Pacific Graphics 2020
E. Eisemann, A. Jacobson, and F.-L Zhang
(Guest Editors)

*Volume 39 (2020), Number 7*

# Pixel-wise Dense Detector for Image Inpainting

# (Supplementary Material)

Ruisong Zhang[1,2], Weize Quan[1,2], Baoyuan Wu[3,4], Zhifeng Li[5] and Dong-Ming Yan[1,2]

[1]National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China
[2]School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China
[3]School of Data Science, the Chinese University of Hong Kong, Shenzhen, China
[4]Secure Computing Lab of Big Data, Shenzhen Research Institute of Big Data, China
[5]Tencent AI Lab, Shenzhen, China

In this supplementary material, we provides detail implementation of network architectures including the generator and the detector in Section 1. Section 2 presents more quantitative comparisons, and we show additional qualitative results in Section 3.

## 1. Details of Network Architecture

Table 1 shows the encoder-decoder architecture of the whole generator with eight residual blocks as bottleneck, and the architecture of residual block lists in Table 2. The input of the generator is the concatenation of the corrupted image and the mask with four channels, and the range of the input is $[0,1]$ after normalization. The output of the generator is the completion prediction also with range $[0,1]$. Obviously, the generator constructs a mapping from $\mathbb{R}^{256\times256\times4} \in [0,1]$ to $\mathbb{R}^{256\times256\times3} \in [0,1]$. The architecture of the detector reports in Table 3. The input of the detector is the completion prediction, and the detector outputs the evaluation result with two-layer probability map.

## 2. More Quantitative Comparisons

Table 4 reports quantitative comparison results of PConv [LRS*18], PEN [ZFCG19], GConv [YLY*19] and our method on Paris StreetView [DSG*12] dataset, which is a complement of Table 1 in the paper to fully measure above four methods. Our method achieves best results among all methods except "FID" in the range of (0.01-0.1] and (0.5-0.6].

## 3. More Qualitative Results

Fig. 1, Fig. 2 and Fig. 3 show more qualitative comparisons on Celeba-HQ [LLWT15, KALL17], Places2 [ZLK*17] and Paris StreetView [DSG*12] dataset, respectively. Moreover, additional results by our proposed method are shown in Fig. 4 and Fig. 5.



## References

[DSG*12] DOERSCH C., SINGH S., GUPTA A., SIVIC J., EFROS A. A.: What makes paris look like paris? *ACM Transactions on Graphics 31*, 4 (2012), 101:1–101:9. 1

[KALL17] KARRAS T., AILA T., LAINE S., LEHTINEN J.: Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196* (2017). 1

[LLWT15] LIU Z., LUO P., WANG X., TANG X.: Deep learning face attributes in the wild. In *Proceedings of the IEEE international conference on computer vision* (2015), pp. 3730–3738. 1

[LRS*18] LIU G., REDA F. A., SHIH K. J., WANG T.-C., TAO A., CATANZARO B.: Image inpainting for irregular holes using partial convolutions. In *European Conference on Computer Vision* (2018), pp. 85–100. 1

[YLY*19] YU J., LIN Z., YANG J., SHEN X., LU X., HUANG T. S.: Free-form image inpainting with gated convolution. In *Proceedings of the IEEE international conference on computer vision* (2019), pp. 4471–4480. 1

[ZFCG19] ZENG Y., FU J., CHAO H., GUO B.: 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. 1

[ZLK*17] ZHOU B., LAPEDRIZA A., KHOSLA A., OLIVA A., TORRALBA A.: Places: A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence 40*, 6 (2017), 1452–1464. 1

**Table 1:** *The architecture of the generator. The column of "Type" distinguishes convolution (Conv) and deconvolution (DeConv). The type of padding is classified into Reflect and Zero. The column of "Chanel" means the number of filters in this layer or the number of output feature maps. The first three layers are in encoder stage, whereas the last three layers are in decoder stage.*

| Type | Channel | Kernel Size | Stride | Padding | Padding Type | Instance Norm | Nonlinearity |
|---|---|---|---|---|---|---|---|
| Conv | 64 | $7 \times 7$ | 1 | 3 | Reflect | Y | ReLU(·) |
| Conv | 128 | $4 \times 4$ | 2 | 1 | Zero | Y | ReLU(·) |
| Conv | 256 | $4 \times 4$ | 2 | 1 | Zero | Y | ReLU(·) |
| Bottleneck: 8 $\times$ Residual Blocks | | | | | | | |
| DeConv | 128 | $4 \times 4$ | 2 | 1 | Zero | Y | ReLu(·) |
| DeConv | 64 | $4 \times 4$ | 2 | 1 | Zero | Y | ReLu(·) |
| Conv | 3 | $7 \times 7$ | 1 | 3 | Reflect | N | [Tanh(·)+1]/2 |

**Table 2:** *The architecture of the residual block with two convolutional layers. The type of all padding is Reflect.*

| Type | Channel | Kernel Size | Stride | Dilation Rate | Padding | Instance Norm | ReLU |
|---|---|---|---|---|---|---|---|
| Conv | 256 | $3 \times 3$ | 1 | 2 | 2 | Y | Y |
| Conv | 256 | $3 \times 3$ | 1 | 1 | 1 | Y | N |

**Table 3:** *The architecture of the detector. All layers do not include normalization operations. The last two layers are in decoder stage to upsample the evaluation with the same size as the input.*

| Type | Channel | Kernel Size | Stride | Padding | Padding Type | Nonlinearity |
|---|---|---|---|---|---|---|
| Conv | 32 | $4 \times 4$ | 1 | 2 | Zero | LeakyReLU(0.2) |
| Conv | 64 | $4 \times 4$ | 1 | 2 | Zero | LeakyReLU(0.2) |
| Conv | 128 | $4 \times 4$ | 2 | 1 | Zero | LeakyReLU(0.2) |
| Conv | 256 | $4 \times 4$ | 2 | 1 | Zero | LeakyReLU(0.2) |
| Conv | 256 | $4 \times 4$ | 1 | 2 | Zero | LeakyReLU(0.2) |
| DeConv | 128 | $4 \times 4$ | 2 | 1 | Zero | / |
| DeConv | 2 | $4 \times 4$ | 2 | 1 | Zero | SoftMax |

**Table 4:** *Comparison with various methods on Paris StreetView dataset. † Lower is better. ¶ Higher is better.*

| | Mask | (0.01-0.1] | (0.1-0.2] | (0.2-0.3] | (0.3-0.4] | (0.4-0.5] | (0.5-0.6] |
|---|---|---|---|---|---|---|---|
| $\ell_1(\%)$† | PConv | 1.17 | 2.87 | 4.87 | 6.96 | 9.38 | 13.34 |
| | PEN | 0.97 | 2.58 | 4.65 | 6.84 | 9.35 | 13.00 |
| | GConv | 0.93 | 2.55 | 4.67 | 6.99 | 9.58 | 14.19 |
| | Ours | **0.85** | **1.96** | **3.41** | **5.07** | **7.01** | **10.71** |
| PSNR¶ | PConv | 32.76 | 28.02 | 25.47 | 23.80 | 22.36 | 20.37 |
| | PEN | 34.25 | 28.97 | 26.03 | 24.12 | 22.56 | 20.72 |
| | GConv | 34.72 | 28.95 | 25.73 | 23.62 | 21.95 | 19.59 |
| | Ours | **34.88** | **31.05** | **28.23** | **26.17** | **24.48** | **21.90** |
| SSIM¶ | PConv | 0.968 | 0.925 | 0.874 | 0.820 | 0.752 | 0.629 |
| | PEN | 0.979 | 0.939 | 0.884 | 0.821 | 0.745 | 0.625 |
| | GConv | 0.980 | 0.940 | 0.885 | 0.825 | 0.757 | 0.629 |
| | Ours | **0.983** | **0.960** | **0.926** | **0.882** | **0.827** | **0.706** |
| FID† | PConv | 15.34 | 30.42 | 46.58 | 62.90 | 82.00 | 102.75 |
| | PEN | 9.63 | 25.71 | 46.52 | 67.88 | 91.65 | 117.94 |
| | GConv | **7.84** | 20.27 | 34.50 | 46.92 | 59.73 | **75.11** |
| | Ours | 9.13 | **17.27** | **29.75** | **43.54** | **58.86** | 83.09 |

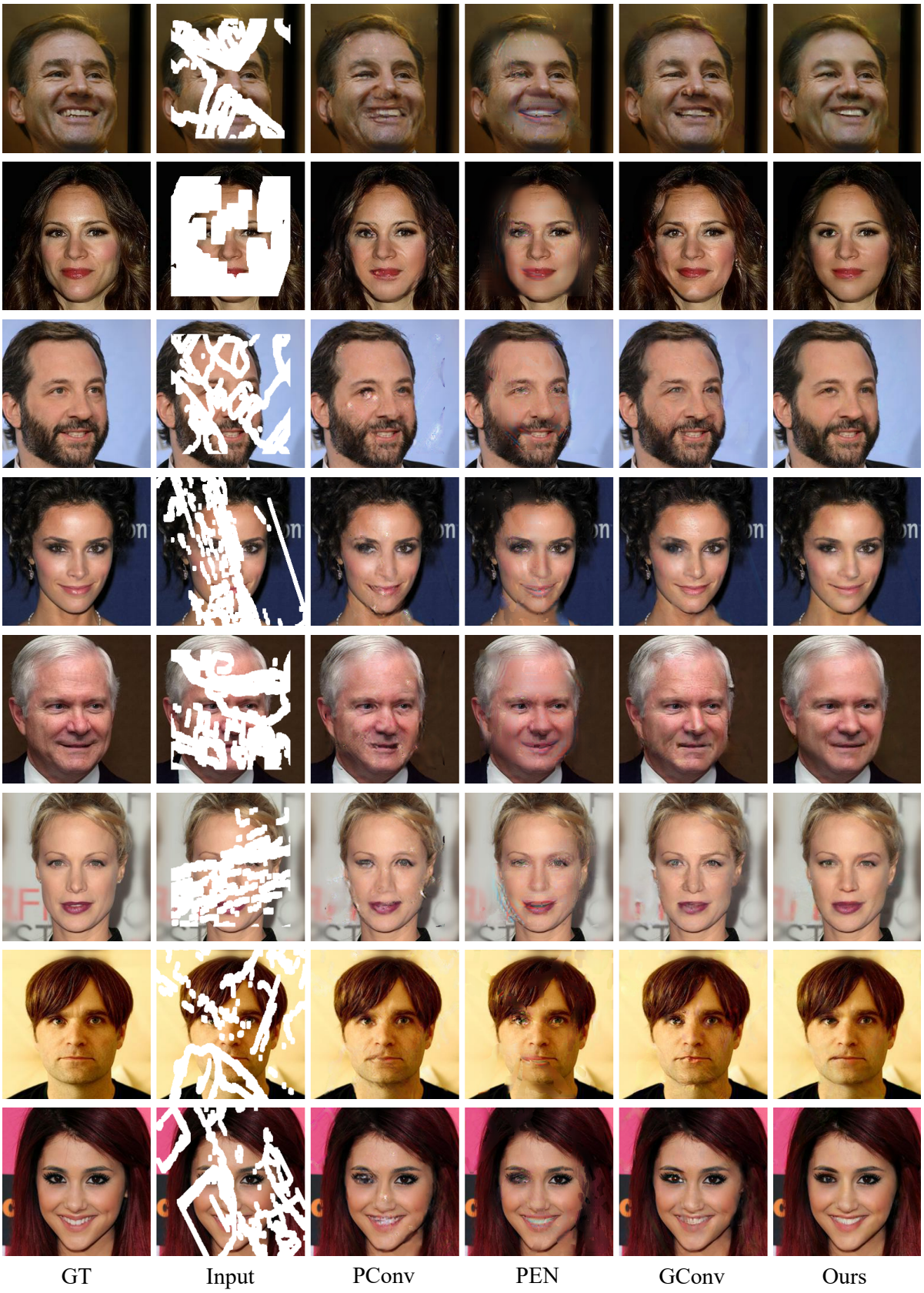


**Figure 1:** *Qualitative comparison over Celeba-HQ dataset. The notations of each method are the same as the Fig. 4 in the paper.*

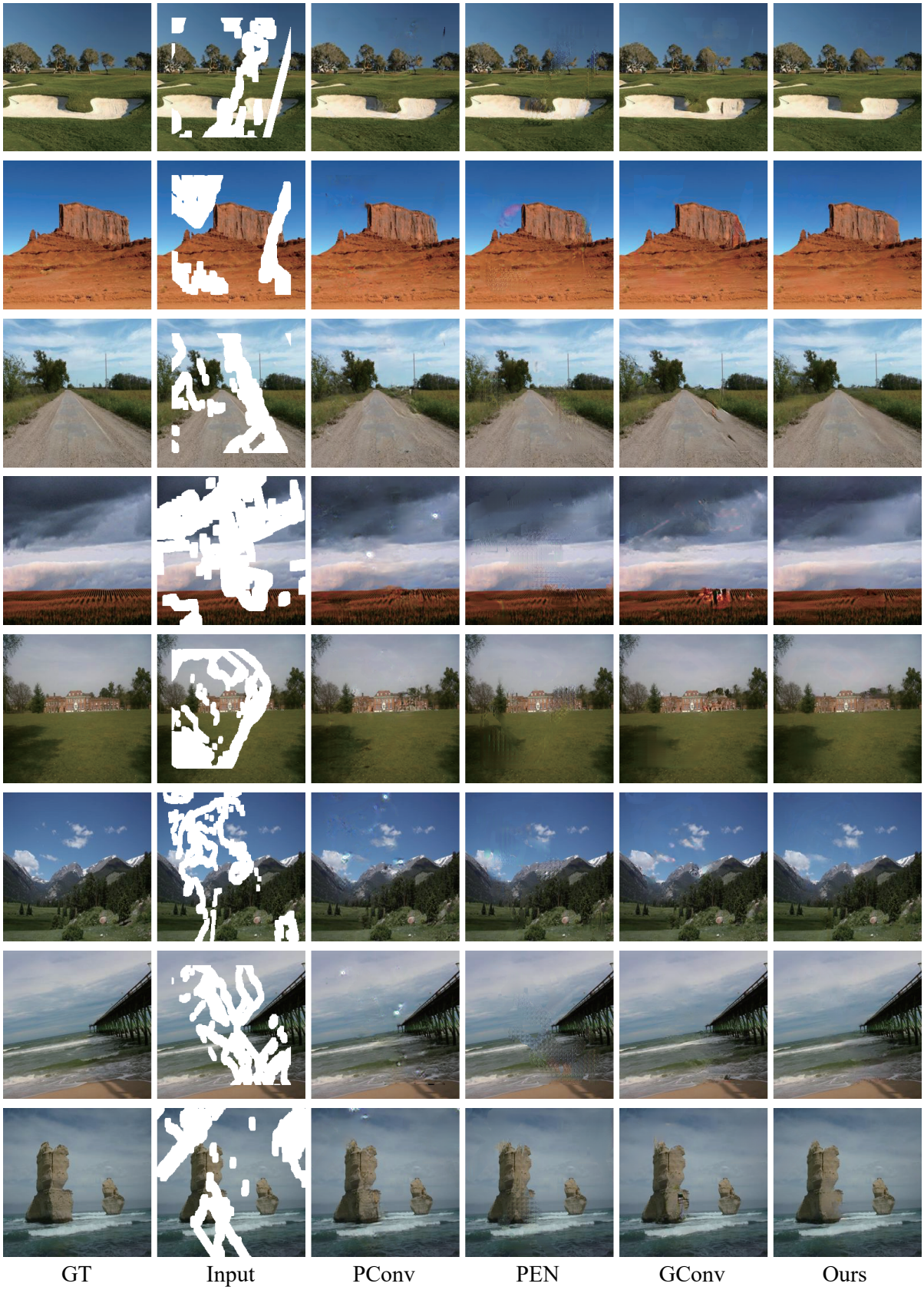


**Figure 2:** *Qualitative comparison over Places2 dataset. The notations of each method are the same as the Fig. 4 in the paper.*


**Figure 3:** *Qualitative comparison over Paris StreetView dataset. The notations of each method are the same as the Fig. 4 in the paper.*


**Figure 4:** *Additional results of our proposed method.*


**Figure 5:** *Additional results of our proposed method.*