Limitations of Generative Models
Variational Auto Encoders and Generative Adversarial Networks

Rama Krishna Raju
Center for
Data Science
New York University
rks395@nyu.edu

Pavan Sudhir Nallam
Center for
Data Science
New York University
psn240@nyu.edu

Abstract

Several deep generative models like variational auto encoders and generative adversarial networks have seen success recently and resulted in many applications. In this work, we wanted to take a step back and understand their power and limitations in approximating various data density estimations. Following various experiments we show that these generative models are incredibly powerful in modelling datasets with high global structure/information but they under-perform or even break off when it comes to modelling datasets with high local structure. Further we also studied their data modelling capabilities in various noisy conditions. Here both models have shown some noteworthy similarities and differences and we present them in detail.

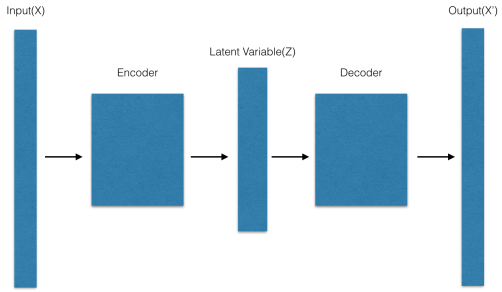
1 Generative Models

Generative modeling is the task of building a model of data that we can sample from. This is in contrast to discriminative modeling, such as regression or classification, which tries to estimate conditional distributions such as $p(class|data)$.

Traditionally to obtain a generative model g given training data X, it is posed as a maximum-likelihood estimate problem, which requires calculations of marginal probabilities, partition functions, most-likely estimates, etc. This may be feasible when the generative model is a GMM, but to build a generative model out of a deep neural network, this quickly becomes intractable, and we discuss two recently proposed approaches to solve this problem namely Variational Auto-encoders (VAE) and Generative Adversarial Networks (GAN).

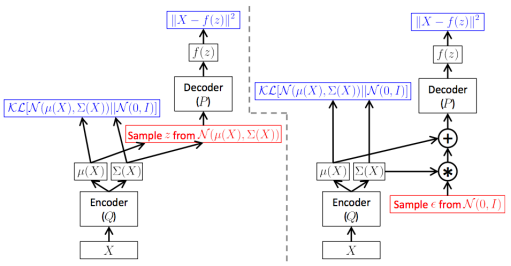
1.1 Variational Auto-encoders

Variational Auto-encoders is one of the implementations of Generative Models. As explained in (Kingma, D. P. and Welling, M, 2013), VAE has two main neural network components- encoder and decoder. Encoder takes in the input(X) and converts to latent variable(Z). And decoder takes the latent variable and converts it back to original input(X').



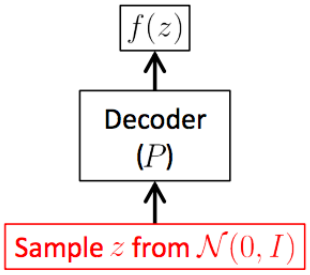
VAE model architecture

During the training we would feed the encoder with real inputs, which would train the encoder neural network and generate the latent variable. We would take gaussian prior over the latent variable. And the encoder takes the latent variable and generated the output, which is compared to the input.



VAE Training(Carl Doersch, 2016)

Once the training is done we can generate latent variable(Z) from a zero-mean and one standard deviation gaussian distribution and pass it to the decoder to generate new samples of data.



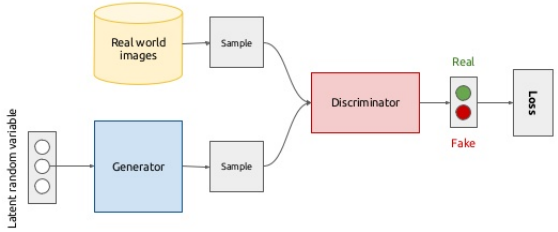
Sample generation from VAE(Carl Doersch, 2016)

1.2 Generative Adversarial Networks

Adversarial training is an another way to train a generative model without the previously mentioned intractable calculations. Let's assume our training data X^d. The basic idea is that we will have two adversarial models - a generator $G :^{n\ d}$ and a discriminator $D :^{d}\ 0, 1$. The task of the generator is to with take a latent noise vector from a standard random distribution (e.g. an n-dimensional Gaussian) and produce a sample from the same distribution as X. The discriminator, on the other hand, should discriminate between samples from the true data X and the data generated by G. Each model is trying to best the other - the generator's objective is to fool the discriminator and the discriminator's objective is to not be fooled by the generator, and their framework is as shown below.

So, it is a new framework for estimating generative models via an adversarial process with:

- Two models trainable simultaneously through backpropagation corresponding to a minimax two-player game.
- A generative model G:
  - Captures the data distribution.
  - Train to minimize the probability of D making a mistake.
- A discriminative model D:
  - Estimates the probability that a sample came from the training data rather than G.



Generative Adversarial Model

When both G and D are neural nets, they can be trained by expressing their objectives as a loss function that can be optimized via gradient descent. In other words, D and G play the following two-player minimax game with objective function $V(G, D)$:

$$\min_G \max_D V(D, G) = \mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))].$$

After G and D are trained, the result is that the generator and the discriminator each get better at their objectives in tandem, so that at the end, the generator is able to or is close to being able to fool the most sophisticated discriminator.

Below is the Adversarial training algorithm as mentioned by Ian J. Goodfellow (Ian J. Goodfellow et.al, 2014)

for number of training iterations **do**
  for k steps **do**
  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
  - Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

  end for
  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Update the generator by descending its stochastic gradient:

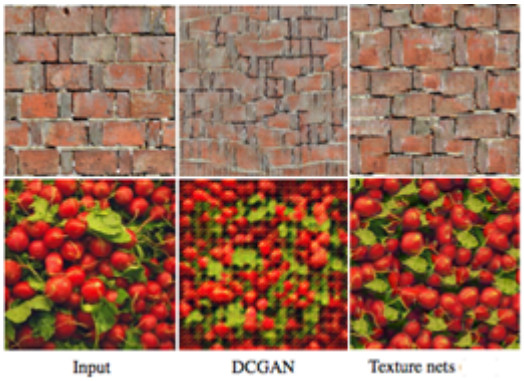
$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

2 Understanding the limitations of Generative Models

Most published research work, report many success and applications of these generative models. They have been found to generate decent quality images from datasets like MNIST (digits), CelebA/LFW(faces), ImageNet(various objects), LSUN(bedrooms).

We wanted to understand the limitations of these generative models so as to get an idea of why and when these methods are applicable. Literature study revealed that DCGAN (Alec Radford et.al, 2015) was not good in recovering textures as shown below in (Dmitry Ulyanov et.al, 2016), so they propose TextureNet to model/generate texture images.



Comaparision of images generated from DCGAN to TextureNet

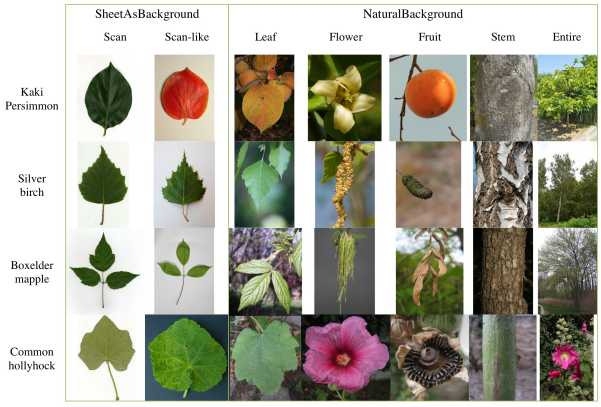


Figure 1: Description of Plant Dataset

2.1 Data

The main visual difference between the datasets which has seen success and others is that texture dataset has a lot of low level details when compared to others which has high level details or a global structure. Hence we have explored various datasets namely textures (M. Cimpoi et.al, 2014), trees (authors, 2013) and faces (Gary B. Huang et.al, 2007), (Ziwei Liu et.al, 2015) as we wanted to see where and how these generative models break.

Texture dataset, consists of 5640 images, organized according to a list of 47 terms (categories) inspired from human perception. There are 120 images for each category.

Plant dataset contains images on 250 herb and tree species from France area. It contains 26077 pictures belonging each to one of the 2 following categories.

- SheetAsBackground: exclusively pictures of leaves in front of a white or colored uniform background - prominent global details
- NaturalBackground: free natural photographs of different views on different sub parts of a plant into the wild - prominent local details

These 2 main categories are also subdivided into 2 and 5 sub-categories as shown in Figure 1.

LFW face dataset contains nearly 13,000 images of faces of different persons collected from the web. Similarly we also trained on celebA dataset with 202600 images.

2.2 Experiments

We have used VAE (Ilya Kostrikov, 2016) GAN (Taehoon Kim, 2016) to model these datasets. Figure 2 shows the images generated over texture data. Figure 3 shows images generated over trees dataset. And finally figure 4 shows the images generated over faces dataset. As we can see the faces generated through GAN are not that clear, so we have used celebA dataset, which have far more data than LFW (Although we have not used this dataset for further experimentation due to time constraints apart from understanding the power of GAN to model faces). GAN generated images through this celebA dataset is shown in figure 5

2.3 Conclusions

As we can see, we are able to model faces with both VAE and GAN, We have also found that both VAE and GAN were performing very good at generating MNIST digits. But both VAE and GAN are finding it difficult to model other datasets like textures and plants. They were performing very badly on texture dataset. In case of plant dataset, they are slightly better in generating images containing objects like leaves and fruits but they were not able to do the same for full images of plants/ trees.

From these results we were able to draw some insights as mentioned below.

- Images with Global Structure: Both VAE and GAN's better at reconstructing the global structure.
- Images with Local Structure/ Minute Details: Both VAE and GAN's are very bad at reconstructing them.

VAE vs GAN: While both are bad at generating the local structure, GAN has slight advantage while generating the global structure as the images are less blurrer than that of VAE generated

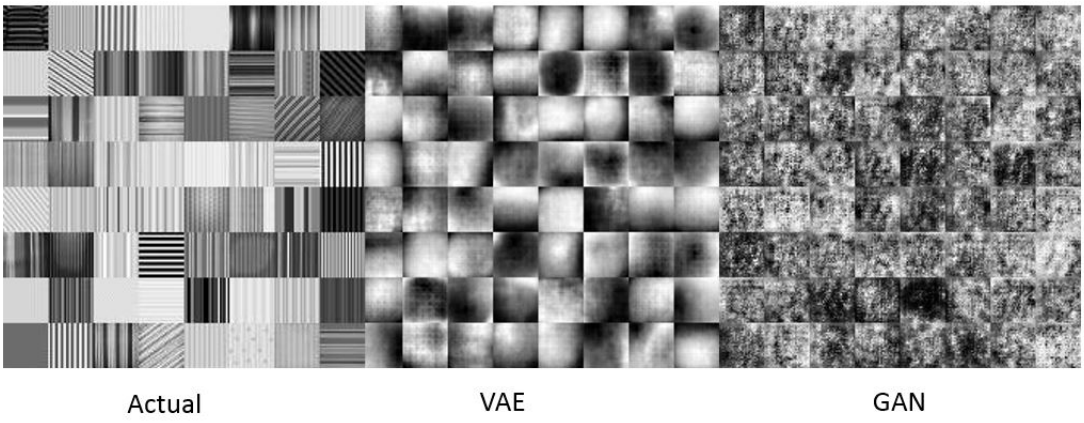


Figure 2: Texture Data - Original, Generated from VAE, Generated from GAN

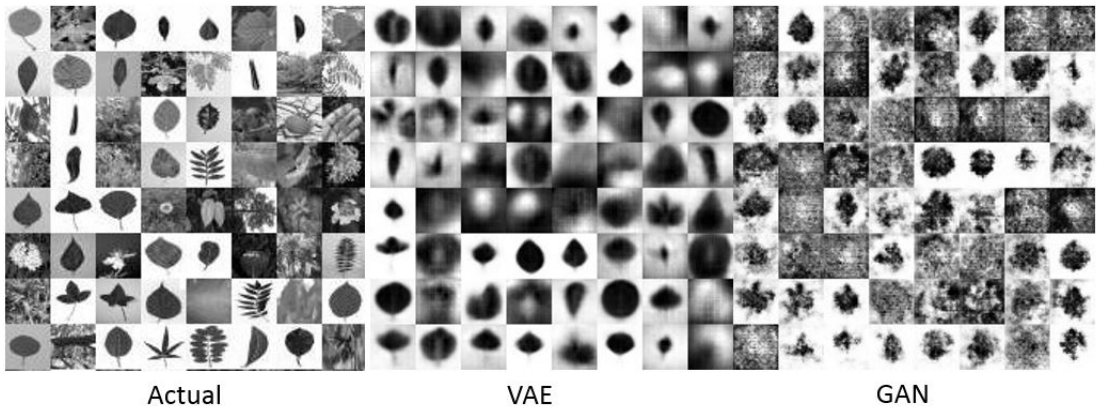


Figure 3: Plant Data - Original, Generated from VAE, Generated from GAN

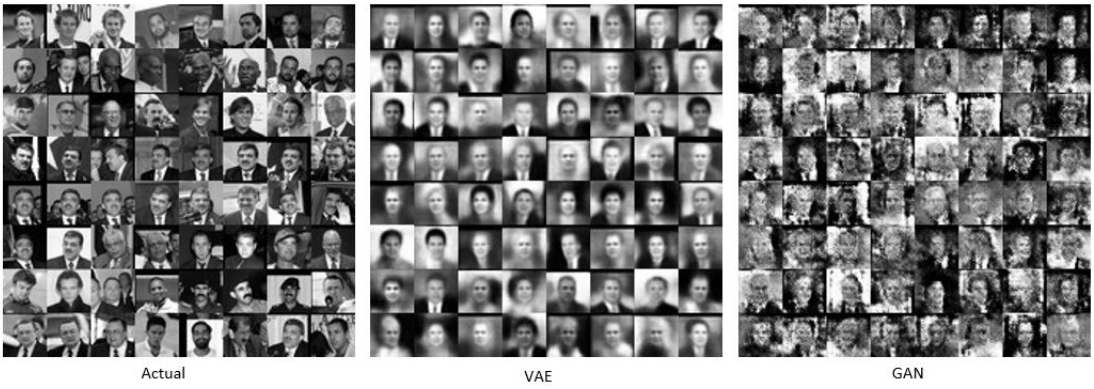


Figure 4: Face Data - Original, Generated from VAE, Generated from GAN

Figure 5: Face Data - Generated from GAN with more training data and time

images. This might be because we are using two different kinds of Neural Networks playing an adversarial game (minimizing jensen shannon divergence) whose combined strength is more than sum of its parts. Since we have found that GAN training takes a lot of time, (leaving us to somehow reduce the experiment scope) we have reduced the image size and training data, but that gave results not as good as VAE.

In general, while humans draw the images, most of them are good at quickly drawing the global structure. What is really hard or time taking is filling in the minute details or the local structure (as it has to be done with care). Since faces, leaves or fruits or digits are kind of objects with relatively easy structure VAE/GAN's are able to generate them. In case of texture or full plants, where minute details dominate the global structure GAN's or VAE's are not able to generate them effectively.

3 Generative Modelling of Images under noisy conditions

Now that we know VAE and GAN's were able to model datasets like faces, digits and objects like leaves/ fruits etc, we wanted to understand if they would be able to model them in the presence of various kinds of noises. Hence we have added additive (white) noise as well as geometric (spline image warp) noise to the face dataset to experiment if we can model them or not.

3.1 Data

We have prepared various datasets with both kinds of noises and at varying level of noises added to the face dataset i.e we have added low additive noise, medium additive noise, high additive noise, low geometric noise, medium geometric noise, high geometric noise etc. A brief overview of the kinds of noise is given below.

- Additive white Gaussian noise(AWGN): It is a basic noise model used in Information theory to mimic the effect of many random processes that occur in nature. The modifiers denote specific characteristics:
  - Additive because it is added to any noise that might be intrinsic to the information system.
  - White refers to the idea that it has uniform power across the frequency band for the information system. It is an analogy to the color white which has uniform emissions at all frequencies in the visible spectrum.
  - Gaussian because it has a normal distribution in the time domain with an average time domain value of zero.

  $Y_i = X_i + Z_i \sim \mathcal{N}(X_i, N)$

  Figure 6 shows how the figure distorts while we vary the variance of the white noise.

- Geometric Noise: Warping, or domain distortion is a very common technique in computer graphics for generating procedural textures and geometry. It's often used to pinch an object, stretch, twist or bend an object or make it thicker or apply any deformation we want. We have performed spline warping which produces a smooth vector field.

  Figure 7 shows for different levels of geometric noise (spline warping) distorts the image.

3.2 Experiments

We have experimented with new noise added face datasets and results are shown in figures 8, 9, 10 and 11.

3.3 Conclusion

From the above and similar experiments it seems that both VAE and GAN handles these different

Figure 6: Original image, along with various levels of additive noise

Figure 7: Original image, along with various levels of spline warping transformations to it

Figure 8: Faces generated from VAE with various level of additive noise

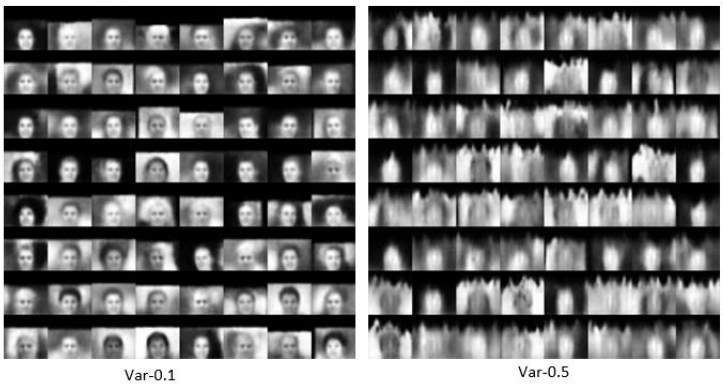


Figure 9: Faces generated from VAE with various level of geometric noise

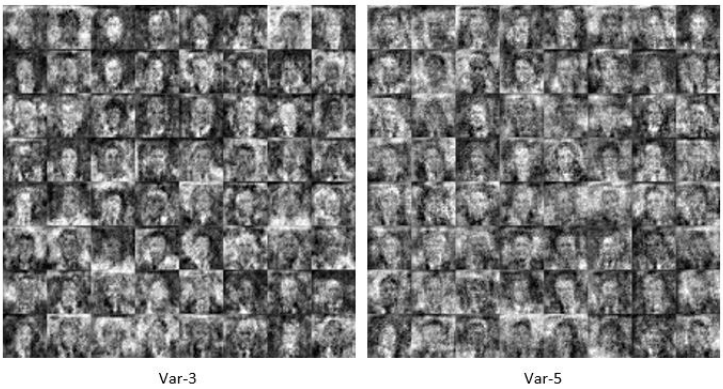


Figure 10: Faces generated from GAN with various level of additive noise

Figure 11: Faces generated from GAN with various level of geometric noise

kinds of noises in a different way. However we observed that their behaviour changed when our point of interest as mentioned below.

Preserving latent information: With increasing levels of additive noise the VAE generated images are centered more towards the mean of the face data distribution having very little variance. In the dataset we have several latent factors/ or variables for the face images like gender, angle, direction/orientation, with varying moods and expressions etc. But the VAE generated images seem to average out these so that we can only infer that the image is a face, but cant really infer the other latent factors. To the contrary GAN seem to preserve this latent information, even when the quality of the generated images was just moderate.

Quality of Generation: Based on the experimental set up, VAE was able to generate better quality images, but we are not really sure about GAN where the quality suffered a bit. We are not sure to conclude this as GAN's need more training data and time, along with some hard parameter tuning to train better.

Breaking point: Both models break often and early in the presence of geometric noise, but VAE is able to handle additive noise better than GAN (under the current experimental set up)

An interesting observation is the one . For images with geometric noise, it seems that both the models, were failing badly and we can attribute this behaviour to that fact that geometric warping was adding more chaotic behaviour to the images resulting in complete shape and structure deformation thus transforming the dataset to appear more like texture than like faces. Hence we believe, both VAE an GAN were finding it hard to model such noises.

4 Future Work and Challenges

As we have seen VAE's and GAN's both have their perks and drawbacks, when come to generating images. VAE's are all about generating smooth images with a correct global subject, so they are easier and faster to train. Whereas GAN generated images have intrinsic local features with less emphasis on the global structure, so they have control on the latent space. Hence some researchers are combining both to get best of both the worlds (Anders Boesen Lindbo Larsel et.al, 2016) (Alireza Makhzani et.al, 2016).

Below are some images generated by the VAE/GAN model.The authors claim that compared to the plain VAE, the VAE/GAN images are more interesting as they contain more details.

As we have seen, VAE and GAN handle noises in different way, so it would be interesting to see how the combined model would handle noise.

Acknowledgments

We would like to thank Professor Joan Bruna for his valuable time and help and suggestions in designing the experimental framework and insights.

References

Kingma, D. P. and Welling, M. 2013. *Auto-encoding variational Bayes*.

Carl Doersch. 2016. *Tutorial on Variational Autoencoders*.

Ian J. Goodfellow et.al 2014 *Generative Adversarial Nets*

Alec Radford et.al 2015 Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks

Anders Boesen Lindbo Larsel et.al 2016 Autoencoding beyond pixels using a learned similarity metric

Alireza Makhzani et.al 2016 Adversarial Autoencoders

Søren Kaae Sønderby 2016 https://github.com/skaae/vaeblog/,

Dmitry Ulyanov et.al 2016 Texture Networks: Feedforward Synthesis of Textures and Stylized Images,

M. Cimpoi et.al 2014 Describing Textures in the Wild

2013 http://www.imageclef.org/2013/plant,

Gary B. Huang et.al 2007 Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments,

Ziwei Liu et.al 2015 Deep Learning Face Attributes in the Wild

Ilya Kostrikov 2016 https://github.com/ikostrikov/TensorFlow-VAE-GAN-DRAW

Taehoon Kim, 2016 https://github.com/carpedm20/DCGAN-tensorflow