Toward better reconstruction of style images with GANs

Alexander Lorbert, Nir Ben-Zvi, Arridhana Ciptadi, Eduard Oks, Ambrish Tyagi
Amazon Lab126
[lorbert,nirbenz,ciptadia,oksed,ambrisht]@amazon.com

ABSTRACT

Generative adversarial networks (GANs) have recently seen a surge of interest in learning an inverse mapping—projecting the data back to the latent space. This learned mapping allows for image reconstruction—encoding and decoding—from a compact latent space. The choice of loss function(s) in this framework plays a critical role in determining the quality of the reconstruction. In this paper we explore possible options for loss functions and present qualitative evaluations applied to style images. We also introduce a loss that penalizes imperfect latent space reconstruction and integrate it with the Bidirectional GAN framework for encoding and generating style images. Our experiments show that this penalty aids in producing a more realistic reconstruction.

1 INTRODUCTION

The domain of fashion style is ever-changing with varying individual tastes, and encompasses copious interdependent factors. Even for an outfit that can be perceived as simple, many details such as color, patterns and clothing type must integrate well. Accessories such as jewelry, hats or scarves add another layer of intricacies since the quality of an outfit depends on the synergy of the entire ensemble. This complex nature of style and fashion makes the domain difficult to quantify and model.

As recent as 2014, with the advent of Generative Adversarial Networks (GANs) [4], generative modeling has seen a rebirth-of-sorts due to its surprising success in numerous complex domains. A GAN posits a multidimensional latent space and uses deep learning to learn a mapping from this latent space to the domain of interest. The underlying framework is motivated by a game between a generator G and a discriminator D—two neural nets whose parameters are learned via stochastic gradient descent (SGD). The role of the discriminator D is to discern real images from generated (fake) images produced by G, while the role of G is to produce images which can fool the discriminator. Alternatively, the GAN objective uses variational bounds to minimize the mutual information between the image type (real or fake) and the image itself.

Building upon the success of GANs, Donahue *et al.* [2] and Dumoulin *et al.* [3] propose a framework to perform inference with GANs by adding an encoder network. The idea is that rather than limit the learning process to a single mapping from the latent space to the observed data space, the Bidirectional GAN (BiGAN) also learns an encoding mechanism to map data to the latent space. This comes at the cost of having to train three neural nets as opposed to two, with a discriminator that accepts inputs from both the data and the latent space.

The problem we focus on in this paper is learning to encode and decode (generate) fashion images. Specifically, we are interested in using GANs for reconstruction: mapping a fashion image to a latent space via the encoder and then mapping the latent code to the image space via the generator, thereby reconstructing the original image or the original *style* content of the image. When coupled with "perfect" reconstruction, the ability to represent a complex style image in a compact latent space opens up the possibilities to a host of real-world applications. For example, perfect reconstruction allows us to change the style of a particular outfit by simply walking along the latent space. It also allows for interesting applications such as virtual try-on by reconstructing a particular fashion image conditioned on some new clothing item.

The rationale of using GANs over other generative modeling techniques stems from the realistic images they produce. To date, GANs produce the most life-like, realistic images with sharp edges and defined contours. Other reconstruction approaches such as auto-encoders and variational auto-encoders produce images that are just too smooth or blurry for a real-world application. This advantage, however, comes at a cost in that GANs are exceedingly difficult to train. Perhaps the biggest hurdle for training a GAN is *mode collapse*: empirically, when the generator succeeds in fooling the discriminator with a particular set of generated images, it subsequently learns to bias the mapping of the latent space toward this set. When mode collapse occurs its presence is immediately detectable since a random sampling of latent codes generate the same few images. Image encoding and decoding—and therefore reconstruction—will fail to generate realistic images when mode collapse arises during training.

In contrast to an auto-encoder objective, we add a loss that stems from the latent space and not the image space. Smoothing effects in the image space (or pixel level) can drastically alter image quality and content (they tend to result

in blurry images). On the other hand, under the assumption that images are encoded on a smooth manifold in the latent space, then small perturbations in this space should have no effect on image quality.

The main contribution of our paper is a loss that encourages better reconstruction. Starting with a BiGAN architecture, we employ many of the techniques of [19] for trainining. Additionally, we add our proposed reconstruction loss that penalizes imperfect *latent* reconstructions. We have found that latent reconstruction loss aids in training and mode collapse avoidance.

The paper is organized as follows: in §2 we present prior work relating to encoding and decoding with GANs. In §3 we review BiGANs and in §4 we introduce our loss. Experiments are presented in §5 and we conclude in §6.

2 RELATED WORK

A popular loss that is used for image reconstruction is the pixel-level squared loss. This type of loss has been used in various autoencoder approaches such as Invertible Conditional GANs [13] and Variational Autoencoders (VAE) [8, 16]. However, pixel-level loss tends to produce blurry images [4] due to the averaging effects. There are attempts to alleviate this problem by introducing some form of structural loss in the image pixel domain (*e.g.*, the work by Ridgeway *et al.* [17]) to create a more perceptually-correct similarity metric. While this approach results in sharper images, it does not have the same potential to capture high level structure as a similarity metric that uses a deep neural network (DNN) trained from millions of real world images. Our loss makes use of the intermediate representation of a DNN to compute a similarity metric. This allows our network to capture high-level semantic information to guide the reconstruction process.

Using a pre-trained DNN to guide the training of another network has been shown to achieve good results in a knowledge transfer task [5, 18]. Our loss can be considered as a type of knowledge transfer method where we use a DNN as a way to guide the training of the generator and encoder network in a BiGAN. The main difference between our approach and those existing methods is that we do not aim to transfer knowledge from one bottom-up DNN to another. Instead, our goal is to use the intermediate representation of a DNN to guide the style reconstruction process.

Although there have also been attempts to incorporate pre-trained DNNs in GANs [6], their goal is to provide additional supervision signal to the generator network so that it can generate better samples while being more stable to train. In contrast to our work, they do not address the inference problem in GANs (*i.e.*, how to recover the latent representation of the real input image).

Similar to our work is the work by Larsen *et al.* [10] that combines the autoencoder structure with adversarial training. This approach uses the intermediate representation from the discriminator network to compute a similarity metric for the reconstruction. While this method has been shown to produce sharper reconstruction images than VAEs, it is unclear that the intermediate representation learned by the discriminative network carries the semantic information that allows for accurate style reconstruction. In [1], Brock *et al.*use an adversarially trained model for image editing. They train their model to reconstruct an image using a combination of pixel loss, feature loss, GAN loss, and VAE's KL divergence loss. In their model, they reuse the discriminator network as the feature extractor of the encoder. In contrast to their work, we train a separate feature encoder and use a specific latent space reconstruction loss rather than the KL divergence loss.

3 BIDIRECTIONAL GANS

We briefly review the BiGAN framework [2, 3]. Let $\mathcal{X}$ and $\mathcal{Z}$ denote the image space and latent space, respectively. A real image $x \in \mathcal{X}$ is sampled from the distribution $q(x)$ and a latent code $z \in \mathcal{Z}$ is sampled from the distribution $p(z)$. Subsequently, a latent code $z' \in \mathcal{Z}$ is generated according to $q(z|x)$ and $x' \in \mathcal{X}$ is generated according to $p(x|z)$. This process gives rise to the two pairs (x, z') and (x', z). The role of the discriminator, $D : \mathcal{X} \times \mathcal{Z} \to [0, 1]$, is to determine the probability that the input pair is a sample from the generating process governed by $q(z|x)q(x)$.

Sampling according to $q(x)$ is accomplished by sampling actual images. To sample z, a multidimensional Gaussian or uniform distribution is commonly used as the sampling distribution $p(z)$. Sampling from $q(x|z)$ and $p(z|x)$ is effectively dictated by the generator and encoder, respectively. For simplicity, we assume that the generator is some function $G : \mathcal{Z} \to \mathcal{X}$ and the encoder is some function $E : \mathcal{X} \to \mathcal{Z}$.[1]

Now the game begins: samples are generated according to q or p. The discriminator is trained to predict the generating distribution, while the encoder+generator are trained to fool the discriminator. The objective function is given by

$$\begin{aligned} \mathcal{L}_{GAN}(D,E,G) &\triangleq \mathbb{E}_q[\log D(X,Z)] \\ &\quad + \mathbb{E}_p[\log(1 - D(X,Z))] \qquad (1) \\ &= \mathbb{E}_{q(x)}\Big[\mathbb{E}_{q(z|x)}[\log D(X,Z)]\Big] \\ &\quad + \mathbb{E}_{p(x)}\Big[\mathbb{E}_{p(z|x)}[\log(1 - D(X,Z))]\Big] \qquad (2) \end{aligned}$$

and we seek to optimize

$$\min_{G,E} \max_{D} \mathcal{L}_{GAN}(D,E,G) \quad . \qquad (3)$$

[1] In practice $E(x)$ can have dimension, say, $2|\mathcal{Z}|$ to store both mean and variance vectors along each dimension. The variance is incorporated during training via the reparameterization trick [8].

For an optimal encoder+generator, the optimal discriminator is given by $q(x,z)/(q(x,z)+p(x,z))$, and for an optimal discriminator the optimization is equivalent to minimizing the Jensen-Shannon divergence between $q(x,z)$ and $p(x,z)$. If the global optimum of (3) is achieved, supplying the functions $D^\star$, $E^\star$ and $G^\star$, then the two generating processes are stochastically equivalent, i.e., $q(x,z) = p(x,z)$, and $E^\star$ and $G^\star$ are inverses of each other (almost everywhere). Thus, training a BiGAN indirectly encourages perfect reconstruction. Furthermore, the authors of [2] demonstrate that, given an optimal discriminator, the optimization reduces to an ℓ_0 autoencoder.

4 LATENT RECONSTRUCTION LOSS

With a focus on reconstruction, $\mathcal{L}_{GAN}$ is empirically insufficient. As of the writing of this paper, image generation via GANs is still imperfect. Style images also pose a significant challenge because as humans we can be highly sensitive to the most minor of details surrounding fashion. This motivates the addition of another loss function.

The familiar choice would be the autoencoder loss,

$$\mathcal{L}_{AE}(x) \triangleq \|x - (G \circ E)(x)\| \ . \tag{4}$$

Here, we penalize the actual pixel values to match before and after reconstruction. During training we consider the reconstruction of real images only. As stated previously, autoencoders tend to produce smooth reconstructions so the relative weight assigned to $\mathcal{L}_{AE}$ should be small.

Let $H : \mathcal{X} \to \mathbb{R}^n$ be a *fixed*, differentiable function. Rather than penalize reconstruction at the pixel level, we penalize at the feature level. This gives

$$\mathcal{L}_H(x) \triangleq \|H(x) - (H \circ G \circ E)(x)\| \ , \tag{5}$$

which is used by [12] and is similar to feature matching [19]. In our experiments, we consider the `pool5` layer output of AlexNet [9] for H.

When we think of reconstruction, pixel-level matching is the typical presumption. The style domain, as manifested in the pixel space, is not a smooth manifold. Empirically, the latent space possesses smooth properties, such as the decoded trajectory from z_1 to z_2 produces a symantically smooth transition from x_1 to x_2 [14]. This motivates two penalties for latent space reconstruction:

$$\mathcal{L}_{AD}(z) \triangleq \|z - (E \circ G)(z)\| \tag{6}$$

$$\mathcal{L}'_{AD}(x) \triangleq \|E(x) - (E \circ G \circ E)(x)\| \ . \tag{7}$$

The penalty of (6) targets the p generative process while the penalty of (7) targets the q generative process. This is in contrast to feature matching which attempts to match feature statistics across processes. The subscript "AD" stands for autodecoder—a complementary term for autoencoder. In our training we use $\mathcal{L}_{AD}$ and not $\mathcal{L}'_{AD}$ because sampling from p is immune from mode collapse by design, which cannot be said for sampling from $q(z|x)$.

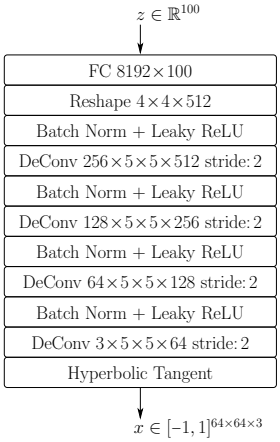

Figure 1: $G(z)$

If the latent space indeed encodes style information, then penalizing reconstruction along each dimension of $\mathcal{Z}$ makes sense since we want to preserve this style information. This is not the case with pixel space where we do not care about background pixel reconstruction as we do foreground pixel reconstruction. And since real-world style images, or style images "in the wild", do not all have the same background, we will be assigning equal importance to all pixel reconstructions regardless of pixel type (foreground/background).

5 EXPERIMENTS

Fashion Dataset. Our data consists of 520,000 fashion images, most of which contain a single individual standing in front of the camera wearing an outfit. For the purpose of training our models, we use a custom version of Faster R-CNN [15] to crop backgrounds from the images, leaving only the person. Images are subsequently resized to 64×64 and each color component value is linearly mapped to $[-1, 1]$.

Architecture. The architecture of the three BiGAN blocks are featured in Figures 1, 2, and 3. In the figures "FC" = fully connected, "Conv" = convolution, "DeConv" = transposed convolution, "Batch Norm" = Batch Normalization, and "Leaky ReLU" = Leaky Rectified Linear Unit (lReLU(t) = $\max(\alpha t, t)$ for positive scalar parameter α). The generator architecture is based off of the successful DCGAN architecture [14]. The image space is $\mathcal{X} = [-1, 1]^{64 \times 64 \times 3}$ and the latent space is $\mathcal{Z} = \mathbb{R}^{100}$. Each z is sampled independently from the uniform distribution over $[-1, 1]^{100}$.

Training. For the purpose of training our models we employ the same procedure from [14] and augment it with the techniques presented in [19] and [11]. We used one-sided label smoothing of 0.9 for the real images and executed 8

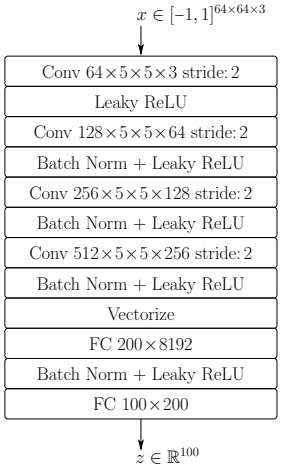

Figure 2: $E(x)$

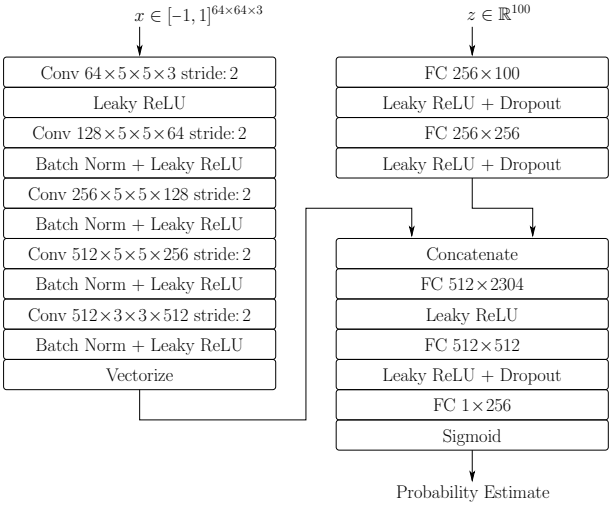

Figure 3: $D(x,z)$

discriminator unrolling steps per iteration. The decay parameter for the batch normalization was 0.97, the dropout parameter was 0.8, and the leak parameter was 0.2. We used the Adam solver [7] with learning rate 0.0002 and $\beta_1 = 0.5$. The batch size was 32. Our models were implemented using Google's TensorFlow and training was executed on an NVidia Tesla K80 GPU.

The objectives we consider take the general form

$$\mathcal{L}_{GAN} + \lambda_1 \mathcal{L}_{AD} + \lambda_2 \mathcal{L}_{AE} + \lambda_3 \mathcal{L}_H \quad . \tag{8}$$

The loss associated with $\mathcal{L}_H$ uses the ℓ_2-normalized `pool5` layer of AlexNet. We use a pseudo-Huber loss of the form $\gamma(\sqrt{1+(t/\gamma)^2}-1)$ for all penalties so as not to be too sensitive to outliers (see Figure 4). We present the results associated with the weight configurations as shown in Table 1

Experiment	λ_1	λ_2	λ_3
EX1	0.05	0	0
EX2	0.05	1	0
EX3	0.05	0	1
EX4	1	0	0
EX5	1	1	0
EX6	1	0	1
EX7	2.5	0	0
EX8	2.5	1	0
EX9	2.5	0	1

Table 1: Experiment configurations

after approximately 200K iterations (for the pseudo-Huber loss we set $\gamma = 0.05$ for all runs).

In Figures 7 through 22, we present sample generation (injecting noise into the generator) and image reconstruction. Figures presenting reconstruction have original images in the odd columns and the corresponding reconstructions in the even columns. The two digit number appearing at the top-left corner of each tiled image is the percent output provided by the discriminator. The reconstructions feature 10 randomly selected images that were not included in the training set.

For the experiments where $\lambda_1 = 0.05$ we see the BiGAN has difficulty converging to something we would call a decent encoder+generator pair. The generated samples and reconstructions are also indicative of mode collapse. Things improve with increasing λ_1 as SGD yields a generator capable of producing similar images from the dataset.

The experiments that utilize $\mathcal{L}_H$ (EX3, EX6, EX9) are more robust to pose but suffer with regard to preserving style content. Intuitively, `pool5` of AlexNet was not trained on pose-dependent images and style images, so this outcome is to be expected. Nevertheless, it may prove fruitful to use a feature vector from a DNN trained on style images to yield better reconstructions.

Visually, we find that the injection of $\mathcal{L}_{AD}$ yields the best samples and reconstructions. Image sharpness is slightly compromised with $\mathcal{L}_{AE}$. Also note how the generator is better at "fooling" the discriminator with a greater λ_1, *i.e.*, the discriminator outputs higher probability on the reconstructed images.

6 CONCLUSION AND FUTURE WORK

We have explored various losses in the BiGAN framework to achieve better reconstruction for fashion images. We have also demonstrated that the improved reconstructions are achieved without having to compromise on the quality of the generated synthetic images. For future work, we want to revisit the use of $\mathcal{L}_H$ with a network that is specifically tuned

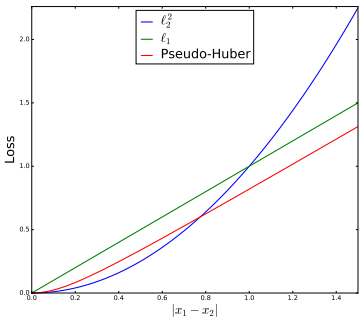

Figure 4: Various loss functions

on fashion images rather than ImageNet to achieve better style reconstruction. It will also be interesting to explore if we can add some constraints such as pose, color and outfit type to the reconstruction process.

REFERENCES

[1] Andrew Brock, Theodore Lim, James M. Ritchie, and Nick Weston. 2016. Neural photo editing with introspective adversarial networks. *CoRR* abs/1609.07093 (2016).

[2] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. 2016. Adversarial Feature Learning. *arXiv preprint arXiv:1605.09782* (2016).

[3] Vincent Dumoulin, Ishmael Belghazi, Ben Poole, Alex Lamb, Martin Arjovsky, Olivier Mastropietro, and Aaron Courville. 2016. Adversarially Learned Inference. *arXiv preprint arXiv:1606.00704* (2016).

[4] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems*. 2672–2680.

[5] Saurabh Gupta, Judy Hoffman, and Jitendra Malik. 2016. Cross modal distillation for supervision transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2827–2836.

[6] Xun Huang, Yixuan Li, Omid Poursaeed, John Hopcroft, and Serge Belongie. 2016. Stacked Generative Adversarial Networks. *arXiv preprint arXiv:1612.04357* (2016).

[7] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[8] Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

[9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*. 1097–1105.

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

[11] Luke Metz, Ben Poole, David Pfau, and Jascha Sohl-Dickstein. 2016. Unrolled Generative Adversarial Networks. *arXiv preprint arXiv:1611.02163* (2016).

[12] Anh Nguyen, Jason Yosinski, Yoshua Bengio, Alexey Dosovitskiy, and Jeff Clune. 2016. Plug & play generative networks: Conditional iterative generation of images in latent space. *arXiv preprint arXiv:1612.00005* (2016).

[13] Guim Perarnau, Joost van de Weijer, Bogdan Raducanu, and Jose M Álvarez. 2016. Invertible Conditional GANs for image editing. *arXiv preprint arXiv:1611.06355* (2016).

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

[15] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. (2015), 91–99.

[16] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082* (2014).

[17] Karl Ridgeway, Jake Snell, Brett D Roads, Richard S Zemel, and Michael C Mozer. 2015. Learning to generate images with perceptual similarity metrics. *arXiv preprint arXiv:1511.06409* (2015).

[18] Adriana Romero, Nicolas Ballas, Samira Ebrahimi Kahou, Antoine Chassang, Carlo Gatta, and Yoshua Bengio. 2014. Fitnets: Hints for thin deep nets. *arXiv preprint arXiv:1412.6550* (2014).

[19] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016. Improved techniques for training gans. In *Advances in Neural Information Processing Systems*. 2226–2234.

Figure 5: Samples of $p(x|z)$ from EX1.

Figure 6: EX1 reconstruction results. Odd colums are the original images, the even columns to the right are the reconstructed images.

Figure 7: Samples of $p(x|z)$ from EX2.

Figure 8: EX2 reconstruction results. Odd colums are the original images, the even columns to the right are the reconstructed images.

Figure 9: Samples of $p(x|z)$ from EX3.

Figure 10: EX3 reconstruction results. Odd colums are the original images, the even columns to the right are the reconstructed images.

Figure 11: Samples of $p(x|z)$ from EX4.

Figure 12: EX4 reconstruction results. Odd colums are the original images, the even columns to the right are the reconstructed images.

Figure 13: Samples of $p(x|z)$ from EX5.

Figure 14: EX5 reconstruction results. Odd colums are the original images, the even columns to the right are the reconstructed images.

Figure 15: Samples of $p(x|z)$ from EX6.

Figure 16: EX6 reconstruction results. Odd colums are the original images, the even columns to the right are the reconstructed images.

Figure 17: Samples of $p(x|z)$ from EX7.

Figure 18: EX7 reconstruction results. Odd colums are the original images, the even columns to the right are the reconstructed images.

Figure 19: Samples of $p(x|z)$ from EX8.

Figure 20: EX8 reconstruction results. Odd colums are the original images, the even columns to the right are the reconstructed images.

Figure 21: Samples of $p(x|z)$ from EX9.

Figure 22: EX9 reconstruction results. Odd colums are the original images, the even columns to the right are the reconstructed images.