MVA
Introduction to Digital Imaging - Project Report

Generative Image Inpainting with Contextual Attention

Louis DUMONT - Antoine GUEDON
ENS Paris-Saclay

January 21, 2020

Abstract

We present a method for image completion, based on recent advances in Generative Adversarial Networks (GANs) and the introduction of a Contextual Attention layer that helps to re-use features from background patches in the reconstruction. We present the original method proposed by Iizuka *et al* [1], and emphasize the changes introduced by Yu *et al* [2]. Reproducing their experimental conditions, we then study the influence of several factors on the reconstruction performed by their trained model.

Introduction

Motivations and context

The task of image inpainting, or filling out missing pixels in an image, appears in a wide variety of problems, from image restoration to photo editing ([3]). First approaches to inpainting were based on copy-pasting patches from the background image on the missing regions. As useful as they can be, they lacked the capacity to introduce elements missing from the background (to "hallucinate") and to capture properly semantics larger than patch size.
The development of Convolutional Neural Networks (CNNs) and Generative Adversarial Networks (GANs), that disrupted the field of image processing as a whole, led to other approaches to the problem of image inpainting. GANs, which relies on the idea of training jointly two networks, a generator, tasked with generating data from a certain distribution, and a critic, that evaluates the distance of the generated distribution to the target one, triggered enthusiasm for their applications to image generation and style transfer ([4]). The use of GANs for image inpainting was first proposed in the form of Context Encoders ([5]). Methods based on GANs often require post-processing steps, and typically suffer from the difficulty of training GANs. Recent progresses were made in that field with the introduction of Wassertein-GANs and the study of their training ([6], [7]).
The methods presented here build upon these ideas presenting a new architecture using two

critics, and introduce the Contextual Attention mechanism to encourage the model to use background patches for reconstruction.

Related works

Iizuka *et al.* [1] presented in 2017 a GAN-based approach for image completion that results in images that are both locally and globally consistent. With a fully-convolutional neural network, they were able to complete images of arbitrary resolutions by filling-in missing regions of any shape.
To this end, they used global and local context discriminators trained to distinguish real images from completed ones. The global discriminator looks at the full image to assess if it is coherent as a whole, while the local discriminator looks only at a small area centered at the completed region to evaluate the local consistency of the generated patches.
The image completion network is then trained to fool both context discriminator networks, which requires it to generate images that are indistinguishable from real ones with regard to overall consistency as well as in details.

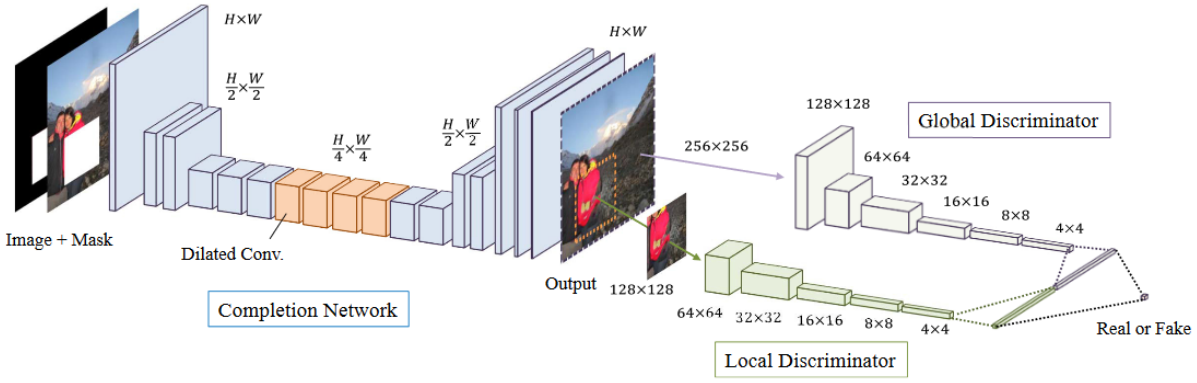

Figure 1: Overview of the architecture proposed by Iizuka *et al.* [1] for learning image completion. The completion network, which generates the inpainted image, is trained thanks to two discriminators (not used during training). Both discriminator act as GANs, the Global one taking as input the whole image while the Local one only uses the inpainted zone. The last layer of both network are concatenated before computing the output.

Content of the report

In this report, we will describe how Yu *et al.* improved the model proposed by Iizuka *et al.* [1] for inpainting using generative adversarial networks. We will start by doing some reminder about GANs and specifically Wasserstein GANs, describing in the same time how the structure of such a network is adapted to a Wasserstein optimization problem. Then we will present the different improvements proposed by the authors and how they impact the network. To this end we will

especially focus on how they managed to include features about contextual information on a global scale in a convolutional network. Finally, we will study the performance of this model by presenting results of experiences, some presented in the paper but others performed by ourselves.

1 Network structure

The architecture designed by the authors of the article [2] is heavily inspired by the work of Iizuka *et al.* [1] that focuses on keeping both local and global consistency in image completion. Indeed, they improve this model and propose a way to incorporate contextual attention in the network. With such an approach, they were able to outperform state-of-the-art methods.

Reminder : Generating data with GANs

Given a training set, *Generative Adversarial Networks* (or GANs) aim to create new data with the same statistics. To this end, two neural networks contest with each other in a game : the *generative network* generates candidates and the *discriminative network* evaluates them in terms of data distribution. In other words, the generative network learns to map from a latent space to a data distribution of interest, while the discriminative network distinguishes candidates produced by the generator from the true data distribution. Consequently, the generative network's training objective is to increase the error rate of the discriminative network by producing better images, while the discriminative network becomes more skilled at detecting synthetic images thanks to the application of backpropagation in both network. The generator is typically a deconvolutional neural network, and the discriminator is a convolutional neural network.

Structure of the generative network

The generator network takes an image with white pixels filled in the holes and a binary mask indicating the hole regions as input pairs, and outputs the final completed image. The input is paired with a corresponding binary mask to handle holes with variable sizes, shapes and locations. Consequently, even if during training the input to the network is a 256×256 image with a rectangle missing region sampled randomly, the trained model can take an image of different sizes with multiple holes in it.

In image inpainting tasks, the size of the receptive fields should be sufficiently large, and Iizuka et al. [1] adopted dilated convolution for that purpose, instead of using channel-wise fully connected layers.

Definition 1.1 (Dilated convolution) *For $l \in \mathbb{N}$, the l-dilated convolution of an image f with a kernel k along a 1D axis is defined as follows:*

$$(k *_l f)_t = \sum_{\tau=-\infty}^{\infty} k_\tau \times f_{t-l\tau}$$

The idea is to obtain an exponential growth (rather than linear) of the receptive field through the convolutional layers by increasing the dilation factor, and without increasing the number of

parameters (i.e. the size of the kernel) nor losing coverage and resolution.
For example, starting from a standard convolution (that is, a 1-dilated convolution), one can apply a 2^{l-1}-dilated convolution in the layer l to get a receptive field of size $(2^{l+1}-1)\times(2^{l+1}-1)$. No coverage is lost, and the receptive field grows exponentially without changing the number of parameters in the kernel.

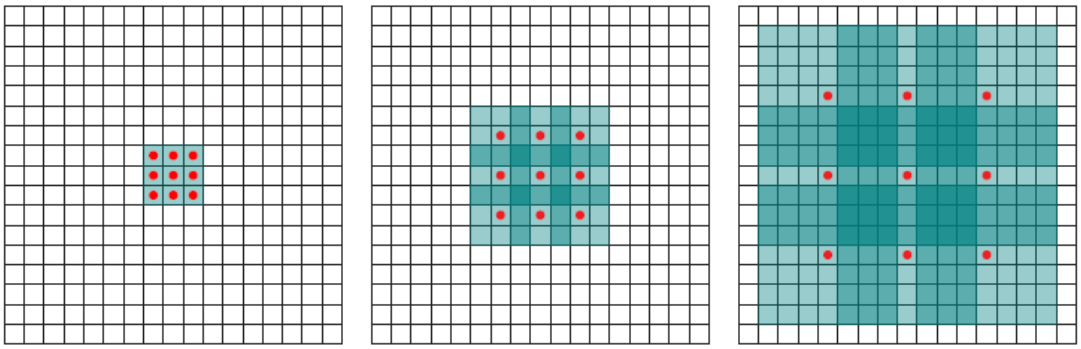

Figure 2: A geometric increase of the dilation factor through layers leads to an exponential expansion of the receptive field without any loss in coverage or resolution (all the blue squares above are covered). (a) f_1 is produced from an image f_0 and a kernel k with a standard or 1-dilated convolution. Consequently each element of f_1 has a receptive field of 3×3. (b) f_2 is produced from f_1 and a kernel of the same size than before by a 2-dilated convolution. Then each element in f_2 has a receptive field of 7×7. (c) f_3 is produced from f_2 by a 4-dilated convolution. Therefore each element in f_3 has a receptive field of 15×15.

1.1 The coarse-to-fine structure of the generative network

To further enlarge the receptive fields and stabilize training, the authors introduce a two-stage coarse-to-fine network architecture. A first network makes an initial coarse prediction, and a second network takes the coarse prediction as inputs and predict refined results. The coarse network is trained with the reconstruction loss explicitly, while the refinement network is trained with the reconstruction as well as GAN losses.
Remark: This two-stage network architecture is actually similar to residual learning or deep supervision ([8], [9]).

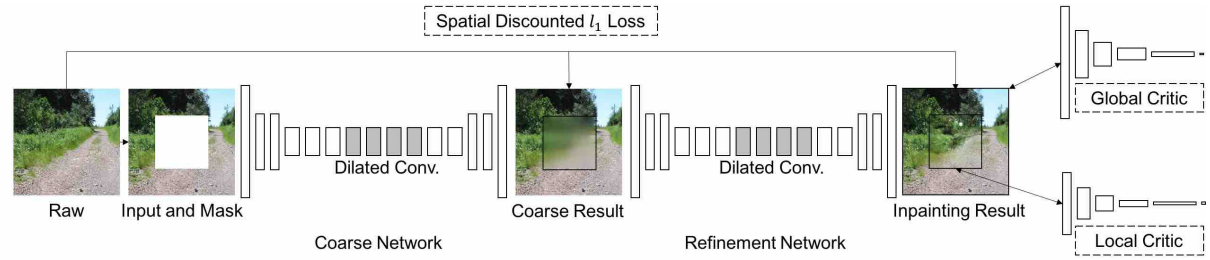

Figure 3: Generative inpainting framework designed by the authors. The coarse network is trained with reconstruction loss explicitly, while the refinement network is trained with reconstruction loss, global and local WGAN-GP adversarial loss.

Concerning the implementation, mirror padding is used in all convolution layers (the image is extended with mirror copies of itself on the borders so that the convolution can be computed on the edges) and batch normalization layers are removed (the authors found that it deteriorates color coherence). Furthermore, contrary to Iizuka *et al.* [1] who use ReLU activation functions and tanh or sigmoid functions for output filter values, the authors use ELUs activation function and simply clip the output filter values.

Definition 1.2 (ELU function) *The Exponential Linear Unit function (or ELU) is defined as follows:*

$$ELU(x) = \begin{cases} x & if\ x > 0 \\ \alpha(exp(x) - 1) & otherwise \end{cases}$$

where $\alpha > 0$.

Using ELUs as activation functions presumably compensates the removal of batch normalization layers: as they map $\mathbb{R}^*_-$ to negative values, the meant unit activation is pushed closer to zero. Contrarily to Leaky ReLus, their negative values saturate to -1, which makes them more robust to noise ([10]).

1.2 Spatially discounted reconstruction loss

Intuitively, when it comes to inpainting, big holes in an image lead to far more uncertainty than smaller ones. Indeed, almost all pixels of a small hole (what we call the "foreground pixels") are close to a "background pixel" on the boundary (i.e. non-missing pixel). On the contrary, when it comes to a bigger hole, there is much more ambiguity concerning the content to infer for a pixel in the center of the hole since every background pixels are far from it, and even for a human it becomes harder to imagine with precision what content could fill in.

In other words, missing pixels near the hole boundaries appear to have much less ambiguity than those in the center. This problem is reminiscent of an issue observed in reinforcement learning: when long-term rewards have large variations during sampling, people use temporal discounted rewards over sampled trajectories. Similarly, the authors of the paper introduce

spatially discounted reconstruction loss using a weight mask M: the weight of each pixel in the mask is computed as γ^l, where l is the distance of the pixel to the nearest known pixel.
The authors use discounted l_1 reconstruction loss in the implementation, and γ is set to 0.99 in all experiments.

1.3 Better results and consistency with WGANs

Inspired by Iizuka *et al.* [1], Yu *et al.* decided to encourage both global and local consistency by submitting a global-focused output and a local-focused output to two auxiliary context discriminator networks during training. However, instead of relying on DCGAN [1] for adversarial supervision, they use a modified version of WGAN-GP and compute two separate loss, one for the Global Discriminator and one for the Local Discriminator, instead on concatenating the last layers and computing a single loss.

The relevance of using WGANs was highlighted by Arjovsky *et al* ([6]): WGANs are linked to the Earth-Mover distance (or Wasserstein-1 distance), which has much nicer topological properties than the Jensen-Shannon one (implicitly underlying GANs), and thus ensures more reliably the correspondance between the convergence of the Generator parameters and the convergence of the generated distribution toward the target one. This is confirmed by empirical results ([6]).

Now, we will study the link between WGANs and the Earth-Mover or Wasserstein-1 distance; let's begin with a definition.

Definition 1.3 (Earth-mover or Wasserstein-1 distance) *The Earth-mover distance $W(\mathbb{P}_r, \mathbb{P}_g)$ between a generated and real data distribution $\mathbb{P}_g$ and $\mathbb{P}_r$ is defined as*

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \Gamma(\mathbb{P}_r, \mathbb{P}_g)} \int_{M \times M} \|x - y\|_1 \, d\gamma(x, y)$$

where $\Gamma(\mathbb{P}_r, \mathbb{P}_g)$ is the set of couplings between $\mathbb{P}_r$ and $\mathbb{P}_g$.
From a probabilistic point of view, we rewrite with random variables

$$W(\mathbb{P}_r, \mathbb{P}_g) = \inf_{\gamma \in \Gamma(\mathbb{P}_r, \mathbb{P}_g)} \mathbb{E}_{(x,y)\sim\gamma} \left[\|x - y\|_1\right]$$

The biggest asset of such a distance lies in a result about Kantorovitch-Rubenstein duality, that gives another expression of its value:

$$W(\mathbb{P}_r, \mathbb{P}_g) = \sup_{D \in \mathcal{D}} \mathbb{E}_{x\sim\mathbb{P}_r} \left[D(x)\right] - \mathbb{E}_{\tilde{x}\sim\mathbb{P}_g} \left[D(\tilde{x})\right]$$

where $\mathcal{D}$ is the set of 1-Lipschitz functions and $\mathbb{P}_g$ is the model distribution implicitly defined by $\tilde{x} = G(z)$ with z the input to the generator.
Our goal is to minimize on G (the parameters of the generative network) this loss that involves a parameter D whose aim is to maximize the error $\mathbb{E}_{x\sim\mathbb{P}_r} \left[D(x)\right] - \mathbb{E}_{\tilde{x}\sim\mathbb{P}_g} \left[D(\tilde{x})\right]$. D is asked to be lipschitzian, and is suitable to be approximated by a neural network. This is precisely the

reason why GAN are perfectly suited for Wasserstein optimization: using the dual formulation, the idea is to consider D as the parameter of a discriminative layer that aims to maximize the error made by the generator, while the generator tries to fool the discriminator by decreasing the error with backpropagation. Consequently, we compute and optimize a Wasserstein-1 loss with the help of a GAN: this is the purpose of a Wasserstein GAN, or WGAN.

To ensure the Lipschitz continuity during optimization, a simple way performed by Arjovsky *et al.* [6] is to clamp the weights. However, Gulrajani *et al.* [7] consider it to be a inefficient way to enforce a lipschitz constraint ("we find that weight clipping in WGAN leads to optimization difficulties, and that even when optimization succeeds the resulting critic can have a pathological value surface." See [7], p.3 for more information) and propose to enforce the constraint through a gradient penalty to obtain better results. Inspired by this work, Yu *et al.* introduce such a penalty term in their model.
The gradient penalty term is the following:

$$\lambda \mathbb{E}_{\hat{x} \sim \mathbb{P}_{\hat{x}}} \left(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1 \right)^2$$

where $\hat{x}$ is sampled from the straight line between points sampled from distribution $\mathbb{P}_g$ and $\mathbb{P}_r$. The reason is that the gradient of D^* at all points $\hat{x} = (1-t)x + t\tilde{x}$ on the straight line should point directly towards current sample $\tilde{x}$, meaning $\nabla_{\hat{x}} D^*(\hat{x}) = \frac{\tilde{x} - \hat{x}}{\|\tilde{x} - \hat{x}\|}$ ([7]).
However we only try to predict hole regions for image inpainting, thus the gradient penalty should only be computed in the regions to be inpainted. This is with multiplication of gradients and input mask m:

$$\lambda \mathbb{E}_{\hat{x} \sim \mathbb{P}_{\hat{x}}} \left(\|\nabla_{\hat{X}} D(\hat{x}) \odot (1 - m)\|_2 - 1 \right)^2$$

where the mask value is 0 for missing pixels and 1 for elsewhere, and $\odot$ is the point-wise multiplication. λ is a constant factor set to 10 in all experiments.
In a nutshell, the pixel-wise reconstruction loss is a supervision signal that teaches the generator to match missing regions with ground-truth patches, while WGANs learn to recognize plausible images and teach the generator to make its output plausible, no matter its relation with the ground truth.

Since the WGAN loss is based on the Wasserstein-1 distance (that consists in the sum of l_1-distances between pixels in its primal form), using the l_1 reconstruction loss creates homogeneity in the nature of the losses and makes the training not only easier but also stabler (in practice, the WGAN loss in its dual form is related to the l_1 distance metric by the constraint enforced; indeed WGAN requires the discriminator D to be lipschitzian w.r.t the l_1 distance).

1.4 Introducing contextual attention

Convolutional neural networks by nature only exploit the local information of the image in each of their layer. Consequently, they cannot borrow or replicate features more distant than their receptive fields. Yu *et al.* introduce the contextual attention mechanism to overcome this limitation and allow each pixel's features to be computed using information from distant locations.

The intuition behind the contextual attention layer is that it will learn how to borrow information from distant places so as to generate similar patches on the missing region. It is implemented to be differentiable and convolutional, so that it can be trained and used on arbitrary resolutions.

Strategy of the contextual layer

The coarse result we saw previously is sent to the refinement network that consists in two parallel encoders. One of them simply focuses on generating content with dilated convolutions. For each "foreground" patch (i.e. 3×3 patches centered on the pixels that were missing at the beginning), the other encoder computes attention scores with every 3×3 "background" patches. The idea is to match each foreground patch to a background one that is similar, i.e. from which feature information could be borrowed. The encoder finally computes what can be visualized as an attention map (a map on which each foreground pixel has a given color that corresponds to the location of the background pixels that are similar in the full image), and uses background patches as deconvolutional filters to reconstruct forward patches. The results of both encoders are aggregated before being sent in a decoder that gives the final output.

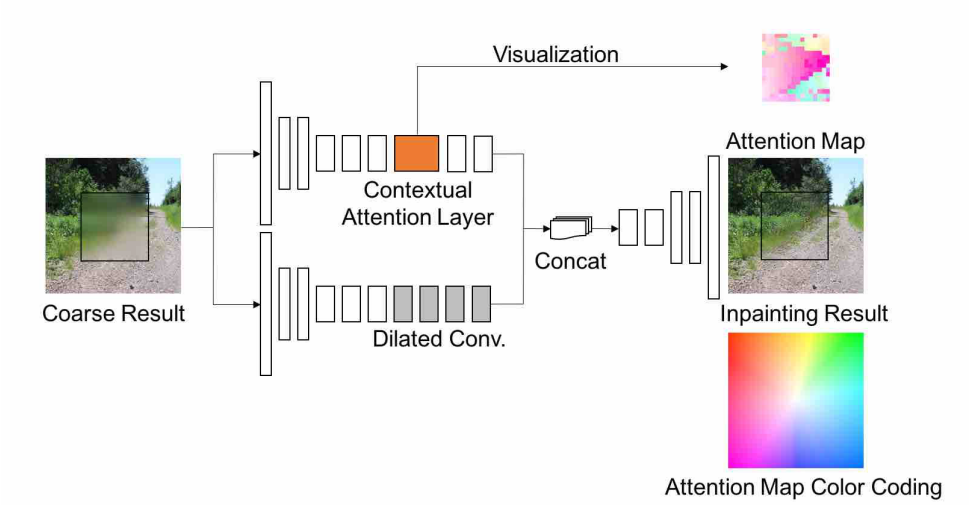

Figure 4: Once the coarse result is computed by the first stage of the generative network, it is fed into two streams. The first one focuses on attention, while the second one is another convolutional network. Their output are then concatenated for the final layer. For visualization of attention map, color indicates relative location of the most interested background patch for each pixel in foreground. For examples, white (center of color coding map) means the pixel attends on itself, pink on bottom-left, green means on top-right (from [2]).

During training, the outputs will then be judged by the discriminative network. Those that interpret correctly the attention scores (and which will certainly do better when analyzed by the discriminative network) will naturally be encouraged by backpropagation.

Computation of attention scores

As we said before, the idea is to match features of missing pixels (what we call foreground) to surroundings (or background). 3×3 patches are extracted from the background and reshaped as convolutional filters. Then a similarity score is measured as a normalized inner product between foreground patches $\{f_{x,y}\}$ and background patches $\{b_{x',y'}\}$ respectively centered in (x, y) and (x', y'):

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

Next the similarity is weighed with the application of scaled softmax along $x'y'$-dimension to get an attention score for each pair of patches:

Definition 1.4 (Attention score) *The attention score between a foreground patch $f_{x,y}$ and a background patch $b_{x',y'}$ respectively centered in (x, y) and (x', y') is defined as*

$$s^*_{x,y,x',y'} = softmax_{x',y'}(\lambda s_{x,y,x',y'})$$

where λ is a constant value.

Remark: The attention score is efficiently implemented as convolution and channel-wise softmax.

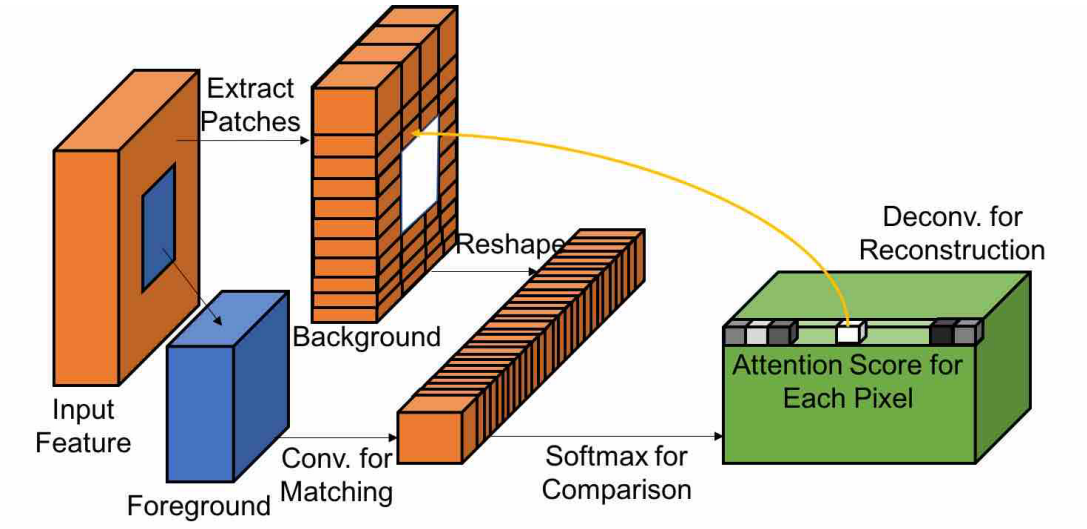

Figure 5: Illustration of the contextual attention layer from [2]. Each foreground patch is convoluted to each background patch to compute a similarity score, which is converted to an attention score after the application of softmax function, and refined with directional propagation. Finally deconvolutions with background patches are used to reconstruct foreground patches from the scores.

However, to encourage coherency of attention, a last step is performed in the computation of attention scores. The idea is that a shift in foreground patch is likely corresponding to an equal

shift in background patch for attention. To model and encourage this coherency, the authors do a left-right propagation followed by a top-down propagation with a kernel size of k. As an example, the left-right propagation is defined as

$$\hat{s}_{x,y,x',y'} = \sum_{i \in \{-k,\dots,k\}} s^*_{x+i,y,x'+i,y'}$$

To implement efficiently this propagation, a convolution with identity matrix as kernels is used. According to the authors, attention propagation significantly improves inpainting results in testing and enriches gradients in training.

Remark: The attention layer asks for a lot of convolutions ad computations, and can create a problem in memory efficiency. As the authors suggest it, assuming that a 64×64 region is missing in a 128×128 feature map, then the number of convolutional filters extracted from backgrounds is $128^2 - 64^2 = 12,288$. This may, of course, cause memory overhead for GPUs. To overcome this issue, they introduce two options:

- extracting background patches with strides to reduce the number of filters.
- downscaling resolution of foreground inputs before convolution and upscaling attention map after propagation.

1.5 Training algorithm

Now that we know all the specificities of the model proposed in the paper, we can conclude this part by presenting the training algorithm.

Algorithm 1 Training of our proposed framework.

while G has not converged **do**
  for $i = 1, ..., 5$ **do**
    Sample batch images $\mathbf{x}$ from training data;
    Generate random masks $\mathbf{m}$ for $\mathbf{x}$;
    Construct inputs $\mathbf{z} \leftarrow \mathbf{x} \odot \mathbf{m}$;
    Get predictions $\tilde{\mathbf{x}} \leftarrow \mathbf{z} + G(\mathbf{z}, \mathbf{m}) \odot (\mathbf{1} - \mathbf{m})$;
    Sample $t \sim U[0, 1]$ and $\hat{\mathbf{x}} \leftarrow (1 - t)\mathbf{x} + t\tilde{\mathbf{x}}$;
    Update two critics with $\mathbf{x}$, $\tilde{\mathbf{x}}$ and $\hat{\mathbf{x}}$;
  end for
  Sample batch images $\mathbf{x}$ from training data;
  Generate random masks $\mathbf{m}$ for $\mathbf{x}$;
  Update inpainting network G with spatial discounted ℓ_1 loss and two adversarial critic losses;
end while

Given a raw image x, a binary image mask m is sampled at a random location. Input image z is corrupted from the raw image as $z = x \odot m$. Inpainting network G takes concatenation of z and m as input, and output predicted image $x' = G(z; m)$ with the same size as input. Pasting the masked region of x' to input image, we obtain the inpainting output $\tilde{x} = z + x' \odot (1 - m)$. Image values of input and output are linearly scaled to $[-1, 1]$ in all experiments.

2 Experiments

To demonstrate the efficiency of the approach proposed in the paper, we begin with a quick reminder of the benchmark performed by the authors and then we display some examples of inpainting from both the paper and our own experiments.

2.1 Benchmark of inpainting methods

The authors evaluate their inpainting model on four different datasets that include Places2 [11], CelebA faces [12], CelebA-HQ faces[13], DTD textures [14] and ImageNet [15].

Qualitative benchmark

First the authors compare the output results of their baseline model (the two-stage model without contextual attention network) with results copied from the paper of Iizuka *et al.* [1] to ensure that their model has comparable performance. This is an important observation since no post-processing step is performed for their baseline model (on the contrary, image blending is applied in result of [1]) and their full model has a total of 2.9M parameters, which represents roughly half of the model proposed in [1].

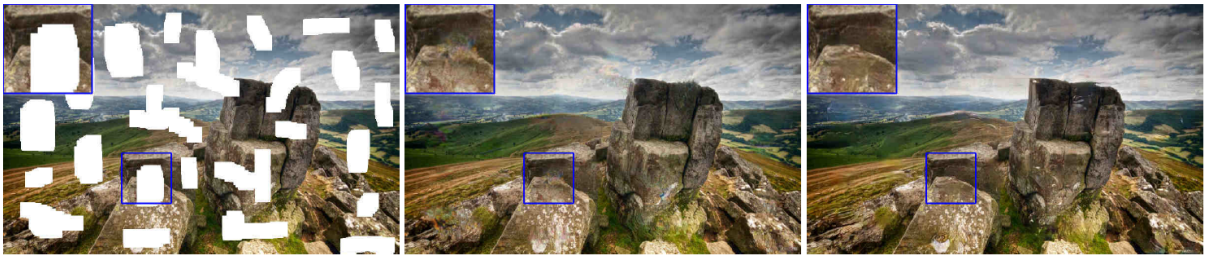

Figure 6: Comparison of the baseline model with Iizuka *et al.* [1]. From left to right, the input image, the result copied from [1], and result of the baseline model. Best viewed with zoom-in.

After confirming that the baseline model is at least as effective as the model of Iizuka *et al.*, the authors evaluate their full model (with the contextual attention network) to the simpler baseline model they created inspired by [1]. For the training they used images of resolution 256×256 with largest hole soze 128×128 from the dataset Places2 [11]. Then the models are tested on 512×680 images.

Figure 7: Qualitative results and comparisons to the baseline model. From left to right: the original image, input image, result of the baseline model, result and attention map (upscaled $\times 4$) of the full model. Best viewed with zoom-in.

The full model appears to leverage the surrounding textures and structures and consequently generates more realistic results, containing less artifacts than the results provided by the baseline model. As the authors suggest, the attention maps seem to confirm that the model is aware of contextual image structures and therefore can integrate information from surroundings in the generation of content.

Quantitative benchmark

An issue raised by the authors is the lack of a satisfying quantitative tool to compare inpainting methods. For instance, evaluating the performances with reconstruction errors can be criticized since many solutions different from the original image can be a satisfying solution when it comes to inpainting (actually sometimes inpainting is precisely used to generate content different from the reality -i.e. the original image-, for example to remove elements from a picture). Nevertheless, the authors report an evaluation in terms of reconstruction errors for different losses (mean l_1 error, mean l_2 error, peak signal-to-noise ratio or PSNR, and total variation or TV) on a validation set extracted from Places2. The methods used for the benchmark are first a methods directly copying raw image patches [3], then the baseline model without contextual attention, and finally the full model that incorporates contextual attention layers.

Method	l_1 loss	l_2 loss	PSNR	TV loss
PatchMatch	16.1%	3.9%	16.62	**25.0%**
Baseline model	9.4%	2.4%	18.15	25.7%
Full model	**8.6%**	**2.1%**	**18.91**	25.3%

Table 1: Result of the validation test on a Places2 subset.

Overall, the full model proposed in the paper outperforms other methods.

2.2 Our own experiments

To verify the efficiency of the model for image inpainting, and understand how it behaves relatively to several factors, we conducted several experiments on a custom dataset. To be more precise, we studied the influence of the size of the missing region, of its texture, and finally we studied how the model behaves when the image contains, in the background, regions similar to the missing one.

2.3 Experimental set-up

In order to be precisely in line with the author's model, and to avoid the prohibiting training time of the network, we re-used the implementation from [2] (`https://github.com/JiahuiYu/generative\_inpainting`) and the model trained on the Places2 dataset (Places2) was deemed closer to our test images than the CelebA dataset). Consequently, the network used and all parameters are precisely identical to the original implementation.The images used are landscaped and monuments images. Similarly to the images given as exemples in the implementation, all images are resized to size 512×680 before using the network. In the absence of proper quantitative metrics to evaluate the inpainting, we only evaluate the results with our own perception, in a qualitative way. More than 400 examples were processed, and the results below are representative of the general trends observed.
The script used to generate the experiences can be found at `https://github.com/LouisDumont/generative\_inpainting`.

First, we tested the influence of the size of the missing zone on the quality of the result. To this end, each original image was used to generate several inputs, with a square missing region of increasing size (the location of the region remains identical whatever the size).

We also compared inpainted images where the missing region include different textures of different complexities, and visualize the results.

In addition to this, we wished to evaluate to what extend the model could effectively borrow patches from the background. To this end, we generated a specific set of images, that we call replicated images. Each one is composed of the concatenation of twice the same window along the x axis. This ensures that the missing zone can precisely be found in the background the network has at its disposal. In such a situation, we expect the results to be much closer to the

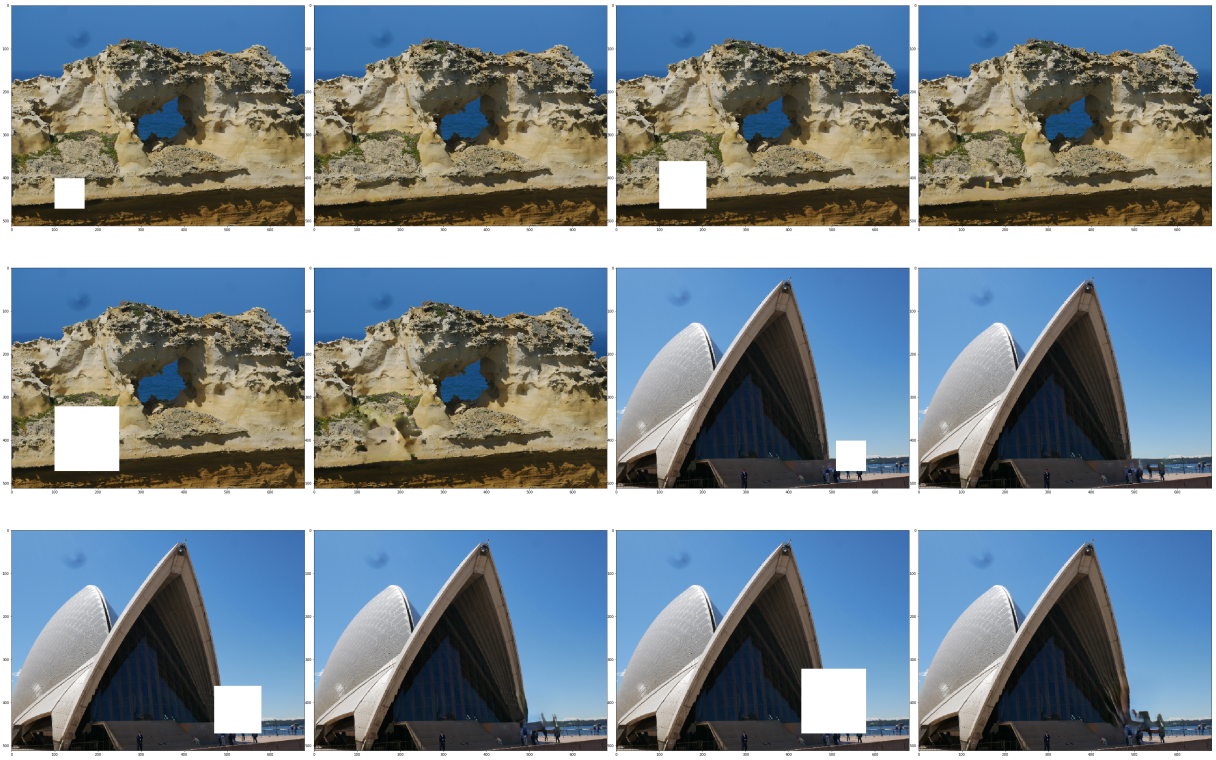

Figure 8: Examples obtained when varying the masked regions' size. For each pair of images, the input of the model is presented on the left, and its output on the right

original images; that is why we compare those results with the first ones obtained on full original images.

2.4 Results and Analysis

Figure 8 illustrates the results observed when changing the size of the masked region in an image. When the region is surrounded by a uniform texture, changing its size does not seem to influence the visual quality of the output. In contrast to this, for region in zones of high contrast and without uniform texture, extending the maske size leads to poorer results, especially far from known pixel. This is coherent with intuition, as the model cannot rely on known pixels nor on similar patches.

The influence of texture on the output is shown by figure 9. We found that masked regions in uniformly textured zones (the sky, sea or rocks) tends to be adequately inpainted, whereas the lack of uniform textures (for example when at the intersection of several objects, without simple shapes) systematically leads to poorer results.

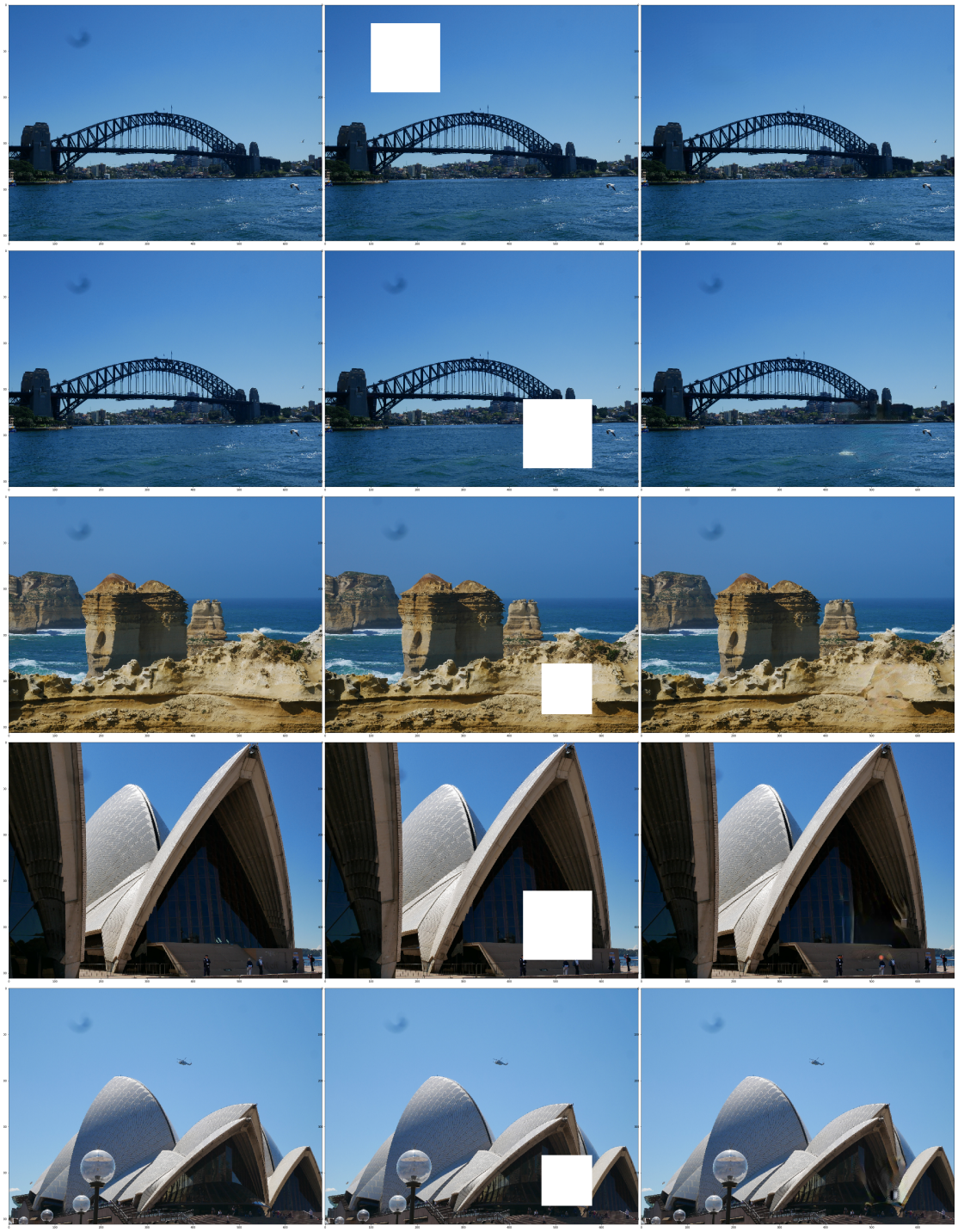

Figure 9: Example of results with different textures surrounding the masked zones. Left: original image, Centre: input of the model, Right: Model output

Lastly, we studied to which point having background patches exactly similar to the masked region improved the results. Some representative outputs of those tests can be found in figure 10. Contrary to our first intuition, we found that having such guidance does not improve the visual quality of the results, which are similar whether or not an exact copy of the masked zone can be found in the background. This shows the limits of the contextual attention: having a single example of the patch in the background does not concentrate enough the attention to lead to a replication of this patch in the image.

Note: In addition to the results of our experiments, we noticed that, in the most difficult zones (with the highest irregularities), some artifacts sometimes appear in the inpainting. This is presented in figure 11. Those artifacts are always identical, which leads us to believe that they may be linked to an issue when training the model. This does not seem specific to a region size or location, but seems to appear mostly when the inpainting is intuitively "difficult" (far from know pixels, in highly irregular regions not similar to the background). A reasonable hypothesis is thus that it appears when the network "hallucinates" the most (when it cannot resort to known patches or close pixels), and would thus denote a strong bias in the hallucinated patches.

Conclusion

In conclusion, Yu *et al.* developed a model that outperforms state-of-the-art methods for Inpainting in images. Inspired by the work of Iizuka *et al.* [1], they propose a model that not only achieve better results without any post-processing step, but also contains far less parameters and can be trained faster. As an example, their baseline generative network reaches convergence much faster than the one used in [1]: the training time was reduced from 11,520 GPU-hours (K80) to 120 GPU-hours (GTX 1080) which corresponds to almost 100× speedup.
Moreover, the authors managed to design a contextual attention layer that is able to borrow information from distant location in an image to infer values of missing pixels, improving significantly image inpainting results.

Finally, the perspective they suggest to bring their model further is to extend the method to very high resolution inpainting, by taking inspiration from the current and impressive growing of GANs [13].

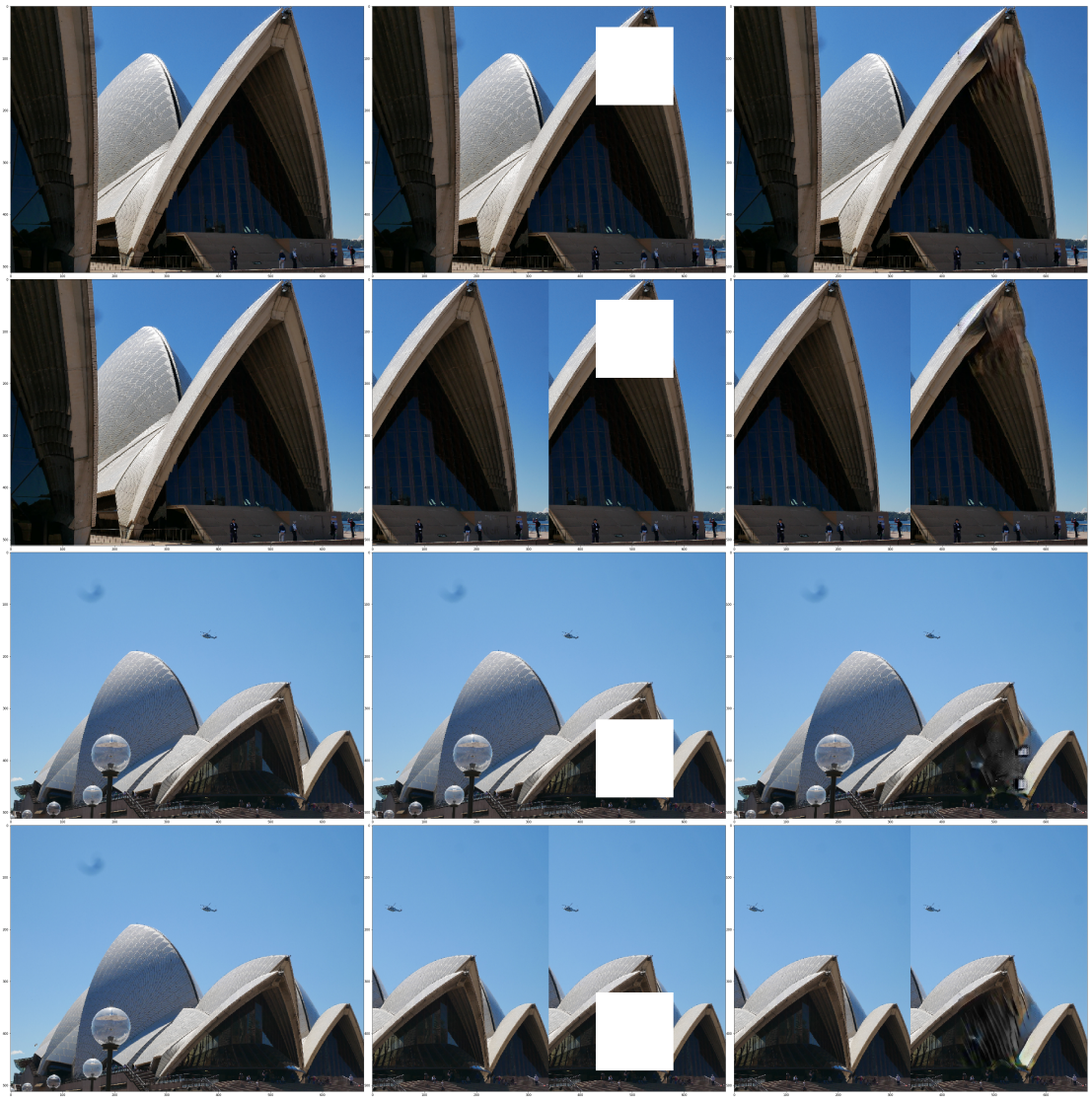

Figure 10: Comparison of results with and without replication. Left: original image, Center: input of the model, Right: model output. For the second and forth example, all patches needed to reconstruct the masked region are present, however the output of the model is not significantly improved.

Figure 11: Examples of artifacts observed in the results. Left: original image, Center: input of the model, Right: model output. By observing closely, one can see that the center of the artifacts is always the same.

References

[1] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Globally and locally consistent image completion. *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, 36(4):107:1–107:14, 2017.

[2] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S. Huang. Generative image inpainting with contextual attention. *CoRR*, abs/1801.07892, 2018.

[3] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. PatchMatch: A randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 28(3), August 2009.

[4] Soumith Chintala Alec Radford, Luke Metz. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv:1511.06434*, 2016.

[5] Deepak Pathak, Philipp Krähenbühl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. Context encoders: Feature learning by inpainting. *CoRR*, abs/1604.07379, 2016.

[6] Léon Bottou Martin Arjovsky, Soumith Chintala. Wasserstein gan. *arXiv:1701.07875*, 2017.

[7] Ishaan Gulrajani, Faruk Ahmed, Martín Arjovsky, Vincent Dumoulin, and Aaron C. Courville. Improved training of wasserstein gans. *CoRR*, abs/1704.00028, 2017.

[8] James Hays and Alexei A Efros. Scene completion using millions of photographs. *ACM Transactions on Graphics (SIGGRAPH 2007)*, 26(3), 2007.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[10] Thomas Unterthiner Sepp Hochreiter Djork-Arne Clevert. Fast and accurate deeep network learning by exponential linear units (elus). *arXiv:1511.07289v5 [cs.LG] 22 Feb 2016*, 2016.

[11] Bolei Zhou, Àgata 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*, PP:1–1, 07 2017.

[12] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. *CoRR*, abs/1411.7766, 2014.

[13] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *CoRR*, abs/1710.10196, 2017.

[14] M. Cimpoiand S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi. Proceedings of ieee computer society conference on computer vision and pattern recognition. pages i–, June 1997.

[15] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Fei-Fei Li. Imagenet large scale visual recognition challenge. *CoRR*, abs/1409.0575, 2014.

[16] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Free-form image inpainting with gated convolution. *arXiv preprint arXiv:1806.03589*, 2018.

[17] C Villani. *Optimal transport – Old and new*, volume 338, pages xxii+973. 01 2008.