Applications of Generative Adversarial Networks in Single Image Datasets

Dylan E. Cramer
crame160@morris.umn.edu
Division of Science and Mathematics
University of Minnesota, Morris
Morris, Minnesota, USA, 56267

Abstract

One of the main difficulties faced in most generative machine learning models is how much data is required to train it, especially when collecting a large dataset is not feasible. Recently there have been breakthroughs in tackling this issue in SinGAN, with its researchers being able to train a Generative Adversarial Network (GAN) on just a single image with a model that can perform many novel tasks, such as image harmonization. ConSinGAN is a model that builds upon this work by concurrently training several stages in a sequential multi-stage manner while retaining the ability to perform those novel tasks.

Keywords: datasets, neural networks, generative adversarial networks, convolutional, unsupervised learning, machine learning

1 Introduction

Envision a world where you could naturally mold together any two images you'd like or completely re-imagine them at the touch of a button, all the while maintaining the qualities that make them good. This task is just a small subset in a diverse field of models dedicated to solving the generative problem. *Generative Adversarial Networks* (GANs) are a kind of artificial intelligence algorithm, one at the forefront of machine learning models dedicated to solving that problem.

The goal of this kind of generative model is to study a collection of training examples, learn the probability distribution that best approximates generating them [1], and ultimately generate images that are indistinguishable from the real images. Most GAN models take tens of thousands of carefully selected images and an expensive amount of computing power in order to train. Though the basic idea of the model is to generate new examples, they can perform many tasks, some of which we will discuss later in this paper.

In this paper we take a look at ConSinGAN, a model created by a group of researchers at the University of Hamburg, Germany [4], that takes on the challenging task of training a GAN on just a single image. Since the dataset is limited to that single image, ConSinGAN is an example of an *unconditional image generator*, as the alternative refers to conditionally generating from a selection of a dataset that is labelled, and ConSinGAN has neither multiple images to choose from nor labels. However, that is far from the only task it can perform. One task in particular is showcased by the authors: image harmonization. *Image harmonization* learns how to composite the different objects and color distributions of another image within the context of the original image.

The most important thing to highlight is how those capabilities contrast in comparison to what its predecessor, SinGAN, offers. Though SinGAN can also be tweaked to perform the same functions, ConSinGAN does it while being a smaller model and taking much less time to train. Users also expressed a significant preference for the new model [2].

Before we can get to exploring the intricacies of ConSinGAN and how it builds upon its predecessor in a way that resulted in user preference, we must begin by covering the basics of machine learning, neural networks, and other concepts that are key to understanding this model in Section 2. From there, we move onto the methods and results of ConSinGAN in Section 3 and 4, as well as the breakthroughs the researchers made refining the groundwork SinGAN laid. Finally, we end with our conclusion in Section 5.

2 Background

2.1 Machine Learning

Machine learning (ML) is a part of the field of artificial intelligence (AI) dedicated to developing algorithms to construct a *model* that makes decisions or predictions without being explicitly programmed to do so, with each model exploring some way of computing the output based on the inputs. In order to train the model, it requires a set of sample data, which is typically known as its training data or dataset. A model has many *parameters* whose values start randomly and are adjusted based on the training data to represent statistical patterns in the data to predict the output value, such as what class a data point belongs to. The training process also has *hyperparameters* such as the learning rate, which differ from parameters in that they do not depend on the training data, whereas the value of other all the other parameters are derived from the training.

One of the most important concepts in training a model is the method of learning that the model is going to use. Though there are many different paradigms, the core two to understand are *supervised* and *unsupervised learning*. Supervised learning requires a set of data that has both the

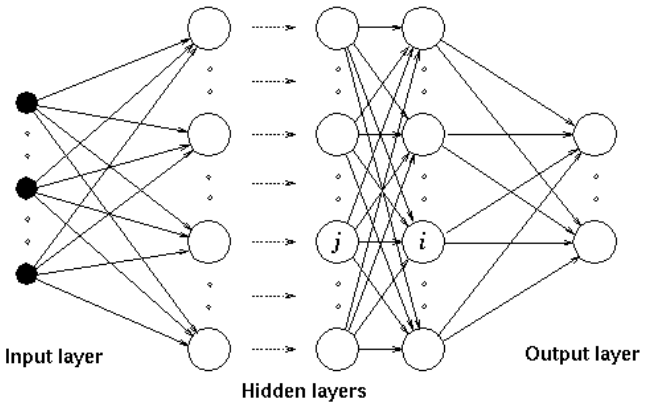

Figure 1. Example of a neural network.

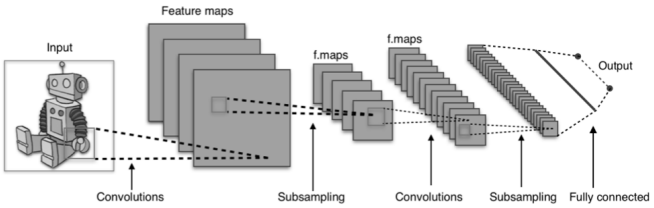

Figure 2. Example of a convolutional neural network.

inputs and outputs paired, and the ultimate goal is to have a model that can predict the associated outputs with given inputs accurately. An example of a problem that it would typically be used for is classification, such as identifying and labeling objects in an input image. Labeling is a very tedious process with tens of thousands of images, since any given dataset has to be manually tagged by humans before it can be used to train a model. In contrast there is unsupervised learning, which only takes the input (unlabelled) with the goal of learning patterns or properties of the data, such as its probability distribution.

2.2 Convolutional Neural Networks

The anatomy of an *artificial neural network* (ANN), which is a type of ML model, consists of an input layer, any number of hidden layers, and an output layer (see Figure 1). These layers contain a number of nodes, also referred to as neurons. Between the layers these nodes are connected, with every connection being defined by a weight that is initialized randomly within a given range. This framework is inspired by how the human brain works, and is designed to recognize patterns in data without being programmed. Throughout the training process the weights of the edges between the nodes are influenced by backpropagation of the difference between the output of the model and the dataset. The *learning rate* hyperparameter determines how big the weight adjustments are in that backpropagation process.

The input nodes themselves do no calculations and merely feed the external data forward to whatever nodes they are connected to. The hidden layers are where the learning happens, the nodes contained within them take in the previous layer's inputs after they have been multiplied by the weights of the edges that connect them. The higher the weight, the more influence the node has on a subsequent node. Each node produces a single output which can be sent to multiple other neurons, and that output is defined by the sum of the aforementioned weighted nodes of all the inputs then fed into an activation function. An activation function is typically a non-linear function that determines how much a given node will influence the nodes in the next layer it's connected to. The non-linearity of the activation function allows the network to model more complex structures. The output nodes themselves define that structure, with each corresponding to a label or simply a probability.

Convolutional Neural Networks (CNNs) are a specialized type of ANN that provide especially good performance in image classification and processing, as they are specifically designed to process pixel data. They are inspired by our understanding how a neuron in the visual cortex responds to a specific stimulus. Unlike many other NNs that treat layers of nodes as vectors, CNNs maintain the data as a 2D matrix through most of the layers, starting with the pixels of the original image and slowly reducing the matrix size for analysis. CNNs are composed of three main types of hidden layers: the convolutional layers, the pooling layers, and the fully connected layers (see Figure 2). The convolutional layer is the first hidden layer of a typical CNN, and can be followed up with either a pooling layer or more convolutional layers, before the network concludes with the fully connected layers. With each intermixed convolutional layer and pooling layer (typically they are paired one after another), the CNN analyzes greater portions of the image, referred to as feature extraction. The core idea behind this is that earlier layers identify simple details like colors and edges, whereas when the CNN progresses deeper, it starts to identify larger structures such as shapes or entire objects [3].

As we previously mentioned, the purpose of the convolutional layer is feature extraction. It performs this by taking a filter, which is essentially a matrix of weights that is typically a size of 3x3, and slides it along the input image (matrix), combining it with a portion of the image with every step of the convolution to produce a real number in the output. This output is then put into another output matrix that's called a feature map, and the process is repeated until the filter has swept across the entire image [3].

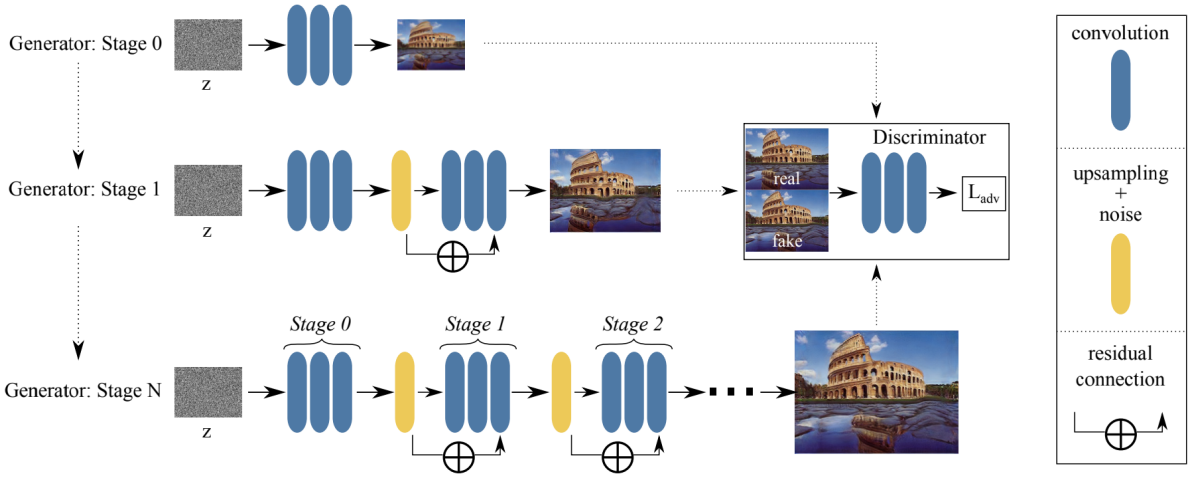

Figure 3. Overview of ConSinGAN. They start training at 'Stage 0' with a small generator and small image resolution. With increasing number of stages both the generator capacity and image resolution increase [2].

Next is pooling layers, which conduct dimensionality reduction by reducing the size of the input. The manner in which it does this is reminiscent of convolutional layers, in that the pooling process sweeps a filter across the entire input, but the difference is that this filter does not have any weights. Instead of applying weights, the filter has a function that performs a specific pooling operation on the input matrix. This is typically either max pooling or average pooling. As the name implies in both cases, max pooling takes the max value within the filter area and maps it to the output pooled feature map, while average pooling takes the average of all the values.

Lastly is the fully connected layers. In these layers, each node in the output layer connects directly to every node in the previous layer, and classification is performed using an activation function that outputs the probability of belonging to a given class based on the features extracted in those previous layers. CNNs can have more than one fully connected hidden layer, and often do.

2.3 Generative Adversarial Networks

Generative adversarial networks are based on a game, in the sense of game theory, between two machine learning models, typically implemented using neural networks [1]. One network is called the generator and the other is the discriminator. The goal of the generator is to match its output distribution as closely as it can with the dataset, while the goal of the discriminator is to output 1 when the image is real, and 0 when it was produced by the generator.

The discriminator is essentially outputting the probability of whether it believes the input is real or fake, and the result is backpropagated throughout both of the networks using something called a *loss function* or *cost function*. The loss function essentially calculates the difference between the expected value of the network and the actual value of the network at the end of an iteration and that cost difference is backpropagated throughout the network via adjusting the weights of the connections between the nodes from the last layer to the first layer.

Though the specific loss functions themselves are outside of the scope of the paper, the essential thing to take away is that one network's gain is another's loss. For example, if the discriminator is given an image from the generator and it outputs 0.8, its loss is 0.8 since that's the difference between the real and actual value, whereas the generator is given a loss of 0.2. The core idea of a GAN is based on the indirect training through the discriminator, while both are being updated dynamically. The generator is not trained to minimize the distance to a specific image, but rather to fool the discriminator. This is similar to mimicry in evolutionary biology, with an evolutionary arms race between the two networks [5].

3 Image Generation

Now that we've covered the bare basics to understand ConSinGAN itself, let's move onto the methodology behind the model. We take a deep examination of its predecessor SinGAN, which laid much of the groundwork that ConSinGAN operates on. SinGAN's author's contextualize how their model being unconditional distinguishes it from other models:

> Here, we take the use of GANs into a new realm – unconditional generation learned from a single natural image. Specifically, we show that the internal statistics of [filters] within a single natural image typically carry enough information for learning a powerful generative model. SinGAN, our new single image generative model, allows us to deal with general natural images

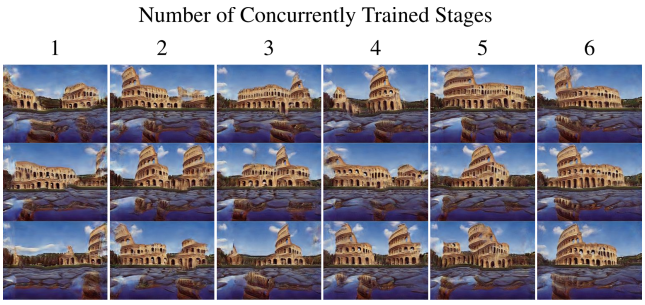

Figure 4. Effect of concurrently trained stages for a model with six stages [2].

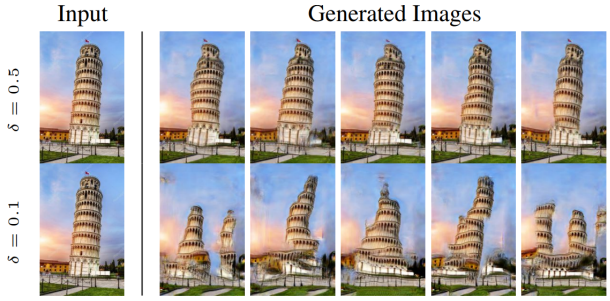

Figure 5. Effect of the learning rate scale δ during training of ConSinGAN [2].

Model	Confusion ↑	SIFID ↓	Train Time	# Stages	# Parameters
ConSinGAN	16.0% ± 1.4%	0.06 ± 0.03	24 min	5.9	~660k
SinGAN	17.0% ± 1.5%	0.09 ± 0.07	152 min	9.7	~1.34m

Table 1. Results of the user study and SIFID on images from the 'Places' dataset [2].

> that contain complex structures and textures, without the need to rely on the existence of a database of images from the same class. This is achieved by a pyramid of fully convolutional light-weight GANs, each is responsible for capturing the distribution of [filters] at a different scale. Once trained, SinGAN can produce diverse high quality image samples (of arbitrary dimensions), which semantically resemble the training image, yet contain new object configurations and structures [4].

The critical thing to take away from this excerpt is both the structure of SinGAN, its pyramid of convolutional GANs that learn from progressively higher resolutions of the image itself, and the underlying principle that the resulting model is *multi functional*. Other single image GANs have been explored only in the context of texture generation.

3.1 Methods

The structure of SinGAN is important because ConSinGAN takes the same approach of building a pyramid of lightweight convolutional GANs. However, one of the decisions that SinGAN makes is to freeze all the previously trained stages and only train the next layer of the pyramid. Each stage is its own lightweight GAN that is trained at progressively higher resolutions with each stage, with correspondingly smaller filter sizes that results in the model focusing on finer and finer details. The researchers of ConSinGAN found that the decision to train only one stage at a time limited the amount of learning that could be passed on from different stages, and that propagating the resulting images instead of the feature maps from the convolutions negatively affected the learning process. ConSinGAN takes a new approach of scaling up the feature maps from previous stages and connecting them to the convolutions of the stage next in line, which is termed 'residual connections' (see Figure 3).

As previously mentioned, one essential area ConSinGAN diverges is by training multiple stages in parallel instead of training the network end-to-end like SinGAN does. Since ConSinGAN trains several stages of the model concurrently for a single image, it's hence named 'Concurrent-Single-Image-GAN' (ConSinGAN) [2]. The researchers found that the concurrency not only improves the learning process and reduces training time, but also had the knock on effect of being able to trade image variance for image quality through the new hyperparameters introduced to balance this new methodology. One of the new hyperparameters is the number of stages that are to be trained in parallel. The more stages trained, the more consistent the global image is. However, training many stages concurrently limits the variation so drastically that it's classified as *overfitting* (see Figure 4), wherein the generator output only results in the training image itself (or something very close to). The healthy middle ground that the researchers found worked best was 3 stages being trained concurrently.

One other important area where ConSinGAN differs in methodology is in scaling the learning rate hyperparameter differently for each stage by introducing a new scaling hyperparameter, δ. The whole purpose of this hyperparameter is to emphasize the learning in the earlier stages of the model when the broad details are being learned and lower later on when the finer details are being learned. This allows for another method of being able to trade off image fidelity with image variance, as can be seen in Figure 5. For any given stage n, the learning rate is scaled by δ (which is less than 1) to the power of the stage number itself.

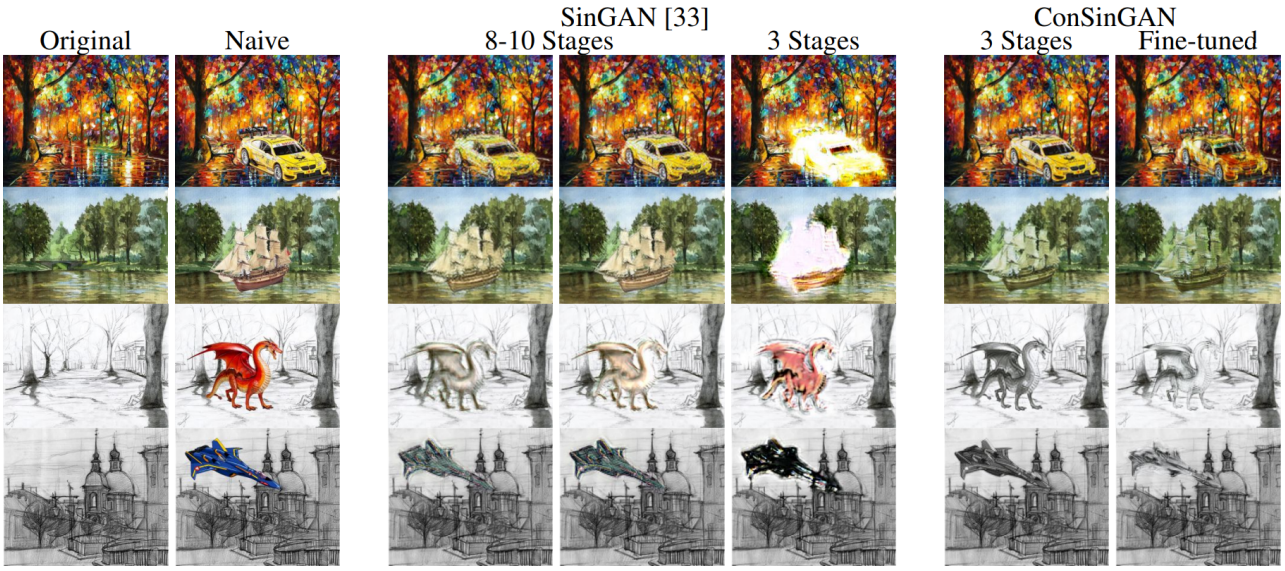

Figure 6. Image harmonization with SinGAN and ConSinGAN [2].

Model	Random ↑	Paired ↑	SIFID ↓	Train Time	# Stages	# Parameters
ConSinGAN	56.7% ± 1.9%	63.1% ± 1.8%	0.11 ± 0.06	20 min	5.9	~660K
SinGAN	43.3% ± 1.9%	36.9% ± 1.8%	0.23 ± 0.15	135 min	9.1	~1.0M

Table 2. Results of the user studies and SIFID on images from the LSUN dataset [2].

3.2 Results

One of the ways of quantitatively analyzing the results of both ConSinGAN and SinGAN researchers use is the Single Image Frechet Inception Distance (SIFID). The Frechet Inception Distance (FID) itself is the distribution of a pre-trained network's activations between the sets of real images and generated images, and is a test specific to GANs that has been shown to correlate with a higher quality model. The SIFID is an adaptation of the FID created by SinGAN's researchers, and it compares the distribution of features of the convolutional layer before the second pooling layer instead of the activation vector after the last pooling layer [2]. The lower the SIFID score, the better the model. This will be an essential metric in the two tests ConSinGAN's authors perform.

In the first of the two tests, the ConSinGAN researchers use both the same dataset (the 'Places' dataset, as shown in Figures 4 and 5) as SinGAN, and the same evaluation procedure that SinGAN does. The procedure entails showing the generated image and its respective training image for one second each to a user, and asking them to identify the real image. The *confusion* is the percentage of users that chose the model's image over the real one. Though ConSinGAN's confusion score is slightly worse than SinGAN's, its SIFID score is considerably better, and its training time is drastically reduced (See Table 1).

In the second test, we take a look at a much more challenging dataset, the LSUN dataset [6], which is a collection of one million images with 10 scene categories and 20 object categories. Because the images are more challenging, ConSinGAN's researchers made the decision to forgo comparing the generated images to real images as in the previous test, and instead they compare ConSinGAN's generated results directly to SinGAN's generated results. The reason the LSUN dataset's images are more challenging is because of the wide variance in global structure that each model needs to learn, as well as the existence of objects in the scene. In each of the tests, ConSinGAN's researchers generated 10 images per training image for a total of 500 generated images from both ConSinGAN's model and SinGAN's models respectively [2]. In both studies, images generated by the models are compared side by side, and unlike the previous test, there is no time limit for the user to decide. Users need to judge which of the two images are better, in contrast to asking which one is real for the first test. Both user studies use Amazon Mechanical Turk (AMT), with 50 participants comparing 60 pairs of images for each study.

For the first study, images are chosen randomly from each dataset, meaning that each image is more than likely generated from completely different classes (e.g. 'park' vs. 'hotel room'). This makes it more difficult to directly compare the performance of SinGAN and ConSinGAN, but provides useful insight on a given user's preferences. In the second study,

the generated sample from each of the models are from the same class. Table 2 shows how often users picked images generated by a given model for each of the two settings [2]. In this test we can see a stark preference for ConSinGAN in both of the studies, not to mention the significantly better SIFID score too. Overall, the studies suggest that ConSinGAN generates much more believable images, while also having a drastically reduced training time and model size in comparison to SinGAN.

4 Image Harmonization

Similarly to image generation, we first discuss the methods SinGAN uses to implement image harmonization to help characterize ConSinGAN's differences and then compare them to the ConSinGAN approach. In SinGAN, the naive image, meaning the image of an object simply cut and pasted onto the background image, is input into the model at test time. The model has to be fully trained on the background image before this can be done, and the input is thus interpolated to match the background features that the model has been trained to learn.

4.1 Methods

Because ConSinGAN can be fine tuned to train on any number of stages and at any learning rate, this is the exact sort of task where its versatility shines even more than before. Though the general model structure is the same as for unconditional image generation, the model is trained for exactly three stages per image, and trained for 1,000 iterations per stage. With every iteration, a random sample from the generator's output is chosen and different data augmentation techniques are performed to acquire a 'new' training image.

These data augmentation techniques entail randomly applying combinations of additive noise and color transformations to the original image. This augmented image is then input into the generator to continue the training until the next iteration. The intuition behind this process is that the generator is attempting to turn that augmented image into something that resembles the training image so it can trick the discriminator, which is basically what image harmonization itself is. An object itself is essentially color distributions and noise, and the generator needs to integrate it to match the features of its background image. [2].

The last method to discuss is the fine-tuning of the model, which refers to taking the model trained using the data augmentation techniques and inputting the target image just like SinGAN does for 500 additional iterations, instead of applying the data augmentation alone.

4.2 Results

Figure 6 shows the results of both of these methodologies, with the far left two columns displaying the background image and the object naively cut and pasted onto the background images respectively. In the middle is SinGAN and ConSinGAN is on the far right, both trained as described. Since SinGAN needs to be fully trained in order to perform image harmonization, it took the full 120 minutes to complete its training. In contrast, ConSinGAN took a mere 10 minutes to train for each image, though it does need an extra 2-3 minutes in the case of the model being fine-tuned [2].

Though there is no quantitative analysis concerning this function of either models, it's clear to see that ConSinGAN performs better than SinGAN (see Figure 6). Where SinGAN never quite takes on the same color of the background no matter how much training it gets, ConSinGAN's examples are 'absorbed' into the image much more smoothly. This is especially evident in the images with black and white backgrounds, where ConSinGAN completely matches this style, but color still bleeds through with SinGAN. The last detail that's important to highlight is the amount of artifacting SinGAN's model results in, which is in stark contrast to how consistent ConSinGAN's original objects are in comparison.

5 Conclusion

In this paper we did a deep dive on ConSinGAN and how the authors took the groundwork of SinGAN and synthesized with the best practices they discovered for training single-image GANs. Using their method of concurrently training stages of the GAN pyramid versus freezing the model with every stage, they found through their tests that users significantly preferred their results over SinGAN's through both random and paired comparisons with the LSUN dataset. Their model takes much less time to train, is significantly smaller, and consistently has a better SIFID score than SinGAN in both of the datasets.

The model still has its shortcomings. The user study in comparison with SinGAN on the 'Places' dataset isn't significant, and the LSUN dataset being used in the second tests could be a source of inconsistency because they also changed the user study too (real vs better). Future studies could shore up these inconsistencies by more stringent testing and answer the resulting big questions. Specifically, ConSinGAN's researchers could perform the same user study as the first test with realness as the metric while using the LSUN dataset for both of the models.

Acknowledgments

I would like to thank Elena Machkasova, my adviser for senior seminar, for the invaluable amount of feedback and all the help throughout the process of writing this paper. I would also like to thank Kristin Lamberty, my professor, for all of her essential feedback and help. Lastly, I would like to thank my alumni reviewer Joe Walbran for his thorough feedback.

References

[1] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2020. Generative Adversarial Networks. *Commun. ACM* 63, 11 (oct 2020), 139–144. https://doi.org/10.1145/3422622

[2] Tobias Hinz, Matthew Fisher, Oliver Wang, and Stefan Wermter. 2020. Improved Techniques for Training Single-Image GANs. https://doi.org/10.48550/ARXIV.2003.11512

[3] IBM. [n. d.]. What are convolutional neural networks? https://www.ibm.com/cloud/learn/convolutional-neural-networks

[4] Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. 2019. SinGAN: Learning a Generative Model From a Single Natural Image. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE Computer Society, Seoul, 4569–4579. https://doi.org/10.1109/ICCV.2019.00467

[5] Wikipedia. 2022. Convolutional neural network — Wikipedia, The Free Encyclopedia. http://en.wikipedia.org/w/index.php?title=Convolutional%20neural%20network&oldid=1116964560. [Online; accessed 20-October-2022].

[6] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. 2015. LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop. https://doi.org/10.48550/ARXIV.1506.03365