Application of Generative Adversarial Networks on handwriting and other image datasets

Ye Yushi (12207894)
Department of Mathematics
The Hong Kong University of Science and Technology
yyeaf@connect.ust.hk

May 21, 2017

Abstract

In this final project, we try to study the *generative adversarial networks* method introduced by Ian Goodfellow et al. in 2014 [1] and apply it on the handwriting and other image datasets (including the Raphael dataset provided by Professor Wang Yang, though the result is bad due to limit sample size). Generative adversarial networks (GAN) is designed as an unsupervised learning algorithm and its most astounding application is to produce samples of photorealistic images. Some revised version of GAN, such as *DCGAN* introduced by [2] can produce images which mix the spurious with the genuine if the training set and model are well designed. However, *CatGAN*, introduced by [3], can expand the application of GAN to *semi-supervised learning* problems. We will also study this novel method and apply it on handwriting datasets.

1 Generative adversarial networks

Generative adversarial networks method was introduced by Ian Goodfellow et al. in 2014 [1] and soon became a hotspot learning algorithm in artificial intelligence area. The idea of GAN is to train two learners, or objective functions simultaneously. One of them is called *generator*, or G, which is designed to generate input-like data from noise. The other is called *discriminator*, or D, which aims to tell apart genuine samples from fake samples produced by G. GAN can be treated as a two-player zero sum game between G and D. In each step, the generator tries to produce examples which can fool the discriminator; by combining these fake examples with some genuine ones, the discriminator tries to classify them as "real" or "fake". Usually people use deep neural networks as both G and D. By stochastic gradient descent (SGD) steps, both models will be updated and their ability can be boosted.

Formally, let $X = \{x_1, \ldots, x_N\}$ is the training set and each x contains m features. The generator G maps random vectors $z \in \mathbb{R}^k$ to generate fake inputs $\hat{x} = G(z)$ while the discriminator D aims to predict the probability of example x being present in X by a softmax $p(y = 1|x, D) = \frac{1}{1+exp(-D(x))}$. And the whole objective function of GAN is given by

$$\min_G \max_D \{\mathbb{E}_{x \in X}[\log p(y = 1|x, D)] + \mathbb{E}_{z \sim P(z)}[\log(1 - p(y = 1|G(z), D))]\} \tag{1}$$

Here $P(z)$ is an arbitrary noise distribution which usually takes uniform distribution. If both G and D are differentiable, then they can be trained through iteratively in a SGD way.

Submitted to MATH 6380J Mini-project 2 .

(a) Epoch 1

(b) Epoch 3

(c) Epoch 5

(d) Epoch 7

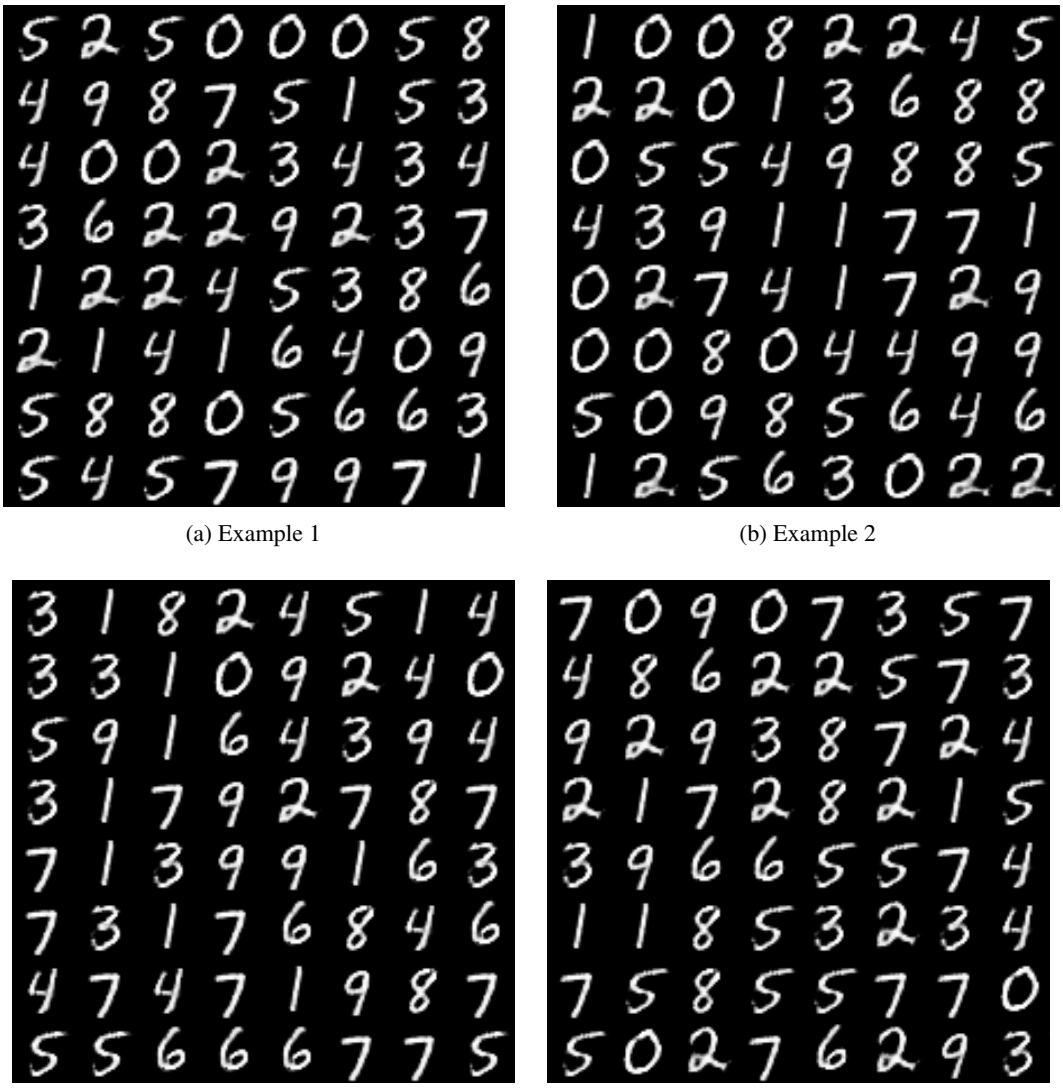
Figure 1: Training results of DCGAN on MNIST dataset

2 Deep Convolutional Generative Adversarial Network

Deep convolutional generative adversarial networks (DCGAN) was introduced by Radford and Metz (2015). The main modification is to use deep convolutional networks (CNNs) for G and D. It also has some other modifications such as applying *batch normalization* in mini-batch searching and using *ReLU* as activation functions in the hidden layers while using *tanh* in the last layer. Figure 1 shows the training results after epoch 1, 3, 5 and 7. One can see that after epoch 1, only some blurry outlines can be identified, while after epoch 7, almost all numbers can be seen as genuine as the original inputs.

After 25 epoch's training, we can use this fully-training model to generate more images of "handwriting" images, four of them are shown in Figure 2. It is nearly impossible to identify them as "fake" images generated by models!

We also applied this algorithm on the Raphael dataset provided by Professor Wang Yang, the result is very bad (shown in the Appendix). Even after 50 epoch, the output of training is still meaningless. The main reason is due to the limitation of sample size. After all, we only have dozens of images where only 12 of them are drawn by Raphael himself. In Appendix, we also present the results from



(a) Example 1

(b) Example 2

(c) Example 3

(d) Example 4

Figure 2: Testing results of DCGAN on MNIST dataset after 25 epoch

celebrity datasets (http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html) with more than 50000 images of celebrities. The result is much better.

3 Categorical Generative Adversarial Networks (CatGAN)

As the previous introduction, GAN is born to be an algorithm for unsupervised learning. However, there are some modifications which can apply GAN on supervised or semi-supervised learning problems, such as CatGAN introduced by Tobias (2015) [3]. This method can also be used for unsupervised learning. The idea is to let discriminator D has multiple jobs: instead of asking $D$to predict the probability of an example x belonging to the original training set X or not, D in CatGAN tries to assign "real" x to one of K categories (here K is a pre-chosen number, for example, for MNIST dataset, it is naturally to set $K = 10$) *with relatively high certainty* while assign $\hat{x}$ (i.e. fake examples generated by generator G) to any categorical *with relatively low certainty*. Here such "certainty" can be measured by the *conditional entropy* of $p(y|x, D)$ or $p(y|G(z), D)$. That is

$$\mathbb{E}_{x\sim X}\{H[p(y|x,D)]\} = \frac{1}{N}\sum_{i=1}^{N} H[p(y|x_i,D) = \frac{1}{N}\sum_{i=1}^{N}\{-\sum_{k=1}^{K} p(y=k|x_i,D)\log p(y=k|x_i,D)\} \tag{2}$$

and

$$\mathbb{E}_{z\sim P(z)}\{H[p(y|G(z),D)]\} = \frac{1}{M}\sum_{i=1}^{M} H[p(y|G(z_i),D)] \tag{3}$$

Here $p(y=k|x,D) = \frac{exp(D_k(x))}{\sum_{k=1}^{K} exp(D_k(x))}$ is generated through a softmax assignment based on D's output. M denotes the number of fake samples generated by G.

So the aims of D are to maximize $\mathbb{E}_{z\sim P(z)}\{H[p(y|G(z),D)]\}$ so that D assigns fake $\hat{x}$ to any category with low certainty[1] while minimize $\mathbb{E}_{x\sim X}\{H[p(y|x,D)]\}$ so that real x can be classified to some class with high certainty, i.e. resulting a peaked conditional class distribution. Finally, one need to maximize the *marginal class entropy* over all data points since in this paper's setting, the author use uniform distribution as prior distribution of $p(y)$[2], i.e., each categories contains approximately same number of examples. This marginal class entropy can be written as

$$H_X[p(y|X,D)] = H[\frac{1}{N}\sum_{i=1}^{N} p(y|x_i,D)] \tag{4}$$

As a consequence, the overall objective function for D will be

$$L_D = \max_D\{H_X[p(y|X,D)] - \mathbb{E}_{x\sim X}\{H[p(y|x,D)]\} + \mathbb{E}_{z\sim P(z)}\{H[p(y|G(z),D)]\}\} \tag{5}$$

Similarly, the marginal class entropy for samples from G can be written as

$$H_G[p(y|D)] = H[\frac{1}{M}\sum_{i=1}^{M} p(y|G(z_i),D)] \tag{6}$$

The generator G's aims are to maximize $H_G[p(y|D)]$ and minimize $\mathbb{E}_{z\sim P(z)}\{H[p(y|G(z),D)]\}$ so that it can fool D completely in one step.

As a consequence, the overall objective function for G will be

$$L_G = \min_D\{-H_G[p(y|D)] + \mathbb{E}_{z\sim P(z)}\{H[p(y|G(z),D)]\}\} \tag{7}$$

The above deduction can be illustrated by Figure 3

3.1 Extension of CatGAN on Semi-supervised Learning

Now let $X = \{\S^N, (\S^L, \dagger^L)\}$ to be the training set where part of it $\S^L$ has labels $\dagger_L \in \{1, 2, \ldots, K\}$. For a labeled pair (x, y), its *cross-entropy* can be given by

$$CE[y, p(y|x,D)] = -\sum_{k=1}^{K} y_k \log p(y=y_k|x,D) \tag{8}$$

Then the objective function for D in semi-supervised learning problems can be

[1] Actually, the limitation will lead to a uniform distribution

[2] This is also the noise distribution of G.

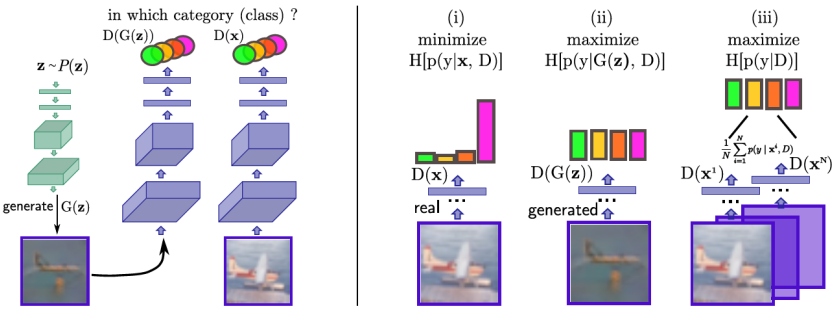


Figure 3: Illustration of CatGAN algorithm

$$L_D^L = \max_D \{H_X[p(y|X,D)] - \mathbb{E}_{x\sim X}\{H[p(y|x,D)]\} + \mathbb{E}_{z\sim P(z)}\{H[p(y|G(z),D)]\} + \lambda \mathbb{E}_{(x,y)\sim X^L}[CE[y,p(y|x,D)]]\} \tag{9}$$

Here λ is a hyper-parameter weights the cross-entropy term. And the generator is the same as the previous case. The author has applied CatGAN on MNIST dataset and found even for no label case, the test error rate can still be as low as 9.6%. If given 100 examples with labels (totally 60000 examples), the test error rate can be lowered to 1.76%. If given all labels (i.e. supervised learning case), the test error rate can be as low as 0.91%.[3]

4 Conclusion

From the previous study, we found that GAN is indeed a powerful tool in artificial intelligence which can be applied on unsupervised or semi-supervised learning cases. The results of GAN application on image generation is astounding. However, there are some drawbacks: firstly, due to its nature, the discriminator and generator need to achieve a Nash equilibrium point of a two-player zero sum game but GSD procedures may fail to find such point; Secondly, sample size is an important issue for the success of generation. For Raphael dataset, since we only have dozens of pictures, the result is very bad. But for MNIST datasets which contains 60000 examples, the result is astounding.

References

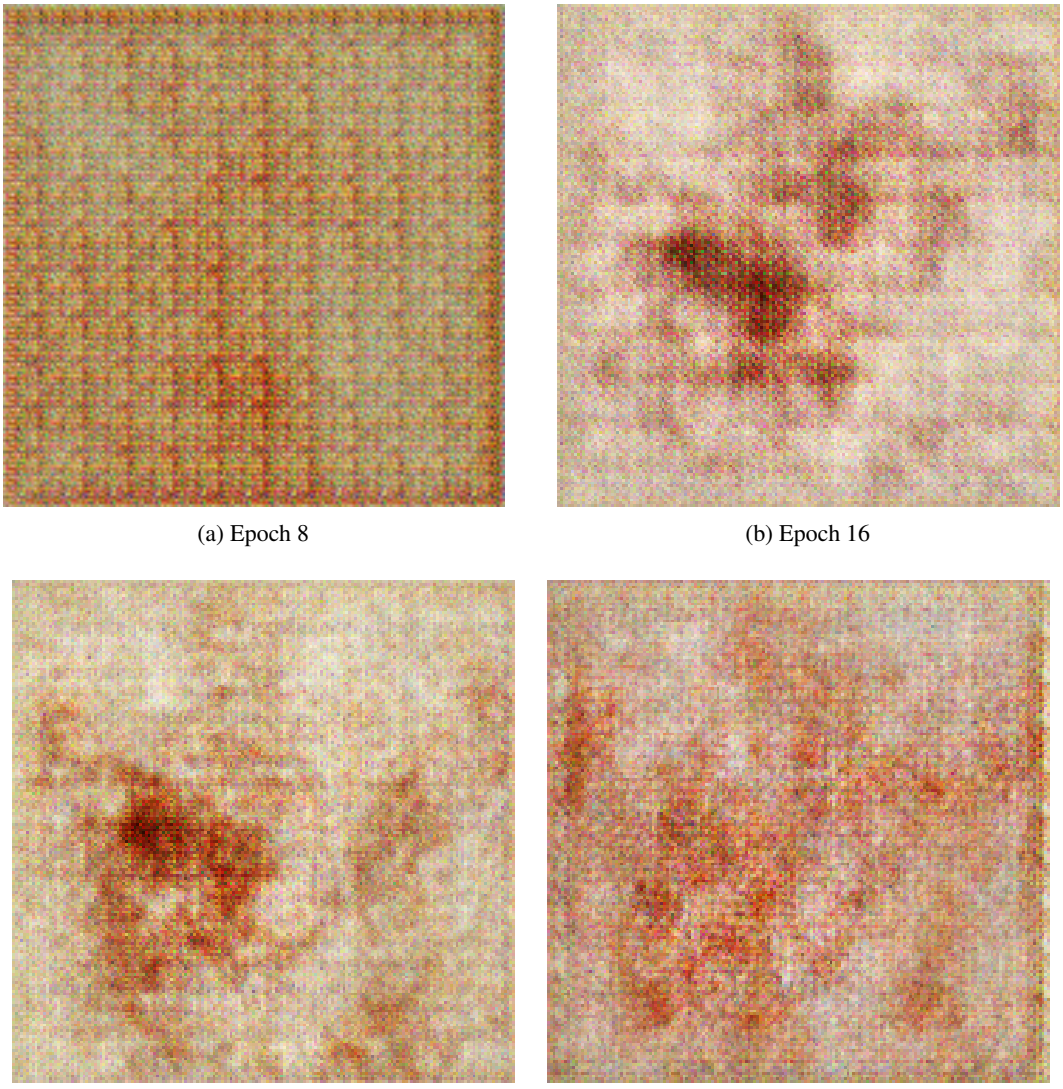
[1] Goodfellow, Ian, et al. "Generative adversarial nets." Advances in neural information processing systems. 2014.

[2] Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." arXiv preprint arXiv:1511.06434 (2015).

[3] Springenberg, Jost Tobias. "Unsupervised and semi-supervised learning with categorical generative adversarial networks." arXiv preprint arXiv:1511.06390 (2015).

Appendix

[3] We tried to code this algorithm ourselves but failed since it need us to change some internal functions in some deep learning frameworks such as tensorflow which beyonds the scope of our coding skills.



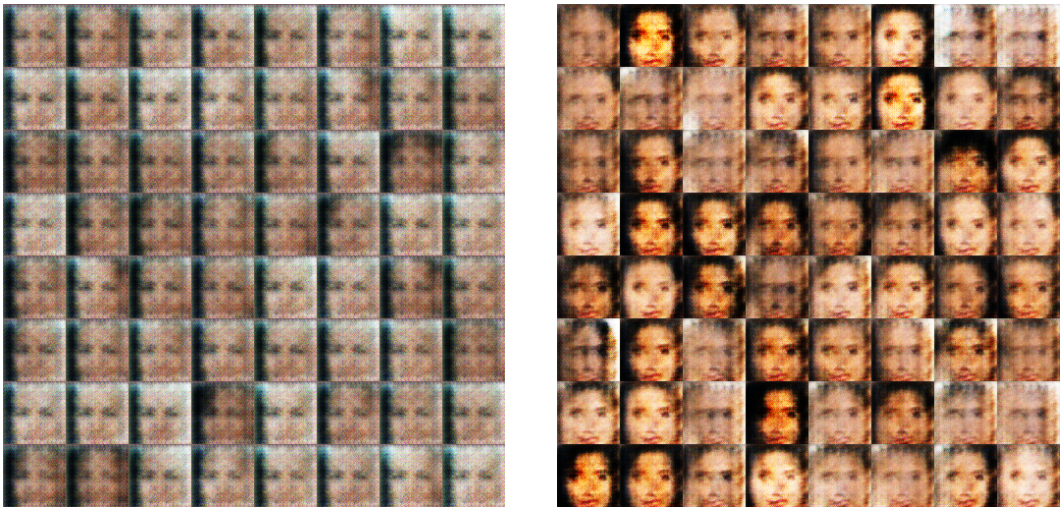
(a) Epoch 8

(b) Epoch 16

(c) Epoch 24

(d) Epoch 50

Figure 4: Training results of DCGAN on Raphael dataset



(a) Epoch 1

(b) Epoch 5

(c) Epoch 10

(d) Epoch 15

Figure 5: Training results of DCGAN on celebrity dataset