A Probe into Understanding GAN and VAE models

Jingzhao Zhang [*1] **Lu Mi** [*1] **Macheng Shen** [*1]

[1] **Massachusetts Institute of Technology, Cambridge, USA**

$*$ **These authors equally contributed to this work**

Abstract

Both generative adversarial network models and variational autoencoders have been widely used to approximate probability distributions of datasets. Although they both use parametrized distributions to approximate the underlying data distribution, whose exact inference is intractable, their behaviors are very different. In this report, we summarize our experiment results that compare these two categories of models in terms of fidelity and mode collapse. We provide a hypothesis to explain their different behaviors and propose a new model based on this hypothesis. We further tested our proposed model on MNIST dataset and CelebA dataset.

1. Introduction

One way to interpret the goal of unsupervised learning algorithms (Blei et al., 2003; Wainwright et al., 2008; Andrieu et al., 2003) is that they try to describe the distribution of the true data using samples from the dataset. These algorithms all model the dataset with some probability distribution, and learn an approximate distribution from the data samples. However, without further constraints, solving such problems in high dimension is intractable, which requires exponentially growing number of samples. As a result, practical algorithms balance the model complexity and sample complexity to trade off model accuracy for efficiency.

More formally, assuming that random samples from a data set are drawn from an underlying true distribution $X \sim p(x)$, our goal is to design an algorithm that produces a distribution $q(x)$ based on i.i.d samples $x_1, x_2, ..., x_n$ from the true distribution. Mathematically, the algorithm aims to minimize the divergence

$$D_\phi(p|q) = E_p[\phi(\frac{q(x)}{p(x)})],$$

where the function ϕ is determined by the actual application.

For most function ϕ (e.g. $\phi = \log$ in KL-divergence), the divergence is minimized when $q(x) = p(x)$ almost everywhere. However, finding such a distribution $q(x)$ exactly requires infinite number of samples. Therefore, many algorithms parametrize the approximate distribution by $q(x;\theta)$, such that it only searches within a probability family $Q = q(x;\theta)|\theta \in \Theta$ of nice properties that make solving the problem more tractable. The latent Dirichlet analysis, for example, uses conjugate prior distributions in graphical models to allow deriving analytic expression of maximum likelihood estimators. In addition, parameterizing builds prior information into the model as a regularization and leads to better generalization results.

However, the problem of finding a good distribution family Q itself may be hard. When Q is too general, the algorithm may consume too many samples or require too much computing power. When Q is not expressive enough, the result may have a large bias. In order to solve this problem, some works (Rezende & Mohamed, 2015; Ranganath et al., 2016; Loaiza-Ganem et al., 2017) proposed to use neural networks to parametrize probability distribution. The high representation ability of neural networks along with backpropagation algorithms make these algorithms very generalizable and efficient.

In addition to these results, another line of works (Goodfellow et al., 2014; Kingma & Welling, 2013; Arjovsky et al., 2017; Larsen et al., 2015; Mirza & Osindero, 2014; Radford et al., 2015) is more dedicated to approximating data distribution in image datasets. These models generate high quality natural images and hence have attracted much attention in recent years. However, it is also widely known that GAN style models are very sensitive to training parameter tuning and suffers from unstable convergence and mode collapse. In our project, we first provide a brief overview of these models in section 2. We then reproduce the experiments of four different generative models and compare their performance in terms of image diversity (measured by

entropy) and image fidelity in section 3. Based on these results, we propose a hypothesis that explains the difference between GAN and VAE in section 4. We further propose a new model and test on MNIST and CelebA datasets. The experiment results are also included in this project.

2. Deep generative models

Generative adversarial network (GAN) and Variational autoencoder (VAE) are two commonly used deep generative models that can generate complicated synthetic images. In this section, we will introduce four variations of GAN and VAE: (1) Vanilla GAN (Goodfellow et al., 2014), (2) Wasserstein GAN (WGAN) (Arjovsky et al., 2017), (3) Vanilla VAE (Kingma & Welling, 2013), (4) VAE-GAN (Larsen et al., 2015). We will focus on the intuition, mathematical formulation and the issues with each of the models.

2.1. Generative adversarial network

GAN (Goodfellow et al., 2014) uses two deep neural networks (namely, a generator and a discriminator) to train a generator of images. The generator is typically a de-convolutional neural network (DCN), and the discriminator is typically a convolutional neural network (CNN). During training, the generator takes in fixed dimensional noise vectors, which are called the latent variable, and outputs images. The generated synthetic images are blended with the true images from a dataset and fed into the discriminator. The classification accuracy of the discriminator is then fed back to the generator. Therefore, the training objective of the generator is to increase the classification error of the discriminator and that of the discriminator is to decrease the classification error. This training objective can be summarized as the following minimax problem:

$$\min_G \max_D V(D,G) = \mathbb{E}_{\mathbf{x}\sim\mathbf{p_{data}(x)}}[logD(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim\mathbf{p_z(z)}}[1 - logD(G(\mathbf{z}))], \tag{1}$$

where G is the mapping from the latent space to the data space, and D is the discriminator loss measuring how well the discriminator classifies the blended data.

The optimization problem defined by Eq. 1 can be viewed as a zero-sum game, which is shown to have a unique equilibrium point. This equilibrium point corresponds to the optimal distribution of the generated image, induced by the generative network, that solves the optimization problem. This provides a general framework for training of deep generative models. Nonetheless, it turns out that the training of this model is difficult when the discriminator is trained too well. That is, if the discriminator is too powerful, then the training gradient for the generator will vanish. Therefore, the authors of the GAN paper proposed another loss function:

$$\min_G \max_D V(D,G) = \mathbb{E}_{\mathbf{x}\sim\mathbf{p_{data}(x)}}[logD(\mathbf{x})] - \mathbb{E}_{\mathbf{z}\sim\mathbf{p_z(z)}}[logD(G(\mathbf{z}))]. \tag{2}$$

The problem with this optimization is that the resulted optimal distribution suffers from mode collapse. That is, the optimal distribution can only represent a sub-class of instances appearing in the data distribution. It turns out that both of the training difficulty and the mode collapse problem are due to the inappropriate functional form of the loss function. This is modified in WGAN such that these two problems are avoided.

2.2. Wasserstein GAN

It has been shown in (Arjovsky et al., 2017) that the first optimization, Eq. 1, is essentially equivalent to minimizing the following objective, when the discriminator is fixed and optimal:

$$2JS(P_{data}||P_G) - 2log2, \tag{3}$$

where JS is the Jensen–Shannon divergence. When P_{data} and P_G are quite different from each other, $JS(P_{data}||P_G)$ becomes a constant. Therefore, the gradient vanishes, which is problematic for training with gradient descent.

Likewise, the second optimization, Eq. 2, is essentially equivalent to minimizing the following objective, when the discriminator is fixed and optimal:

$$KL(P_G||P_{data}) - 2JS(P_{data}||P_G), \tag{4}$$

where P_G is the distribution of the generator, P_{data} is the data distribution, KL is the Kullback–Leibler divergence. This is undesirable, as it wants to minimize the KL divergence while maximize the JS divergence simultaneously, which does not make sense.

Moreover, this objective function assigns different penalty to two different types of error that the generator makes. Suppose $P_G(x) \to 0, P_{data}(x) \to 1$, which means the generator does not generate a realistic image, the corresponding penalty $KL(P_G||P_{data}) \to 0$. However, suppose $P_G(x) \to 1, P_{data}(x) \to 0$, which means the generator generates images that do not look like those in the data, the corresponding penalty $KL(P_G||P_{data}) \to +\infty$. Therefore, this loss encourages generating replicated images that have low penalty rather than generating diverse data that could result in a high penalty, thus causing mode collapse.

WGAN solves the training difficulty and the mode collapse problem by using a modified loss function shown in Eq. 5, which essentially corresponds to minimizing the Wasserstein distance between the

generative distribution and the data distribution. The W-distance has nice property that even two distributions have little overlap, the W-distance still varies smoothly.

$$\min_G \max_D V(D,G) = \mathbb{E}_{\mathbf{x}\sim\mathbf{p_{data}}(\mathbf{x})}[D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim\mathbf{p_z}(\mathbf{z})}[1 - D(G(\mathbf{z}))], \tag{5}$$

Although WGAN avoids mode collapse, we found that the generated images still do not look very realistic, as there is no term in the objective function that encourages the synthetic data to look like the training data. This is encouraged implicitly in another type of deep generative model, VAE.

2.3. Variational autoencoder

The idea behind VAE is to use a generative neural network and a recognition neural network to solve the variational inference problem that maximizes the marginalized data likelihood. The generative network obtained at the end of this process can generate synthetic data that looks similar to the training data. Nonetheless, the exact data likelihood is not easily obtainable, thus VAE approximately maximizes the evidence lower bound (ELBO) by gradient ascent on the following objective function:

$$\mathcal{L}(\theta,\phi;\mathbf{x}) = \sum_{i=1}^{N} -KL(q_\phi(\mathbf{z}|\mathbf{x^{(i)}})||\mathbf{p}_\theta(\mathbf{z})) \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x^{(i)}})}[log\ p_\theta(\mathbf{x^{(i)}}|\mathbf{z})], \tag{6}$$

where θ is the parameter of the generative network, and ϕ is the parameter of the recognition network. $p_\theta(\mathbf{z})$ is the distribution of the latent variable $\mathbf{z}$, which is represented by a Gaussian distribution whose mean and covariance is obtained by passing a noise parameter ϵ through the generative network. $q_\phi(\mathbf{z}|\mathbf{x^{(i)}})$ is the approximate posterior distribution of the latent variable conditioned on the data instance, which is approximated as a Gaussian whose mean and covariance are obtained by passing the data instances through the recognition network. By minimizing the KL divergence between these two distributions, the model encourages the generated data to look similar to the training data. On the other hand, the second term in the objective function encourages the generative distribution $p_\theta(\mathbf{x^{(i)}}|\mathbf{z})$ to be as diffusive as possible. Therefore, the resulted synthetic image is blurred, which is not desirable.

2.4. VAE-GAN

One advantage of VAE models over GAN models is that it could map an input in the original dataset to latent factors and further to an image in the generator's approximation. However, the sample images generated by VAE are usually blurry and of lower quality compared to those from GAN models. To get the benefits of both, the work (Larsen et al., 2015) proposed an architecture shown in fig.1, on top of the original VAE models. The VAE-GAN model adds a discriminator on top of the generated image. The loss function for the discriminator is the same as the one in GAN. The loss for the decoder and the encoder have two components. The first component is the same as Eq.6 from VAE. The second component is

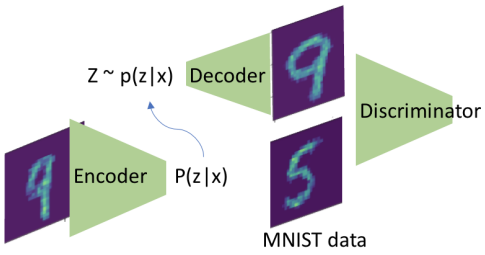

Figure 1. The architecture of the VAE-GAN model.

$$L_{GAN} = -\mathbb{E}_{\mathbf{z}\sim\mathbf{p_z}(\mathbf{z})}[logD(G(\mathbf{z}))]. \tag{7}$$

This component is minimized when the generator successfully fools the discriminator. With this architecture, the VAE-GAN successfully generates GAN-style images while preserving the functionality to map a sample image back to its latent variables.

3. Experiments

3.1. Settings

In this section, we experiment with GAN, WGAN, VAE and VAE-GAN to quantitatively analyze the performance of mitigating mode collapse based on MNIST dataset.

Firstly, we implemented all the generative adversarial models discussed so far with Tensorflow using the same fully connected neural networks for both the generator and the discriminator. In each model, the discriminative network is composed of five layers, where the size of input layer is 784, corresponding to the size of each 28×28 handwritten image, and the size of output layer is 10×10. The size of the layers between the output and the input are 392, 196, 98; the activation function in each layer between is ReLu function, and we use sigmoid function for the output layer. The generative network has exactly the same structure with the discriminative network, with the layers in the reverse order. The VAE model has analogous encoder net-

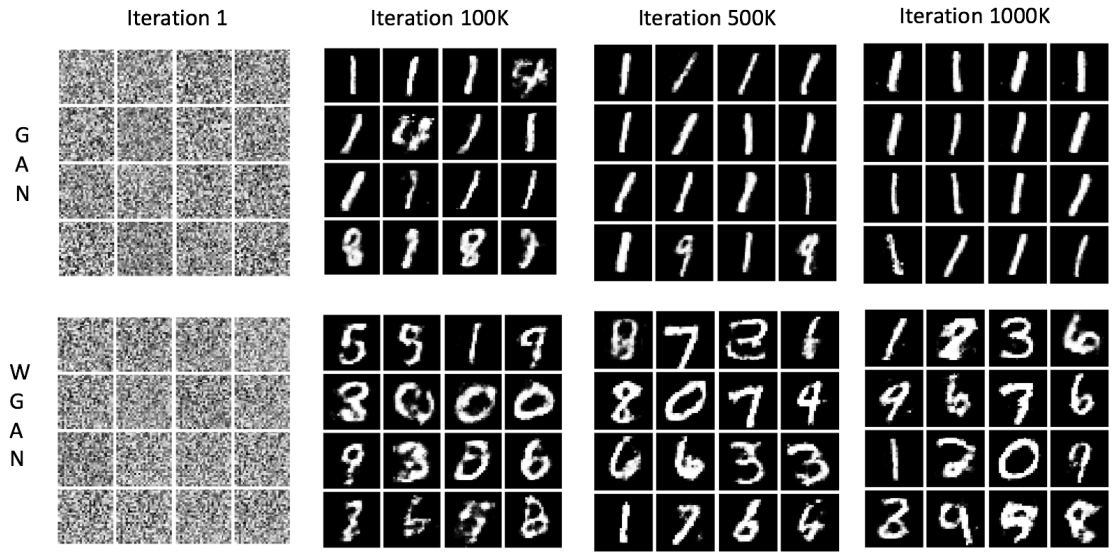

Figure 2. Generated images by GAN and WGAN models trained on MNIST after $1{,}100k$,$500k$,$1000k$ iterations.

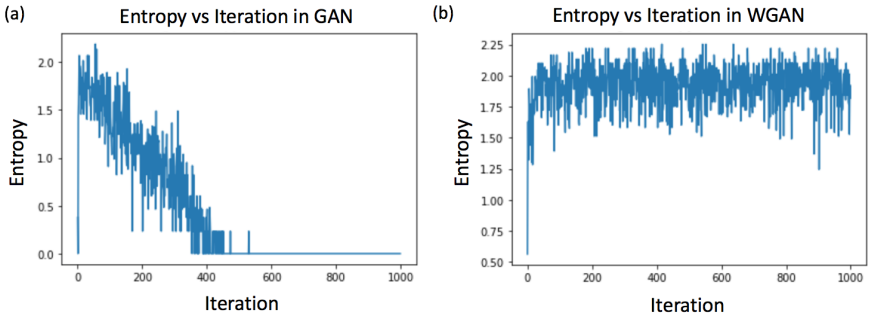

Figure 3. (a) The variation of entropy vs iterations in training process of GAN. (b) The variation of entropy vs iterations in training process of WGAN.

work and decoder network structure as the discriminative and generative network mentioned above. For VAE-GAN, in addition to the same encoder and decoder network as in VAE model, an additional discriminative network are added to the end of decoder and another input layer of MNIST dataset. Differences between each model are mainly the definitions of loss functions and methods to realize gradient descent and decrease loss. The training process of GAN is to decrease loss from discriminator and generator defined as Eq.2. The process of WGAN is to minimize the loss shown in Eq.5, which is implicitly minimizing the Wasserstein distance between the generative distribution and the data distribution. For VAE, the training process is to reduce the mean squared error between itself and the target and the KL divergence between the encoded latent variable and standard normal distribution, as defined in Eq.6. For VAE-GAN, the final loss function combined two parts, the loss generated in VAE part as Eq.6 and loss generated in GAN part as Eq.7, are finally used to generate the synthetic images.

3.2. Entropy

In order to provide a quantitative analysis, we used entropy of the synthetic data distribution to measure the severity of mode collapse. A classifier composed of 2-layer neural networks is firstly trained on the full MNIST dataset, which achieved 95.4% accuracy on the test dataset. Then we used the classifier to recognize the handwritten digits generated from GAN and WGAN and calculated the entropy of the generative distribution for each training iteration. Let p_i represent the probability of each digit i sampled from the generative network at each iter-

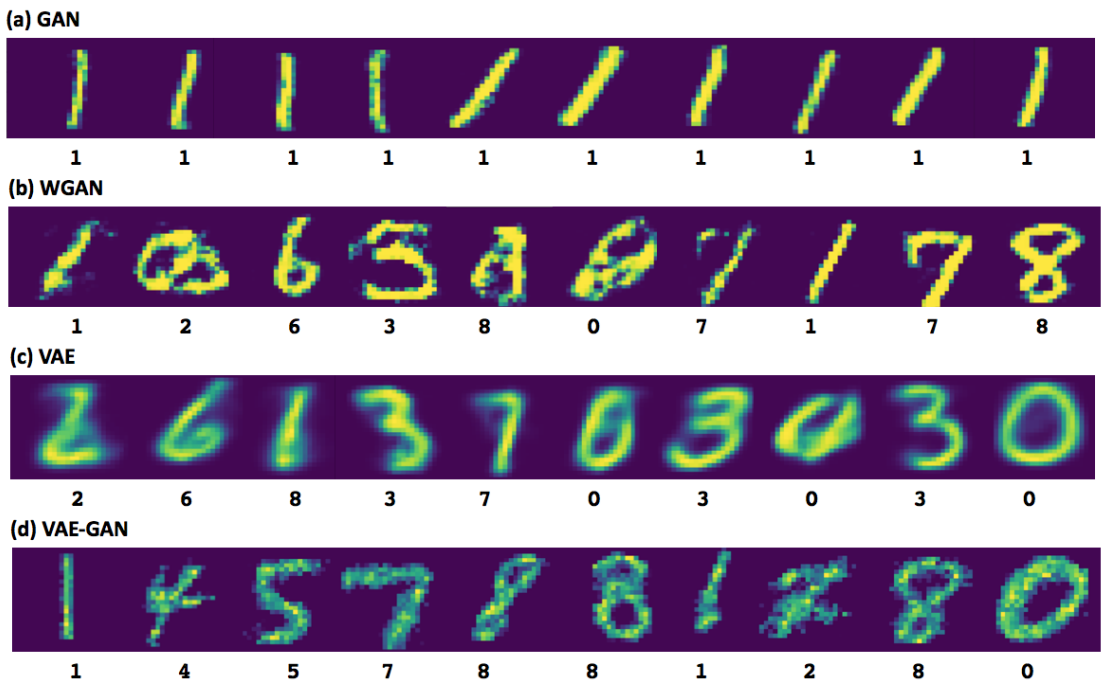

Figure 4. 10 handwritten images sampled from model (a)GAN (b)WGAN (c)VAE (d)VAE-GAN, the labels under each row of images are predicted by a well-trained two-layer fully-connected neural network classifier.

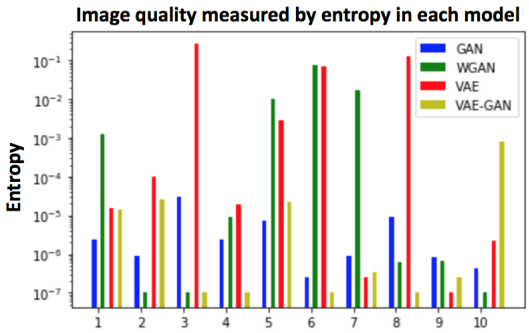

Figure 5. The single image entropy of each handwritten image in Fig.4 computed from the prediction probability of a pre-trained classifier. A low entropy means that the image quality is high as it suggests that the classifier recognizes the image easily. The x axis label represents the index of each image generated from different models.

ation. We use the empirical estimator

$$p_i = \frac{\text{number images classified as digit i}}{\text{number of images generated in total}}$$

Then the entropy of the generator can be estimated as

$$Entropy = -\sum_{i}^{10} p_i \times log(p_i)$$

When mode collapse happens, the entropy will keep decreasing. For example, we show the training process of GAN and WGAN, shown in Fig. 2. After 1000k iterations, the final images sampled from GAN only contain digit 1, which indicates that it is prone to mode collapse. In contrast, for WGAN, the final result is composed of various different digits, the mode collapse issue is mitigated significantly. The entropy shown in Fig. 3 of each iteration decreases rapidly, and reaches zero after 480k iterations, which represents complete mode collapse. In contrast, the entropy approaches a relatively steady value in the training process of WGAN.

Model	GAN	WGAN	VAE	VAE-GAN
Entropy	0.0	2.280	2.266	2.263

Table 1. Entropy of images generated from GAN, WGAN, VAE, VAE-GAN.

Furthermore, we also compared the entropy of the output in the last iteration from GAN, WGAN, VAE, VAE-GAN, shown in Table.1. Only the entropy of the output in GAN is zero, and the results in the other models are all around 2.27, which indicates that all the modes are preserved. In other words, no mode collapse.

3.3. Image quality

We further compare the image quality from each well-trained model, and sample 10 images from the output layer of GAN, WGAN, VAE and VAE-GAN in the last iteration, shown in Fig. 4. The

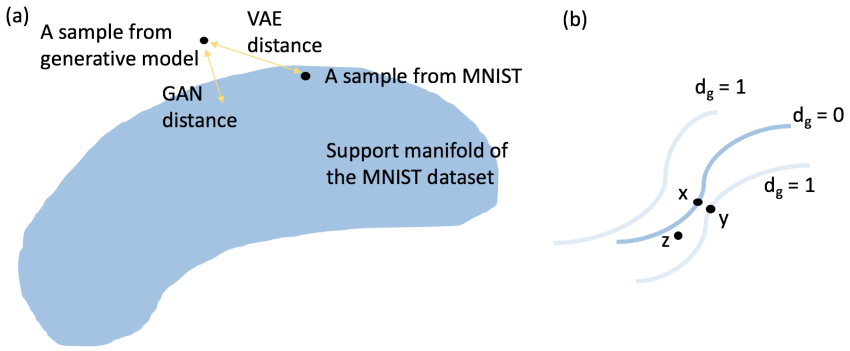

Figure 6. (a) The blue surface represents the low dimension structure of the MNIST data embedded in $\mathcal{R}^{28\times28}$. (b) Assume that the center blue line represent the support of the true distribution. The light colored lines are points in the high dimensional space that have distance 1 to the true distribution, measured by the discriminator in GAN model. Points x, y, z represent images in $\mathcal{R}^{28\times28}$.

labels under each row of images are predicted by a well-trained two-layer fully-connected neural network classifier. Images sampled from GAN only contain digit 1 due to mode collapse, from which other models do not suffer. However, the result of WGAN has a relatively low quality compared with the the original dataset: some images are hard to recognize with naked eyes. The results of VAE also do not achieve a satisfactory quality due to blurs. The images in VAE-GAN have the best performance in terms of image quality among those models.

In order to provide a quantitative study for image quality, we calculate the entropy of each single generated image. More specifically, for the $k-th$ image, we compute $\sum_i p_{i,k}\log(p_{i,k})$, where $p_{i,k}$ is the predicted probability that image k belongs to class i. The prediction is done using a pre-trained classifier. A high-quality output is likely to be classified with high confidence and hence will have a low entropy. The result is shown in Fig. 5. The entropy of VAE-GAN is lower than that of the other models, while the entropy of several images generated from VAE and WGAN are high due to their low quality. The entropy of GAN is low due to the fact that mode collapse enables the model to be easily optimized.

4. Our proposed model

In this section, we wish to provide an argument on why VAE models and GAN models generate very different styles of samples. Based on this argument, we explain how the VAE-GAN models can be potentially improved. We then implement this idea and test the model on MNIST dataset.

4.1. Motivation

From our experiment results in Section 3, we notice that each model has its own advantage over the rest even though they share the same network structure. The original GAN model proposed in (Goodfellow et al., 2014) produces images of the highest quality. However, since its loss function over-emphasizes its ability to fool the discriminator, the model only produces simple images and suffers from mode collapse. The WGAN model (Arjovsky et al., 2017) generates images that resemble digits with various shapes and hence solves the mode collapsing problem. However, some samples clearly belong to none of the 10 classes from a human's perspective. VAE models (Kingma & Welling, 2013), on the other hand, produces pretty images without mode collapsing. However, these images are blurry and can be easily distinguished from samples from the original dataset. The VAE-GAN model (Larsen et al., 2015) attempts to get the best of both, but ends up producing samples similar to the ones from WGAN. However, it has the same advantage of getting latent vector distribution from an input image as the VAE models do.

We propose the following explanation for the behaviors described above. Assume that the true distribution of MNIST dataset lies on a low dimensional manifold as shown in Fig. 6 and an instance is sampled from the deep generative model. We think of the loss functions for VAE models and GAN models as distance functions. The VAE distance measures its distance to the original input image. In particular, when the conditional distribution is Gaussian in VAE models, the decoder's loss is proportional to Euclidean distance. As a result, original VAE models generate images close to the input data points in Euclidean distance. Hence, these images have similar shapes as the true data

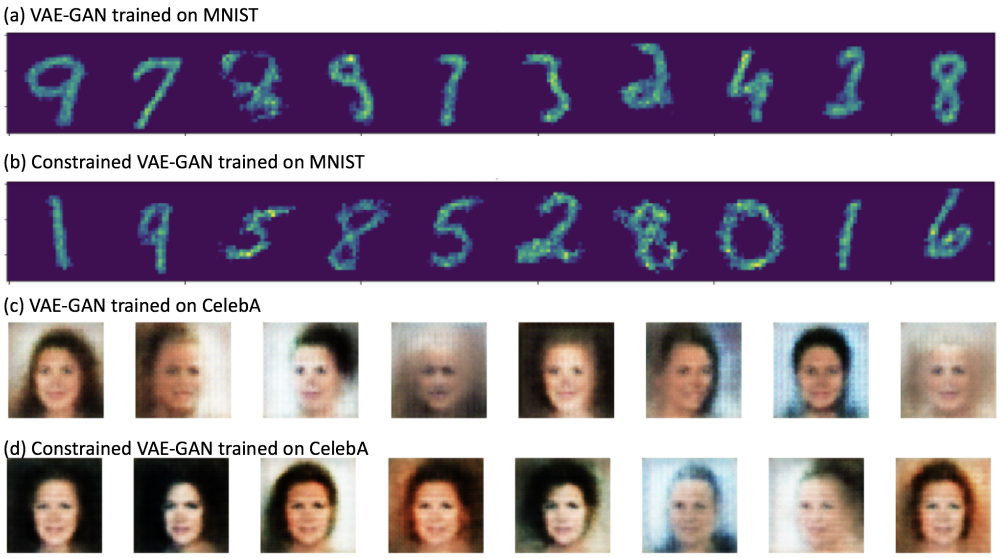

Figure 7. (a) Images sampled from VAE-GAN after training on the MNIST dataset. (b) Images sampled from our proposed model after training on the MNIST dataset. (c) Generated images from VAE-GAN after training 25 epochs on the CelebA dataset. (d) Generated images from constrained VAE-GAN after training 25 epochs on the CelebA dataset.

but admit small deviations in pixel values and are blurry. On the other hand, GAN distance measures the fake point's distance to the manifold, and equals zero as long as the point is on the manifold. Therefore, the GAN models produce points that are very close to the true distribution's support. However, without regularization based on the true data points, these generated images may not span the entire manifold. Furthermore, the discriminator learned may not properly compute the distance due to the difficulty of non-convex optimization.
As explained in section 2.4, the VAE-GAN model tries to optimize both loss functions at the same time. The loss for the decoder and encoder can be written into two components as follows,

$$L_{vae-gan}(x) = L_{gan}(x) + L_{vae}$$

Yet, the result suggests that the GAN loss might dominate the other since it strengthens over iterations. We wish to design a model that allows VAE loss to take effect. Hence, we propose a constrained loss

$$\min_x L_{vae}(x),\ s.t.\ L_{gan}(x) \leq d$$

The justification for this model is illustrated in Fig. 6. If point x is the input image. Then point z has a lower loss in the original VAE-GAN model due to its low GAN loss, but y would have a lower loss in our constrained model, since it is in the feasible region and has a lower VAE loss. Solving this constrained problem allows the model to focus on imitating the shape of the original data samples as long as the image quality can almost fool the discriminator. In order to solve this constrained problem, we rewrite it in its Lagrangian form

$$\mathcal{L}(\lambda, x) = L_{vae}(x) + \lambda(L_{gan}(x) - d)\ s.t.\ \lambda \geq 0$$

By KKT conditions, we can find local optima by solving $\mathcal{L}(\lambda^*, x)$, where λ^* that maximize $\mathcal{L}(\lambda, x)$. It is straightforward to check that $\lambda^* = 0$ when $L_{gan}(x) \leq d$, and $\lambda^* = \infty$ when $L_{gan}(x) > d$. As an approximation, we solve the following problem for a fixed $\lambda > 0$,

$$\min_x [L_{vae}(x) + \lambda \max\{L_{gan}(x) - d, 0\}]$$

The sub-gradient for this loss function can be easily computed and we can train the neural network with back-propagation as usual.

4.2. Experiment

The only difference between our model and the VAE-GAN model is that it uses a nonlinear combination of the loss from the VAE model and the loss from the discriminator. Hence, to control all other factors, we only changed the loss function in our original code for training VAE-GAN models, and ran stochastic gradient descent algorithm for the same number of iterations.

First, we run both models on MNIST dataset. Both models use the same 5-layer fully connected neural networks as introduced in Section 3. The sampled random images are shown in fig. 7(a)(b). Neither of

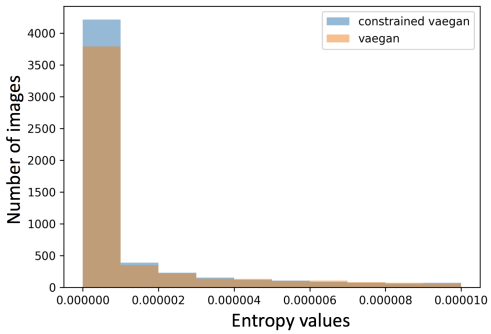

Figure 8. We sample 10k images from both constrained vaegan and vaegan models. We then use a pretrained classifier to classify each single image. For every image, we can compute its own entropy using predicted probability for each class. A high quality image can be recognized easily and hence should have low entropy.

the two models has a dominating performance, but our proposed one seems to have more stable image quality. This is verified in fig. 8. The entropy here is defined the same way as the entropy in fig. 5. Low entropy is associated with high image quality. We notice that our proposed algorithm has higher concentration of low entropy images compared to the original VAEGAN model.

Then we tested both models on the CelebA dataset with convolutional neural networks. Our network architecture has three convolution layers and is the same as the original paper in (Larsen et al., 2015). The original code runs the training process for about 50 epochs, but due to our limited computation resource, we have to terminate the process at epoch 25 after training for an entire week. Some preliminary results are shown in fig. 7(c)(d). We are not able to draw any interesting conclusion since the network has not converged.

5. Discussion

There are a few problems unsolved due to our limited time and computation resource. First, MNIST is a dataset with a simple structure. Therefore, the conclusions we draw based on MNIST experiments may not generalize to more complicated data. Even though we attempt to train some convolutional neural networks(CNN) on the CelebA data set, the lack of GPU access forces us to terminate the experiment before it finishes training. Therefore, it would be interesting to check if our proposed model can have greater improvement when the manifold in high dimensional space is more complicated. Second, the quality of the images generated in our experiments are relatively low compared to results in more recent literatures. This results from the disadvantage of fully connected neural network compared with CNN in learning image structures. We suspect that our WGAN model generates low quality images because the network is not powerful enough. Again, we do not have enough resources to conduct experiments that require convolution operations.

In the results shown, we tried to make fair comparison by using the same network architecture in all of our models. We trained VAE-GAN and constrained VAE-GAN with the same parameters for the same number of iterations. Other models are trained until the image quality stabilizes. From these results, we may conclude that our explanation in in section 4 aligns with the experiments in Section 3. However, the experiment comparing VAE-GAN and constrained VAE-GAN shows little difference, and more efforts are needed before we could get stronger evidence to support our claim.

References

Andrieu, Christophe, De Freitas, Nando, Doucet, Arnaud, and Jordan, Michael I. An introduction to mcmc for machine learning. *Machine learning*, 50(1-2):5–43, 2003.

Arjovsky, Martin, Chintala, Soumith, and Bottou, Léon. Wasserstein gan. *arXiv preprint arXiv:1701.07875*, 2017.

Blei, David M, Ng, Andrew Y, and Jordan, Michael I. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

Goodfellow, Ian, Pouget-Abadie, Jean, Mirza, Mehdi, Xu, Bing, Warde-Farley, David, Ozair, Sherjil, Courville, Aaron, and Bengio, Yoshua. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Kingma, Diederik P and Welling, Max. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Larsen, Anders Boesen Lindbo, Sønderby, Søren Kaae, Larochelle, Hugo, and Winther, Ole. Autoencoding beyond pixels using a learned similarity metric. *arXiv preprint arXiv:1512.09300*, 2015.

Loaiza-Ganem, Gabriel, Gao, Yuanjun, and Cunningham, John P. Maximum entropy flow networks. *arXiv preprint arXiv:1701.03504*, 2017.

Mirza, Mehdi and Osindero, Simon. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

Radford, Alec, Metz, Luke, and Chintala, Soumith. Unsupervised representation learning with deep

convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Ranganath, Rajesh, Tran, Dustin, and Blei, David. Hierarchical variational models. In *International Conference on Machine Learning*, pp. 324–333, 2016.

Rezende, Danilo Jimenez and Mohamed, Shakir. Variational inference with normalizing flows. *arXiv preprint arXiv:1505.05770*, 2015.

Wainwright, Martin J, Jordan, Michael I, et al. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1–2):1–305, 2008.