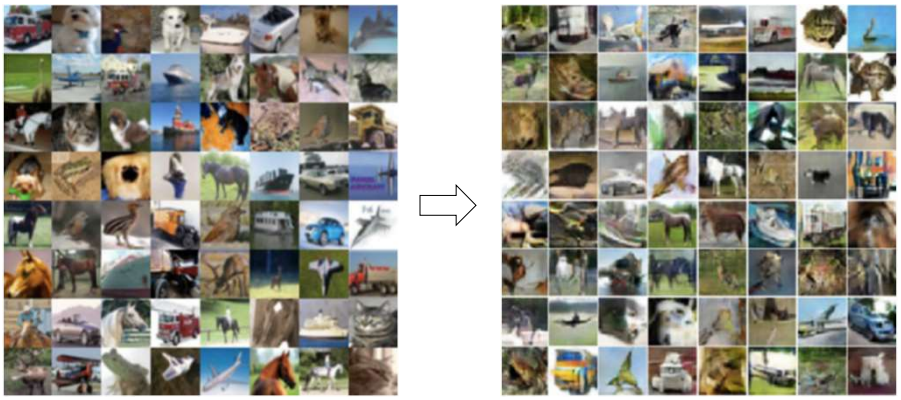
Introduction to Generative Models - Generative Adversarial Networks (GANs)

Dr. Dileep A. D.

Associate Professor,
Multimedia Analytics Networks And Systems (MANAS) Lab,
School of Computing and Electrical Engineering (SCEE),
Indian Institute of Technology Mandi, Kamand, H.P.
Email: addileep@iitmandi.ac.in

Generative Models

- The term "generative model" is used in many different ways
- We focus on generative models that generate new samples from the same distribution as given training data



Training data ~ $p_{\text{data}}(\mathbf{X})$ Generated samples~ $p_{\text{model}}(\mathbf{X})$

2

Generative Models

- The term "generative model" is used in many different ways
- We focus on generative models that generate new samples from the same distribution as given training data
- Generative models take a training set, consisting of samples drawn from a distribution $p_{\text{data}}(\mathbf{X})$, and learns to represent an estimate of that distribution, $p_{\text{model}}(\mathbf{X})$
- In some cases, the model estimates $p_{\text{model}}(\mathbf{X})$ explicitly
  - Generative models explicitly model the joint probability distribution or conditional probability distribution
  - The abstraction and generation of sample happening on the explicitly computed distributions
  - Examples: PixelRNN, variational autoencoder (VAE), restricted Boltzmann machine (RBM) etc.

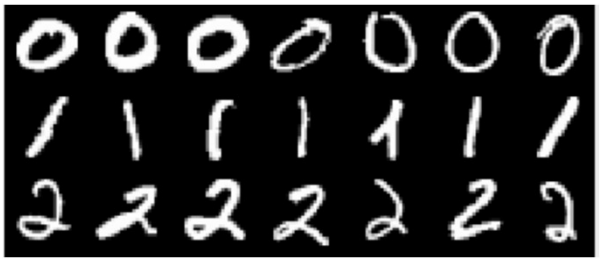
3

Generative Models

- The term "generative model" is used in many different ways
- We focus on generative models that generate new samples from the same distribution as given training data
- Generative models take a training set, consisting of samples drawn from a distribution $p_{\text{data}}(\mathbf{X})$, and learns to represent an estimate of that distribution, $p_{\text{model}}(\mathbf{X})$
- In other cases, the model is only able to generate samples from $p_{\text{model}}(\mathbf{X})$
  - Learn model that can sample from $p_{\text{model}}(\mathbf{X})$ without explicitly computing it
  - Generative adversarial networks (GANs) focus primarily on sample generation without caring about the what $p_{\text{data}}(\mathbf{X})$ actually is

4

Introduction to GANs

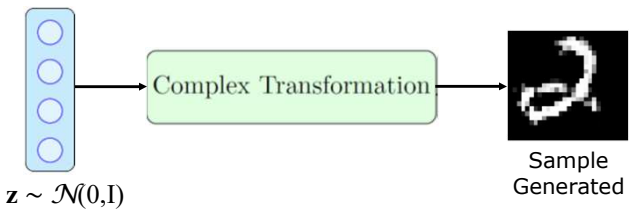


- We are given some training data (say, MNIST images) which obviously comes from some underlying distribution
- Our goal is to generate more images from this distribution (i.e., create images which look similar to the images from the training data)
- In other words, we want to sample from a complex high dimensional distribution which is intractable
  - RBMs and VAE models deal with this intractability in their own way

5

Introduction to GANs

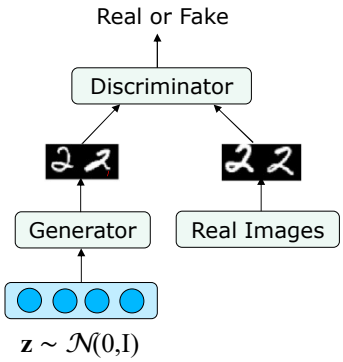
- GANs take a different approach to this problem
- GAN completely bypass the process of learning $p(\mathbf{X})$ i.e. no explicit $p(\mathbf{X})$ is learnt
- The idea is to sample from a simple tractable distribution (say, $\mathbf{z} \sim \mathcal{N}(0,\mathrm{I})$) and then learn a complex transformation from this to the training distribution



- In other words, take a $\mathbf{z} \sim \mathcal{N}(0,\mathrm{I})$, learn to make a series of complex transformations on it so that the output looks as if it came from training distribution
- Neural networks is used to model that complex transformation

6

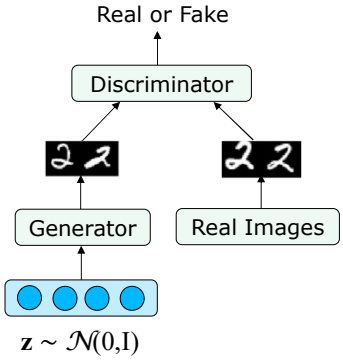
Introduction to GANs



- Neural networks is used to model that complex transformation
- Train the neural networks using two player game
- There are two players in the game:
  - Generator
  - Discriminator
- The job of the generator is to produce images which look so natural that the discriminator thinks that the images came from the real data distribution
- The job of the discriminator is to get better and better at distinguishing between true images and generated (fake) images

7

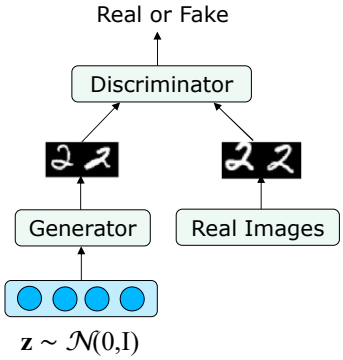
Introduction to GANs



- Let us define the problem formally
- Let $G_\varphi(.)$ be the generator and $D_\theta(.)$ be the discriminator
  - φ and θ are the parameters of $G_\varphi(.)$ and $D_\theta(.)$, respectively
- We have a neural network based generator which takes as input a noise vector $\mathbf{z} \sim \mathcal{N}(0,\mathrm{I})$ and produces $G_\varphi(\mathbf{z})$ [generated image]
- We have a neural network based discriminator which could take as input a real image $\mathbf{X}$ or a generated image $G_\varphi(\mathbf{z})$ and classify the input as real (1) or fake (0)

8

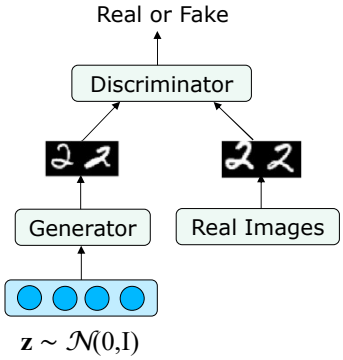
Introduction to GANs



- Let us define the objective function of overall network
- Generator and discriminator has contradicting objective functions
  - One is minimizing and other is maximizing

9

GANs: Objective Function of Generator



- Given an image generated by the generator as $G_\varphi(\mathbf{z})$ the discriminator assigns a score $D_\theta(G_\varphi(\mathbf{z}))$ to it
- This score will be between 0 and 1 and will tell us the probability of the image being real or fake
- Generator wishes $D_\theta(G_\varphi(\mathbf{z}))$ should be as high as possible (close to 1)
- For a given $\mathbf{z}$, the generator would want to maximize the log-likelihood of $D_\theta(G_\varphi(\mathbf{z}))$

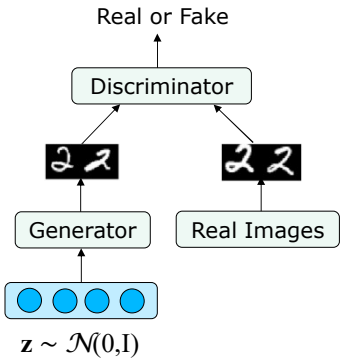
$$\text{maximise } \log D_\theta(G_\varphi(\mathbf{z}))$$

- It is also same as minimize the log-likelihood of $(1\text{-}\, D_\theta(G_\varphi(\mathbf{z})))$

$$\text{minimise } \log\left(1 - D_\theta(G_\varphi(\mathbf{z}))\right)$$

10

GANs: Objective Function of Generator



- This is just for a single $\mathbf{z}$ and the generator would like to do this for all possible values of $\mathbf{z}$
- For example, if $\mathbf{z}_n$ was discrete and drawn from a uniform distribution, i.e., $p(\mathbf{z}) = \frac{1}{N}$, then the generator's objective function would be

$$\underset{\varphi}{\text{minimise}} \sum_{n=1}^{N} \frac{1}{N} \log\left(1 - D_\theta(G_\varphi(\mathbf{z}_n))\right)$$

  - where N is the number of $\mathbf{z}$'s sampled from the distribution

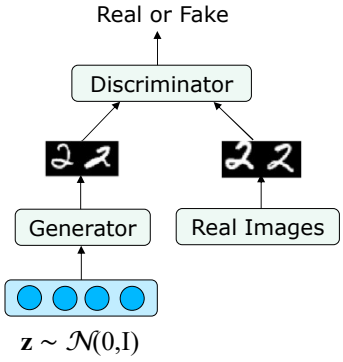
- However, in our case, $\mathbf{z}$ is continuous and not uniform ($\mathbf{z} \sim \mathcal{N}(0,\mathrm{I})$) so the equivalent objective function would be

$$\underset{\varphi}{\text{minimise}} \int_{n=1}^{N} p(\mathbf{z}_n) \log\left(1 - D_\theta(G_\varphi(\mathbf{z}_n))\right)$$

$$\underset{\varphi}{\text{minimise}}\, \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z})}\left[\log\left(1 - D_\theta(G_\varphi(\mathbf{z}))\right)\right]$$

11

GANs: Objective Function of Discriminator



- The task of the discriminator is to assign a high score to real images and a low score to generated (fake) images
- And it should do this for all possible real images and all possible generated (fake) images
- For every possible $\mathbf{z}$ maximise the score assigned to real images ($D_\theta(\mathbf{X})$)

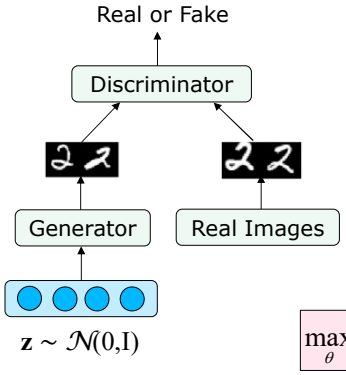
$$\text{maximise } \log D_\theta(\mathbf{X})$$

- For every possible $\mathbf{z}$ minimise the score assigned to generated (fake) images ($D_\theta(G_\varphi(\mathbf{z}))$)
- It is same as maximise one minus the score assigned to generated (fake) images ($D_\theta(G_\varphi(\mathbf{z}))$)

$$\text{maximise } \log(1 - D_\theta(G_\varphi(\mathbf{z})))$$

12

GANs: Objective Function of Discriminator

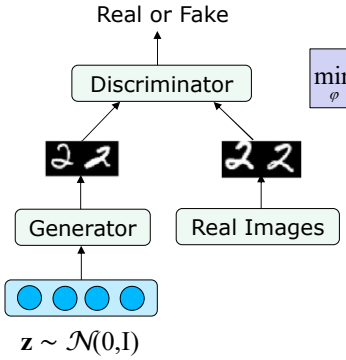


- The task of the discriminator is to assign a high score to real images and a low score to generated (fake) images
- And it should do this for all possible real images and all possible generated (fake) images
- Overall, it should maximize the following objective function

$$\max_{\theta} \mathbb{E}_{\mathbf{X}\sim p_{data}}\left[\log D_{\theta}(\mathbf{X})\right] + \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z})}\left[\log\left(1 - D_{\theta}(G_{\varphi}(\mathbf{z}))\right)\right]$$

13

Objective Function of GAN



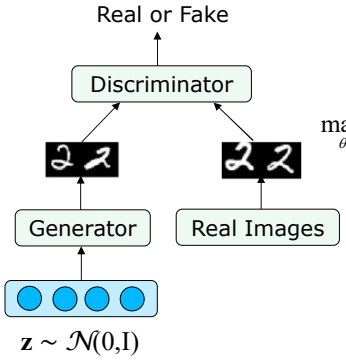
- If we put the objectives of the generator and discriminator together we get a minimax game

$$\min_{\varphi}\max_{\theta}\left(\mathbb{E}_{\mathbf{X}\sim p_{data}}\left[\log D_{\theta}(\mathbf{X})\right] + \mathbb{E}_{\mathbf{z}\sim p(\mathbf{z})}\left[\log\left(1 - D_{\theta}(G_{\varphi}(\mathbf{z}))\right)\right]\right)$$

- The first term in the objective is only w.r.t. the parameters of the discriminator (θ)
- The second term in the objective is w.r.t. the parameters of the generator (φ) as well as the discriminator (θ)
- The discriminator wants to maximize the second term whereas the generator wants to minimize it (hence it is a two-player game)

14

Training the GAN



- The overall training proceeds by alternating between these two step
- **Step1**
  - Gradient Ascent on Discriminator

$$\max_{\theta}\left(\mathbb{E}_{\mathbf{x}\sim p_{data}}\left[\log D_{\theta}(\mathbf{X})\right]+\mathbb{E}_{\mathbf{z}\sim p(\mathbf{z})}\left[\log\left(1-D_{\theta}(G_{\varphi}(\mathbf{z}))\right)\right]\right)$$

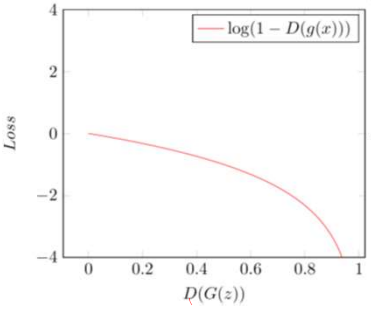
- **Step2**
  - Gradient Descent on Generator

$$\min_{\varphi}\left(\mathbb{E}_{\mathbf{z}\sim p(\mathbf{z})}\left[\log\left(1-D_{\theta}(G_{\varphi}(\mathbf{z}))\right)\right]\right)$$

- In practice, the above generator objective does not work well and we use a slightly modified objective

15

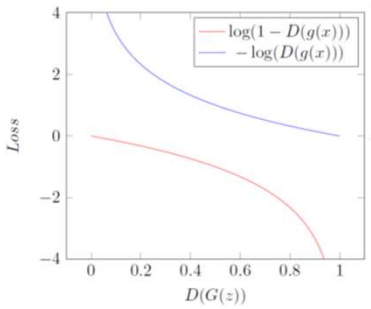
Training the GAN



- When the sample is likely fake (generated) image, we want to give a feedback to the generator (using gradients)
- However, in this region where $D_{\theta}(G_{\varphi}(\mathbf{z}))$ is close to 0,
  - the curve of the loss function is very flat and
  - the gradient would be close to 0
- Trick: Instead of minimizing the likelihood of the discriminator being correct, maximize the likelihood of the discriminator being wrong

16

Training the GAN



- When the sample is likely fake (generated) image, we want to give a feedback to the generator (using gradients)
- However, in this region where $D_\theta(G_\varphi(\mathbf{z}))$ is close to 0,
  - the curve of the loss function is very flat and
  - the gradient would be close to 0
- Trick: Instead of minimizing the likelihood of the discriminator being correct, maximize the likelihood of the discriminator being wrong
- In effect, the objective remains the same but the gradient signal becomes better

17

Algorithm for Training GANs

1: **procedure** GAN TRAINING
2: **for** number of training iterations **do**
3: **for** k steps **do**
4: Sample minibatch of m noise vectors $\mathbf{z}_1, \mathbf{z}_2,\ldots,\mathbf{z}_m$ from $\mathcal{N}(0,I)$
5: Sample minibatch of m examples $\mathbf{X}_1, \mathbf{X}_2,\ldots,\mathbf{X}_m$ from data generating distribution $p_{\text{data}}(\mathbf{X})$
6: Update the discriminator by ascending its stochastic gradient:

$$\nabla_\theta \frac{1}{m}\sum_{i=1}^{m}\left[\log D_\theta(\mathbf{X}_i) + \log\left(1 - D_\theta(G_\varphi(\mathbf{z}_i))\right)\right]$$

7: **end for**
8: Sample minibatch of m noise vectors $\mathbf{z}_1, \mathbf{z}_2,\ldots,\mathbf{z}_m$ from $\mathcal{N}(0,I)$
9: Update the generator by ascending its stochastic gradient

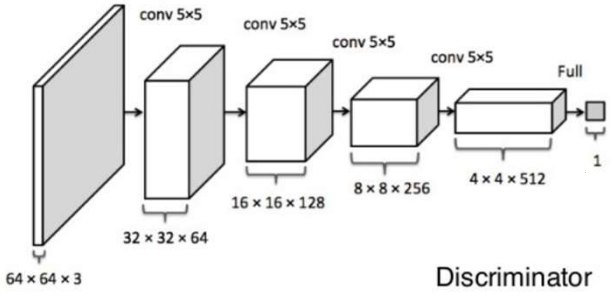
$$\nabla_\varphi \frac{1}{m}\sum_{i=1}^{m}\left[\log\left(D_\theta(G_\varphi(\mathbf{z}_i))\right)\right]$$

10: **end for**
11: **end procedure**

18

Deep Convolutional GANs (DCGAN)

- One of the popular architecture for GANs (i.e., neural networks used for the generator and discriminator)
- For discriminator, any CNN based classifier with 1 class (real) at the output can be used (e.g. VGG, ResNet, etc.)



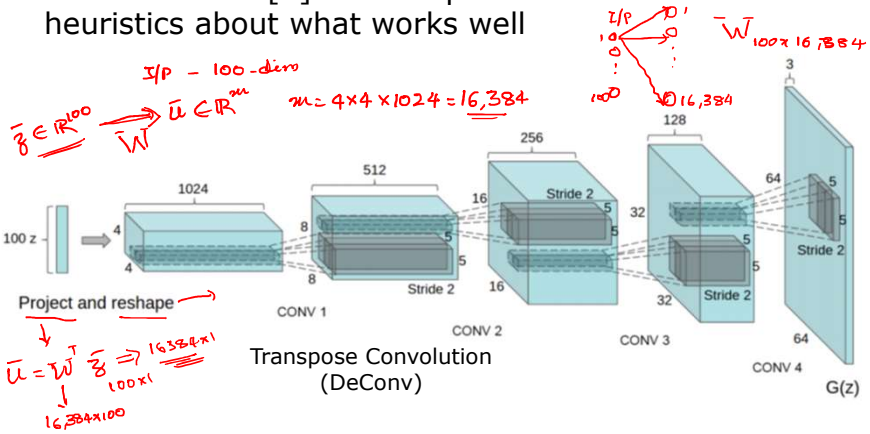
[1]

[1] Raymond A. Yeh, Chen Chen, Teck Yian Lim, Alexander G. Schwing, Mark Hasegawa-Johnson and Minh N. Do, "Semantic Image Inpainting with Deep Generative Models", in *Proceedings of the Computer Vision and Pattern Recognition (CVPR-2017)*, pp. 6882-6890, 2017.

19

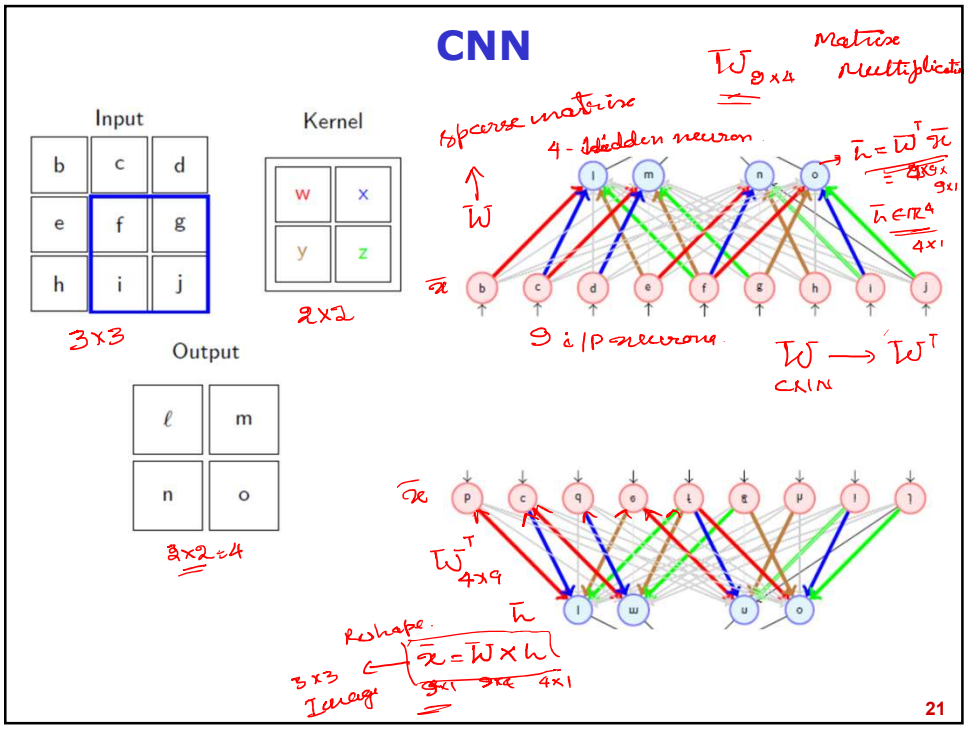
Deep Convolutional GANs (DCGAN)

- For generator, it is not straight forward
- Redford et al [2] came up with a architecture with heuristics about what works well



[2] Alec Radford, Luke Metz, Soumith Chintala, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks", in *Proceedings of 4th International Conference on Learning Representations (ICLR 2016)*, San Juan, Puerto Rico, May 2-4, 2016

20



21

Architecture Guidelines for Stable DCGANs

- Replace any pooling layers with strided convolutions in discriminator and fractional-strided convolutions (transpose convolution) in generator
- Use batchnorm in both the generator and the discriminator
- Remove fully connected hidden layers for deeper architectures
- Use ReLU activation in generator for all layers except for the output, which uses tanh
- Use LeakyReLU activation in the discriminator for all layers

22

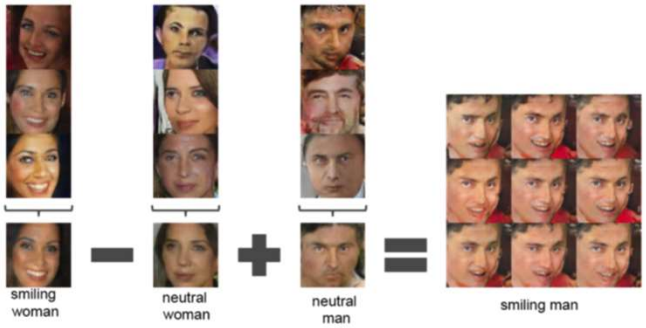
Illustrations: GANs

- Samples of images of bedrooms generated by a DCGAN trained on the LSUN dataset [2]

[2] Alec Radford, Luke Metz, Soumith Chintala, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks", in *Proceedings of 4th International Conference on Learning Representations (ICLR 2016)*, San Juan, Puerto Rico, May 2-4, 2016

23

Illustrations: GANs

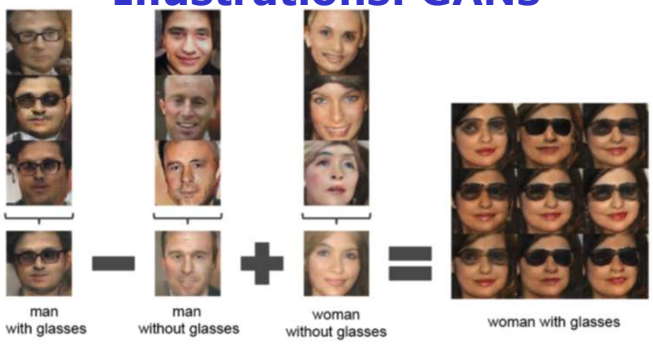


- The first 3 images in the first column were generated by feeding some $\mathbf{z}_{11}$, $\mathbf{z}_{12}$, $\mathbf{z}_{13}$ respectively as the input to the generator
- The fourth image, $\mathbf{z}_1$, was generated by taking an average of $\mathbf{z}_{11}$, $\mathbf{z}_{12}$, $\mathbf{z}_{13}$ and feeding it to the generator
- Similarly the average vectors $\mathbf{z}_2$ and $\mathbf{z}_3$ are obtained for the 2nd and 3rd columns
- If we do a simple vector arithmetic on these averaged vectors then we see the corresponding effect in the generated images

[2] Alec Radford, Luke Metz, Soumith Chintala, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks", in *Proceedings of 4th International Conference on Learning Representations (ICLR 2016)*, San Juan, Puerto Rico, May 2-4, 2016

24

Illustrations: GANs



- The first 3 images in the first column were generated by feeding some $\mathbf{z}_{11}$, $\mathbf{z}_{12}$, $\mathbf{z}_{13}$ respectively as the input to the generator
- The fourth image, $\mathbf{z}_1$, was generated by taking an average of $\mathbf{z}_{11}$, $\mathbf{z}_{12}$, $\mathbf{z}_{13}$ and feeding it to the generator
- Similarly the average vectors $\mathbf{z}_2$ and $\mathbf{z}_3$ are obtained for the 2nd and 3rd columns
- If we do a simple vector arithmetic on these averaged vectors then we see the corresponding effect in the generated images

[2] Alec Radford, Luke Metz, Soumith Chintala, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks", in *Proceedings of 4th International Conference on Learning Representations (ICLR 2016)*, San Juan, Puerto Rico, May 2-4, 2016

25

Applications

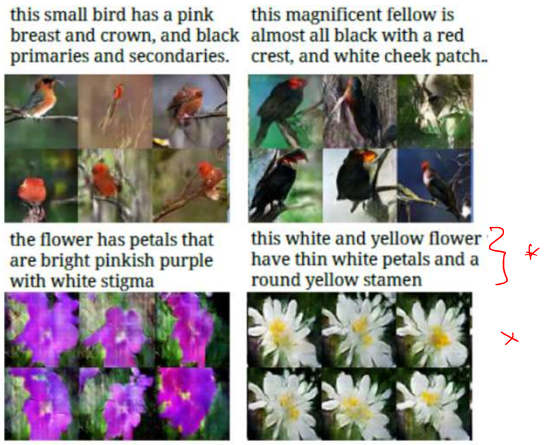
- Image to image translation [3]
  - Encompassing many kinds of transformations of an image: converting a satellite photo into a map, converting a sketch into a photorealistic image, etc.

[3] Isola, P., Zhu, J.-Y., Zhou, T., and Efros, A. A., Image-to-image translation with conditional adversarial networks, arXiv preprint arXiv:1611.07004. 2016

26

Applications

- Text-to-image synthesis with GANs [4]

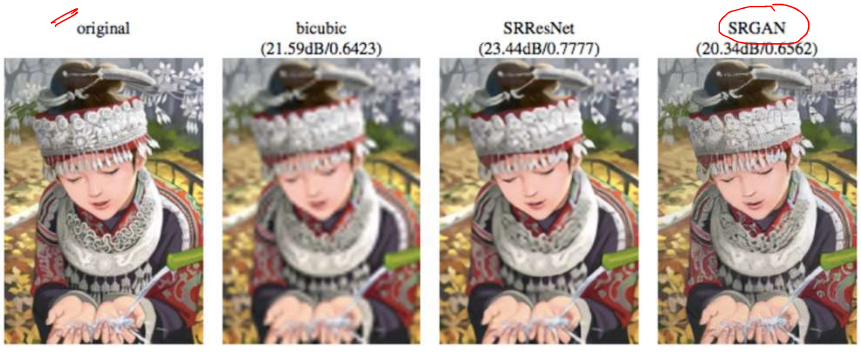


[4] Reed, S., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., and Lee, H., "Generative adversarial text to image synthesis", arXiv preprint, arXiv:1605.05396, 2016.

27

Applications

- Super-resolution [5]
  - The leftmost image is an original high-resolution image
  - It is then down sampled to make a low-resolution image, and different methods are used to attempt to recover the high-resolution image



[5] Ledig, C., Theis, L., Huszar, F., Caballero, J., Aitken, A. P., Tejani, A., Totz, J.,Wang, Z., and Shi, W. , Photo-realistic single image super-resolution using a generative adversarial network. CoRR, abs/1609.04802, 2016.

28

Applications

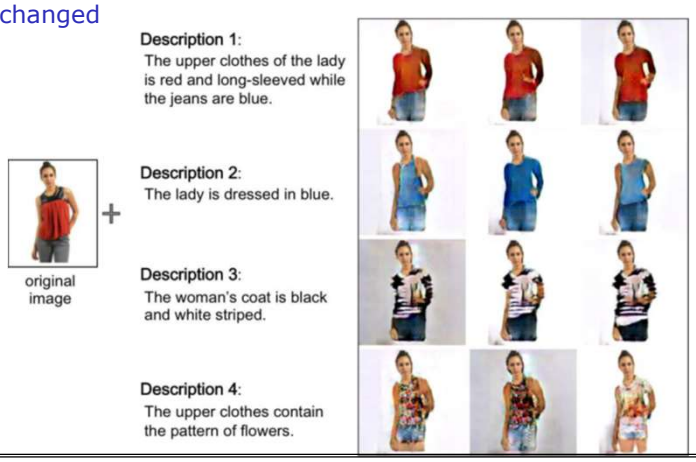
- Generating art [6]
  - Generating art by looking at art and learning about styles

[6] Elgammal, A., Liu, B., Elhoseiny, M. and Mazzone, M., CAN: Creative adversarial networks, generating" art" by learning about styles and deviating from style norms. *arXiv preprint arXiv:1706.07068*, 2017.

29

Applications

- Fashion Synthesis [7]
  - Given an input image of a person and a sentence describing a different outfit, the model "redresses" the person as desired, while at the same time keeping the wearer and her/his pose unchanged



[7] Zhu, S., Urtasun, R., Fidler, S., Lin, D. and Change Loy, C., Be your own prada: Fashion synthesis with structural coherence. In Proceedings of the IEEE international conference on computer vision, pp. 1680-1688, 2017

30

Summary: GAN

- Generative adversarial network (GAN) is a generative model
- It generate the samples close to the training distribution without explicitly learning it
- In other words, take a $\mathbf{z} \sim \mathcal{N}(0,\mathrm{I})$, learn to make a series of complex transformations on it so that the output looks as if it came from training distribution
  - Neural networks is used to model that complex transformation
- Neural networks are trained using two player game
  - Two players in the game are a generator and a discriminator
- The job of the generator is to produce images which look so natural that the discriminator thinks that the images came from the real data distribution
- The job of the discriminator is to get better and better at distinguishing between true images and generated (fake) images

31

Text Books

1. Ian Goodfellow, Yoshua Bengio and Aaron Courville, Deep learning, MIT Press, Available online: http://www.deeplearningbook.org, 2016
2. Charu C. Aggarwal, Neural Networks and Deep Learning, Springer, 2018
3. B. Yegnanarayana, Artificial Neural Networks, Prentice-Hall of India, 1999.
4. Satish Kumar, Neural Networks - A Class Room Approach, Second Edition, Tata McGraw-Hill, 2013.
5. S. Haykin, Neural Networks and Learning Machines, Prentice Hall of India, 2010.
6. C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.
7. J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, Third Edition, Morgan Kaufmann Publishers, 2011.
8. S. Theodoridis and K. Koutroumbas, *Pattern Recognition*, Academic Press, 2009.

32