

Unsupervised Learning: Introduction to Generative Models

Olivier Dubrule and Navjot Kukreja

1

Objectives of the Day

- Recall what unsupervised learning is, with PCA as simplest linear case
- Autoencoders and latent vectors for image parametrization
- Introduce transposed convolution
- Variational Autoencoders for coding and generating random images
- Generative Adversarial Networks for generating random images

2

Imperial College
London

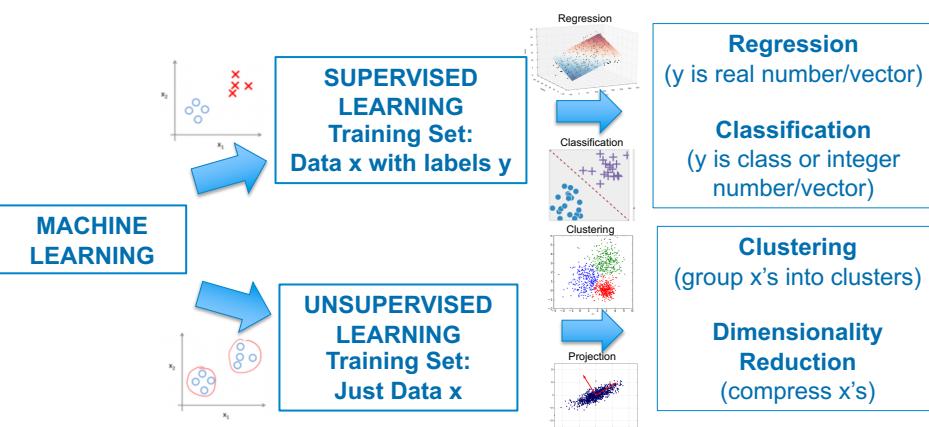
Introduction to Generative Models

1. PCA and Autoencoders
2. Transposed Convolution
3. Variational Autoencoders
4. Generative Adversarial Networks

3

Imperial College
London

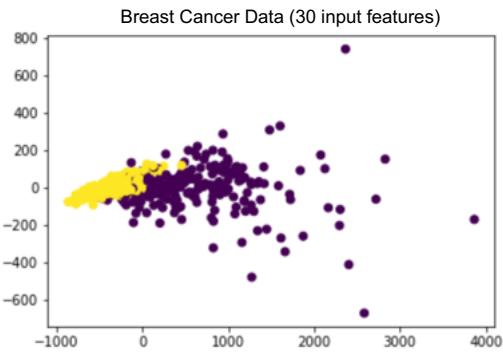
Supervised vs Unsupervised Learning



4

Imperial College
London

PCA Dimensionality Reduction



5

Imperial College
London

The PCA Method is Linear Dimensionality Reduction

If the first p eigenvectors are denoted as $(\varphi_k)_{k=1,\dots,p}$, each data vector $x^{(i)}$ of dimension n can be approximated by a linear combination of the eigenvectors:

$$x^{(i)} \approx \sum_{k=1}^p z_k^{(i)} \varphi_k$$

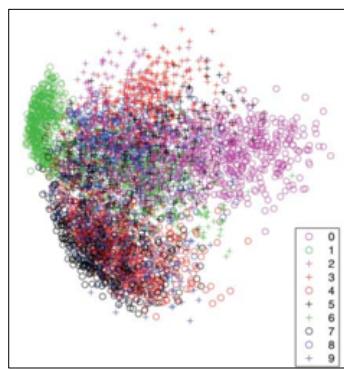
So, instead of being defined by its n original features, each $x^{(i)}$ is now defined by its p coordinates $z_k^{(i)}$ in the basis $(\varphi_k)_{k=1,\dots,p}$, where $p \ll n$: this is Dimensionality Reduction.

6

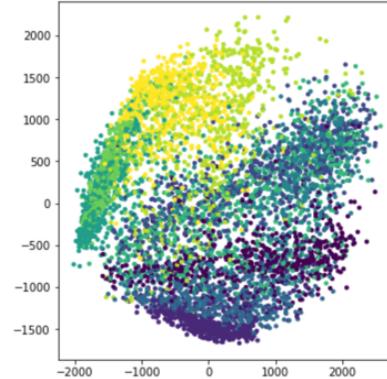
Imperial College
London

PCA Dimensionality Reduction on Images

MNIST (784 input features)



Fashion MNIST (784 input features)



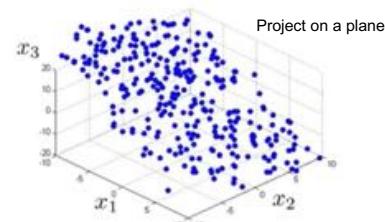
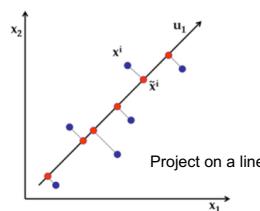
7

Imperial College
London

Dimensionality Reduction with Latent Vectors

Idea:

PCA projects only on a linear manifold, we wish to generalize to non-linear manifolds.



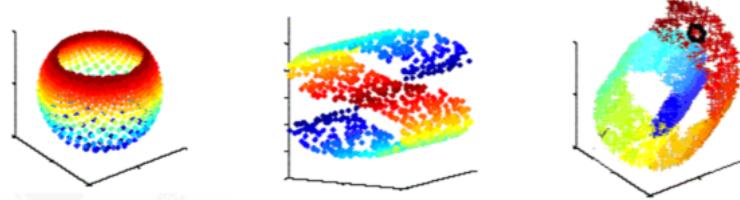
Potential applications

Simple parametrization of images, classification or regression based on the feature vectors $z^{(i)}$ rather than the raw data $x^{(i)}$, visualization, data compression, inversion....

8

Imperial College
London

Why do we Need a Non-Linear Approach ?

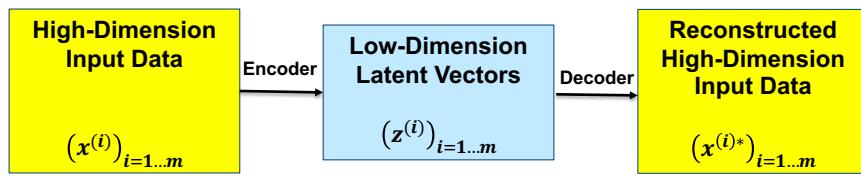


Examples of data that cannot be separated by projecting on a linear manifold.

9

Imperial College
London

Autoencoders: the Basic Idea



Minimize the reconstruction loss
 $\sum_{i=1}^m \|x^{(i)} - x^{(i)*}\|^2$ if real numbers
(or cross-entropy if classes)

Note that we use Supervised Learning to address an Unsupervised Learning problem!

10

Imperial College
London

Autoencoders Historical Example on MNIST (~12,000 citations)

Reducing the Dimensionality of Data with Neural Networks

G. E. Hinton* and R. R. Salakhutdinov

High-dimensional data can be converted to low-dimensional codes by training a multilayer neural network with a small central layer to reconstruct high-dimensional input vectors. Gradient descent can be used for fine-tuning the weights in such “autoencoder” networks, but this works well only if the initial weights are close to a good solution. We describe an effective way of initializing the weights that allows deep autoencoder networks to learn low-dimensional codes that work much better than principal components analysis as a tool to reduce the dimensionality of data.

Dimensionality reduction facilitates the classification, visualization, communication, and storage of high-dimensional data. A simple and widely used method is principal components analysis (PCA), which

finds the directions of greatest variance in the data set and represents each data point by its coordinates along each of these directions. We describe a nonlinear generalization of PCA that uses an adaptive, multilayer “encoder” network

2006 VOL 313 SCIENCE www.sciencemag.org

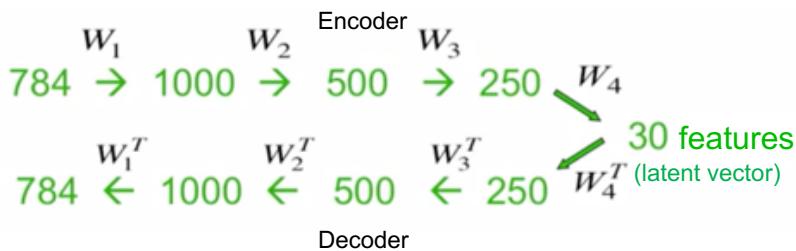
From Hinton and Salakhutdinov, Science, Science, July 2006
<https://www.youtube.com/watch?v=Ex1Ur85AVs>

11

Imperial College
London

Autoencoder Historical Example on MNIST

Structure of Hinton and Salakhutdinov's autoencoders (loss function is cross-entropy)



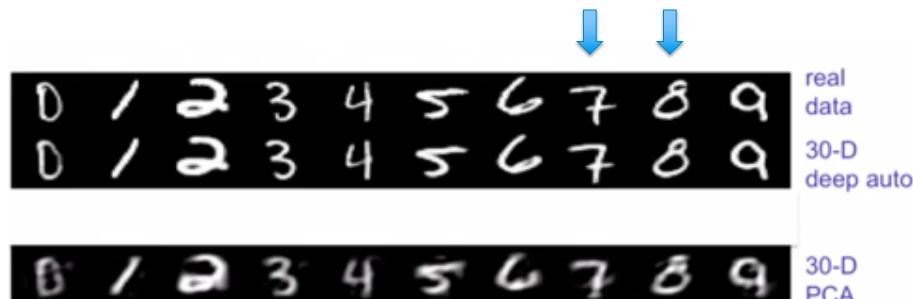
Each MNIST image represented now in $p=30$ dimensions instead of originally in $28 \times 28 = 784$ dimensions.

From Hinton and Salakhutdinov, Science, Science, July 2006
<https://www.youtube.com/watch?v=Ex1Ur85AVs>

12

Imperial College
London

Autoencoder Historical Example on MNIST



The reconstructed numbers (second line) look better than the originals, because the autoencoder model does not have enough capacity to represent the subtle variations/noise in the data.

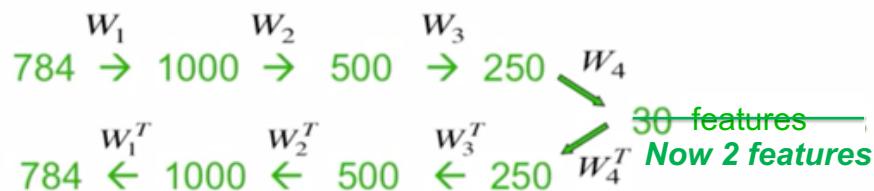
From Hinton and Salakhutdinov, Science, Science, July 2006
<https://www.youtube.com/watch?v=Ex1Ur85AVs>

13

Imperial College
London

Autoencoder Historical Example on MNIST

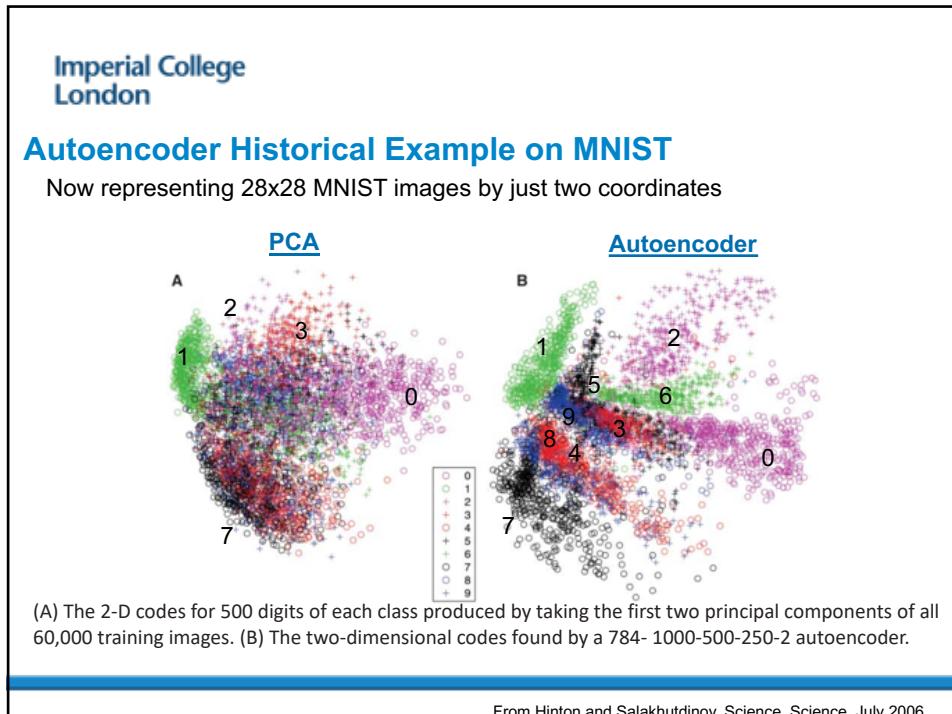
Now using a 2-dimensional instead of 30-dimensional code



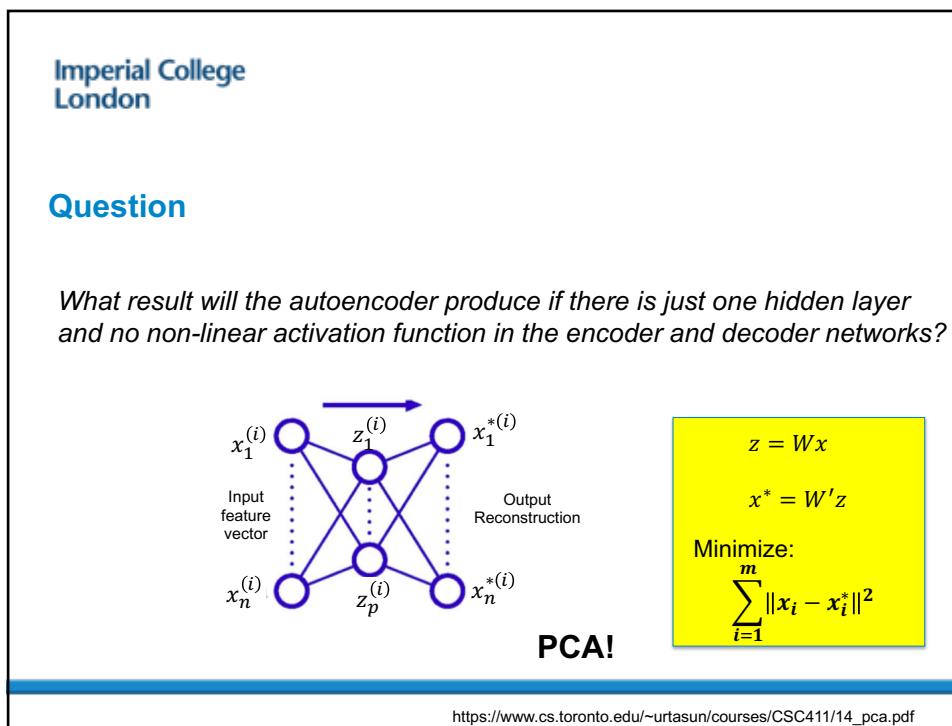
*Each MNIST number represented now in $p=2$ dimensions
instead of originally in $28 \times 28 = 784$ dimensions!*

From Hinton and Salakhutdinov, Science, Science, July 2006
<https://www.youtube.com/watch?v=Ex1Ur85AVs>

14



15



16

Imperial College
London

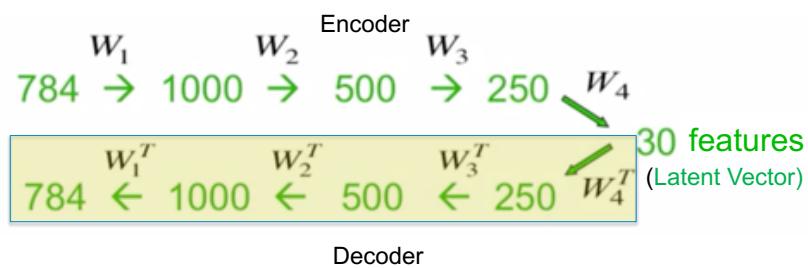
Introduction to Generative Models

1. PCA and Autoencoders
2. Transposed Convolution
3. Variational Autoencoders
4. Generative Adversarial Networks

17

Imperial College
London

Transposed Convolution Filters

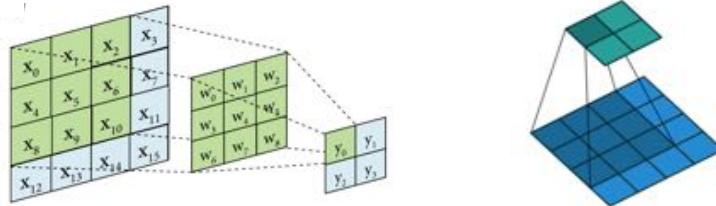


18

Imperial College
London

Encoder: Convolution Reduces Dimensionality

Simple 2-D Convolution Example (stride 1, no-padding)



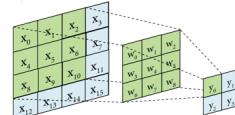
$$y_0 = w_0 x_0 + w_1 x_1 + w_2 x_2 + w_3 x_4 + w_4 x_5 + w_5 x_6 + w_6 x_8 + w_7 x_9 + w_8 x_{10} + b$$

Can we write the Convolution Operation in Matrix Form?

https://github.com/vdumoulin/conv_arithmetic
<https://towardsdatascience.com/up-sampling-with-transposed-convolution-9ae4f2df52d0>

19

Imperial College
London



Encoder: Writing Convolution as a Matrix Operation

$$\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} w_0 & w_1 & w_2 & 0 & w_3 & w_4 & w_5 & 0 & w_6 & w_7 & w_8 & 0 & 0 & 0 & 0 & 0 \\ 0 & w_0 & w_1 & w_2 & 0 & w_3 & w_4 & w_5 & 0 & w_6 & w_7 & w_8 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & w_0 & w_1 & w_2 & 0 & w_3 & w_4 & w_5 & 0 & w_6 & w_7 & w_8 & 0 \\ 0 & 0 & 0 & 0 & 0 & w_0 & w_1 & w_2 & 0 & w_3 & w_4 & w_5 & 0 & w_6 & w_7 & w_8 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \\ x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{15} \end{pmatrix}$$

Convolution can be written in matrix form $y = Wx$ with x reshaped as a 16×1 vector, y reshaped as a 4×1 vector, and a 4×16 matrix W .

https://github.com/vdumoulin/conv_arithmetic
<https://towardsdatascience.com/up-sampling-with-transposed-convolution-9ae4f2df52d0>

20

Imperial College London

Decoder: Transposed Convolution to Increase Dimensionality

Input image

Output image

y' reshaped into 4×1 vector, x' reshaped into 16×1 vector and $W^T 16 \times 4$ transposed of W

$$x' = W^T y'$$

$$\begin{pmatrix} x'_0 \\ x'_1 \\ x'_2 \\ x'_3 \\ x'_4 \\ x'_5 \\ x'_6 \\ x'_7 \\ x'_8 \\ x'_9 \\ x'_ {10} \\ x'_ {11} \\ x'_ {12} \\ x'_ {13} \\ x'_ {14} \\ x'_ {15} \end{pmatrix} = \begin{pmatrix} w_0 & 0 & 0 & 0 \\ w_1 & w_0 & 0 & 0 \\ w_2 & w_1 & 0 & 0 \\ 0 & w_2 & 0 & 0 \\ w_3 & 0 & w_0 & 0 \\ w_4 & w_3 & w_1 & w_0 \\ w_5 & w_4 & w_2 & w_1 \\ 0 & w_5 & 0 & w_2 \\ w_6 & 0 & w_3 & 0 \\ w_7 & w_6 & w_4 & w_3 \\ w_8 & w_7 & w_5 & w_4 \\ 0 & w_8 & 0 & w_5 \\ x'_ {11} & 0 & 0 & w_6 \\ x'_ {12} & 0 & 0 & w_7 \\ x'_ {13} & 0 & 0 & w_8 \\ 0 & 0 & 0 & w_9 \end{pmatrix} \times \begin{pmatrix} y'_0 \\ y'_1 \\ y'_2 \\ y'_3 \end{pmatrix}$$

https://github.com/vdumoulin/conv_arithmetic

21

Imperial College London

Transposed Convolution can also be Calculated by Convolution

Apply padding = 2 to array

$$\begin{matrix} y'_0 & y'_1 \\ y'_2 & y'_3 \end{matrix}$$

And convolve by filter W with stride 1.

Blue : input image
Green: output image

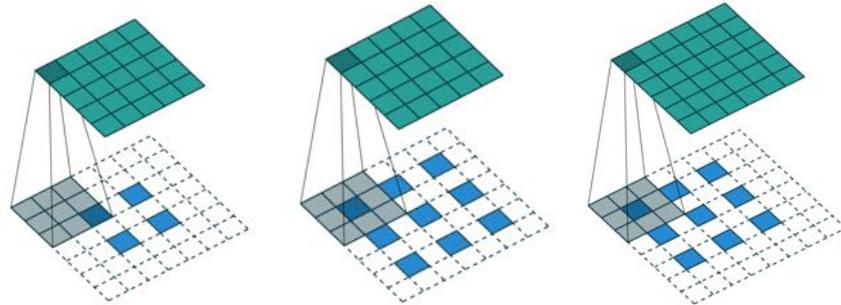
See afternoon exercise

https://github.com/vdumoulin/conv_arithmetic

22

Imperial College
London

Transposed Convolution can also be Calculated by Convolution



Blue : input image
Green: output image

Issue of checker board artefacts: <https://distill.pub/2016/deconv-checkerboard/>

https://github.com/vdumoulin/conv_arithmetic

23

Imperial College
London

Approach to calculate Transposed Convolution Operator

1. Identify the associated Direct Convolution
2. Calculate the Matrix associated with the Direct Convolution
3. Obtain Transposed Convolution from Transposed Matrix
4. Identify the Direct Convolution mimicking the Transposed one

<https://arxiv.org/pdf/1603.07285.pdf>

24

Imperial College
London

Introduction to Generative Models

1. PCA and Autoencoders
2. Transposed Convolution
3. Variational Autoencoders
4. Generative Adversarial Networks

25

Imperial College
London

Which Use for Autoencoders?



Once the autoencoder has been trained, throw away the calibrated Decoder and use the Encoder for Dimensionality Reduction!

Every Image of MNIST Dataset now represented by only 30 features instead of by 784 pixel values.

Possible Application: a Supervised Learning exercise on the MNIST Dataset can use the 30 features as input features, instead of the 784 pixel values. Helps avoid overfitting the $x^{(i)}$ when not many input images are labelled.

From Hinton and Salakhutdinov, Science, Science, July 2006

26

Imperial College London

Autoencoders only Encode and Decode Existing Images

The diagram illustrates an autoencoder architecture. On the left, input data (a grid of images) is processed by an Encoder (4-layer Convolution) to produce Latent Vectors (Reconstructed data). On the right, a 2D scatter plot shows latent vectors for different classes (0-9) as colored points. A legend indicates: 0 (light blue circle), 1 (light green circle), 2 (light red circle), 3 (light orange circle), 4 (light purple circle), 5 (light pink circle), 6 (light yellow circle), 7 (light teal circle), 8 (light blue plus), and 9 (light green plus). Three arrows point from the text 'Opportunities to generate new images?' to the plot area, with question marks. Below the plot, the text reads: "... we have not used the decoder yet. Why not make it generate more random images than just the existing ones?"

Encoder: 4-layer Convolution
Decoder: 4-layer Transposed Convolution

Reconstructed data

Input data

B

Opportunities to generate new images?

... we have not used the decoder yet. Why not make it generate more random images than just the existing ones?

<https://www.youtube.com/watch?v=5WoltGTWV54>

From Hinton and Salakhutdinov, Science, Science, July 2006

27

Imperial College London

Variational Autoencoders (VAEs)

The diagram shows a VAE architecture. High-Dimension Input Data ($(x_i)_{i=1 \dots m}$) is processed by an Encoder to produce Low-Dimension Latent Vectors ($(z_i)_{i=1 \dots m}$). These latent vectors are then processed by a Decoder to produce Reconstructed High-Dimension Input Data ($(x'_i)_{i=1 \dots m}$).

High-Dimension Input Data
 $(x_i)_{i=1 \dots m}$

Encoder

Low-Dimension Latent Vectors
 $(z_i)_{i=1 \dots m}$

Decoder

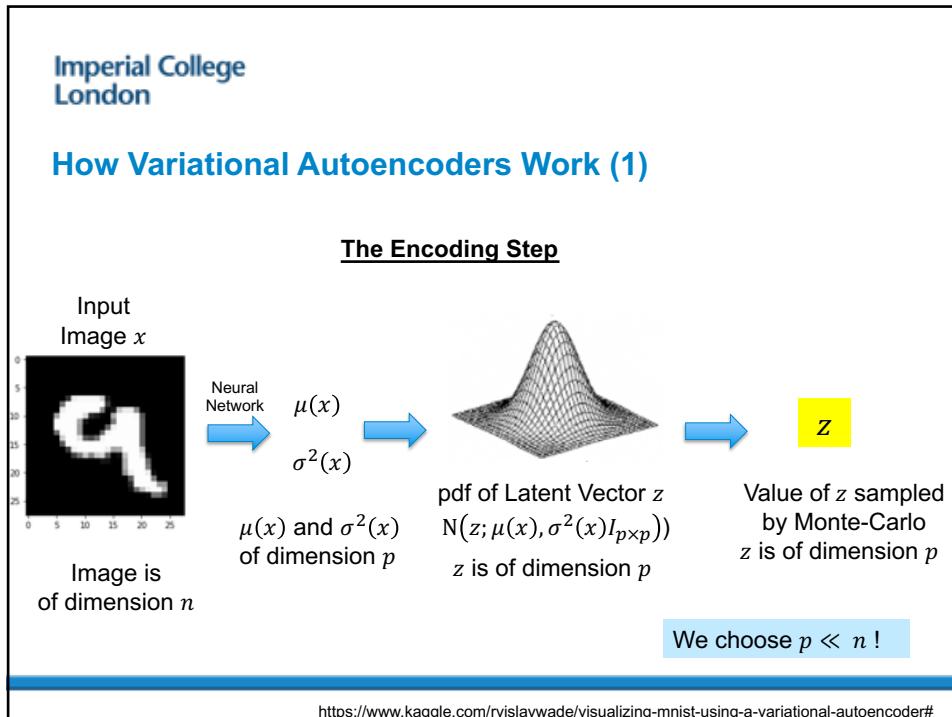
Reconstructed High-Dimension Input Data
 $(x'_i)_{i=1 \dots m}$

Variational autoencoders will be used to generate new images, that is images that are "variations" or belong to the same general pdf as images from the training set .

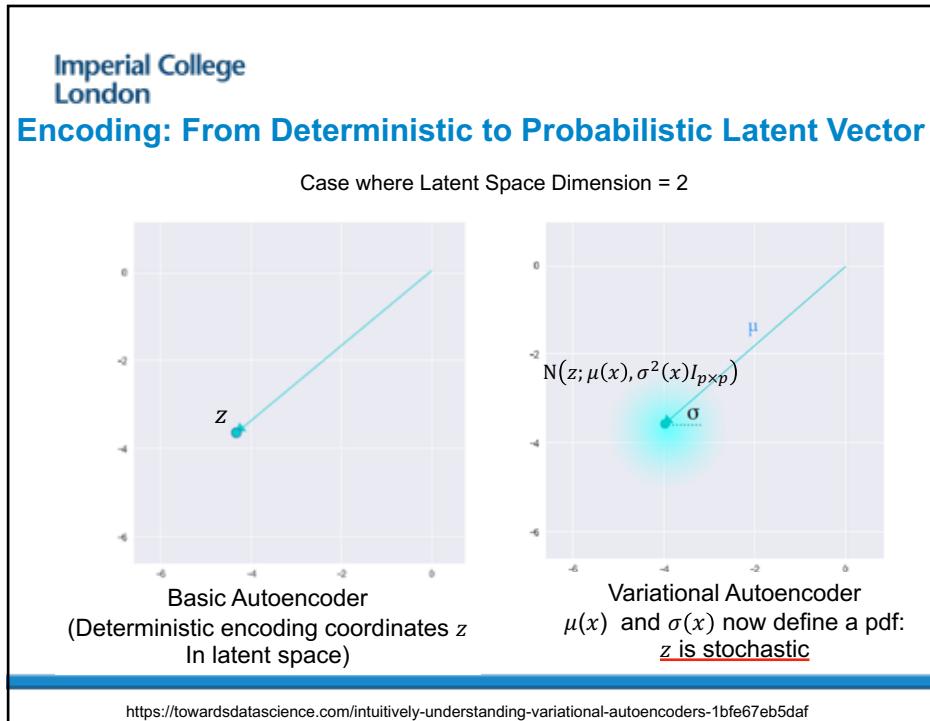
Make z and x random, instead of keeping them deterministic.

By sampling z, we will generate images that are not in the Training Set!

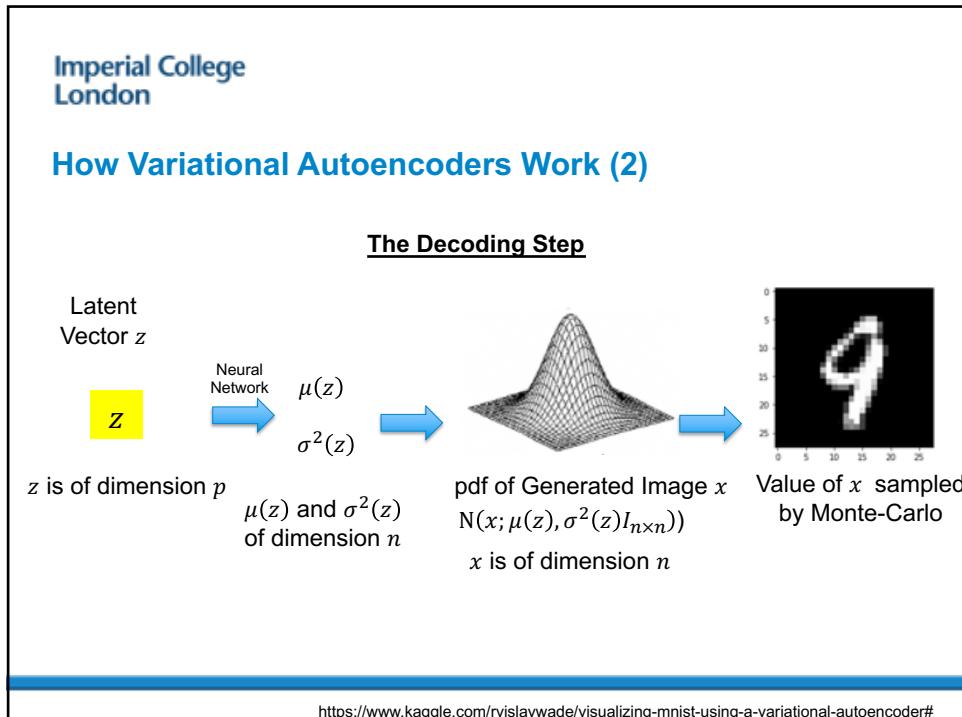
28



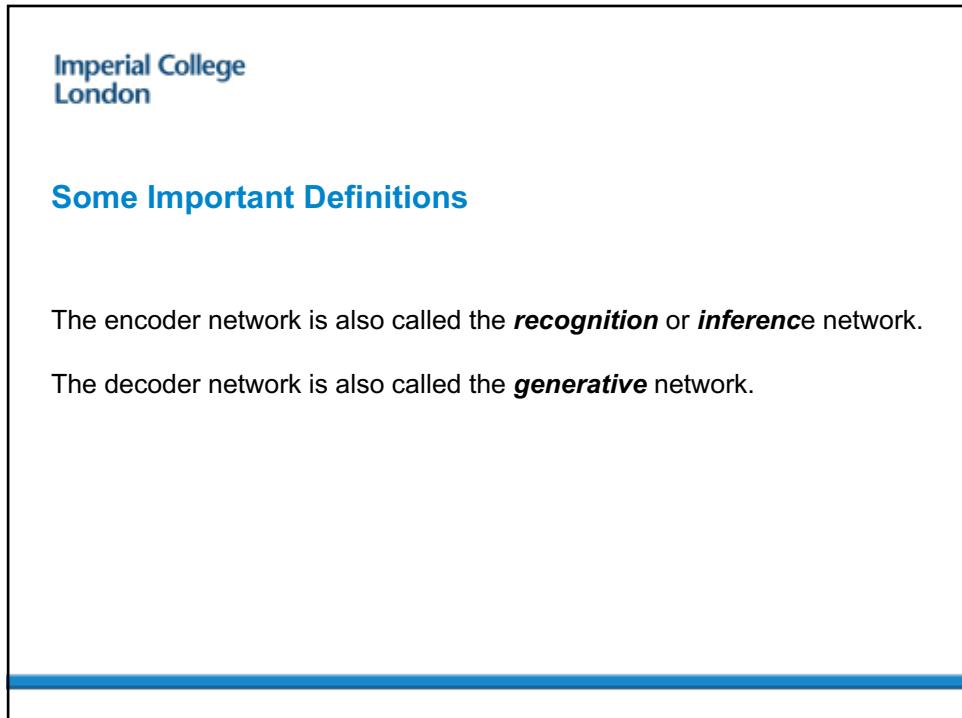
29



30



31



32

Imperial College
London

How Variational Autoencoders Work (3)

Call $q_\phi(z/x)$ the multivariate normal diagonal pdf $N(z; \mu(x), \sigma^2(x)I_{p \times p})$ of dimension p

$q_\phi(z/x)$ is the pdf of the latent vectors z associated with a given image x

$\mu(x)$ and $\sigma(x)$, which have the dimension p of z , are calculated from x using a neural network. The parameters of this neural network are the ϕ 's of $q_\phi(z/x)$.

Call $p_\theta(x/z)$ the multivariate normal diagonal pdf $N(x; \mu(z), \sigma^2(z)I_{n \times n})$ of dimension n

$p_\theta(x/z)$ is the pdf of the images x associated with a given latent vector z

$\mu(z)$ and $\sigma(z)$, which have the dimension n of x , are calculated from z using a neural network. The parameters of this neural network are the θ 's of $p_\theta(x/z)$.

33

Imperial College
London

The Variational Autoencoder Model (in case of images)

Training Set Images $(x^{(i)})_{i=1,\dots,m}$



Going back-and-forth between images $x^{(i)}$ and Latent vectors $z^{(i)}$:

How to calculate the parameters of the Neural Networks calculating $\mu(x)$ and $\sigma(x)$ from x and $\mu(z)$ and $\sigma(z)$ from z ?

By maximizing the likelihood of the Training Set images $(x^{(i)})_{i=1,\dots,m}!$

34

Imperial College
London

How Variational Autoencoders Work (4) (Kingma & Welling, 2014)

For one single data point $x^{(i)}$ the exact log-likelihood $\log p(x^{(i)})$ cannot be calculated, only a lower bound can

$$\log p(x^{(i)}) \geq E_{z \sim q_\phi(z/x^{(i)})}(\log p_\theta(x^{(i)}|z)) - \text{KL}(q_\phi(z/x^{(i)})||q(z))$$

The right-hand side is called the Evidence Lower Bound (ELBO).
The ELBO is negative, as the sum of two negative terms.

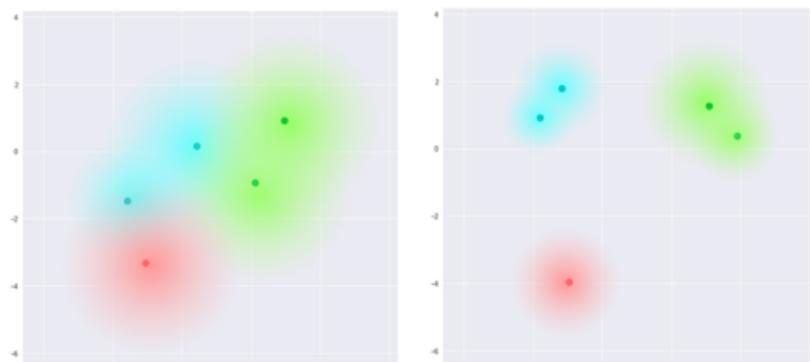
The first term of the ELBO is a **reconstruction likelihood**, quantifying the quality of the match to the data point $x^{(i)}$, or of the « round trip » between the training image $x^{(i)}$ and its associated latent vector pdf $q_\phi(z/x^{(i)})$. The second one is a **KL Divergence** that aims to « spread » the range of latent vectors associated to the image $x^{(i)}$ by making their pdf close to a multivariate standardized Gaussian $q(z)$.

For a mini-batch of m images $(x^{(i)})_{i=1,\dots,m}$, the ELBO is summed over the data.

35

Imperial College
London

If Only First Term of Objective Function is Used...



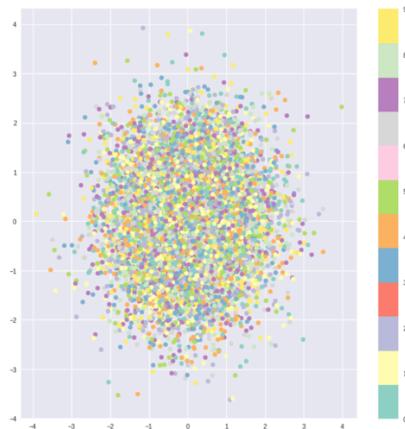
We want a continuous latent space.....But the Likelihood Term tends to create clusters

<https://towardsdatascience.com/intuitively-understanding-variational-autoencoders-1bfe67eb5daf>

36



If Only Second Term of Objective Function is Used



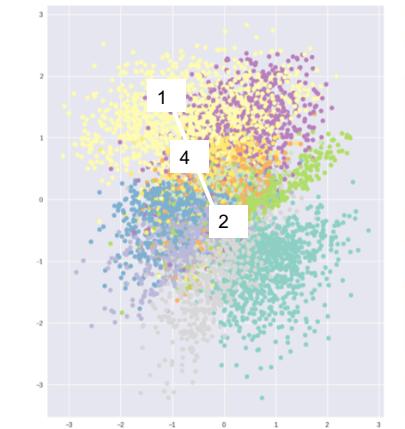
Now the space is nicely continuous but there is no more difference between coded digits!

<https://towardsdatascience.com/intuitively-understanding-variational-autoencoders-1bfe67eb5daf>

37



If Both Terms of Objective Function are Used



Digits are nicely separated but in a continuous space!

<https://towardsdatascience.com/intuitively-understanding-variational-autoencoders-1bfe67eb5daf>

38

This slide can be skipped**How Variational Autoencoders Work (5) (Kingma & Welling, 2014)****Calculating $-KL(q_\phi(z/x^{(i)})||q(z))$** This is the KL divergence between two Gaussian distributions, each of dimension p :

$$q_\phi(z/x^{(i)}) \text{ is } N(z; \mu(x^{(i)}), \sigma^2(x^{(i)})I_{p \times p}) \text{ or } N(z; \mu^{(i)}, \sigma^{(i)2})$$

$$q(z) \text{ is } N(z; 0_p, I_{p \times p})$$

Using the formula giving the KL divergence of two multivariate Gaussians (see next slide) we get

$$-KL(q_\phi(z/x^{(i)})||q(z)) = \frac{1}{2} \sum_{j=1}^p \left[1 + \log \left((\sigma_j^{(i)})^2 \right) - (\mu_j^{(i)})^2 - (\sigma_j^{(i)})^2 \right]$$

Where p is the dimension of the latent vector**This slide can be skipped****Recall: KL divergence of 2 Multivariate Gaussians**Suppose the two distributions are: $N(\mu_1, \Sigma_1)$ and $N(\mu_2, \Sigma_2)$, each of dimension p .

The KL divergence between the two distributions is:

$$KL = \frac{1}{2} \left[\log \frac{\det \Sigma_2}{\det \Sigma_1} - p + \text{tr}(\Sigma_2^{-1} \Sigma_1) + (\mu_2 - \mu_1)^T \Sigma_2^{-1} (\mu_2 - \mu_1) \right]$$

In our case the first distribution is that of the conditional Gaussian of mean $\mu = \begin{pmatrix} \mu_1 \\ \vdots \\ \mu_p \end{pmatrix}$ anddiagonal variance-covariance $\Sigma = \begin{pmatrix} \sigma_1^2 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_p^2 \end{pmatrix}$ and the second one is thestandardized multivariate normal: $\mu_2 = 0$ and $\Sigma_2 = I_{p \times p}$. Thus, in our case:

$$KL = \frac{1}{2} \left[\log \frac{1}{\prod_{i=1}^p \sigma_i^2} - p + \sum_{i=1}^p \sigma_i^2 + \sum_{i=1}^p \mu_i^2 \right] = \frac{1}{2} \left[- \sum_{i=1}^p (1 + \log(\sigma_i^2)) + \sum_{i=1}^p (\sigma_i^2 + \mu_i^2) \right]$$

*This slide can be skipped***How Variational Autoencoders Work (6) (Kingma & Welling, 2014)****Calculating $E_{z \sim q_\phi(z/x^{(i)})}(\log p_\theta(x^{(i)}/z))$ for one data point $x^{(i)}$**

This expression looks complex but simply quantifies the likelihood associated with the « round trip » between the image x_i and the pdf of associated latent vectors z . This is an expectation (because z does not take just one value as with autoencoders, but many possible values), which can be calculated by Monte Carlo sampling and averaging over a large set of L simulated values of z .

$$E_{z \sim q_\phi(z/x^{(i)})}(\log p_\theta(x^{(i)}/z)) = \frac{1}{L} \sum_{l=1}^L \log p_\theta(x^{(i)}/z^{(i,l)})$$

Since $z^{(i,l)}$ must be a sample from the multivariate diagonal normal pdf $q_\phi(z/x^{(i)})$ of mean $\mu^{(i)} = \mu(x_i)$ and variance $\sigma_l^2 = \sigma^2(x_i)$, we write:

$$z^{(i,l)} = \mu^{(i)} + \sigma^{(i)} \odot \varepsilon^{(l)} \text{ where } \varepsilon^{(l)} \text{ is } N(\varepsilon; 0, I_{p \times p})$$

41

*This slide can be skipped***Likelihood for Decoder $p_\theta(x/z)$ Multivariate Diagonal Gaussian**

$$f(x) = (2\pi)^{-\frac{n}{2}} |\Sigma|^{-\frac{1}{2}} \exp \left[-\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu) \right]$$

If $\Sigma = \sigma^2 I_n$, this simplifies into: $f(x) = (2\pi)^{-\frac{n}{2}} \frac{1}{\prod_{j=1}^n \sigma_j} \exp \left[-\frac{1}{2} \sum_{i=1}^n \left(\frac{x_j - \mu_j}{\sigma_j} \right)^2 \right]$

$$\log p_\theta(x^{(i)}/z^{i,l}) = -\frac{n}{2} \log(2\pi) - \sum_{j=1}^n \log(\sigma_j(z^{i,l})) - \frac{1}{2} \sum_{j=1}^n \left(\frac{x_j^{(i)} - \mu_j(z^{i,l})}{\sigma_j(z^{i,l})} \right)^2$$

<https://stats.stackexchange.com/questions/351549/maximum-likelihood-estimators-multivariate-gaussian>

42

*This slide can be skipped***Likelihood for Decoder $p_\theta(x/z)$ Multivariate Bernouilli**

$$\log p_\theta(x^{(i)} / z^{i,l}) = -\frac{1}{n} \sum_{j=1}^n \left[x_j^{(i)} \log(p_{j,l}^i) + (1 - x_j^{(i)}) \log(1 - p_{j,l}^i) \right]$$

Where $p_{j,l}^i$ is the j^{th} coordinate of the n – dimensional probability vector p_l^i associated (via neural network with final sigmoid activation) with $z^{i,l}$.

43

Summarizing how we calculate a VAE

We have defined the ELBO as the right-hand side of :

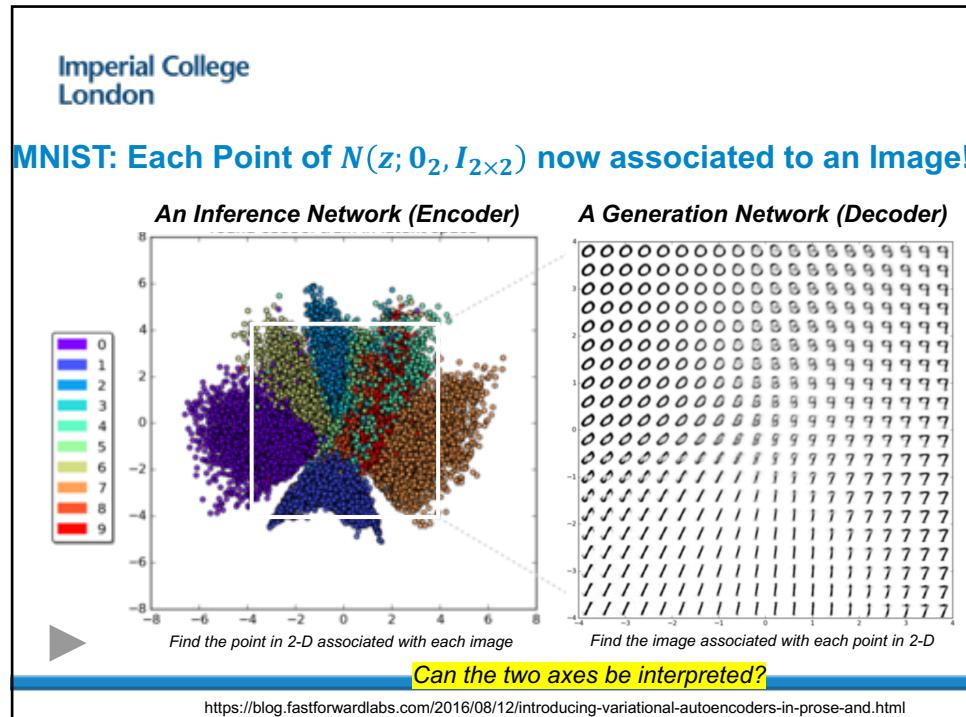
$$\log p(x^{(i)}) \geq E_{z \sim q_\phi(z/x^{(i)})} (\log p_\theta(x^{(i)}/z)) - \text{KL}(q_\phi(z/x^{(i)}) \| q(z))$$

$E_{z \sim q_\phi(z/x^{(i)})} (\log p_\theta(x^{(i)}/z))$ is a function of the vectors $\mu(x_i), \sigma^2(x_i)$ and $\mu(z), \sigma^2(z)$ for known samples of z ,

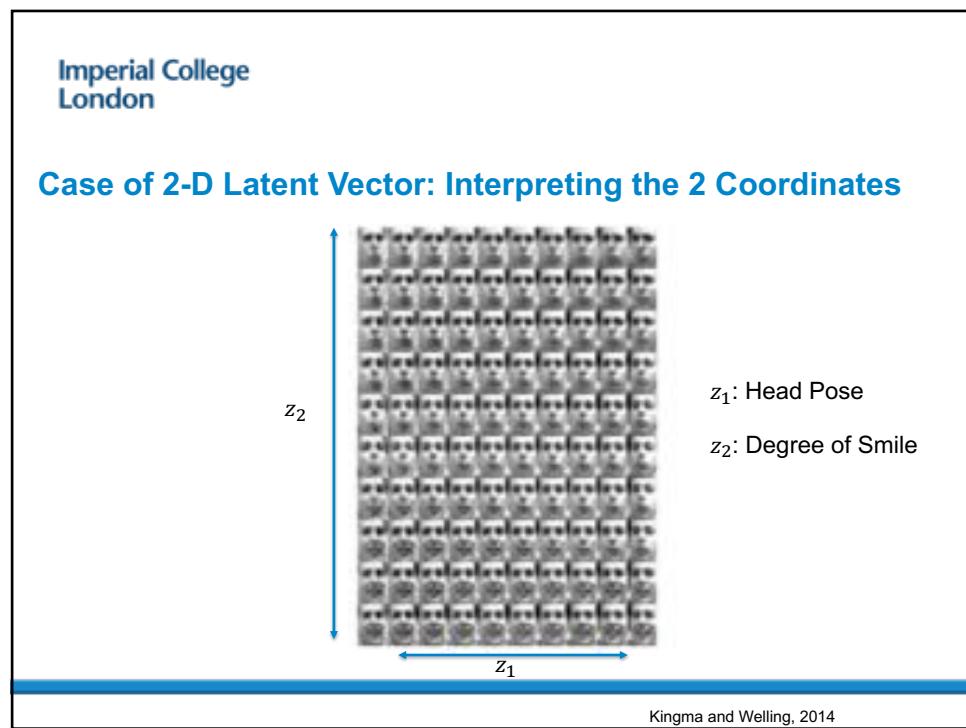
$-\text{KL}(q_\phi(z/x^{(i)}) \| q(z))$ is a function of the vectors $\mu(x_i), \sigma^2(x_i)$

The ELBO, sum of these two terms, is thus a function of the parameter vectors θ and ϕ of the neural networks respectively calculating $\mu(x), \sigma^2(x)$ as a function of x and $\mu(z), \sigma^2(z)$ as a function of z . θ and ϕ are obtained by maximizing the ELBO .

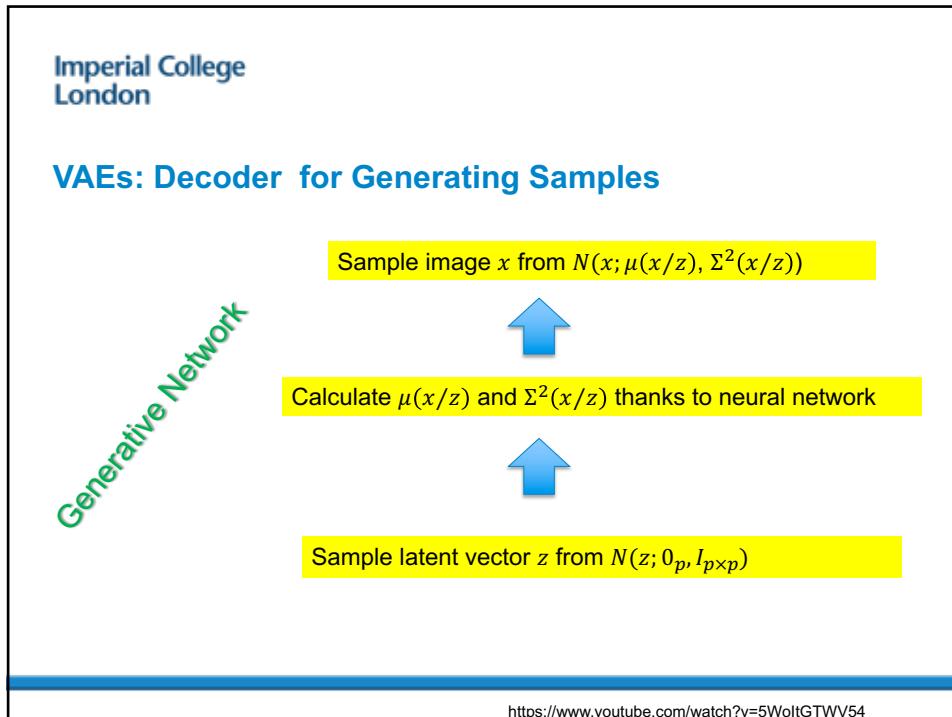
44



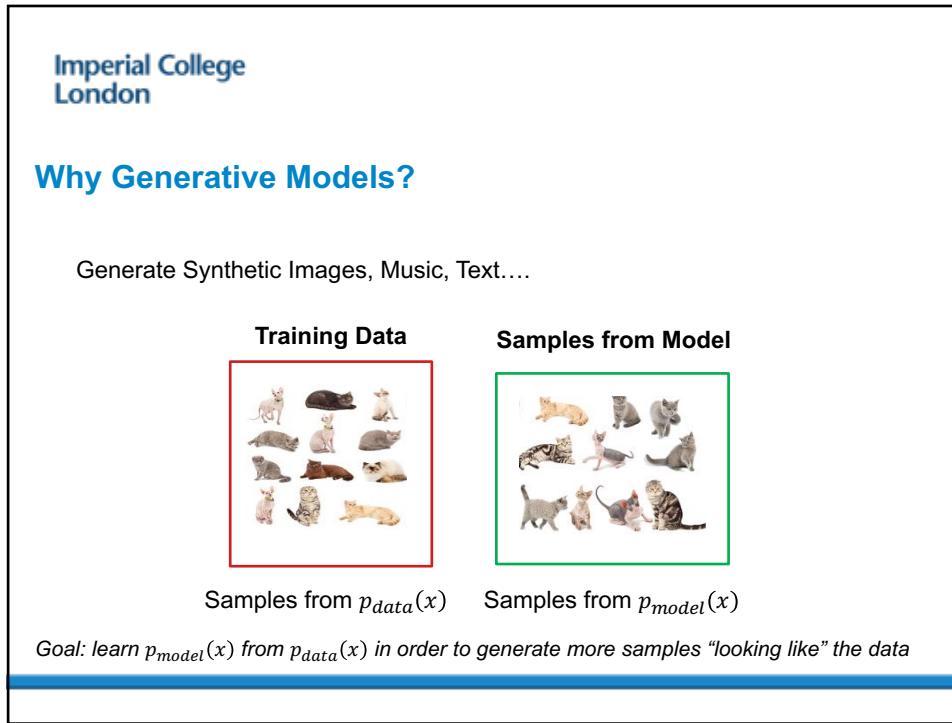
45



46



47



48

Imperial College
London

Introduction to Generative Models

1. PCA and Autoencoders
2. Transposed Convolution
3. Variational Autoencoders
4. Generative Adversarial Networks

49

Imperial College
London

Do you know any of these people?



50

Imperial College
London

Do you know any of these people?

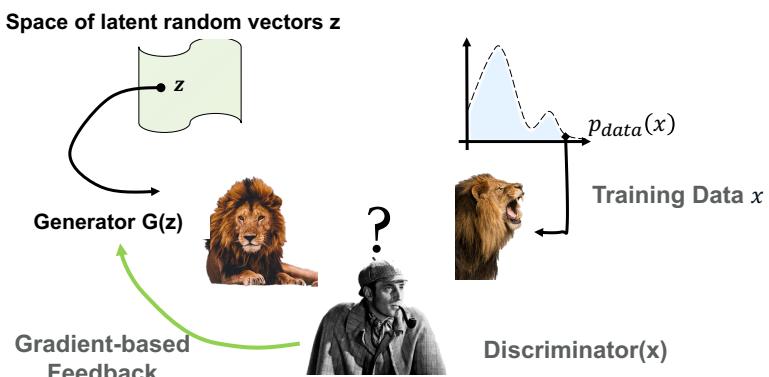


<https://thispersondoesnotexist.com/>

51

Imperial College
London

Generative Adversarial Networks – Toy Example



(Goodfellow et. al. 2014)

52

Imperial College
London

Compare two pdfs: Generative Adversarial Networks

If $x = G(z)$ is a sample from $p_{model}(x)$, a Discriminator function $D(x)$ is applied to x and calculates a number between 0 and 1. The closer $D(x)$ is to 1, the higher the probability that the Discriminator will accept x as a sample of $p_{data}(x)$.

Both $D(x)$ and $G(z)$ are neural networks.

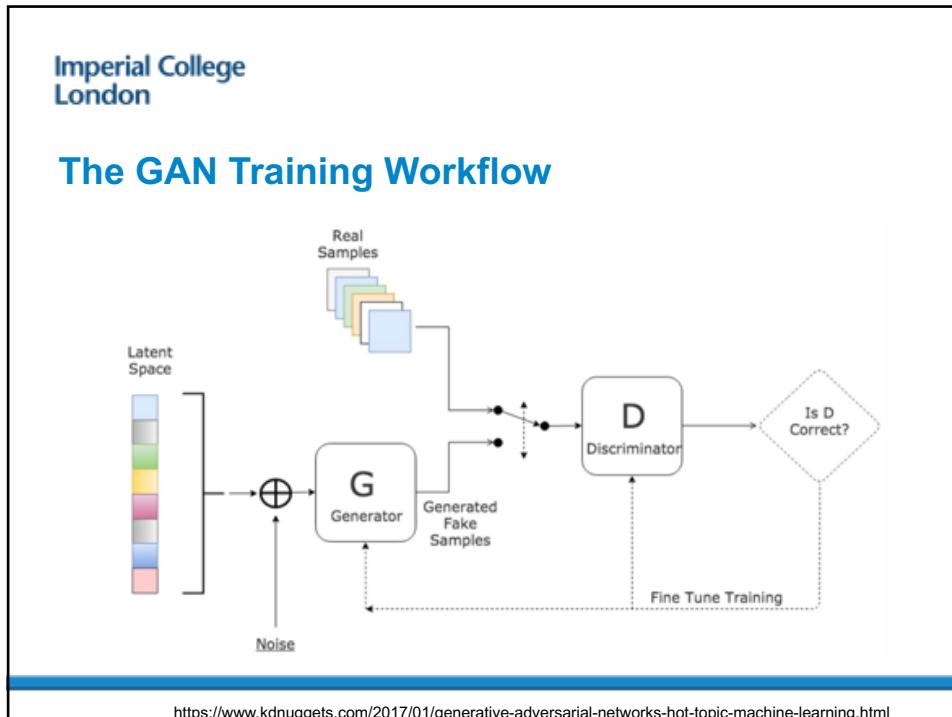
53

Imperial College
London

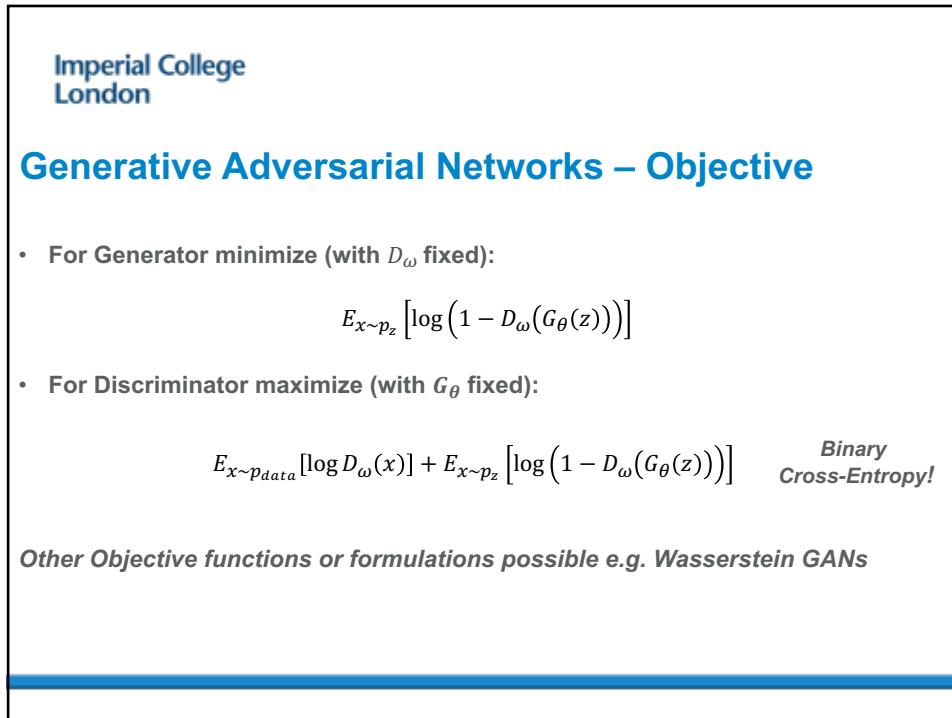
Generative Adversarial Networks

- Two Neural Networks are required:
 - Generator (or Decoder) – creates sample images x
 $x = G_\theta(z)$ with pdf of vector z : $N(z; 0_p, I_{p \times p})$
 - Discriminator – evaluates images, to help decide whether “fake” or “real”
 $D_\omega(x) \in [0,1]$

54



55



56

Imperial College
London

GAN Training Example - MNIST

Training Images



Training Iterations

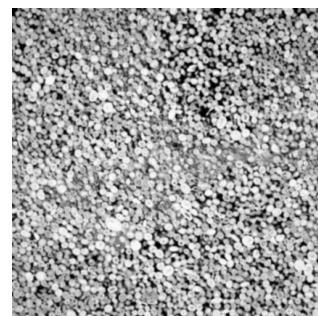
<https://www.youtube.com/watch?v=R7HTX79JIBg&feature=youtu.be>

57

Imperial College
London

Ketton Limestone Dataset and Preprocessing

- Oolitic Limestone
- Intergranular pores
- Intragranular Micro-Porosity
- Ellipsoidal grains
- 99% Calcite
- Image Size:
- 900^3 voxels @ $26.7 \mu\text{m}$

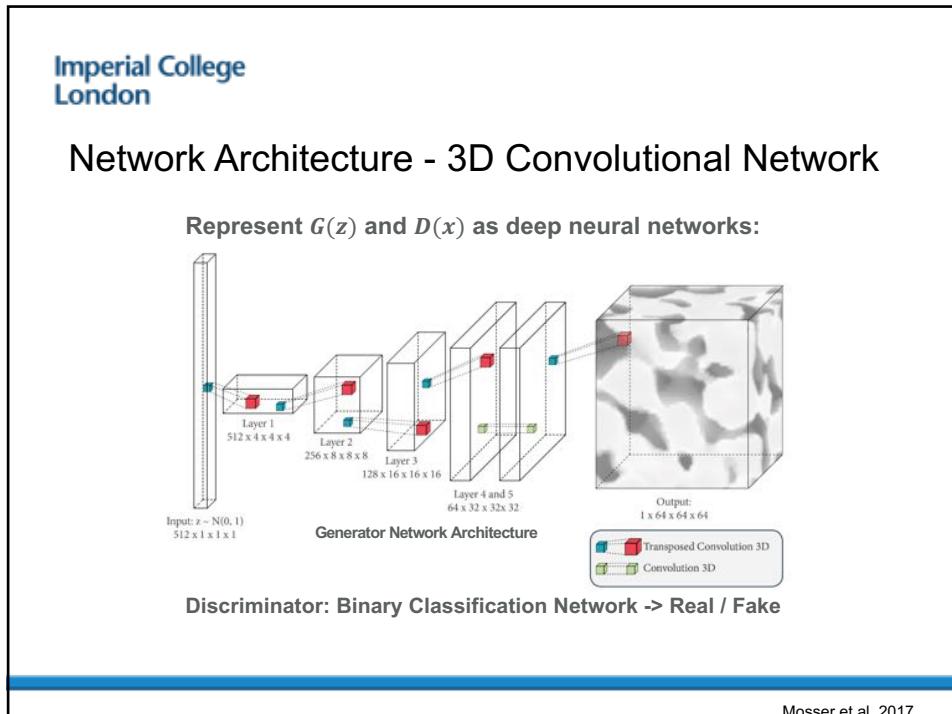


Extract Non-Overlapping
Training Images (64^3 voxels)

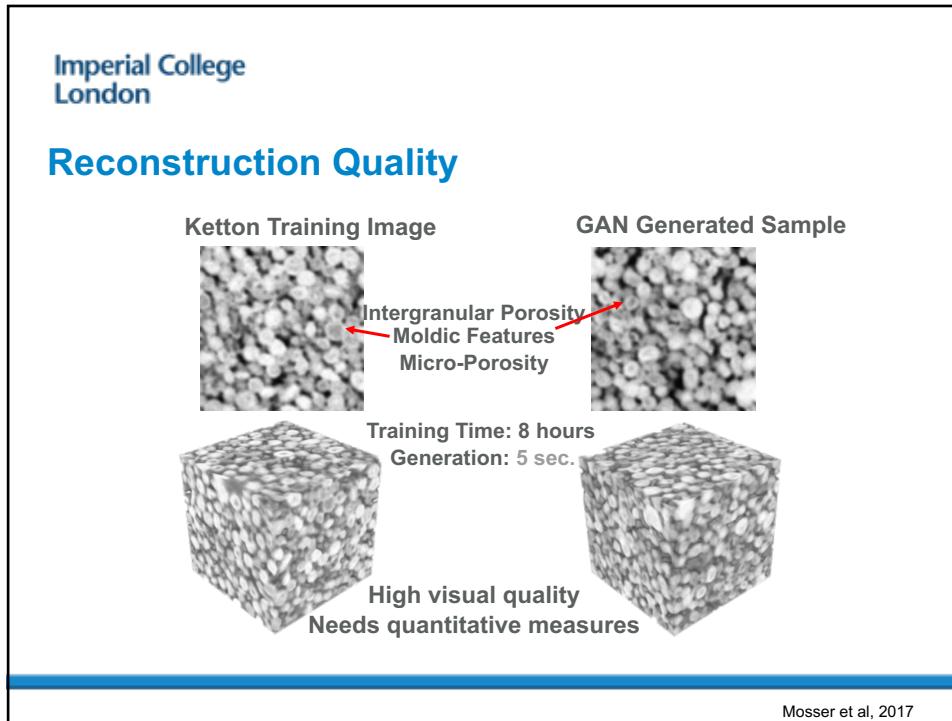


Mosser et al, 2017

58



59

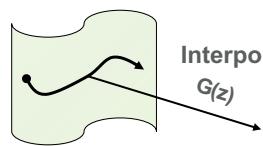


60

Imperial College
London

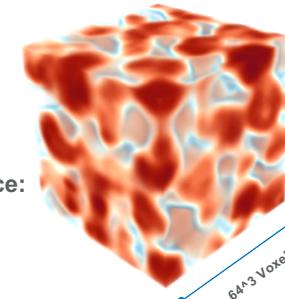
Latent Space Interpolation

Latent space \mathbf{z}



$$\mathbf{z}^* = \beta \mathbf{z}_{start} + (1 - \beta) \mathbf{z}_{end}, \beta \in [0, 1]$$

Interpolation path visualization



Interpolation in latent space:

Shows that generator has
learned a meaningful
representation in a lower
dimensional space!

Mosser et al, 2017

61

Imperial College
London

Latent Space Interpolation for fake celebrities

62

Imperial College London

Conditioning of Generative Models

Generative model can incorporate additional information:
 $G(z, class) \rightarrow G(z, class = \{Cat, Dog, Bird\})$

Train generative model on images with associated class labels

Generative model input: Latent Vector + Class

Possible to perform smooth inter-class interpolation

This model \rightarrow Trained on Imagenet (1000 classes)

(Cat, Dog, Leopard, Dachshund)

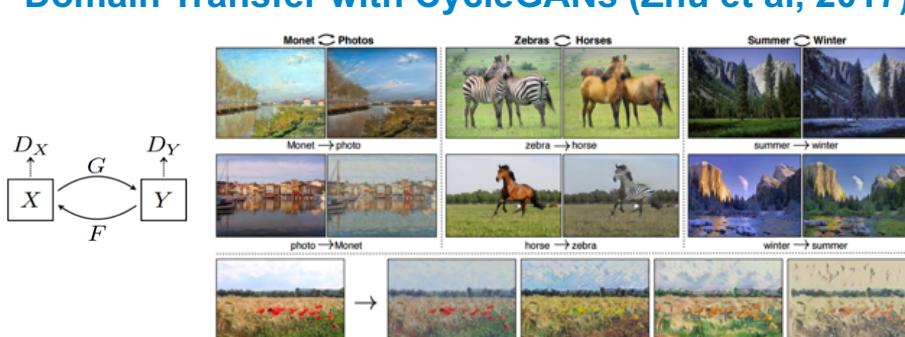


(Miyato and Koyama 2018)

63

Imperial College London

Domain Transfer with CycleGANs (Zhu et al, 2017)



Find functions $G(X)$ and $F(Y)$ to map between two unpaired domains

64

Imperial College
London

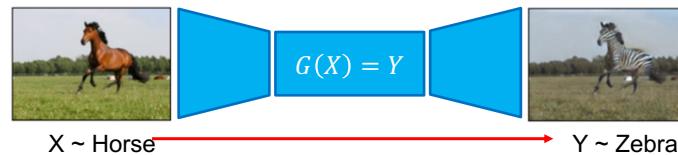
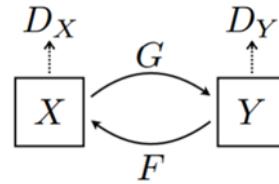
CycleGAN – Detailed Analysis (1)

CycleGAN consists of 4 Convolutional Neural Networks:

- 2 Transformer Networks: $G(X) = Y$ and $F(Y) = X$
- 2 Discriminators: $D(X)$ and $D(Y)$

Transformer Networks:

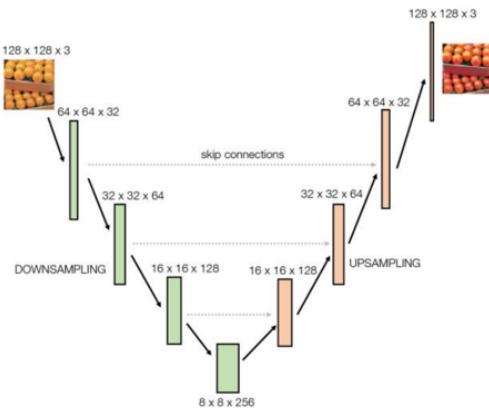
- Perform A \rightarrow B mapping, where A and B are images
e.g.: (Horse \rightarrow Zebra), (Winter \rightarrow Summer), (Blurred \rightarrow Detailed)



65

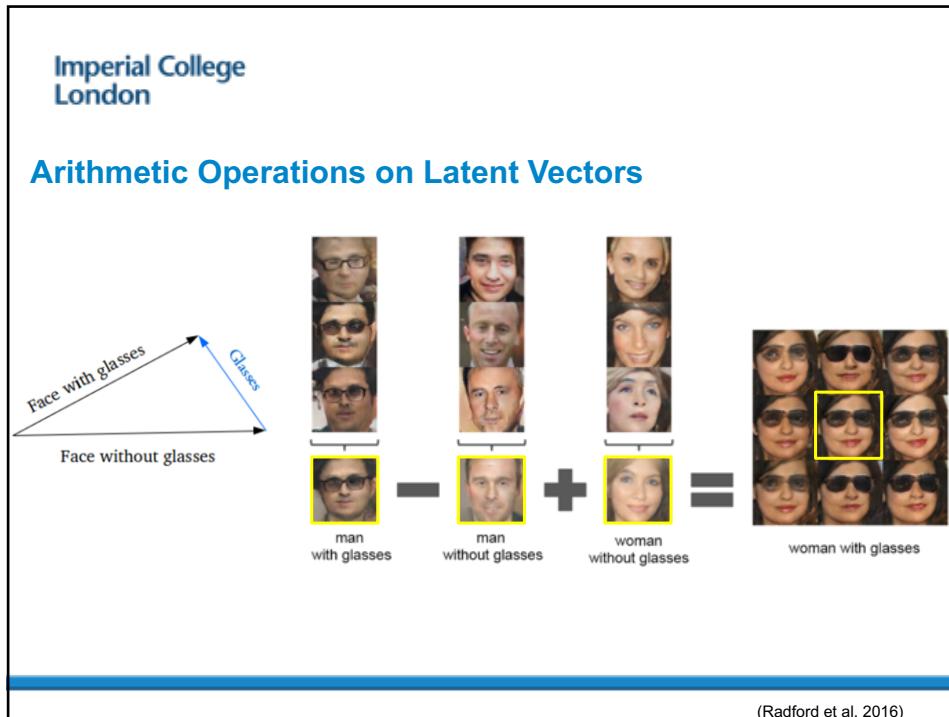
Imperial College
London

CycleGAN – Detailed Analysis (2)

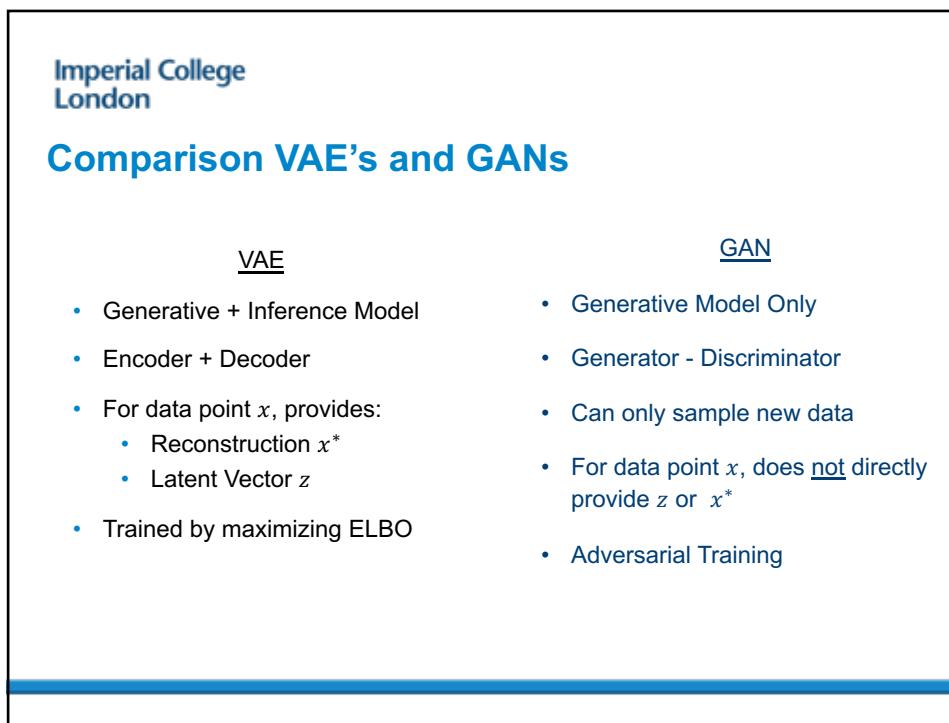


Example of U-Net Generator
for CycleGan (Foster, 2019)

66



67



68

Imperial College
London

Conclusion

- Autoencoders a Generalization of PCAs to non-Linear Manifolds
- Variational Autoencoders give a Probabilistic Spin to Autoencoders
- Variational Autoencoders produce both an Inference and a Generative Network
- Generative Adversarial Networks are a simpler Generative method, but which does not produce Inference.