A Sparse Substitute for Deconvolution Layers in GANs

Juzheng Li[1], Pengfei Ge[1], and Chuan-Xian Ren[1,2(✉)]

[1] School of Mathematics, Sun Yat-sen University, Guangzhou 510275, China
{lijzh29,gepengf}@mail2.sysu.edu.cn, rchuanx@mail.sysu.edu.cn
[2] Shenzhen Research Institute, Sun Yat-sen University, Shenzhen 518000, China

Abstract. Generative adversarial networks are useful tools in image generation task, but training and running them are relatively slow due to the large amount parameters introduced by their generators. In this paper, S-Deconv, a sparse drop-in substitute for deconvolution layers, is proposed to alleviate this issue. S-Deconv decouples reshaping input tensor from reweighing it by first processing it with a sparse fixed filter into desired form then reweighing them using learnable one. By doing so, S-Deconv reduces the numbers of learnable and total parameters with sparsity. Our experiments on Fashion-MNIST, CelebA and Anime-Faces verify the feasibility of our method. We also give another interpretation of our method from the perspective of regularization.

Keywords: Generative adversarial networks · Sparsity Deconvolution

1 Introduction

Generative adversarial networks (GANs [5]) have been a heated topic in the deep learning community since they provide a powerful framework that allows us to learn complex distributions in tasks where no explicit merits to apply such as image generation and manipulation. To this end various network architectures were designed, many of which have multiple generators with great depth [3,4,13]. Along with their mirrored counterparts, those generators introduce too many parameters making it slow to train GANs. There has been methods [6,8] focusing on substitutes with fewer parameters for convolution layers used in recognition tasks; however, to the best of our knowledge, no similar substitutes has been proposed for deconvolution layers in GANs.

In this paper, inspired by the success of introducing atrous convolution into semantic segmentation tasks [2], we propose a new building block of neural

C.-X. Ren—This work is supported in part by the Science and Technology Program of Shenzhen under Grant JCYJ20170818155415617, the National Natural Science Foundation of China under Grants 61572536, and the Science and Technology Program of GuangZhou under Grant 201804010248.

J.-H. Lai et al. (Eds.): PRCV 2018, LNCS 11257, pp. 446–456, 2018.
https://doi.org/10.1007/978-3-030-03335-4_39

networks called S-Deconv that functions similarly to a vanilla deconvolution layer but with controllable sparsity built within and fewer parameters to learn. These two properties are believed to helpful in terms of accelerating GANs during training. Our proposed method can be considered as a combination of fixed and learnable kernels. Although we adopt the similar setting of fixed kernel in [8], our approaches are fundamentally different. Firstly, the problem settings are not the same: generation and classification. Secondly, we use these fixed kernels as mean to reshape the input tensor reserving much of its information despite the sparsity, an encouraging result due to the nature of deconvolution while in [8], fixed kernels, followed by Sigmoid activation, were used to approximate the standard convolution layers.

2 Related Work

Generative Adversarial Networks. Being an upcoming technique in the unsupervised learning field, generative adversarial networks [5] provide a way to implicitly model distributions of high-dimensional data, for example, images and neural languages [9,10]. In general, a GAN can be well characterized by training two networks simultaneously in competition with each other. When a GAN is applied in image synthesis tasks, one can image that one of the networks is a forger whose specialty is to imitate master pieces while another is a expert whose majors in art. In this scenario, the forger G, or Generator in GAN terminology, will try its best to generate plausible art, or realistic images, from noises while the expert D, or Discriminator, will receive both fake and real art then try to identify which one is authentic and which one is not. In our experiments, we use WGAN [1], a variant of GAN approximates EM distance rather than JS distance, as our baseline model.

Local Binary Convolution. Local binary convolution (followed by a ReLU activation), an approximation of vanilla convolution which is also followed by a ReLU activation is proposed [8]. A LB-Conv consists of three parts: a convolution layer with fixed LB-kernels, a sigmoid activation and a standard 1×1 convolution. A LB-Conv kernels is many alike to a standard kernel used in convolution and deconvolution layers except that elements in a LB-kernel are only zero, one and negative one. We utilize this kind of kernels in our method for its sparsity and other desirable properties.

3 Our New Method

3.1 S-Deconv

Our new method is based on one observation that after a feature map which is passed to a deconvolution layer, two things happened *simultaneously*: (1) this tensor is reshaped and (2) new values are given by a weighted summation

of old ones. We call them *reshapingphase* and *reweighingphase*. Due to this simultaneity, a kernel in a deconvolution layer must be lager enough to catch the surrounding of a pixel in the feature map. However this often leads to vain efforts since this kernel is operated on a feature map that is seriously padded with zeros. If we shrink the size of this kernel or enlarge the stride of it, it may receive more non-padded values giving more meaningful reweighing results; however, doing so will jeopardize reshaping results forcing us to add more layers. To address this dilemma, we trade simultaneity for freedom in designing kernels. This decoupling reshaping and reweighing results our proposed method, S-Deconv layer.

In S-Deconv, reshaping is done by fixing sparse kernels containing only -1, 1 and 0 and then reweighing is done by channel-wise weighted summations. To best utilize existing deep learning library, both phases in S-Deconv are implemented by vanilla deconvolution and convolution layers: reshaping phase can be considered as a deconvolution layer with fixed LB-kernels while reweighing phase can be considered as a convolution layer with 1×1 kernels. However, these two operations may increase the computational afford of GAN since S-Deconv works best with sparse matrix multiplication. We present our main operations in Algorithm 1 to have a clear understanding of how our method works in practice and make a comparison to vanilla deconvolution layers in Table 1. And in Eq. (1) we show the ratio of numbers of learnable parameters

$$\frac{\#Deconv}{\#Our\ Method} = \frac{p \times h \times w}{m} \tag{1}$$

and total parameters in Eq. (2)

$$\frac{\#Deconv}{\#Our\ Method} = \frac{p \times h \times w \times q}{p \times h \times w \times m \times (1-\theta) + m \times q} \tag{2}$$

where p, q and m are numbers of input, output and intermediate channels, $h \times w$ is the size of deconvolution kernels and θ is the sparsity. Under mild settings, we can see that both ratios are lager than one, which indicates that our method uses fewer learnable and total parameters.

Algorithm 1. S-Deconv layer

Input: Tensor X; Fixed LB-Conv kernels K_{lb}; Learnable 1×1 Kernels $K_{1\times1}$.
Reshaping Phase: $X_{inter} = Deconv(X; K_{lb})$
Reweighing Phase: $Y = Conv(X_{inter}; K_{1\times1})$
Output: Y

We discuss some technical details as follows.

Why the Name? The name of our proposed method may be confusing. The meanings of "sparse" here are of two folds. First, weights in a S-Deconv layer are most zeros. Second, most of those weights need not to be learned.

Table 1. A comparison of our method and deconvolution

Layer	Components	Output
Vanilla deconvolution	Deconvolution	Feature map
Our method	Fixed deconvolution	Reshaped input
	1×1 convolution	Feature map

Drop-In Substitute. Given LB-Conv kernels and intermediate channel size, our method is a drop-in substitute for deconvolution layers. Our experiments show that replacing a deconvolution layer with a S-Deconv layer often require no changes in hyperparameter setting.

Why LB-Conv Kernels. When it comes to fixed hand-crafted kernels, there are many options including randomly initialized weights, among which we take LB-Conv kernels as our reshaping kernels for two reasons: (1) they have controllable sparse nature built within; (2) aside from zeros for sparsity, it contains only one and negative one which can help with persevering information.

Channels of Intermediate Feature Maps. After reshaping phase, input feature maps have been transformed into new feature maps we call intermediate feature maps which would be sent to reweighing. About the number of channels of those maps, it seems it should be as large as possible to best preserve information of the input ones; however, setting it too large will increase number of parameters to learn in reweighing phase, and experiments have shown that keeping the number of channels unchanged would be enough.

Runtime Analysis. In theory, our method should be faster due to its sparsity nature. However, the convolution operation we use to implement our method in the experiments are designed for sparse matrices in terms of forward, backward ans storage; thus the significant improvement of runtime is not observed in our pilot experiments. How to modify this operation would be our further work.

Relation with Other Methods. In [6], a vanilla convolution layer is decomposed into a channel-wise convolution and a point-wise convolution layer to save parameters. In [8], fixed kernels are constructed by a Bernoulli distribution of $(1, -1, 0)$. Our method shares certain similarity with these two methods, but our goals and approaches are different:

(1) We present a drop-in substitute that use the sparse nature of LB-Conv kernels and paddings in deconvolution which is not a approximation to deconvolution layers differencing from LB-Conv in [8]. And we do not intend to make our method an approximated deconvolution layer but a regularized one. See Sect. 3.2 for a regularization view.
(2) We aim to accelerate networks with sparsity which is different from [6].

3.2 A Regularization View

In this section, we interpret our method in a regularization perspective.
A deconvolution layer can be written as

$$Vec(Y)_{deconv} = K^T Vec(X) \tag{3}$$

where K^T is the kernel matrix used in deconvolution. In a S-Deconv setting, we would have

$$Vec(Y)_{our} = VB^T Vec(X) \tag{4}$$

where B is a fixed LB-Conv kernel matrix and V is a learnable 1×1 convolution kernel matrix. In a regularization perspective, we can see that we add regularization to kernels in standard deconvolution layers forcing their corresponding matrix K^T to be decomposed as matrix multiplication of V and B^T.

4 Experiments

To verify the feasibility of our method S-Deconv, we conduct several experiments of image generation on different datasets. We also show Inception scores to compare qualities of images generated by different generators used in our experiments.

4.1 Datasets and Preprocessing

Datasets used in our experiments are summarized as follows:

(1) Fashion-MNIST [12] is a Fashion version of MNIST dataset only with more complex data structure. It was proposed to be direct replacement for MNIST for benchmarking. We will focus on this dataset in terms of computing Inception scores since it is complex enough to show that our method can also work well on other datasets while not too complex that we have enough computational resources to obtain Inception scores with Monte Carlo method.
(2) CelebA [11] is a large scale real life dataset of face attributes, we use its face images in our experiments; since it's not a label dataset, we cannot report Inception scores.
(3) Anime-Faces[1] contains 143,000 anime styled images with more than 100 tags (attributes).

As for preprocessing, we resized all images into 64×64, and centralized them. Resizing may add extra difficulties but it can save us a lot of time reorganizing models.

[1] Dataset obtained from https://github.com/jayleicn/animeGAN.

4.2 Architectures and Hyperparameters Settings

We used a WGAN [1] with a five deconvolution layers generator and a mirrored discriminator as our baseline model. Batch Normalization layers [7] are used. Modifications made to verify the feasibility of our approach are replacing deconvolution layers with S-Deconv ones. Details of this architecture is presented in Fig. 1, where arrows indicate the directions of information flowing.

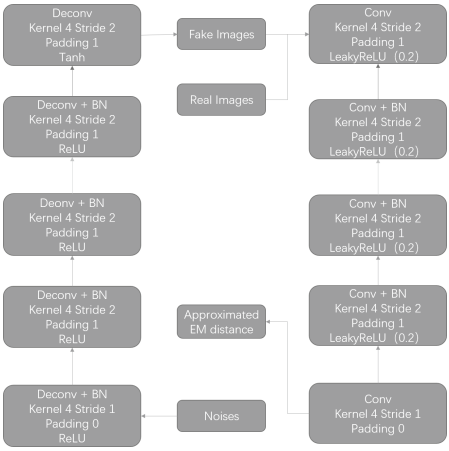


Fig. 1. Architecture of the baseline WGAN

All hyperparameters are kept unchanged for the same datasets in our experiments to avoid cherry-picking. We pay no attention to adjusting those hyperparameters to create very smart results. In contrary, we use some default settings that can be applied to many models and datasets to achieve not state-of- the-art but satisfying enough results to show that those models work. Details of those parameters are presented in Table 2[2]. The classifier used to compute Inception scores is built and trained by ourselves, the accuracy of which is a leaderboard performance.

4.3 Results

Feasibility. We focus on the results, Inception scores especially, of experiments on Fashion-MNIST dataset and show those scores in Table 3. Exact value of Inception score indicates nothing, and to show a clearer idea of how well models performed, we also compute the Inception scores of pure noises and real images.

From Table 3, we observe that by replacing a few deconvolution layers in our baseline model, the resulting models can actually generate images with higher

[2] Internal Coefficients: Ratio of internal channels and input channels in a SLBP layer. Density: Controlling the sparsity of SLBP layers.

Table 2. Hyperparameter sittings for three datasets

Hyperparameter	Fashion-Mnist	CelebA	Anime
Image size	64×64	64×64	64×64
Dimension of noise	100	200	200
Optimizer	RMSprop	RMSprop	RMSprop
Learning rate	0.0002	0.00005	0.00005
Beta	0.5	0.5	0.5
Batch size	64	16	16
Clipping	0.01	0.01	0.01
Basic channels for generators	64	128	128
Basic channels for discriminators	64	64	64
Internal coefficients	1	1	1
Density	0.5	0.5	0.5

Table 3. Inception scores

Source of evaluated images	Inception score
Pure nosies	3.02
Real images	9.67
Baseline model	6.6364
Model−1	6.9164
Model−1−2	6.6427
Model-1−2−3	6.7098
Model-1−2−3−4	1.4297
Model-1−2−3−4−5	1.0000

Inception scores which verifies the feasibility of our method. And there is a dramatic drop after $Model-1-2-3$ where the term $Model-1-2-3$ means the first three deconvolution layers has been replaced. This may be caused by over-regularization. Models with four or five S-Deconv layers would a small capacity since their learnable parameters are of a small number. Such capacities may not be enough to learn a complex image distribution.

Effects of Regularization. Experimental results in Table 3 verify the feasibility of our proposed S-Deconv method. Now we further investigate the regularization effect of different combinations of S-Deconv layers (in total $2^5 = 32$ cases). Table 4 shows Inceptions scores under different settings where array (0, 1, 0, 1, 0) means the second and fourth deconvolution layers being replaced by SLBP layers. Note that over regularization (4 or 5 layers replaced, 2 or 3 top layers

Table 4. More inception scores

Model	Inception score	Model	Inception score
0 0 0 0 0	6.6364	0 0 0 0 1	1.0000
0 0 0 1 0	6.8365	0 0 0 1 1	1.0000
0 0 1 0 0	6.7918	0 0 1 0 1	1.5593
0 0 1 1 0	6.5145	0 0 1 1 1	1.0000
0 1 0 0 0	1.0457	0 1 0 0 1	6.3385
0 1 0 1 0	6.4304	0 1 0 1 1	1.5357
0 1 1 0 0	6.8262	0 1 1 0 1	1.0000
0 1 1 1 0	1.0956	0 1 1 1 1	1.0000
1 0 0 0 0	6.9164	1 0 0 0 1	6.5662
1 0 0 1 0	1.0071	1 0 0 1 1	6.1790
1 0 1 0 0	4.7278	1 0 1 0 1	1.0000
1 0 1 1 0	1.0000	1 0 1 1 1	5.2147
1 1 0 0 0	6.6427	1 1 0 0 1	1.0075
1 1 0 1 0	1.0007	1 1 0 1 1	5.0681
1 1 1 0 0	6.7098	1 1 1 0 1	5.3925
1 1 1 1 0	1.4297	1 1 1 1 1	1.0000

replaced) would lead to catastrophic results since networks under such regularizations would not have sufficient model capacity to learn complex distributions.

Generated Images. We show some samples of images generated by $Model-1$ and Baseline model as a comparison on all three datasets using hyperparameter

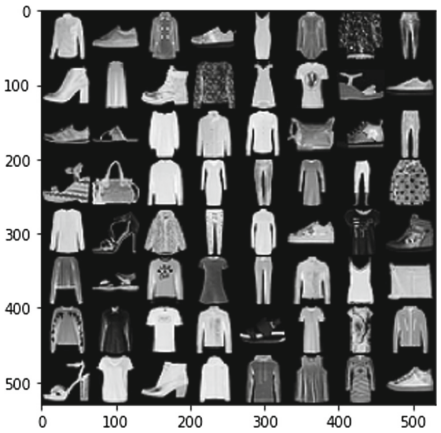


Fig. 2. Real Fashion-MNIST images

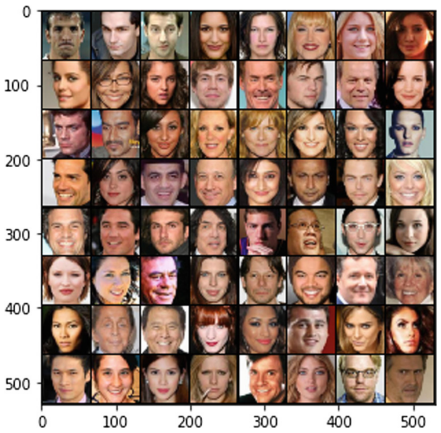


Fig. 3. Real CelebA images

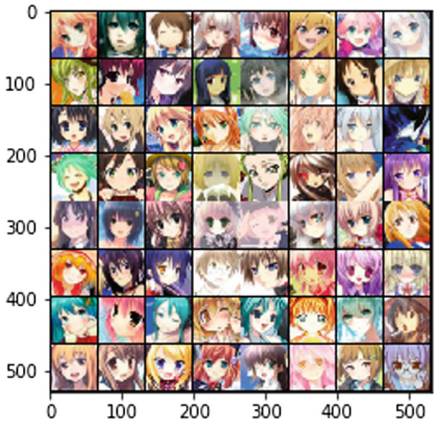
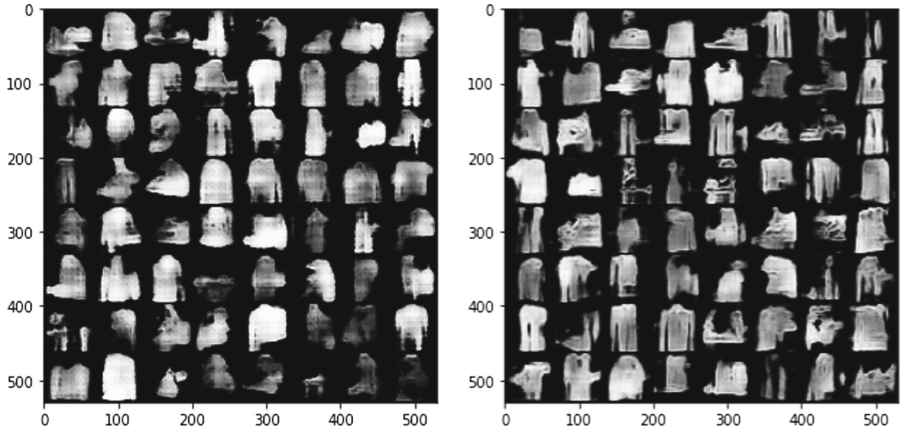


Fig. 4. Real Anime-Faces images

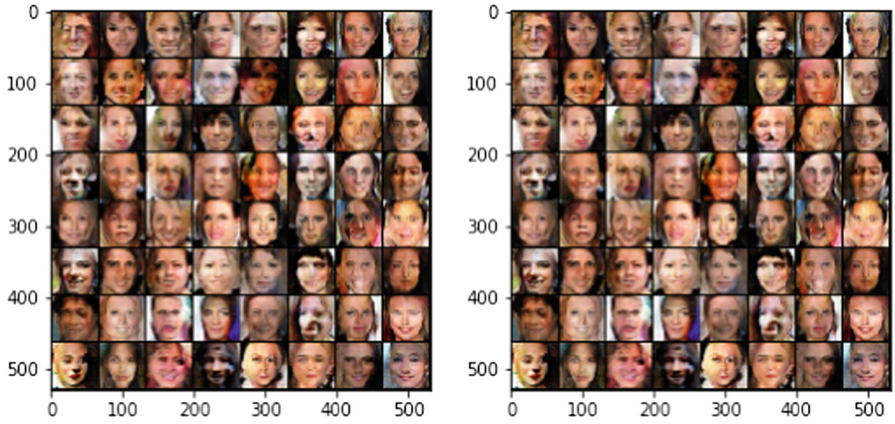


(a) Generated by Baseline Model (b) Generated by Model-1

Fig. 5. Generated Fashion-MNIST images

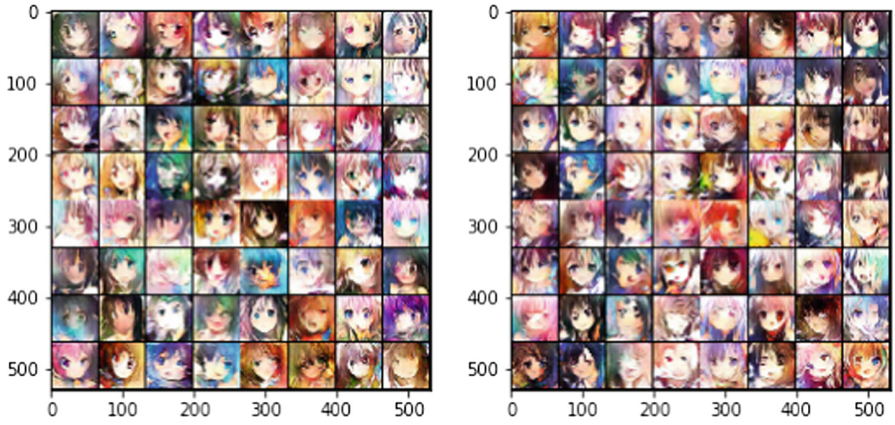
settings described in Sect. 4.2. All images were generated during training with no hand-picking. Real images are provided for evaluation and comparison with generated images shown in Figs. 5, 6 and 7.

We can see that on Fashion-MNIST dataset, model equipped with our method clearly outperforms the baseline model as indicated by Inception Scores in Table 3. As for CelebA and Anime-Faces datasets, we believe that the two models perform very similarly.



(a) Generated by Baseline Model (b) Generated by Model-1

Fig. 6. Generated CelebA faces



(a) Generated by Baseline Model (b) Generated by Model-1

Fig. 7. Generated Anime Faces

5 Discussion

We have proposed S-Deconv, a sparse substitute for deconvolution layers in GANs settings, and have shown that our method is feasible in practice. In experiments, we use WGAN as our baseline model and make modifications only on generators since that WGAN add extra constraints to discriminators. However, in other GAN architectures, such constraints are not needed, which means that we can replace convolution layers in discriminators to further reduce parameters. As we all know, neural networks are, in most cases, over-parameterized and the search in parameter space is constrained by network structures and guided by

optimizers using SGD-based algorithms. Thus, to obtain better search results, we can

(1) design architectures that encode human pilot knowledge;
(2) design building blocks that are less redundant than stand deep learning layers;
(3) design new optimization algorithms that can make full use of information contained in data.

In this paper, our method should be classed into option (1) or option (2) since a S-Deconv layer is less redundant than a standard deconvolution layer that it replaces and can be considered as two layers (a deconvolution one with fixed kernels and a convolution one with 1×1 kernel size) taken place in sequence.

Development of deconvolution operation implemented with sparse matrix multiplication is our further work.

References

1. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein gan. arXiv preprint arXiv:1701.07875 (2017)
2. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: DeepLab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. arXiv preprint arXiv:1606.00915 (2016)
3. Denton, E.L., Chintala, S., Fergus, R., et al.: Deep generative image models using a Laplacian pyramid of adversarial networks. In: Advances in Neural Information Processing Systems, pp. 1486–1494 (2015)
4. Ghosh, A., Kulharia, V., Namboodiri, V., Torr, P.H., Dokania, P.K.: Multi-agent diverse generative adversarial networks. arXiv preprint arXiv:1704.02906 (2017)
5. Goodfellow, I., et al.: Generative adversarial nets. In: Advances in Neural Information Processing Systems, pp. 2672–2680 (2014)
6. Howard, A.G., et al.: MobileNets: efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)
7. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167 (2015)
8. Juefei-Xu, F., Boddeti, V.N., Savvides, M.: Local binary convolutional neural networks. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), vol. 1 (2017)
9. Li, J., Monroe, W., Shi, T., Ritter, A., Jurafsky, D.: Adversarial learning for neural dialogue generation. arXiv preprint arXiv:1701.06547 (2017)
10. Liu, L., Lu, Y., Yang, M., Qu, Q., Zhu, J., Li, H.: Generative adversarial network for abstractive text summarization. arXiv preprint arXiv:1711.09357 (2017)
11. Liu, Z., Luo, P., Wang, X., Tang, X.: Deep learning face attributes in the wild. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 3730–3738 (2015)
12. Xiao, H., Rasul, K., Vollgraf, R.: Fashion-MNIST: a novel image dataset for benchmarking machine learning algorithms. arXiv preprint arXiv:1708.07747 (2017)
13. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. arXiv preprint arXiv:1703.10593 (2017)