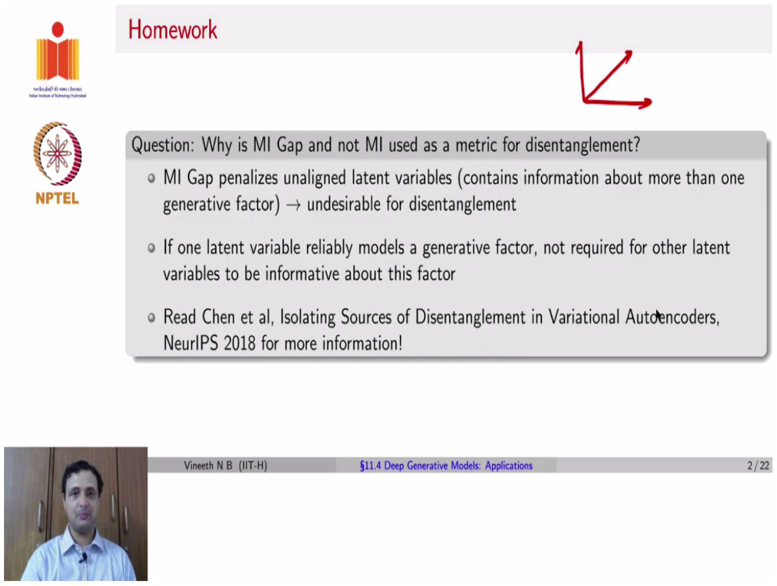
Deep Learning for Computer Vision
Professor. Vineeth N Balasubramanian
Department of Computer Science and Engineering
Indian Institute of Technology Hyderabad
Lecture 71
Deep Generative Models: Image Applications

In the remaining lectures of this week, we will see a few applications of GANs and Generative Models, images and videos. We will try to cover simple applications of GANs to different domains as well as a few adaptations of these Generative Models.

(Refer Slide Time: 00:39)



Homework

Question: Why is MI Gap and not MI used as a metric for disentanglement?

- MI Gap penalizes unaligned latent variables (contains information about more than one generative factor) → undesirable for disentanglement
- If one latent variable reliably models a generative factor, not required for other latent variables to be informative about this factor
- Read Chen et al, Isolating Sources of Disentanglement in Variational Autoencoders, NeurIPS 2018 for more information!

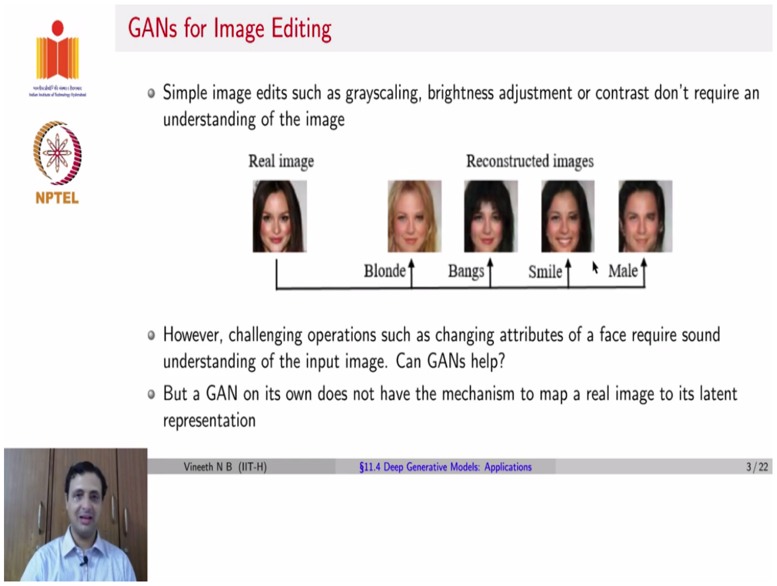
Vineeth N B (IIT-H) §11.4 Deep Generative Models: Applications 2 / 22

Before we go there, the question that we left behind from the last lecture, which was, why is Mutual Information Gap used and not just the mutual information in the metric that we discussed for disentanglement. Hope you had a chance to try finding this out. The Mutual Information Gap tries to penalize unaligned latent variables. Because we are looking at the gap between two latent variables, which have the highest mutual information with a generative factor.

So, if there is a latent, if there is a certain generative factor, which is a combination of two latent variables that would be penalized in a Mutual Information Gap, we would then get a 0 or a lower value, which indicates poorer disentanglement. The other answer is, if one latent variable models a generative factor, we do not need any other latent variable to model the same generative factor and if that happens, we would like to say that we have poorer disentanglement.

This also necessitates the definition of a gap, rather than just using the mutual information of the top latent variable with respect to the generative factor. More on this is there in the isolating sources of disentanglement in VAEs paper. Do read it if you are interested and need more information.

(Refer Slide Time: 02:27)



GANs for Image Editing

- Simple image edits such as grayscaling, brightness adjustment or contrast don't require an understanding of the image

- However, challenging operations such as changing attributes of a face require sound understanding of the input image. Can GANs help?
- But a GAN on its own does not have the mechanism to map a real image to its latent representation

Vineeth N B (IIT-H) §11.4 Deep Generative Models: Applications 3 / 22

Let us now come to a few applications of GANs. The first application that we will talk about is for Image Editing. Simple image edits, such as making an image grayscale or adjusting the brightness or contrast, do not require an understanding of the semantics inside the image. However, if we want to take a face image and make the hair black or add bangs or add a smile, make the image male, then you need to understand the semantic content in the image.

Can GANs help here? Possibly. However, GANs do not have an inherent mechanism to map an image to its latent representation. We saw VAEs can do it, but VAEs sometimes can suffer from reconstruction issues, as we discussed in the VAE GAN lecture. How do you use GANs to be able to achieve something like this?

(Refer Slide Time: 03:38)

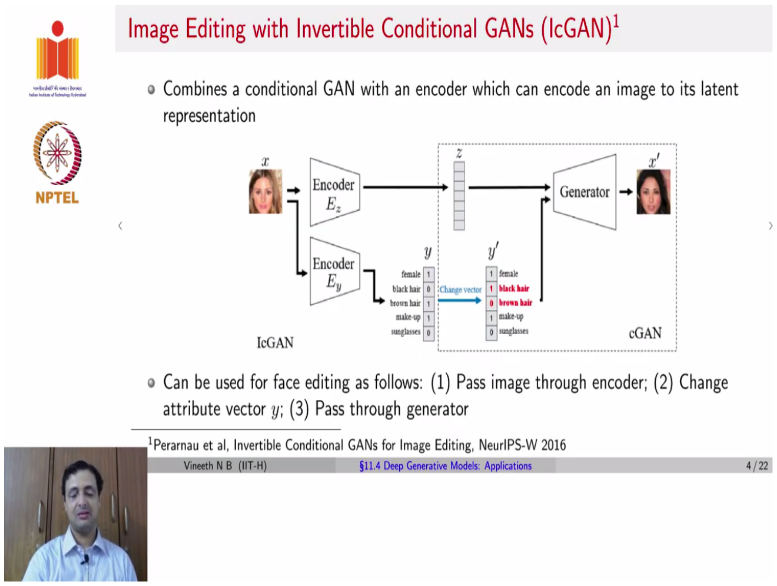
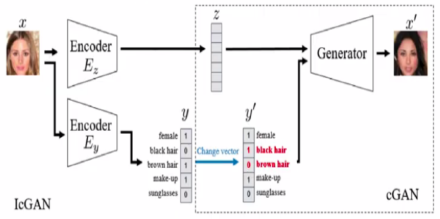


Image Editing with Invertible Conditional GANs (IcGAN)[1]

- Combines a conditional GAN with an encoder which can encode an image to its latent representation



- Can be used for face editing as follows: (1) Pass image through encoder; (2) Change attribute vector y; (3) Pass through generator

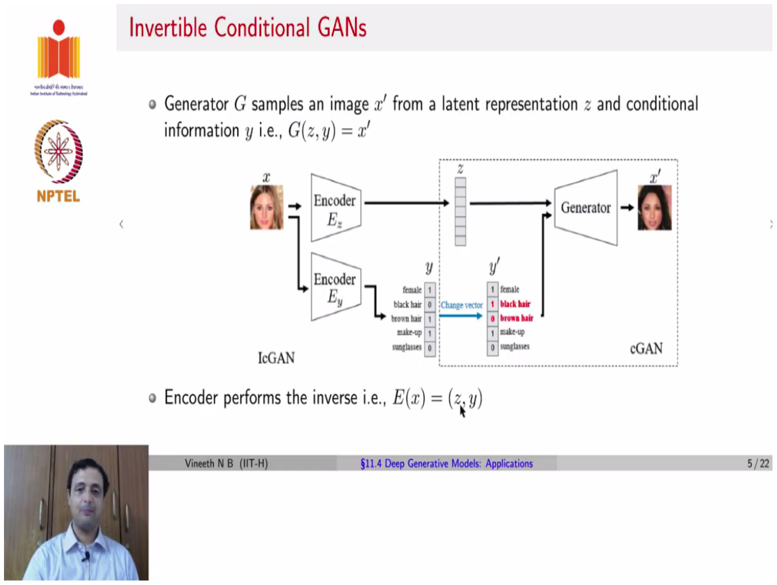
[1]Perarnau et al, Invertible Conditional GANs for Image Editing, NeurIPS-W 2016

Vineeth N B (IIT-H) §11.4 Deep Generative Models: Applications 4 / 22

This was proposed in a variant of GAN called Invertible Conditional GANs in NeurIPS workshop 2016 called IcGAN and the way IcGAN achieves this objective is using this architecture. It is a conditional GAN that is used, very similar to what we saw earlier this week of GANs across domains, a similar idea. But now there is also an encoder in the architecture, which can encode an image to its latent attributes.

So what is the overall architecture here? You have two encoders. One encoder gives a latent of an image. One encoder gives the attributes of an image. Now to be able to generate a new image with different attributes. You pass the image through the encoder, get an attribute vector y, change its attributes, change black hair from 0 to 1, change brown hair from 1 to 0 and now you get the same image with black hair.

(Refer Slide Time: 04:54)



Invertible Conditional GANs

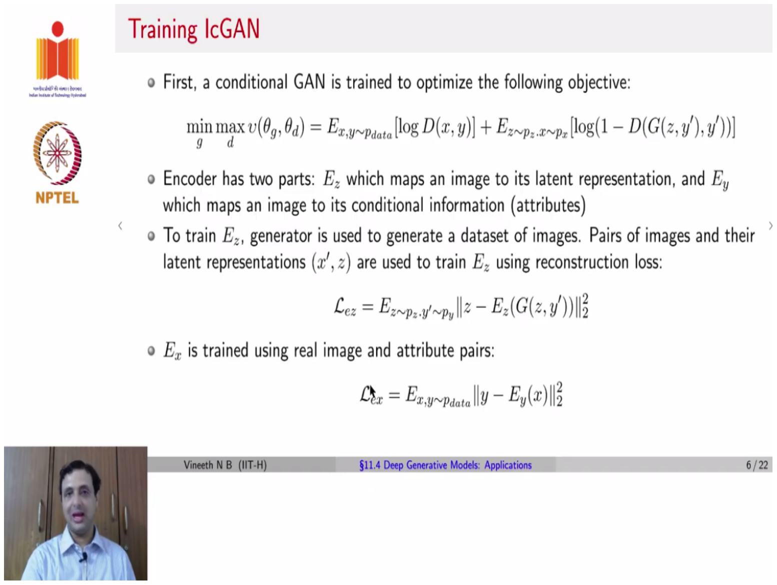
- Generator G samples an image x' from a latent representation z and conditional information y i.e., $G(z,y) = x'$

- Encoder performs the inverse i.e., $E(x) = (z, y)$

Vineeth N B (IIT-H) §11.4 Deep Generative Models: Applications 5 / 22

So how does this actually work? The generator G samples an image from a latent representation z and some conditional information y. So G is defined on$(z,\ y)$and gives us a generated image x'. The encoder performs the inverse, given an image x, it outputs$(z,\ y)$.

(Refer Slide Time: 05:22)



Training IcGAN

- First, a conditional GAN is trained to optimize the following objective:

$$\min_g \max_d v(\theta_g, \theta_d) = E_{x,y \sim p_{data}}[\log D(x,y)] + E_{z \sim p_z . x \sim p_x}[\log(1 - D(G(z,y'),y'))]$$

- Encoder has two parts: E_z which maps an image to its latent representation, and E_y which maps an image to its conditional information (attributes)
- To train E_z, generator is used to generate a dataset of images. Pairs of images and their latent representations (x', z) are used to train E_z using reconstruction loss:

$$\mathcal{L}_{ez} = E_{z \sim p_z . y' \sim p_y} \|z - E_z(G(z,y'))\|_2^2$$

- E_x is trained using real image and attribute pairs:

$$\mathcal{L}_{ex} = E_{x,y \sim p_{data}} \|y - E_y(x)\|_2^2$$

Vineeth N B (IIT-H) §11.4 Deep Generative Models: Applications 6 / 22

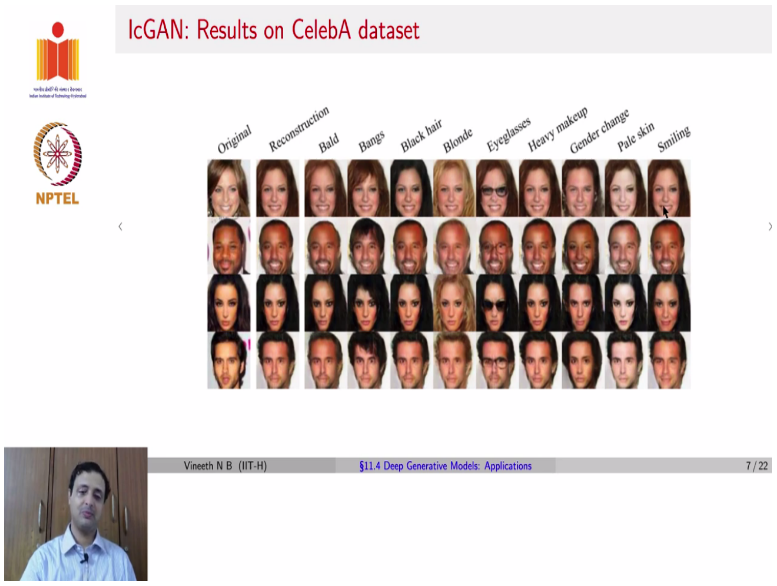
How do you train such a model? The broad objective is similar to what we saw with conditional GANs. This is what we saw with the earlier lecture on GANs across domains. So, you have a log likelihood of$(x,\ y)$the image and the attribute, which we would like to maximize, as given in an input training dataset and we would also like to look at the second objective in the in the GAN as $D(G(z,\ y')$, which gives you an x'.

The discriminator now looks at the tuple$(x',\ y')$and says whether it is real or fake. This is similar to conditional GAN, where we try to generate images from text. The encoder here has two parts, E_z and E_y one for the latent and one for the attributes. To train the E_z encoder, the generator is first trained and the generator is used to generate a dataset of images. Now, we have pairs of images and their corresponding latent representations as a dataset through the generator's performance.

That is now used to train the encoder. So, you have $G(z,\ y')$, which was the generated image and the corresponding z that was used to generate the image for the generator. Now, trying to minimize the L_2norm between these two or the mean square error between these two is used to train the encoder z. The encoder y, the other one, is now trained using the data itself.

We assume that the training dataset has data and attributes that, so, you now consider y to be the attributes that are provided in the dataset, $E_y(x)$to be the predicted attributes and you do a mean square error between these two to be able to train your E_y.

(Refer Slide Time: 07:42)

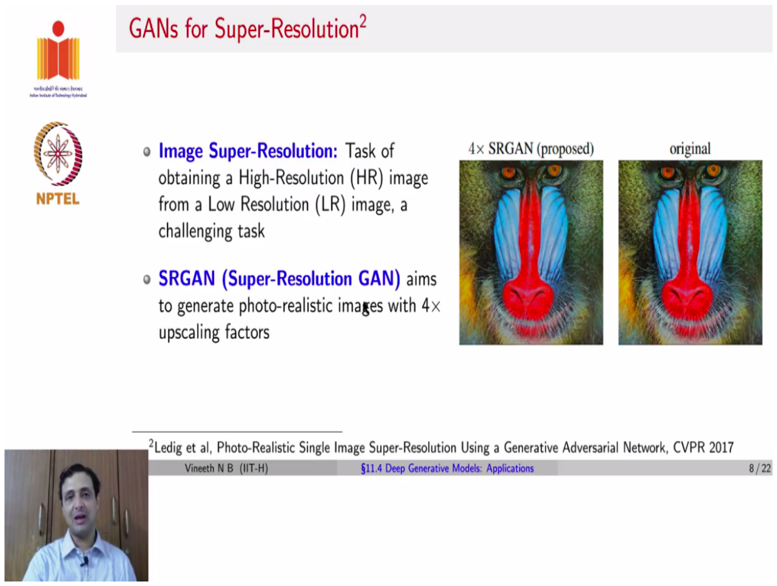


IcGAN: Results on CelebA dataset

Vineeth N B (IIT-H) §11.4 Deep Generative Models: Applications 7 / 22

Here are some results on a popular data set known as CelebA. So, you have an original image and by varying several attributes, you see that you get several reconstructions that are fairly realistic, several generations that are fairly realistic, where you see a balding person, the same image with bangs, same image with black hair, blonde, glasses, more makeup, gender change, pale skin, smiling, so on and so forth.

(Refer Slide Time: 08:18)



GANs for Super-Resolution[2]

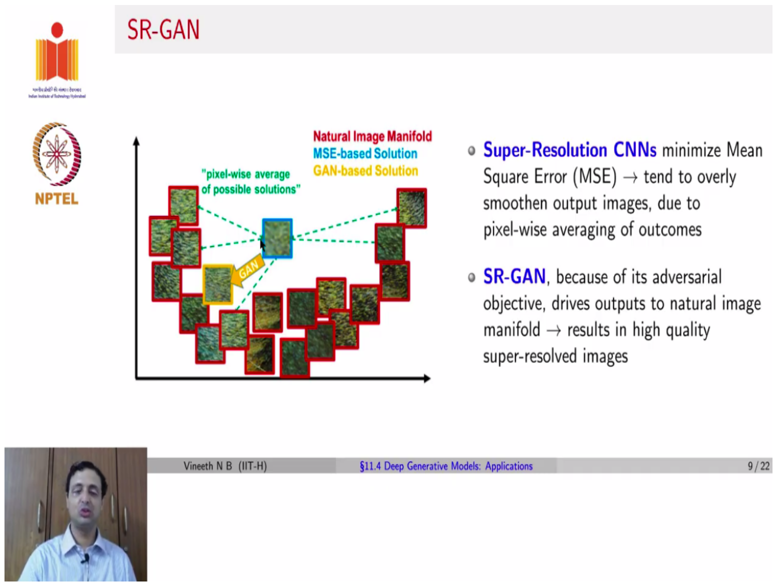
- **Image Super-Resolution:** Task of obtaining a High-Resolution (HR) image from a Low Resolution (LR) image, a challenging task
- **SRGAN (Super-Resolution GAN)** aims to generate photo-realistic images with 4× upscaling factors

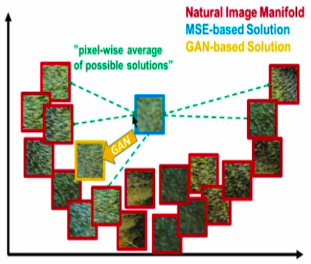
[2]Ledig et al, Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network, CVPR 2017

Vineeth N B (IIT-H) §11.4 Deep Generative Models: Applications 8 / 22

Another application of GANs is for a task known as Super-Resolution. We also talked about this in the context of CNNs. Let us revisit it now. Image Super Resolution is the task of obtaining a high resolution image from a low resolution image, which can be a challenging task, because you have to fill in information that was not available in the first place. SR-GAN or Super Resolution GAN aims to generate photo-realistic images with up to 4 x upscaling factors.

(Refer Slide Time: 09:01)


SR-GAN



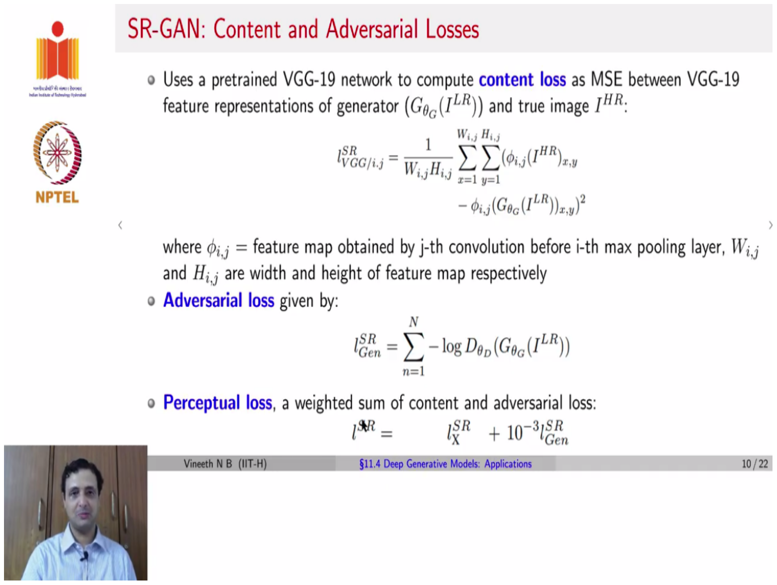
- **Super-Resolution CNNs** minimize Mean Square Error (MSE) → tend to overly smoothen output images, due to pixel-wise averaging of outcomes
- **SR-GAN**, because of its adversarial objective, drives outputs to natural image manifold → results in high quality super-resolved images

Vineeth N B (IIT-H) §11.4 Deep Generative Models: Applications 9 / 22

Why SR-GAN? Even CNNs can be used for Super-Resolution by doing deconvolution or upsampling through the second stage of a network through an encoder decoder architecture. In those architectures, the loss minimizes the mean square error between a generated output and the true ground truth high resolution image. Mean square error can tend to overly smoothen output images, because of the averaging effect that you get across all pixels.

So, because you try to take the mean squared error, it may not try to make specific pixels sharper, which may be important for Super-Resolution. GANs by using adversarial loss drives outputs closer to the natural image manifold. So, if this was the natural image manifold given by these red boxes, GANs try to push the generated image onto that manifold, which helps in getting high quality super-resolved images.

(Refer Slide Time: 10:18)



SR-GAN: Content and Adversarial Losses

- Uses a pretrained VGG-19 network to compute **content loss** as MSE between VGG-19 feature representations of generator $(G_{\theta_G}(I^{LR}))$ and true image I^{HR}:

$$l^{SR}_{VGG/i,j} = \frac{1}{W_{i,j}H_{i,j}} \sum_{x=1}^{W_{i,j}} \sum_{y=1}^{H_{i,j}} (\phi_{i,j}(I^{HR})_{x,y} - \phi_{i,j}(G_{\theta_G}(I^{LR}))_{x,y})^2$$

where $\phi_{i,j}$ = feature map obtained by j-th convolution before i-th max pooling layer, $W_{i,j}$ and $H_{i,j}$ are width and height of feature map respectively

- **Adversarial loss** given by:

$$l^{SR}_{Gen} = \sum_{n=1}^{N} -\log D_{\theta_D}(G_{\theta_G}(I^{LR}))$$

- **Perceptual loss**, a weighted sum of content and adversarial loss:

$$l^{SR} = l^{SR}_{X} + 10^{-3} l^{SR}_{Gen}$$

Vineeth N B (IIT-H) §11.4 Deep Generative Models: Applications 10 / 22

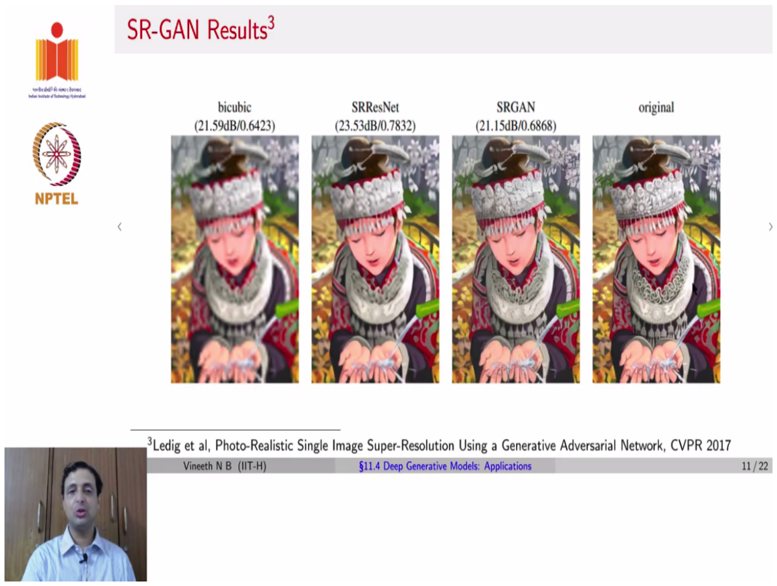
How is SR-GAN trained? SR-GAN has a couple of components. One of the losses is known as a content loss. The content loss minimizes the mean square error between the output of the generator which is given by G_{θ_G} and the true high resolution image. But instead of measuring the mean square error between the images themselves, it minimizes the mean square error between the feature representations of these two images.

What features? Both these images are sent through a VGG-19 network and an output of a certain convolutional layer in the VGG-19 network, that feature representation is taken and the mean square error between those representations are considered to minimize as a loss function for training the SR-GAN. In this particular case, $\phi_{i,j}$ corresponds to the feature map obtained by the jth convolution before the ith max pooling layer.

And $W_{i,j}$ and $H_{i,j}$ are the width and height of the feature map respectively. That is used for normalization and the second loss that is used to train SR-GAN is the standard adversarial loss, which looks at fooling the discriminator similar to every other GAN that we have seen so far.

The final loss used to train SR-GAN is a weighted sum of the content loss and the adversarial loss. In this particular case, it was shown that the adversarial loss can be weighted by a smaller quantity 10 power minus 3 in their case, to get good performance on Super-Resolution.

(Refer Slide Time: 12:21)

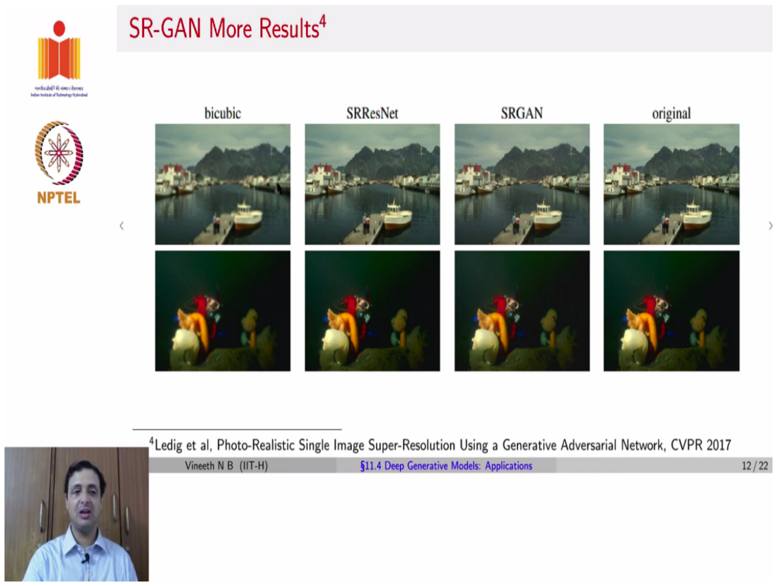

SR-GAN Results[3]

[3]Ledig et al, Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network, CVPR 2017

Vineeth N B (IIT-H) §11.4 Deep Generative Models: Applications 11 / 22

With this loss, here are some sample results. You can see that bicubic interpolation of this image gives us this super resolved image. Using a super-resolution CNN ResNet gives this image and using a GAN, gives a far sharper generation of the image when compared to the original.

(Refer Slide Time: 12:46)

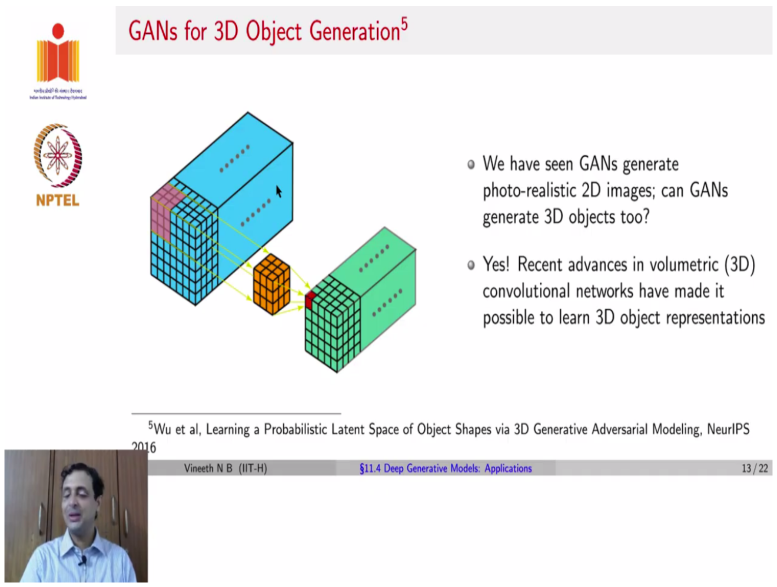

SR-GAN More Results[4]

[4]Ledig et al, Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network, CVPR 2017

Vineeth N B (IIT-H) §11.4 Deep Generative Models: Applications 12 / 22

More images on natural scenes, bicubic interpolation, SR-CNN with a ResNet and SR-GAN and the final original picture in high resolution. You can see that SR-GAN gets local details fairly more clearly, than super-resolution CNN and much better than bicubic interpolation.

(Refer Slide Time: 13:11)



GANs for 3D Object Generation[5]

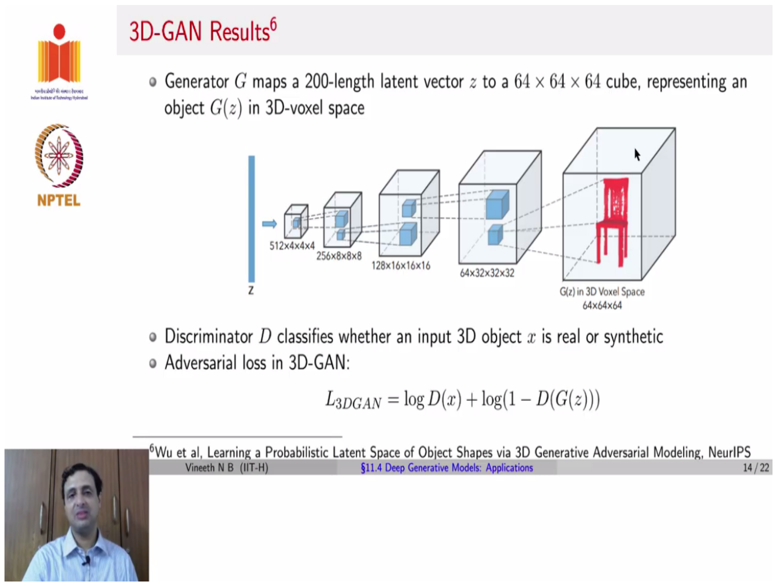
- We have seen GANs generate photo-realistic 2D images; can GANs generate 3D objects too?
- Yes! Recent advances in volumetric (3D) convolutional networks have made it possible to learn 3D object representations

[5] Wu et al, Learning a Probabilistic Latent Space of Object Shapes via 3D Generative Adversarial Modeling, NeurIPS 2016

Vineeth N B (IIT-H) §11.4 Deep Generative Models: Applications 13 / 22

The third application is GANs for 3D Object Generation. So far, we have seen GANs generate 2D images. Can GANs also generate 3D objects? Fairly straightforward. Recent advances with 3D CNNs and Volumetric Convolution Networks have made it possible to learn 3D Object representations. So, the layers in the generator have to now generate a volume. So as long as you can replace those layers, you can generate any object through a GAN framework.

(Refer Slide Time: 13:52)



3D-GAN Results[6]

- Generator G maps a 200-length latent vector z to a $64 \times 64 \times 64$ cube, representing an object $G(z)$ in 3D-voxel space

- Discriminator D classifies whether an input 3D object x is real or synthetic
- Adversarial loss in 3D-GAN:

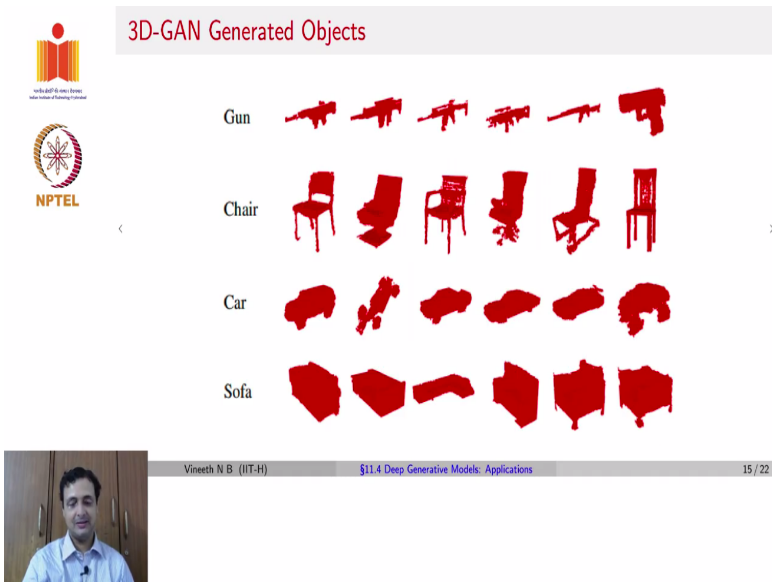
$$L_{3DGAN} = \log D(x) + \log(1 - D(G(z)))$$

[6] Wu et al, Learning a Probabilistic Latent Space of Object Shapes via 3D Generative Adversarial Modeling, NeurIPS

Vineeth N B (IIT-H) §11.4 Deep Generative Models: Applications 14 / 22

In this particular example, this work was published in NeurIPS of 2016. So, the 3D GAN, the way it works is the generator G maps a 200 length latent vector to a $64x64x64$cube, which represents an object $G(z)$in 3D voxel space. The discriminator classifies whether the object now is real or synthetic. So, only the architecture in the generator now is different from a standard GAN. What is the loss? Standard GAN loss. This is a straightforward application of GANs to 3D objects by changing the architecture of the generator to handle 3D volumes.

(Refer Slide Time: 14:47)

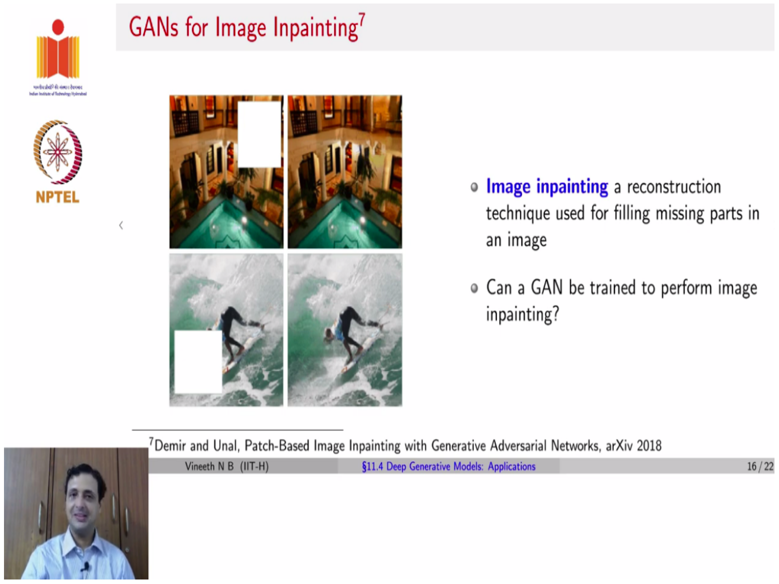


3D-GAN Generated Objects

Vineeth N B (IIT-H) §11.4 Deep Generative Models: Applications 15 / 22

And with that loss, when a GAN, is trained to generate 3D objects. You get fairly good generations of different kinds of 3D objects, as you see on this slide.

(Refer Slide Time: 15:06)



GANs for Image Inpainting[7]

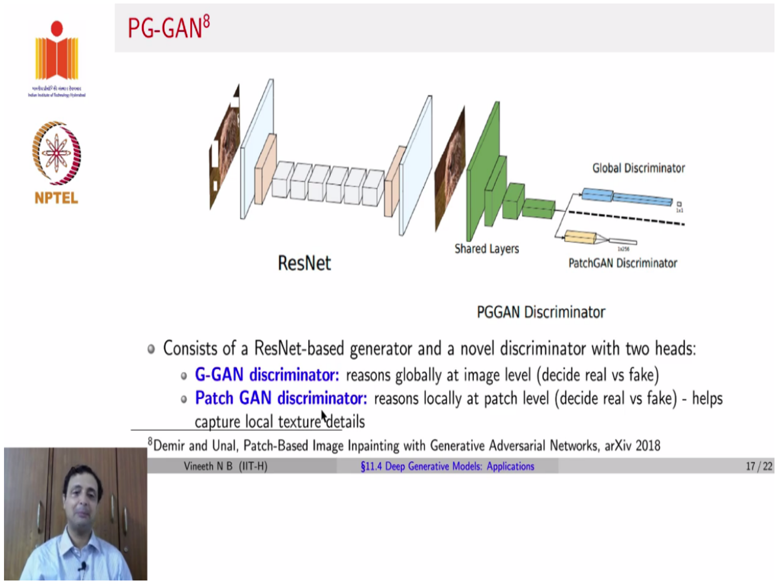
- **Image inpainting** a reconstruction technique used for filling missing parts in an image
- Can a GAN be trained to perform image inpainting?

[7]Demir and Unal, Patch-Based Image Inpainting with Generative Adversarial Networks, arXiv 2018

Vineeth N B (IIT-H) §11.4 Deep Generative Models: Applications 16 / 22

The last application that we will see is GANs for Image Inpainting. Image Inpainting is a task where a part of an image is missing and the job of a GAN is to fill in that missing part in the image. Can a GAN be trained to perform Image Inpainting? You know the answer?

(Refer Slide Time: 15:31)



PG-GAN[8]

- Consists of a ResNet-based generator and a novel discriminator with two heads:
  - **G-GAN discriminator:** reasons globally at image level (decide real vs fake)
  - **Patch GAN discriminator:** reasons locally at patch level (decide real vs fake) - helps capture local texture details

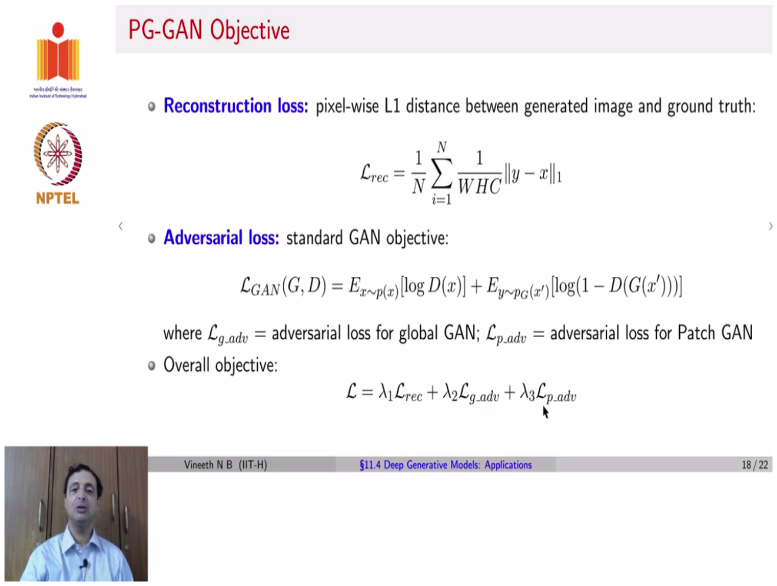
[8]Demir and Unal, Patch-Based Image Inpainting with Generative Adversarial Networks, arXiv 2018

Vineeth N B (IIT-H) §11.4 Deep Generative Models: Applications 17 / 22

The answer is, Yes. One of the methods proposed for this is known as PG-GAN, which uses a standard GAN framework where the generator is a residual network, which takes an image as input, the one which has parts missing, generates an image of the same resolution and the discriminator is divided into two parts, a global discriminator and a Patch GAN discriminator.

The global discriminator reasons at a global or an image level to decide whether the image is real or fake and the Patch GAN discriminator reasons at a local level, at a patch level to decide whether an image is real or fake. So, the Patch GAN discriminator helps capture local texture details and fill in those missing gaps in the image.

(Refer Slide Time: 16:32)



PG-GAN Objective

- **Reconstruction loss:** pixel-wise L1 distance between generated image and ground truth:

$$\mathcal{L}_{rec} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{WHC}\|y - x\|_1$$

- **Adversarial loss:** standard GAN objective:

$$\mathcal{L}_{GAN}(G, D) = E_{x\sim p(x)}[\log D(x)] + E_{y\sim p_G(x')}[\log(1 - D(G(x')))]$$

where $\mathcal{L}_{g\_adv}$ = adversarial loss for global GAN; $\mathcal{L}_{p\_adv}$ = adversarial loss for Patch GAN

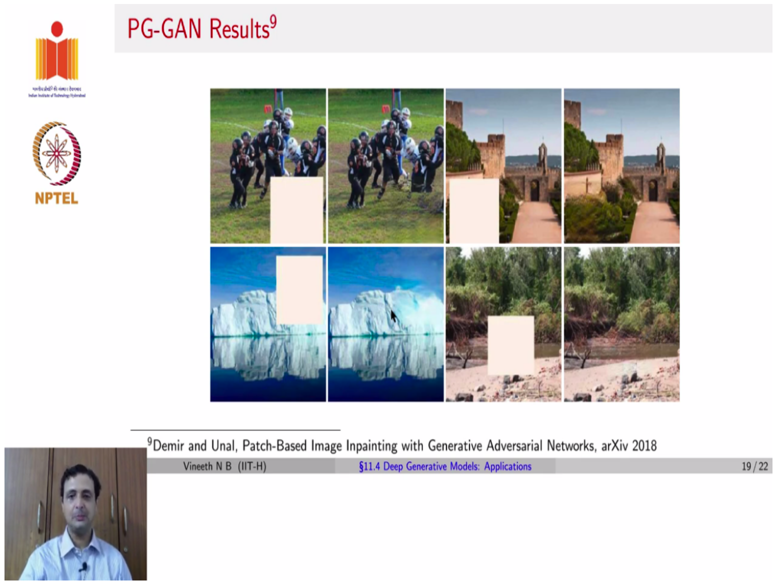
- Overall objective:

$$\mathcal{L} = \lambda_1\mathcal{L}_{rec} + \lambda_2\mathcal{L}_{g\_adv} + \lambda_3\mathcal{L}_{p\_adv}$$

Vineeth N B (IIT-H) §11.4 Deep Generative Models: Applications 18 / 22

What is the loss? So there is a reconstruction loss, which measures a pixel wise L1 distance between the generated image and ground truth and an Adversarial Loss, where the adversarial loss is divided into two parts, one for the global GAN and one for the Patch GAN and the overall objective is a combination of the reconstruction loss, the adversarial loss for the global GAN and the adversarial loss for the Patch GAN.

(Refer Slide Time: 17:07)



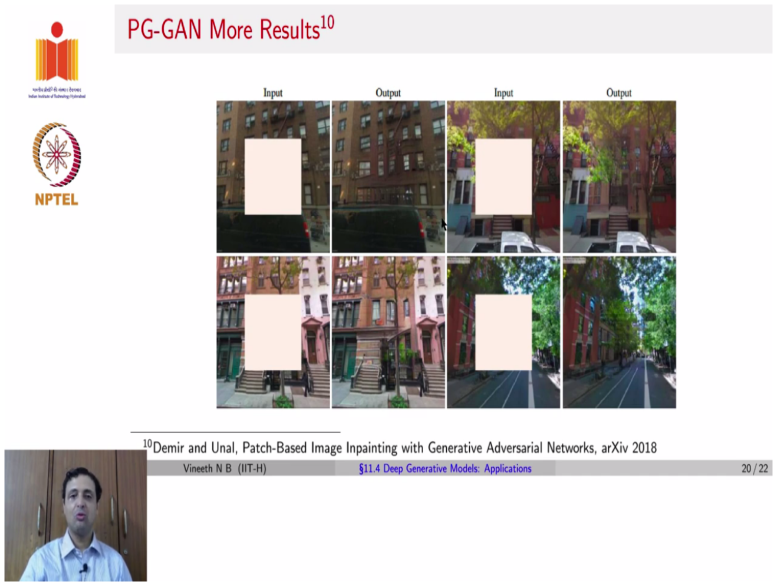
PG-GAN Results[9]

[9]Demir and Unal, Patch-Based Image Inpainting with Generative Adversarial Networks, arXiv 2018

Vineeth N B (IIT-H) §11.4 Deep Generative Models: Applications 19 / 22

With this kind of a training procedure and architecture, here are some results of Image Inpainting, where when a patch is taken out from the image, the GAN gets a fairly true reconstruction of those parts. You can see many examples here, where you can see some artifacts where the GAN generates content, but it is still fairly realistic in its generation.

(Refer Slide Time: 17:37)



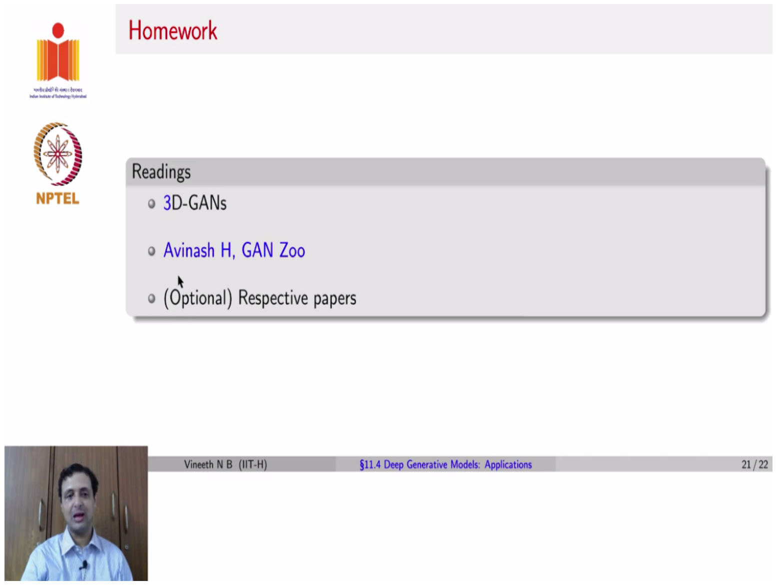
PG-GAN More Results[10]

[10]Demir and Unal, Patch-Based Image Inpainting with Generative Adversarial Networks, arXiv 2018

Vineeth N B (IIT-H) §11.4 Deep Generative Models: Applications 20 / 22

Here are more results of Building facades, where once again, the GAN fills in information fairly in a photorealistic manner.

(Refer Slide Time: 17:49)



Homework

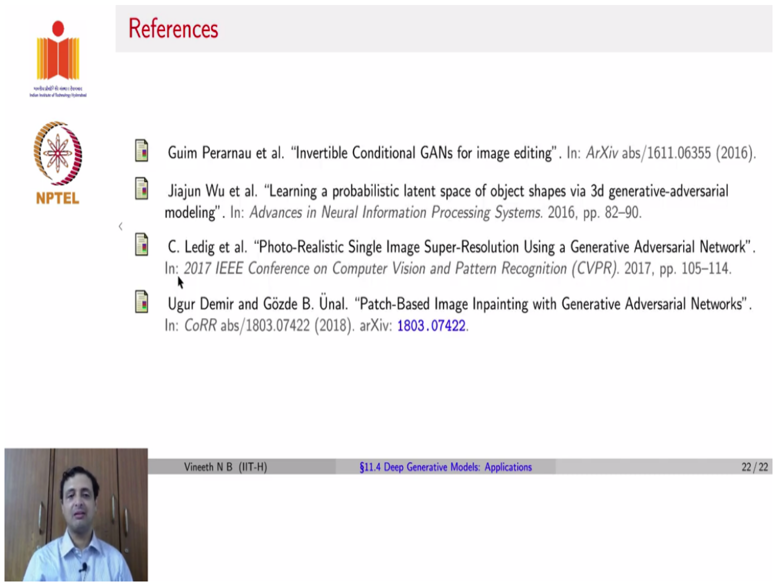
Readings

- 3D-GANs
- Avinash H, GAN Zoo
- (Optional) Respective papers

Vineeth N B (IIT-H) §11.4 Deep Generative Models: Applications 21 / 22

For more information on these applications, please see this link for 3D GANs and a very nice link called GAN Zoo, which presents an overview of various kinds of GANs, which you can use for different applications.

(Refer Slide Time: 18:06)



References

Guim Perarnau et al. "Invertible Conditional GANs for image editing". In: *ArXiv* abs/1611.06355 (2016).

Jiajun Wu et al. "Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling". In: *Advances in Neural Information Processing Systems*. 2016, pp. 82–90.

C. Ledig et al. "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 105–114.

Ugur Demir and Gözde B. Ünal. "Patch-Based Image Inpainting with Generative Adversarial Networks". In: *CoRR* abs/1803.07422 (2018). arXiv: 1803.07422.

Vineeth N B (IIT-H) §11.4 Deep Generative Models: Applications 22 / 22

Here are references.