A Case for Conditioning: Conditioning Generative Models May Improve Image Quality

Justin Ho [*1]

Abstract

The Generative Adversarial Network (GAN) and the Variational Auto-Encoder (VAE) were among the first architectures to try to approach the problem of generative image modeling. A noise vector is passed into the generator and the decoder of each respective model and after an image is produced, sampled from the distribution that both models attempted to learn. In this paper, I propose that the simple act of conditioning, passing in an extra label to the generative part of both models during the training process, can lead to an overall improvement of image generation quality by VAEs. Conditioning resulted in a 20% improvement in FID score for VAEs that were conditioned over their un-conditioned counterparts. For IS, conditioning resulted in an improvement for VAEs as well. For GANs, conditioning resulted in a slight decrease in performance of 8% for IS and FID. Overall there is a case to be made for using the conditioned variants of generative models over their unconditional variants, at least in the case of VAE as the use of conditioning resulted in a massive increase in performance. For GANs more research must be done to determine the cause of this decrease.

1. Introduction

Generative AI has never been bigger. From text to images, recent research advancements in model architectures have greatly increased the ability and performance of machine learning algorithms on tasks that were once thought to be difficult (Rogers, 2021). For textual data, this recent innovation was spurned on by the transformer and the BERT architecture. For images, the diffusion model paved the way for modern advancements in the task of image generation. A landmark paper by Dhariwal and Nichol titled "Diffusion Models Beat GANs on Image Synthesis" described the architecture as being superior to Generative Adversarial Networks (GANs) in almost all major metrics (Dhariwal & Nichol, 2021). This new class of models created higher fidelity images, were much more stable during training, and completely outclassed its competitors in terms of diversity for the generated images.

However, despite the superiority of diffusion models for image synthesis, it is still important to review the predecessors to the model such as the GAN and the Variational Auto Encoder (VAE) to examine if there could be missed insight within previous model that could be applied to improve future models and their design.

During an analysis of prior literature on top-performing GANs and VAEs, a common trend that was noticed among all these architectures was that they all utilized the class labels of images during the training process (Brock et al., 2019). It is clear that conditioning may have a role to play in increasing the quality of images generated by these models. However, a gap that is not explicitly addressed in the literature is whether or not conditioning will consistently improve the image generation performance of a generative model.

I hypothesize that conditioning will improve any generative model's image quality in terms of metrics such as Inception Score and Frechet Inception Distance regardless of the architecture. I suspect that the addition of a label during the training process helps the generative model organize its learned distribution better than if there were no labels which may be what causes an improvement. In this paper, I will be comparing the base formulation of the Convolution GAN and VAE to the Conditional variants to reaffirm and explore the effects of conditioning on each model. I will also be comparing the quality of images generated for two different upscaling techniques: upscaling convolutions and transposed convolutions.

The paper will be organized as follows. In Section 2, we will briefly cover the formulations of the GAN and VAE architecture along with their conditional and convolution variants. In Section 3 we will cover the experiment design and the metrics we will be using. In Sections 4 and 5 we will cover the results and an analysis of them.

2. Background

In this section we will be covering the formulation of all these approaches from a high level view. If a more detailed description is needed we recommend the reader consider the original papers for each architecture.

2.1. Generative Adversarial Network

The task of learning the distribution of a collection of images is very complex. Attempts to directly solve this distribution have had some successes, but overall there have been mixed results and extensive limitations with this approach (Li et al., 2015). To get around the challenges of this, the GAN was created to approach the task of image generation by utilizing two networks that are competing against each other in a zero-sum game in order to learn the distribution of a set of images (Goodfellow et al., 2014). In the first network, the generator takes in a random noise vector z and produces a generated image. The goal of the generator is to try to fool the second network, the discriminator, which is passed in the real images used during training and the fake image created by the generator. As the discriminator discerns between which images are fake and which images are real, the error accumulated during this process is backpropagated to the discriminator, in order to improve its ability to detect real or fake images and then this error is also backpropagated to the generator in order to improve its ability to fool the discriminator. This process of pitting two networks against each other essentially allows the generator network to implicitly learn the latent distribution of images it is trying to reproduce.

2.2. Variational Autoencoder

The VAE utilizes a different approach to model the underlying probability distribution of the data. The VAE utilizes two networks as well, a decoder and an encoder but this time the networks work in tandem to learn the distribution of data. In the typical autoencoder structure, the encoder takes in an image and projects it into a lower dimension. The decoder takes this lower-dimension piece of data and attempts to reconstruct it and reproduce the image. The VAE modifies this approach a little bit by learning the probabilistic representation of the input data. The encoder in the VAE framework takes in the input image and represents the data as a mean and variance of a multivariate Gaussian distribution (Kingma & Welling, 2022). Afterward, using this mean and variance, a random noise vector is sampled from this multivariate Gaussian distribution and is passed to the decoder where the decoder attempts to reconstruct the image. Once training is completed, the decoder can be used to generate new samples from the learned distribution.

2.3. The Convolutional VAE and GAN

The original formulations of the GAN and the VAE did not use convolutional layers as a part of their architectures. They both utilized fully connected dense layers of neurons for both networks within each respective architecture (Goodfellow et al., 2014; Kingma & Welling, 2022). The downside of using exclusively dense layers is that dense layers are sub-optimal when it comes to processing image data as they are unable to capture spatial data which is especially important when it comes to vision tasks (Taye, 2023). By replacing the dense layers within both networks with their convolutional counterparts, a significant improvement gain was made in terms of convergence, image generation quality, and stability (Pu et al., 2016; Radford et al., 2016).

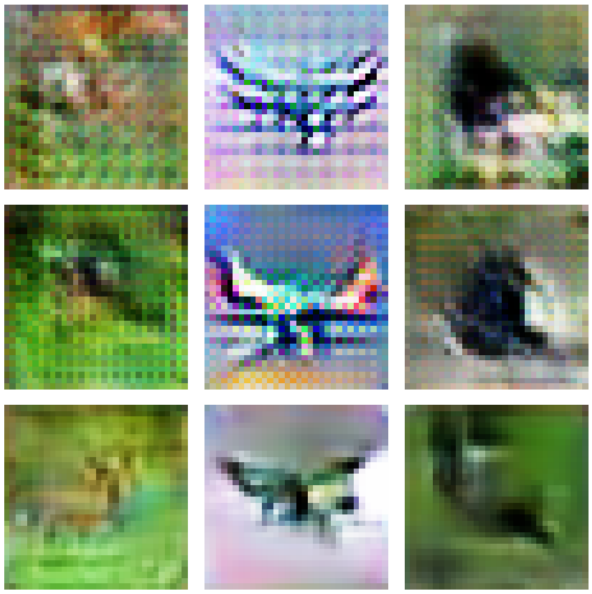

Figure 1. A display of the checkerboarding that can be apparent when using deconvolutions. The first row uses deconvolutions in the last two layers, the second row uses deconvolutions in the last layer, and the last row uses only upscale-convolutions. (Odena et al., 2016)

2.3.1. Upscaling Convolution

For the generative portion of both the convolutional VAE and GAN, they typically utilize the transposed convolution or deconvolution operation in the upscaling process. However, one downside that has been noted by the use of these operations is that they can sometimes lead to images that have a checkerboard pattern and excessive artifacts (Odena et al., 2016). This can be caused by deconvolutions layers with a kernel that is non-divisible by the stride. This causes overlaps to occur during the deconvolution process and results in certain pixels having their intensities essen-

tially excessively multiplied during the upscaling process. The solution proposed by the authors who discovered this phenomenon was to separate the upscaling process from the convolution process. Essentially, the ideal solution would be to upscale the image first, using some form of interpolation, and then performing a convolution. By doing this, you would entirely avoid any concerns with overlapping pixels during the upscaling process.

2.4. The Conditional VAE and GAN

A weakness of the original formulations of the GAN and VAE is that they both discard the class label of images they are training on (Goodfellow et al., 2014; Kingma & Welling, 2022). What this means is that after training, the user has no control over what kind of images are generated. When a random noise vector is passed into the decoder and the generator for the VAE and GAN respectively, a completely random image is created. This is especially problematic when working with datasets that consist of multiple classes of images. To get around this problem, instead of discarding the class labels of images during training, the class labels are utilized during different portions of the training process.

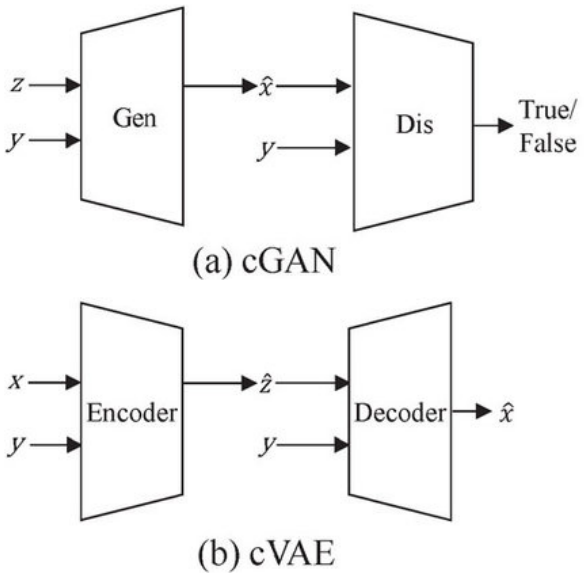

Figure 2. The architectures of the conditional GAN and the conditional VAE. Notice the conditional label y being passed into both networks (Xue et al., 2022).

For the conditional GAN, during training a random noise vector along with the labels of the real image would be passed into the generator. After this, the fake and real images and their associated labels would be passed into the discriminator where the typical backpropagation process would take place (Mirza & Osindero, 2014). This simple modification to the training process had a significant effect on how the conditional GAN organizes its learned distributions. The addition of a label allows the conditional GAN to learn the conditional distribution of the underlying dataset (Kwak & Zhang, 2016). Typically this "condition" came in the form of a label but it could hypothetically be any external piece of information.

For the conditional VAE, the training process was similar. During training, the image and its associated label are passed into both the encoder and the decoder (Sohn et al., 2015). Like the conditional GAN, this had the similar effect of allowing the model to learn the conditional probability of the image data and allow for some control over the decoding process.

3. Experiment Design

In this section, we will be discussing the overall design of the experiment and justifying the choices made during its creation. We will also provide some brief context on the metrics being used for this study.

3.1. Metrics

Before automatic metrics, generative techniques were primarily evaluated by human evaluators to see if the images generated were aligned with human perceptions of quality. This technique was simple but had the downside of being very expensive and also being unable to scale particularly well (Borji, 2018). To counter this, automatic metrics based on neural techniques were created. These techniques have been shown to correlate with human judgment, and are computationally inexpensive, allowing them to practically replace humans in the evaluation process (Alqahtani et al., 2019). The objective nature of automatic metrics also offers an attractive solution to counteract issues of potential bias that can occur in human annotators. These metrics are not without their flaw, but the seemingly deterministic and consistent nature of the metrics offers a compelling alternative compared to human annotaters. The two standard metrics that are used are Inception Score and Fréchet inception distance.

3.1.1. Inception Score

Inception Score (IS) is an automatic metric created in 2016 that aims to score the images produced by generative models based on two primary factors: the variety and the quality of the images (Salimans et al., 2016). Variety refers to the diverseness in the class of images generated and quality refers to alignment with human judgment (features such as clarity, discernability, etc). IS utilizes the pre-trained Inception classifier, to generate outputs that correspond to probability distributions of what that classifier thinks an image is. This process is repeated for every single class and summed

together to obtain a marginal distribution. This marginal distribution is utilized as a metric to judge the diversity of a generative model while the class distribution is used to judge image quality (Barratt & Sharma, 2018). To compare two distributions, KL divergence is used to compare the difference between the two distributions, and the more different a class distribution is from the marginal distribution, the higher of a score we should give the generative model. Lastly, we take the exponential of this value to make the metric more sensitive to improvements (Salimans et al., 2016).

3.1.2. Fréchet inception distance

Fréchet inception distance (FID) is another automatic Inception based metric introduced in 2017 that improves upon the weaknesses of IS and aims to provide a more comprehensive metric that is less susceptible to problems that plague IS such as no sensitivity to mode collapse, bias to high definition images, etc (Borji, 2018). FID works by first using the Inception network to project samples from real and generated images into an embedding. Next, the assumption is made that the distribution of this projection is of a multivariate Gaussian, and with that, the two embeddings are then compared using Fréchet distance for similarity (Heusel et al., 2017). If the generated embeddings and the real embeddings overlap, the FID score will be small, and vice versa.

3.2. Experiment Design

For this experiment, I will be working with the CIFAR-10 dataset. This dataset consists of 60,000 images of 10 different classes in a resolution of 32 x 32 with 3 color channels (Krizhevsky, 2012). This dataset was selected as it is quite typically used as a baseline for these kinds of studies (Odena et al., 2017; Gulrajani et al., 2017). As for the models I will be examining, I will be exploring 4 broad architectures: The Convolutional VAE, the convolutional GAN, the conditional convolutional VAE, and the conditional convolutional GAN. Within these models, I will also be exploring the effects of using upscaling convolutions over transposed convolutions in all 4 architectures. Overall I will be examining 8 different models with 4 being conditioned and 4 being unconditional.

In terms of the lower-level details of the model architectures, for the sake of an even comparison, all models across GANs and VAEs have similar weights and layer structures outside of the difference between the upscaling techniques. The details such as optimization parameters, batch size, etc, can be found in the appendix attached below. The experimentation steps will be as follows. First, all models will be trained on all instances of the CIFAR-10 dataset. Afterwards, a batch of 120 images will be generated by each model and each model will be evaluated using FID and IS.

4. Results

Find the results of the experiment below and generated images in the appendix.

Unconditioned Models		
Model	FID (Lower better)	IS (Higher)
Trans-con GAN	184.16078	4.224521
Upsc-con GAN	186.089	5.3463454
Trans-con VAE	324.03116	1.9790983
Upsc-con VAE	351.15283	1.4193777

Figure 3. Unconditioned model performance on FID and IS. Note that GANs performed significantly better than VAE accross the board. Also worth noting that models that utilized upscaling convolutions fared worse than models that use the traditional transposed convolutions.

Conditioned Models		
Model	FID (Lower better)	IS (Higher)
Trans-con GAN	200.14644	3.9170582
Upsc-con GAN	201.97002	3.741853
Trans-con VAE	262.48502	2.0973198
Upsc-con VAE	268.64572	2.0683987

Figure 4. Conditioned model performance on FID and IS. Note that VAEs performance dramatically increased while GANs suffered a minor decrease in performance.

5. Analysis

The results obtained from this study were interesting. Overall, GANs outclassed VAEs in almost every metric applied for this study. This result was to be expected as it has been shown that VAEs typically generate blurrier images than GANs and both FID and IS are metrics that punish models that produce samples that are classified as blurry (Cai et al., 2017; Barratt & Sharma, 2018).

A surprising result that I noticed was that across all models, models that used the upscaling convolution method for image generation scored poorer overall than their peers that utilized transposed convolutions. This was surprising to me because previous literature has noted images generated using upscaling convolutions produced images that were clearer and devoid of the checkerboarding pattern that is typically associated with transposed convolution. However, according to FID and IS, the results obtained from the use of these techniques were worse. In prior experiments I ran with the upscaling convolution blocks, I noticed that the images generated by this model were typically less sharp than those generated by models that utilized the transposed convolutions. I suspect because images that were produced using this method were less sharp, FID and IS most likely would have punished models that used this approach hence

the score. Overall, more analysis and digging needs to be done to find out the cause for this decrease.

Another surprising result that I encountered was that conditioned GANs performed worse than their non-conditioned counterparts. This went counter to my hypothesis as I predicted that the additional label would help the conditional GAN organize its learned distribution of images better than the traditional unconditioned GAN. Even more interestingly, the VAE improved significantly as a result of conditioning which was surprising to me as these results go in contrast to what occurred with the GANs. I suspect this could be due to the stochastic and non-deterministic process of training, but more work must be done to isolate the cause of the conflicting results between the GAN and the VAE.

6. Limitations

In terms of the limitations of this study, I think the biggest one to consider is the stochastic nature of training and replication on multiple datasets which was not performed in this study due to time constraints. An extension of this study to other datasets would help to determine if these results are consistent across multiple datasets which would allow for a more robust conclusion to be drawn in terms of the results of this study.

7. Conclusion

Despite the fact that GANs and VAEs have been succeeded by more performant models like the Diffusion model, there still can be vital insight obtained by exploring and experimenting with these older models. In my exploration of GANs and VAEs, I found that conditioning the VAE resulted in a significant performance gain over the unconditioned variant of the model indicating that if this performance is consistent across multiple datasets of various sizes and types, there may be a case to only utilize the conditional variant.

This insight could be particularly important because if we know there is some kind of consistent relationship between image quality and the inclusion of a label during training, then this could have a dramatic impact in a lot of areas of generative visual machine learning, from data collection to the design of future techniques. However, one important takeaway that should be considered is that more investigation should be done into conditioning and how it affects GANs specifically over VAEs, as the results I had showed that GANs got worse slightly with conditioning and VAEs got significantly better.

References

Alqahtani, H., Kavakli, M., and Kumar Ahuja, D. G. An analysis of evaluation metrics of gans. 07 2019.

Barratt, S. and Sharma, R. A note on the inception score, 2018.

Borji, A. Pros and cons of GAN evaluation measures. *CoRR*, abs/1802.03446, 2018. URL http://arxiv.org/abs/1802.03446.

Brock, A., Donahue, J., and Simonyan, K. Large scale gan training for high fidelity natural image synthesis, 2019.

Cai, L., Gao, H., and Ji, S. Multi-stage variational auto-encoders for coarse-to-fine image generation. *CoRR*, abs/1705.07202, 2017. URL http://arxiv.org/abs/1705.07202.

Dhariwal, P. and Nichol, A. Diffusion models beat gans on image synthesis. *CoRR*, abs/2105.05233, 2021. URL https://arxiv.org/abs/2105.05233.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial networks, 2014.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., and Courville, A. C. Improved training of wasserstein gans. *CoRR*, abs/1704.00028, 2017. URL http://arxiv.org/abs/1704.00028.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Klambauer, G., and Hochreiter, S. Gans trained by a two time-scale update rule converge to a nash equilibrium. *CoRR*, abs/1706.08500, 2017. URL http://arxiv.org/abs/1706.08500.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes, 2022.

Krizhevsky, A. Learning multiple layers of features from tiny images. *University of Toronto*, 05 2012.

Kwak, H. and Zhang, B. Ways of conditioning generative adversarial networks. *CoRR*, abs/1611.01455, 2016. URL http://arxiv.org/abs/1611.01455.

Li, Y., Swersky, K., and Zemel, R. S. Generative moment matching networks. *CoRR*, abs/1502.02761, 2015. URL http://arxiv.org/abs/1502.02761.

Mirza, M. and Osindero, S. Conditional generative adversarial nets. *CoRR*, abs/1411.1784, 2014. URL http://arxiv.org/abs/1411.1784.

Odena, A., Dumoulin, V., and Olah, C. Deconvolution and checkerboard artifacts. *Distill*, 2016. doi: 10.23915/distill.00003. URL http://distill.pub/2016/deconv-checkerboard.

Odena, A., Olah, C., and Shlens, J. Conditional image synthesis with auxiliary classifier gans, 2017.

Pu, Y., Gan, Z., Henao, R., Yuan, X., Li, C., Stevens, A., and Carin, L. Variational autoencoder for deep learning of images, labels and captions, 2016.

Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks, 2016.

Rogers, A. Changing the world by changing the data. *CoRR*, abs/2105.13947, 2021. URL https://arxiv.org/abs/2105.13947.

Salimans, T., Goodfellow, I. J., Zaremba, W., Cheung, V., Radford, A., and Chen, X. Improved techniques for training gans. *CoRR*, abs/1606.03498, 2016. URL http://arxiv.org/abs/1606.03498.

Sohn, K., Lee, H., and Yan, X. Learning structured output representation using deep conditional generative models. In Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper_files/paper/2015/file/8d55a249e6baa5c06772297520da2051-Paper.pdf.

Taye, M. M. Theoretical understanding of convolutional neural network: Concepts, architectures, applications, future directions. *Computation*, 11(3), 2023. ISSN 2079-3197. doi: 10.3390/computation11030052. URL https://www.mdpi.com/2079-3197/11/3/52.

Xue, Y., Guo, Y.-C., Zhang, H., Xu, T., Zhang, S.-H., and Huang, X. Deep image synthesis from intuitive user input: A review and perspectives. *Computational Visual Media*, 8:3–31, 03 2022. doi: 10.1007/s41095-021-0234-8.

Table 1: Transposed Convolution GAN Architecture

Operation	Kernel	Strides	Feature maps	BN?	Nonlinearity
$G_x(z)$ - 110 x 1 x 1 input					
Linear	N/A	N/A	3520		Leaky ReLU
Transposed Convolution	4 x 4	2 x 2	192	✓	Leaky ReLU
Transposed Convolution	4 x 4	2 x 2	96	✓	Leaky ReLU
Transposed Convolution	4 x 4	2 x 2	48	✓	Leaky ReLU
Transposed Convolution	4 x 4	2 x 2	3		Sigmoid
$D(x)$ - 32 x 32 x 3 input					
Convolution	3 x 3	2 x 2	62	✓	Leaky ReLU
Convolution	3 x 3	2 x 2	128	✓	Leaky ReLU
Convolution	3 x 3	2 x 2	256	✓	Leaky ReLU
Linear	N/A	N/A	1		Sigmoid

All GANs using the following:
Optimizer: Adam ($\alpha = 0.0002$, $\beta_1 = 0.5$, $\beta_2 = 0.999$)
Batch Size: 100
Leaky ReLU slope: 0.2
Epochs: 100

Table 2: Upscale Convolution GAN Architecture

Operation	Kernel	Strides	Feature maps	BN?	Nonlinearity
$G_x(z)$ - 110 x 1 x 1 input					
Linear	N/A	N/A	3520		Leaky ReLU
Upscale	N/A	N/A	N/A	N/A	N/A
Convolution	4 x 4	1 x 1	192	✓	Leaky ReLU
Upscale	N/A	N/A	N/A	N/A	N/A
Convolution	4 x 4	1 x 1	96	✓	Leaky ReLU
Upscale	N/A	N/A	N/A	N/A	N/A
Convolution	4 x 4	1 x 1	48	✓	Leaky ReLU
Convolution	4 x 4	1 x 1	3		Sigmoid
$D(x)$ - 32 x 32 x 3 input					
Convolution	3 x 3	2 x 2	62	✓	Leaky ReLU
Convolution	3 x 3	2 x 2	128	✓	Leaky ReLU
Convolution	3 x 3	2 x 2	256	✓	Leaky ReLU
Linear	N/A	N/A	1		Sigmoid

Table 3: Conditional Transposed Convolution GAN Architecture

Operation	Kernel	Strides	Feature maps	BN?	Nonlinearity
$G_x(z)$ - 120 x 1 x 1 input					
Linear	N/A	N/A	3520		Leaky ReLU
Transposed Convolution	4 x 4	2 x 2	192	✓	Leaky ReLU
Transposed Convolution	4 x 4	2 x 2	96	✓	Leaky ReLU
Transposed Convolution	4 x 4	2 x 2	48	✓	Leaky ReLU
Transposed Convolution	4 x 4	2 x 2	3		Sigmoid
$D(x)$ - 32 x 32 x 13 input					
Convolution	3 x 3	2 x 2	62	✓	Leaky ReLU
Convolution	3 x 3	2 x 2	128	✓	Leaky ReLU
Convolution	3 x 3	2 x 2	256	✓	Leaky ReLU
Linear	N/A	N/A	1		Sigmoid

Table 4: Conditional Upscale Convolution GAN Architecture

Operation	Kernel	Strides	Feature maps	BN?	Nonlinearity
$G_x(z)$ - 120 x 1 x 1 input					
Linear	N/A	N/A	3520		Leaky ReLU
Upscale	N/A	N/A	N/A	N/A	N/A
Convolution	4 x 4	1 x 1	192	✓	Leaky ReLU
Upscale	N/A	N/A	N/A	N/A	N/A
Convolution	4 x 4	1 x 1	96	✓	Leaky ReLU
Upscale	N/A	N/A	N/A	N/A	N/A
Convolution	4 x 4	1 x 1	48	✓	Leaky ReLU
Convolution	4 x 4	1 x 1	3		Sigmoid
$D(x)$ - 32 x 32 x 13 input					
Convolution	3 x 3	2 x 2	62	✓	Leaky ReLU
Convolution	3 x 3	2 x 2	128	✓	Leaky ReLU
Convolution	3 x 3	2 x 2	256	✓	Leaky ReLU
Linear	N/A	N/A	1		Sigmoid

Table 5: Transposed Convolution VAE Architecture

Operation	Kernel	Strides	Feature maps	BN?	Nonlinearity
$D_x(z)$ - 110 x 1 x 1 input					
Linear	N/A	N/A	2048	N/A	ReLU
Transposed Convolution	4 x 4	1 x 1	192	✓	ReLU
Transposed Convolution	4 x 4	1 x 1	96	✓	ReLU
Convolution	4 x 4	1 x 1	3	N/A	N/A
$D(x)$ - 32 x 32 x 3 input					
Convolution	3 x 3	2 x 2	62	✓	ReLU
Convolution	3 x 3	2 x 2	128	✓	ReLU
Convolution	3 x 3	2 x 2	256	✓	ReLU
Linear	N/A	N/A	1	N/A	Sigmoid

All VAEs using the following:
Optimizer: Adam ($\alpha = 0.0002$, $\beta_1 = 0.9$, $\beta_2 = 0.999$)
Batch Size: 100
Leaky ReLU slope: 0.2
Epochs: 100

Table 6: Upscale Convolution VAE Architecture

Operation	Kernel	Strides	Feature maps	BN?	Nonlinearity
$D_x(z)$ - 110 x 1 x 1 input					
Linear	N/A	N/A	2048	N/A	ReLU
Upscale	N/A	N/A	N/A	N/A	N/A
Convolution	4 x 4	1 x 1	192	✓	ReLU
Upscale	N/A	N/A	N/A	N/A	N/A
Convolution	4 x 4	1 x 1	96	✓	ReLU
Convolution	4 x 4	1 x 1	3	N/A	N/A
$D(x)$ - 32 x 32 x 3 input					
Convolution	3 x 3	2 x 2	62	✓	ReLU
Convolution	3 x 3	2 x 2	128	✓	ReLU
Convolution	3 x 3	2 x 2	256	✓	ReLU
Linear	N/A	N/A	1	N/A	Sigmoid

Table 7: Conditional Transposed Convolution VAE Architecture

Operation	Kernel	Strides	Feature maps	BN?	Nonlinearity
$D_x(z)$ - 120 x 1 x 1 input					
Linear	N/A	N/A	2048	N/A	ReLU
Transposed Convolution	4 x 4	1 x 1	192	✓	ReLU
Transposed Convolution	4 x 4	1 x 1	96	✓	ReLU
Convolution	4 x 4	1 x 1	3	N/A	N/A
$D(x)$ - 32 x 32 x 13 input					
Convolution	3 x 3	2 x 2	62	✓	ReLU
Convolution	3 x 3	2 x 2	128	✓	ReLU
Convolution	3 x 3	2 x 2	256	✓	ReLU
Linear	N/A	N/A	1	N/A	Sigmoid

Table 8: Upscale Convolution VAE Architecture

Operation	Kernel	Strides	Feature maps	BN?	Nonlinearity
$D_x(z)$ - 120 x 1 x 1 input					
Linear	N/A	N/A	2048	N/A	ReLU
Upscale	N/A	N/A	N/A	N/A	N/A
Convolution	4 x 4	1 x 1	192	✓	ReLU
Upscale	N/A	N/A	N/A	N/A	N/A
Convolution	4 x 4	1 x 1	96	✓	ReLU
Convolution	4 x 4	1 x 1	3	N/A	N/A
$D(x)$ - 32 x 32 x 13 input					
Convolution	3 x 3	2 x 2	62	✓	ReLU
Convolution	3 x 3	2 x 2	128	✓	ReLU
Convolution	3 x 3	2 x 2	256	✓	ReLU
Linear	N/A	N/A	1	N/A	Sigmoid

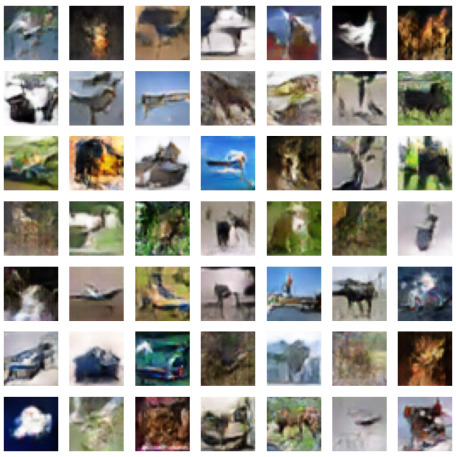

Figure 1: Unconditioned transpose convolution GAN images

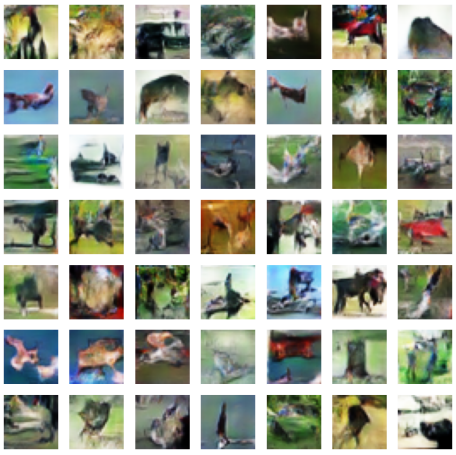

Figure 2: Unconditioned upscale convolution GAN images

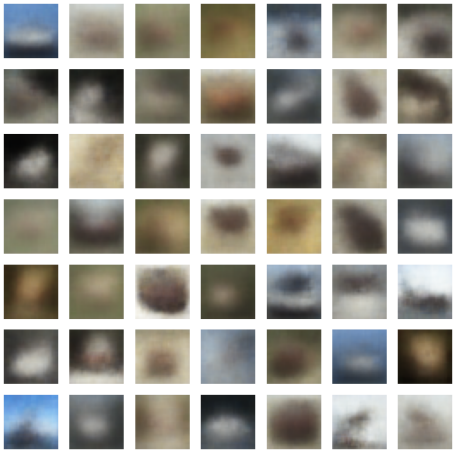

Figure 3: Unconditioned transpose convolution VAE images

Figure 4: Unconditioned upscale convolution VAE images

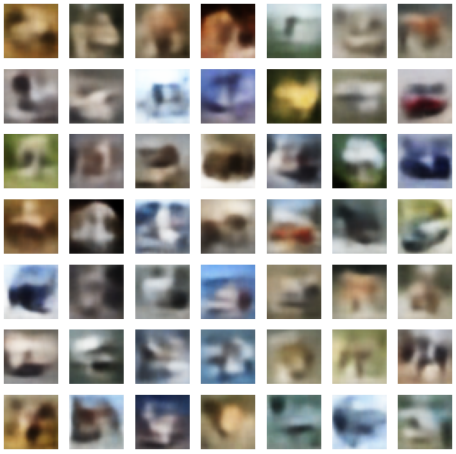

Figure 5: Conditioned transpose convolution VAE images

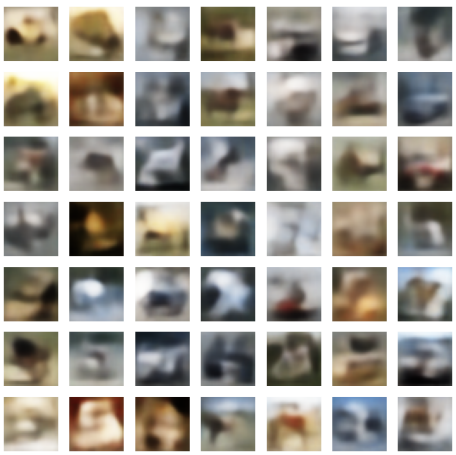

Figure 6: Conditioned upscale convolution VAE images

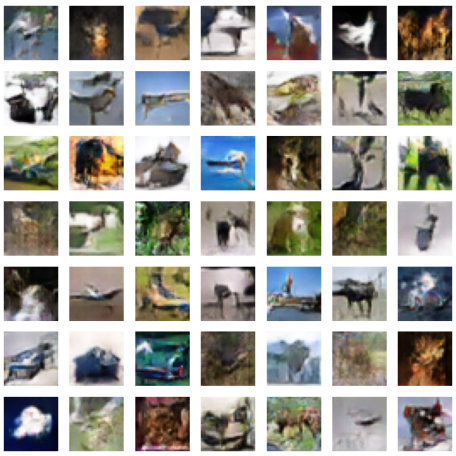

Figure 7: Conditioned transpose convolution GAN images

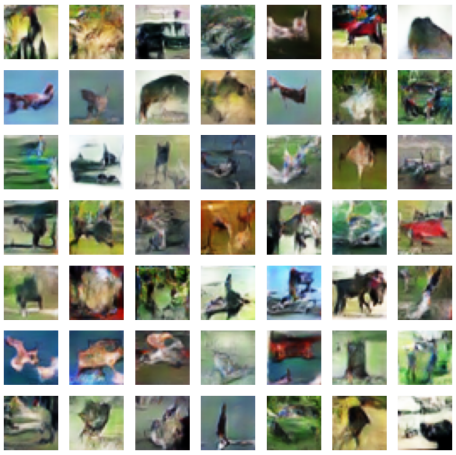

Figure 8: Conditioned upscale convolution GAN images