Generative Models

Shenao Zhang Tingyu Zhang Haidong Zhi Xueren Ge Xinghao Chen

1. Summary

In this work, we are interested in one core question in machine learning: approximating the intractable distributions. Directly adopting maximum likelihood estimation (MLE) or Bayesian parameter estimation often fail to fit the complex distributions, due to the high-dimension and multi-modality. Thus, special designs are needed.

Solving the above question is important since it can benefit many related research fields. In computer vision, the datasets are subject to potentially complex distributions, and the costs to obtain data that lie on the assumed distribution are usually expensive. It will be exciting to see synthesis samples that are amenable to the desired distribution, i.e., the generated samples similar to ones in the original datasets. Other applications including dynamic framework (Sutton, 1990) and CIA framework (Levine, 2018) in reinforcement learning (RL), and estimating mutual information in information theory (Belghazi et al., 2018). This work focuses on generating synthesized images in experiments, and surveys the two most popular frameworks: generative adversarial models and variational inference based methods.

The main task is to design a latent space or a low–dimensional manifold that the data is assumed to live on, as well as the projection function that maximizes the likelihood of true data distribution. In other words, it is equivalent to minimize the divergence between the true distribution and the approximated distribution. Drawing samples from the approximated distributions should not only be able to reconstruct the training data, but also should generate unseen synthesis data that is desired. Briefly speaking, the GAN framework adopts a generator network as the projection function, with input generated from the prior. The discriminator is then used as a metric of similarity of the two distributions (or each sample). The other VAE framework adopts latent variables as the assumed code for reconstructing the data, which also possesses a generator network to be trained with the cross-entropy reconstruction loss.

Our contributions include four folds. Firstly, we analyze the above two frameworks and provide derivation and insights behind them. Secondly, we make analyses of the drawbacks of these generative models, and review the improvements over them. Thirdly, we provide our findings that both of them can be formulated as a divergence minimization problem. Based on this finding, we discover the connection between these two frameworks and propose potential improvements. Lastly, we conduct experiments to compare the performances and make further analyses, including ablation studies and visualizations.

2. Generative Adversarial Framework

2.1. Overview

2.1.1. The Construction of the GAN Model

We are to denote a general handwritten digit image as a vector $\mathbf{X}$. Given the whole MNIST dataset, we assume that the true distribution of handwritten digit images $P(\mathbf{X})$ can be approximated by the dataset. Our goal is to generate images that are subject to the distribution $P(\mathbf{X})$.

Now we are to build a generative model G with its output subject to the distribution $P_G(\mathbf{X};\theta_G)$, and its input as a random vector $\mathbf{z}$. The distribution P_G is controlled by the parameter vector θ which can be trained. Extracting the images from the MNIST dataset $\{\mathbf{X}_i\}_{i=1}^N$, we can compute the likelihood of P_G as

$$L = \prod_{i=1}^{N} P_G(\mathbf{X}_i;\theta_G) \tag{1}$$

In order to let the generator G output images that look like real handwritten digits, we can maximize the likelihood. That is, we can find ${\theta_G}^*$ such that

$$\begin{aligned}
{\theta_G}^* &= \arg\max_{\theta_G} \sum_{i=1}^{N} \log P_G(\mathbf{X}_i;\theta_G) \\
&= \arg\max_{\theta_G} \int_{\mathbf{X}} P(\mathbf{X}) \left(\log \frac{P_G(\mathbf{X}_i;\theta_G)}{P(\mathbf{X})} \right) \mathrm{d}\mathbf{X} \\
&= \arg\min_{\theta_G} \int_{\mathbf{X}} P(\mathbf{X}) \left(\log \frac{P(\mathbf{X})}{P_G(\mathbf{X}_i;\theta_G)} \right) \mathrm{d}\mathbf{X} \\
&\triangleq \arg\min_{\theta_G} \mathrm{KL}\left(P(\mathbf{X}) \,||\, P_G(\mathbf{X}_i;\theta_G)\right)
\end{aligned} \tag{2}$$

To numerically find ${\theta_G}^*$, we introduce another discriminator model $D(\mathbf{X};\theta_D)$ which takes a handwritten image $\mathbf{X}$ as input, and outputs a scalar. The scalar $D(\mathbf{X})$ is a probability expected to discriminate the source of inputs. Meanwhile,

G is trained to maximize the probability that D thinks $G(\mathbf{z})$, which is actually generated by G, is picked from the dataset. To fulfill the goal, we introduce the value function as

$$V(G,D) = \mathbb{E}_{\mathbf{X}}[\log D(\mathbf{X})] + \mathbb{E}_{\mathbf{X}\sim P_G(\mathbf{X})}[\log(1 - D(\mathbf{X}))] \tag{3}$$

or furtherly,

$$V(G,D) = \mathbb{E}_{\mathbf{X}}[\log D(\mathbf{X})] + \mathbb{E}_{\mathbf{z}\sim P_{\mathbf{z}}(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))] \tag{4}$$

and we are to find

$$\theta_G^*, \theta_D^* = \arg\min_{\theta_G} \arg\max_{\theta_D} V(G,D) \tag{5}$$

2.1.2. The optimization of G and D

First, we are to fix G and consider the optimal D for the given G. That is, we are to maximize $V(G,D)$.

$$\begin{aligned} V(G,D) &= \int_{\mathbf{X}} P(\mathbf{X}) \log(D(\mathbf{X}))\, \mathrm{d}\mathbf{X} \\ &+ \int_{\mathbf{z}} P_{\mathbf{z}}(\mathbf{z}) \log(1 - D(G(\mathbf{z})))\, \mathrm{d}\mathbf{z} \\ &= \int_{\mathbf{X}} (P(\mathbf{X}) \log(D(\mathbf{X})) + P_G(\mathbf{X}) \log(1 - D(\mathbf{X})))\, \mathrm{d}\mathbf{X} \end{aligned} \tag{6}$$

By requiring $\frac{\partial V(G,D)}{\partial D} = 0$, since we do not need to include the integral operation when finding the optimal D, we finally have

$$D_G^*(\mathbf{X}) = \frac{P(\mathbf{X})}{P_G(\mathbf{X}) + P(\mathbf{X})} \tag{7}$$

Now we are to find the optimal G.

$$\begin{aligned} V(G, D_G^*) &= \mathbb{E}_{\mathbf{X}\sim \text{data}}\left[\log \frac{P(\mathbf{X})}{P_G(\mathbf{X}) + P(\mathbf{X})}\right] \\ &+ \mathbb{E}_{\mathbf{X}\sim P_G(\mathbf{X})}\left[\log \frac{P_G(\mathbf{X})}{P(\mathbf{X}) + P_G(\mathbf{X})}\right] \\ &= \int_{\mathbf{X}} P(\mathbf{X}) \log \frac{P(\mathbf{X})}{P_G(\mathbf{X}) + P(\mathbf{X})} \mathrm{d}\mathbf{X} \\ &+ \int_{\mathbf{X}} P_G(\mathbf{X}) \log \frac{P_G(\mathbf{X})}{P_G(\mathbf{X}) + P(\mathbf{X})} \mathrm{d}\mathbf{X} \\ &= -2\log(2) + \int_{\mathbf{X}} P(\mathbf{X}) \log \frac{2P(\mathbf{X})}{P_G(\mathbf{X}) + P(\mathbf{X})} \mathrm{d}\mathbf{X} \\ &+ \int_{\mathbf{X}} P_G(\mathbf{X}) \log \frac{2P_G(\mathbf{X})}{P_G(\mathbf{X}) + P(\mathbf{X})} \mathrm{d}\mathbf{X} \\ &\triangleq -2\log(2) + \mathrm{KL}\left(P(\mathbf{X}) \,||\, \frac{P(\mathbf{X}) + P_G(\mathbf{X})}{2}\right) \\ &+ \mathrm{KL}\left(P_G(X) \,||\, \frac{P(\mathbf{X}) + P_G(\mathbf{X})}{2}\right) \\ &\triangleq -2\log(2) + 2\mathrm{JS}(P(\mathbf{X}) \,||\, P_G(\mathbf{X})) \end{aligned} \tag{8}$$

Here $\mathrm{JS}(\cdot)$ is the Jensen-Shannon divergence function. Since JS divergence is zero when and only when the two distributions are equal, we have the global optimal G as

$$P_G^*(\mathbf{X}) = P(\mathbf{X}) \tag{9}$$

2.2. Drawbacks of GAN and the Improvements

In GANs, we did not make full use of the data. For example, some datasets have additional information, particularly a class label. It seems to be more efficient if we make efforts to take advantage of additional information. Besides, there may be the hidden trouble in GANs of generating a random image from the noise domain $\mathbf{z}$. Undoubtedly, there is a relationship between samples in the $\mathbf{z}$ domain to the generated images, which is however complicated and difficult to map.

To overcome the disadvantages, conditional GAN (CGAN) (Mirza & Osindero, 2014) was proposed. The major enhancement of CGAN is that both generator and discriminator are conditioned on class label. We can perform the conditioning by feeding class label into the generator and discriminator as an additional input layer. Through conditioning, images of a given class can be generated. Accordingly, the optimization function is revised as follows,

$$\begin{aligned} \min_G \max_D V(D,G) = &\mathbb{E}_{x\sim p_{\text{data}}(x)}[\log D(x|y)] \\ &+\mathbb{E}_{z\sim p_z(z)}[\log(1 - D(G(z|y)))] \end{aligned} \tag{10}$$

Another popular improvements over GAN in vision models is deep convolutional GAN (DCGAN) (Radford et al., 2015). DCGAN is designed for visual inputs by replacing pooling layers with strided convolutions at discriminator and fractional-strided convolutions at generator. Besides, BN is added with fully connected layer removed to achieve deep architectures. Other slight designs also exist, including using LeackReLU at discriminator.

2.3. General Divergences in GAN

Despite the insightful adversarial game proposed by Ian et al. (Goodfellow et al., 2014), which is formulated as a minimax optimization problem, we now provide our understanding of how GAN can be derived from a general divergence perspective.

Considering the KL divergence between two distributions P and Q:

$$D_{KL}(P||Q) = \int p(x) \log \frac{p(x)}{q(x)} dx \tag{11}$$

Divergence minimization problem aims to approximate the true probability density P with the proposed probability Q. However, in most cases, calculating the above divergence precisely with Eq. 11 is tractable only for discrete variables or low-dimensional data. Previous works have proposed

the dual representation of the above KL divergence, which is called the Donsker-Varadhan representation (Donsker & Varadhan, 1975):

$$\begin{aligned} D_{KL}(P||Q) &= \sup_{T:\Omega\to R} \mathbb{E}_P[T] - \log \mathbb{E}_Q[e^T] \\ &\geq \sup_{T\in F} \mathbb{E}_P[T] - \log \mathbb{E}_Q[e^T], \end{aligned} \tag{12}$$

where F is any class of functions $T : \Omega \to R$ that satisfying the integrability constraints of the theorem, and the inequality holds due to the compression lemma in the PAC-Bayes literature (Banerjee, 2006).

The lower bound in Eq. 12 provides a tractable method for KL divergence estimation and other related information metrics, including mutual information. For functions of f_θ that estimate the lower bound in the parameter space $\theta \in \Theta$, the optimization can be performed in a single iteration expectation-maximization manner. Specifically, in each optimization iteration, the lower bound is first evaluated and then θ is updated by minimizing Eq. 12.

Next, we describe how the above representation leads to the derivation of GAN. Instead of KL divergence, we now consider more general divergences, Ali-Silvey divergence (also known as f-divergence).

$$D_f(P||Q) = \int q(x) f(\frac{p(x)}{q(x)}) dx \tag{13}$$

It is shown that similar with Eq. 12, the dual representation of such f-divergence is:

$$\begin{aligned} D_f(P||Q) &= \int q(x) \sup_{t\in dom_{f*}} t\frac{p(x)}{q(x)} - f^*(x) dx \\ &\geq \sup_{T\in F} \mathbb{E}_P[T] - \log \mathbb{E}_Q[f^*(T(x))], \end{aligned} \tag{14}$$

where f^* is Fenchel conjugate (Hiriart-Urruty & Lemaréchal, 2012) of f, defined as $f^*(t) = \sup_{t\in dom_{f*}}(ut - f(u))$.

Recall that our task is to minimize the divergence between the proposed Q with the true P, which is aligned with the task of GAN to approximate the true distribution (of the dataset) with the generated distribution. So if we parameterize Q with parameter θ, Q_θ can be regarded as the generative model in GAN. After evaluating the lower bound in Eq. 14, function T is optimized to minimize the variational bound.

Thus, a general and close connection between GAN and divergences can be constructed. GAN can be formulated as a special case of f-divergence, with the following realization:

$$\begin{aligned} f(u) &= u\log u - (u+1)\log(u+1) \\ T^*(x) &= \log \frac{p(x)}{p(x)+q(x)} \end{aligned} \tag{15}$$

3. Variational Inference Framework

3.1. Overview

We now focus on another elegant variational inference framework for generation models. We first give the background of why the latent variable is beneficial for generative models. In some machine learning tasks, e.g., the auto–regression (AR) model, all random variables are observed. Whereas in the latent variable models, some random variables are assumed hidden and can not be observed:

$$z \overset{p_\theta(x|z)}{\longrightarrow} x \text{ where } p_\theta(x) = \textstyle\sum_z p_z(z) p_\theta(x|z)$$

In the model above, z is unobservable. The drawback of AR models is their slow sampling time due to data dependency. On the contrary, latent variable models are relatively simple to process and can describe the original data set with fewer variables. We can make part of observation space independent and make the space conditioned on latent variables. Faster sampling is achieved by exploiting statistical patterns. Overall, working with latent space is more efficient than directly working with pixels or data.

However, from the model, we generally do not know the form of latent variable z and how they interact with observations x. They could be in any form, such as Bernoulli variables or Normal variables. The training objective is to maximize the likelihood over all choices of data.

$$\max_\theta \sum_i log\, p_\theta(x^{(i)}) = \sum_i \log \sum_z p_z(z) p_\theta(x^{(i)}|z) \tag{16}$$

The above objective is tractable if z is finite, but most of the time, z has impractical numbers of values and an approximation is needed. Directly sampling z from assumed distribution and run the SGD on the object is still impractical. For one specific x, most sampling modes z are wasted since they are likely not to contribute to the objective. Either $p_z(z)p_\theta(x^{(i)}|z)$ or the gradient may go to zero, especially when there are high dimensional data. It is unlikely to get a meaningful full choice of z into the objective.

Let's define the expectation $\mathbb{E}_{z\sim p_z(z)}[f(z)]$ as follow and apply importance sampling algorithm.

$$\begin{aligned} \mathbb{E}_{z\sim p_z(z)}\,[f(z)] &= \sum_z p_z(z) f(z) \\ &= \mathbb{E}_{z\sim q(z)}\,[\frac{p_z(z)}{q(z)} f(z)] \end{aligned} \tag{17}$$

The method of importance sampling is an unbiased estimation and allows us to sample from $q(z)$ to compute expectation with respect to $p_z(z)$. Ideally, a large sample size would guarantee the estimation to be the true expected value. Nevertheless, fewer samples are needed because samples from $p_z(z)$ are not informative and most of them will have zero contribution. Now that the training objective can be

rewritten as follow,

$$\sum_i log \frac{p_z(z^{(i)})}{q(z^{(i)})} f(z) p_\theta(x^{(i)}|z^{(i)}) \tag{18}$$

Latent variable z is sampled from q distribution. The objective tells that if sampling under $q(z)$ is likely but under $p_z(z)$ is unlikely, the contribution should scale down. To make samples from q compatible with $x^{(i)}$, $q(z)$ is set as follow,

$$q(z) = p_\theta(z|x^{(i)}) = \frac{p_\theta(x^{(i)}|z)p_z(z)}{p_\theta(x^{(i)})}$$

Finding $q(z)$ tells which z is likely given x, but the p_θ term in the denominator is still complicated to get.

Since we can not approximate $p_\theta(z|x(i))$ directly, a parameterized distribution q is proposed. The goal is to find the best parameter setting that is the closest to $p_\theta(z|x^{(i)})$. One common assumption is the normal distribution. We are tying to find μ and σ as close as possible to $p_\theta(z|x^{(i)})$.
Recall that the generative models aim to fit the desired distribution (e.g., a dataset) $p(x)$ with another tractable density function. One common way is to adopt MLE. However, the latent variable makes the whole optimization intractable. To make it tractable, we first consider the following divergence.

$$\begin{aligned} & D_{KL}(q(z)\|p(z|x)) \\ =&\mathbb{E}_{z\sim q(z)} \log(\frac{q(z)}{p(z|x)}) \\ =&\mathbb{E}_{z\sim q(z)} \left[\log q(z) - \log p_z(z) - \log p(x|z)\right] + \log p(x) \end{aligned} \tag{19}$$

The above relationship between KL divergence and $p(x)$ gives the following objective:

$$\begin{aligned} \log p(x_i) =& D_{KL}(q(z_i)\|p(z|x_i)) \\ &+\mathbb{E}_{z\sim q(z_i)}[-\log q(z_i) + \log p_z(z) + \log p(x_i|z)] \\ &\geq \mathbb{E}_{z\sim q(z_i)} \left[-\log q(z_i) + \log p_z(z) + \log p(x_i|z)\right] \end{aligned} \tag{20}$$

The above inequality holds since KL divergence is non-negative. It is worth noting that the above inequality is also obtained by adopting Jensen's inequality in (Kingma & Welling, 2013), where the equality holds when the divergence equals zero. Maximizing the Evidence Lower Bound (ELBO) in Eq. 20 has the following approximation in VAE. First, denote the ELBO as $L_i(p, q_i)$. To overcome the intractably large parameter space that grows up with the size of the dataset, amortized inference is adopted. Specifically, instead of finding the proper distributions of each $q(z_i)$, an amortized inference (recognition) network $q_\phi(z|x_i)$ is fitted to generate latent variables of each sample in terms of auxiliary variables with fixed distributions, such that the samples from the recognition model are a deterministic function of the inputs and auxiliary variables. Also, leveraging the more precise measurements of KL divergence between distributions, like Gaussian distributions, the ELBO can be further simplified as:

$$\begin{aligned} L_i(p, q_i) =& \mathbb{E}_{z\sim q_\phi(z|x_i)} \log p_\theta(x_i|z) + \mathbb{E}_{z\sim q_\phi(z|x_i)} \log p(z) \\ &+H(q_\phi(z|x_i)) \\ =& \mathbb{E}_{z\sim q_\phi(z|x_i)} \log p_\theta(x_i|z) - D_{KL}(q_\phi(z|x_i)||p(z)) \end{aligned} \tag{21}$$

The above objective function indicts the designing of variational inference based auto-encoder. A parameterized encoder q_ϕ and a decoder p_θ are updated to optimize Eq. 21.

3.2. Drawbacks of VAE and the Improvements

In the last part, we review the background of variational inference (VI) and its corresponding generative models. The recent advances in VI can be categorized into three lines, including improvements over vanilla VAE (Burda et al., 2015; Larsen et al., 2016), large scale VI methods, and tighter variational bounds. In this section, we focus on the first category and review some related works. Then we focus on the divergences and variational bounds in the next section.
The objective function Eq. 21 makes intuitive sense to approximate the log-likelihood of the true distribution $\log p(x)$ while minimizing the KL divergence $D_{KL}(q_\phi(z|x_i)||p(z))$ to encourage the model to learn a representation where posterior inference is easy to approximate. It penalizes the approximated posterior samples which fail to explain the true observation. However, such constraints are too strong for some distributions. When a small fraction of measures sampled from the recognition network which possess high posterior probability may already be sufficient for accurate inference. Vanilla VAE, by making the simplified assumption, will fail to obtain a reliable model.
To make the training process more stable for more complex distribution of interest, importance weighted AE (IWAE) (Burda et al., 2015) is proposed. IWAE is a generative model with the same architecture as the VAE, but uses a strictly tighter log-likelihood lower bound derived from importance weighting. The inference network in IWAE uses multiple samples to approximate the posterior, which increases the flexibility to model complex posteriors that VAE does not have. Instead of having the simplified objective in Eq. 21 that KL divergence is directly calculated, IWAE needs to sample multiple z^k and approximate the expectation following the original objective. The gradient for IWAE is as

follows:

$$\begin{aligned}\nabla_\theta L_i(p,q_i) =& \frac{1}{K}\nabla_\theta \sum_{k=1}^{K} \log p_\theta(x_i|z^k) \\ \nabla_\phi L_i(p,q_i) =& \frac{1}{K}\nabla_\phi \sum_{k=1}^{K} (\log q_\phi(z^k)\ [\log\ p_z(z^k) \\ &+ \log p_\theta(x_i|z^k) - \log\ q_\phi\ (z^k|x_i)]\end{aligned} \tag{22}$$

We show in experiments section how such improvements can boost the performance under different settings, as well as the impacts of different class labels.

Another insightful improvement is to combine the advantages of the GAN and VAE models (Larsen et al., 2016). In VAE model, the maximum likelihood estimation is the primary objective function. In other words, element-wise reconstruction error between the true sample and generated sample is adopted when implementation, e.g., in vision tasks. However, computing and back-propagate error in element-wise manner often cause low efficiency when training and take longer time to converge. Representation learning, on the contrary, is a more suitable way to measure the distances between representations. Benefiting from GAN, the discriminator is adopted to distinguish the samples between the real and generated, measuring the reconstruction error with high-level features extracted.

3.3. Divergences in VAE

The key estimation problem that most of the variational inference generative models solve is to minimize the divergence between posterior while keeping the interpretability to the true data distribution, as can be observed in Eq. 21. Similar to the GAN framework, there are also existing works that provide a unified divergence minimization problem for the VI framework. Despite of the common KL divergence in VI (Jordan et al., 1999) and its application on VAE, Renyi's α-divergence VI (Li & Turner, 2016) unified a number of works, including the previously introduced VAE and IWAE. Robust β-divergence is also proposed by (Futami et al., 2018) to overcome the problem of outlier data points, and the misspecified transition model in the reinforcement learning literature. The measures with low probabilities are assigned low importance weights, penalizing their impact on the training process. Wasserstein VI (Ambrogioni et al., 2018) is another insightful work based on optimal transport theory. Recent work by Wan et al. (Wan et al., 2020) proposed a more elegant f-divergence framework that provides further insight on VI from any f-divergence's perspective.

4. Unifying GAN and VAE

As we have discussed in the GAN and VAE section of how they can be derived from the divergence perspective, we now further try to give a unified model of GAN and VAE, which also motivates potential improvements of generative models.

Vanilla GAN (Goodfellow et al., 2014) provides an interpretation from a JS divergence perspective, and we extend the contents to general f-divergences. Vanilla VAE (Kingma & Welling, 2013) also naturally handles with KL divergences, though recent works have shown the interpretations from different divergences. The correlations between these two frameworks are thus more obvious from a divergence perspective.

It is worth noting that VAE naturally deals with reverse (KL) divergences at Eq. 21. The equation is non-trivial since it leads to a mode–covering behaviour, as opposed to the divergence of GAN, the mode–seeking behavior. VAE often results in blurred, implausible samples, while GAN is analyzed to suffer from mode collapse.

If the two directions of divergences are combined into a single divergence, both of the disadvantages will be alleviated. New generative models are expected to approximate the data distribution better. We believe it is a fruitful research field.

5. Experiments

As discussed at summary, although applications of generative models are wide, we only focus on simple vision tasks for experiments. We re-implement multiple generative models, including GAN, VAE and their variants mainly on MNIST dataset. We conduct experiments to show the performance of each method. The results are also analyzed and compared between different methods. To show how intuitively each method works, we design and perform visualizations. Ablation studies are also provided to give further insights.

5.1. Experiments on GAN models

5.1.1. Primitive GAN

We trained a convolutional GAN model for 100 epochs on MNIST dataset. The convolutional layers at generator are of 64, 32 and 1 output dimensions, respectively, and all with kernel size of 4. The discriminator is made up of 2 convolutional layers with 64 and 128 output dimensions, respectively, and a final dense layer of size 1. BN and ReLU functions are used in all the layers.

Figure 1 and 5 shows the intermediate and final results. These results justify the potential of GAN models.

5.1.2. CGAN

Here, we uses MNIST data to do the experiments, in our model, we set different epoches to see different effects. Running the example creates a figure with a plot of 100

(a) Output after 1 epoch (b) Output after 25 epochs (c) Output after 75 epochs

Figure 1. GAN intermediate generating samples.

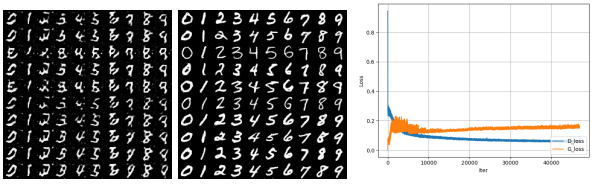


Figure 2. Training results of CGAN. **Left:** 1 epoch. **Middle:** 50 epoch. **Right:** Training loss curve.

images from the MNIST training dataset, arranged in a 10×10 grid. In this case, Most digits look quite plausible and could have come from the MNIST dataset. They are not perfect, however, as there are some "2" and "3" with blurred trail that look like a mess.

When setting the epoch parameter as 50, we can clearly see that the output images is much better. Running the example loads the saved conditional GAN model and uses it to generate 100 items of handwritten images.

Besides, we also have plotted the trend of loss function. The loss of generator grows with the increase of iterations. The variances is also gradually becomes smaller.

5.1.3. DCGAN

By inputting some noise, our DCGAN model generated different images which denote different number, at figure 6.

5.2. Experiments on VAE models

We show the generation manifold of VAE at Fig. 4.

5.2.1. Reconstruction Error in VAE

We first analyze the performance of VAE with element-wise reconstruction error and VAE with discriminator based representation error (Larsen et al., 2016).

Since the data in MNIST is assumed to live on a low-dimension manifold, we choose the latent dimension as 2. All the hyperparameters are set the same, with batch size 100 and 100 epoch. Fig. 3 shows the 2-dim latent distribution before the recognition network. We can see that when training with element-wise reconstruction error, the inter-

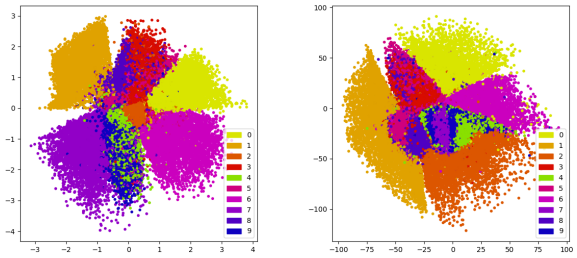


Figure 3. Latent space manifold visualization. **Left:** Element-wise reconstruction error in VAE. **Right:** VAE with discriminator, i.e., representation error.

Table 1. Comparisons of VAE (Kingma & Welling, 2013) and IWAE (Burda et al., 2015) under different settings. Negative log likelihood is compared (lower is better), trained on MNIST dataset. K is the number of samples in IWAE. Number of layers is the stochastic layer number that generates proposals.

	1 LAYER		2 LAYER	
K	VAE	IWAE	VAE	IWAE
1	88.69	88.67	89.03	88.97
5	NA	86.91	NA	85.77
50	NA	85.76	NA	84.32

class distance is larger. On the contrary, the discriminator benefits learning by encouraging the representation of each class. For instance, the latent distributions of 6 and 8 digits are closer, indicting potential close relationship between these two categories, which also speeds up convergence.

5.2.2. Importance Weighted AE

We then conduct experiments on IWAE (Burda et al., 2015) and VAE. We can see in Table 1 that when breaking the modeling assumption of VAE and loose the constraints by sampling more samples, the performance will get better (with the cost of more time until convergence).When changing from 1 stochastic layer to 2, the performance will further be boosted, as more complex projections can be learned.

6. Conclusion

In this work, we review two widely used frameworks of generative models, generative adversarial framework and variational inference framework. We analyze the drawbacks and improvements of them. From the perspective of minimizing general divergences, the mode seeking and mode covering nature of these frameworks is revealed, which also provides potential improvements by combining divergences from the two directions. Not only are the generative models suitable for image synthesis, but also can be used in other research fields, including RL and information theory.

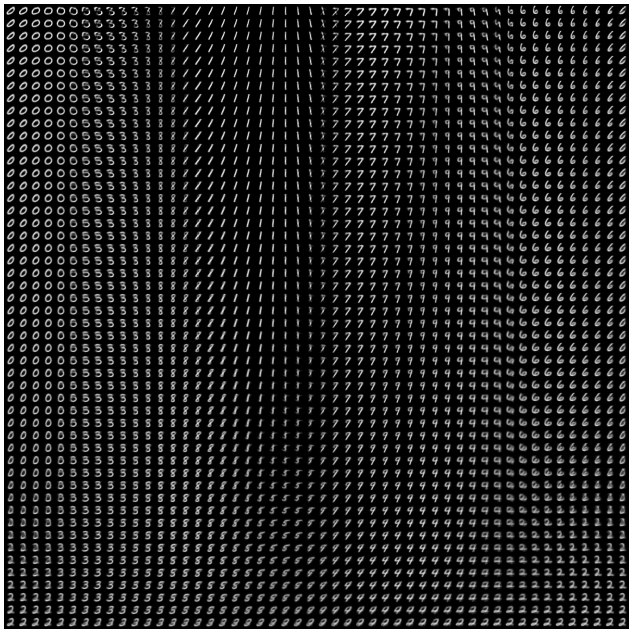


Figure 4. Manifold of generated samples of VAE. We can see that by sampling different latent codes, the generated samples show diversities as expected. From the manifold, VAE is believed to capture the low-dimensional distribution of the digit images.

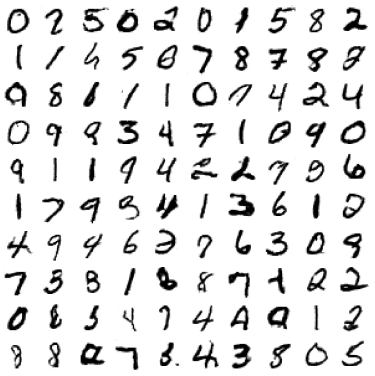
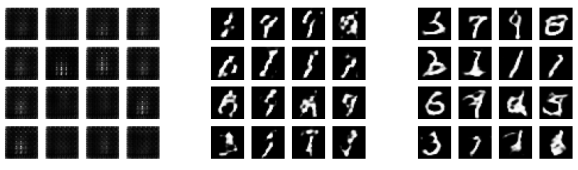


Figure 5. Final output after 100 epochs of GAN,



(a) Output at epoch 1 (b) Output at epoch 25 (c) Output at epoch 50

Figure 6. DCGAN intermediate generating samples.

References

Ambrogioni, L., Güçlü, U., Güçlütürk, Y., Hinne, M., van Gerven, M. A., and Maris, E. Wasserstein variational inference. In *Advances in Neural Information Processing Systems*, pp. 2473–2482, 2018.

Banerjee, A. On bayesian bounds. In *Proceedings of the 23rd international conference on Machine learning*, pp. 81–88, 2006.

Belghazi, M. I., Baratin, A., Rajeswar, S., Ozair, S., Bengio, Y., Courville, A., and Hjelm, R. D. Mine: mutual information neural estimation. *arXiv preprint arXiv:1801.04062*, 2018.

Burda, Y., Grosse, R., and Salakhutdinov, R. Importance weighted autoencoders. *arXiv preprint arXiv:1509.00519*, 2015.

Donsker, M. D. and Varadhan, S. S. Asymptotic evaluation of certain markov process expectations for large time, i. *Communications on Pure and Applied Mathematics*, 28 (1):1–47, 1975.

Futami, F., Sato, I., and Sugiyama, M. Variational inference based on robust divergences. In *International Conference on Artificial Intelligence and Statistics*, pp. 813–822, 2018.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In *Advances in neural information processing systems*, pp. 2672–2680, 2014.

Hiriart-Urruty, J.-B. and Lemaréchal, C. *Fundamentals of convex analysis*. Springer Science & Business Media, 2012.

Jordan, M. I., Ghahramani, Z., Jaakkola, T. S., and Saul, L. K. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

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

Larsen, A. B. L., Sønderby, S. K., Larochelle, H., and Winther, O. Autoencoding beyond pixels using a learned similarity metric. In *International conference on machine learning*, pp. 1558–1566. PMLR, 2016.

Levine, S. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*, 2018.

Li, Y. and Turner, R. E. Rényi divergence variational inference. *Advances in Neural Information Processing Systems*, 29:1073–1081, 2016.

Mirza, M. and Osindero, S. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015.

Sutton, R. S. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Machine learning proceedings 1990*, pp. 216–224. Elsevier, 1990.

Wan, N., Li, D., and Hovakimyan, N. f-divergence variational inference. *Advances in Neural Information Processing Systems*, 33, 2020.