Image Inpainting with Deep Learning

September 16, 2019

Image inpainting is the task of filling in holes in the input image so that the output looks consistent as a whole. Research in using deep learning for image inpainting has become popular since 2017. This document is written as I read some of the papers in this topic. These include Pathak et al. [7], Iizuka et al. [2], Yeh et al. [11], Yu et al.'s CVPR 2018 paper [13], Liu et al.'s ECCV 2018 paper [5], and Yu et al.'s ICCV 2019 paper [14].

1 Pathak et al. (CVPR 2016)

- The input to the system is an image with a some region marked out with white pixels. The pixels that are not marked out are called the context.
- The authors call their network the *context encoder*. It has an encoder-decoder architecture.
  - The encoder compresses the context into a small latent feature representation.
  - The decoder converts the representation into a full image with the missing region filled in.
- The whole network is trained to simultaneously minimize two lossses:
  - The *reconstruction loss* is just the L2 loss over the pixels.
  - The *adversarial loss* is the loss computed from the a discriminator network trained to distinguish the network-filled images from the real images before hole introduction.
- Using only reconstruction loss results in blurry results. Adding adversarial loss yields much sharper results.
- The paper also reports that the features computed by the encoder are also useful for image retrieval.

1.1 The Architecture

- The encoder is connected to the decoder with a *channel-wise fully connected layer*.
  - This allows the unit to reason about the whole image.
  - It is not economical to feed the latent vectors to a fully connected layer which produces latent vectors of the same size.
  - The channel-wise fully-connected layer, on the other hand, operates on each chanels independently, which making it much more economical than the full-blown fully connected layers.
- The decoder consists of a series of up-convolution layers (i.e., fractionally strided convolution), each followed by a ReLU.

1.2 The Loss Function

- Symbol definitions:
  - Let x denote the ground truth image.
  - Let the context encoder be denoted by the function F.
  - Let $\hat{M}$ be the binary mask corresponding to the dopped region. The value of $\hat{M}$ is 1 when the pixel is dropped, and 0 if the pixel is the same as that of x.
- The reconstruction loss is the normalized masked L2 distance:

  $$\mathcal{L}_{rec} = E_x[\|\hat{M} \odot (x - (F(x \odot (1 - \hat{M}))))\|_2^2]$$

  where $\odot$ is the element-wise multiplication operation.
- The authors also experimented with the L1 loss but found no difference between it and the L2 loss.
- The reconstruction loss causes the network to produce rough outline of the predicted object. However, the results do not have high frequency details.
- The advesarial loss depends on the discriminator network D. The function is the standard GAN loss applied to the real and generated images:

  $$\begin{aligned}\mathcal{L}_{adv,D} &= -E_x[\log D(x) + \log(1 - D(F(x \odot (1 - \hat{M}))))] \\ \mathcal{L}_{adv,F} &= -E_x[\log D(F(x \odot (1 - \hat{M})))]\end{aligned}$$

- It is interesting to note that D does not text the masked image as input. That is, D is a standard GAN discriminator rather than a conditional GAN discriminator. The authors observe that giving the masked image as an input would allow the discriminator to easily exploit the discrepancy at mask boundaries. Making the discriminator looking at the whole image would encourage the generator to create wholely realistic images.
- The whole loss function for the context encoder is the linear combination between the two losses:

  $$\mathcal{L}_F = \lambda_{rec}\mathcal{L}_{rec} + \lambda_{adv}\mathcal{L}_{adv,F}.$$

  For the discriminator, it includes only the adversarial loss:

  $$\mathcal{L}_D = \lambda_{adv}\mathcal{L}_{adv,F}.$$

1.3 Region Masks

- The paper generate masks from random masks in the PASCAL VOC 2012 dataset. The masks are deformed and pasted at random places in the ground truth image.
- The masks cover up to 1/4 of the ground truth image.
- The masks are irregular, so it prevents the network from learning features that are specific to particular mask shapes.

1.4 Implementation Details

- The paper uses $\lambda_{rec} = 0.999$ and $\lambda_{adv} = 0.001$.
- The encoder and decoder are similar to the encoder in the DCGAN paper [9]. The bottleneck layer has 4,000 nodes.
- The paper uses higher learning (10x of the discriminator) rate for the context encoder.

2 Iizuka et al. (SIGGRAPH 2017)

- This work extends the Context Encoder, making it applicable to arbitrary image size and adding a method to sure local consistency of the filled pixels.
- Data involved:
  - x is the ground truth image (without holes). This is the desired output of the system.
  - M_c is the binary mask for the holes. The value is 1 for pixels that are holes, and 0 for pixels retaining the value of the original image.
  - An image with whole is created by filling the hole pixels with the mean of the pixels of the training sets. The system takes this image along with M_c as input.
- The work uses three networks:
  - The *completion network* takes in the image with holes and the hole mask and tries to fill the holes.
  - The *global context discriminator* takes in the full image, resized to 256×256, and tries to determine whether it is generated by the completion network or is a ground truth image.
  - The *local context discriminator* takes in a 128×128 image that is centered around a whole and also tries to do the same job as the global discriminator. During trainng, there's always one whole in a training example, and this hole is smaller than 128×128. However, the network as a whole can be used to filled in multiple holes if they are not too large.
- The insight is that the global discriminator assesses whether the filled image is consistent as a whole while the local discriminator asesses whether the filled pixels are consistent with its surrounding.

2.1 Architectural Details

- Details of the completion network:
  - It takes as input a 256×256 image with a whole and the hole mask as an extra channel.
  - Through convolutional layers, the image is resized down by a factor of 4 while increasing the number channels. This downscaled image would then pass through the bottleneck layers.
  - The bottleneck layers are mostly convolutional layers which preserve image size. 4 of these layers are *dilated* convolutional layers [12], which are included in order to increase the receptive field of each pixel in the network.
  - After the bottleneck layers, the image is resized back to its original size by transposed convolution layers.
- Details of the global context discriminators:
  - It takes in a 256×256 image.
  - It contains 6 convolutional layers followed by a fully connected layer.
  - It outputs a 1024 dimentional vector.
  - All convolutional layers have 5×5 filters and a stride of 2, which results in downsizing the input image by a factor of 2.
- The local context discriminator follows the same pattern as the global context discriminator. It outputs a 1024 dimensional vector but takes in a 128×128 image centered at the hole to be filled.
- The outputs of the discriminators are concatenated to form a single 2048 dimensional vector, which is then passed to a fully connected network to produce a single scalar, the reality score.

2.2 Loss Functions

- Let the completion network be denoted by the function $C(x, M_c)$, and let the discriminator network (including both the global and local discriminators) be denoted by $D(x, M_d)$.
- The first loss employed is the reconstruction loss:

$$\mathcal{L}_{rec} = E_{x,M_c}[\|M_c \odot (C(x, M_c) - x)\|_2^2]$$

- The adversarial loss is the standard GAN loss:

$$\begin{aligned}\mathcal{L}_{adv,D} &= -E_{x,M_d}[\log D(x, M_d)] - E_{x,M_c}[\log(1 - D(C(x, M_c), M_c))]\\ \mathcal{L}_{adv,C} &= -E_{x,M_c}[\log D(C(x, M_c), M_c)]\end{aligned}$$

- Combining the two losses, we have:

$$\begin{aligned}\mathcal{L}_D &= -\alpha E_{x,M_d}[\log D(x, M_d)] - \alpha E_{x,M_c}[\log(1 - D(C(x, M_c), M_c))]\\ \mathcal{L}_C &= E_{x,M_c}[\|M_c \odot (C(x, M_c) - x)\|_2^2] - \alpha E_{x,M_c}[\log D(C(x, M_c), M_c)]\end{aligned}$$

2.3 Training Details

- One of the more interesting part of the paper is their training scheme. It has three phases:
  - The completion network is trained only with the reconstruction loss for some time.
  - The completion network is fixed, and the discriminators are trained for some time.
  - All networks are then trained jointly.
- The paper uses $\alpha = 0.0004$ and batch size of 96. The first phase has $90,000$ iteration, the second phase $10,000$, and the last phase $500,000$.

2.4 Postprocessing

- The inpainted pixels can still have color discrepancy with the surrounding pixels. The paper fixes this by applying pixel blending techniques: fast marching [10] followed by Poisson image blending [8].

3 Yeh et al. (CVPR 2017)

- This is a completely different take on image editing. Other papers uses a single feedforward network to do the job. This paper assumes that there's a pretrained GAN for a dataset. The paper's algorithm searches for a latent code that would make the GAN produces an image that is "closest" to the corrupted image.
  - I think this is an interesting perspective, but it is much more expensive than the specialized network approach.
- Let us denote the pretrained generator and discriminator by G and D, respectively.
  - The generator takes in a latent vector $\mathbf{z}$, drawn from a probability distribution $p_{\mathbf{z}}$, and produces an image $G(\mathbf{z})$ that looks like it comes from the training dataset.
  - The discriminator takes in an image $\mathbf{x}$ and produces a "reality score" $D(\mathbf{x})$, which is the probability that the image comes from the training dataset rather than being generated by the generator.
- The paper attempts to compute the latent vector $\hat{\mathbf{z}}$ that is the "closest" to the corrupted image. It then uses $\hat{\mathbf{z}}$ to generate $G(\hat{\mathbf{z}})$, which is then blended with the uncorrupted part to form the final output.

- Let $\mathbf{y}$ be the corrupted image, and $\mathbf{M}$ be the binary mask indicating the missing part. The closest $\hat{\mathbf{z}}$ is given by:

$$\hat{\mathbf{z}} = \arg\min_{\mathbf{z}} \Big(\mathcal{L}_c(\mathbf{z}|\mathbf{y}, \mathbf{M}) + \mathcal{L}_p(\mathbf{z}) \Big)$$

  where

  - $\mathcal{L}_c$ denotes the *context loss*, which constrains the generated image to be close to $\mathbf{y}$ in the unmasked parts, and
  - $\mathcal{L}_p$ denotes the *prior loss*, which penalizes unrealistic images:

$$\mathcal{L}_p = \lambda \log(1 - D(G(\mathbf{z}))).$$

- The definition of the context loss $\mathcal{L}_c$ is closely related to how $G(\hat{\mathbf{z}})$ is used. Basically, we will only use pixels that are marked as corrupted. As a result, it is not advisable to consider pixels in $G(\hat{\mathbf{z}})$ that are far away from the holes.

- The paper captures the notion that only pixels near the holes should be used by introducting the weighting image $\mathbf{W}$ where

$$\mathbf{W}_i = \begin{cases} 0, & \mathbf{M}_i = 0 \\ \sum_{j \in N(i)} \frac{1 - \mathbf{M}_j}{|N_i|}, & \mathbf{M}_i \neq 0 \end{cases}.$$

  Here,

  - i denotes a pixel index.
  - $N(i)$ denotes the set of neighbor pixels of i. The paper uses a window of size 7.
  - $|N(i)|$ denotes the size of $N(i)$.

- The context loss is given by:

$$\mathcal{L}_c(\mathbf{z}|\mathbf{y}, \mathbf{M}) = \|\mathbf{W} \odot (G(\mathbf{z}) - \mathbf{y})\|_1.$$

  This is slightly different from other papers, which just multiplies $\mathbf{M}$ to the difference. The main difference is that pixels near the boundaries of the holes are downweighted.

- After $\hat{\mathbf{z}}$ is found and $G(\hat{\mathbf{z}})$ is computed, it must be blended with the uncorrupted part. The final image $\hat{\mathbf{x}}$ is given by:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x}} \|\nabla \mathbf{x} - \nabla G(\hat{\mathbf{z}})\|_2^2 \text{ subjected to } \mathbf{x}_i = \mathbf{y}_i \text{ where } \mathbf{M}_i = 0.$$

  This can be solved by the solver of the Poisson image editing paper [8].

- For the GAN, the paper uses DCGAN [9]. The value of λ is 0.003. The paper finds $\hat{\mathbf{z}}$ by starting with a random $\mathbf{z}$ and running ADAM for 1500 iterations.

4 Yu et al. (CVPR 2018)

- This paper consists of two main parts:

  - It proposes several improvements to the network of IIzuka et al. [2].
  - It proposes a new unit called the *contextual attention unit*, which allows the network to copy similar patches that are far away from the missing parts.

4.1 Improved Network

- The input consists of:
  - a 256×256 image with holes filled with white pixels, and
  - a binary mask, indicating which pixels are the holes.
- The network is divided into two parts.
  - The first part takes the input and makes the coarse, blurry prediction. That is, it fills the white pixels with color pixels.
  - The second part takes the coarse prediction as input and predict the fine result.
- The coarse part is trained with the reconstruction loss explicitly. The refinement part is trained with both the reconstruction loss and the GAN losses.
- Changes in details from Iizuka et al.:
  - The actual network has fewer parameters than that of Iizuka et al.
  - Mirror padding is used for all convolutional layers.
  - No batch normalization.
  - ELUs [1] are used instead of ReLUs.
  - Instead of using the hyperbolic tangent or the sigmoid function at the layer that produces the output, the paper just clips the output into the range.
- The network uses a modified version of the WGAN-GP loss in both the global and local critics. The modification is that only the gradients of the hole pixels are taken into account. That is, the gradient term is given by:

$$\lambda E_{\hat{\mathbf{x}}}\Big[\big(\|\nabla_{\hat{\mathbf{x}}} D(\hat{\mathbf{x}} \odot (1-\mathbf{m}))\|_2 - 1\big)^2\Big].$$

  Here, the mask is 0 for hole pixels and 1 otherwise, and $\hat{\mathbf{x}}$ represents an image obtained by interpolating between a real image and a fake image.

  From the paper, I think the reality score is still based on the whole input image.
- The reconstruction loss is modified so that the hole pixels that are far from the hole boundaries have smaller weights. The weight is computed as γ^l where l is the distance of the pixel to the nearest non-hole pixel, and γ is set to 0.99.

4.2 Contextual Attention Layer

- The contextual attention layer copies simlar backgound patches that can be far away from the coarsely filled holes (called "foreground" in the paper).
- Let's say that the patch size is 3×3. Let
  - $b_{x,y}$ denotes the background patch centered at Pixel (x, y).
  - $f_{x',y'}$ denotes the foreground patch centered at Pixel x', y'.
- We first extract all the $b_{x,y}$ patches and shape them as a convolutional filter of size $N_b \times C \times 3 \times 3$, where N_b is the number of background patches, and C is the number of channels.

- Next, the similarly between all pairs of foreground and background patches is computed by the cosine similarity:

$$s_{x,y,x',y'} = \left\langle \frac{f_{x,y}}{\|f_{x,y}\|}, \frac{b_{x',y'}}{\|b_{x',y'}\|} \right\rangle.$$

  This can be computed efficiently by convolving the convolutional filter we just constructed in the last step with the foreground image, normalizing when appropriate. This results in a $N_b \times h_f \times w_f$, where h_f and w_f are the height and the width of the foreground image, respectively.

- The similarity computed above is then turned into a discrete probability distribution over the background patch by performing scaled softmax along the (x', y') dimention (i.e., along the channels):

$$s^*_{x,y,x',y'} = \frac{\exp(\lambda s_{x,y,x',y'})}{\sum_{x',y'} \exp(\lambda s_{x,y,x',y'})}.$$

- The convolution filter created from the background patches is then reused as a deconvolution filter against the output of the last step. The values of overlapped pixels are averaged.

- Before performing the deconvolution in the last step, the paper performs a diffusion-like steps on the probability distribution.

  The idea is that $s^*_{x,y,x',y'}$ should be similar to $s^*_{x+a,y+b,x'+a,y+b}$ for small values of a and b. The paper makes this the case by performing the following propagation steps:

$$\hat{s}_{x,y,x',y'} = \sum_{-k \leq i \leq k} s^*_{x+i,y,x'+i,y}$$
$$\tilde{s}_{x,y,x',y'} = \sum_{-k \leq j \leq k} \hat{s}_{x,y+j,x',y+j}$$

  Then, the $\tilde{s}_{x,y,x',y'}$ is normalized along the (x', y') dimension. This can be efficiently implemented by a convolution.

- N_b and $w_f \times h_f$ can be large. To reduce it, the paper proposes two strategies:
  - Extracting background patches with strides.
  - Downscaling the foreground image before convolution and upscaling attention map after propagation.

4.3 The Refinement Network

- The refinement network contains two encoder branches.
  - The first branch directly refines the coarse image. It has dilated convolutions in the bottleneck.
  - The second branch has a contextual attention layer in the bottleneck so that it can borrows patches from the background.

- The output of the two decoder branches are concatenated and fed to a single decoder.

- Let's say the input image is denoated by $\mathbf{x}$, the mask by $\mathbf{m}$. Let $\mathbf{x}'$ denote the output of the refinement network. The final output of the whole generator is:

$$G(\mathbf{x}, \mathbf{m}) = \mathbf{x} \odot \mathbf{m} + \mathbf{x}' \odot (1 - \mathbf{m}).$$

5 Liu et al. (ECCV 2018)

- Methods we have discussed so far have several problems:
  - They do not work well with arbitrary hole shapes. This is because they are sometimes trained with rectangular holes [7] or holes that are at the center of the image [2, 13].
  - The filled pixels often have color discrepancy with the surrounding pixels, which must be removed by post processing.
  - Convolutional layers process hole pixels as if they are valid pixels. So, they have dependence on the filled colors. This can results in:
    - Lack of texture in hole regions.
    - Obvious color contrast.
    - Artificial edge responses around the holes.

  Liu et al. try to solve these problems.

- The paper introduces the *partial convolution layer*.
  - It has two steps:
    - A masked and renormalized convolution.
    - A mark update.
  - Given a binary mask, the result of the convolution depends only on the non-hole pixels.
  - The mask update step removes masking of holes pixels that are corruped by non-hole pixels.
  - More specifically, let $\mathbf{W}$ be the convolutional filter weights, $\mathbf{X}$ be the feature values at the current sliding window, and $\mathbf{M}$ be the corresponding mask (hole = 0, background = 1). The partial convolution value $\mathbf{x}'$ at a particular location is given by:

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

    Here, $\mathbf{1}$ has the same shape as $\mathbf{M}$, but all of its components' values are 1.
  - The convolution can be easiler implemented by just multiplying the mask before applying the convolution.
  - After the above convolution, the mask is updated. The value of the new mask m' at a given location is given by:

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

  - The weight update can be easily implemented by a convolution with filter whose weight is all one followed by a clamp.

- Network architecture.
  - The paper uses a UNet architecture similar to the one used by the Pix2pix paper [3].
  - All convolution layers are replached by partial convolution layers.
  - The skip links concatenate two feature maps and two masks. The masks will act on there respective features for the next convolutional layer.
    - It is unclear in the paper how to compute the new mask. It presume the mask that does not come from the skip link will be the new mask because this is the only sensible way.

  - The last partial convolution layer will also have the original image and mask as inputs. This is to make it possible for the network to copy non-hole pixels.
  - Note that the network *does not* automatically copy non-hole pixels from the input. It must learn to do so by itself.

- The paper uses several loss functions. The main surprise is that, in contrast to other papers, it does not use the adversarial loss at all.
  - We first introduce some notations.
    * Let $\mathbf{I}_{in}$ denote the input image.
    * Let $\mathbf{M}$ denote the initial binary mask.
    * Let $\mathbf{I}_{out}$ denote the output image.
    * Let $\mathbf{I}_{gt}$ denote the groundtruth image.
    * Let $\mathbf{I}_{comp}$ denote $\mathbf{I}_{out}$ where non-hole pixels are replaced by the original pixels.
    * $N(\cdot)$ denote the number components of the given vector.
  - The *per-pixel losses* are given by:

$$\mathcal{L}_{hole} = \frac{1}{N(\mathbf{I}_{gt})} \|(1 - \mathbf{M}) \cdot (\mathbf{I}_{out} - \mathbf{I}_{gt})\|_1$$
$$\mathcal{L}_{valid} = \frac{1}{N(\mathbf{I}_{gt})} \|\mathbf{M} \cdot (\mathbf{I}_{out} - \mathbf{I}_{gt})\|_1.$$

  - The paper uses the perceptual loss [4].
    * The paper uses the outputs of the *pool1*, *pool2*, and *pool3* layers of VGG-16.
    * In general, let's say the paper uses P layers. Let Ψ_p denote the output of the pth used layer.
    * Let's say that the output of the pth use layer has shape $C_p \times H_p \times W_p$.
    * The *Gram matrix* of the output of the pth used layer is given by:

$$G_p(\mathbf{I}) = \frac{1}{C_p H_p W_p} \Psi_p(\mathbf{I})^T \Psi_p(\mathbf{I}).$$

    Here, we view $\Psi_p(\mathbf{I})$ as a $C_p \times (H_p W_p)$ matrix. So, the Gram matrix has size $C_p \times C_p$.
  - The *perceptual (content) loss* term is given by:

$$\mathcal{L}_{perceptual} = \sum_{p=1}^{P} \frac{\|\Psi_p(\mathbf{I}_{out}) - \Psi_p(\mathbf{I}_{gt})\|_1}{N(\Psi_p(\mathbf{I}_{gt}))} + \sum_{p=1}^{P} \frac{\|\Psi_p(\mathbf{I}_{comp}) - \Psi_p(\mathbf{I}_{gt})\|_1}{N(\Psi_p(\mathbf{I}_{gt}))}$$

  - The *style loss* term is given by:

$$\mathcal{L}_{style} = \sum_{p=1}^{P} \frac{\|G_p(\mathbf{I}_{out}) - G_p(\mathbf{I}_{gt})\|_1}{N(G_p(\mathbf{I}_{gt}))} + \sum_{p=1}^{P} \frac{\|G_p(\mathbf{I}_{comp}) - G_p(\mathbf{I}_{gt})\|_1}{N(G_p(\mathbf{I}_{gt}))}$$

  - The *total variation loss* is a smoothing penalty on R, where R is the region of the 1-pixel dilation of the hole region. It encourages the filled pixels to transition smoothly into the background.

$$\mathcal{L}_{tv} = \sum_{(i,j) \in R, (i,j+1) \in R} \frac{\|\mathbf{I}_{comp}[i, j+1] - \mathbf{I}_{comp}[i, j]\|_1}{N(\mathbf{I}_{comp})} + \sum_{(i,j) \in R, (i+1,j) \in R} \frac{\|\mathbf{I}_{comp}[i+1, j] - \mathbf{I}_{comp}[i, j]\|_1}{N(\mathbf{I}_{comp})}$$

  - The total loss is loss the weighted linear combination of the above:

$$\mathcal{L}_{total} = \mathcal{L}_{valid} + 6\mathcal{L}_{hole} + 0.05\mathcal{L}_{perceptual} + 120\mathcal{L}_{style} + 0.1\mathcal{L}_{tv}.$$

    The weights are determined by hyperparameter search on 100 validation images. The paper did not mention which algorithm it used.
  - Insights about the loss.
    * Perceptual loss leads to *checkerboard artifacts*. However, leaving it out leads to *grid-shaped artifacts*.
    * The paper relies heavily on the style loss. Results are not good when the style loss weight is low. More specifically, low style loss weight leads to *fish scale artifacts*.

6 Yu et al. (ICCV 2019)

- This paper generalizes the partial convolution unit with the *gated convolution unit*. It also proposes a new loss network called the *SN-PatchGAN*.
  - The big advantage of the gated convolution unit is that it can incorporate signals other than the mask into the hole filling process. This means that the user sketch and other forms of suggestions can be taken into account when filling holes.
- The paper claims that partial convolution has several limitations:
  - It either classifies pixels to be hole or non-hole. This makes in incompatible with additional user inputs; that is, should this user input be a hole or a non-hole?
  - In the mask update, a hole pixel will be marked 1 if it is touch by a filter. This does not take into account how many non-hole pixels are under the filter. Moreover, all non-hole pixels will eventually disappear. The system would be more flexible if it is allowed to learn the mask by itself.
- The gated convolution unit is defined as follows:

$$\begin{aligned} Gating[y,x] &= \sum\sum W_g \cdot I \\ Feature[y,x] &= \sum\sum W_f \cdot I \\ O[y,x] &= \phi(Feature[y,x]) \odot \sigma(Gating[y,x]) \end{aligned}$$

  where σ is the sigmoid function, and ϕ is any activation function. W_g and W_f are two convolutional features.
- SN-PatchGAN:
  - The discriminator is a convolutional network.
  - The discriminator consists of 6 convolutional units. (See the paper for details.) Each unit is normalized by spectral normalization [6].
  - The output of the GAN is a 3D feature of shape $\mathbb{R}^{c\times h\times w}$ where $c = 256$, $h = H/32$ and $w = W/32$ where H and W are the original image height and width, respectively.
  - The hinge GAN loss is then applied to each $c \times h \times w$ location:

$$\begin{aligned} \mathcal{L}_D &= E_{x\sim p_{data}(x)}[ReLU(1 - D(x))] + E_{z\sim p_z(z)}[ReLU(1 + D(G(z)))] \\ \mathcal{L}_G &= -E_{z\sim p_z(z)}[D(G(z))]. \end{aligned}$$

    (Huh, the generator loss does not have ReLU?) The paper does not explicitly tell how these $c \times h \times w$ numbers are combined to a single number. I presume they just average them.

- The overall loss of the network is just the L1 pixel loss and then adversarial loss based on SN-PatchGAN. The ratio between the losses is 1 : 1.
- The overall architecture is the same as that of the 2018 paper by the same authors. All vanialla convolution layers are replaced by gated convolution layers.

References

[1] Clevert, D.-A., Unterthiner, T., and Hochreiter, S. Fast and accurate deep network learning by exponential linear units (elus). *CoRR abs/1511.07289* (2015).

[2] Iizuka, S., Simo-Serra, E., and Ishikawa, H. Globally and locally consistent image completion. *ACM Trans. Graph. 36*, 4 (July 2017), 107:1–107:14.

[3] Isola, P., Zhu, J., Zhou, T., and Efros, A. A. Image-to-image translation with conditional adversarial networks. *CoRR abs/1611.07004* (2016).

[4] Johnson, J., Alahi, A., and Li, F. Perceptual losses for real-time style transfer and super-resolution. *CoRR abs/1603.08155* (2016).

[5] Liu, G., Reda, F. A., Shih, K. J., Wang, T., Tao, A., and Catanzaro, B. Image inpainting for irregular holes using partial convolutions. *CoRR abs/1804.07723* (2018).

[6] Miyato, T., Kataoka, T., Koyama, M., and Yoshida, Y. Spectral normalization for generative adversarial networks. *CoRR abs/1802.05957* (2018).

[7] Pathak, D., Krähenbühl, P., Donahue, J., Darrell, T., and Efros, A. Context encoders: Feature learning by inpainting.

[8] Pérez, P., Gangnet, M., and Blake, A. Poisson image editing. In *ACM SIGGRAPH 2003 Papers* (New York, NY, USA, 2003), SIGGRAPH '03, ACM, pp. 313–318.

[9] Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *CoRR abs/1511.06434* (2015).

[10] Telea, A. An image inpainting technique based on the fast marching method. *J. Graphics, GPU, and Game Tools 9* (2004), 23–34.

[11] Yeh, R., Chen, C., Yian Lim, T., Schwing, A., Hasegawa-Johnson, M., and Do, M. Semantic image inpainting with deep generative models. In *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017* (United States, 11 2017), Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Institute of Electrical and Electronics Engineers Inc., pp. 6882–6890.

[12] Yu, F., and Koltun, V. Multi-scale context aggregation by dilated convolutions. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings* (2016).

[13] Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., and Huang, T. S. Generative image inpainting with contextual attention. *arXiv preprint arXiv:1801.07892* (2018).

[14] Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., and Huang, T. S. Free-form image inpainting with gated convolution. *arXiv preprint arXiv:1806.03589* (2019).