Image Inpainting Using GAN, Partial Convolution and Region Normalization

Chengyu Shi
University of Michigan
Ann Arbor, MI
chengyus@umich.edu

Yilin Li
University of Michigan
Ann Arbor, MI
yilinliz@umich.edu

Yixuan Jia
University of Michigan
Ann Arbor, MI
jiayx@umich.edu

Abstract

Previous work using Partial Convolution can achieve decent results for image inpainting with irregular holes, and the application of GAN in this area seems to be promising. We proposed a model based on a conditional GAN. The generator G is a UNet-like architecture, replacing all convolutional layers with partial convolutional layers. We implement Region Normalization in the Generator, which normalizes corrupted and uncorrupted regions separately. The discriminator D is like the one used in pix2pix model, which takes both the corrupted image and target image as inputs. Our model outperforms the baseline model and we show qualitative and quantitative comparisons to validate our approach.

1. Introduction

1.1. Specific Motivation and Significance

Sometimes, the images we obtained may not be "perfect", which means they were damaged or had missing parts. For example, shared images in the social networks can contain many objects, including signature, rectangles, and so on. The addition of these objects may change the semantic of the images. **Image inpainting** is the task of reconstructing missing regions in an image, which is, to be more specific, the process of completing or recovering the missing region in the image or removing some objects added to it. By using recently developed algorithms, one could restore coherently both texture and structure components of the image. Our goal is to develop a CGAN-based techniques for this task, and we expect it to outperform other networks on certain evaluation matrices.

1.2. Contributions

In this project, we designed a neural network basically based on three computer vision techniques: Partial Convolution, Region Normalization, and Generative Adversarial Network, for this image inpainting task. We trained and tested our network on **Places** dataset, which are built for human visual cognition and visual understanding purposes. Finally we achieved improvement with our network on widely used numerical evaluation matrices, i.e. L_1 loss, $PSNR$ and $SSIM$ compared with baseline.

2. Related work

2.1. Previous Work, Strength and Weakness

Popular approaches related to image inpainting can be classified into three basic categories: Patch-based approaches, CNN-based approaches, GAN-based approaches.

Patch-based approaches are based on methods to fill in the missing region patch-by-patch via searching for well-matching replacement patches in the unmasked area of the input image and copying them to corresponding masked locations. Ruzic and Pizurica[13] proposed a patch-based method consisting of searching the well-matched patch in the texture component using Markov Random Field (MRF). Jin and Ye[4] proposed a patch-based method based on annihilation property filter and low rank structured matrix. Kawai et al.[5] proposed a patch-based method based on selecting the target object and put a limit on the searching near the target. Mo and Zhou[10] proposed a method based on dictionary learning using sparse representation. All these patch-base methods could provide promising results on simple image inpainting tasks. However, when the image is complex, such as consisting of many texture and objects, searching for similar patch could be more than difficult.

CNN-based approaches are based on deep convolutional networks. CNNs are commonly used when using large-scale training data. Shift-Net[16] is designed based on U-Net, and it is one of the approaches that recover the missing parts with good accuracy with structure and fine-detailed texture. Using the similar encoder-decoder techniques, Zeng et al.[18] proposed a pyramidal-context architecture named PEN-NET for high quality image inpainting. Besides, Nakamura et al.[11] proposed a text erasing method using CNN.

GANs, first proposed by Ian Goodfellow et al.[2], con-

tains two feed-forward networks, a generator and a discriminator. They were trained alternatively and both of them are trying to succeed in the training game, which makes both of them become stronger. However, existing image inpainting methods based on GANs are generally a few. Chen and Hu[12] proposed a GAN-based progressive inpainting method for semantic image inpainting. Vitoria et al.[14] built an improved version of WGAN with the incorporation of Generator and Discriminator network. Dong et al.[1] proposed a DCGAN for filling the missing parts of images.

CNN-based and GAN-based architecture improved the accuracy of inpainting generally. For furthermore improvement, we combined GAN, Region Normalization, and Partial Convolution together.

3. Method

Our proposed model is based on a conditional GAN[9]. The generator G is a UNet-like architecture, replacing all convolutional layers with partial convolutional layers[7]. We implement Region Normalization[17] in the Generator, which normalizes corrupted and uncorrupted regions separately. The discriminator D is like the one used in pix2pix model[3], which takes both the corrupted image and target image as inputs.

3.1. Partial Convolutional Layer

Partial Convolution (PC) is the operation where the convolution output only depends on the unmasked inputs. The partial convolution at every location is expressed as:

$$x' = \begin{cases} \mathbf{W}^T(\mathbf{X} \odot \mathbf{M}) \frac{\text{sum}(\mathbf{1})}{\text{sum}(\mathbf{M})} + b, & \text{if sum}(\mathbf{M}) > 0 \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

Let W be the convolution filter weights, X and M be the feature and binary mask values for the current sliding window respectively, and $\frac{sum(1)}{sum(M)}$ is a scaling factor that adjusts for the varying amount of unmasked inputs. The mask will be updated after each partial convolution operation by the following rules:

$$m' = \begin{cases} 1, & \text{if sum}(\mathrm{M}) > 0 \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

3.2. Region Normalization

Region Normalization (RN) normalizes and transforms corrupted and uncorrupted regions separately, which can solve the mean and variance shift problem of normalization and also avoid information mixing in affine transformation.

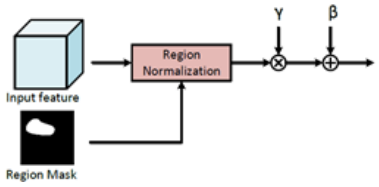
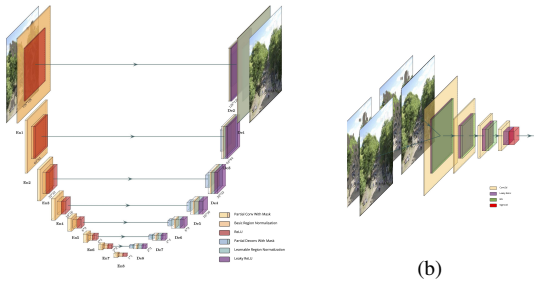


Figure 1: Region Normalization

3.3. Network Architecture

The design of the generator is based on U-Net architecture and we make some modifications to it. All convolution layers in the generator are replaced with PC, and all the normalizations used in the generator are RN. We implement affine transformation to different regions after RN in the encoding stage with two sets of learnable parameters. Pixel-wise affine transformation for the global region is used in the decoding stage, which can enhance the fusion of corrupted and uncorrupted regions. Nearest neighbor upsampling is used in the decoding stage. The skip links will concatenate two feature maps and two masks respectively, acting as the feature and mask inputs for the next partial convolution layer. Our discriminator is like the one used in pix2pix model. It takes both the corrupted image and target image as inputs so that it can observe both corrupted image to generated image and to ground truth.



(a) (b)

Figure 2: Our improved Generator and discriminator models: (a)Generator; (b)Discriminator

3.4. Loss functions

We used an ImageNet-pretrained VGG-16 to learn high-level features and some of our loss functions are based on these features. Our total generator loss is the combination of all the loss functions listed in[7] and BCE loss. Discriminator loss is BCE loss. Loss function details are found in

the supplementary file.

$$\begin{aligned}\mathcal{L}_{G_{total}} &= \lambda_{G\_BCE}\mathcal{L}_{G\_BCE} + \lambda_{valid}\mathcal{L}_{valid} \\ &+ \lambda_{hole}\mathcal{L}_{hole} + \lambda_{perceptual}\mathcal{L}_{perceptual} \\ &+ \lambda_{style}(\mathcal{L}_{styleOut} + \mathcal{L}_{styleComp}) + \lambda_{tv}\mathcal{L}_{tv}\end{aligned} \tag{3}$$

$$\mathcal{L}_D = \mathcal{L}_{D\_BCE} \tag{4}$$

4. Experiments

For simplicity and efficiency, we first manually screen and preprocess the Places datasets[19] and the Nvidia Mask datasets[8]. Then, we train both our improved CGAN-PC-RN model and the PC baseline model on the Google-colab platform. Finally, we evaluate our improved model against the baseline module through qualitative and quantitative results analysis, demonstrating how effective our model is.

4.1. Datasets

We use Places datasets for image training and testing. The Places datasets contain 10 million scene photographs, labeled with scene semantic categories, comprising a large and diverse list of the types of environments encountered in the world. Here we choose only the following four categories: Rainforest, Orchard, Tree Farm, and Vegetable Garden. Each category has 5000 images for training and 100 images for testing.

We use the Nvidia mask datasets for mask training and testing. The Nvidia mask datasets cover different hole-to-image area ratios: (0.01, 0.1], (0.1, 0.2], (0.2, 0.3], (0.3, 0.4], (0.4, 0.5], (0.5, 0.6]. And in order to achieve the best visual contrast effect, we select only the second category of the datasets. Then We select 8 masks from this subset of the mask dataset and train on them. We finally test with the same set of masks on the validation set for better comparison.

4.2. Training process

To evaluate the performance of our improved model, we choose the model using only Partial Convolutions as the baseline. And we train both two models with the same settings for image datasets, mask datasets, hyperparameters, etc.

After continuous experiments in practice, we finally choose the following optimal hyperparameters. We use the Adam optimizer[6] with learning rate of $2e-4$. We train our model with 16 threads and a batch size of 16 on Google-colab platform. For loss function Equation 3, we use the following parameters: $\lambda_{valid} = 1.0, \lambda_{hole} = 6.0, \lambda_{tv} = 0.1, \lambda_{prc} = 0.05, \lambda_{style} = 120.0, \lambda_{G\_BCE} = 1.0$. We perform approximately 2-3 iterations per second during the training process, and we finally take more than 28 hours to train 140k iterations.

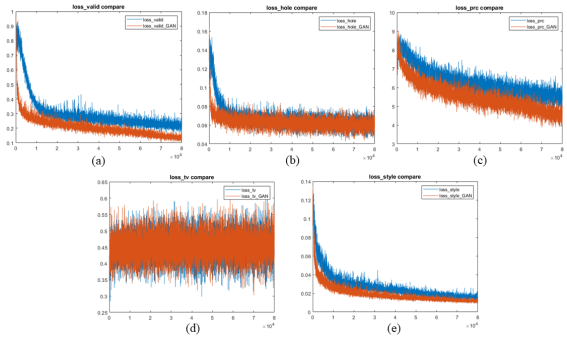


Figure 3: Comparison of the same type of loss function value between our improved model and the baseline model: (a)valid loss; (b)hole loss; (c)perceptual loss; (d)total variance Loss; (e)style loss

Comparisons of five separate loss function values of the same type of two models are as shown in Figure 3, while comparisons of total loss functions of the two models are as shown in Figure 4. Obviously, the loss function of our improved model decreases faster and lower as the number of iterations increases than baseline.

For our improved CGAN-PC-RN model, we can also get the discriminator loss values and the generator loss values. As shown in Figure 5, the discriminator loss fluctuates greatly at first and finally stabilizes at about 0.57, while the generator BCE loss fluctuates greatly at first and finally stabilizes at about 0.43. Based on that, we can also find the discriminator is relatively weak compared with the generator.

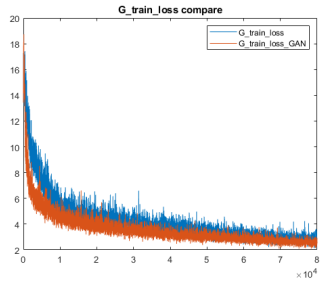


Figure 4: Comparison of total loss function values of two models

4.3. Qualitative evaluation

We show some of our inpainting test results after 80,000 iterations on Figure 6 and the results of comparison with the baseline on Figure 7. Observing from Figure 6, the repaired result is visually similar to the original image. And although we don't have many faces in our training set, our network

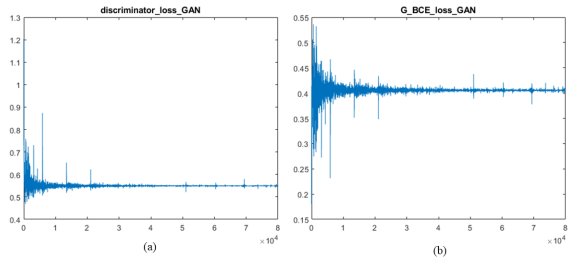


Figure 5: Discriminator loss and Generator loss for CGAN-PC-RN model: (a)Discriminator loss; (b)Generator BCE loss

Figure 6: Inpainting test results after 80,000 iterations: (a)Input images; (b)Synthetic images; (c)Ground truth

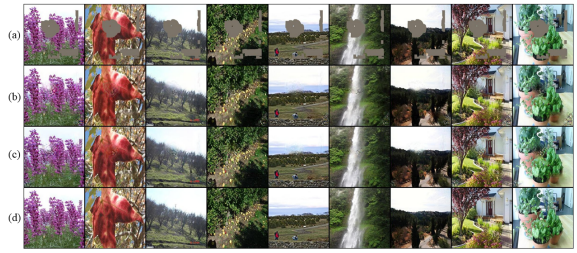


Figure 7: Inpainting test results after 80,000 iterations: (a)Input images; (b)Output with PC only; (c)Output with cGAN+PC+RN; (d)Ground truth

can still generate relatively reasonable content. Based on the visual comparison, our model shows realistic and coherent output images. Observing from Figure 7, output with our improved CGAN-PC-RN model shows unobvious artifact and better inpainting effect compared with baseline model.

4.4. Quantitative evaluation

By convention, we use L1 error, Peak Signal to Noise Ratio(PSNR), and SSIM[15] as numerical metrics to evaluate image inpainting results. The smaller L1 error explains that the gap between the result and Ground truth(GT) is smaller. The larger PSNR explains that the damaged image repair quality is better. And the larger SSIM shows that the structural similarity between the result and Ground truth(GT) is higher. As shown in Table 1, our improved CGAN-PC-RN model is better when compared with the baseline model after training 80,000 iterations. And our improved model after training 80,000 iterations for about 20 hours performs even better than the baseline model after training 140,000 iterations for about 25 hours.

	$L_1(\%)$	PSNR	SSIM
CGAN+PC+RN 80k iters	**4.371**	**22.605**	**0.776**
PC 80k iters	5.747	20.788	0.658
PC 140k iters	4.904	21.856	0.728

Table 1: Quantitative results

5. Conculsions

5.1. Discussion

We proposed a network that combines techniques of CGAN, Partial Convolution, and Region Normalization dealing with Image Inpainting tasks.

With GAN, the generator can produce more concrete and sharper images compared with the baseline model.

Given the same training iterations, the loss of our model is much smaller than that of the baseline model, and it still keeps on decreasing when the loss of the baseline model is starting to converge. Then we can make assumptions that our model trains faster and has a higher performance upper bound than the baseline model.

5.2. Future Work

Due to the limited time and GPU resources, we didn't fully train our model to compare the performance so that there are still some distinct artifacts on the edge areas, and we only trained the model on 20000 images (4 categories of Places2). We still have lots of things to verify, like the performance on large masks.

We observed that the discriminator is relatively weak compared with the generator. A complicated design of discriminator might help or we can feed the discriminator just the output of the masked area instead of the whole generated image.

References

[1] Junyu Dong, Ruiying Yin, Xin Sun, Qiong Li, Yuting Yang, and Xukun Qin. Inpainting of remote sensing sst images with deep convolutional generative adversarial network. *IEEE Geoscience and Remote Sensing Letters*, 16(2):173–177, 2019.

[2] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Commun. ACM*, 63(11):139–144, oct 2020.

[3] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[4] Kyong Hwan Jin and Jong Chul Ye. Annihilating filter-based low-rank hankel matrix approach for image inpainting. *IEEE Transactions on Image Processing*, 24(11):3498–3511, 2015.

[5] Norihiko Kawai, Tomokazu Sato, and Naokazu Yokoya. Diminished reality based on image inpainting considering background geometry. *IEEE Transactions on Visualization and Computer Graphics*, 22(3):1236–1247, 2016.

[6] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017.

[7] Guilin Liu, Fitsum A. Reda, Kevin J. Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Image inpainting for irregular holes using partial convolutions. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[8] Guilin Liu, Fitsum A. Reda, Kevin J. Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Nvidia irregular mask dataset. In *https://nv-adlr.github.io/publication/partialconv-inpainting*, 2018.

[9] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets, 2014.

[10] Jiangchun Mo and Yucai Zhou. The research of image inpainting algorithm using self-adaptive group structure and sparse representation. *Cluster Computing*, 22(3):7593–7601, 2019.

[11] Toshiki Nakamura, Anna Zhu, Keiji Yanai, and Seiichi Uchida. Scene text eraser. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 01, pages 832–837, 2017.

[12] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[13] Tijana Ružić and Aleksandra Pižurica. Context-aware patch-based image inpainting using markov random field modeling. *IEEE Transactions on Image Processing*, 24(1):444–456, 2015.

[14] Patricia Vitoria, Joan Sintes, and Coloma Ballester. Semantic image inpainting through improved wasserstein generative adversarial networks, 2018.

[15] Zhou 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*, 13(4):600–612, 2004.

[16] Zhaoyi Yan, Xiaoming Li, Mu Li, Wangmeng Zuo, and Shiguang Shan. Shift-net: Image inpainting via deep feature rearrangement. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[17] Tao Yu, Zongyu Guo, Xin Jin, Shilin Wu, Zhibo Chen, Weiping Li, Zhizheng Zhang, and Sen Liu. Region normalization for image inpainting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(07):12733–12740, Apr. 2020.

[18] Yanhong Zeng, Jianlong Fu, Hongyang Chao, and Baining Guo. Learning pyramid-context encoder network for high-quality image inpainting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[19] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(6):1452–1464, 2018.