# Image Inpainting with GANs

Adam Austerberry
University of Michigan
adamau@umich.edu

Andriy Massimilla
University of Michigan
andriym@umich.edu

## Abstract

*Image inpainting is an arguably fundamental computer vision task which can demonstrate a model's understanding of the scene and a its ability to predict it. This paper attempts an implementation of Context Encoder, an image inpainting approach, and compares our results to the results achieved in the original Context Encoder paper.*

## 1. Introduction

Image inpainting is the task of filling in a part or parts of an image using the surrounding context to create a believable result. It has applications in data loss recovery, image editing, and more. We chose this for our project because the task of image inpainting is fundamentally challenging in nature. In this paper, we attempt to use GAN image inpainting techniques to fill in a 64x64 black square in the center of a 128x128 image. Predicting large amounts of data from a limited context, such as with our problem, is hard – but GAN approaches do a lot better and give some promising results.

Our paper focuses on our implementation of the Context Encoder approach, compares our results to the results achieved in the original Context Encoder paper, and experiments with different model parameters and architecture changes.

## 2. Related Work

Image inpainting has a somewhat deeper history, starting with attempts in the late 1990s and continuing today. Early image inpainting attempts can be divided into categories, including texture synthesis, PDE, and semiautomatic approaches. The GAN approach is fundamentally different than these methods, but they paved the way for GANs and had surprisingly decent results for the time.

### 2.1. Texture synthesis

Texture synthesis approaches, such as the one proposed by Efros and Leung [2], involve marching pixel by pixel across occluded image regions, searching for nearby blocks for suitable similarities. Some texture synthesis approaches involved marching block by block instead, improving efficiency but sacrificing some quality.

### 2.2. PDE

PDE methods involve utilising isophote lines (curves on a surface connecting points of equal brightness) to preserve structural features. This results in decent-quality results for images with narrow occlusions, but has issues with larger occlusions as it does not generate textures well.

### 2.3. Semiautomatic

Semiautomatic approaches rely partially on user input to sketch contours of occluded image regions, and then apply texture synthesis approaches using these contours to generate more accurate results. This approach results in much better quality as compared to texture synthesis methods, but has drastically reduced efficiency since a human must be involved in the inpainting process.

## 3. Method

Context Encoder [3] uses a completely different approach as compared to previous image inpainting techniques. The paper combines an generator, which generates an unoccluded image given an occluded image, with a discriminator, which looks only at the occluded region and determines if it's fake or real. The generator and discriminator work against each other to continuously improve each other through feedback. This adversarial process (GAN) produces a generator which can generate believable images. See Figure 1 for architecture specifics.

The generator takes the form of an autoencoder with a series of convolutional layers. Context Encoder improves on the autoencoder format by adding a fully connected linear layer in the center, increasing the level of context that can be encoded into the surroundings (hence the name Context Encoder). It is fed in a 128x128 image with a 64x64 occluded region (black pixels) in the center, and outputs a 64x64 image which represents the generated occluded region.

Unlike the in the original Context Encoder paper [3], we decided to use two discriminators, rather than just one. The local discriminator looks only at the output of the generator. It's job was to determine whether or not the inpainted image looked realistic. The global discriminator gets to look at the entire image after being inpainted with the result of the generator. It's job is to determine if the inpainting matches well with the rest of the image. Out hypothesis was that having multiple discriminators would allow us to train more precisely. These discriminators are series of convolutional layers that reduces to a 0 to 1 floating point output which describes whether it "believes" the image is an actual photograph (1) or the result of the generator (0).

See Figure 3 for a diagram of these models.

## 4. Experiments

We decided to use a smaller, more manageable dataset as compared to the one used in the Context Encoder paper. The dataset we selected, STL-10 [1], has a much more reasonable 96x96 image size, with a large number of unlabeled samples. We chose this because Google Colab has usage limits which make it difficult to use anything with a larger size. In order to make this dataset work with the proposed architecture, we scaled the images to 128x128. Some results achieved in the paper can be seen below in Figure 2.

The hyperparameters that we chose were mostly based off of what we had used for problem set 6, since that assignment was also about GANs. We used Adam optimizers with learning rates of 0.0002 as well as beta values 0.5 and 0.999.

Although our original plan was to train our model as a GAN, we actually found that that the generator became more accurate when it was trained using L2 loss against the ground-truth training data rather than the discriminators (adversarial loss). We did also attempt a hybrid between L2 loss and adversarial loss, but this produced an overall worse result. We had hoped that tweaking the hyperparameters would be able to solve this problem, but nothing ended up working. We suspect this is because there is a tensor transformation error somewhere in our code causing the artifacting seen in Figure 5.

We also attempted adding a global discriminator which analyses the entire image to ensure the generated inpaint is more believable when inserted back into the occluded image. We factored into the loss, weighting it equally with the local discriminator. The results however did not improve.

## 5. Conclusions

Context Encoder was a breakthrough for image inpainting – papers proceeding it use the same overall structure with small to large modifications. Although we weren't able to outperform the paper's results without major changes, playing around with Context Encoder's parameters was really insightful into how this GAN implementation works. In the future, we'd have liked to see our implementation with the discriminators properly integrated to better train the generator and get much more realistic results.

## 6. Figures

Figure 1. Context Encoder architecture

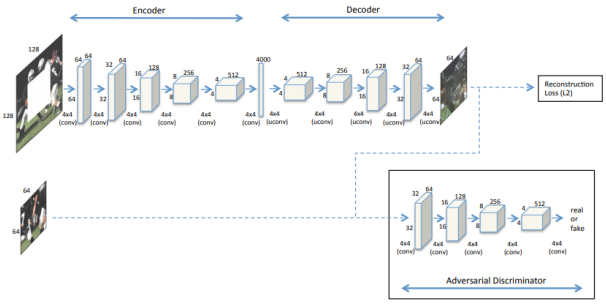



Figure 2. Context Encoder baseline results

(a) Input context (b) Human artist

(c) Context Encoder ($L2$ loss) (d) Context Encoder ($L2$ + Adversarial loss)

Figure 3. Our architecture

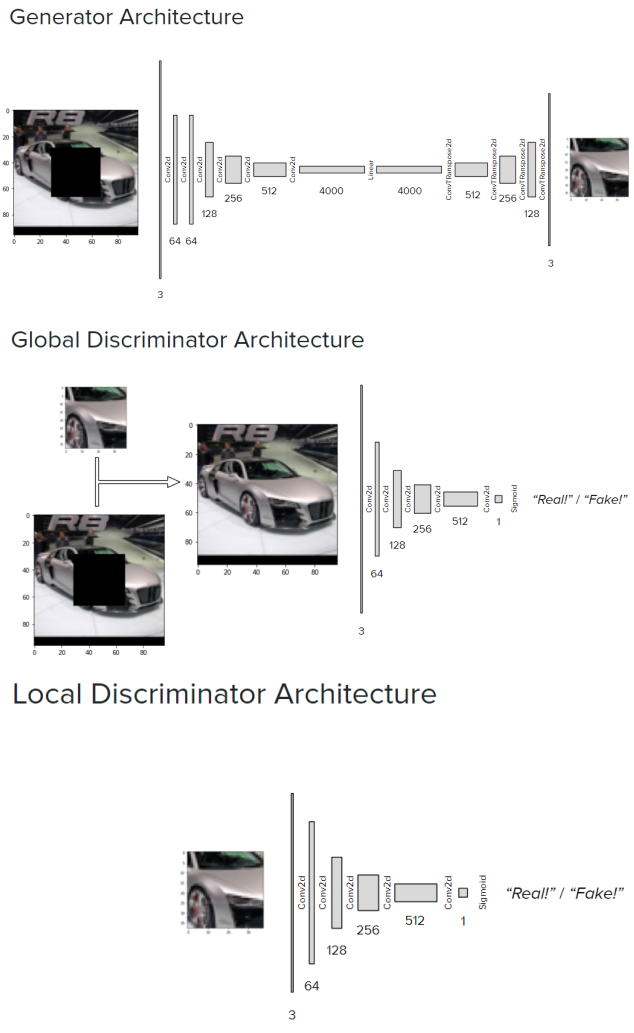
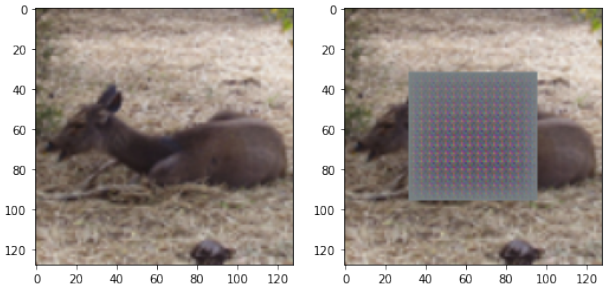



Figure 4. Our results (L2 loss only)

Figure 5. Our results (L2 + Adv loss)



## References


[1] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In Geoffrey Gordon, David Dunson, and Miroslav Dudík, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 215–223, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR.

[2] Alexei A. Efros and Thomas K. Leung. Texture synthesis by non-parametric sampling. *IEEE International Conference on Computer Vision*, 1999.

[3] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. Context encoders: Feature learning by inpainting, 2016.