Multi-feature Co-learning for Image Inpainting

Jiayu Lin†, Yuan-Gen Wang†∗, Wenzhi Tang‡, and Aifeng Li ‡
†School of Computer Science and Cyber Engineering, Guangzhou University, Guangzhou, China
‡Network Center, Guangzhou University, Guangzhou, China

Abstract—Image inpainting has achieved great advances by simultaneously leveraging image structure and texture features. However, due to lack of effective multi-feature fusion techniques, existing image inpainting methods still show limited improvement. In this paper, we design a deep multi-feature co-learning network for image inpainting, which includes Soft-gating Dual Feature Fusion (SDFF) and Bilateral Propagation Feature Aggregation (BPFA) modules. To be specific, we first use two branches to learn structure features and texture features separately. Then the proposed SDFF module integrates structure features into texture features, and meanwhile uses texture features as an auxiliary in generating structure features. Such a co-learning strategy makes the structure and texture features more consistent. Next, the proposed BPFA module enhances the connection from local feature to overall consistency by co-learning contextual attention, channel-wise information and feature space, which can further refine the generated structures and textures. Finally, extensive experiments are performed on benchmark datasets, including CelebA, Places2, and Paris StreetView. Experimental results demonstrate the superiority of the proposed method over the state-of-the-art. The source codes are available at https://github.com/GZHU-DVL/MFCL-Inpainting.

I. INTRODUCTION

Image inpainting [2] aims at reconstructing damaged regions or removing unwanted regions of images while improving the visual aesthetics of the inpainted images, which has been widely used in low-level vision tasks, such as restoring corrupted photos and object removal. The main challenge of image inpainting is how to generate reasonable structures and realistic textures. Traditional image inpainting, such as patch-based methods [1], [5], fill out the hole with the most similar patch as the to-be-inpainted patch by searching on undamaged region. Since the similarity is computed on pixel domain of the image, these methods fail to generate objects with strong semantic information.

Deep learning-based methods have shown remarkable performance on image inpainting tasks [22], [23], [25], [30]. They can generate semantically consistent results by understanding high-level features of the images. Among these methods, the encoder-decoder architecture has been widely developed since this architecture can extract semantic features and generate visually pleasing contents even if the hole is large. The variants of encoder-decoder architecture like U-Net [32] are also developed for image inpainting, which can enhance the feature connection between the encoder and the decoder. For irregularly corrupted images, Liu et al. [17] and Yu et al. [35]

*Corresponding Email: wangyg@gzhu.edu.cn. This work was supported in part by the National Natural Science Foundation of China (61872099) and in part by the Scientific Research Project of Guangzhou University (YJ2021004).

Fig. 1: Visual results of the proposed method. From left to right: (a) Input images with holes. (b) Inpainted images without SDFF. (c) Inpainted images without BPFA. (d) Inpainted images with SDFF and BPFA. (e) Ground-truth images. We can observe that the inpainted images by our method have visually pleasing content.
Fusion (SDFF) module is designed to reorganize structure and texture features so that their consistency is greatly enhanced. 2) A Bilateral Propagation Feature Aggregation (BPFA) module is designed to capture the connection between contextual attention, channel-wise information, and feature space. This BPFA greatly enhances the connection from local feature to overall consistency. The major contributions of the proposed method are as follows:

- We propose a novel SDFF module. With SDFF, the blur and artifacts around the holes are significantly reduced.
- We propose a novel BPFA module. With BPFA, the inpainted images show more rational structures and more detailed textures.
- Extensive experimental results demonstrate the superiority of the proposed method over the state-of-the-art.

II. RELATED WORKS

Traditional image inpainting methods are usually divided into two main categories: diffusion-based methods [2], [5] and patch-based methods [3], [6]. The first one propagates the appearance of adjacent content to fill out the missing regions. However, due to the limitation of search mechanism on adjacent content, there are obvious artifacts within images when facing large area masks. The second one fills in the missing region with the most similar patch as the to-be-inpainted patch. Although they can capture long-distance information, it is difficult to generate semantically reasonable images due to the lack of high-level structure understanding.

Deep learning-based methods [11], [16], [20], [24], [34], [37] have been widely explored in the field of image inpainting. Pathak et al. [23] firstly developed encoder-decoder architecture and adversarial training for image inpainting. Iizuka et al. [9] overcame the information bottleneck defect by introducing a series of dilated convolution layers. Recently, Nazeri et al. [22] proposed EdgeConnect to generate possible edges and fill in the holes with precondition information. Like EdgeConnect, Xiong et al. [30] designed a similar model by adopting a contour generator as structure prior instead of the edge generator. Ren et al. [25] utilized the edge-preserving smoothing method to obtain sharp edges and low-frequency structures. Yang et al. [33] proposed a multi-task learning framework by introducing structure embedding to generate refined structures. Liu et al. [18] designed a mutual encoder-decoder network to learn the structure and texture features separately. Peng et al. [24] presented the conditional autoregressive network and structure attention module, which can learn the distribution of structure features and capture the distance relationship between structures, respectively. However, the above-reviewed methods do not fully consider the relationship between the structure and texture features, it is difficult to generate the images with reasonable structures and sophisticated textures.

III. PROPOSED METHOD

The overall pipeline of the proposed method is shown in Fig. 2, which is built upon the generative adversarial network. The generative network consists of mutual encoder-decoder, structure and texture branches, Soft-gating Dual Feature Fusion (SDFF), and Bilateral Propagation Feature Aggregation (BPFA). The discriminative network consists of global discriminator and local discriminator. By convention, the generative network aims to generate the inpainted images by co-learning image structures and textures, while the discriminative network aims to distinguish between the inpainted images and real ones. In the following, we describe the proposed network architecture and loss functions in detail.

A. Generator

The generator can be divided into the five parts: 1) The encoder consists of six convolutional layers. The three shallow-layer features are reorganized as structure features to represent image details. Meanwhile, the three deep-layer features are reorganized as structure features to represent image semantics. 2) We adopt two branches to separately learn the structure and texture features. 3) We design a SDFF module to fuse the structure and texture features generated by the above two branches. 4) We design a BPFA module to equalize the features between contextual attention, channel-wise information, and feature space. 5) The skip connection is used to supplement decoder features, which helps synthesize structure and texture branches to produce more sophisticated images.

Structure and Texture Branches. The texture feature reorganized by shallow-layer convolution is denoted as $F_{tc}$ and the structure feature reorganized by deep-layer convolution is denoted as $F_{st}$. In each branch, three parallel streams are used to fill out the corrupted regions at multiple scales. For each stream, we replace the vanilla convolutions with padding based partial convolution in order to better fill in irregular holes. Note that each stream consists of five convolutional layers, and the convolutional kernel sizes of the three streams are $3 \times 3$, $5 \times 5$ and $7 \times 7$, respectively. We can obtain the filled features by first combining the output feature maps of the three streams and then mapping the combined features into the same size of the input feature. Here, we denote $F_{fst}$ and $F_{fct}$ as the outputs of the structure and texture branches, respectively. To ensure that the two branches focus on structures and textures respectively, we use two reconstruction losses, denoted as $L_{st}$ and $L_{tc}$ respectively. The pixel-wise loss is defined as:

$$L_{st} = \| g(F_{fst}) - I_{st} \|_1, \quad (1)$$

$$L_{tc} = \| g(F_{fct}) - I_{gt} \|_1, \quad (2)$$

where $g(\cdot)$ is the convolution operation with the kernel size of 1, which aims to map $F_{fst}$ and $F_{fct}$ to two color images respectively. $I_{gt}$ and $I_{st}$ denote the ground-truth images and its structure image, respectively. We follow [25] by using an edge-preserving smoothing method [31] to generate $I_{st}$.

Soft-gating Dual Feature Fusion (SDFF). This module is designed to better exchange the structure features $F_{fst}$ and texture features $F_{fct}$ generated by the above two branches, respectively. The exchange is implemented by utilizing a soft gating way to dynamically adjust the fusion ratio between the structure and texture features. Fig. 3 illustrates the proposed
Fig. 2: Description of the proposed pipeline. **Generator:** We propose a variant of U-Net architecture to jointly learn image structures and textures. The Soft-gating Dual Feature Fusion (SDFF) module and Bilateral Propagation Feature Aggregation (BPFA) module are designed to refine the generated structures and textures. **Discriminator:** We adopt the local and global discriminators to ensure from local to global content consistency.

\[
G_{ct} = \sigma (SE(k ([F_{ct}, F_{cte}]))) ,
\]
\[
F'_{ct} = \gamma (G_{ct} \odot F_{cte}) \oplus F_{cte},
\]
where \(k(\cdot)\) is a convolution layer with the kernel size of 3 and \(\gamma\) is a learnable parameter. Thus, we can concatenate \(F'_{cte}\) and \(F'_{ct}\), and use a convolution layer \(v\) with the kernel size of 1 to generate the integrated feature map \(F_{fu}\):

\[
F_{fu} = v ([F'_{ct}, F'_{cte}]).
\]

**Bilateral Propagation Feature Aggregation (BPFA).** This module is designed to co-learn contextual attention, channel-wise information, and feature space so as to enhance the overall consistency. Fig. 4 illustrates the proposed BPFA module in detail. Specifically, to capture channel-wise information, we use the Selective Kernel Convolution module of SKNet [14] to generate the feature map \(F'_{fu}\). Inspired by [34], we introduce the Contextual Attention (CA) module to capture the correlation between feature patches. For a given input feature \(F'_{fu}\), we divide it into non-overlapping patches with size 3×3 and calculate the cosine similarity between these patches as:

\[
S_{i,j}^{\text{contextual}} = \frac{p_i}{\|p_i\|_2} \cdot \frac{p_j}{\|p_j\|_2},
\]

where \(p_i\) and \(p_j\) are the \(i\)-th and \(j\)-th patches of the input feature \(F'_{fu}\), respectively. We utilize the Softmax function to get the attention score of each pair of patches:

\[
\bar{S}_{i,j}^{\text{contextual}} = \exp \left( \frac{S_{i,j}^{\text{contextual}}}{\sum_{j=1}^{N} \exp (S_{i,j}^{\text{contextual}})} \right),
\]

where \(N\) is the total number of patches of the input feature \(F'_{fu}\). Next, the attention score is used to compute the feature patches \(\tilde{p}_i, i = 1, ..., N\) by

\[
\bar{p}_i = \sum_{j=1}^{N} p_j \cdot \bar{S}_{i,j}^{\text{contextual}}.
\]

Similarly, the texture-guided feature \(F'_{ct}\) can be calculated as follows:

\[
F'_{ct} = \alpha (G_{cte} \odot F) \oplus F_{cte},
\]

where \(\alpha\) and \(\beta\) are two learnable parameters, \(\odot\) and \(\oplus\) denote element-wise multiplication and addition respectively.

Fig. 3: Description of Soft-gating Dual Feature Fusion, which can effectively fuse the structure and texture features. SDFF module. Specifically, in order to construct the structure-guided texture features, we utilize the soft gating \(G_{cte}\) to control a degree of refining the texture information. The soft gating can be defined as:

\[
G_{cte} = \sigma (SE(h ([F_{cte}, F_{cte}]))),
\]

where \(h(\cdot)\) is a convolution layer with the kernel size of 3, \(SE(\cdot)\) is a squeeze and excitation operation [8] to capture important channel information, and \(\sigma(\cdot)\) is the Sigmoid activation function. With soft gating \(G_{cte}\), we can dynamically merge \(F_{cte}\) into \(F_{cte}'\) by

\[
F_{cte}' = \alpha (G_{cte} \odot F_{cte}) \oplus F_{cte},
\]

and calculate the cosine similarity between these patches as:

\[
S_{i,j}^{\text{contextual}} = \frac{p_i}{\|p_i\|_2} \cdot \frac{p_j}{\|p_j\|_2},
\]

where \(p_i\) and \(p_j\) are the \(i\)-th and \(j\)-th patches of the input feature \(F_{cte}'\), respectively. We use the Selective Kernel Convolution module of SKNet to enhance the overall consistency.
The reconstructed feature map $\tilde{F}_{fu}$ can be obtained by directly reorganize all the feature patches.

In the range and spatial domains, we introduce the Bilateral Propagation Activation (BPA) module to generate the feature maps based on the range and spatial distances. The feature map of the range domain can be calculated as:

$$y_i^r = \frac{1}{C(x)} \sum_{j \in v} f(x_i, x_j) x_j,$$

(11)

$$f(x_i, x_j) = (x_i)T(x_j),$$

(12)

where we use the unfold function of PyTorch to reshape $\tilde{F}_{fu}$ to two kinds of vectors, which are of $HW \times 3 \times 3 \times C$ and $HW \times 1 \times 1 \times C$ dimensions. Here, $x_i$ is the $i$-th output channel of the input feature $\tilde{F}_{fu}, x_j$ is a neighboring channel at position $j$ around position $i$. The pairwise function $f(\cdot)$ is dot-product similarity. The operations from the unfold function to $P_1$ represent Eq. (12). $v$ is a neighboring region of position $i$ and its size is set to $3 \times 3$. For a given $i$, $\frac{1}{C(x)} \sum_{j \in v} f(x_i, x_j)$ can be seen as the Softmax computation along dimension $j$. $C(x)$ is the number of channels in $\tilde{F}_{fu}$. Similarly, Eq. (11) represent the operations from on $\tilde{F}_{fu}$ until the position $P_2$. The feature map of the spatial domain can be calculated as:

$$y_i^s = \frac{1}{C(x)} \sum_{s \in s} g_{os}(\|y_i - \alpha\|) x_j,$$

(13)

where we use the unfold function of PyTorch to reshape $\tilde{F}_{fu}$ to a vector, which is of $HW \times C \times H \times W$ dimension. $g_{os}$ is a Gaussian function to adjust the spatial contributions from neighboring region. We explore $j$ in a neighboring region $s$ for global propagation. In the experiment, $s$ is set to the same size as the input feature. Eq. 13 represent the operations from the unfold function to $P_3$. Therefore, we can obtain the feature maps $y_i^r$ and $y_i^s$ by the spatial and range similarity measurement methods, respectively. We can see that the bilateral propagation considers both local consistency via $y_i^r$ and global consistency via $y_i^s$. Each feature channel can be computed by

$$y_i = q|(y_i^r, y_i^s)|,$$

(14)

where $q$ denotes a convolution layer with the kernel size of 1. Next, the reconstructed feature map $\tilde{F}_{ar}$ is obtained by aggregating all feature channels $y_i$ ($i = 1, ..., C(x)$). Finally, $F'_{fu}$ and $\tilde{F}_{ar}$ are concatenated and mapped to $F_{sc}$ by

$$F_{sc} = z \left( [F'_{fu}, \tilde{F}_{ar}] \right),$$

(15)

where $z$ is a convolution layer with the kernel size of 1.

B. Discriminator

The discriminator consists of the global critic network and local critic network, which can ensure the local and global content consistency. Each critic network includes six convolution layers with the kernel size of 4 and stride of 2, and use the Leaky ReLu with the slope of 0.2 for all but the last layer. Furthermore, the spectral normalization is adopted in our network to achieve stable training.

C. Loss Function

Our network is trained with a series of loss functions, including pixel reconstruction loss, perceptual loss, style loss, and adversarial loss so that the finally generated image looks more visually realistic.

Pixel Reconstruction Loss. We adopt the $l_1$ distance as the reconstruction loss from two aspects. The first aspect is to supervise the structure and texture branches. The corresponding loss functions are formulated in Eqs. (1) and (2), respectively. The second aspect is to measure the similarity between the final output result $I_{out}$ and the ground-truth image $I_{gt}$ by

$$L_{rec} = ||I_{out} - I_{gt}||_1.$$

(16)
Visual comparison on the Places2 (the first row) and Paris StreetView (the second row) datasets.

Fig. 5: Visual comparison on the CelebA dataset.

Perceptual Loss. We utilize the perceptual loss $L_{\text{perc}}$ to capture the high-level semantics [10] by computing the $l_1$ distance between the feature spaces of $I_{\text{out}}$ and $I_{\text{gt}}$ through ImageNet-pretrained VGG-16 backbone, which can be written as:

$$
L_{\text{perc}} = \mathbb{E} \left[ \sum_i ||\phi_i(I_{\text{out}}) - \phi_i(I_{\text{gt}})||_1 \right],
$$

where $\phi_i(\cdot), i = 1, \ldots, 5$ denote the five activation maps from VGG-16, which are ReLu1, ReLu2, ReLu3, ReLu4, and ReLu5.

Style Loss. We introduce the style loss $L_{\text{style}}$ to mitigate style differences, which is defined as:

$$
L_{\text{style}} = \mathbb{E} \left[ \sum_i ||\varphi_i(I_{\text{out}}) - \varphi_i(I_{\text{gt}})||_1 \right],
$$

where $\varphi_i(\cdot) = \phi_i^T \phi_i$ denotes the Gram matrix constructed from the above-mentioned five activation maps.

Adversarial Loss. The relativistic average least squares adversarial loss $L_{\text{adv}}$ is to ensure the local and global content consistency. For the generator, the adversarial loss is defined as:

$$
L_{\text{adv}} = -\mathbb{E}_{x_r} \left[ \log (1 - D_{ta}(x_r, x_f)) \right] - \mathbb{E}_{x_f} \left[ \log (D_{ta}(x_f, x_r)) \right],
$$

where $D_{ta}(x_r, x_f) = \text{Sigmoid} \left( D_{gl}(x_r) - \mathbb{E}_{x_f} [D_{gl}(x_f)] \right)$, $D_{gl}$ denotes the local or global discriminator without the last Sigmoid function, and $(x_r, x_f)$ denotes a pair of the ground-truth and output images.

Total Loss. The total loss of the proposed method can be obtained by:

$$
L_{\text{total}} = \lambda_r L_{\text{rec}} + \lambda_p L_{\text{precomp}} + \lambda_s L_{\text{style}} + \lambda_{\text{adv}} L_{\text{adv}} + \lambda_{\text{pte}} L_{\text{pte}} + \lambda_{\text{st}} L_{\text{st}},
$$

where $\lambda_r, \lambda_p, \lambda_s, \lambda_{\text{adv}}, \lambda_{\text{pte}}$ and $\lambda_{\text{st}}$ are the tradeoff parameters, and we empirically set $\lambda_r = 1$, $\lambda_p = 0.2$, $\lambda_s = 250$, $\lambda_{\text{adv}} = 0.2$, $\lambda_{\text{pte}} = 1$ and $\lambda_{\text{st}} = 1$.

IV. EXPERIMENTAL RESULTS

In our experiment, three public datasets are used for the verification, including Places2 [38], Paris StreetView [4], and CelebA [21]. We follow the training, testing, and validation splits of these three datasets themselves. The irregular masks are taken from PConv [17] and classified based on the ratios of the hole to the entire image with an increment of 10%. Our network is built on the PyTorch framework, trained on a single NVIDIA 2080 Ti GPU (11GB), and optimized by the Adam optimizer with a learning rate of $2 \times 10^{-4}$. The CelebA, Paris StreetView, and Places2 models require around 10, 55, and 100 epochs, respectively. All the masks and images are resized to $256 \times 256$.

A. Performance comparison with state-of-the-art

We compare the proposed method with eight state-of-the-art methods, including GC [35], EC [22], MED [18], RFR [13], DSI [24], CE [23], CA [34] and SH [32]. For the evaluation metrics, we use four common metrics: PSNR, SSIM, MAE and FID (Fréchet Inception Distance) [7]. For fair comparison, we use the same experimental setup for all the compared methods. The experiments are conducted on two types of damaged images containing center hole and irregular hole. In the following, we present the experimental results and analysis in detail.

For images with irregular holes, we take the Places2 [38] and Paris StreetView [4] datasets for evaluation, and select several representative methods GC [35], EC [22], MED [18], RFR [13] and DSI [24] for performance comparison. Meanwhile, we use the same validation images as the MED method. Experimental results are shown in Tables I and II. We can see from the two tables that our method achieves significant improvement over the compared methods in terms of the PSNR, SSIM, and MAE metrics. This is due to the fact that our
**TABLE I:** Performance comparison on the Places2 dataset.

| Mask Ratio | PSNR↑ | SSIM↑ | MAE↓ | FID↓ |
|------------|-------|-------|------|------|
| 10-20%     | 28.31 | 0.909 | 0.828 | 0.0391 |
| 20-30%     | 30.52 | 0.919 | 0.836 | 0.0415 |
| 30-40%     | 32.86 | 0.930 | 0.854 | 0.0435 |
| 40-50%     | 35.19 | 0.938 | 0.869 | 0.0455 |

The table shows the performance comparison on the Places2 dataset with different mask ratios. The PSNR, SSIM, MAE, and FID metrics are used to evaluate the performance. The values indicate a better performance with higher PSNR and SSIM and lower MAE and FID. The PSNR and SSIM values increase, while the MAE and FID values decrease with increasing mask ratios, demonstrating the effectiveness of introducing the structure prior in the encoding stage.

**TABLE II:** Performance comparison on the Paris StreetView dataset.

| Mask Ratio | PSNR↑ | SSIM↑ | MAE↓ | FID↓ |
|------------|-------|-------|------|------|
| 10-20%     | 28.31 | 0.909 | 0.828 | 0.0391 |
| 20-30%     | 30.52 | 0.919 | 0.836 | 0.0415 |
| 30-40%     | 32.86 | 0.930 | 0.854 | 0.0435 |
| 40-50%     | 35.19 | 0.938 | 0.869 | 0.0455 |

The table shows the performance comparison on the Paris StreetView dataset with different mask ratios. The PSNR, SSIM, MAE, and FID metrics are used to evaluate the performance. The values indicate a better performance with higher PSNR and SSIM and lower MAE and FID. The PSNR and SSIM values increase, while the MAE and FID values decrease with increasing mask ratios, demonstrating the effectiveness of introducing the structure prior in the encoding stage.

**TABLE III:** Performance comparison on the CelebA dataset.

| Mask Ratio | PSNR↑ | SSIM↑ | MAE↓ | FID↓ |
|------------|-------|-------|------|------|
| 10-20%     | 28.31 | 0.909 | 0.828 | 0.0391 |
| 20-30%     | 30.52 | 0.919 | 0.836 | 0.0415 |
| 30-40%     | 32.86 | 0.930 | 0.854 | 0.0435 |
| 40-50%     | 35.19 | 0.938 | 0.869 | 0.0455 |

The table shows the performance comparison on the CelebA dataset with different mask ratios. The PSNR, SSIM, MAE, and FID metrics are used to evaluate the performance. The values indicate a better performance with higher PSNR and SSIM and lower MAE and FID. The PSNR and SSIM values increase, while the MAE and FID values decrease with increasing mask ratios, demonstrating the effectiveness of introducing the structure prior in the encoding stage.

**TABLE IV:** Ablation study of different modules on Paris StreetView. Here, random masks with mask ratio 30%-40% are used.

| Metrics | PSNR↑ | SSIM↑ | MAE↓ | FID↓ |
|---------|-------|-------|------|------|
| Ours    | 36.38 | 0.922 | 0.0232 | 1.40 |

The table shows the performance comparison on the Paris StreetView dataset with different modules. The PSNR, SSIM, MAE, and FID metrics are used to evaluate the performance. The values indicate a better performance with higher PSNR and SSIM and lower MAE and FID. The PSNR and SSIM values increase, while the MAE and FID values decrease with increasing mask ratios, demonstrating the effectiveness of introducing the structure prior in the encoding stage.

**TABLE V:** Performance comparison on the CelebA dataset.

| Mask Ratio | PSNR↑ | SSIM↑ | MAE↓ | FID↓ |
|------------|-------|-------|------|------|
| 10-20%     | 28.31 | 0.909 | 0.828 | 0.0391 |
| 20-30%     | 30.52 | 0.919 | 0.836 | 0.0415 |
| 30-40%     | 32.86 | 0.930 | 0.854 | 0.0435 |
| 40-50%     | 35.19 | 0.938 | 0.869 | 0.0455 |

The table shows the performance comparison on the CelebA dataset with different mask ratios. The PSNR, SSIM, MAE, and FID metrics are used to evaluate the performance. The values indicate a better performance with higher PSNR and SSIM and lower MAE and FID. The PSNR and SSIM values increase, while the MAE and FID values decrease with increasing mask ratios, demonstrating the effectiveness of introducing the structure prior in the encoding stage.

**TABLE VI:** Performance comparison on the Places2 dataset.

| Mask Ratio | PSNR↑ | SSIM↑ | MAE↓ | FID↓ |
|------------|-------|-------|------|------|
| 10-20%     | 28.31 | 0.909 | 0.828 | 0.0391 |
| 20-30%     | 30.52 | 0.919 | 0.836 | 0.0415 |
| 30-40%     | 32.86 | 0.930 | 0.854 | 0.0435 |
| 40-50%     | 35.19 | 0.938 | 0.869 | 0.0455 |

The table shows the performance comparison on the Places2 dataset with different mask ratios. The PSNR, SSIM, MAE, and FID metrics are used to evaluate the performance. The values indicate a better performance with higher PSNR and SSIM and lower MAE and FID. The PSNR and SSIM values increase, while the MAE and FID values decrease with increasing mask ratios, demonstrating the effectiveness of introducing the structure prior in the encoding stage.

**TABLE VII:** Performance comparison on the Paris StreetView dataset.

| Mask Ratio | PSNR↑ | SSIM↑ | MAE↓ | FID↓ |
|------------|-------|-------|------|------|
| 10-20%     | 28.31 | 0.909 | 0.828 | 0.0391 |
| 20-30%     | 30.52 | 0.919 | 0.836 | 0.0415 |
| 30-40%     | 32.86 | 0.930 | 0.854 | 0.0435 |
| 40-50%     | 35.19 | 0.938 | 0.869 | 0.0455 |

The table shows the performance comparison on the Paris StreetView dataset with different mask ratios. The PSNR, SSIM, MAE, and FID metrics are used to evaluate the performance. The values indicate a better performance with higher PSNR and SSIM and lower MAE and FID. The PSNR and SSIM values increase, while the MAE and FID values decrease with increasing mask ratios, demonstrating the effectiveness of introducing the structure prior in the encoding stage.

In this paper, we have presented a deep multi-feature co-learning network for image inpainting, which can yield the detailed textures and reasonable structures. Our network designs two new fusion modules: Soft-gating Dual Feature Fusion (SDFF) and Bilateral Propagation Feature Aggregation (BPFA). SDFF can control the fusion ratio through a soft gating technique to refine the structure and texture features, making the structures and textures more consistent. BPFA can co-learn contextual attention, channel-wise information and feature space. By the co-learning strategy, the inpainted images preserve the from local to overall consistency. In the future, we will explore the performance of introducing the structure prior in the encoding stage.

V. CONCLUSION

Speaking, the partial convolution based padding (PC) module [17] improves the performance by making the model pay more attention to visible pixels (undamaged regions). Moreover, our designed BPFA module contributes to the whole network the best. This is because BPFA enhances the connection from local feature to overall consistency, thus reducing visible artifacts and unpleasant contents. In addition, we can find that the SDFF module has the second contribution. As expected, SDFF captures the correlation between the structure and texture features. Interestingly, the CA module can learn contextual feature representations and the SKNet module can effectively equalize the structure and texture features generated by the multi-scale filling stage, thereby benefiting the proposed network.

301
REFERENCES

[1] C. Barnes, E. Shechtman, A. Finkelstein, and D. B. Goldman. Patch-match: A randomized correspondence algorithm for structural image editing. ACM Transactions on Graphics (ToG), 28(3):1–11, 2009.

[2] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester. Image inpainting. In Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH), pages 417–424, 2000.

[3] S. Dugar, C. Barnes, D. B. Goldman, and P. Sen. Image melding: Combining inconsistent images using patch-based synthesis. ACM Transactions on Graphics (ToG), 31(4):1–10, 2012.

[4] C. Doersch, S. Singh, A. Gupta, J. Sivic, and A. Efros. What makes part look like parts? ACM Transactions on Graphics (ToG), 31(4):1–9, 2012.

[5] A. A. Efros and W. T. Freeman. Image quilting for texture synthesis and transfer. In Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH), pages 341–346, 2001.

[6] J. Hays and A. A. Efros. Scene completion using millions of photographs. ACM Transactions on Graphics (ToG), 26(3):1–7, 2007.

[7] H. 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. In Proceedings of the Advances in Neural Information Processing Systems (NIPS), pages 6629–6640, 2017.

[8] J. Hu, L. Shen, and G. Sun. Squeeze-and-excitation networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 7132–7141, 2018.

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

[10] 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 (ECCV), pages 694–711, 2016.

[11] A. Lahiri, A. K. Jain, S. Agrawal, P. Mitra, and P. K. Biswas. Prior guided gan based semantic inpainting. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 13696–13705, 2020.

[12] J. Li, Z. Li, J. Cao, X. Song, and R. He. Face inpaintnet: High fidelity face adaptation to heterogeneous domains. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 5089–5098, 2021.

[13] L. Li, N. Wang, L. Zhang, B. Du, and D. Tao. Recurrent feature reasoning for image inpainting. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 7760–7768, 2020.

[14] X. Li, W. Wang, X. Hu, and J. Yang. Selective kernel networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 510–519, 2019.

[15] Y. Li, S. Liu, J. Yang, and M.-H. Yang. Generative face completion. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 3911–3919, 2017.

[16] L. Liao, J. Xiao, Z. Wang, C.-W. Lin, and S. Satoh. Image inpainting guided by coherence priors of semantics and textures. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 6539–6548, 2021.

[17] 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 (ECCV), pages 85–100, 2018.

[18] 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 (ECCV), pages 725–741, 2020.

[19] 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 (ICCV), pages 4170–4179, 2019.

[20] H. Liu, Z. Wan, W. Huang, Y. Song, X. Han, and J. Liao. Pd-gan: Prosody-driven image inpainting. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 9371–9381, 2021.

[21] 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 (ICCV), pages 3730–3738, 2015.

[22] K. Nazeri, E. Ng, T. Joseph, F. Z. Qureshi, and M. Ebrahimi. Edgeconnect: Generative image inpainting with adversarial edge learning. In Proceedings of the IEEE International Conference on Computer Vision Workshop (ICCVW), pages 3265–3274, 2019.

[23] 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 (CVPR), pages 2536–2544, 2016.

[24] J. Peng, D. Liu, S. Xu, and H. Li. Generating diverse structure for image inpainting with hierarchical vq-vae. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 10775–10784, 2021.

[25] Y. Ren, X. Yu, R. Zhang, T. H. Li, S. Liu, and G. Li. Structureflow: Image inpainting via structure-aware steerable flow. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), pages 181–190, 2019.

[26] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 839–846, 1998.

[27] T. Wang, H. Ouyang, and Q. Chen. Image inpainting with external-internal learning and monochromatic bottleneck. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 6539–6548, 2019.

[28] 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 (CVPR), pages 7794–7803, 2018.

[29] X. Wu, S. Liu, C. Li, M.-M. Cheng, W. Zuo, X. Liu, S. Wen, and E. Ding. Image inpainting with learnable bidirectional attention maps. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), pages 8858–8867, 2019.

[30] W. Xiong, J. Yu, Z. Lin, J. Yang, X. Lu, C. Barnes, and J. Luo. Foreground-aware image inpainting. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages S840–S848, 2019.

[31] L. Yuan, Q. Yan, Y. Xia, and J. Jia. Structure extraction from texture via relative total variation. ACM Transactions on Graphics (ToG), 31(6):1–10, 2012.

[32] Z. Yan, X. Li, M. Li, W. Zuo, and S. Shan. Shift-net: Image inpainting via deep feature rearrangement. In Proceedings of the European Conference on Computer Vision (ECCV), pages 1–17, 2018.

[33] J. Yang, Z. Qi, and Y. Shi. Learning to incorporate structure knowledge for image inpainting. In Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), pages 12605–12612, 2020.

[34] 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 (CVPR), pages 5505–5514, 2018.

[35] 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 (ICCV), pages 4471–4480, 2019.

[36] X. Z, Z. Guo, X. Jin, S. Wu, Z. Chen, W. Li, Z. Zhang, and S. Liu. Region normalization for image inpainting. In Proceedings of the AAAI Conference on Artificial Intelligence (AAAI), pages 12733–12740, 2020.

[37] Y. Zeng, Z. Lin, J. Yang, Z. Shechtman, and H. Lu. High-resolution image inpainting with iterative confidence feedback and guided upsampling. In Proceedings of the European Conference on Computer Vision (ECCV), pages 1–17, 2020.

[38] 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 (TPAMI), 40(6):1452–1464, 2017.

[39] Y. Zou, C. Barnes, E. Shechtman, and S. Amirghodsi. Transfill: Reference-guided image inpainting by merging multiple color and spatial transformations. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 2266–2276, 2021.