Combining Variational Autoencoders & Generative Adversarial Networks to Improve Image Quality

Atin S Hussain, Grade 12 CS

10.31219/osf.io/8bmdu

Abstract

Generative Adversarial Networks(GAN)^[1] are trained to generate images from random noise vectors, but often these images turn out poorly due to any of several reasons such as model collapse, lack of proper training data, lack of training, etc. To combat this issue this paper, makes use of a Variational Autoencoder(VAE)^[2]. The VAE is trained on a combination of the training & generated data, after this the VAE can be used to map images generated by the GAN to better versions of it. (This is similar to Denoising, but with few variations in the image). In addition to improving quality the proposed model is shown to work better than normal WGAN's^[3] on sparse datasets with higher variety, in equal number of training epochs.

Keywords: Generative Adversarial Networks(GAN)^[1], Variational Autoencoder(VAE)^[2], Generated Image Denoising

Introduction

The Generative Model used in this paper is the Wasserstein GAN^[3] since it eliminates the issue of convergence and model collapse. The Autoencoder model is a Variational model. Once the image generated by the GAN is mapped on to the latent space by training the autoencoder, the decoder will map it back to an image of improved quality. This method of generating images is especially useful in case of small datasets with a wide variety or image generation with limited processing power.

Figure 1: The proposed architecture is a combination of a GAN[consisting of a generator(1a) & a critic(1b)] & a VAE (2).

Model Summary

The Model consists of these two main networks:

Generative Adversarial Network^[1]

The Generative Model used is a typical Wasserstein GAN^[3]. The model consists of two networks, a Generative & a Critic Network which combine to form the GAN. The architectures are shown in the figure 2(a),(b),(c).

Variational Autoencoder^[2]

The Variational Autoencoder maps the training images and the generated images onto the latent space. Once mapped, the decoder outputs an image of improved quality by sampling from the latent space. The architecture is shown in figure 2(d).

Both networks work independent of each other. The WGAN takes the real images and generates fake images. The loss for the generator & discriminator is used to improve their performance. After the critic iteration is done the images generated with the current weights is fed into the Autoencoder along with the training data. This is mapped onto the latent space and random samples are taken from it. This aims to improve quality of image generated by the GAN as well as to increase variations in the generated images.

Figure 2: The Architecture for (a) Generator (b) Critic (c) GAN (d) VAE (Visualized using Netron^[4])

Algorithm

The algorithm proposed in this paper, is a combination of a WGAN^[3] and a VAE^[2] working in collaboration to improve overall performance. The loss function optimized for each of them are:

1. Discriminator Loss: $\nabla_w [\frac{1}{m} \sum_{i=1}^m f_w(x^{(i)}) - \frac{1}{m} \sum_{i=1}^m f_w(g_\theta(z^{(i)}))]$
2. Generator Loss: $-\nabla_\theta [\frac{1}{m} \sum_{i=1}^m f_w(g_\theta(z^{(i)}))]$
3. VAE Loss: $\mathcal{L} = - \sum_{j=1}^M \frac{1}{2} [1 + \log(\sigma_i^2) - \sigma_i^2 - \mu_i^2] - \frac{1}{L} \sum_l E_{\sim q_\theta(z|x_i)} [\log_p(x_i | z^{(i,l)})]$

The proposed algorithm: all experiments in the paper used the default values $\alpha_{gan} = 0.00005$, $c = 0.01$, $m = 64$, $n_{critic} = 25$, $\mathcal{L} = 1$, $M = 100$, $n_{vae} = 500$.

Parameters: : α_{gan} , the learning rate. c , the clipping parameter. m , the batch size for gan. M , the batch size for VAE, n_{critic} , the number of iterations of the critic per generator iteration. n_{vae} , the number of iterations of the VAE per generator iteration

Hyperparameters: : w_0 , initial critic parameters. θ_{0gan} , initial generator's parameters, θ_{0vae} , initial autoencoder's parameter

---

```

1: while  $\theta_{gan}$  has not converged do
2:   for  $t = 0, \dots, n_{critic}$  do
3:     Sample  $\{x^{(i)}\}_{i=1}^m \sim P_r$  a batch from the real data.
4:     Sample  $\{z^{(i)}\}_{i=1}^m \sim p(z)$  a batch of prior samples
5:      $g_w \leftarrow \nabla_w [ \frac{1}{m} \sum_{i=1}^m f_w(x^{(i)}) - \frac{1}{m} \sum_{i=1}^m f_w(g_\theta(z^{(i)})) ]$ 
6:      $w \leftarrow w + \alpha_{gan} \cdot \text{RMSProp}(w, g_w)$ 
7:      $w \leftarrow \text{clip}(w, -c, c)$ 
8:   end for
9:   Sample  $\{z^{(i)}\}_{i=1}^m \sim p(z)$  a batch of prior samples
10:   $g_\theta \leftarrow -\nabla_\theta [ \frac{1}{m} \sum_{i=1}^m f_w(g_\theta(z^{(i)})) ]$ 
11:   $\theta_{gan} \leftarrow \theta_{gan} - \alpha_{gan} \cdot \text{RMSProp}(\theta, g_\theta)$ 
12:  for  $t = 0, \dots, n_{vae}$  do
13:     $X^M \leftarrow$  Random batch of  $M$  samples from Training set
14:     $Y^M \leftarrow$  Random batch of  $M$  samples from Generated set
15:     $X \leftarrow \text{join}(X^M, Y^M)$ 
14:     $\mu, \sigma \leftarrow \epsilon \leftarrow$  Random samples from noise distribution  $p(\epsilon)$ 
15:     $g \leftarrow \nabla_{\theta_{vae}, \varphi} \tilde{\mathcal{L}}^M(\theta_{vae}, \varphi; X, \epsilon)$ 
16:     $\theta_{vae}, \varphi \leftarrow \text{SGD}^{[8]}$ 
17:  end for
18:   $p(z) \leftarrow \epsilon$  Add generated to samples
19: end while

```

Latent Space Visualization^[5]

In this proposed model the final images are taken from a latent space^[5] which is graphed in a 2D space using matplotlib and new images can be sampled by hovering over any point on the 2D space. To show the effectiveness of this model, a sparse dataset containing varying types of portraits of people is used. A sample result is shown in figure 3.

Figure 3: Example of an Image sampled from a random point (*) on the latent space after 150,000 epochs

As shown in figure 3, the images in the dataset and generated images are spread out widely over the latent space and the sample can be generated from any point in this space.

Model Results

With the MNIST^[6] dataset the proposed model has an image quality comparable to that of a normal WGAN (As shown in figure 4).

Figure 4: Images generated by (a) WGAN+VAE (b) WGAN using the MNIST^[6] dataset after 10,000 epochs

Figure 5: Images generated by (a) WGAN+VAE (b) WGAN using the Celeb-A^[7] dataset after 200,000 epochs

Whereas with the Celeb-A^[7] dataset the images generated (figure 5) by the model proposed in this paper is of better quality in the same number of training epochs. This lack of quality with a normal WGAN may have been due to the diverse nature of the dataset. In case of the proposed model, this issue is removed by the use of a variational autoencoder to map out the differences in the latent space^[5] and sample from it.

References

[1] Generative Adversarial Networks:

<https://arxiv.org/abs/1406.2661>

[2] Variational Autoencoders:

[https://www.researchgate.net/publication/304163568 Tutorial on Variational Autoencoders](https://www.researchgate.net/publication/304163568_Tutorial_on_Variational_Autoencoders)

[3] Wasserstein GANs:

<https://arxiv.org/abs/1701.07875>

[4] Netron:

<https://github.com/lutzroeder/netron>

[5] Latent Space Visualization:

<https://idl.cs.washington.edu/files/2019-LatentSpaceCartography-EuroVis.pdf>

[6] MNIST:

<http://yann.lecun.com/exdb/mnist/>

[7] Celeb-A

<http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html>

[8] Stochastic Gradient Descent

<https://leon.bottou.org/publications/pdf/compstat-2010.pdf>