Enhancing Symmetry in GAN Generated Fashion Images

Vishnu Makkapati[1] and Arun Patro[2⋆]

[1] Myntra Designs Pvt. Ltd., Bengaluru - 560068, India
vishnu.makkapati@myntra.com
[2] Department of Electrical Engineering,
Indian Institute of Technology, Kharagpur - 721302, India
arun.patro@ee.iitkgp.ernet.in

Abstract. Generative adversarial networks (GANs) are being used in several fields to produce new images that are similar to those in the input set. We train a GAN to generate images of articles pertaining to fashion that have inherent horizontal symmetry in most cases. Variants of GAN proposed so far do not exploit symmetry and hence may or may not produce fashion designs that are realistic. We propose two methods to exploit symmetry, leading to better designs - a) Introduce a new loss to check if the flipped version of the generated image is equivalently classified by the discriminator b) Invert the flipped version of the generated image to reconstruct an image with minimal distortions. We present experimental results to show that imposing the new symmetry loss produces better looking images and also reduces the training time.

Keywords: Generative Adversarial Networks, Deep Learning, Symmetry Loss, Generator, Discriminator

1 Introduction

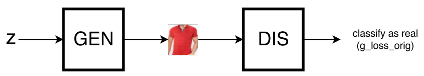
Generative Adversarial Networks (GANs) [3] are generative models that learn the distribution of the data without any supervision. They can be used to generate data (images or text) that are similar to the original dataset which look real enough to be indistinguishable by a human. GANs use adversarial learning that puts two networks, a Generator network and a Discriminator network in competition to learn the distribution of the input dataset. A generator tries to produce data that can fool the discriminator wheres the discriminator tries to identify them correctly as fake. The convergence of a GAN is highly empirical and is decided when the generator and discriminator losses are stable and the decision boundary is equi-probable.

We attempt to train a GAN to generate new fashion designs. The idea here is to learn the distribution of the input designs and produce new ones that are

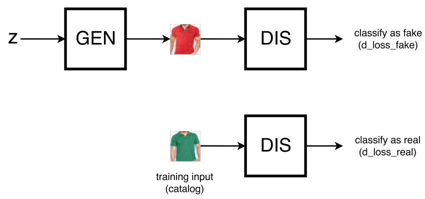
⋆ This work was performed when A. Patro was an intern with Myntra Designs Pvt. Ltd.

inspired by them. Most of the fashion article types such as t-shirts, shirts, jeans and trousers are symmetric. Hence it is expected that the GAN learns the inherent symmetry in the input data used for training. However, we noticed that DCGAN [6], a widely used implementation of GAN using conv-nets does not necessarily produce symmetric images. Also we observed that there are distortions in the images generated using it. But the location of the distortions is not symmetric. We propose some enhancements to DCGAN to get rid of undesirable artifacts in the synthesized images.

We propose a trick to reduce the distortions in the generated images by exploiting symmetry. There are some recent attempts to invert the Generator in GAN [1,2,5]. These methods try to estimate the latent vector used to produce the generated image such that the reconstructed image is very close to the original. We reconstruct an image with minimal distortions by estimating the latent vector from the flipped version of the generated image.



(a) Generator Losses



(b) Discriminator Losses

Fig. 1: Framework of DCGAN

2 Proposed Methods

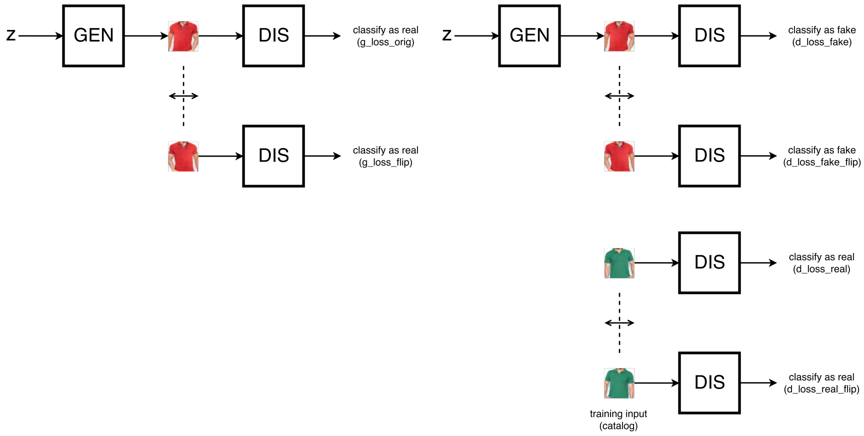
We exploit the inherent symmetry in fashion designs to train the GAN. To satisfy the symmetry condition, the horizontally flipped image of the generated image should look similar to the original. But it's not necessary that it is exactly same at a pixel level owing to certain design elements that are typically placed only on one side of the article (e.g., pocket/crest of a t-shirt). We exploit symmetry in the images to produce aesthetically better looking designs and reduce distortions in the generated images. Our contributions are summarized in this section.

2.1 Enhancements to DCGAN to Generate Symmetric Images

We impose a new symmetry loss where the flipped version of a generated image is discriminated equivalently by the discriminator. We realize it using DCGAN, a popular implementation of a GAN. Training the traditional DCGAN consists of minimizing three losses (Fig. 1):

1. *d_loss_real*: error in identifying input training images as real

2. *d_loss_fake*: error in identifying GAN generated images as fake
3. *g_loss_orig*: error in identifying GAN generated images as real



(a) Generator Losses (b) Discriminator Losses

Fig. 2: Framework of Proposed GAN

We trained a DCGAN and conducted experiments to check if it has learnt the symmetry of the images in the dataset. We flipped the generated images horizontally and passed them through the discriminator. If the distribution of the flipped images was also learnt, then the losses for the generated images and their flipped versions should be approximately equal. But we noticed that the losses for the flipped images are significantly higher (Fig. 3a). This proves that DCGAN has not learnt the inherent symmetry in the fashion designs.

We enhance DCGAN to learn symmetry in the input data by:

1. **Augmentation**: Training the GAN by augmenting the input images with their flipped versions.
2. **Classification Loss**: Introducing a new loss to check if the flipped version of the GAN generated image is equivalently classified by the discriminator (Fig. 2).

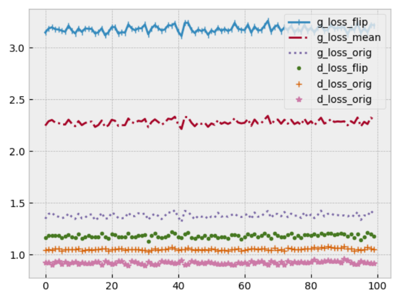
If the flipped augmented images are also used for training, the DCGAN should learn the distribution of them as well. But our experiments showed that it was not able to learn it (Fig. 3b). It can be seen that the generator loss for the flipped images is quite high though the discriminator learns well. We impose a symmetry loss and thus our GAN consists of evaluating six losses (Fig. 2):

1. *d_loss_real*: error in identifying input images as real
2. *d_loss_fake*: error in identifying GAN generated images as fake
3. *g_loss_orig*: error in identifying GAN generated images as real

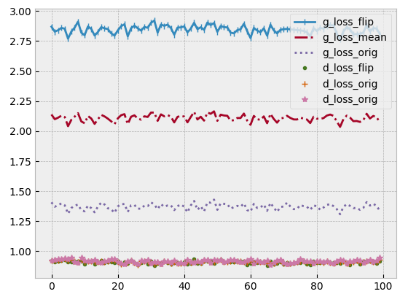
4. $\mathit{d\_loss\_real\_flip}$: error in identifying flipped input images as real
5. $\mathit{d\_loss\_fake\_flip}$: error in identifying flipped GAN generated images as fake
6. $\mathit{g\_loss\_flip}$: error in identifying flipped GAN generated images as real

wherein the final losses that are minimized are:

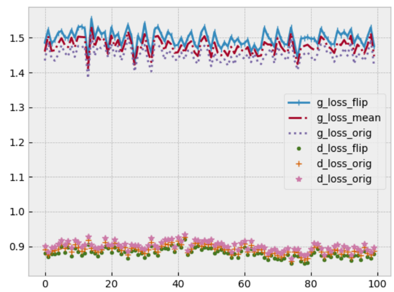
1. $\mathit{g\_loss\_mean} = (\mathit{g\_loss\_orig} + \mathit{g\_loss\_flip})/2$
2. $\mathit{d\_loss\_mean} = (\mathit{d\_loss\_real} + \mathit{d\_loss\_fake} + \mathit{d\_loss\_real\_flip} + \mathit{d\_loss\_fake\_flip})/4$



(a) DCGAN



(b) DCGAN with augmented images



(c) Proposed GAN

Fig. 3: Mean losses per epoch

We aid the discriminator by running the flipped images through it. Since the losses for the flipped images are used to train the discriminator, it will become better at identifying both the generated image and it's flipped version as fake. It can be found from Fig. 3c that the losses for the flipped images are just marginally higher than that for the original generated images. This proves that the generator trained using our method is able to produce images that are near symmetric.

2.2 Minimize the Distortions in Generated Images

When we visually observed the images produced by the generator, we found that they suffered from distortions (Fig. 4a). But the location of these distortions is asymmetric. We present a trick using symmetry to minimize them. We flip the generated images (Fig. 4b) and estimate the latent vector from which they are generated [1, 5]. We run the estimated latent vector through the generator and reconstruct them (Fig. 4c).

(a) Original (b) Flipped (c) Reconstructed

Fig. 4: Reconstruction of Flipped Images from GAN

We know that nearby latent vectors have close representations in the image space. Using this property, we try to reconstruct the generated images (X). An L2 loss between the generated image and its reconstructed version is minimized by using a regularizer on the magnitude of latent vector z [1].

$$L_{\text{recon}} = ||G(\hat{z}) - X||^2 + \lambda||\hat{z}||^2 \tag{1}$$

We can notice from Fig. 4c that the distortions present in Fig. 4b are minimized. One possible explanation for this result is that the location of the distortion in the flipped image is different (flipped) from that produced by the generator. The generator, in general, does not produce the distortion at the same location as that in the flipped image. Hence in the process of estimating the latent vector and reconstructing the flipped image from it, the distortions are reduced.

For example the images in (Row 2, Column 3) and (Row 4, Column 1) have hands occluding the t-shirt (Fig. 4b). The corresponding images in Fig. 4c do not suffer from these artifacts. This is also true for the other asymmetric distortions in the rest of the images.

3 Performance Evaluation

We evaluate the performance of the methods presented by training DCGAN and our proposed variant on t-shirt images from our catalog. The dataset comprised of 45,000 solid t-shirts with varying attributes (e.g., collar type, color, and sleeve length). We train GAN to generate images of resolution 64x64 pixels using a modified version of [4]. We quantitatively assess the two methods by comparing the losses (Fig. 3). We can notice from Fig. 5 that images generated using our method are near symmetric and do not suffer much from distortions.

Fig. 5: Samples from Proposed GAN at Epoch 100

4 Conclusion

We introduce a symmetry loss to train GAN to produce better looking images. We evaluated the performance of the scheme and demonstrated that the proposed method converges faster. We also present a trick to reduce the distortions in the generated images by inverting the flipped versions of them. The visual results show that the reconstructed images do not suffer from artifacts that are generally produced by GAN.

References

1. Bora, A., Jalal, A., Price, E., Dimakis, A.G.: Compressed sensing using generative models. arXiv preprint arXiv:1703.03208 (2017)
2. Creswell, A., Bharath, A.A.: Inverting the generator of a generative adversarial network. arXiv preprint arXiv:1611.05644 (2016)
3. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in neural information processing systems. pp. 2672–2680 (2014)
4. Kim, T.: A tensorflow implementation of deep convolutional generative adversarial networks, https://github.com/carpedm20/DCGAN-tensorflow
5. Lipton, Z.C., Tripathi, S.: Precise recovery of latent vectors from generative adversarial networks. arXiv preprint arXiv:1702.04782 (2017)
6. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434 (2015)