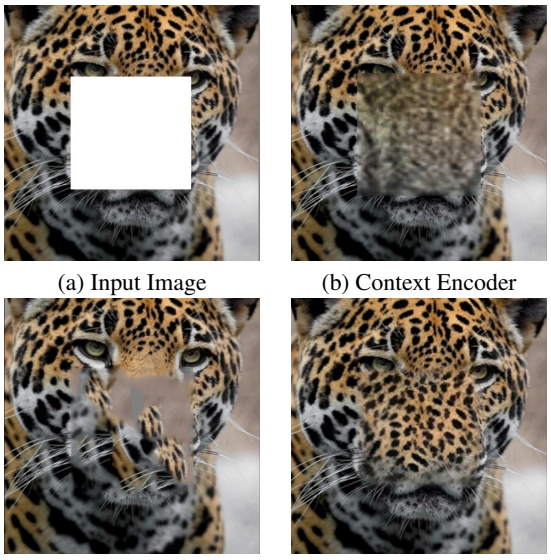
High-Resolution Image Inpainting using Multi-Scale Neural Patch Synthesis

Chao Yang[i 1], Xin Lu[§ 2], Zhe Lin[† 2], Eli Shechtman[‡ 2], Oliver Wang[* 2], and Hao Li[‖ 1,3,4]

[1]University of Southern California
[2]Adobe Research
[3]Pinscreen
[4]USC Institute for Creative Technologies

Abstract

Recent advances in deep learning have shown exciting promise in filling large holes in natural images with semantically plausible and context aware details, impacting fundamental image manipulation tasks such as object removal. While these learning-based methods are significantly more effective in capturing high-level features than prior techniques, they can only handle very low-resolution inputs due to memory limitations and difficulty in training. Even for slightly larger images, the inpainted regions would appear blurry and unpleasant boundaries become visible. We propose a multi-scale neural patch synthesis approach based on joint optimization of image content and texture constraints, which not only preserves contextual structures but also produces high-frequency details by matching and adapting patches with the most similar mid-layer feature correlations of a deep classification network. We evaluate our method on the ImageNet and Paris Streetview datasets and achieved state-of-the-art inpainting accuracy. We show our approach produces sharper and more coherent results than prior methods, especially for high-resolution images.



(a) Input Image (b) Context Encoder (c) PatchMatch (d) Our Result

Figure 1. Qualitative illustration of the task. Given an image (512×512) with a missing hole (256×256) (a), our algorithm can synthesize sharper and more coherent hole content (d) comparing with Context Encoder [32] (b) and Content-Aware Fill using PatchMatch [1] (c).

1. Introduction

Before sharing a photo, users may want to make modifications such as erasing distracting scene elements, adjusting object positions in an image for better composition, or recovering the image content in occluded image areas. These, and many other editing operations, require automated hole-filling (image completion), which has been an active research topic in the computer vision and graphics communities for the past few decades. Due to its inherent ambiguity and the complexity of natural images, general hole-filling remains challenging.

Existing methods that address the hole-filling problem fall into two groups. The first group of approaches relies on texture synthesis techniques, which fills in the hole by extending textures from surrounding regions [14, 13, 27, 26, 6, 12, 40, 41, 23, 24, 2]. A common idea in these techniques is to synthesize the content of the hole region in a coarse to fine manner, using patches of similar textures. In [12, 41], multiple scales and orientations are introduced to find better matching patches. Barnes et al. [2] proposed PatchMatch as a fast approximate nearest

[i]chaoy@usc.edu
[§]xinl@adobe.com
[†]zlin@adobe.com
[‡]elishe@adobe.com
[*]owang@adobe.com
[‖]hao@hao-li.com

一种方式是利用纹理合成相似的patch 进行填充
一种是利用外部数据库寻找相似数据填充缺失内容

what is this?

neighbor patch search algorithm. Although such methods are good at propagating high-frequency texture details, they do not capture the semantics or global structure of the image. The second group of approaches hallucinates missing image regions in a data-driven fashion, leveraging large external databases. These approaches assume that regions surrounded by similar context likely possess similar content [19]. This approach is very effective when it finds an example image with sufficient visual similarity to the query but could fail when the query image is not well represented in the database. Additionally, such methods require access to the external database, which greatly restricts possible application scenarios.

More recently, deep neural network is introduced for texture synthesis and image stylization [15, 16, 28, 3, 39, 22]. In particular, Phatak et al. [32] trained an encoder-decoder CNN (Context Encoder) with combined ℓ_2 and adversarial loss [17] to directly predict missing image regions. This work is able to predict plausible image structures, and is very fast to evaluate, as the hole region is predicted in a single forward pass. Although the results are encouraging, the inpainting results of this method sometimes lack fine texture details, which creates visible artifacts around the border of the hole. This method is also unable to handle high-resolution images due to the difficulty of training regarding adversarial loss when the input is large.

In a recent work, Li and Wand [28] showed that impressive image stylization results can be achieved by optimizing for an image whose neural response at mid-layer is similar to that of a content image, and whose local responses at a low convolutional layers resemble local responses from a style image. Those local responses were represented by small (typically 3×3) *neural patches*. This method proves able to transfer high-frequency details from the style image to the content image, hence suitable for realistic transfer tasks (e.g., transfer of the look of faces or cars). Nevertheless, transferring of more artistic styles are better addressed by using gram matrices of neural responses [15].

To overcome the limitations of aforementioned methods, we propose a hybrid optimization approach that leverages the structured prediction power of encoder-decoder CNN and the power of neural patches to synthesize realistic, high-frequency details. Similar to the style transfer task, our approach treats the encoder-decoder prediction as the global content constraint, and the local neural patch similarity between the hole and the known region as the texture constraint.

More specifically, the content constraint can be constructed by training a global content prediction network similar to Context Encoder, and the texture constraint can be modeled with the image content surrounding the hole, using the patch response of the intermediate layers using the pre-trained classification network. The two constraints can be optimized using backpropagation with limited-memory BFGS. In order to further handle high-resolution images with large holes, we propose a multi-scale neural patch synthesis approach. For simplicity of formulation, we assume the test image is always cropped to 512×512 with a 256×256 hole in the center. We then create a three-level pyramid with step-size two, downsizing the image by half at each level. It renders the lowest resolution of a 128×128 image with a 64×64 hole. We then perform the hole filling task in a coarse-to-fine manner. Initialized with the output of content prediction network at the lowest level, at each scale (1) we perform the joint optimization to update the hole, (2) upsample to initialize the joint optimization and set content constraint for the next scale. We then repeat this until the joint optimization is finished at the highest resolution (Sec. 3).

We show experimentally that the proposed multi-scale neural patch synthesis approach can generate more realistic and coherent results preserving both the structure and texture details. We evaluate the proposed method quantitatively and qualitatively on two public datasets and demonstrate its effectiveness over various baselines and existing techniques as shown in Fig. 1 (Sec. 4).

The main contributions of this paper are summarized as follows:

- We propose a joint optimization framework that can hallucinates missing image regions by modeling a global content constraint and local texture constraint with convolutional neural networks.
- We further introduce a multi-scale neural patch synthesis algorithm for high-resolution image inpainting based on the joint optimization framework.
- We show that features extracted from middle layers of the neural network could be used to synthesize realistic image contents and textures, in addition to previous works that use them to transfer artistic styles.

优化后的图像，中层响应类似于内容图像，底层相应类似于风格图像

从神经网络的中间层提取的特征可以用于合成逼真的图像内容和纹理

2. Related Work

Structure Prediction using Deep Networks Over the recent years, convolutional neural networks have significantly advanced the image classification performance, as presented in [25, 36, 37, 20]. Meanwhile, researchers use deep neural networks for structure prediction [29, 4, 30, 7, 38, 17, 18, 21, 9, 31], semantic segmentation [29, 4, 30], and image generation [17, 18, 7, 31]. We are motivated by the generative power of deep neural network and use it as the backbone of our hole-filling approach. Unlike the image generation tasks discussed in [11, 17, 18, 7], where the input is a random noise vector and the output is an image, our goal is to predict the content in the hole, conditioned on the *known image regions*. Recently, [32]

编码解码器预测结果作为内容约束
丢失部分和已知区域的局部神经patch 相似度作为纹理约束
使用分类网络的中间层patch 响应来model 纹理约束

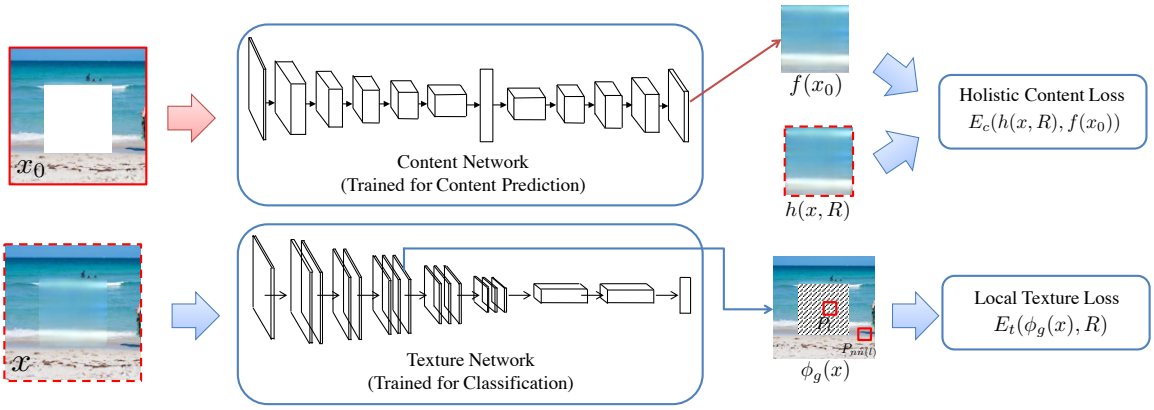


Figure 2. Framework Overview. Our method solves for an unknown image x using two loss functions, the holistic content loss (E_c) and the local texture loss (E_t). At the smallest scale, the holistic content loss is conditioned on the output of the pre-trained content network given the input x_0 ($f(x_0)$). The local texture loss is derived by feeding x into a pre-trained network (the texture network) and comparing the local neural patches between R (the hole) and the boundary.

使用ce的输出来初始化我们算法的最小规模的神经patch

proposed an encoder-decoder network for image inpainting, using the combination of the ℓ_2 loss and the adversarial loss (Context Encoder). In our work, we adapt Context Encoder as the global content prediction network and use the output to initialize our multi-scale neural patch synthesis algorithm at the smallest scale.

Style Transfer In order to create realistic image textures, our work is motivated by the recent success of neural style transfer [15, 16, 28, 3, 39, 22]. These approaches are largely used to generate images combining the "style" of one image and the "content" of another image. Our technique is motivated by the astounding performance of neural style transfer. In particular, we show neural features are also extremely powerful to create fine textures and high-frequency details of natural images.

3. The Approach

3.1. Framework Overview

We seek an inpainted image $\tilde{x}$ that optimizes over the loss function, which is formulated as a combination of three terms: the holistic *content* term, the local *texture* term, and the *tv-loss* term. The content term is a global structure constraint that captures the semantics and the global structure of the image, and the texture term models the local texture statistics of the input image. We first train the content network and use it to initialize the content term. The texture term is computed using the VGG-19 network [35](Figure 2) pre-trained on ImageNet.

To model the content constraint, we first train the holistic **content network** f. The input is an image with the central squared region removed and filled with the mean color, and the ground truth image x_t is the original content in the center. We trained on two datasets, as discussed in Section 4. Once the content network is trained, we can use the output of the network $f(x_0)$ as the initial content constraint for joint optimization.

纹理项的目标是保证生成的部分纹理与外面一致

The goal of the texture term is to ensure that the fine details in the missing hole are similar to the details outside of the hole. We define such similarity with *neural patches*, which have been successfully used in the past to capture image styles. In order to optimize the texture term, we feed the image x into the pre-trained VGG network (we refer to this network as local **texture network** in this paper) and enforce that the response of the small (typically 3×3) neural patches inside the hole region are similar to neural patches *outside* the hole at predetermined feature layers of the network. In practice we use the combination of *relu3_1* and *relu4_1* layers to compute the neural features. We iteratively update x by minimizing the joint content and texture loss using limited-memory BFGS.

The proposed framework naturally applies to the high-resolution image inpainting problem using multiscale scheme. Given a high-resolution image with a large hole, we first downsize the image and obtain a reference content using the prediction of the content network. Given the reference content we optimize w.r.t. the content and texture constraints at the low resolution. The optimization result is then upsampled and used as the initialization for joint optimization at the fine scales. In practice, we set the number of scales to be 3 for images of size 512×512.

We describe the details of the three loss terms in the following.

3.2. The Joint Loss Function

Given the input image x_0 we would like to find the unknown output image x. We use R to denote a hole region in x, and R^ϕ to denote the corresponding region in a feature map $\phi(x)$ of the VGG-19 network. $h(\cdot)$ defines the operation of extracting a sub-image or sub-feature-map in a rectangular region, i.e. $h(x, R)$ returns the color content of x within R, and $h(\phi(x), R^\phi)$ returns the content of $\phi(x)$ within R^ϕ, respectively. We denote the content

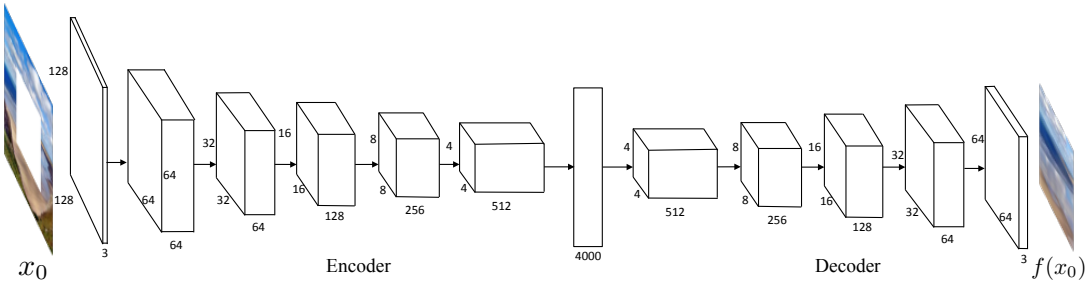


Figure 3. The network architecture for structured content prediction. Unlike the ℓ_2 loss architecture presented in [32], we replaced all ReLU/ReLU leaky layers with the ELU layer [5] and adopted fully-connected layers instead of channel-wise fully-connected layers. The ELU unit makes the regression network training more stable than the ReLU leaky layers as it can handle large negative responses during the training process.

network as f and the texture network as t.

At each scale $i = 1, 2..., N$ (N is the number of scales), the optimal reconstruction (hole filling) result $\tilde{x}$ is obtained by solving the following minimization problem:

$$\begin{aligned}\tilde{x}_{i+1} = \quad & \arg\min_x E_c(h(x, R), h(x_i, R)) \\ & +\alpha E_t(\phi_t(x), R^\phi) + \beta\Upsilon(x) \end{aligned} \tag{1}$$

where $h(x_1, R) = f(x_0)$, $\phi_t(\cdot)$ represents a feature map (or a combination of feature maps) at an intermediate layer in the texture network t, and α is a weight reflecting the importance between the two terms. Empirically setting α and β to be $5e^{-6}$ balances the magnitude of each loss and gives best results in our experiment.

The first term E_c in Equation 1 which models **the holistic content constraint** is defined to penalize the ℓ_2 difference between the optimization result and the previous content prediction (from the content network or the result of optimization at the coarser scale):

$$E_c(h(x, R), h(x_i, R)) = \| h(x, R) - h(x_i, R) \|_2^2 \tag{2}$$

The second term E_t in Equation 1 models **the local texture constraint**, which penalizes the discrepancy of the texture appearance inside and outside the hole. We first choose a certain feature layer (or a combination of feature layers) in the network t, and extract its feature map ϕ_t. For each local query patch P of size $s \times s \times c$ in the hole R^ϕ, we find its most similar patch outside the hole, and compute the loss by averaging the distances of the query patch and its nearest neighbor.

$$\begin{aligned} & E_t(\phi_t(x), R) = \\ & \frac{1}{|R^\phi|} \sum_{i \in R^\phi} \| h(\phi_t(x), P_i) - h(\phi_t(x), P_{nn(i)}) \|_2^2 \end{aligned} \tag{3}$$

where $|R^\phi|$ is the number of patches sampled in the region R^ϕ, P_i is the local neural patch centered at location i, and $nn(i)$ is the computed as

$$nn(i) = \arg\min_{j \in \mathcal{N}(i) \wedge j \notin R^\phi} \| h(\phi_t(x), P_i) - h(\phi_t(x), P_j) \|_2^2 \tag{4}$$

where $\mathcal{N}(i)$ is the set of neighboring locations of i excluding the overlap with R^ϕ. The nearest neighbor can be fast computed as a convolutional layer, as shown in [28].

We also add the TV loss term to encourage smoothness:

$$\Upsilon(x) = \sum_{i,j}((x_{i,j+1} - x_{i,j})^2 + (x_{i+1,j} - x_{i,j})^2) \tag{5}$$

3.3. The Content Network

A straightforward way to learn the initial content prediction network is to train a regression network f to use the response $f(x)$ of an input image x (with the unknown region) to approximate the ground truth x_g at the region R. Recent studies have used various loss functions for image restoration tasks, for instance, ℓ_2 loss, SSIM loss [42, 10, 33], ℓ_1 loss [42], perceptual loss [22], and adversarial loss [32]. We experimented with ℓ_2 loss and adversarial loss. For each training image, the ℓ_2 loss is defined as:

$$L_{l2}(x, x_g, R) = \| f(x) - h(x_g, R) \|_2^2 \tag{6}$$

The adversarial loss is defined as:

$$\begin{aligned} L_{adv}(x, x_g, R) = \max_D E_{x \in \mathcal{X}}[\log(D(h(x_g, R))) \\ + \log(1 - D(f(x)))] \end{aligned} \tag{7}$$

where D is the adversarial discriminator.

We use the joint ℓ_2 loss and the adversarial loss the same way as the Context Encoder [32]:

$$L = \lambda L_{l2}(x, x_g, R) + (1 - \lambda) L_{adv}(x, x_g, R) \tag{8}$$

where λ is 0.999 in our implementation.

3.4. The Texture Network

We use the VGG-19[35] network pre-trained for ImageNet classification as the texture network, and use the *relu3_1* layer and the *relu4_1* layer to calculate the texture term. We found using a combination of *relu3_1* and *relu4_1* leads to more accurate results than using a single layer. As an alternative, we tried to use the content network discussed in the previous section as the texture network, but found the results are of lower quality than using the pre-trained VGG-19. This can be explained by the fact that the VGG-19 network was trained for semantic classification, so features of its intermediate layers manifest strong invariance w.r.t. texture distortions. This helps infer more accurate reconstruction of the hole content.

4. Experiments

This section evaluates our proposed approach visually and quantitatively. We first introduce the datasets and then compare our approach with other methods, demonstrating its effectiveness in high-resolution image inpainting. At the end of this section we show a real world application where we remove distractors from photos.

Datasets We evaluate the proposed approach on two different datasets: Paris StreetView [8] and ImageNet [34]. Labels or other information associated with these images are not used. The Paris StreetView contains 14,900 training images and 100 test images. ImageNet has 1,260,000 training images, and 200 test images that are randomly picked from the validation set. We also picked 20 images with distractors to test out our algorithm for distractor removal.

Experimental Settings We first compare our method with several baseline methods in the low-resolution setting (128×128). First, we compared with results of Context Encoder trained with ℓ_2 loss. Second, we compare our method with the best results that Context Encoder have achieved using adversarial loss, which is the state-of-the-art in the area of image inpainting using deep learning. Finally, we compare with the results of Content-Aware Fill using PatchMatch algorithm from Adobe Photoshop. Our comparisons demonstrate the effectiveness of the proposed joint optimization framework.

While comparisons with baselines show the effectiveness of the overall joint optimization algorithm and the role of the texture network in joint optimization, we further analyze the separate role of the content network and the texture network by changing their weights in the joint optimization.

Finally, we show our results on high-resolution image inpainting and compare with Content-Aware Fill and Context Encoder (ℓ_2 and adversarial loss). Note that for Context Encoder the high-resolution results are acquired by directly upsampling from the low-resolution outputs. Our approach shows significant improvement in terms of the visual quality.

Quantitative Comparisons We first compare our method quantitatively with the baseline methods on low-resolution images (128×128) on the Paris StreetView dataset. Results in Table 1 show that our method achieves highest numerical performance. We attribute this to the nature of our method – it can infer the correct structure of the image where Content-Aware Fill fails, and can also synthesize better image details comparing with the results of Context Encoder (Fig. 4). In addition, we argue that the quantitative evaluation may not be most effective measure of the inpainting task given that the goal is to generate realistic-looking content, rather than exact same content that was in the original image.

Method	Mean L1 Loss	Mean L2 Loss	PSNR
Context Encoder ℓ_2 loss	10.47%	2.41%	17.34 dB
Content-Aware Fill	12.59%	3.14%	16.82 dB
Context Encoder (ℓ_2 + adversarial loss)	10.33%	2.35%	17.59 dB
Our Method	**10.01%**	**2.21%**	**18.00 dB**

Table 1. Numerical comparison on Paris StreetView dataset. Higher PSNR value is better. Note % in the Table is to facilitate reading.

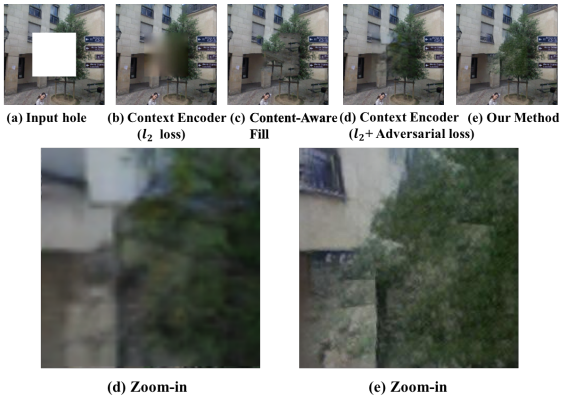


Figure 4. Comparison with Context Encoder (ℓ_2 loss), Context Encoder (ℓ_2 loss + adversarial loss) and Content-Aware Fill. We can see that our approach fixes the wrong textures generated by Content-Aware Fill, and is also more clear than the output of Context Encoder.

The effects of content and texture networks One ablation study we did was to drop the content constraint term and only use the texture term in the joint optimization. As shown in Fig. 8, without using the content term to guide the optimization, the structure of the inpainting results is completely incorrect. We also adjusted the relative weight between the content term and the texture term. Our finding is that by using more content constraint, the result is more consistent with the initial prediction of the content network but may lack high frequency details. Similarly, using more texture term gives sharp result but does not guarantee the overall image structure is correct (Fig. 6).

The effect of the adversarial loss We analyze the effect of using adversarial loss in training the content net-

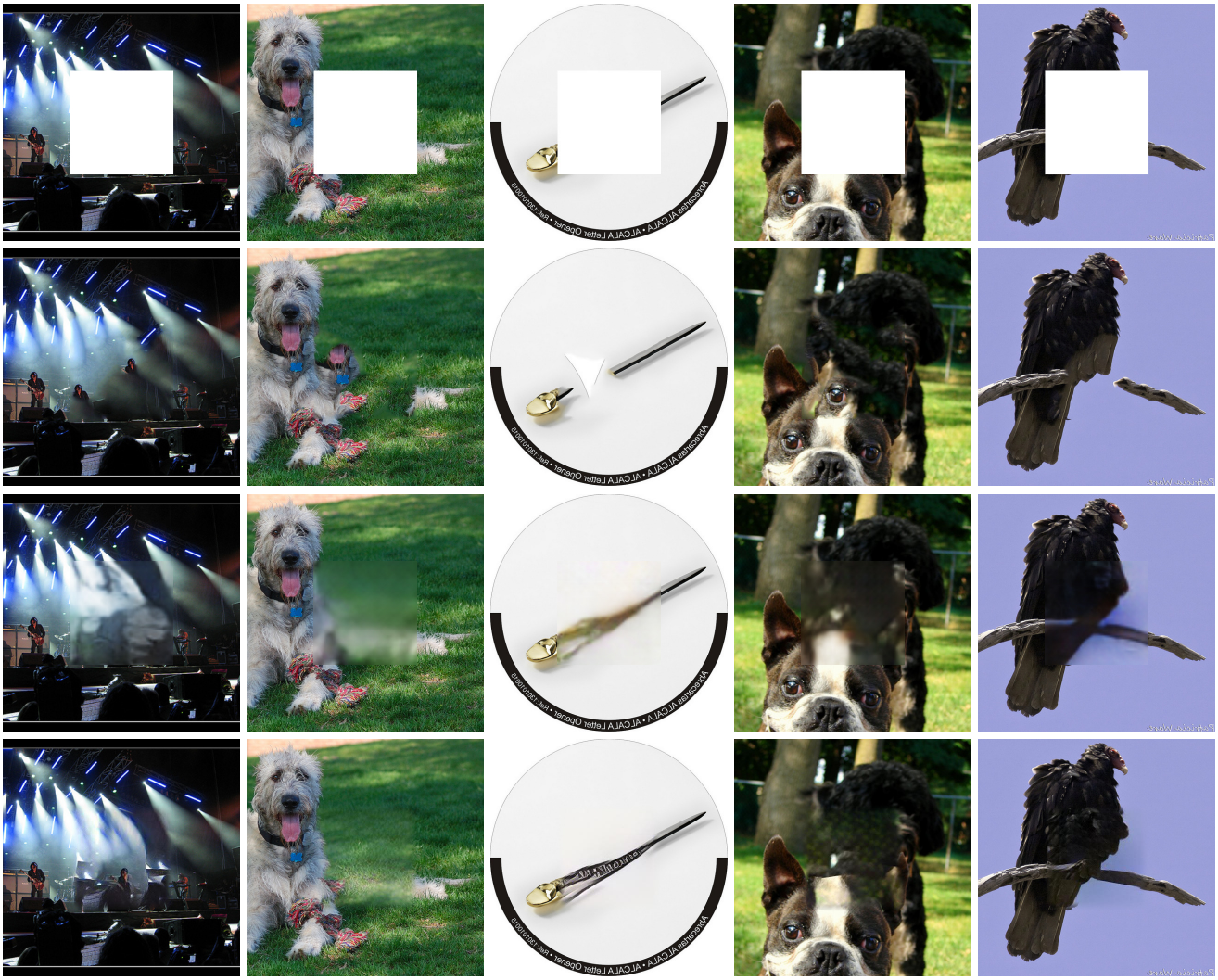


Figure 5. Visual comparisons of ImageNet result. From top to bottom: input image, Content-Aware Fill, Context Encoder (ℓ_2 and adversarial loss), our result. All images are scaled from 512×512 to fit the page size.

(a) Input image (b) $\alpha = 1e-6$ (c) $\alpha = 1e-5$ (d) $\alpha = 4e-5$

Figure 6. The effect of different texture weight α.

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

Figure 7. (a) Output of content network trained with ℓ_2 loss (b) The final result using (a). (c) Output of content network trained with ℓ_2 and adversarial loss. (d) The final result using (c).

work. One may argue without using the adversarial loss, the content network is still able to predict the structure of the image and the joint optimization will calibrate the textures later. However we found that the quality of the initialization given by the content network is important to the final result. When the initial prediction is blurry (using ℓ_2 loss only), the final result becomes more blurry as well comparing with using the content network trained with both ℓ_2 and adversarial loss (Fig. 7).

High-Resolution image inpainting We demonstrate our result of high-resolution image (512×512) inpainting in Fig. 5 and Fig. 10 and compare with Content-Aware Fill and Context Encoder (ℓ_2 + adversarial loss). Since Context Encoder only works with 128x128 images and when the input is larger, we directly upsample the 128×128 output to 512×512 using bilinear interpolation. In most of the results, our multi-scale, iterative approach combines the advantage of the other approaches, producing results

(a) (b) (c)

Figure 8. Evaluation of different components. (a) input image. (b) result without using content constraint. (c) our result.

with coherent global structure as well as high-frequency details. As shown in figures, a significant advantage of our approach over Content-Aware Fill is that we are able to generate new textures as we do not propagate the existing patches directly. However, one disadvantage is that given our current implementation, our algorithm takes roughly 1 min to fill in a 256×256 hole of a 512×512 image with a Titan X GPU, which is significantly slower than Content-Aware Fill.

Figure 9. Failure cases of our method.

Real-World Distractor Removal Scenario Finally, our algorithm is easily extended to handle arbitrary shape of holes. We first use a bounding rectangle to cover the arbitrary hole, which is again filled with mean-pixel values. After proper cropping and padding such that the rectangle is positioned at the center, the image is given as input to the content network. In the joint optimization, the content constraint is initialized with the output of the content network inside the arbitrary hole. The texture constraint is based on the region outside the hole. Fig. 11 shows several examples and its comparison with Content-Aware Fill algorithm (note that Context Encoder is unable to handle arbitrary holes explicitly so we do not compare with it here).

5. Conclusion

We have advanced the state of the art in semantic inpainting using neural patch synthesis. The insight is that the texture network is very powerful in generating high-frequency details while the content network gives strong prior about the semantics and global structure. This may be potentially useful to other applications such as denoising, superresolution, retargeting and view/time interpolation. There are cases when our approach introduces discontinuity and artifacts (Fig. 9) when the scene is complicated. In addition, the speed remains a bottleneck of our algorithm. We aim to address these issues in future work.

6. Acknowledgment

This research is supported in part by Adobe, Oculus & Facebook, Huawei, the Google Faculty Research Award, the Okawa Foundation Research Grant, the Office of Naval Research (ONR) / U.S. Navy, under award number N00014-15-1-2639, the Office of the Director of National Intelligence (ODNI) and Intelligence Advanced Research Projects Activity (IARPA), under contract number 2014-14071600010, and the U.S. Army Research Laboratory (ARL) under contract W911NF-14-D-0005. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of ODNI, IARPA, ARL, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purpose notwithstanding any copyright annotation thereon.

References

[1] https://research.adobe.com/project/content-aware-fill. 1

[2] C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. *TOG*, 28(3):24:1–24:11, 2009. 1

[3] A. J. Champandard. Semantic style transfer and turning two-bit doodles into fine artwork. In *arXiv:1603.01768v1*, 2016. 2, 3

[4] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. In *ICLR*, 2015. 2

[5] D. Clevert, T. Unterhiner, Hochreiter, and S. Fast and accurate deepnetwork learning by exponential linear unites (ELUS). In *ICLR*, 2016. 4

[6] A. Criminisi, P. Pérez, and K. Toyama. Object removal by exemplar-based inpainting. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages II–721 – II–728 vol.2, 2003. 1

[7] E. Denton, S. Chintala, A. Szlam, and R. Fergus. Deep generative image models using a laplacian pyramid of adversarial networks. In *arXiv:1506.05751v1*, 2015. 2

[8] C. Doersch, S. Singh, A. Gupta, J. Sivic, and A. Efros. What makes paris look like paris? *TOG*, 31(4), 2012. 5

[9] A. Dosovitskiy and T. Brox. Inverting visual representations with convolutional networks. In *arXiv:1602.02644v1*, 2015. 2

[10] A. Dosovitskiy and T. Brox. Generating images with perceptual similarity metrics based on deep networks. In *arXiv:1602.02644v1*, 2016. 4

[11] A. Dosovitskiy, J. T. Springenberg, and T. Brox. Learning to generate chairs with convolutional neural networks. In *CVPR*, pages 1538–1546, 2015. 2

[12] I. Drori, D. Cohen-Or, and H. Yeshurun. Fragment-based image completion. *TOG*, 22(3):303–312, 2003. 1

[13] A. A. Efros and W. T. Freeman. Image quilting for texture synthesis and transfer. In *ACM SIGGRAPH*, pages 341–346, 2001. 1

Figure 10. Visual comparisons of Paris Streetview result. From top to bottom: input image, Content-Aware Fill, Context Encoder (ℓ_2 and adversarial loss) and our result. All images are scaled from 512×512 to fit the page size.

Figure 11. Arbitrary object removal. From left to right: input image, object mask, Content-Aware Fill result, our result.

[14] A. A. Efros and T. K. Leung. Texture synthesis by non-parametric sampling. In *ICCV*, pages 1033–1038, 1999. 1

[15] L. A. Gatys, A. S. Ecker, and M. Bethge. A neural algorithm of artistic style. In *arXiv:1508.06576v2*, 2015. 2, 3

[16] L. A. Gatys, A. S. Ecker, and M. Bethge. Texture synthesis and the controlled generation of natural stimuli using convolutional neural networks. In *NIPS*, 2015. 2, 3

[17] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS*, pages 2672–2680, 2014. 2

[18] K. Gregor, I. Danihelka, A. Graves, D. Rezende, and D. Wierstra. DRAW: A recurrent neural network for image generation. In *arXiv:1511.08446v2*, 2015. 2

[19] J. Hays and A. A. Efros. Scene completion using millions of photographs. *TOG*, 26(3), 2007. 2

[20] K. He, X. Y. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *arXiv:1409.1556v6*, 2016. 2

[21] D. Im, C. Kim, H. Jiang, and R. Memisevic. Generating images with recurrent adversarial networks. In *arXiv:1602.05110v1*, 2016. 2

[22] J. Johnson, A. Alahi, and F. Li. Perceptual losses for real-time style transfer and super-resolution. In *arXiv:1603.08155v1*, 2016. 2, 3, 4

[23] N. Komodakis. Image completion using global optimization. In *CVPR*, pages 442–452, 2006. 1

[24] N. Komodakis and G. Tziritas. Image completion using efficient belief propagation via priority scheduling and dynamic pruning. *TIP*, 16(11):2649–2661, 2007. 1

[25] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1106–1114, 2012. 2

[26] V. Kwatra, I. Essa, A. Bobick, and N. Kwatra. Texture optimization for example-based synthesis. *TOG*, 24(3):795–802, 2005. 1

[27] V. Kwatra, A. Schödl, I. Essa, G. Turk, and A. Bobick. Graphcut textures: Image and video synthesis using graph cuts. In *ACM SIGGRAPH*, pages 277–286, 2003. 1

[28] C. Li and M. Wand. Combining markov random fields and convolutional neural networks for image synthesis. In *arXiv:1601.04589v1*, 2016. 2, 3, 4

[29] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, pages 1–9, 2015. 2

[30] J. Long, E. Shelhamer, and T. Darrell. Learning deconvolution network for semantic segmentation. In *ICCV*, pages 1520–1528, 2015. 2

[31] A. Oord, N. Kalchbrenner, and K. Kavukcuoglu. Pixel recurrent neural networks. In *arXiv:1601.06759v1*, 2016. 2

[32] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. Efros. Context encoders: Feature learning by inpainting. In *CVPR*, 2016. 1, 2, 4

[33] K. Ridgeway, J. Snell, B. Roads, R. Zemel, and M. Mozer. Learning to generate images with perceptual similarity metrics. In *arXiv:1511.06409v1*, 2015. 4

[34] 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. *IJCV*, 115(3):211–252, 2015. 5

[35] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2014. 3, 4

[36] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, pages 1–9, 2015. 2

[37] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Very deep convolutional networks for large-scale image recognition. In *arXiv:1409.1556v6*, 2015. 2

[38] L. Theis and M. Bethge. Generative image modeling using spatial LSTMs. In *arXiv:1603.03417v1*, 2015. 2

[39] D. Ulyanov, V. Lebedev, A. Vedaldi, and V. Lempitsky. Texture networks: Feed-forward synthesis of textures and stylized images. In *arXiv:1603.03417v1*, 2016. 2, 3

[40] Y. Wexler, E. Shechtman, and M. Irani. Space-time video completion. In *CVPR*, pages 120–127, 2001. 1

[41] M. Wilczkowiak, G. Brostow, B. Tordoff, and R. Cipolla. Hole fill through photomontage. In *BMVC*, pages 492–501, 2005. 1

[42] H. Zhao, O. Gallo, I. Frosio, and J. Kautz. Is l_2 a good loss function for neural networks for image processing? In *arXiv:1511.08861v1*, 2015. 4