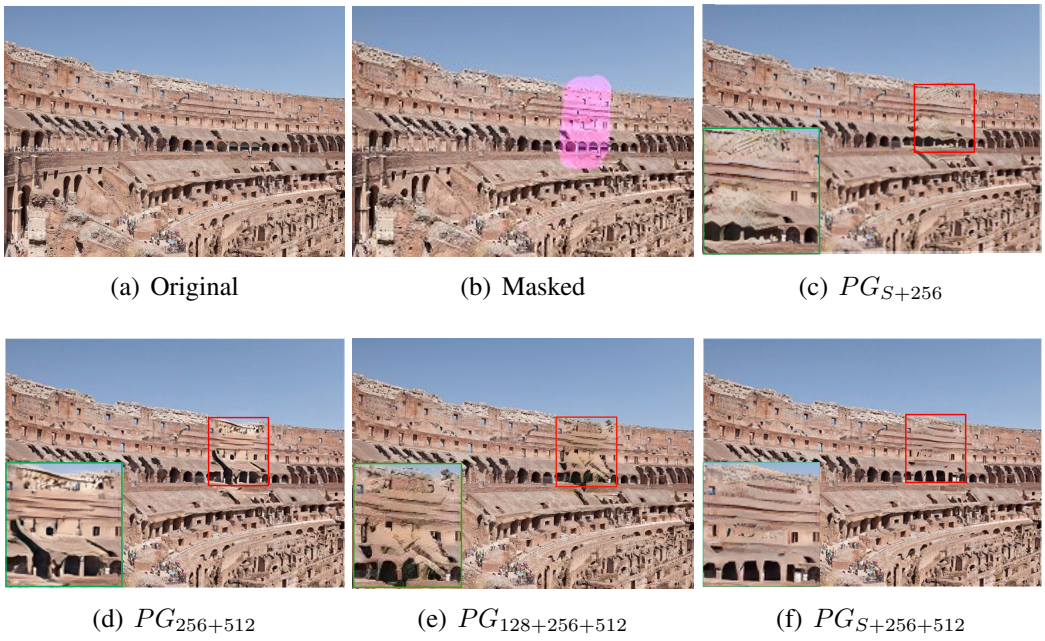


Original Masked Ours

Fig. 3.8: Visual examples on DIV2K with 1024×1024 resolution. Better to view by zooming-in.

3.3.4 Ablation Study

Except the recent architecture of our pyramid generator described in Section 3.2.1, there are many other design options. We have evaluated those design options and will discuss their performance in this section. Such evaluation are conducted on DIV2K val dataset.



(a) Original (b) Masked (c) PG_{S+256}

(d) $PG_{256+512}$ (e) $PG_{128+256+512}$ (f) $PG_{S+256+512}$

Fig. 3.9: Qualitative comparisons between our final model and other variants.

How many layers should we have?

Since our pyramid generator has a multi-layer architecture where each layer corresponds to a specific image scale, we could have many choices for the number of layers. Apart from our current implementation of three layers with structural layer, we also try two variants of two-layer architecture: one variant is the combination of the structure layer and

Table 3.4: Numerical comparisons between our three-layer model and other variants.

	SSIM↑	PSNR↑	L1↓
PG_{S+256}	0.809	24.07	0.042
$PG_{256+512}$	0.827	26.03	0.035
$PG_{128+256+512}$	0.827	26.47	0.035
$PG_{S+256+512}$ (Ours)	**0.842**	**27.20**	**0.032**

256-layer (noted as PG_{S+256}), the other variant is the combination of 256-layer and 512-layer (noted as $PG_{256+512}$). Nonetheless, to better validate the effectiveness of our bottom structure branch, we also test results on another three-layer variant ($PG_{128+256+512}$) with 128×128 images as inputs of the bottom layer.

The comparisons of our final three-layer model and the other variants are illustrated in Table 3.4 and Figure 3.9. It shows that the other variants have distinct drawbacks. The model PG_{S+256} tends to generate blurry content due to its disadvantage on modeling image high-frequency details. Secondly, the model $PG_{256+512}$ tends to generate incoherent image structures because of the lack of image global information through the structure layer. While the three-layer model $PG_{128+256+512}$ with images on the bottom layer also failed on the structure generation. In contrast, our final three-layer model $PG_{S+256+512}$ enjoys the advantages of all the variants, and could generate both coherent image structures and realistic image details.

Table 3.5: Quantitative comparisons on different fusion strategies.

	feature-level		image-level		
	coarse	refine	att. path	non-att path	**ours**
SSIM↑	0.786	-	0.825	0.833	**0.842**
PSNR↑	23.16	-	25.94	25.96	**27.20**
L1↓	0.044	-	0.036	0.036	**0.032**

How to fuse different layers?

As mentioned above, there are many possible fusion options, and we compare them in this section. First, we consider feature-level fusion, which can be further categorized as coarse-stage fusion (i.e., fuse with G_2^c) and refine-stage fusion (i.e., fuse with G_2^r). In the feature-level fusion, we fuse the low-level encoders' features with higher level ones through upsampling and concatenation. Next, we consider image-level fusion. Besides

the fusion scheme adopted in our model, there are additional two options. Since there are two paths for the G_2^r (i.e., attention path and non-attention path), we can feed the output of G_1 to one path only. From Table 3.5, we can see that feature-level refine-stage fusion cannot converge, and our final scheme with image-level fusion is the best among them.

Table 3.6: Quantitative comparisons between the standard and our adaptive dilation scheme.

	SSIM↑	PSNR↑	L1↓
standard dilation	0.829	26.38	0.035
adaptive dilation (Ours)	**0.842**	**27.20**	**0.032**

Is adaptive dilation effective?

Due to the pyramid structure, our approach could adopt the *adaptive dilation* mechanism, which means the sub-generators at different layers could have different number of dilation layers, and different dilation rate configurations. In this section, we use the baseline that all sub-generators have same number of dilation layers and same dilation rate configurations. In particular, it has 4 dilation layers with dilation rates $\{2, 4, 8, 16\}$, which is the same as [6]. In contrast, in our approach G_0 has three dilation layers with dilation rates $\{2, 4, 8\}$, while four in G_1 with dilation rates $\{2, 4, 8, 16\}$. The top layer G_1 has five layer model with dilation rates $\{2, 4, 8, 12, 16\}$. As shown in Table 3.6, we find that our adaptive dilation mechanism benefits the performance improvement compared to the baseline method.

3.3.5 Failure Cases Analysis

Although the proposed pyramid generator could eliminate artifacts and blur current methods, some failure cases exist. In some complicated scenarios, there is a need to understand the global semantics and synthesis of suitable pixels based on their surrounding known areas, which requires the model to have high-level recognition ability. As shown in the first example in Fig. 3.10, the proposed method failed to generate reasonable leaves and tree trunks. Also, when inpainting the image with texts, it is easy to introduce more artifacts or unmeaningful fillings, as shown in the second example in Fig. 3.10, and that requires the system can address the texts dedicatedly.

(a) Masked (b) Original (c) Ours

Fig. 3.10: Failure cases of the proposed pyramid generator.

3.4 Summary

This chapter proposes a "structure first detail next" workflow for image inpainting. In particular, we introduce a Pyramid Generator by stacking several sub-generators, where image global structures and local details could be better separately modeled at different pyramid layers. Notice that our approach has a progressive learning scheme that allows it to restore images with large masks. In addition, our model is suitable for inpainting high-resolution images. Extensive experiments show that our approach outperforms the other state-of-the-art methods. For future work, we aim at improving our model for 2K/4K image inpainting on two aspects. Firstly, we focus on reducing the current network parameters for the larger resolution inference. Secondly, we believe it is reasonable to follow our current multi-scale philosophy on ultra-resolution image inpainting, and our target is to build a better model that boost the advantage of multi-scale design.

Chapter 4

Image Inpainting with Attention Feature Fusion

Recent deep generative inpainting models are usually built in a two-stage manner. On the first stage, they either generate coarse output [6] or other auxiliary guidance (e.g. edge [10], semantic map [28], smoothed image [29]). These models borrow previously learned information in the second stage and generate the final refined outputs. Two-stage methods can separate the learning of global structure and local texture. Thus, these methods advance the single encoder-decoder architectures to more robust and powerful feature representative.

Apart from two-stage approaches, another group of methods tries to model the structure and texture within feature space. Some representative work build structure & texture branches [27] , AOT block [12] and multi-scale attention module [7] on vanilla encoder-decoder features. Instead of progressively learning the structure and texture of different stages or pyramids, these works build novel modules to enhance the learned features with more accurate structure and texture.

However, these two categories of methods either introduce external information or complex modules which consume more computation. This chapter investigates the efficient representative enhancement from a feature fusion perspective. The Attention Feature Fusion (AFF) [74] has recently shown advances in fusing different levels of features with inconsistent semantics. Motivated by this observation, we build our model on top of the existing two-stage method and adopt the AFF block to better model the structure and texture information in two parts. First, we build Skip Connection Attention Fusion (SCAF) which uses AFF to fuse the texture information from the encoder with semantics from the decoder. The aggregation is then passed to the decoder through skip connections. Second, to further strengthen the vanilla dilated convolutions architecture which is vital to capture the global structure and multi-scale objects on inpainting, we build Dilated Convolution Residual (DCR), which could better fuse the features with multi-receptive fields. With

the help of SCAF and DCR, our model can better understand global and local information with an image and could alleviate the blur and artifacts of the current model. Our contribution of this section can be concluded as:

- To alleviate the blurry results and artifacts caused by the lack of understanding of image global and local information, we propose a feature fusion-inspired method;
- We apply Attention Feature Fusion to better model the vanilla two-stage encoder-decoder features through Skip Connection Attention Fusion and Dilated Convolution Residual;
- The quantitative and qualitative results on Places2 and Celeba-HQ have shown that our model outperforms other state-of-the-arts.

4.1 Feature Fusion in Computer Vision

The feature fusion technique has been studied extensively as it is a powerful strategy to improve the representation of neural networks [75–77]. In the InceptionNet series [61, 75, 78], they employed a parallel of convolutions to extract features with various sizes and fused them to obtain the output feature. The fundamental of the popular ResNet [38] also adopted feature fusion. They fuse the identity mapping feature with the residual learned feature with the addition operation, and this technique is effective in the network degradation problem. In object detection, the Feature Pyramid Networks (FPN) [76] have shown that the fusion of the low-level and high-level features could generate high-resolution features with strong semantics. In semantic segmentation, the skip connection is also a form of feature fusion, and it has shown the effectiveness of fine-grained segmentation (e.g. U-Net [77]).

Recently, feature fusion has focused on leveraging the feature pyramid and multi-scale features for enhanced representatives. HRNet [79] build a network with pyramids on different scales, and they interweave these features through downsampling and upsampling. The parallel model uses repetitious fusion across multi-scale features and obtains a robust human pose estimation backbone.

Apart from fusing explicitly on multi-scale features, another branch of methods tries to use the inherent multi-scale of CNN through the attention mechanism. The attention mechanism firstly caught the spotlight on machine translation task [80], and it was quickly adopted in the computer vision field [81, 82]. Squeeze-and-Excitation Networks (SENet) reschedule the features along the channel through the learned channel-wise dependencies. Selective Kernel Convolution (SKNet) [83] proposed to build multi-scale features within an input through convolution in different kernel sizes.

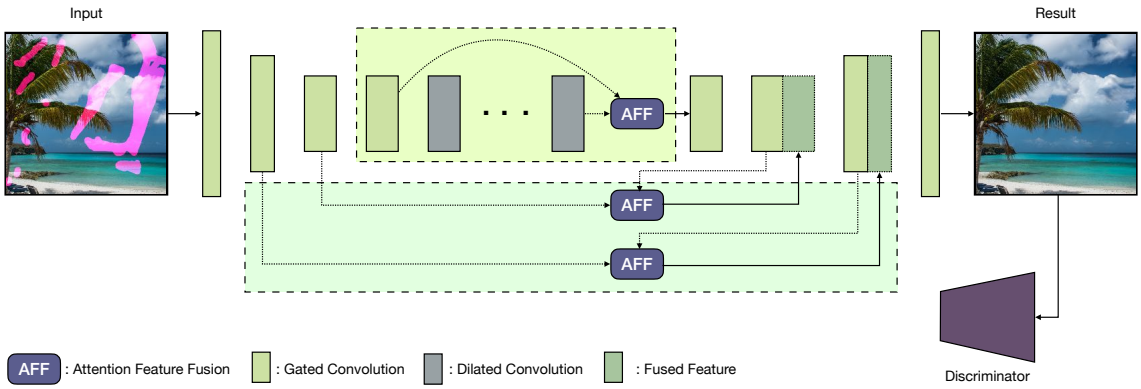


Fig. 4.1: Architecture of our feature fusion-based method. For simplicity, we only show the generator with single stage. The proposed Skip Connection Attention Fusion (SCAF) and Dilated Convolution Residual (DCR) is shown in the figure, and we apply Attention Feature Fusion (AFF) [74] within these two modules.

Feature fusion is not explicitly used in image inpainting. To eliminate the artifacts and blur of current models, researchers aim to build a model to fuse image structure and textures. Liu *et al.* [27] exploited different level features of the encoder to build structure and texture branches for further refinement. PEN-Net [9] built a feature pyramid to obtain the enriched features through a non-local module. HiFill [7] applied the multi-scale attention module to learn the contexts from multi-level features. Unlike the above methods, we focus on building a model with enriched features to model images' texture and structure better.

4.2 Rethinking Image Inpainting with Attention Feature Fusion

This section introduces our model design and the components in detail. Section 4.2.1 includes the overall architecture of our model. In Section 4.2.2, we first revisit the attention feature fusion layer and then highlight how we integrate this mechanism with our model from two perspectives. The optimization objectives are elaborated in Section 4.2.3.

4.2.1 Architecture

In a modern inpainting task, given a color image I multiplied with a binary mask m, the masked results can be represented as $I' = I \odot m$. This image I' is concatenated with the mask as the final input $x = concat(I', m)$. A typical inpainting model uses the generative adversarial network (GAN) that consists of a generator $f_{\theta^G}(\cdot)$ and a discriminator network $g_{\theta^D}(\cdot)$. The final inpainted image is denoted as $I_o = f_{\theta^G}(x)$. During training, each pair of inputs consists of one image from the selected dataset and one randomly generated mask.

The overall model is shown in Fig. 4.1 . We build our algorithms on top of the baseline model from [6]. This baseline is a coarse-to-refine model with an attention branch and dilation branch in the refining stage. In the first stage, they generate a coarse result that mainly focuses on the global structure within the image. This first stage output is then fed into the next stage for refining the details within the global completion. Both stages applied the U-Net architecture constructed by stacking multiple fully convolutional layers. The convolutional layers used in the model are gated convs, which have better results on free-from inpainting. The main problem of this architecture is the insufficient learning of global context and local details. We propose to address this problem by fusing features on different levels. According to the learning process of an encoder-decoder model, the low-level features from the encoder contain a more delicate texture, and the high-level features from the decoder have rich semantic structure. As shown in Fig. 4.1, we fuse the low-level feature to its corresponding high-level feature through skip-connection. We use the Attention Feature Fusion layer to guarantee this fusion without inconsistency. Meanwhile, dilated convolutions can bring more semantics. We further enhance the semantics aggregation by applying the Attention Feature Fusion in the multi-dilated residual learning, see Fig. 4.1.

More details about Attention Feature Fusion and the integration are introduced in the following part.

4.2.2 Feature Fusion Modules

In this section, we first revisit the attention feature fusion technique initially designed for image classification and semantic segmentation [74]. In the following part, we elaborate on how this feature fusion technique can benefit image inpainting and how we integrate it into our model from two perspectives.

Attention Feature Fusion

Feature fusion is a vital strategy to increase further the representation power of deep convolutional networks (DCNs). It has been an important research field and widely adopted into many popular models. Previous fusion strategies commonly exploit simple linear operations such as addition or concatenation. These strategies lack non-linear aggregation of features and are hence not optimal for representative learning. Inspired by the global channel attention mechanism [84] and [85], the proposed attention feature fusion framework is based on a novel Multi-scale channel attention mechanism (MS-CAM) [74]. Given two different features X and Y to be fused. The fusion process can be depicted as:

$$F = M(X + Y) \otimes X + (1 - M(X + Y)) \otimes Y. \tag{4.1}$$

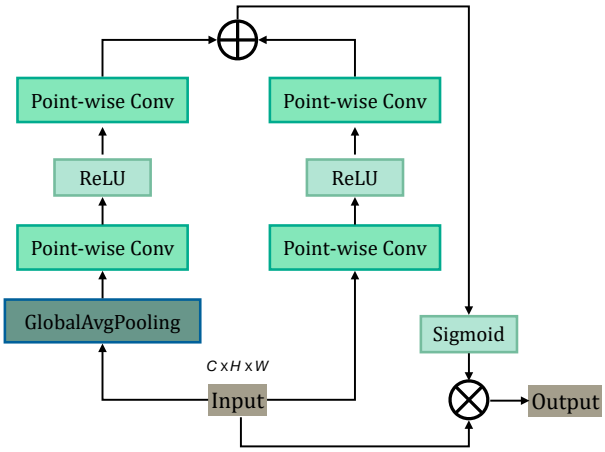


Fig. 4.2: The MS-CAM module [74].

Where M is the MS-CAM, the element-wise addition is processed on the two features, $\otimes$ denotes the element-wise multiplication. The MS-CAM combines the global and local features within the attention mechanism. It is constructed with point-wise convolutions (PWConv) by varying the spatial pooling size along the channel dimension. The architecture of MS-CAM is shown in Fig. 4.2. There are two branches within the module: the global channel context $\boldsymbol{g}(X) \in \mathbb{R}^{C\times 1\times 1}$ and the local channel context $\boldsymbol{L}(X) \in \mathbb{R}^{C\times H\times W}$. The global context is inspired by [84] and

$$G(X) = \mathcal{B}(PWC_2^g(ReLU(\mathcal{B}(PWC_1^g(GAP(X)))))), \tag{4.2}$$

where GAP is the global average pooling, $\mathcal{B}$ is the Batch Normalization (BN) [86]. The PWC is used for exploiting less parameters. For the local context, is can be depicted as:

$$L(X) = \mathcal{B}(PWC_2^l(ReLU(\mathcal{B}(PWC_1^l(X))))). \tag{4.3}$$

The output context has the same shape as the input feature, so the local information is well-preserved. For the input feature X, the feature after the attention computation $X' \in \mathbb{R}^{C\times H\times W}$ can be represented as:

$$X' = X \otimes W(X) = X \otimes Sigmoid(L(X) \oplus g(X)), \tag{4.4}$$

where $W \in \mathbb{R}^{C\times H\times W}$is the computed weights and $\oplus$ is the broadcasting addition.

Skip Connection Attention Fusion

Skip connection is widely used in many encoder-decoder models, such as semantic segmentation and object detection. This technique helps to pass long-range information from encoder to decoder [77]. Features from the encoder are rich in low-level texture, and as the network goes deep, these textures are attenuated by the successive convolutions and poolings. Meanwhile, the learned features from the decoder have rich high-level semantic context. Using skip connections could help to obtain a more robust feature with both low-level textures and high-level semantics and thus benefit the further tasks. The vanilla skip connection was proved that have a limited effect on the inpainting task as in [6]. The reason is mainly that a large part of masked regions is involved in the low-level features, and it is hard to extract helpful texture information (e.g. details, edges, colors) through the simple fusion operation (e.g. concatenation). More experimental results can be found in Section 4.3.3.

We propose to use AFF as our fusion strategy for its advantage in integrating features with long distances. As shown in Fig. 4.1, a feature from the encoder and its corresponding feature in the decoder are fed into the AFF module as two distinct inputs. We follow the fusion process mentioned above and generate the enhanced feature map with attention computed on both low-level and high-level features. Then, we concatenate this feature with the high-level input feature as the final decoder feature. Ultimately, we archive to compensate for the original decoder features with fine textures from the encoder.

Dilated Convolution Residual

At the same time, the dilated convolution is also a vital component of the image inpainting model. At first, Iizuka [34] proposed to use dilated convolution in image inpainting to increase the receptive field. Many following works find that more sophisticated dilated architectures could generate better results [6, 12]. In [6], they adopted a series of dilated convolution blocks with increasing dilation rates, which is as the same framework in semantic segmentation [44]. This dilation architecture could better model the multi-scale objects in the image and help generate a good semantic initial for the late decoder. We argue that for image inpainting, a more dedicated treatment is needed for a better fusion of these features with different receptive fields. Thus, we use AFF as a residual for the dilated block series. We found this simple addition help to improve the understanding of image global semantics and hence generates results with visually meaningful patches. Initially, the dilation blocks with dilation rates 2, 4, 8, and 16 are in a sequential arrangement. Inspired by residual learning [38], we use AFF to fuse these different receptive field features. The dilated convolution residual is implemented using the input of dilation blocks as identity mapping and the output of the last dilated convolution layer as learned residual. Finally, the two features are fused through the attention feature fusion module.

4.2.3 Training

We introduce the training details in this section. Our model is trained in the GAN fashion, including a generator and a discriminator. The generator is a commonly used encoder-decoder model described above, and we apply the patch discriminator [6] for stable and fast training. The objective function includes reconstruction loss $\mathcal{L}_{re}$ and adversarial loss $\mathcal{L}_{adv}$. The total loss function is:

$$\mathcal{L} = \mathcal{L}_{adv}(G, D) + \alpha \mathcal{L}_{re}(G), \tag{4.5}$$

where α is the hyperparameter for weight among different losses, we adopted $\alpha = 1$ in our experiments.

Reconstruction loss

For fast convergence and accuracy, we choose the $\mathcal{L}_1$ distance to guarantee that the reconstructed image is as similar as possible to the ground truth. The $\mathcal{L}_{re}$ can be calculated with input I and the generated output I':

$$\mathcal{L}_{re}(G) = ||I' - I||_1. \tag{4.6}$$

Adversarial loss

To further ensure the generated results have fine-grained details and semantically meaningful structure, we have trained our model on different losses: Wasserstein-GP GAN losses from [40] and SN-PatchGAN loss [6]. Finally, we choose to use SN-PatchGAN with a more stabilized training. In this section, we disscuss the details of the choice.

WGAN-GP loss is a well-known type of loss, especially on image generation tasks. Based on WGAN (Wasserstein GAN) [39], which uses the Earth-Mover (Wasserstein) distance $W(\mathbb{P}_r, \mathbb{P}_g)$ to measure the difference between the real data and generated samples. The Wasserstein distance for the real data $\mathbb{P}_r$ and the generated samples $\mathbb{P}_g$ can be defined as:

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \prod(\mathbb{P}_r, \mathbb{P}_g)} E_{(x,y)\sim\gamma}[||x - y||], \tag{4.7}$$

where $\prod(\mathbb{P}_r, \mathbb{P}_g)$ is set of all joint distributions $\gamma(x, y)$ whose marginals are $\mathbb{P}_r$ and $\mathbb{P}_g$, x and y are samples from this joint distribution $\gamma(x, y)$.

By using Kantorovich-Rubinstein duality, the overall loss function of WGAN can be expressed as:

$$\min_G \max_{D \in \mathcal{D}} E_{x\sim\mathbb{P}_r}[D(x)] - E_{G(z)\sim\mathbb{P}_g}[D(G(z))], \tag{4.8}$$

where z is the input of generator, $\mathcal{D}$ is the set of 1-Lipschitz functions.

Compared with two commonly used divergences (KL-Divergence and JS-Divergence) in GANs, the Earth-Mover distance is ensured to have a smoother gradient everywhere. An improved version of WGAN with gradient penalty (GP) is proposed, where the GP term is

$$\lambda E_{\hat{x}\sim\mathbb{P}_{\hat{x}}}(||\nabla_{\hat{x}}D(\hat{x})||_2-1)^2, \tag{4.9}$$

$\hat{x}$ is the points which sampled from the straight line between the distributation $\mathbb{P}_r$ and $\mathbb{P}_g$, λ is the hyperparameter.

This term can be extended to image inpainting with only masked regions evolved. The modified version with gradient penalty multiplied with mask region can be shown as:

$$\lambda E_{\hat{x}\sim\mathbb{P}_{\hat{x}}}(||\nabla_{\hat{x}}D(\hat{x})\odot(\mathbf{1}-\boldsymbol{m})||_2-1)^2, \tag{4.10}$$

where $\boldsymbol{m}$ is the input mask, value equals 0 means the missing pixel, and 1 indicates the known pixel.

Although the WGAN-GP loss results in stable training, we found it may not works best on image inpainting tasks. The reason is that the generated results of the inpainting task often contain many holes, and there should be an evaluation focus on the quality of the local area, especially when inpainting on irregular masks with arbitrary shapes, the WAGN type of loss is not efficient. Inspired by MarkovianGAN (PatchGAN) [87], where the discriminator learns the semantics and contexts of these Markovian patches from the generated inputs. The output of the discriminator is based on applying GANs on each "patch" of the last feature map and predicting the final results. These patches are in different locations and could evaluate the arbitrary mask area. Nonetheless, we also add spectral normalization (SN) [88] on the PatchGAN and the resulting SN-PatchGAN as our final GAN loss. The spectral normalization could stabilize the training and thus cost a short time during the training, and we can briefly express this process as:

$$\bar{W}_{SN}(W):=\frac{W}{\sigma(W)}, \tag{4.11}$$

where $\sigma(W)$ is the spectral norm of weight matrix W.

The overall adversarial loss can be formulated as:

$$\mathcal{L}_G=-\mathbb{E}_{z\sim p_z,y\sim p_{data}}D^{sn}\left(G\left(z\right),y\right), \tag{4.12}$$

and

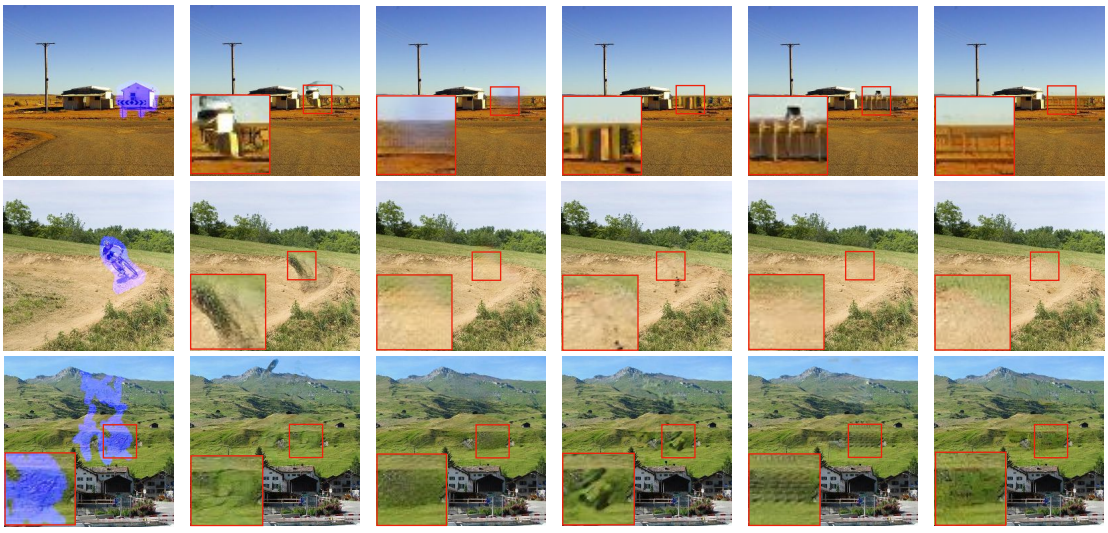
$$\begin{aligned}\mathcal{L}_{D^{sn}}=&\mathbb{E}_{x\sim p_{data(x)}}ReLU\left(1-D^{sn}\left(x\right)\right)\\&+\mathbb{E}_{z\sim p_{z(z)}}ReLU\left(1+D^{sn}\left(G\left(z\right)\right)\right),\end{aligned} \tag{4.13}$$

where D^{sn} denotes the spectral-normalized discriminator the discriminator network, $ReLU$ is the activation function [89].

4.3 Experiments

We evaluate our proposed approach on Places2 [72] and CelebA-HQ faces [53]. During training, the images are with 256×256, and we use a single NVIDIA RTX 2080Ti GPU to train our model with a batchsize of 8. For Places2, we adopt random-crop before sending the data into the model, and for CelebA-HQ, we downsample the data to 256×256 for training. We utilize the same arbitrary-shape mask generation process as in [6]. The whole model is built on TensorFlow v1.15, CUDNN v7.6.5, CUDA v10.2. We apply Adam optimizer with a learning rate of 0.0001 for both generator and discriminator. In Section 4.3.1 and Section 4.3.2, we show the qualitative and quantitative results compared with other SOTA methods that include CA [8], EC [10], Gated [6] and HiFill [7]. In the last part, we introduce the ablation study on our model, elaborating the importance of our proposed skip connection attention fusion and dilated convolution residual.

4.3.1 Qualitative Results

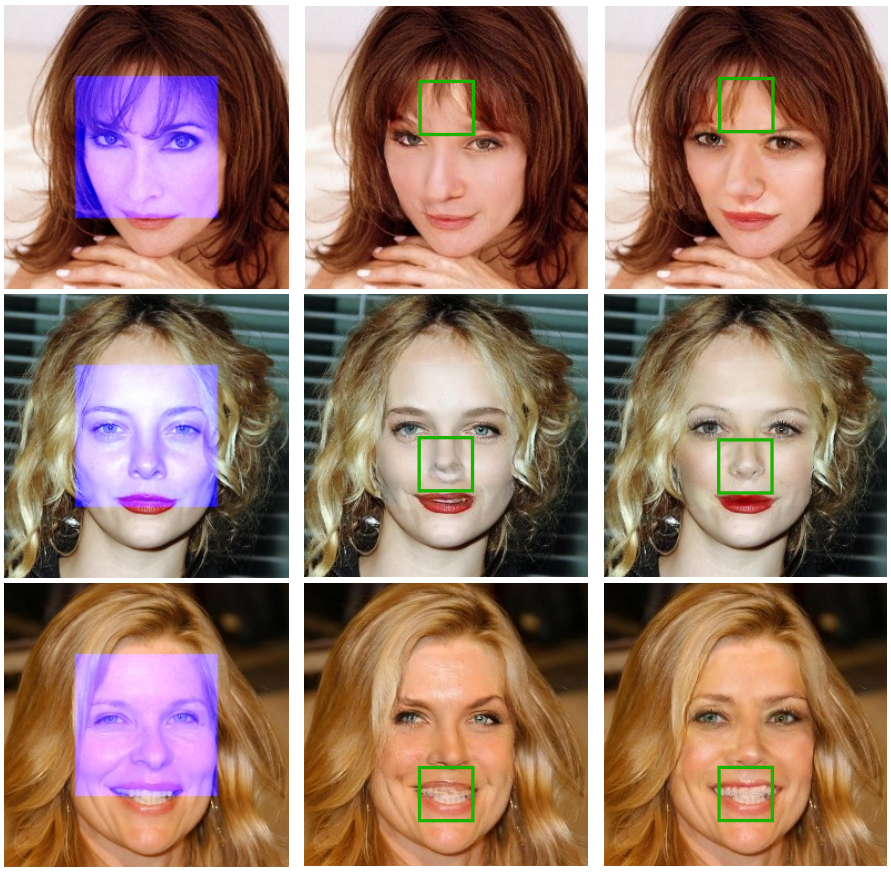


(a) Masked (b) CA [10] (c) EC [9] (d) Gated [6] (e) HiFill [7] (f) Ours

Fig. 4.3: Qualitative comparisons on Places2 validation dataset. Images are in 512×512. Best viewed by zooming-in.

In Fig. 4.3, we show the visual comparison results. We obtain the results on Places2 dataset with a resolution of 512×512. The input of masked images is shown in Fig. 4.3(a). We choose masks with arbitrary shapes and show the results of object removal, people deletion, and free-form inpainting. The results show that CA tends to generate artifacts with irregular patches. EC could generate sound structure due to its edge completion network, but the blur results indicate that it struggled with filling in high-frequency details.

For Gated, noisy patches still exist, e.g. the dissonant grass patches on the third row. The results of HiFill have shown that it could generate good textures, while these details are usually duplicated. We argue this phenomenon is because their model lacks modeling the image structure, and it is hard to organize the patterns well. In contrast, our model could learn the enriched features with image texture and structure through attention feature fusion. The last column's results show that our model could fill the hole with plausible content without blur and artifacts.



(a) Masked (b) Gated (c) Ours

Fig. 4.4: Qualitative results on CelebA-HQ val set (with image resolution 256×256).

We also show our results on CelabA-HQ faces in Fig. 4.4. We compare our method with Gated [6], showing that our methods could synthesize plausible content with meaningful human facial contour and fine details (e.g. nose in the second row and teeth in the last row).

4.3.2 Quantitative Results

The quantitative comparison is conducted on Places2 dataset with free-form and center square masks. For a fair comparison, we choose images with 256×256 from Places2 val-

Table 4.1: Quantitative results on the Places2 validation dataset. Images are in 256×256. For the free-from mask, we adopt the settings from [26]. Up-arrow (↑) indicates higher score is better, down-arrow (↓) indicates lower is better.

Mask type	free-form mask			center square mask		
Metrics	SSIM↑	PSNR↑	L1↓	SSIM↑	PSNR↑	L1↓
CA [6]	0.8299	20.54	0.0461	0.7399	19.47	0.0505
EC [10]	0.9021	24.05	0.0382	**0.7505**	**20.53**	0.0546
Gated [6]	0.9124	24.23	0.0270	0.7357	19.17	0.0519
HiFill [7]	0.8561	21.86	0.0409	0.7008	19.06	0.0553
ours	**0.9137**	**24.36**	**0.0270**	0.7469	20.10	**0.0476**

idation set. These images are first cropped into 512×512 randomly and then resized into 256×256. We obtain the free-from masks from the commonly-used mask set from [26]. We randomly pick 100 masks for validation from 20% to 30% mask to image ratio. We use standard metrics that include SSIM [37], PSNR, and $\mathcal{L}_1$. Table 4.1 shows that our method outperforms other state-of-the-art on most of the metrics, with a little inferior to EC [10] on center square masks.

4.3.3 Ablation Study

Table 4.2: Ablation study on our feature fusion-based model. SCAF is our skip connection attention fusion, and DCR is our dilated convolution residual.

Model	SCAF	DCR	SSIM↑	PSNR↑	L1↓
Skip-connection w/o AFF			0.9531	27.35	0.0151
Skip-connection w/ AFF	✓		0.9562	27.65	0.0149
Dilated conv residual		✓	0.9575	27.78	0.0145
Final model	✓	✓	**0.9596**	**28.12**	**0.0140**

To evaluate our skip connection attention fusion (SCAF) and dilated convolution residual (DCR), we have done experiments and show the effectiveness of these two components. In these experiments, we choose a subset of images from Places2 validation set

in the resolution 256×256. Random masks from [26] are used. From Table 4.2, we could find that both SCAF and DCR modules are essential for our model. The complete model with both modules reaches favorable results.

4.3.4 Failure Cases Analysis

This section presents some failed examples and analyses. From Fig. 4.5, the first example shows the proposed method brings some color inconsistency, indicating the current architecture still has space to improve. From the feature fusion perspective, the color consistency could be enhanced through a better fusion of high-level semantics and low-level details. The second example in Fig. 4.5 has shown the failed case when restoring the glasses in a human face inpainting task. This failure is due to the number of samples wearing glasses being scarce, and these testing images are commonly treated as humans without glasses. One possible solution for this scenario is to introduce external guidance, which is beyond this chapter's scope.

(a) Masked (b) Original (c) Ours

Fig. 4.5: Some failure cases of the proposed method.

4.4 Summary

This chapter introduces a feature fusion-based method for high-quality image inpainting. Current methods cannot model the image texture and structure appropriately, resulting in blur and artifacts in the synthesized results. This novel method leverages the attention feature fusion technique to address this issue by passing the shallow details to deep

semantics and building a modified multi-dilated residual block. In this way, this model could obtain features with rich global and local information. Extensive experiments on the benchmark datasets have shown the outperformance of the proposed approach to state-of-the-art methods.

Chapter 5

Efficient User-oriented Image Inpainting

In Chapter 3 and Chapter 4, we discussed the deep learning-based image inpainting theoretically. To archive high-quality inpainting, we proposed deep generative models based on multi-scale and feature fusion mechanisms, respectively. In practice, image inpainting has been widely applied to many downstream tasks, and many user interactive actions are needed. Take the object removal as an example, the mask of the distracting object should be given by the user before using the inpainting system.

However, image inpainting researchers have not paid much attention to building efficient impainting systems that could be easily applied to these downstream tasks. In particular, an AI system's need for interactive image inpainting is surging. In that system, efficiency, interactive ability, and interoperability are highly considered apart from inpainting quality.

This chapter focuses on addressing the above challenges from a practical perspective. The following sections present two user-oriented image inpainting methods where object detection/segmentation and attention mechanism are employed. In the first work, we build a fast object removal system that integrates object segmentation with image inpainting into an aggregated system. In the second work, our proposed guided attention module allows users to input guidance to the inpainting system and obtain favorable results.

The chapter begins by introducing the interactive image inpainting and the related works. The following section describes the proposed efficient object detector algorithms as a preliminary for the fast object removal pipeline. We then introduce the details of the object removal pipeline and the experimental results on our developed PB-101 dataset. In the last part, we elaborate our plug-and-play guided attention module and the results in real-life scenarios.

5.1 Interactive Image Inpainting

Compared with other deep learning-based image inpainting models that focus on improving the quality of restored images, interactive image inpainting has caught researchers' eyes recently. Interactive image inpainting can be divided into two categories. The first one is that the system allows users to give external inputs as guidance for generating the expected results. This information includes bounding boxes via object detection algorithms, semantic segmentation maps via segmentation methods, and other straightforward user inputs, e.g. sketches. During inpainting, these external inputs are integrated with the inpainting backbone model and should guide the inpainting process as the image editing task. Yu *et al.* [6] propose to use an external channel to load user input such as sketches during training. The trained model could handle user interaction with lines, such as edges or sketches. The results on both facial and natural scenes have shown the effectiveness of their user-guided approach, as shown in Fig. 5.1.

(a) Original (b) Masked (c) Gated

Fig. 5.1: Guided inpainting examples of Gated [6].

Another category is methods based on users' feedback. According to how artists draw paintings, the ideal image inpainting model should be a progressive process in which the inpainted results with defects are improved after accepting users' feedback. One most prominent problem of implementing such a feedback system on a deep learning-based model is how to quantify the feedback during the training. It is impossible to obtain

a large-scale labeled dataset with user feedback, and we need a model to mimic how humans evaluate the generated results. Zeng *et al.* [14] proposed an iterative inpainting approach with a feedback mechanism based on confidence map scoring. Their model adopts a progressive inpainting mechanism. On each iteration, it fills the high confidence area within the mask, and the confidence map updates after each iteration. At the same time, the filled pixels after each iteration are recognized as known pixels during the next iteration. The confidence map is obtained by calculating how much each pixel contributes to the overall loss. After a few iterations, the high confidence area will dominate the mask area, and the mask size shrinks to zero.

5.2 MPSSD: the Pre-processing Step of Efficient Object Removal System

This section introduces the proposed multi-path fusion single shot detector. This object detector can detect objects accurately in real-time, and it is designed as a pre-processing step of object removal system in the Section 5.3.

Object detection has been a fundamental task in computer vision which aims at localizing objects via bounding boxes and assigning a certain class to each of them. It has been widely adopted on intelligent systems as a crucial component and applied for specific purposes, such as pedestrian detection, face detection, and text detection [90]. Recent deep learning-based detectors [91–96] have seen great achievements compared with traditional methods. Inspired by these works, we propose an efficient and accurate deep detector.

Most state-of-the-art object detectors fall into two categories: one-stage and two-stage. Two-stage methods are proposed earlier, especially for Faster R-CNN (Regions with CNN features) [93] and R-FCN (Regions with Fully Convolutional Networks) [96].

In the first stage, regions with a high probability of containing foregrounds are extracted as proposals. These proposals are sent to a network for classification and bounding-box regression in the second stage. While archiving high accuracy on benchmark datasets, such as PASCAL VOC [97] and MS COCO [98], two-stage approaches usually run slowly because of the high computation cost of the region proposal generation. Another one-stage branch (e.g. Single Shot Detector (SSD) [95], You Only Look Once (YOLO) [94]) employs a proposal-free pipeline; thus, it needs few computation resources. These detectors usually adopt fully convolutional architecture and calculate class confidence and regression results directly on predefined boxes. One-stage detectors could run efficiently with accuracy slightly inferior to two-stage ones. Thus they are favorable for sensors system due to the crucial need for real-time inference.

The one-stage approach SSD detects objects directly from multiscale features. This

feature pyramid consists of the last layers of the backbone and adjacent convolutional layers. Shallow layers within this pyramid detect small-scale objects and deep ones responsible for larger objects. Although efficient, such a scheme misbehaves on small objects. Since shallow layers always learn localization information while deep layers have more semantics information [99]. The semantics are crucial for detecting small objects. Thus, exploiting shallow layers alone is not enough for small scales detection.

To address this issue, some recent works [76, 100] introduce a top-down pyramid structure. They upsample deeper layers to pass semantics information to shallow layers before combining them with lower ones. Although borrowing some semantics from deep layers, only one feature pyramid is engaged. We believe one single pyramid is still not informative enough for accurate detection. Thus, we propose a multipath model consisting of several feature pyramids to learn the most informative representations. Although novel in deep object detectors, the strategy of multipath is widely applied in computer vision areas [68, 101]. Inspired by Feature Fusion Single Shot Multibox Detector (FSSD) [102], we use Feature Fusion Module (FFM) to obtain fused features from the base pyramid and obtain our multipath feature pyramids. Then we generate our final features by sending these pyramid features to our Pyramid Aggregation Module (PAM). At last, these informative multiscale features are fed into detection heads for final processing. We conduct extensive experiments on challenging datasets PASCAL VOC and MS COCO, and the results show that our algorithm is better than most stage-of-the-art one-stage object detectors. Below are our main contributions:

- We propose a multipath fusion strategy to enhance the feature pyramid in a single shot detector;
- Our feature fusion module and pyramid aggregation module are introduced that proves able to fuse information from the base pyramid and generate our informative pyramid efficiently;
- Extensive experiments on PASCAL VOC 2007 & 2012 and MS COCO 2015 show our proposed Multi-Path fusion Single Shot Detector (MPSSD) can outperform other one-stage detectors consistently.

5.2.1 Object Detection

We present the related work by dividing it into three parts. We will first describe the deep learning-based object detector and introduce the single shot detector branch. Finally, we will discuss the feature enhancement in deep detectors.

Deep Object Detector

Traditional object detection methods always rely on hand-crafted features [103–107]. Histogram of Oriented Gradients (HOG) [104] is a representative feature descriptor which can be calculated on densely uniformed cells. Deformable Part-based Model (DPM) is an improved version of HOG which follows the "divide and conquer" scheme to detect an image on different parts during inference [107]. Traditional methods are limited by the hand-crafted features and less efficient computation resources. Thanks to the great achievements of deep convolutional neural networks on computer vision [38, 62, 108–112], many deep learning-based detectors are proposed with superior performance compared with conventional methods. R-CNN [91] introduces the idea of region proposals, which are regions with high probabilities of including objects. They extract these proposals first and send them to CNN for further prediction. This two-stage scheme has become popular due to its superiority over conventional hand-crafted methods in accuracy and speed. Fast R-CNN [92] accelerates the training of R-CNN through a novel multitask loss to train the classification and bounding box regression simultaneously. To alleviate the high computation cost of the proposals generation process in these algorithms, faster R-CNN [93] designs a learning-based region proposal network (RPN) to generate proposals efficiently. R-FCN [96] adopt fully-convolutional networks, and they propose the position-sensitive RoI Pooling (PSRoI) to replace the RoI Pooling in faster R-CNN to improve the accuracy and efficiency further. To solve the extreme foreground-background class imbalance in training, focal loss [113] as a variant of the standard cross-entropy loss is proposed to learn more hard examples during training.

Single Shot Detector

Apart from region proposal-based detectors, another branch termed one-stage detectors abandon the procedure of proposals extraction to achieve faster inference. Among these methods, YOLO [94] probably presents the first one-stage detector, which successfully applies classification and bounding box regression directly on each predefined image grid. The drop of proposal generation results in a very high speed, but the accuracy is relatively low. YOLO's improved versions [114, 115] have focused more on this problem, especially for small object detection.

Another one-stage detector is SSD [95] that exploits a multiscale fashion to predict objects with various scales, and this significantly improves the performance of the one-stage detector. We adopt SSD as our base model since it satisfies the trade-off between speed and accuracy. At first, adopting VGG16 as the backbone, SSD modifies the last fully connected layers into a convolutional version. For an input image with size 300×300, they extract layers conv4_3 with size 38×38 and conv_fc7 19×19 from backbone. Then, they add several convolution layers conv8_2, conv9_2, conv10_2 and conv11_2 to extract features with size 10×10, 5×5, 3×3 and 1×1 respectively. The extracted

features establish the detection pyramid in SSD, and each feature is responsible for detecting a particular scale. The shallow layer, i.e., conv4_3, detect relative small objects in the image, while the deep layer, i.e., conv9_2, is used to detect large objects. They design many defined boxes for each feature, and each box is assumed to include foregrounds. Each feature's adjacent head block predicts the offsets and classification confidences for these boxes. Finally, post-processing called non-maximum suppression (NMS) is used to refine these predicted results.

Deep Feature Enhancement

The quality of feature representations is essential for object detection. Many recent works study how to build more informative features to improve the accuracy of deep detectors. Feature Pyramid Network (FPN) [76] adopts the top-down feature pyramid design to add lower-level features with higher ones. This straightforward design passes higher-level semantics to lower levels, thus improving the detection performance, especially on small objects. Deconvolutional Single Shot Detector (DSSD) [100] applies the same strategy to vanilla SSD architecture and archives better performance. Still, their deconvolution design would hurt the high-efficient property of the original SSD. HyperNet [116] and ION [117] combine the hierarchical features into one layer before the prediction, aiming to build the most representative feature which enjoys both local and global information. In terms of feature enhancement in one-stage detectors, inspired by FPN, StairNet [118] improves the nearest neighbor upsampling with a newly introduced top-down feature combining modules and attains even better results. RFBNet [119] proposes a novel Receptive Field Block (RFB) with groups of dilated convolution design, and this module can build high-quality representatives by enlarging the receptive field of input features. Some other SSD-based methods try to enhance the base features through the fusion of different layers with their novel fusion modules [102, 120, 121]. This section proposes a new multipath fusion strategy to generate multilevel feature pyramids. We illustrate that our SSD-based object detector could outperform many qualitative and quantitative counterparts on three benchmark datasets.

5.2.2 Method

Before introducing our novel architecture and modules, we revisit the one-stage detector SSD and its variant FSSD. We elaborate our methodology in detail and then offer interpretations accordingly.

Deep Feature Pyramid

Single shot multibox object detector often extracts features from one backbone network and adds several convolution layers to obtain a feature pyramid. We denote the feature of the l^{th} layer is $x_l \in \mathbb{R}^{H \times W \times C}$ (where H, W, and C refer to height, width, channel

respectively). The feature pyramid in SSD for multiscale detection is defined as:

$$X_{Detection} = \{x_k, x_{k+1}, ..., x_L\}, \tag{5.1}$$

where x_l is used to predict objects within a range of scale. Specifically, the shallowest feature x_k is responsible for detecting small objects within inputs, and larger objects are detected in deeper layers. However, the more semantics it could provide global information as the feature goes deeper. Shallow features missing this global guidance would lead to misdetection. Thus, SSD has poor accuracy on relatively small objects.

To address this, many investigations focus on building more informative representations. For instance, the straightforward way is adding higher-level semantics to low-level local features [100]. However, this strategy may need much heavy computation owning to the complicated element-wise operations between adjacent features. An efficient method is proposed by FSSD [102] to fuse low-level and high-level information. They concatenate several features and generate the following pyramid the same as SSD. We show the whole process below:

$$X_{Detection} = \{x_k^{'}, x_{k+1}^{'}, ..., x_L^{'}\}, \tag{5.2}$$

and each level feature is computed by:

$$x_l^{'} = F(\{x_k, x_{k+1}, ..., x_{k+n}\}), \tag{5.3}$$

where F indicates the operation of feature fusion and pyramid generation, and the number of base features from SSD to be fused is $(n + 1)$. We can find in (5.3) only base features from k to $k + n$ are used to generate the final feature pyramid, while the rest of the base features with rich semantics are discarded. We argue that these features can also be used to create the final pyramid through a dedicated transformation. Thus, we design a multipath routine to maximize the usage of base features. Firstly, we develop several groups of features on the base pyramid through feature fusion modules. Then, we pass these pyramids into our aggregation module for final fusion. The detection results are predicted on the final enhanced pyramid:

$$X_{Detection} = \{x_k^{''}, x_{k+1}^{''}, ..., x_L^{''}\}, \tag{5.4}$$

and each feature is expressed by:

$$\begin{aligned} x_L^{''} = A(\{F_1(\{x_k, x_{k+1}, ..., x_{k+1+n}\})\}, ..., \\ \{F_M(\{x_{k+M-1}, x_{k+M}, ..., x_{k+M+n}\})\}), \end{aligned} \tag{5.5}$$

where M is the number of pyramid paths. We apply several feature fusion modules, represented by F_m and the pyramid aggregation module indicated by A. More details about these two modules can be found in the following part.

The final feature pyramid is rich in both localization and semantic information at each level. For instance, the last layer of this pyramid is fused from the same size features in each path, and these features are enhanced through nonlinear transformations of different level base features. Our fusion strategy could generate features rich in local and global clues, and it is lightweight to be inserted in the vanilla SSD model.

MPSSD Architecture

As shown in Fig. 5.2, our architecture is based on single-stage design in vanilla SSD. The backbone network we use is VGG16. Same as that in SSD, we replace the last fc6 and fc7 layers with convolution layers for a fully convolutional fashion. Other base features are generated from the following several convolution layers. These base features are then sent to two modules for enhancement. The enhanced feature pyramid with local and global information is connected with detection heads for prediction. We illustrate the modules in the following sections.

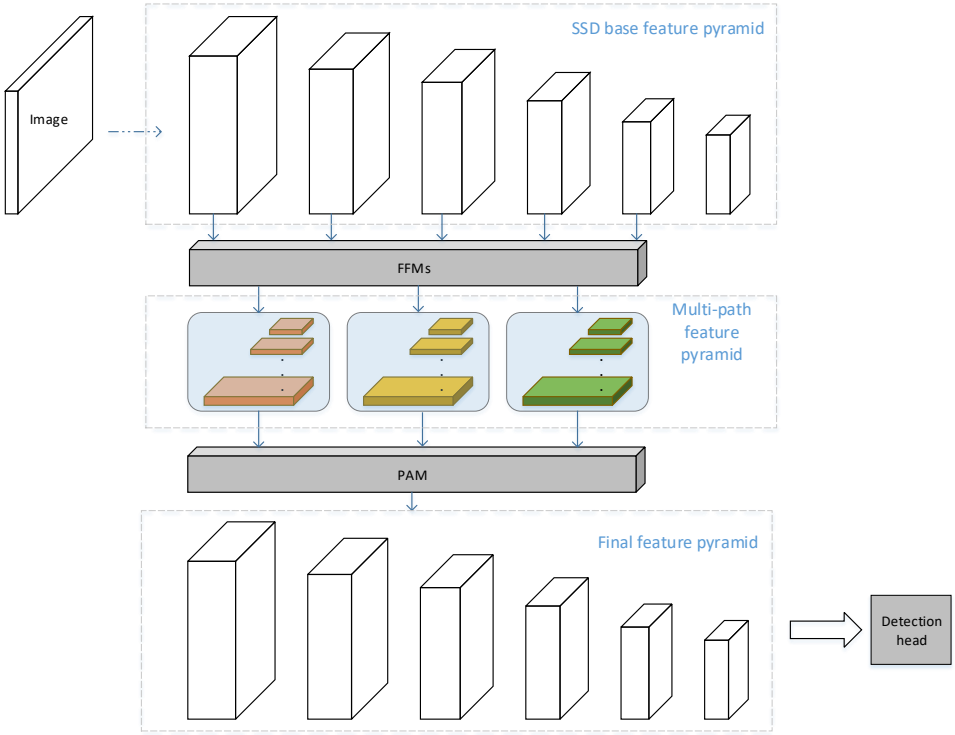


Fig. 5.2: MPSSD model pipeline. From top to bottom, each part indicates Single Shot Detector (SSD) base feature pyramid, multipath feature pyramids generated through Feature Fusion Modules, final feature pyramid out from pyramid aggregation module.

Feature Fusion Module

To better integrate various features with different scales, FSSD engages an efficient fusion method. Inspired by the success, we also exploit this module to generate our novel multipath feature pyramids. The structure is shown in Fig. 5.3. For an input image of size

300 $\times$ 300, we choose layers from conv_fc7, conv6_2, and conv7_2 to fuse. Since these features are in different sizes (size of conv_fc7 is 19 $\times$ 19, conv6_2 is 10 $\times$ 10, conv7_2 is 5 $\times$ 5), smaller features conv6_2 and conv7_2 are upsampled to the same size of conv_fc7 before fusion. We also utilize 1 $\times$ 1 convolution before upsampling to reduce the channels into a particular number, and different channel numbers are set in our FFMs. Next, we concatenate these same size features in channel dimension. The feature pyramid is generated through several 3 $\times$ 3 conv + Rectified Linear Unit (ReLU) layers. For multipath pyramids, we repeat this process several times.

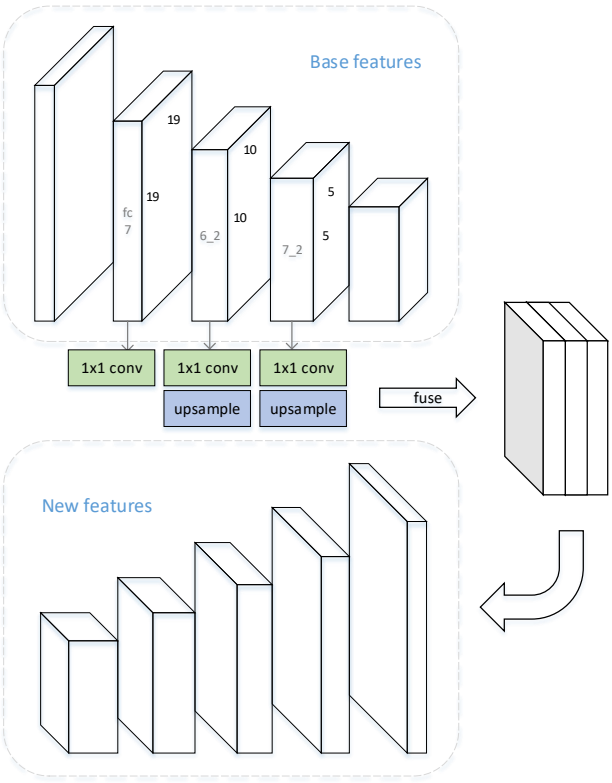


Fig. 5.3: Design of feature fusion module. The top are base features. Three features are selected to be fused. These features are computed through 1 $\times$ 1 conv and upsampled into the same scale and channel. Followed by concatenation to form the aggregation feature. The bottom shows the generated new features from the aggregation.

Pyramid Aggregation Module

We devise a novel pyramid aggregation module to transform the multipath pyramid from feature fusion modules into the final detection pyramid. This module is illustrated at the bottom of Fig. 5.2.

The whole process can be divided into two steps. Firstly, we aim to arrange these features into a single pyramid for further detection. Thus same scale features are concatenated along the channel dimension. It is noticeable that each path pyramid has a different

number of features. In the second step, we adopt an attention block in Squeeze-and-Excitation Networks (SENet) [84] to enhance the concatenated features. This attention block is implemented through squeeze and excitation steps. We show its procedure in the following. Firstly on the squeeze step, the global average pooling is used to generate $z \in \mathbb{R}^C$ for input $X \in \mathbb{R}^{H \times W \times C}$:

$$z_c = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} X_c(i, j). \tag{5.6}$$

Then, the excitation step generates the attention activation feature s by

$$s = F_{ex}(z, W) = \sigma(W_2 \delta(W_1 z)), \tag{5.7}$$

where σ and δ refer to ReLU and sigmoid operation respectively, $W_1 \in \mathbb{R}^{C' \times C}$ and $W_2 \in \mathbb{R}^{C \times C'}$, we adopt $C' = \frac{1}{16}C$ in our experiments. The last stage outputs the result $\widetilde{X} \in \mathbb{R}^{H \times W \times C}$ by

$$\widetilde{X}_c = F_{scale}(X, s_c) = s_c \cdot X_c, \tag{5.8}$$

where s_c is the learned activation feature s on channel c.

5.2.3 Results and Discussion

We conduct extensive experiments on three benchmarks PASCAL VOC2007, PASCAL VOC2012, and MS COCO. As for the evaluation metrics, in VOC, predicted boxes that have Intersection over Union (IoU) with the ground truth higher than 0.5 are defined as positive results. In COCO, the metrics are split into different parts based on different Intersection over Union (IoU) settings. Our implementation is based on Pytorch 0.4.0. Backbone network VGG16 is pretrained on ImageNet [122] with modification into a fully convolutional version. As for the training settings, we keep them mostly the same as the settings in SSD for a fair comparison. On the other hand, we adopt the network initial method from [123].

Dataset

We applied two popular datasets, PASCAL VOC and MS COCO, during training and testing. The PASCAL VOC dataset contains images collected from the Internet. The 2007 and 2012 versions are the two most used in object detection. They have been categorized into 20 classes since 2007 based on four groups: person, animal, vehicle, and indoor. MS COCO is a newly established dataset for object detection, segmentation, and captioning. Compared with VOC, MS COCO not only adopts annotations for the bounding box but each object is labeled with instance segmentation for more precise localization. This dataset is more challenging, with more than 200,000 images categorized into 80 classes,

and there are more labeled objects per image than PASCAL VOC. Table 5.1 shows the detailed statistics of these datasets. For intuitive observations, we also offer some examples from these two datasets in Fig. 5.4.

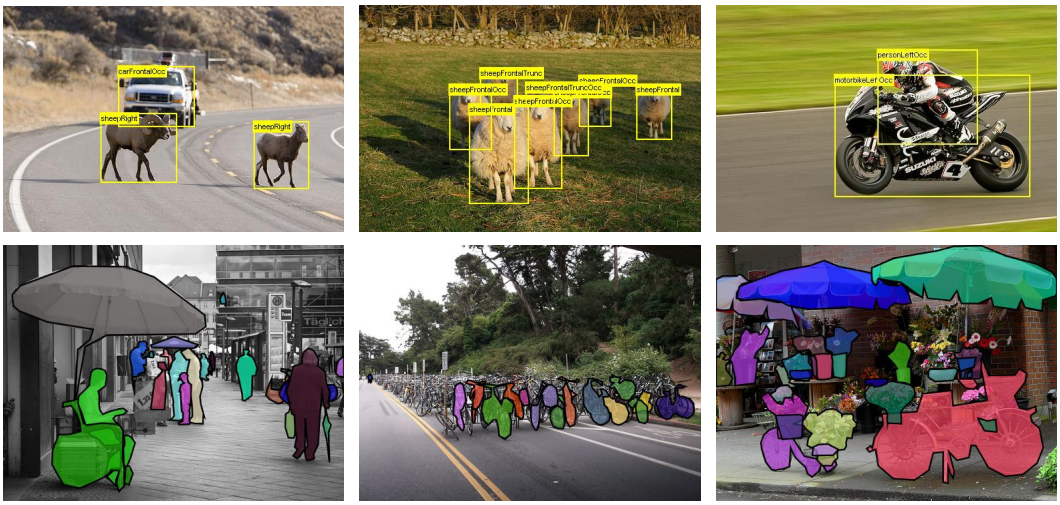


Fig. 5.4: Some object detection examples in Pascal VOC [97] (**upper**) and MS COCO [98] (**lower**).

Table 5.1: Dataset statistics of PASCAL VOC & MS COCO.

Name	train	validation	test	category	total annotated objects
VOC 2007 [97]	2,501	2,510	4,952	20	9,963
VOC 2012 [97]	5,717	5,823	10,991	20	11,530
MS COCO 2015 [98]	82,783	41,620	125,436	80	2,500,000

Results on PASCAL VOC 2007

All the experiments on PASCAL VOC apply the popular split, which uses the union of VOC2007 $trainval$ and VOC2012 $trainval$ as the training data, and VOC2007 $test$ which contains 4952 images for testing. For training on inputs with scale 300×300, the batch size is set at 24 on a single NVIDIA 1080Ti GPU. The initial learning rate is 4×10^{-3}. The learning rate decreases to 4×10^{-4} at 150 epoch and to 4×10^{-5} at 200 epoch. We finish our training on 250 epoch.

Table 5.2 shows the results on VOC2007. Our model outperforms vanilla SSD significantly. As for 300×300 inputs, by embedding our algorithm into SSD, the mAP (mean Average Precision) improves from 77.5% to 80.3%, while increases from 79.5% to 81.8% on 512×512 inputs. We also archive better performance than DSSD with ResNet101 [38] as the backbone. We find that FSSD with a similar fusion strategy but fewer features is inferior to our method. Based on the mAP, we archive 1.5% higher on

Table 5.2: PASCAL VOC 2007 *test* results. Faster R-CNN, HyperNet, ION and R-FCN adopted inputs with minimum size 600, while StairNet kept with 300 × 300.

Method	backbone	mAP(%)	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	person
Faster [93]	VGG16	73.2	76.5	79.0	70.9	65.5	52.1	83.1	84.7	86.4	52.0	81.9	65.7	84.8	84.6	77.5	76.7
Faster [93]	ResNet101	76.4	79.8	80.7	76.2	68.3	55.9	85.1	85.3	89.8	56.7	87.8	69.4	88.3	88.9	80.9	78.4
HyperNet [116]	VGG16	76.3	77.4	83.3	75.0	69.1	62.4	83.1	87.4	87.4	57.1	79.8	71.4	85.1	85.1	80.0	79.1
ION [117]	VGG16	76.5	79.2	79.2	77.4	69.8	55.7	85.2	84.2	89.8	57.5	78.5	73.8	87.8	85.9	81.3	75.3
R-FCN [96]	ResNet101	80.5	79.9	87.2	81.5	72.0	69.8	86.8	88.5	89.8	67.0	88.1	74.5	89.8	90.6	79.9	81.2
SSD300 [95]	VGG16	77.5	79.5	83.9	76.0	69.6	50.5	87.0	85.7	88.1	60.3	81.5	77.0	86.1	87.5	83.9	79.4
DSSD321 [100]	ResNet101	78.6	81.9	84.9	80.5	68.4	53.9	85.6	86.2	88.9	61.1	83.5	78.7	86.7	88.7	86.7	79.7
StairNet [118]	VGG16	78.8	81.3	85.4	77.8	72.1	59.2	86.4	86.8	87.5	62.7	85.7	76.0	84.1	88.4	86.1	78.8
FSSD300 [102]	VGG16	78.8	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Ours300	VGG16	**80.3**	83.5	86.7	79.0	74.4	59.9	87.4	87.4	86.5	64.9	88.1	77.9	86.6	88.0	87.2	80.9
SSD512 [95]	VGG16	79.5	84.8	85.1	81.5	73.0	57.8	87.8	88.3	87.4	63.5	85.4	73.2	86.2	86.7	83.9	82.5
DSSD513 [100]	ResNet101	80.0	92.1	86.6	80.3	68.7	58.2	84.3	85.0	94.6	63.3	85.9	65.6	93.0	88.5	87.8	86.4
FSSD512 [102]	VGG16	80.9	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Ours512	VGG16	**81.8**	87.9	88.4	81.7	76.7	65.1	88.5	88.8	87.9	66.9	89.0	76.9	86.1	89.9	87.4	84.6

300×300 and 0.9% higher on 512×512, and this show our multipath design is effective. Compared with top-down design, such as StairNet, the 1.5 point higher accuracy proves our architecture helps to learn more powerful features. Our method also reaches better performance compared with these two-stage algorithms.

Results on PASCAL VOC 2012

We adopt the same training setting as that in our VOC2007 experiments to evaluate our model on dataset VOC2012 $test$, which contains 10,991 images in total. From Table 5.3, our method achieved the same results as the evaluation on VOC2007 dataset. For both input resolutions, our model improves significantly from the SSD baseline. Compared with other single-stage methods, our MPSSD reached the best performance under the same training settings. It is noteworthy that FSSD obtained better results via adopting an additional training dataset (MS COCO). These results again show the advantages of our proposed method.

Table 5.3: PASCAL VOC 2012 $test$ results from official evaluation server. For training data: "07+12":union of VOC2007 $trainval$ and VOC2012 $trainval$; "07++12": union of VOC2007 $trainval$ and $test$ and VOC2012 $trainval$; "07+12+S" union of VOC2007 trainval and VOC2012 $trainval$, plus SBD segmentation labels [124]; "07++12+COCO": union of "07++12" and COCO $trainval35k$.

Method	backbone	data	mAP(%)
Faster [93]	VGG16	07++12	70.4
Faster [93]	ResNet101	07++12	73.8
HyperNet [116]	VGG16	07++12	71.4
ION [117]	VGG16	07+12+S	76.4
R-FCN [96]	ResNet101	07++12	77.6
SSD300 [95]	VGG16	07++12	72.4
DSSD321 [100]	ResNet101	07++12	76.3
StairNet [118]	VGG16	07++12	76.4
FSSD300 [102]	VGG16	07++12++COCO	82.0
Ours300	VGG16	07++12	**77.7**
SSD512 [95]	VGG16	07++12	74.9
DSSD513 [100]	ResNet101	07++12	80.0
FSSD512 [102]	VGG16	07++12++COCO	84.2
Ours512	VGG16	07++12	**80.3**

Results on MS COCO

We also conduct experiments on a more challenging dataset, MS COCO, for further evaluation. For the training and validation split, we adopt $trainval35k$ for training, $minival$ for validation, and $test\text{-}dev$ 2015 for testing. For input on 300 $\times$ 300, our batch size is 16 on single NVIDIA 1080Ti GPU. The learning rate is initialized at 2×10^{-3} with schedule decay on epochs 90 and 120 by the factor of 10. The training ends at 150 epoch.

As shown in Table 5.4, the evaluation metrics are acquired through the official server on CodaLab. For 300 $\times$ 300 input images, our model outperforms SSD by a large margin, and mAP improves from 25.1% to 27.5%. Compared with FSSD, our model obtains 0.4% higher accuracy, which shows our effectiveness. For 512 $\times$ 512 input images, our model archives 33.1% accuracy, which is 4.3% and 1.3% higher than SSD and FSSD, respectively. Our method is only 0.1% inferior to DSSD513, and we believe this mainly because they adopt a more powerful backbone, ResNet101. Specifically, we can find ours works best on small-scale detection among these algorithms.

Table 5.4: MS COCO $test\text{-}dev2015$ results. Note: The "+++" includes box refinement, context, and multi-scale testing.

Method	backbone	data split	Avg. Precision, IoU:			Avg. Precision, Area		
			0.5:0.95	0.5	0.75	S	M	L
Faster [93]	VGG16	trainval	21.9	42.7	-	-	-	-
Faster+++ [38]	ResNet101	trainval	34.9	55.7	-	-	-	-
ION [117]	VGG16	train	23.6	43.2	23.6	6.4	24.1	38.3
R-FCN [96]	ResNet101	trainval	29.2	51.5	-	10.3	32.4	43.3
SSD300 [95]	VGG16	trainval35k	25.1	43.1	25.8	6.6	25.9	41.4
DSSD321 [100]	ResNet101	trainval35k	28.0	46.1	29.2	7.4	28.1	47.6
FSSD300 [102]	VGG16	trainval35k	27.1	47.7	27.8	8.7	29.2	42.2
Ours300	VGG16	trainval35k	27.5	47.8	28.5	10.2	30.2	41.0
SSD512 [95]	VGG16	trainval35k	28.8	48.5	30.3	10.9	31.8	43.5
DSSD513 [100]	ResNet101	trainval35k	33.2	53.3	35.2	13.0	35.4	51.1
FSSD512 [102]	VGG16	trainval35k	31.8	52.8	33.5	14.2	35.1	45.0
Ours512	VGG16	trainval35k	33.1	53.8	35.1	17.1	36.0	45.7

Inference Time Analysis

We consider the inference time of our algorithms because of the real-time requirement of

the sensor system. To compare fairly, for 4952 images in VOC 2007 $test$ with the size 300×300, we average their total inference time without considering the post-processing (NMS) step. All the results are evaluated on one NVIDIA 1080Ti GPU with batch size set at 1, and these methods are trained under the same settings.

We analyze how our newly introduced modules affect the inference speed. The results are shown in Fig. 5.5. Compared with vanilla SSD, our methods could reach much higher accuracy with little sacrifice of inference time. In addition, in contrast to FSSD, our multiple designs of FFMs and Pyramid Aggregation Module (PAM) would not spend too much computation with better performance. We also compared one-stage baseline Faster RCNN, and our method could again achieve higher accuracy with much less time cost. Since other one-stage methods are based on Faster RCNN baseline, we do not make the comparisons for simplicity. Finally, we test how the number of Feature Fusion Module (FFM) affects detection time. From Fig. 5.5, we find that one more FFM could reduce the inference speed a little, and thus it is essential to select the optimal parameters.

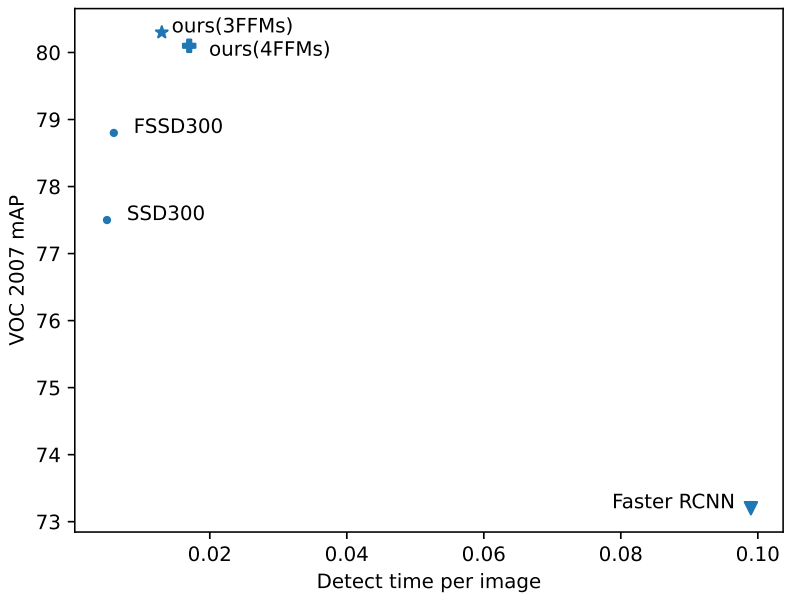


Fig. 5.5: Comparison on inference speed and accuracy. In the figure, 3FFMs indicates the model which adopts 3 feature fusion modules, while 4FFMs applies 4 feature fusion modules.

Qualitative Evaluation

In this part, we show some qualitative results on VOC 2007 $test$. As seen from each example in Fig. 5.6, the results of SSD are in the upper row, and ours is in the lower row. Boxes with classification confidences higher than 0.6 are chosen, and each category is labeled on the top of box.

For the first example, the potted plant with low illumination is misdetected in SSD,

while our model could handle this illumination variance and detect it successfully. The second example shows our model works better in detecting small objects, such as the cows in this image. The following three examples indicate our model outperforms the baseline on occlusion object detection. For the last example, the sofa is neglected by SSD while our model detects it. These results prove our learned semantics help to detect objects in these specific environments.

(a) example 1 (b) example 2 (c) example 3

(d) example 4 (e) example 5 (f) example 6

Fig. 5.6: Some detection results from VOC 2007 $test$. Upper: SSD. Lower: Ours. See more details in Section 5.2.3.

Ablation Study on Feature Fusion Modules

To show the effectiveness of our module, we conduct experiments on different settings of

feature fusion modules. We perform all the experiments on VOC 2007 $test$ with 300 $\times$ 300 inputs, but we should bear in mind that a similar conclusion can also be made in the other datasets.

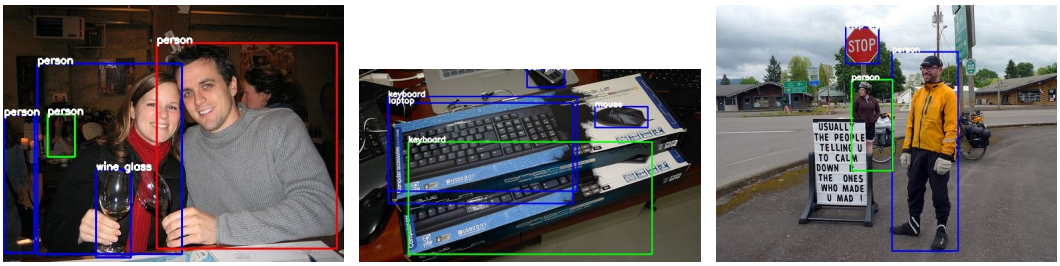
We run experiments with different numbers of FFMs and various choices of channel numbers. As seen from Table 5.5, the results in the first column come from vanilla SSD. By adding FFMs, we could find the accuracy improves a lot progressively. The model with three FFMs and a channel equal to 128 on the third FFM reaches the highest accuracy, and we choose this as our final model.

Table 5.5: Ablation study on VOC 2007 $test$. In the table, ffm3_c indicates the channel number of third feature fusion module.

1 FFM		✓				
2 FFMs			✓			
4 FFMs				✓		
3 FFMs with ffm3_c = 256					✓	
3 FFMs with ffm3_c = 128 (final model)						✓
mAP(%)	77.5	79.0	79.7	80.1	79.8	**80.3**

Failure Cases Analysis

In this part, we analyze the shortcomings of our method. Fig. 5.7 shows some examples with misdetection or incorrect classification from COCO 2015 $test$. In Fig. 5.7(a), the wine glass held by the man is not detected, and we conjecture that the failure is due to the object being tilted to a certain degree, which is not the same as its normal state. On the contrary, the wine glass held by the woman vertically is well-recognized. This case has shown that our model cannot detect objects with abnormal positions. In Fig. 5.7(b), the top keyboard is classified into laptop incorrectly. Although the object "laptop" consists of a "keyboard" and "screen", this incorrect classification is probably because the model is incapable of learning the distinguishable features of these two objects. Finally, in Fig. 5.7(c), the green signpost behind the man with the yellow coat is misdetected. Since this object is in an abnormal aspect ratio (tall and slender), this kind of shape is hard to be recognized, mainly because of the weakness of the anchor design scheme. We adopt a vanilla SSD anchor design which may not be able to cover objects with arbitrary shapes, especially for objects with large aspect ratios.



(a) (b) (c)

Fig. 5.7: Some failure cases of MPSSD. The results a,b,c are all from MS COCO 2015 $test$.

5.2.4 Future Work

We consider three aspects for future improvements. First, to improve the accuracy of our current model, we believe further optimizing of feature pyramid could reach a more accurate detection. Precisely speaking, the current method builds a fixed number of pyramids, which may only optimize on low-resolution inputs, i.e., 300×300. With rapid improvements on camera devices, high-resolution detection is highly required, and thus we need to redesign an optimal strategy. Second, to speed up the inference time of our model, we could focus on alleviating the tedious computation on the detection head. The predictions on defined boxes are redundant due to the original design in SSD, and this hurts the speed a great deal. Finally, we will extend the proposed model to various other intelligent systems and sensor application domains. As a one-shot detector both efficiently and accurately, the proposed multipath fusion strategy-based algorithm could find its applications extensively.

5.2.5 Conclusion

We devise a novel SSD-based one-stage detector. Unlike prior methods, we propose a multipath fusion strategy to utilize the local and global information from different layers fully. To aggregate this information, we first use feature fusion modules to generate feature pyramids, followed by a pyramid aggregation module to fuse and enhance relative features. The generated pyramid is suitable for detection in a multiscale manner with rich information. Quantitative and qualitative experiments on PASCAL VOC and MS COCO show that our MPSSD outperforms vanilla SSD. While for other single-stage counterparts, our method archives comparable results without hurting efficiency.

5.3 An Efficient Object Removal System

As a fundamental task of computer vision, image inpainting has a wide variety of applications on images and graphics, e.g. object removal, image editing, re-targeting, manipulation, compositing, and rendering. With the prevalence of deep learning, these applications have achieved great success in high performance and scalability, significantly impacting people's real life. In order to fit the real-world demand, researchers try to build an intelligent system with integrated image inpainting.

In a user-oriented inpainting system, current inpainting models have two main drawbacks: the difficulty of creating masks and the diversified results caused by user input masks. Take object removal as an example. The user should create the mask of the distracting object before sending it to the inpainting networks. Typically, a traditional interactive system allows users to draw the distracting area with arbitrary sizes of painting brush. However, this procedure is time-consuming and lacks accuracy. Even though current deep learning-based networks could generate more plausible content than non-deep learning methods, modern models still have high requirements for mask quality. As shown in Fig. 5.8, applying the same baseline method, the results are worse when low-quality masks are used.

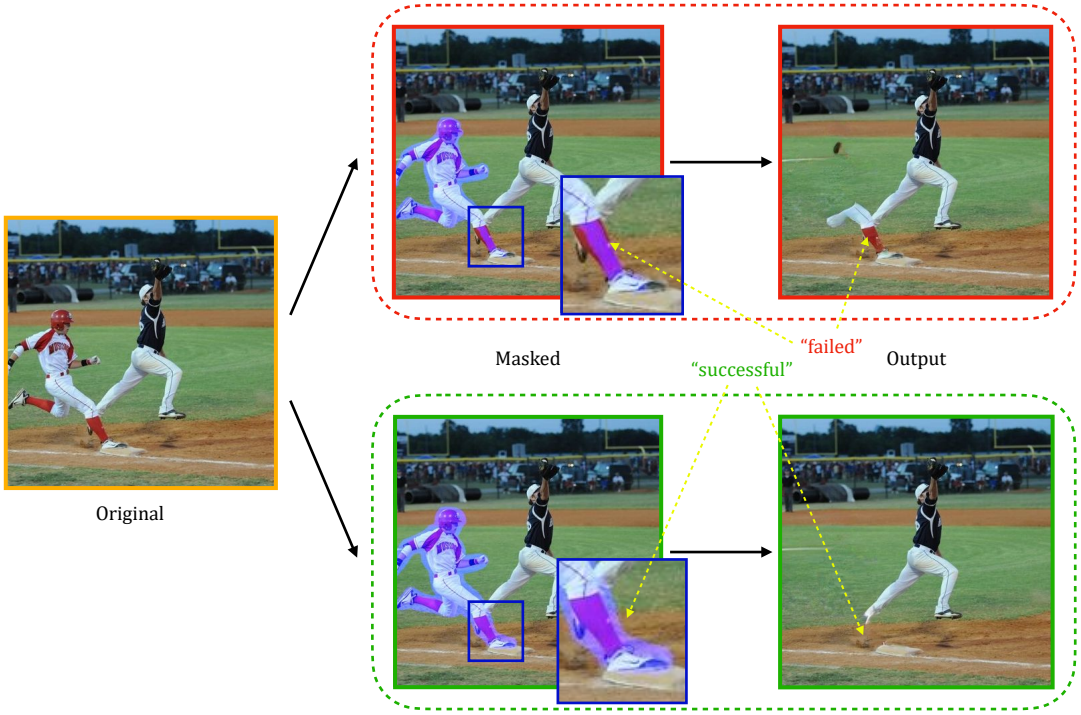


Fig. 5.8: Object removal results of the same input but with different masks. Results are tested on Gated [6]). The top red box is a bad-quality mask (without fully covering the distracting target) with wrong removal results. The bottom green box is an example of a high-quality mask (with the target fully covered) and obtains a good result.

Another defect is the uneven results based on diverse masks as inputs. In Fig. 5.9, we show that even if high-quality masks (masks already fully cover the distracting targets) are used, there may generate diversified inpainting results of the same deep learning-based model due to the high sensitivity of pixels during the generation process. These diversified results would cause inconsistency in the intelligent system in the further steps.

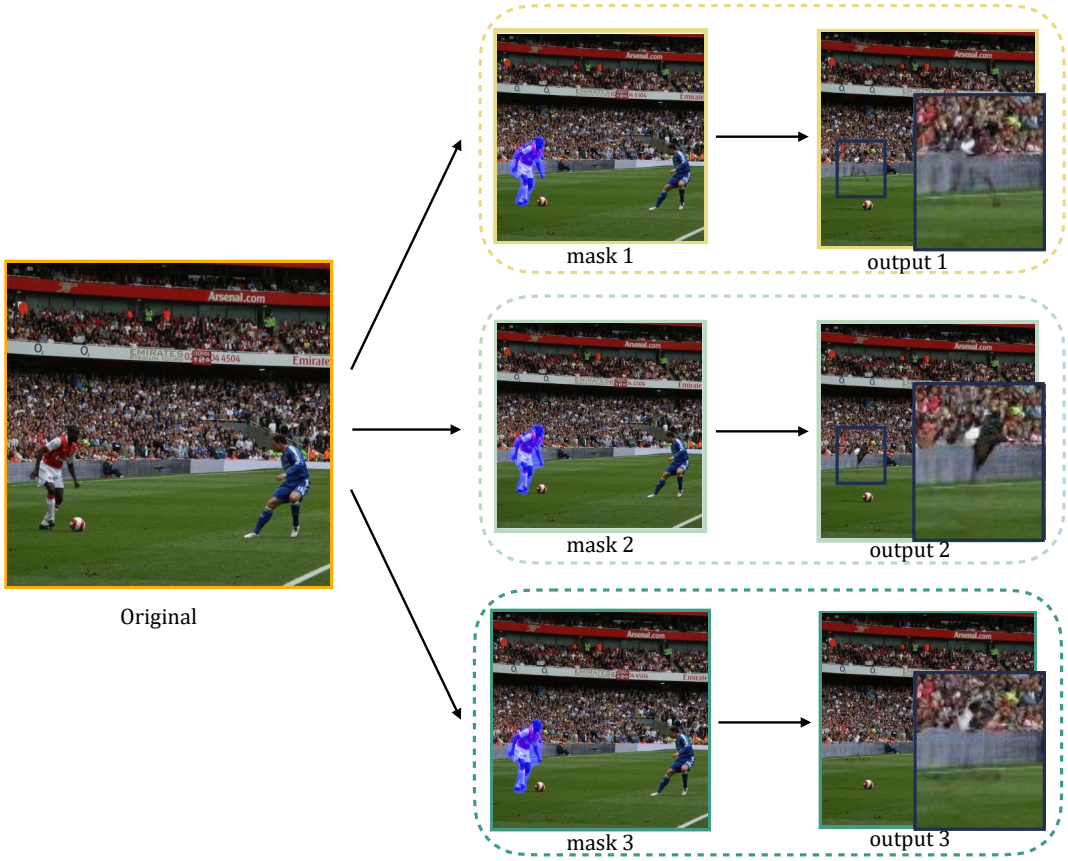


Fig. 5.9: The generated diversified results due to the adoption of different masks. Results tested from Gated [6]. We use three different masks and obtain three diversified results, even though all the masks are "good" (fully covering the removal target).

To address the above two defects, we need a system that could create the masks for the user automatically. At the same time, the generated mask should fully cover the distracting target and have a unified standard to allow further usage. Thus, we propose to build a user-orient inpainting system that is both efficient and accurate. In Section 5.2, we proposed a fast object detector that could be applied as a pre-processing procedure to obtain the area of the distracting targets. The proposed MPSSD could output the bounding boxes of all the detected objects/persons in the images, and the user could point out the target within these bounding boxes. Next, we could apply a post-processing segmentation tool to extract the mask area. While during the system building, we found it is more efficient to adopt semantic/instance segmentation algorithms for the mask extraction. We

will introduce this pre-processing step in detail next section.

5.3.1 The Pre-processing Step

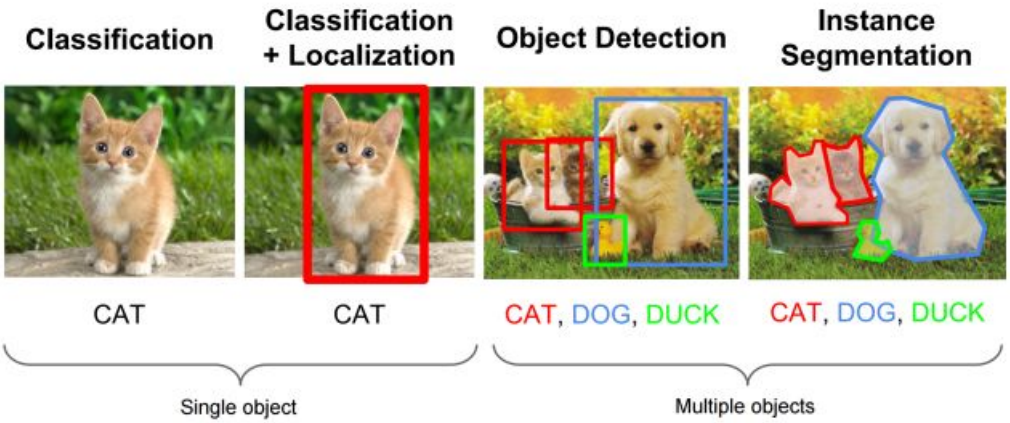


Fig. 5.10: The difference between object detection and instance segmentation. Object detection combines object localization and categorization, and instance segmentation predicts a binary mask on top of the detection results.

In order to generate masks of target objects in an end-to-end manner, we need to do detection first. Object detection refers to the method of localizing and classifying all the objects present in the image, and the output consists of bounding boxes and labels of detected objects. As a more challenging task, instance segmentation requires the model to detect all the objects and segment each instance precisely. Fig. 5.10 indicates the differences between object detection and instance segmentation, where instance segmentation needs to predict a binary mask of the detected objects in terms of foreground and background. He *et al.* [110] presented a flexible and efficient instance segmentation framework called Mask R-CNN. Their model follows the two-stage object detection method Faster R-CNN as introduced in Section 5.2.1 and extends a new branch to predict instance mask. Mask R-CNN has achieved great success and become a backbone for the whole industry. However, the segmentation methods usually predict the pixel labels on down-sampled feature grids (i.e., 28×28 in instance segmentation), which could result in the ambiguous labeling in the contour area. To address this, PointRend [125] borrows the idea from "rendering" in computer graphics. It adopts a subdivision strategy to select a set of points to compute labels. In specific, based on the coarse output of a lightweight segmentation head, PointRend selects a set of points and predicts the label of these points independently through a single multi-layer perceptron (MLP), and these predictions are used to refine the coarse output (refer to Fig. 5.11). The above refining is conducted in an iterative process from a low-resolution grid to the desired-resolution grid. PointRend can be easily

plugged into Mask R-CNN, and it could reach a more accurate segmentation result.

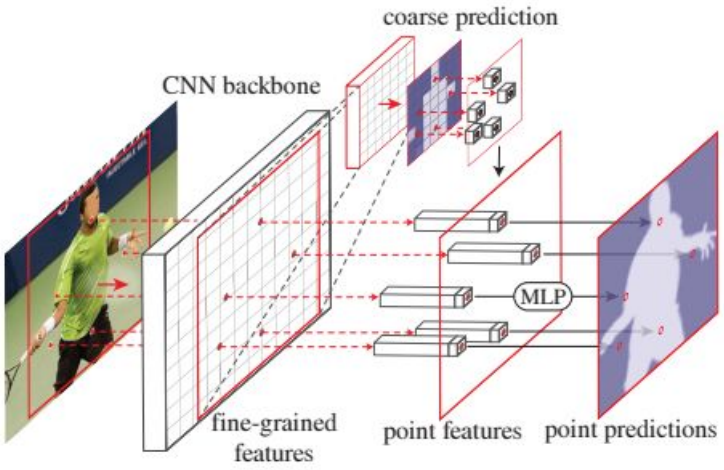


Fig. 5.11: The application of PointRend on instance segmentation [125]. The network first computes a coarse prediction for the object (red box). The PointRend selects a set of points (red dots) and predicts these points through a MLP model. The input features of the MLP are a combination of fine-grained features and coarse prediction. Finally, these predictions are used to obtain a refined prediction map.

5.3.2 The Pipeline

This section introduces the model building of our proposed user-oriented inpainting pipeline and the experimental details. We apply the idea of divide and conquer to build the end-to-end system. As mentioned above, we convert the mask generation into an instance segmentation problem. Given the requirement for high-quality masks for image inpainting, we adopt the state-of-the-art PointRend algorithms built on top of Mask R-CNN backbone. After the pre-processing step, we could extract the masks of detected foregrounds from the segmented predictions. At last, these masks and inputs could be used for object removal through any inpainting methods.

In Fig. 5.12, we depict the entire pipeline of our user-oriented inpainting system. The input is simply a full image with a target signal from the user (as the red dot in the figure), which indicates the distracting object/person to be deleted. The input signal can be a click on the target area. We then send the input image to a trained Mask R-CNN model with PointRend integrated on the mask prediction head. After inference, the instance segmentation results with mask predictions are obtained, including all the detected foreground objects. Next, we extract the foreground binary map with the same format as inpainting masks (value equals 1 is foreground and 0 is background). Then, we pick the selected target from the user input and generate the mask. Noteworthy, we processed the dilation

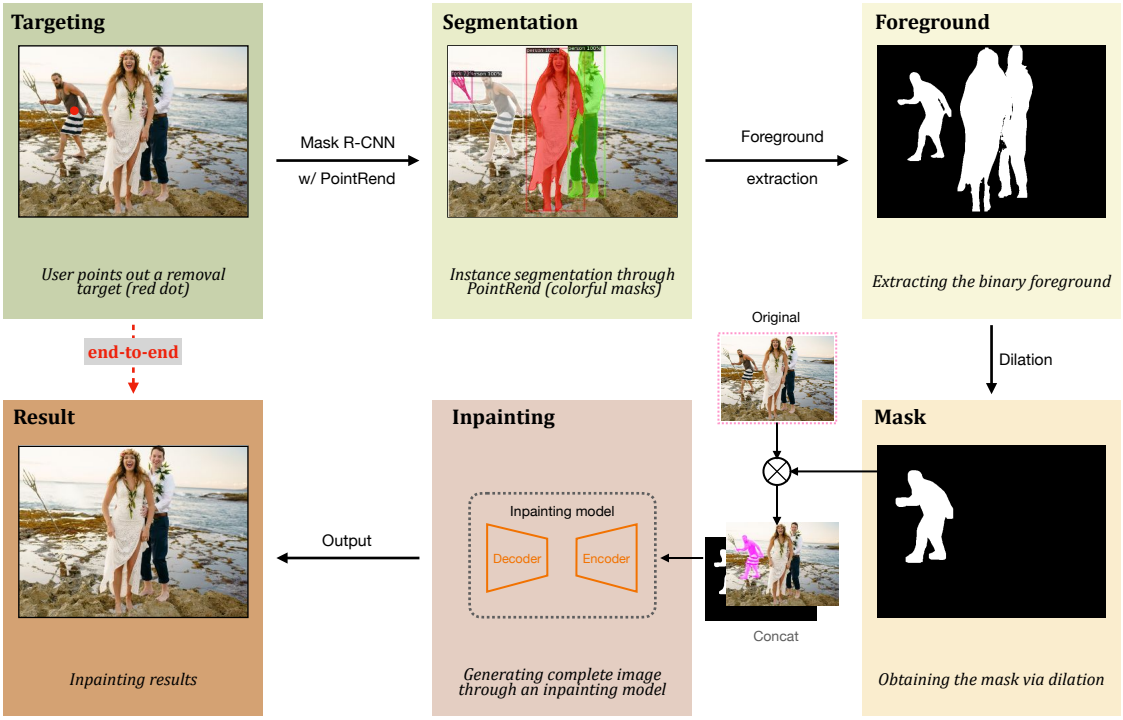


Fig. 5.12: The pipeline of our efficient object removal system. Our ene-to-end pipeline allows the user to indicate the target to be removed and generates the final results without the target. The pipeline contains five steps: receiving user signal, adopting instance segmentation, extracting foreground, trimming mask, and obtaining the result from the inpainting model.

operation before sending it to the next step. The reason is, as mentioned above in Fig. 5.8, we need a fully covered mask to guarantee higher-quality inpainting. The following steps are like all the inpainting models, where the mask is multiplied by the original image pixel-wise, and their concatenation is sent to the inpainting model. Finally, we obtain the inpainted image with the target deleted. Our pipeline is designed in an end-to-end fashion. Algorithm 1 describes the fast object removal pipeline in details.

Algorithm 1: A fast object removal pipeline

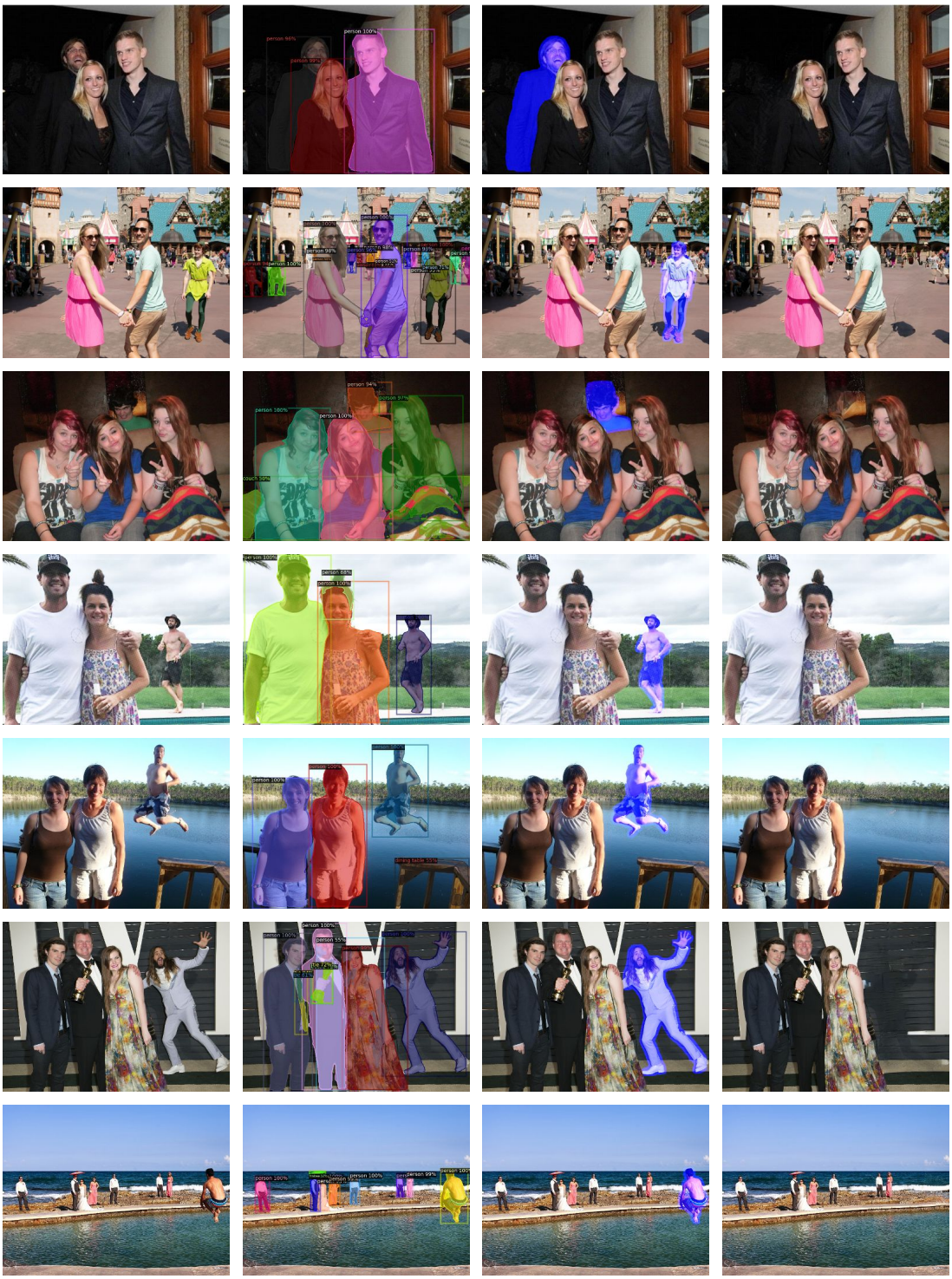
input : Input image i, user signal t of removal target
output: Result image y with target removed

initialization;
while *Receive target signal* t **do**
    Get instance segmentation map $S = PointRend(i)$ by PointRend model
    if *signal* $t \in S$ **then**
        Extract binary foregorund map $F = Binary(S)$
        Get the target foreground $F_x \in F$
        Update target mask by dilation $m_x = Dilation(F_x)$
        Get result from inpainting network $y = G(concat[x \otimes m, m])$
    else
        Go back to the beginning of receiving new singnal
    end
end

5.3.3 Experimental Resutls

This section introduces the experiment details and our experimental results. We use the officially released model from Detectron2 [126]. The trained model uses the backbone Resnet50 + FPN to maintain high accuracy and efficiency. The instance segmentation model is trained on COCO 2017 dataset with images in 640×480. The inpainting model is from Section 3 and is trained on Places2 dataset. To guarantee efficiency, we replace the bottom structural image layer with the one trained on low-resolution data. Although with a bit of sacrifice on the accuracy, we can reach a much higher running speed on mobile devices. The 5×5 kernel is applied in the dilation stage.

We developed a brand new album dataset called PB-101. This dataset contains 101 images collected from the Internet. We aim to collect the pictures with persons, which should look as like album pictures as possible. Each photo includes several people and they cover more than half of the area of the image. We randomly crop all the photos with width at 640 and height at 480. Before testing the entire system, we first adopt the PointRend on the dataset and have validated that all the foregrounds are successfully segmented. Our system is already integrated into the Ali Yunpan AI Album system. Ali



Original Segmentation Masked Result

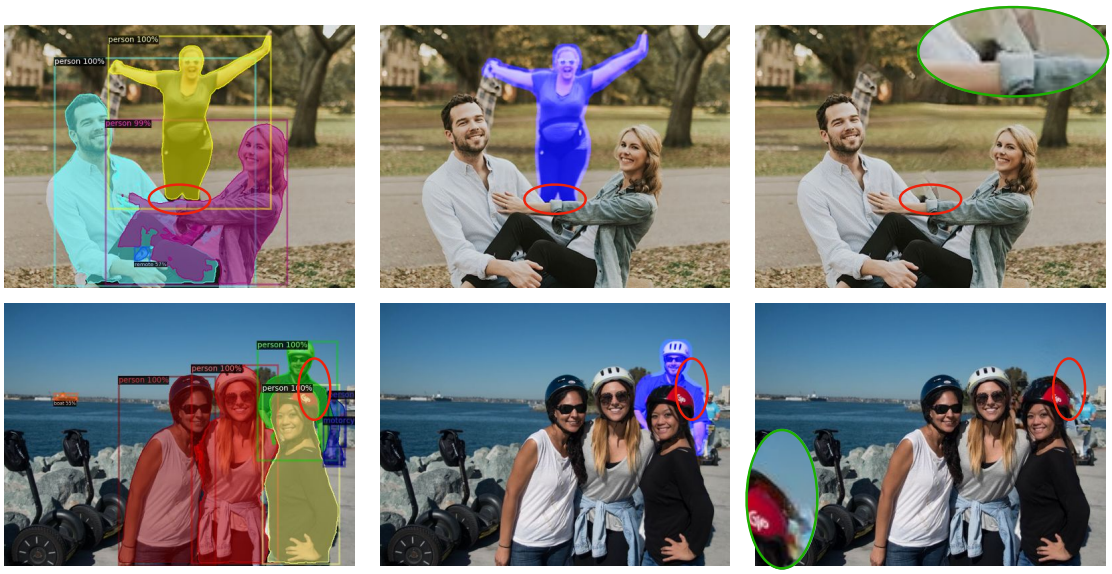
Original Segmentation Masked Result

Fig. 5.13: People deletion examples of our fast object removal inpainting system. From left to right column are original images, instance segmentation results from [125], masked inputs, and the output results of our system.

Cloud backs the pipeline, and the results can be retrieved in real-time. The object removal/people deletion results on PB-101 are shown in Fig. 5.13.

Failure Cases and Future Work

There exist some failure cases due to three main reasons. First, in some photos, objects are too closely connected, and the generated mask may cover the neighboring object/person. The results under this scenario may get deteriorated pixels on the neighboring objects, as shown in Fig. 5.14. We might need self-adjusting mask generation algorithms to address this issue instead of dilation with fixed settings.



(a) Segmentation (b) Masked (c) Result

Fig. 5.14: Failure cases due to the distraction of neighboring objects. The adjacent parts are highlighted for a clear view.

Second, due to the album photos being captured by different devices with different cameras, some depth of field contents are complex for the current inpainting model to recognize, see in Fig 5.15. One possible solution is to fine-tune the model trained in a

large-scale dataset with more album data.

(a) Masked (b) Result

Fig. 5.15: Failure cases due to the discrepancy between training and testing data. The figure shows that the depth of field area is not well-restored since the lack of similar samples in the training data.

The last common failures are caused by failed segmentation, see in Fig. 5.16, people's limbs or other body parts are not segmented correctly due to occlusion, illumination, or under scarce situations. Even though the segmented area is removed successfully, the left body parts make the results look strange. We are searching for more accurate segmentation algorithms to handle these peculiar situations.

(a) Segmentation (b) Masked (c) Result

Fig. 5.16: Failure cases due to the failed segmentation. The failures in segmentation will generate incorrect masks, affecting the final inpainting results.

5.3.4 Conclusion

In conclusion, we propose an efficient object removal system. In a real-life scenario, a traditional system could not efficiently generate unified masks, which limits the application and the generating quality of the image inpainting system. In our system, we send the user's input signal to an instance segmentation-based model, i.e., PointRend, and obtain a high-quality mask of the target objects. In an end-to-end manner, we process the inpainting using the obtained mask to generate the final result. Qualitative validations on our developed PB-101 album dataset have shown the usability of the proposed system.

5.4 Structural Guided Inpainting Module

In a user-oriented image inpainting system, not only is it essential to fulfill the user's demand efficiently and automatically, but it also should provide the function for the user to control the output. In many creative design tasks, filling the targeting area based on users' input guidance is needed. It is also part of interactive image inpainting, as mentioned in the pioneering work [6, 14].

At the same time, image inpainting has the challenge of distracting adjacent foreground objects, which may seriously deteriorate the inpainting quality. Specifically, many current inpainting applications have the effect of an adjacent foreground object creating illogical completion patterns. As mentioned in Section 5.3.3, usually in object removal, due to the area to be completed being adjacent to a foreground object, existing approaches are ambiguous about which pixels should be used to fill the hole. The distraction from those foreground objects would significantly deteriorate the inpainting quality. This phenomenon frequently occurs when inpainting photos that contain many foreground objects. To solve the distraction problem, the inpainting required guidance on which pixels should be referred to and which should not. We often have a chance to utilize external information that could either be given by users or generated through other upstream algorithms.

Therefore, we propose an attention-based structural guided module to address above problems. The contextual attention mechanism [8] calculates the patch similarity between the unseen and seen features and use it as soft weights to guide pixel filling. Inspired by this, we propose a structural guided module as a hard weighting layer, filtering the distracting patches and keeping the demanded ones based on the user input guidance map. Through our proposed module, the user has interactive control in our user-oriented inpainting system.

In this section, we introduce structure-guided image inpainting, our module design, the experimental results, and lastly, we draw a conclusion and future works.

5.4.1 Structure-guided Inpainting

Image structure knowledge has been widely exploited to guide inpainting. EdgeConnect [10] proposes a two-stage method with an edge generator to fill holes with edges followed by an image generator using the first stage's output as prior to synthesize final inpainteing results. Some recent works utilize different structures, i.e. semantic maps [11, 28], appearance flow [29], and gradient maps [127]. These priors are generated through networks or learned with image features. Although more semantically meaningful contents can be generated under guidance, they often require significant computational resources.

5.4.2 Module Design

Recent attention-based inpainting suffers from foreground distraction which may synthesize unwanted patches within holes. The attention is usually acquired through region affinity between known and unknown region on feature map [8, 128]. They adopt a match-and-attend mechanism. The affinity is calculated via the cosine similarity,

$$s_{i,j} = \langle \frac{p_i}{\|p_i\|}, \frac{p_j}{\|p_j\|} \rangle, \tag{5.9}$$

where p_i is the i_{th} patch extracted from known regions, p_j is the j_{th} patch from unknown regions. To obtain the attention score, the Softmax operation is used,

$$a_{i,j} = \frac{e^{s_{i,j}}}{\sum_{n=1}^{N} e^{s_{i,j}}}, \tag{5.10}$$

where N is the number of known patches. In the following attend stage, the corrupted regions are filled with known patches weighted by attention scores,

$$p'_j = \sum_{i=1}^{N} a_{i,j} p_i, \tag{5.11}$$

where p'_j is the j_{th} synthesised patch in the hole. These patches are exploited to generate a new feature map as the output of attention layer.

When deleting unwanted object & people, this mechanism may be disturbed by complex patches of the known region. Under such scenarios, we only want to fill holes with particular background patches. The original attention regards the known region patch without considering whether it belongs to the foreground or background. Thus, we devise a novel structural guidance module to solve this disturbance problem. As shown in Fig. 5.17, this module includes one guided attention layer and one contextual attention layer from [8]. The guided attention layer receives a guidance map and generates a guided signal, which helps filter out distracting foreground features. This signal is used to guide

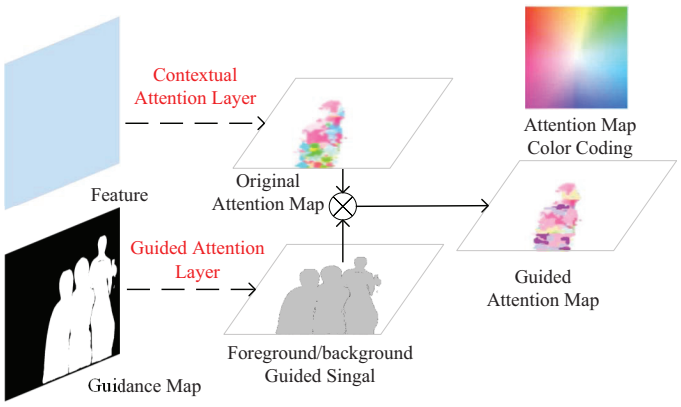


Fig. 5.17: The structural guided module. The feature is used to compute attention score as in [8], and the guidance map is used to guide which features should be referred to. The refined attention score is obtained through the multiplication of these two outputs. See the attention map color coding for reference.

the above cosine similarity in (5.9),

$$s'_{i,j} = s_{i,j} \odot G, \tag{5.12}$$

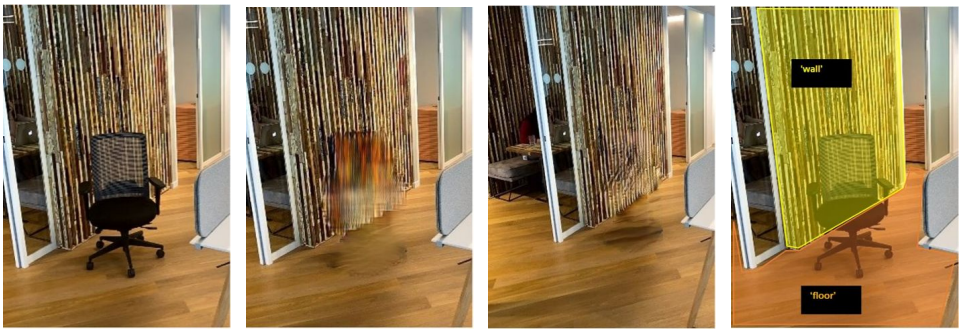
where G is guided signal convert from input semantic map, $s'_{i,j}$ is our guided similarity without foreground distraction. The final synthesized feature is generated by following Softmax operation and match stage as in (5.11).

Notably, our structural guidance is performed in the inference stage and does not need any training.

5.4.3 Experimental Resutls

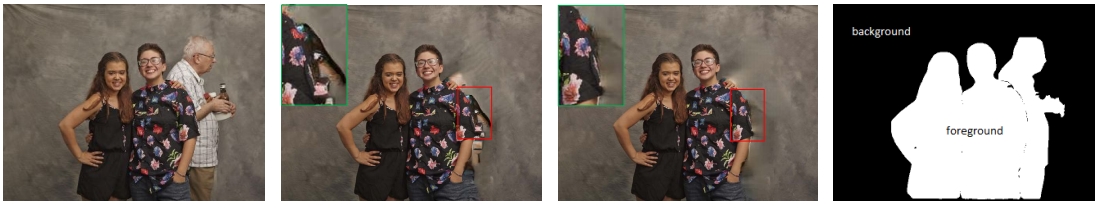
We built the structural guided module on top of the current inpainting model. We adopted the model proposed in Chapter 3 to validate its effectiveness. Our structural guided module accepts various inputs, including segmentation maps and user inputs. For input of segmentation maps, we label all the recognized objects and people as foregrounds and the rest of the regions as background. For user input, the user is required to draw the area of all the foregrounds, and we follow the same labeling step as for segmentation maps. Then, after the automatic labeling, these generated guidance maps can tell apart foreground and background. We send the generated guidance maps to our structural guided module for further guidance.

An example of a real-world scenario with a user input guidance map is shown in Fig. 5.18. The visually unpleasing pixels within the hole are removed because our structural guided module can filter out distracting pixels from the irrelative area. Also, the ambiguous boundary has been addressed.



(a) Input (b) Current result (c) Result with structural guidance (d) Guidance map

Fig. 5.18: An example of a structural guided module with user input guidance. The input guidance indicates the boundary between the wall and floor behind the chair as in (d), and our structural guided module could leverage this information and generate the correct result.



(a) Input (b) Current result (c) Result with structural guidance (d) Guidance map

Fig. 5.19: An example of applying structural guided module on image suffering distraction of foreground. With the external input of (d), we could generate result in (c) with our structural guided module.

In addition, to verify the effectiveness of removing neighboring distractions, we leverage the algorithms [125] as input to generate our people-shaped guidance maps. The generated guidance maps have the same format as masks, where foreground and background are labeled separately. Fig. 5.19 shows a scenario of object removal, where we want to preserve the two posing for the camera but erase the person walking behind them from the photo. As shown in Fig. 5.19(b), after masking off the pixels of that person, the existing method tends to fill the left side of the hole with incorrect pixels, by drawing information from the foreground person, instead of drawing from the desired background area. Due to the area to be completed being adjacent to a foreground object, the existing approach is ambiguous about which pixels should be used to fill the hole. The distraction from those foreground objects would significantly deteriorate the inpainting quality. After employing our attention mechanism, the method could filter out the distracting objects and fill the hole only with information from background pixels (as shown in Fig. 5.19(c)).

5.4.4 Conclusion

This section proposes an attention-based structural guided module that exploits user inputs to guide the inpainting. We show that the proposed module allows the user to remove foreground distractions, edit image layout and interactively manipulate the generated result. This module could improve the inpainting quality and be essential to our efficient user-oriented inpainting system. For future work, we intend to build a more mature mechanism that could accept complex guidance, e.g. mixed curves and lines. We also hope to build a mechanism with diversified outputs based on automatically generated guidance, allowing users to pick the one they prefer.

5.5 Summary

This chapter is dedicated to practical image inpainting. Two modules are proposed to address the problem of applying image inpainting in real-world interactive systems with the integration of related techniques. A multi-path fusion object detector is first introduced as the pre-processing step of the proposed efficient object removal system. The following section presents the object removal system precisely with experimental results. Then, the attention-based guided inpainting module is elaborated, and there are results on the developed dataset PB-101.

Chapter 6

Conclusion and Future Work

This chapter provides the conclusion of this dissertation. First, we review the framework of the thesis, which starts with a literature study of image inpainting methods and the challenges of current models. Among these challenges, we focus on the following: 1) Inadequate learning of image structure and texture; 2) Not efficiently utilizing the generator features; 3) Missing an efficient user-oriented image inpainting framework. To this end, we proposed a multi-scale pyramid, an encoder-decoder architecture with feature fusion integrated, a fast object removal system, and a structural guided module. These models solved the above problems effectively and have been discussed in the previous chapters. We present the future outlook based on our current framework in the remaining part.

6.1 Contributions of the Thesis

Our research dedicates to addressing image inpainting by using deep neural networks. We introduce traditional image inpainting methods before the era of deep learning. The traditional diffusion and patch-based methods could handle easy restoration but were failed on complicated cases. However, the emergence of deep generative models has boosted the field significantly. We discussed the gap between traditional and deep learning-based groups that lie in the capability of learning from image global context. Taking a step further, we focus on how to apply deep generative models to image inpainting adaptively. The most evident challenge is eliminating artifacts and blur in complex scenarios. After reviewing recent deep learning-based models, we found that a lack of global image structures causes the artifacts, and the blur is due to the missing image structure. We managed this problematic situation with more advanced architecture and techniques. Moreover, we focus on efficient interactive image inpainting, which plays an essential role in its applications. Overall, we have contributed to these inpainting problems from both a theoretical and practical perspective.

Based on pyramid architecture in the image generation task, we propose a multi-scale

pyramid generator model with a joint-training scheme. In our model, we separate the learning of image global and local information from lower to higher levels. Our bottom layer utilizes edge-preserved structural images as supervision for emphasizing the learning of image structures. Next, the image textures are learned in the higher layers using high-resolution images as inputs. Finally, the learned structure is passed to higher layers with image details, generating the final results. From our "structure first detail next" strategy, the model alleviates the conflicts between structure and textures learning of a single encoder-decoder model. Another benefit of our proposed multi-scale model is the progressive increase of hole size relative to the model receptive size could generate a plausible result on larger masks. Our method is also applicative to high-resolution image inpainting, thanks to the inherent merit of pyramid architecture. Qualitative and quantitative comparison of the standard and high-resolution 2K image datasets have shown the proposed method's effectiveness.

Inspired by the feature fusion technique, we also proposed an approach to better model image structure and texture. The vanilla encoder-decoder model uses the low-level encoder feature to model image structures and obtains the global semantics in high-level decoder features. The inefficient usage of the extracted features could cause ambiguous patches and semantically meaningless pixels. We manage to aggregate these information through a feature fusion mechanism and thus acquire the enhanced features for high-quality inpainting. First, we fuse the encoder feature with the same-size decoder feature by the attention feature fusion module, and the fused feature is aggregated to the decoder through skip connections. This step effectively compensates the structure information to the late semantics and could alleviate the blurry patches. Second, we upgrade the multi-dilated bottleneck with residual learning. Similar to the previous step, we adopt the attention feature fusion to aggregate these features with different receptive fields to obtain the feature with enhancement on image structure. We have validated our method on both natural and facial datasets. These two feature fusion modules improve the feature representative of the image inpainting task.

Finally, we propose an efficient user-oriented system dedicated to practical image inpainting. Our approach includes two different parts which focus on dealing with various issues. The first efficient object-removal system allows the user to remove the distraction target by only clicking the target object on the image. The previous models need the user to draw the mask of distracting target by hand, which is tedious and may introduce inaccurate results due to the low-quality masks. We first introduce our efficient object detection algorithms to address these problems, which has inspired us to build a detection-based automatic masks generation approach. In our final object reomval system, we apply the instance segmentation algorithms to generate the high-quality mask efficiently. The gen-

erated mask is sent to the following inpainting model with the input image and outputs the final result to the user. We have validated the effectiveness of our system on our developed album dataset PB-101. On the other hand, we design an attention-based structural guided module for interactive image inpainting. In our module, we calculate the attention score based on the user input guidance map and recommend the model on which part of the pixels it should refer to. We have tested it on real-world cases, and the results show that the structural guided module has the ability of interactive inpainting based on guidance map. Meanwhile, our structural guided module could also eliminate the distracting foreground issue, often when the removal target is too close to its adjacent objects.

6.2 Future Works

This section elaborates the future work associating with the aforementioned chapters.

Chapter 3 Image Inpainting with Pyramid Generator

Although the experimental results have shown the potential of tackling high-resolution image inpainting using the proposed method, high/ultra-resolution inpainting still be an intractable problem. Finding a way to use the computation resources carefully and not lose any high-resolution information during the inpainting remains challenging. We try to follow our multi-scale architecture and find a way to model these high-resolution in a generalized manner.

Meanwhile, current image inpainting methods rely on quantitative evaluation on metrics such as PSNR, SSIM, and FID. These metrics could not reflect the realistic generation results. We expect to develop a novel metric that could better reflect human perception.

Chapter 4 Image Inpainting with Attention Feature Fusion

It is expected to follow the current hypothesis to address the inadequate learning of structure texture in current models. Although we have addressed several problems with our proposed models, there are still many challenges to be tackled.

First, despite the powerful image generation ability of current deep generative models, there still exists inefficiency in learning image internal statistics. The recent transformer-based model might be a possible improvement option to understand this information better, especially the global context among pixels. Optimizing the transformer-based image inpainting model could be a pivot research field. Also, in a transformer-based model, it is worth researching the utilization of the learned relationships among tokenized image patches wisely.

Second, for structure and texture modeling in image inpainting, I intend to focus on the interpretability of current models and further improve the inpainting quality.

Chapter 5 Efficient User-oriented Image Inpainting

I believe building a more robust image inpainting system with advanced algorithms is worth building. The proposed method leverages object detection/segmentation to make an intelligent system, and this system has benefited users in real-life. The system still has defects and has space to improve. For example, a more adaptive interactive system should accept many types of user inputs with rich information. Predictably, integrating more cutting-edge techniques in computer vision, and natural language processing would result in a more intelligent system.

Publication List

Shuyi Qu, Kaizhu Huang, Amir Hussain, Yannis Goulermas: MPSSD: Multi-Path Fusion Single Shot Detector. IJCNN 2019: 1-6

Shuyi Qu, Kaizhu Huang, Amir Hussain, Yannis Goulermas: A Multipath Fusion Strategy Based Single Shot Detector. Sensors 21(4): 1360 (2021)

Shuyi Qu, Zhenxing Niu, Kaizhu Huang, Jianke Zhu: Structure First Detail Next: Image Inpainting with Pyramid Generator. article under revision as of July 2022 in Pattern Recognition

Shuyi Qu, Kaizhu Huang, Qiufeng Wang, Bin Dong: Rethinking Image Inpainting with Attention Feature Fusion. article sumbitted to ICONIP 2022

References

[1] B. Museum of Fine Arts, E. Usui, J. Gaviria, and R. Newman, *Conservation and Care of Museum Collections*, ser. MFA highlights. MFA publications, 2011. [Online]. Available: https://books.google.com.hk/books?id=ieRlPgAACAAJ

[2] B. Jähne, *Spatio-temporal image processing: theory and scientific applications*. Springer, 1993.

[3] S. Zarif, I. Faye, and D. Rohaya, "Image completion: Survey and comparative study," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 29, no. 03, p. 1554001, 2015.

[4] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman, "Patchmatch: A randomized correspondence algorithm for structural image editing," *ACM Transactions on Graphics*, vol. 28, no. 3, p. 24, 2009.

[5] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2536–2544.

[6] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Free-form image inpainting with gated convolution," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 4471–4480.

[7] Z. Yi, Q. Tang, S. Azizi, D. Jang, and Z. Xu, "Contextual residual aggregation for ultra high-resolution image inpainting," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[8] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Generative image inpainting with contextual attention," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5505–5514.

[9] Y. Zeng, J. Fu, H. Chao, and B. Guo, "Learning pyramid-context encoder network for high-quality image inpainting," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1486–1494.

[10] K. Nazeri, E. Ng, T. Joseph, F. Qureshi, and M. Ebrahimi, "Edgeconnect: Structure guided image inpainting using edge prediction," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, Oct 2019.

[11] Y. Song, C. Yang, Y. Shen, P. Wang, Q. Huang, and C.-C. J. Kuo, "Spg-net: Segmentation prediction and guidance network for image inpainting," *arXiv preprint arXiv:1805.03356*, 2018.

[12] Y. Zeng, J. Fu, H. Chao, and B. Guo, "Aggregated contextual transformations for high-resolution image inpainting," *IEEE Transactions on Visualization and Computer Graphics*, 2022.

[13] R. Suvorov, E. Logacheva, A. Mashikhin, A. Remizova, A. Ashukha, A. Silvestrov, N. Kong, H. Goka, K. Park, and V. Lempitsky, "Resolution-robust large mask inpainting with fourier convolutions," in *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, 2022, pp. 2149–2159.

[14] Y. Zeng, Z. Lin, J. Yang, J. Zhang, E. Shechtman, and H. Lu, "High-resolution image inpainting with iterative confidence feedback and guided upsampling," in *Proceedings of the European Conference on Computer Vision*. Springer, 2020, pp. 1–17.

[15] A. A. Efros and T. K. Leung, "Texture synthesis by non-parametric sampling," in *Proceedings of the seventh IEEE international conference on computer vision*, vol. 2. IEEE, 1999, pp. 1033–1038.

[16] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, "Image inpainting," in *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, 2000, pp. 417–424.

[17] Z. Qin, Q. Zeng, Y. Zong, and F. Xu, "Image inpainting based on deep learning: A review," *Displays*, vol. 69, p. 102028, 2021.

[18] D. Tschumperlé, "Fast anisotropic smoothing of multi-valued images using curvature-preserving pde's," *International Journal of Computer Vision*, vol. 68, no. 1, pp. 65–82, 2006.

[19] T. F. Chan and J. Shen, *Image processing and analysis: variational, PDE, wavelet, and stochastic methods*. SIAM, 2005.

[20] X.-C. Tai, S. Osher, and R. Holm, "Image inpainting using a tv-stokes equation," in *Image Processing based on partial differential equations*. Springer, 2007, pp. 3–22.

[21] G. Sridevi and S. Srinivas Kumar, "Image inpainting based on fractional-order nonlinear diffusion for image reconstruction," *Circuits, Systems, and Signal Processing*, vol. 38, no. 8, pp. 3802–3817, 2019.

[22] A. Criminisi, P. Pérez, and K. Toyama, "Region filling and object removal by exemplar-based image inpainting," *IEEE Transactions on Image Processing*, vol. 13, no. 9, pp. 1200–1212, 2004.

[23] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[24] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," *Advances in Advances in Neural Information Processing Systems*, vol. 27, 2014.

[25] V. Jain and S. Seung, "Natural image denoising with convolutional networks," *Advances in Advances in Neural Information Processing Systems*, vol. 21, 2008.

[26] G. Liu, F. A. Reda, K. J. Shih, T.-C. Wang, A. Tao, and B. Catanzaro, "Image inpainting for irregular holes using partial convolutions," in *Proceedings of the European Conference on Computer Vision*, 2018, pp. 85–100.

[27] H. Liu, B. Jiang, Y. Song, W. Huang, and C. Yang, "Rethinking image inpainting via a mutual encoder-decoder with feature equalizations," in *Proceedings of the European Conference on Computer Vision*. Springer, 2020, pp. 725–741.

[28] L. Liao, J. Xiao, Z. Wang, C.-W. Lin, and S. Satoh, "Guidance and evaluation: Semantic-aware image inpainting for mixed scenes," in *Proceedings of the European Conference on Computer Vision*. Springer, 2020, pp. 683–700.

[29] Y. Ren, X. Yu, R. Zhang, T. H. Li, S. Liu, and G. Li, "Structureflow: Image inpainting via structure-aware appearance flow," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 181–190.

[30] C. Yang, X. Lu, Z. Lin, E. Shechtman, O. Wang, and H. Li, "High-resolution image inpainting using multi-scale neural patch synthesis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 6721–6729.

[31] D. H. Ballard, "Modular learning in neural networks." in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 647, 1987, pp. 279–284.

[32] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[33] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[34] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Globally and locally consistent image completion," *ACM Transactions on Graphics*, vol. 36, no. 4, pp. 1–14, 2017.

[35] A. Telea, "An image inpainting technique based on the fast marching method," *Journal of graphics tools*, vol. 9, no. 1, pp. 23–34, 2004.

[36] P. Pérez, M. Gangnet, and A. Blake, "Poisson image editing," in *ACM SIGGRAPH*, 2003, pp. 313–318.

[37] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[38] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[39] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan. arxiv 2017," *arXiv preprint arXiv:1701.07875*, vol. 30, no. 4, 2017.

[40] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," *Advances in Advances in Neural Information Processing Systems*, vol. 30, 2017.

[41] H. Liu, B. Jiang, Y. Xiao, and C. Yang, "Coherent semantic attention for image inpainting," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 4170–4179.

[42] T. Wang, H. Ouyang, and Q. Chen, "Image inpainting with external-internal learning and monochromic bottleneck," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021, pp. 5120–5129.

[43] J. Gu, Y. Shen, and B. Zhou, "Image processing using multi-code gan prior," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 3012–3021.

[44] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[45] X. Wang, K. Yu, C. Dong, and C. C. Loy, "Recovering realistic texture in image super-resolution by deep spatial feature transform," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 606–615.

[46] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3213–3223.

[47] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 million image database for scene recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[48] L. Xu, C. Lu, Y. Xu, and J. Jia, "Image smoothing via l0 gradient minimization," in *Proceedings of the 2011 SIGGRAPH Asia conference*, 2011, pp. 1–12.

[49] L. Xu, Q. Yan, Y. Xia, and J. Jia, "Structure extraction from texture via relative total variation," *ACM Transactions on Graphics*, vol. 31, no. 6, pp. 1–10, 2012.

[50] L. Chi, B. Jiang, and Y. Mu, "Fast fourier convolution," *Advances in Advances in Neural Information Processing Systems*, vol. 33, pp. 4479–4488, 2020.

[51] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *Proceedings of the European Conference on Computer Vision*. Springer, 2016, pp. 694–711.

[52] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of the IEEE International Conference on Computer Vision*, December 2015.

[53] T. Karras, T. Aila, S. Laine, and J. Lehtinen, "Progressive growing of gans for improved quality, stability, and variation," in *Proceedings of International Conference on Learning Representations*, 2018, international Conference on Learning Representations, ICLR ; Conference date: 30-04-2018 Through 03-05-2018. [Online]. Available: https://iclr.cc/Conferences/2018

[54] X.-J. Mao, C. Shen, and Y.-B. Yang, "Image restoration using convolutional auto-encoders with symmetric skip connections," *arXiv preprint arXiv:1606.08921*, 2016.

[55] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang *et al.*, "Photo-realistic single image super-resolution

using a generative adversarial network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4681–4690.

[56] C. Doersch, S. Singh, A. Gupta, J. Sivic, and A. Efros, "What makes paris look like paris?" *ACM Transactions on Graphics*, vol. 31, no. 4, 2012.

[57] P. Gronát, M. Havlena, J. Sivic, and T. Pajdla, "Building streetview datasets for place recognition and city reconstruction," *Research Reports of CMP, Czech Technical University in Prague*, 2011.

[58] E. Agustsson and R. Timofte, "Ntire 2017 challenge on single image super-resolution: Dataset and study," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 126–135.

[59] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," *Advances in Advances in Neural Information Processing Systems*, vol. 30, 2017.

[60] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 586–595.

[61] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2818–2826.

[62] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[63] S. Y. Kim, K. Aberman, M. Kim, and O. Liba, "Zoom-to-inpaint: Image inpainting with high frequency details," in *arXiv preprint arXiv: 2012.09401*, 2020.

[64] N. Rahaman, A. Baratin, D. Arpit, F. Draxler, M. Lin, F. A. Hamprecht, Y. Bengio, and A. Courville, "On the spectral bias of neural networks," in *International Conference on Machine Learning*, 2020.

[65] M. Eitz, J. Hays, and M. Alexa, "How do humans sketch objects?" in *ACM Transactions on Graphics*, 2012.

[66] C. Ballester, M. Bertalmio, V. Caselles, G. Sapiro, and J. Verdera, "Filling-in by joint interpolation of vector fields and gray levels," *IEEE Transactions on Image Processing*, vol. 10, no. 8, pp. 1200–1211, 2001.

[67] N. Wang, S. Ma, J. Li, Y. Zhang, and L. Zhang, "Multistage attention network for image inpainting," *Pattern Recognition*, vol. 106, p. 107448, 2020.

[68] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2650–2658.

[69] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional gans," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8798–8807.

[70] T. R. Shaham, T. Dekel, and T. Michaeli, "Singan: Learning a generative model from a single natural image," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 4570–4580.

[71] C. Yang, X. Lu, Z. Lin, E. Shechtmanz, O. Wang, and H. Li, "High-resolution image inpainting using multi-scale neural patch synthesis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[72] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, "Places: A 10 million image database for scene recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[73] C.-H. Lee, Z. Liu, L. Wu, and P. Luo, "Maskgan: Towards diverse and interactive facial image manipulation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 5549–5558.

[74] Y. Dai, F. Gieseke, S. Oehmcke, Y. Wu, and K. Barnard, "Attentional feature fusion," in *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*, 2021, pp. 3560–3569.

[75] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[76] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2117–2125.

[77] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-assisted Intervention*. Springer, 2015, pp. 234–241.

[78] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2017.

[79] K. Sun, B. Xiao, D. Liu, and J. Wang, "Deep high-resolution representation learning for human pose estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5693–5703.

[80] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Advances in Neural Information Processing Systems*, vol. 30, 2017.

[81] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7794–7803.

[82] I. Bello, B. Zoph, A. Vaswani, J. Shlens, and Q. V. Le, "Attention augmented convolutional networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 3286–3295.

[83] X. Li, W. Wang, X. Hu, and J. Yang, "Selective kernel networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 510–519.

[84] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7132–7141.

[85] W. Liu, A. Rabinovich, and A. C. Berg, "Parsenet: Looking wider to see better," *arXiv preprint arXiv:1506.04579*, 2015.

[86] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*. PMLR, 2015, pp. 448–456.

[87] C. Li and M. Wand, "Precomputed real-time texture synthesis with markovian generative adversarial networks," in *Proceedings of the European Conference on Computer Vision*. Springer, 2016, pp. 702–716.

[88] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," *arXiv preprint arXiv:1802.05957*, 2018.

[89] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *International Conference on Machine Learning*, 2010.

[90] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen, "Deep learning for generic object detection: A survey," *International Journal of Computer Vision*, vol. 128, no. 2, pp. 261–318, 2020.

[91] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 580–587.

[92] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1440–1448.

[93] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: towards real-time object detection with region proposal networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2016.

[94] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779–788.

[95] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *Proceedings of the European Conference on Computer Vision*. Springer, 2016, pp. 21–37.

[96] J. Dai, Y. Li, K. He, and J. Sun, "R-fcn: Object detection via region-based fully convolutional networks," in *Advances in Neural Information Processing Systems*, ser. NIPS'16. Red Hook, NY, USA: Curran Associates Inc., 2016, p. 379–387.

[97] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 2010.

[98] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Proceedings of the European Conference on Computer Vision*. Springer, 2014, pp. 740–755.

[99] A. Ghodrati, A. Diba, M. Pedersoli, T. Tuytelaars, and L. Van Gool, "Deepproposal: Hunting objects by cascading deep convolutional layers," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2578–2586.

[100] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg, "Dssd: Deconvolutional single shot detector," *arXiv preprint arXiv:1701.06659*, 2017.

[101] A. Ochoa-Zezzatti, J. Mejía, R. Contreras-Masse, E. Martínez, and A. Hernández, "Ethnic characterization in amalgamated people for airport security using a repository of images and pigeon-inspired optimization (pio) algorithm for the improvement of their results," in *Applications of Hybrid Metaheuristic Algorithms for Image Processing*. Springer, 2020, pp. 105–119.

[102] Z. Li and F. Zhou, "Fssd: Feature fusion single shot multibox detector," *arXiv preprint arXiv:1712.00960*, 2017.

[103] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2004, pp. 1–22.

[104] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1. IEEE, 2005, pp. 886–893.

[105] X. Wang, T. X. Han, and S. Yan, "An hog-lbp human detector with partial occlusion handling," in *Proceedings of the IEEE International Conference on Computer Vision*. IEEE, 2009, pp. 32–39.

[106] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the fisher kernel for large-scale image classification," in *Proceedings of the European Conference on Computer Vision*. Springer, 2010, pp. 143–156.

[107] P. F. Felzenszwalb, R. B. Girshick, and D. McAllester, "Cascade object detection with deformable part models," in *2010 IEEE Computer society conference on computer vision and pattern recognition*. IEEE, 2010, pp. 2241–2248.

[108] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, p. 84–90, May 2017. [Online]. Available: https://doi.org/10.1145/3065386

[109] K. Huang, A. Hussain, Q.-F. Wang, and R. Zhang, *Deep Learning: Fundamentals, Theory and Applications*. Springer, 2019, vol. 2.

[110] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2980–2988.

[111] G. Zhong, S. Yan, K. Huang, Y. Cai, and J. Dong, "Reducing and stretching deep convolutional activation features for accurate image classification," *Cognitive Computation*, vol. 10, no. 1, pp. 179–186, 2018.

[112] X. Yang, K. Huang, R. Zhang, and J. Y. Goulermas, "A novel deep density model for unsupervised learning," *Cognitive Computation*, vol. 11, no. 6, pp. 778–788, 2019.

[113] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2980–2988.

[114] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7263–7271.

[115] ——, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[116] T. Kong, A. Yao, Y. Chen, and F. Sun, "Hypernet: Towards accurate region proposal generation and joint object detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 845–853.

[117] S. Bell, C. Lawrence Zitnick, K. Bala, and R. Girshick, "Inside-outside net: Detecting objects in context with skip pooling and recurrent neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2874–2883.

[118] S. Woo, S. Hwang, and I. S. Kweon, "Stairnet: Top-down semantic aggregation for accurate one shot detection," in *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*. IEEE, 2018, pp. 1093–1102.

[119] S. Liu, D. Huang *et al.*, "Receptive field block net for accurate and fast object detection," in *Proceedings of the European Conference on Computer Vision*, 2018, pp. 385–400.

[120] Y. Pang, T. Wang, R. M. Anwer, F. S. Khan, and L. Shao, "Efficient featurized image pyramid network for single shot detector," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7336–7344.

[121] Q. Zhao, T. Sheng, Y. Wang, Z. Tang, Y. Chen, L. Cai, and H. Ling, "M2det: A single-shot object detector based on multi-level feature pyramid network," in

Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 2019, pp. 9259–9266.

[122] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[123] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1026–1034.

[124] B. Hariharan, P. Arbeláez, L. Bourdev, S. Maji, and J. Malik, "Semantic contours from inverse detectors," in *Proceedings of the IEEE International Conference on Computer Vision*. IEEE, 2011, pp. 991–998.

[125] A. Kirillov, Y. Wu, K. He, and R. Girshick, "Pointrend: Image segmentation as rendering," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9799–9808.

[126] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," https://github.com/facebookresearch/detectron2, 2019.

[127] J. Yang, Z. Qi, and Y. Shi, "Learning to incorporate structure knowledge for image inpainting." in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020, pp. 12 605–12 612.

[128] Y. Song, C. Yang, Z. Lin, X. Liu, Q. Huang, H. Li, and C.-C. Jay Kuo, "Contextual-based image inpainting: Infer, match, and translate," in *Proceedings of the European Conference on Computer Vision*, 2018, pp. 3–19.