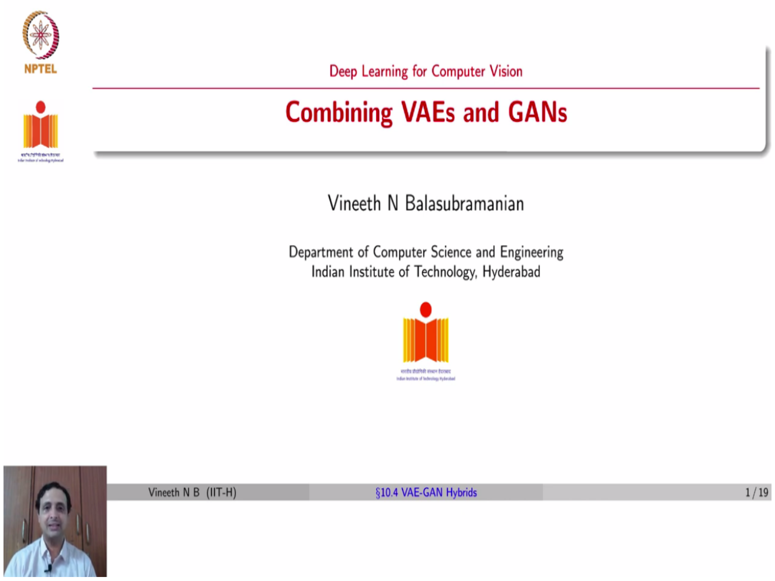
Deep Learning for Computer Vision
Professor Vineeth N Balasubramanian
Department of Computer Science and Engineering
Indian Institute of Technology, Hyderabad
Lecture 65
Combining VAEs and GANs

(Refer Slide Time: 00:15)



Deep Learning for Computer Vision

Combining VAEs and GANs

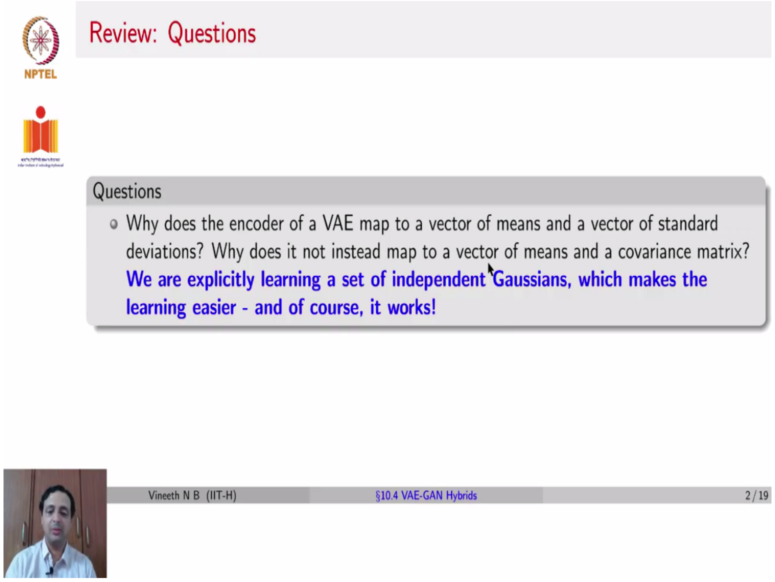
Vineeth N Balasubramanian

Department of Computer Science and Engineering
Indian Institute of Technology, Hyderabad

Vineeth N B (IIT-H) §10.4 VAE-GAN Hybrids 1 / 19

Having discussed GANs and VAEs so far, let us not talk about methods that have attempted to combine GANs and VAEs in a single framework.

(Refer Slide Time: 00:31)



Review: Questions

Questions

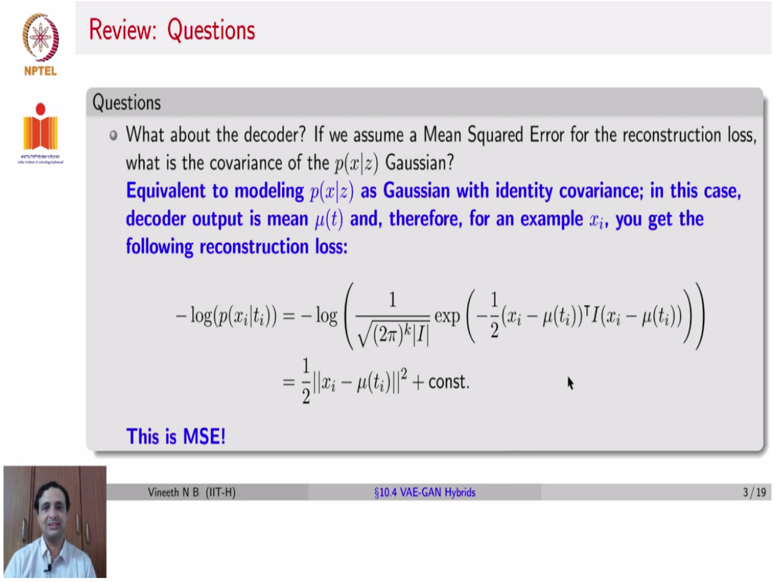
- Why does the encoder of a VAE map to a vector of means and a vector of standard deviations? Why does it not instead map to a vector of means and a covariance matrix? **We are explicitly learning a set of independent Gaussians, which makes the learning easier - and of course, it works!**

Vineeth N B (IIT-H) §10.4 VAE-GAN Hybrids 2 / 19

Before we go there, let us try to recall the questions we left behind in the last lecture. One of the questions was, why does the encoder of a VAE map to a vector of means and standard deviations? Why does it not map to a vector of means and an entire covariance matrix? I hope you had a chance to think about it. In this case, by design, we are explicitly learning a set of independent Gaussians.

That is the reason you only need standard deviations per each dimension of the Gaussian. We are not going to go with learning an entire covariance matrix. Technically speaking, it is possible also to learn a complete covariance matrix, but that does complicate how VAE learns. Importantly, this approach works, and it is easy to learn, relatively speaking, and compared to a complete covariance matrix. And that is the reason we went ahead with that choice.

(Refer Slide Time: 01:45)



Review: Questions

Questions

- What about the decoder? If we assume a Mean Squared Error for the reconstruction loss, what is the covariance of the $p(x|z)$ Gaussian?
  Equivalent to modeling $p(x|z)$ as Gaussian with identity covariance; in this case, decoder output is mean $\mu(t)$ and, therefore, for an example x_i, you get the following reconstruction loss:

$$-\log(p(x_i|t_i)) = -\log\left(\frac{1}{\sqrt{(2\pi)^k|I|}}\exp\left(-\frac{1}{2}(x_i-\mu(t_i))^\intercal I(x_i-\mu(t_i))\right)\right)$$

$$= \frac{1}{2}||x_i-\mu(t_i)||^2 + \text{const.}$$

This is MSE!

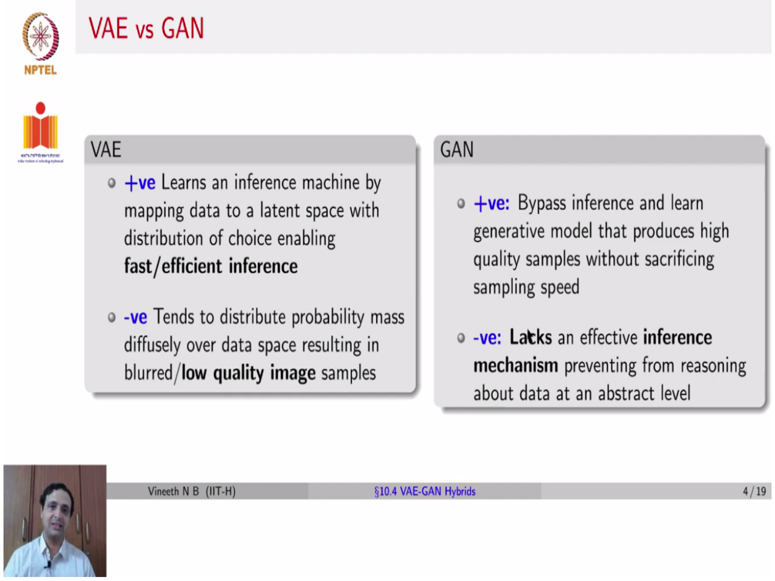
Vineeth N B (IIT-H) §10.4 VAE-GAN Hybrids 3 / 19

What about the decoder? If we assumed a Mean Squared Error for the reconstruction loss, if you recall, VAE had two terms in its objective function: reconstruction loss, which is about maximizing one of a conditional probability and then a KL divergence term. If we used a mean square error for the reconstruction loss, what would be the covariance of $p\ (x \mid z)$, assuming it is a Gaussian. If you thought carefully about this, this particular case of assuming a mean squared error would be equivalent to modelling $p\ (x \mid z)$ as a Gaussian with identity covariance.

In which case, you only need to learn the means; the standard deviations are given to be one. So, the decoder output would be the mean alone, and identical variance would be a given. So, in this case, the reconstruction loss would become $-\ \ log\ p(x_i \mid t_i\ )$, where t is, say, each dimension. If you expand the Gaussian formula in this particular case, you will notice that because you have an identity matrix as a covariance matrix.

Minimizing this negative log-likelihood term simplifies minimizing the mean square error. Inside this, the first term here would become a constant. It does not depend on the minimization term minus log and exponential are inverse operations. You will be left only with this mean square error term. So, using the mean square for the reconstruction loss in a VAE is equivalent to assuming that your distribution $p\ (x \mid z)$ is a multivariate Gaussian but with an identity covariance matrix.

(Refer Slide Time: 03:57)



VAE vs GAN

VAE

- +ve Learns an inference machine by mapping data to a latent space with distribution of choice enabling **fast/efficient inference**
- -ve Tends to distribute probability mass diffusely over data space resulting in blurred/**low quality image** samples

GAN

- +ve: Bypass inference and learn generative model that produces high quality samples without sacrificing sampling speed
- -ve: **Lacks** an effective **inference mechanism** preventing from reasoning about data at an abstract level

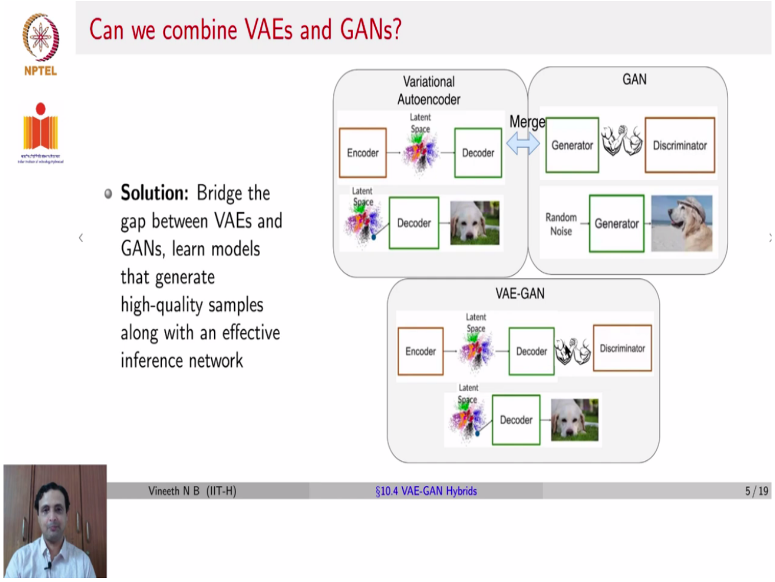
Vineeth N B (IIT-H) §10.4 VAE-GAN Hybrids 4 / 19

Let us try to look at the positives and negatives of VAEs and GANs before we discuss methods that combine them. In VAEs, the biggest positive is learning a very strong inference mechanism or machine by mapping data to a latent space with the distribution of choice with a fast, effective inference step. The negative, however, is because of the use of KL divergence, VAEs tend to distribute the probability mass diffusely over the data space may not cover the entire space, which is one reason for VAEs to result in blurry or low-quality image samples.

The other reason is that by sampling from a distribution, there is always an averaging effect that could also result in blurry generations rather than having sharp image generations from the latent space of a VAE. On the other hand, GANs do not have an inference step. You do not try to learn a latent from data. Recall that for GANs, you just give a Gaussian vector this input without worrying about whether that is the real latent manifold, which captures the data distribution.

You learn a generative model that produces high-quality samples at a good sampling speed. That is the objective of GAN. The negative is that GAN slacked that inference mechanism, which could prevent reasoning about data at an abstract level. For example, you cannot look at the latent variables and attach semantics to each latent variable. For example, you may not be able to say that the first latent variable corresponds to identity. The second latent variable corresponds to expression pose, and so on. It is difficult to do with a GAN. Whereas with a VAE, that procedure is implicit in its design.

(Refer Slide Time: 06:15)



Can we combine VAEs and GANs?

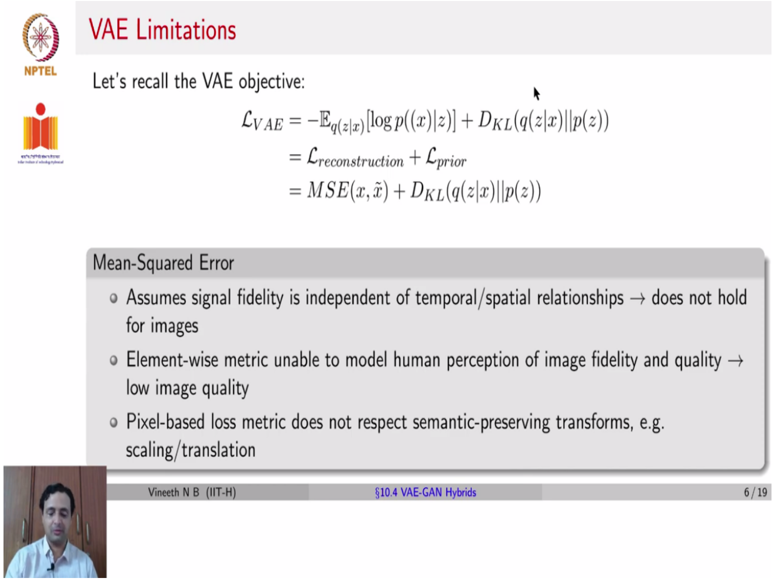
- **Solution:** Bridge the gap between VAEs and GANs, learn models that generate high-quality samples along with an effective inference network

Vineeth N B (IIT-H) §10.4 VAE-GAN Hybrids 5 / 19

Now, the question that we try to ask is, can we try to combine a VAE, and a GAN, to be able to get high-quality samples, as well as have an effective inference network to be able to reason at the level of latent variables. So you see here, that furbished variational autoencoder at training time, you have an encoder, you learn a latent space, which then feeds into a decoder, and a test time you sample from that latent space and pass to a decoder.

A GAN has a generator, which competes with the discriminator and a test time, you provide random noise to the generator, and the generator can generate images. So what we are going to see whether these two pipelines can be combined in some ways.

(Refer Slide Time: 07:14)



VAE Limitations

Let's recall the VAE objective:

$$\begin{aligned}\mathcal{L}_{VAE} &= -\mathbb{E}_{q(z|x)}[\log p((x)|z)] + D_{KL}(q(z|x)||p(z)) \\ &= \mathcal{L}_{reconstruction} + \mathcal{L}_{prior} \\ &= MSE(x, \tilde{x}) + D_{KL}(q(z|x)||p(z))\end{aligned}$$

Mean-Squared Error

- Assumes signal fidelity is independent of temporal/spatial relationships → does not hold for images
- Element-wise metric unable to model human perception of image fidelity and quality → low image quality
- Pixel-based loss metric does not respect semantic-preserving transforms, e.g. scaling/translation

Vineeth N B (IIT-H) §10.4 VAE-GAN Hybrids 6 / 19

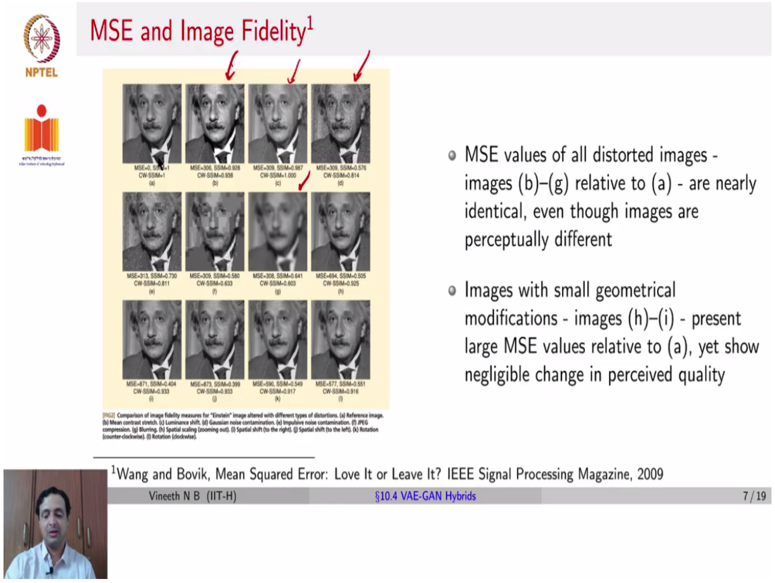
To do that, let us first discuss a few limitations of VAEs in more detail. Recall the VAE objective, which is given by the conditional distribution, the log-likelihood, and a KL divergence term that matches the approximate posterior with the prior. If you look at these terms, this can be the first term that a reconstruction loss can replace. And the second, you can view it as a certain kind of a prior loss.

The first term is implemented, perhaps through a mean squared error, the second term is implemented using the same KL divergence. Suppose you look at Mean Squared Error as a reconstruction loss. Mean squared error is inherently limited by its capabilities. Why is this so? Mean square error is an L2 distance between two images pixel-wise.

The moment you do that, you are assuming that the reconstruction fidelity or the signal fidelity in the in our case, the signal is the generated 2d image is independent of spatial or any temporal relationships across the pixels, which is not true for images, images do have a lot of local spatial correlation. The element-wise metric of finding a mean square error between every pixel in the same location does not model human perception of image fidelity and quality.

We will see an example soon. This could also lower the image quality in VAEs. Finally, the same pixel-based last metric mean squared error does not respect semantic preserving transforms. So you could have two images, which have the same object, but in one image, it is rotated from the other. Semantically, this is still the same image, but pixel-wise, this could result in a huge mean square error.

(Refer Slide Time: 09:35)



MSE and Image Fidelity[1]

[FIG2] Comparison of image fidelity measures for "Einstein" image altered with different types of distortions. (a) Reference image. (b) Mean contrast stretch. (c) Luminance shift. (d) Gaussian noise contamination. (e) Impulsive noise contamination. (f) JPEG compression. (g) Blurring. (h) Spatial scaling (zooming out). (i) Spatial shift (to the right). (j) Spatial shift (to the left). (k) Rotation (counter-clockwise). (l) Rotation (clockwise).

- MSE values of all distorted images - images (b)–(g) relative to (a) - are nearly identical, even though images are perceptually different
- Images with small geometrical modifications - images (h)–(i) - present large MSE values relative to (a), yet show negligible change in perceived quality

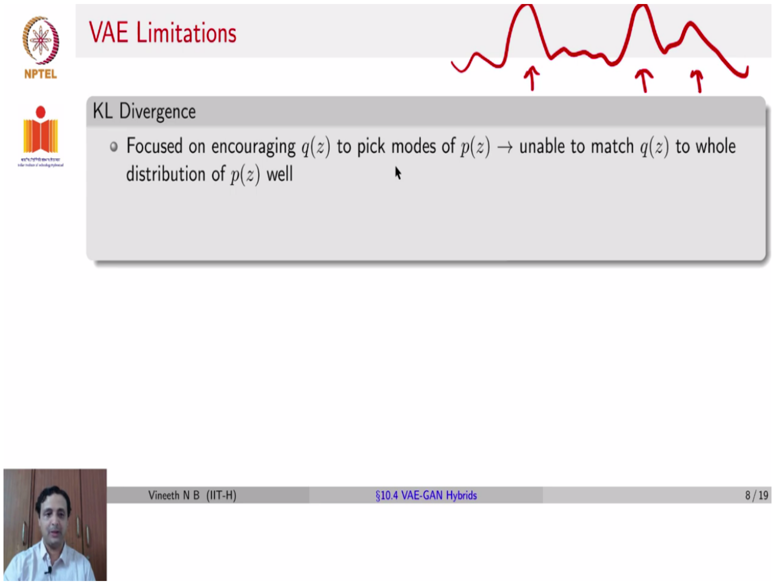
[1] Wang and Bovik, Mean Squared Error: Love It or Leave It? IEEE Signal Processing Magazine, 2009

Vineeth N B (IIT-H) §10.4 VAE-GAN Hybrids 7 / 19

A very nice study in this work known as "*Mean Squared Error: Love it, or Leave it ?*" shows a few tangible examples. Here you see an image of Albert Einstein. So you have an image a here, and you can see images b through i. If you took these images and observed the ones from b to g, these images are in the first and second row. If you compare them to a, you can see that they have significant differences.

You can see some images to be very sharp, some images to have a blur, a certain noise, a significant blur, so on and so forth. But it happens that each of those images from b to g has almost the same mean square error from the first image a. On the other hand, if you consider those images from h to i, they all look the same to the human eye. But they all have very high large mean square error values to the original image. That talks about mean square error as a metric for capturing the goodness of reconstruction.

(Refer Slide Time: 10:59)



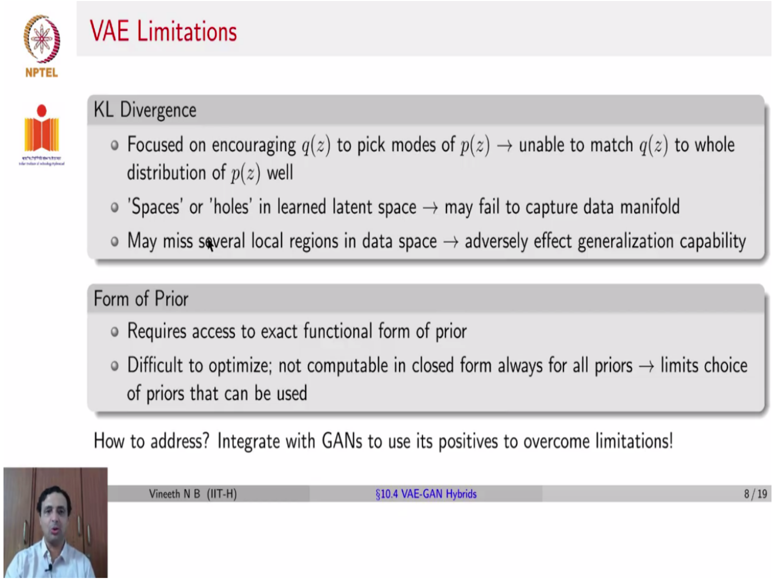
VAE Limitations

KL Divergence

- Focused on encouraging $q(z)$ to pick modes of $p(z) \rightarrow$ unable to match $q(z)$ to whole distribution of $p(z)$ well

In addition to this, VAEs have a couple of other problems also, by using the KL divergence to match the approximate posterior to the prior on the latent variables, z. Inherently KL divergence focuses on encouraging $q(z)$, the approximate posterior in our case, on picking the $p(z)$ $modes$. So if you had $p(z)$ to be a distribution, something like this, what it tries to do is try to ensure that it matches p in these points where there is a high density because that is what would give it a low KL divergence score between q and p. And by doing that, q may not completely match the entire distribution of p.

(Refer Slide Time: 11:52)



VAE Limitations

KL Divergence

- Focused on encouraging $q(z)$ to pick modes of $p(z)$ → unable to match $q(z)$ to whole distribution of $p(z)$ well
- 'Spaces' or 'holes' in learned latent space → may fail to capture data manifold
- May miss several local regions in data space → adversely effect generalization capability

Form of Prior

- Requires access to exact functional form of prior
- Difficult to optimize; not computable in closed form always for all priors → limits choice of priors that can be used

How to address? Integrate with GANs to use its positives to overcome limitations!

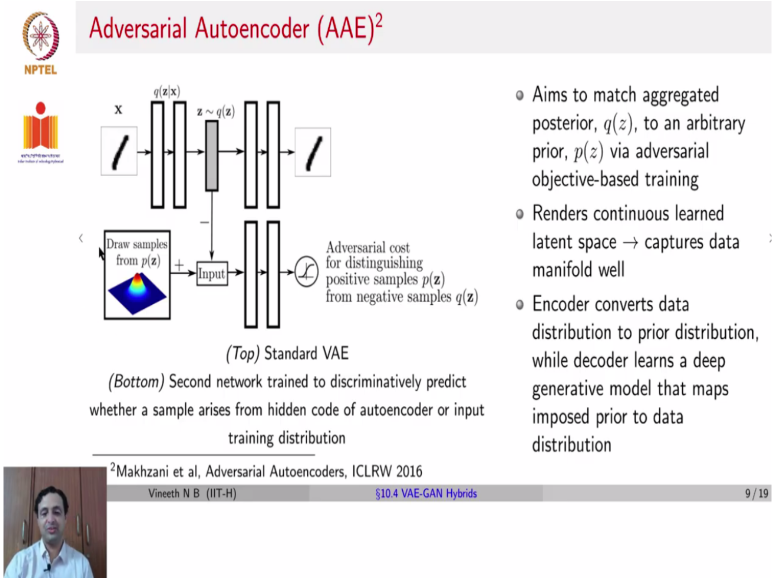
Vineeth N B (IIT-H) §10.4 VAE-GAN Hybrids 8 / 19

That could leave spaces or holes in the learned latent space of z, which may result in failing to capture the data manifold. It could also miss several local regions in the data space, affecting generalization capability of generating examples out of a VAE. Lastly, even the prior considered in VAEs could become a limitation. Remember that VAE is required you to assume a certain functional form of a prior such as a unit Gaussian.

And sometimes, for different kinds of priors, VAEs may be difficult to optimize. You may not get a closed-form solution. In our case, because we assumed the approximate posterior and the prior to be Gaussian, the KL divergence term became a closed-form, there was a closed-form expression for a KL divergence between two Gaussian distributions, which turned out to be differentiable, which allowed us to use it for training the VAE.

That may not be true for other kinds of priors. And this limits us to choices of priors that can be used in a VAE. How do you address these limitations of VAEs? That is what we will talk about by integrating elements of GANs in a VAE to help improve its performance. We will talk about a couple of seminal methods in this context in this lecture.

(Refer Slide Time: 13:34)



x
$q(\mathbf{z}|\mathbf{x})$
$\mathbf{z} \sim q(\mathbf{z})$
Draw samples from $p(\mathbf{z})$
+
−
Input
Adversarial cost for distinguishing positive samples $p(\mathbf{z})$ from negative samples $q(\mathbf{z})$

Adversarial Autoencoder (AAE)[2]

- Aims to match aggregated posterior, $q(z)$, to an arbitrary prior, $p(z)$ via adversarial objective-based training
- Renders continuous learned latent space → captures data manifold well
- Encoder converts data distribution to prior distribution, while decoder learns a deep generative model that maps imposed prior to data distribution

(Top) Standard VAE

(Bottom) Second network trained to discriminatively predict whether a sample arises from hidden code of autoencoder or input training distribution

[2] Makhzani et al, Adversarial Autoencoders, ICLRW 2016

Vineeth N B (IIT-H) §10.4 VAE-GAN Hybrids 9 / 19

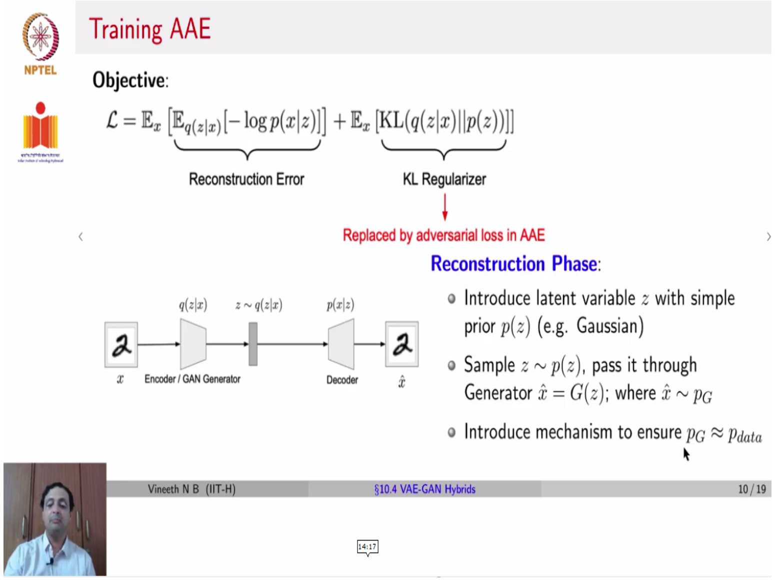
And one of the first efforts here is known as an Adversarial Autoencoder. This illustration on the left gives the adversarial autoencoder. So on the top row, here is a standard variational autoencoder. As you can see, x going to z, a latent variable is a sample from that latent space. You have a decoder that gives you a reconstruction. The bottom part of an adversarial autoencoder has a discriminator. The discriminator's job is to not look at images and say whether they are real or fake.

But to look at the latent space and see whether the latent space came from the real distribution, the latents corresponding to the real distribution, which you may have got from a GAN, for instance. The latent code that comes out while learning the VAE. Why do we do this? This has a very important meaning here. The goal here is to make the VAEs prior match the original prior of the data distribution.

This is important now because no more are you asking the approximate posterior to match a unit Gaussian, but you are asking it to match a data prior that is known, which would allow the VAE to be more powerful in its functioning. So this allows you to render a more continuous learned latent space, which allows you to capture the data manifold well. Through this process, we are converting the data distribution to prior distribution.

And the decoder learns a deep generator model that maps that imposed prior to data distribution. So instead of using a KL divergence between the approximate posterior and the unit Gaussian, we now use an adversarial objective to match that approximate posterior to the prior of the data generating distribution.

(Refer Slide Time: 15:57)



Training AAE

Objective:

$$\mathcal{L} = \mathbb{E}_x \left[\underbrace{\mathbb{E}_{q(z|x)}[-\log p(x|z)]}_{\text{Reconstruction Error}} \right] + \mathbb{E}_x \left[\underbrace{\mathrm{KL}(q(z|x)||p(z))}_{\text{KL Regularizer}} \right]$$

Replaced by adversarial loss in AAE

Reconstruction Phase:

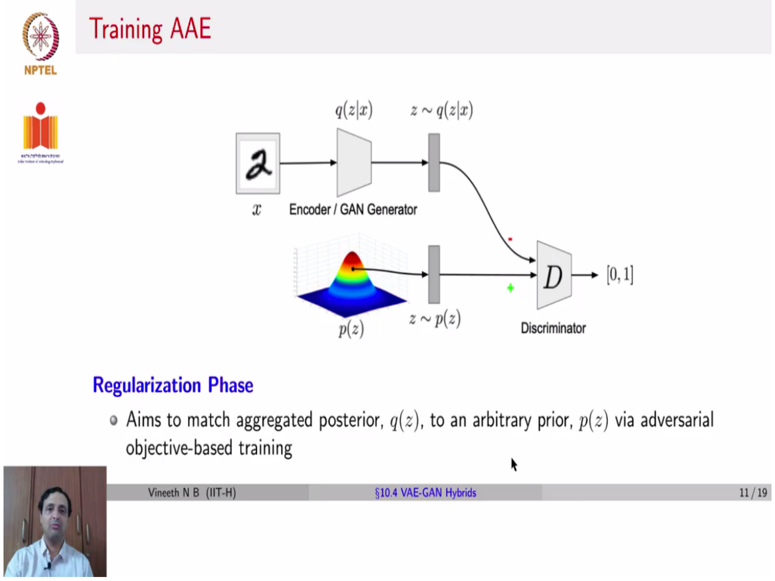
- Introduce latent variable z with simple prior $p(z)$ (e.g. Gaussian)
- Sample $z \sim p(z)$, pass it through Generator $\hat{x} = G(z)$; where $\hat{x} \sim p_G$
- Introduce mechanism to ensure $p_G \approx p_{data}$

Vineeth N B (IIT-H) §10.4 VAE-GAN Hybrids 10 / 19

14:17

So if you look at the original objective of VAE, we had two terms, the reconstruction term and the KL Regularizer. Now in an Adversarial Autoencoder, the KL Regularizer is replaced by an adversarial loss of a discriminator. That is trying to classify the latent code as belonging to the VAE or belonging to the original data distribution. So in the reconstruction phase, you introduce a latent variable with a simple prior, you sample z and pass it through a generator. Remember that we need to introduce a mechanism to ensure p_G is p_{data}.

(Refer Slide Time: 16:45)



Training AAE

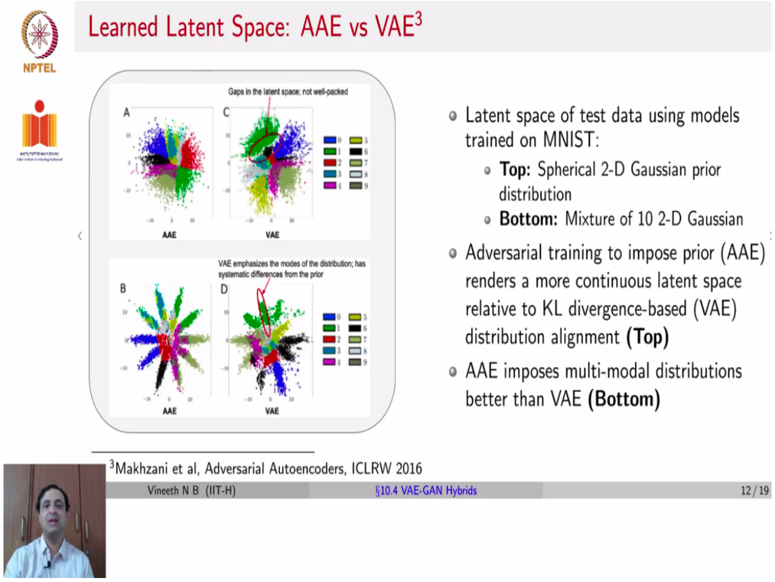
Regularization Phase

- Aims to match aggregated posterior, $q(z)$, to an arbitrary prior, $p(z)$ via adversarial objective-based training

Vineeth N B (IIT-H) §10.4 VAE-GAN Hybrids 11 / 19

And in the adversarial autoencoder, this is done by matching the aggregated posterior, the one from the variational autoencoder, to an arbitrary prior using adversarial objective-based training that comes from your GANs discriminator objective.

(Refer Slide Time: 17:07)



Learned Latent Space: AAE vs VAE[3]

Gaps in the latent space; not well-packed
A
C
10
0
-10
-10
0
10
AAE
VAE
0
1
2
3
4
5
6
7
8
9
VAE emphasizes the modes of the distribution; has systematic differences from the prior
B
D

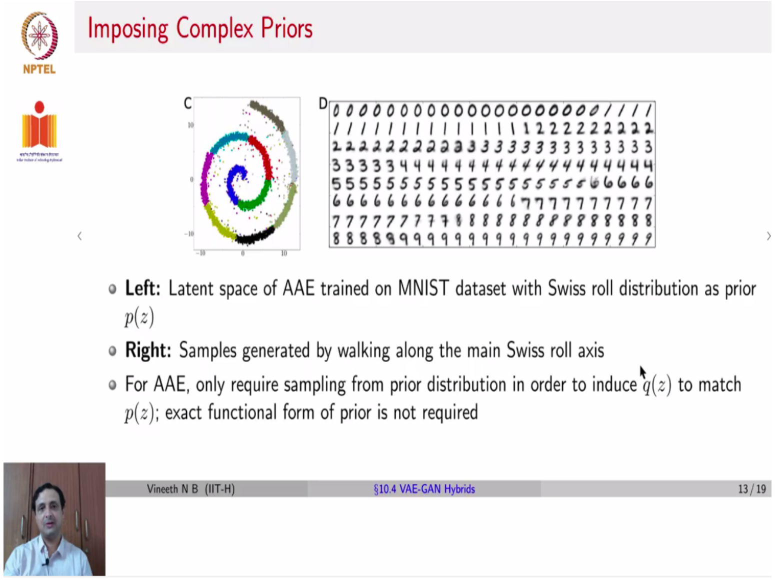
- Latent space of test data using models trained on MNIST:
  - **Top:** Spherical 2-D Gaussian prior distribution
  - **Bottom:** Mixture of 10 2-D Gaussian
- Adversarial training to impose prior (AAE) renders a more continuous latent space relative to KL divergence-based (VAE) distribution alignment **(Top)**
- AAE imposes multi-modal distributions better than VAE **(Bottom)**

[3] Makhzani et al, Adversarial Autoencoders, ICLRW 2016

Vineeth N B (IIT-H) §10.4 VAE-GAN Hybrids 12 / 19

And by doing so, adversarial autoencoders give a very strong performance. Here is an example using a model for an adversarial autoencoder and comparing it against the variational autoencoder on the MNIST dataset. And what you see on top here is where a prior based on a spherical 2D Gaussian is used. And the bottom is where the prior is a mixture of 10 2D Gaussians. And you can see here from the top that the adversarial autoencoder learns a more continuous latent space, whereas the VAE has many discontinuities in that latent space. And suppose you look at the bottom image. In that case, the adversarial autoencoder learns a fairly smooth, multimodal distribution, all those modes along with those different directions, whereas the VAE still struggles even in that setting.

(Refer Slide Time: 18:11)



Imposing Complex Priors

- **Left:** Latent space of AAE trained on MNIST dataset with Swiss roll distribution as prior $p(z)$
- **Right:** Samples generated by walking along the main Swiss roll axis
- For AAE, only require sampling from prior distribution in order to induce $q(z)$ to match $p(z)$; exact functional form of prior is not required

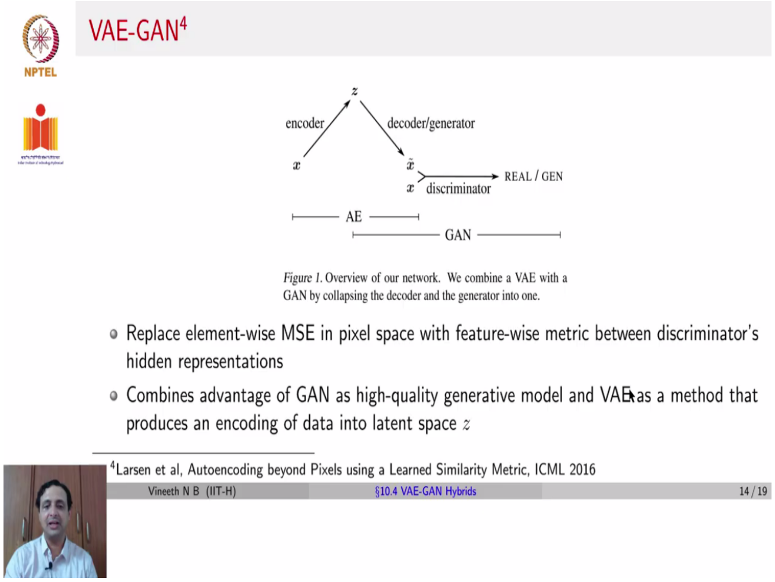
Vineeth N B (IIT-H) §10.4 VAE-GAN Hybrids 13 / 19

This particular work for adversarial autoencoders also showed that you could also use more complex priors if you choose to. Here is an example of where a latent space of an adversarial autoencoder was trained on MNIST, with the prior being a Swiss roll distribution, as you see here. So, you can now sample from this distribution by walking along the axis here in this particular case. The samples were generated by walking along the Swiss roll axis, passing it to the VAEs decoder and generating samples.

And you can see here that you have a fair good amount of variety in the generation of samples in the MNIST dataset by walking along with such a prior. So this entire idea of adversarial autoencoders replaces the KL divergence term in the objective of a variational autoencoder with an adversarial learning term. When we say adversarial learning, we mean the loss corresponding to the discriminator calling an item fake or real.

And the nice part of this approach is there is no functional form of a prior required; whatever prior is provided is what q tries to match.

(Refer Slide Time: 19:41)



VAE-GAN[4]

Figure 1. Overview of our network. We combine a VAE with a GAN by collapsing the decoder and the generator into one.

- Replace element-wise MSE in pixel space with feature-wise metric between discriminator's hidden representations
- Combines advantage of GAN as high-quality generative model and VAE as a method that produces an encoding of data into latent space z

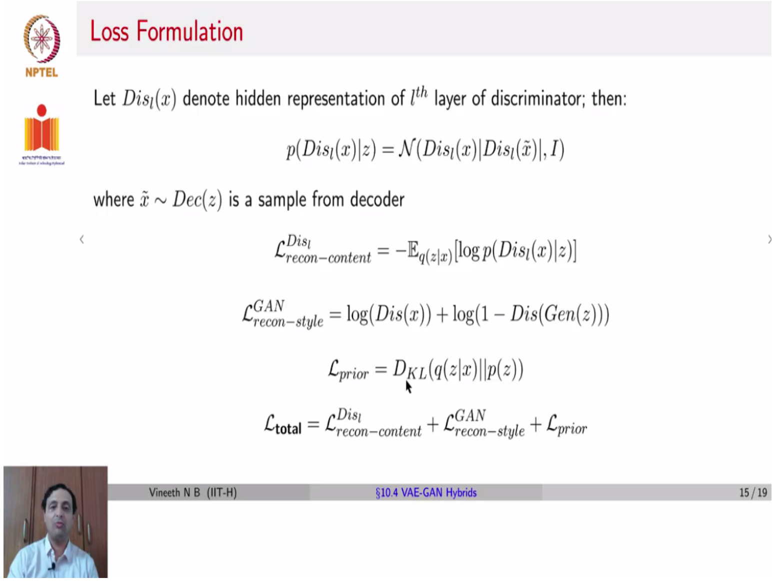
[4] Larsen et al, Autoencoding beyond Pixels using a Learned Similarity Metric, ICML 2016

Vineeth N B (IIT-H) §10.4 VAE-GAN Hybrids 14 / 19

A second popular method tries to look at the objective of a VAE from the other perspective. So while adversarial autoencoders replaced the KL divergence term with an adversarial objective. VAE-GANs try to replace the reconstruction loss with a different term, what do they replace it with? Instead of a pixel-wise mean square error, VAE-GANs replace it as a feature-wise distance in the discriminator’s representation space between outputs that come from a VAE and original data.

This approach combines the advantage of GAN as a high-quality generator model and VAE as a method that can produce an encoding of data into a latent space and then further reasoning at the latent space level.

(Refer Slide Time: 20:42)



Loss Formulation

Let $Dis_l(x)$ denote hidden representation of l^{th} layer of discriminator; then:

$$p(Dis_l(x)|z) = \mathcal{N}(Dis_l(x)|Dis_l(\tilde{x})|, I)$$

where $\tilde{x} \sim Dec(z)$ is a sample from decoder

$$\mathcal{L}^{Dis_l}_{recon-content} = -\mathbb{E}_{q(z|x)}[\log p(Dis_l(x)|z)]$$

$$\mathcal{L}^{GAN}_{recon-style} = \log(Dis(x)) + \log(1 - Dis(Gen(z)))$$

$$\mathcal{L}_{prior} = D_{KL}(q(z|x)||p(z))$$

$$\mathcal{L}_{\textbf{total}} = \mathcal{L}^{Dis_l}_{recon-content} + \mathcal{L}^{GAN}_{recon-style} + \mathcal{L}_{prior}$$

Vineeth N B (IIT-H) §10.4 VAE-GAN Hybrids 15 / 19

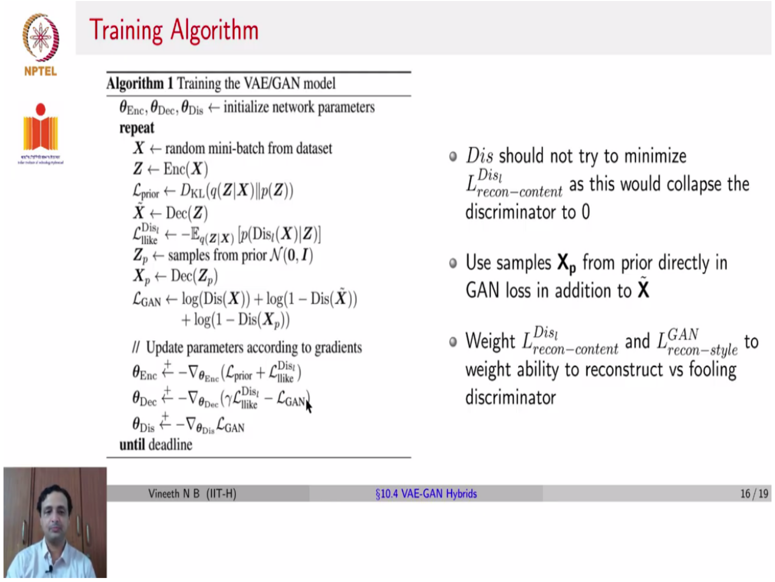
So the loss formulation in this particular case is based on representations of the discriminator. Remember, the discriminator is yet another neural network. So you can take a certain layer of the discriminator given by Dis_l, which denotes the l^{th} layer of the discriminator. Let us assume that the output of that discriminators l^{th} layer is given by $Dis_l(x)$, which are the feature representations that we are going to compare between a VAEs generated output and real data input.

So, $p\ (Dis_l(x)\ |\ z)$is assumed to be a Gaussian distribution, where $\tilde{x}$ is an output of the decoder obtained using a VAE. So, the first loss term is given by $L^{Dis_l}_{recon-content}$, which is the first term of your VAE objective given by the expectation of samples coming from the approximate posterior, $log\ p\ (Dis_l(x)\ |\ z)$, which is very close to the first term that we had in the VAE objective.

The second term comes from a GAN-based objective $L^{GAN}_{recon-style}$, $log\ Dis\ \ (x)$. So if the data comes from the real distribution, which is x, the GAN or the discriminator tries to maximize $log\ Dis\ \ (x)$ and $log\ (1\ -\ Dis\ \ (Gen(z)))$. So, z is latent that comes from the VAEs latent space. Finally, you have your prior loss, which tries to match the approximate posterior to the $p(z)$.

Recall the main difference now is that the first term is based on the features of the discriminator. So the total loss is given by the addition of these three losses.

(Refer Slide Time: 23:03)



Training Algorithm

Algorithm 1 Training the VAE/GAN model

```
$\theta_{\text{Enc}}, \theta_{\text{Dec}}, \theta_{\text{Dis}} \leftarrow$ initialize network parameters
repeat
    $X \leftarrow$ random mini-batch from dataset
    $Z \leftarrow \text{Enc}(X)$
    $\mathcal{L}_{\text{prior}} \leftarrow D_{\text{KL}}(q(Z|X) \| p(Z))$
    $\tilde{X} \leftarrow \text{Dec}(Z)$
    $\mathcal{L}_{\text{llike}}^{\text{Dis}_l} \leftarrow -\mathbb{E}_{q(Z|X)}[p(\text{Dis}_l(X)|Z)]$
    $Z_p \leftarrow$ samples from prior $\mathcal{N}(\mathbf{0}, I)$
    $X_p \leftarrow \text{Dec}(Z_p)$
    $\mathcal{L}_{\text{GAN}} \leftarrow \log(\text{Dis}(X)) + \log(1 - \text{Dis}(\tilde{X}))$
                $+ \log(1 - \text{Dis}(X_p))$
    // Update parameters according to gradients
    $\theta_{\text{Enc}} \overset{+}{\leftarrow} -\nabla_{\theta_{\text{Enc}}}(\mathcal{L}_{\text{prior}} + \mathcal{L}_{\text{llike}}^{\text{Dis}_l})$
    $\theta_{\text{Dec}} \overset{+}{\leftarrow} -\nabla_{\theta_{\text{Dec}}}(\gamma\mathcal{L}_{\text{llike}}^{\text{Dis}_l} - \mathcal{L}_{\text{GAN}})$
    $\theta_{\text{Dis}} \overset{+}{\leftarrow} -\nabla_{\theta_{\text{Dis}}}\mathcal{L}_{\text{GAN}}$
until deadline
```

- Dis should not try to minimize $L_{recon-content}^{Dis_l}$ as this would collapse the discriminator to 0
- Use samples $\mathbf{X_p}$ from prior directly in GAN loss in addition to $\tilde{\mathbf{X}}$
- Weight $L_{recon-content}^{Dis_l}$ and $L_{recon-style}^{GAN}$ to weight ability to reconstruct vs fooling discriminator

Vineeth N B (IIT-H) §10.4 VAE-GAN Hybrids 16 / 19

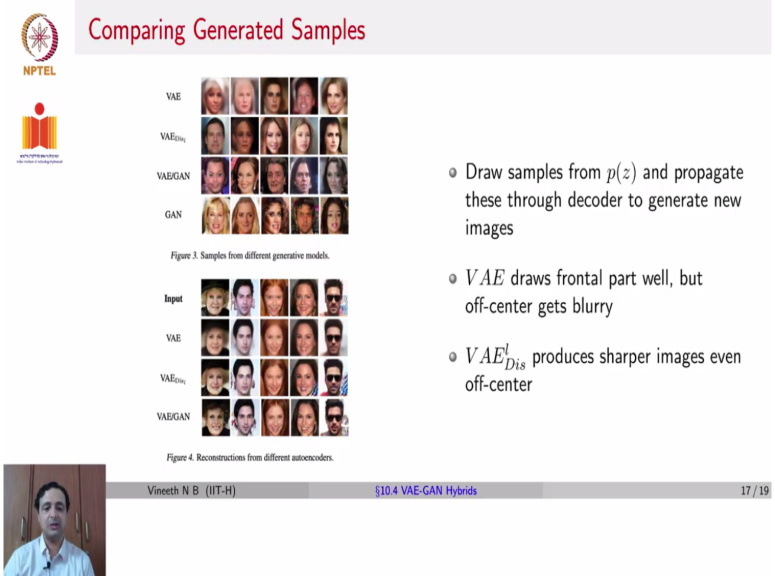
So the overall training algorithm for the VAE-GAN is given. You sample a mini-batch of samples from your training dataset. You get an encoding of X using the encoder in a variational autoencoder. You have your prior loss, which tries to match the approximate posterior obtained through your encoder part of your VAE with $p(Z)$. $\hat{X}$ is obtained through the decoder of the VAE when Z is given as input, Z is a latent variable.

Coming to the discriminator, we already saw that one of the last terms is a minus expectation, $p(Dis(X) \mid Z)$. There is one other component that VAE-GAN adds while training. It improves the performance and lets the user sample from a unit normal prior which is given as Z_p. It is passed through a decoder to get the reconstruction X_p. In addition to minimizing the likelihood of $\hat{X}$ fooling the discriminator, the GAN loss also tries to minimize the likelihood of X_p fooling the discriminator.

So, that is the loss of the GAN. Finally, the encoder, decoder and discriminator are updated using the corresponding gradients that affect each output. So obviously, each of those networks only uses the losses that are relevant for that particular network. So if you observe, you would notice here that the discriminator loss should not try to minimize the reconstruction content loss, which is the first term here, as that would collapse the discriminator to give a 0 at all times.

This is very similar to GANs, where we talked about training the discriminator fully initially. As we just mentioned, the VAE-GAN model also allows using samples X_p, which are obtained from a unit normal prior. In addition to $\widetilde{X}$, which are generated as output of the VAE. Finally, each of these recon-content and recon-style losses, which are the first two losses, are weighted to control a tradeoff between reconstruction quality and fooling the discriminator.

(Refer Slide Time: 26:00)



Comparing Generated Samples

Figure 3. Samples from different generative models.

Figure 4. Reconstructions from different autoencoders.

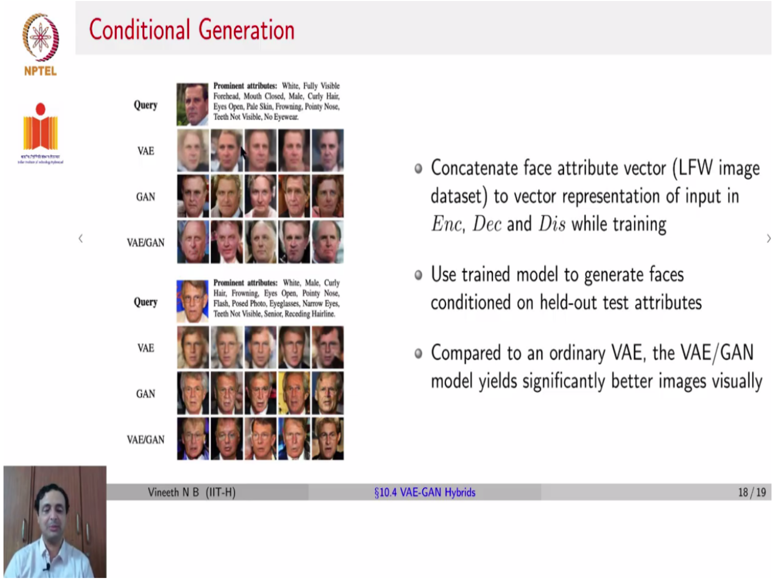
- Draw samples from $p(z)$ and propagate these through decoder to generate new images
- VAE draws frontal part well, but off-center gets blurry
- VAE^l_{Dis} produces sharper images even off-center

Vineeth N B (IIT-H) §10.4 VAE-GAN Hybrids 17 / 19

Here are some examples of results. You can see here that when you train a VAE on face images, you see that there is no complete clarity while you get an overall sense of a face image. In contrast, the centre of the image has a certain acceptable degree of the face. You can see this as you go away to the periphery. The clarity keeps dropping down as you keep going away from the centre. Again, you see that the VAE gives a fairly good performance of obtaining the sharpness and clarity of the face images.

While GANs also do a good job, GANs suffer from some artifacts that miss global information. These are excellent at retaining global information but miss finding sharpness. GANs, on the other hand, do have local sharpness but at times miss global content and can sometimes place different parts in different locations. And VAE-GANs bring the best of both worlds together to generate globally relevant content and keep each of the pixels and each of the local areas sharp in terms of perception.

(Refer Slide Time: 27:32)



Conditional Generation

Prominent attributes: White, Fully Visible Forehead, Mouth Closed, Male, Curly Hair, Eyes Open, Pale Skin, Frowning, Pointy Nose, Teeth Not Visible, No Eyewear.

Prominent attributes: White, Male, Curly Hair, Frowning, Eyes Open, Pointy Nose, Flash, Posed Photo, Eyeglasses, Narrow Eyes, Teeth Not Visible, Senior, Receding Hairline.

- Concatenate face attribute vector (LFW image dataset) to vector representation of input in Enc, Dec and Dis while training
- Use trained model to generate faces conditioned on held-out test attributes
- Compared to an ordinary VAE, the VAE/GAN model yields significantly better images visually

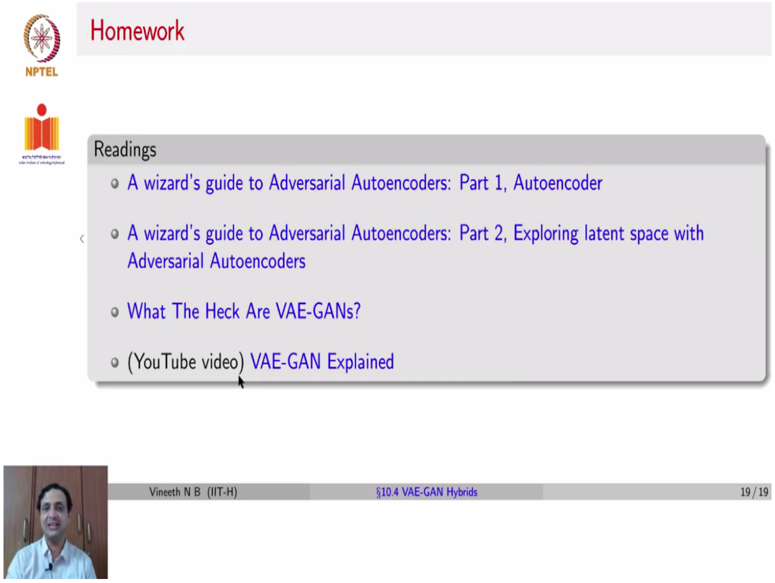
Vineeth N B (IIT-H) §10.4 VAE-GAN Hybrids 18 / 19

Another thing that you can do with these kinds of models, which was shown with the VAE again, is conditional generation. In this particular example, in VAE-GANs, the authors concatenated the face attribute vector. So you could, for example, have these attributes such as the white, fully visible forehead, mouth closed, male, curly hair, eyes open, pale skin, frowning, pointy nose, teeth not visible, and no eyewear, for instance.

So this can be represented as an attribute vector. So you can put 0s and 1s for different attributes, for instance, and that is appended to the vector representation of the input in the encoder-decoder and discriminator modules while training and this trained model is used to generate faces, which are conditioned on some held-out test attributes. So, a test time, a new face attribute vector, which is held-out which was not used before, is concatenated to the input representation, and now, the model can generate faces that satisfy these requirements in the attributes.

So all these images here are generations of a face conditioned on these attributes. And you see that for most of them, certain attributes such as the fully visible forehead, white, male, so on and so forth, are met to a reasonable extent, justifying this kind of an approach. Compared to VAE, the VAE GAN gives significantly good results for such conditional generation experiments.

(Refer Slide Time: 29:33)



Homework

Readings

- A wizard's guide to Adversarial Autoencoders: Part 1, Autoencoder
- A wizard's guide to Adversarial Autoencoders: Part 2, Exploring latent space with Adversarial Autoencoders
- What The Heck Are VAE-GANs?
- (YouTube video) VAE-GAN Explained

Your homework for this lecture will be to go through this excellent link on *"A wizard's guide to adversarial autoencoders, Part 1 and Part 2*", as well as an excellent tutorial on VAE-GANs and a nice video on YouTube on VAE-GANs, if you are interested, go through them.