Imperial College London

May 16th, 2019

Introduction to Generative Models

Olivier Dubrule and Lukas Mosser

Imperial College London

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

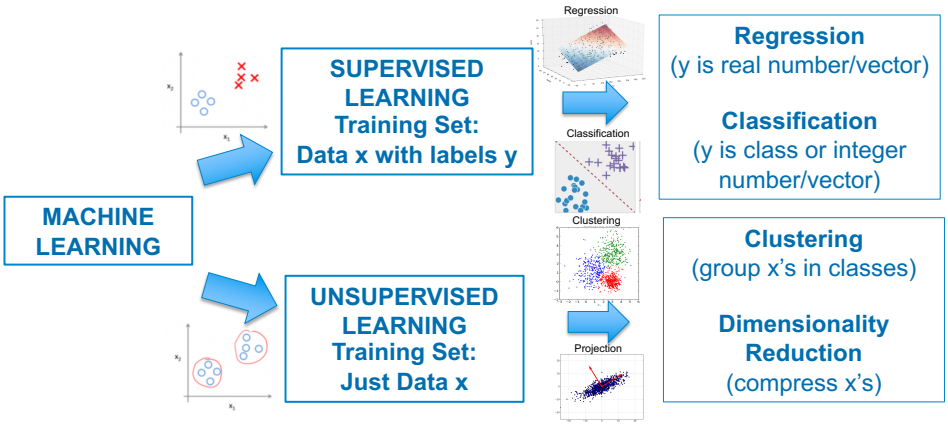
Imperial College London

Introduction to Generative Models

1. PCA and Autoencoders
2. Transposed Convolution
3. Variational Autoencoders
4. Generative Models Generalization

Imperial College London

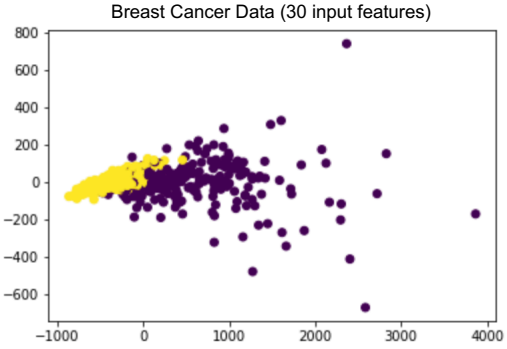
Supervised vs Unsupervised Learning



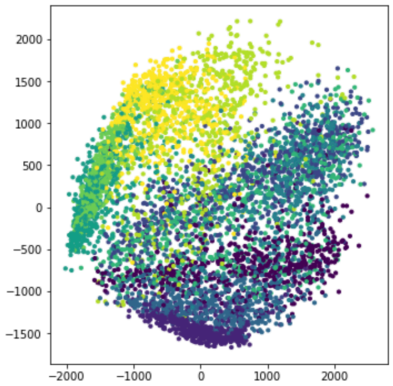
Imperial College London

PCA Dimensionality Reduction

Breast Cancer Data (30 input features)



Fashion MNIST (784 input features)



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:

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

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.

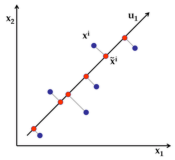
Imperial College London

Dimensionality Reduction with Latent Vectors

Idea

Represent each image $(x^{(i)})_{i=1,\dots,m}$ *where each* x_i *is of high dimension* n *(the number of features), by a set of latent vectors* $(z^{(i)})_{i=1,\dots,m}$, *where the vectors* $z^{(i)}$ *are of much smaller dimension* p *than that* n *of the images.*

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

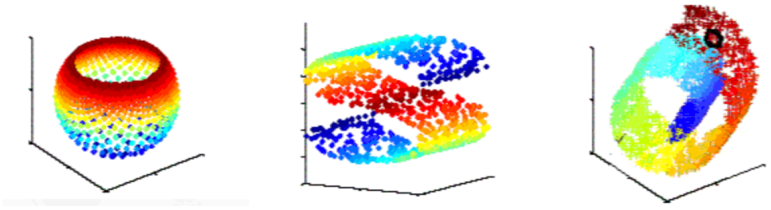


Potential applications

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….*

Imperial College London

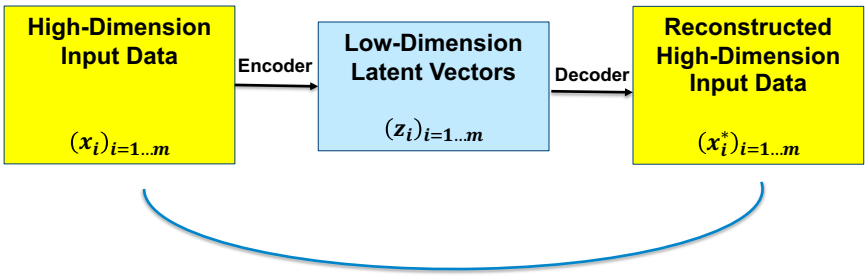
Why do we Need a Non-Linear Approach ?



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

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!

Imperial College London

Autoencoders Historical Example on MNIST

ired SHG
ce in Fig.
we find
the noise
il closely
nt power
emission
ngle with
ons from
RRs (see
detuning
avelength
the SHG
For ex-
vertical
d a small
xcitation
incident
ignificant
gain po-

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

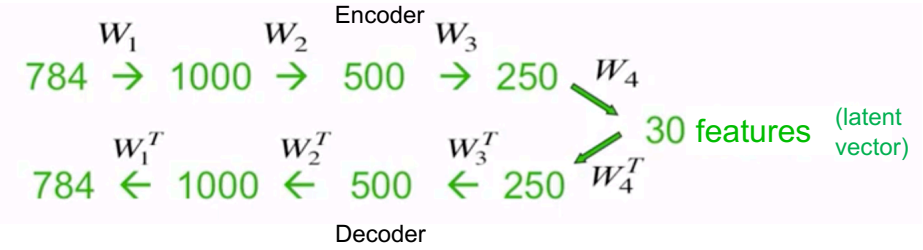
28 JULY 2006 VOL 313 SCIENCE www.sciencemag.org

From Hinton and Salakhutdinov, Science, Science, July 2006
https://www.youtube.com/watch?v=_Ex1Ur85AVs

Imperial College London

Autoencoder Historical Example on MNIST

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

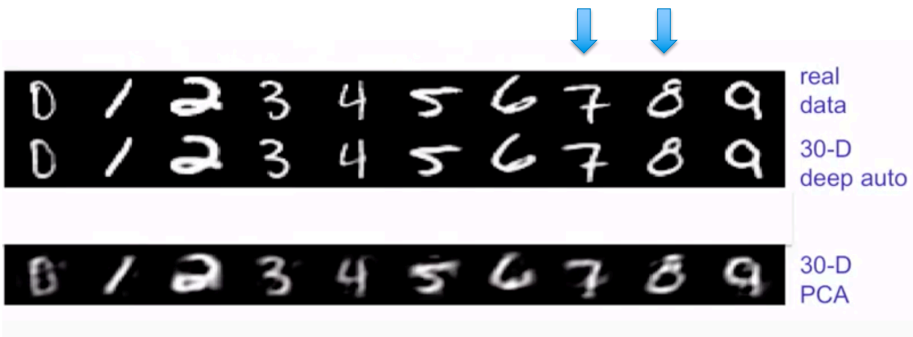


Each MNIST image represented now in 30 dimensions instead of originally in 28x28=784 dimensions.

From Hinton and Salakhutdinov, Science, Science, July 2006
https://www.youtube.com/watch?v=_Ex1Ur85AVs

Imperial College London

Autoencoder Historical Example on MNIST



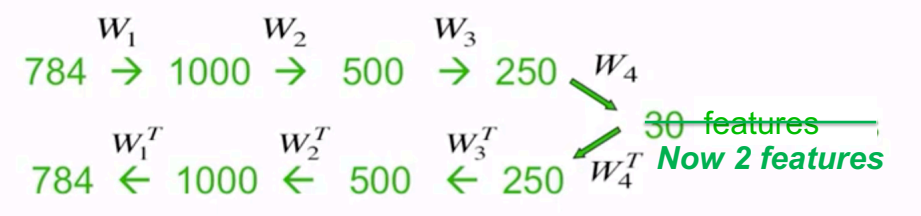
The reconstructed numbers (second line) look better than the originals, because the autoencoder model does not have enough capacity to represent the subtle variations/noise in the data.

From Hinton and Salakhutdinov, Science, Science, July 2006
https://www.youtube.com/watch?v=_Ex1Ur85AVs

Imperial College London

Autoencoder Historical Example on MNIST

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



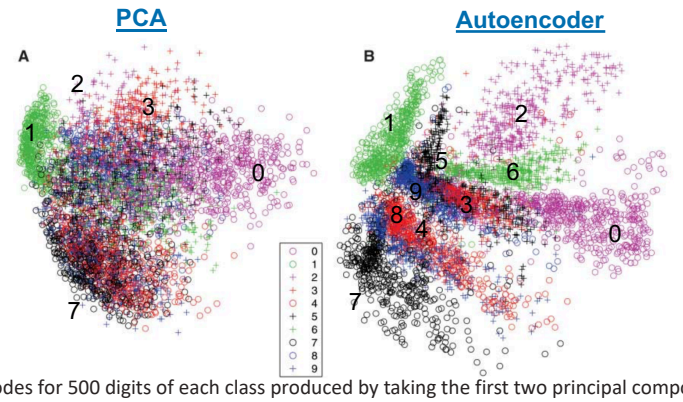
Each MNIST number represented now in 2 dimensions instead of originally in 28x28=784 dimensions!

From Hinton and Salakhutdinov, Science, Science, July 2006
https://www.youtube.com/watch?v=_Ex1Ur85AVs

Imperial College London

Autoencoder Historical Example on MNIST

Now representing 28x28 MNIST images by just two coordinates



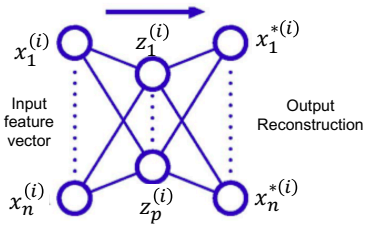
(A) The 2-D.codes for 500 digits of each class produced by taking the first two principal components of all 60,000 training images. (B) The two-dimensional codes found by a 784- 1000-500-250-2 autoencoder.

From Hinton and Salakhutdinov, Science, Science, July 2006

Imperial College London

Question

What result autoencoder will produce if there is just one hidden layer and no non-linear activation function in the encoder and decoder networks?



PCA!

$$z = Wx$$

$$x^* = W'z$$

Minimize:

$$\sum_{i=1}^{m} \|x_i - x_i^*\|^2$$

https://www.cs.toronto.edu/~urtasun/courses/CSC411/14_pca.pdf

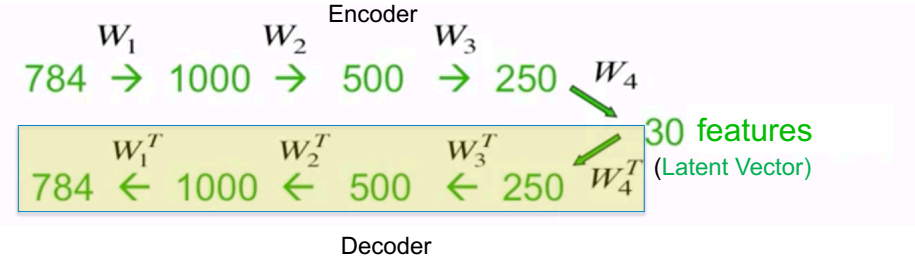
Imperial College London

Introduction to Generative Models

1. **PCA and Autoencoders**
2. **Transposed Convolution**
3. **Variational Autoencoders**
4. **Generative Models Generalization**

Imperial College London

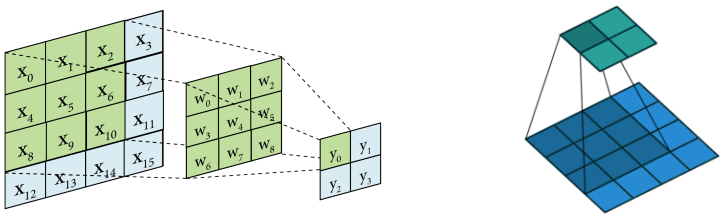
Transposed Convolution Filters



Imperial College London

Encoder: Convolution Reduces Dimensionality

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

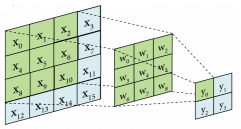


$$y_0 = w_0x_0+w_1x_1+w_2x_2+w_3x_4+w_4x_5+w_5x_6+w_6x_8+w_7x_9+w_8x_{10} + \boldsymbol{b}$$

Can we write the Convolution Operation in Matrix Form?

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

Imperial College London



Encoder: Writing Convolution as a Matrix Operation

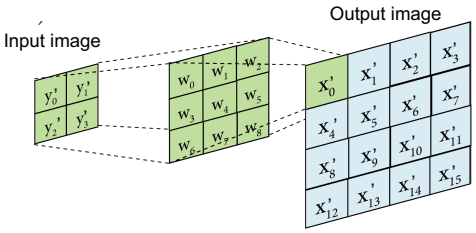
$$\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} w_0 & w_1 & w_2 & 0 & w_3 & w_4 & w_5 & 0 & w_6 & w_7 & w_8 & 0 & 0 & 0 & 0 & 0 \\ 0 & w_0 & w_1 & w_2 & 0 & w_3 & w_4 & w_5 & 0 & w_6 & w_7 & w_8 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & w_0 & w_1 & w_2 & 0 & w_3 & w_4 & w_5 & 0 &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

Convolution can be written in matrix form $\boldsymbol{y} = \boldsymbol{W}\boldsymbol{x}$ with x reshaped as a 16x1 vector, y reshaped as a 4x1 vector, and a 4x16 matrix W.

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

Imperial College London

Decoder: Transposed Convolution to Increase Dimensionality



y' reshaped into 4x1 vector, x' reshaped into 16x1 vector and W^T16x4 transposed of W

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

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

https://github.com/vdumoulin/conv_arithmetic

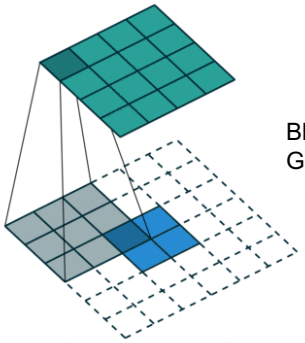
Imperial College London

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.



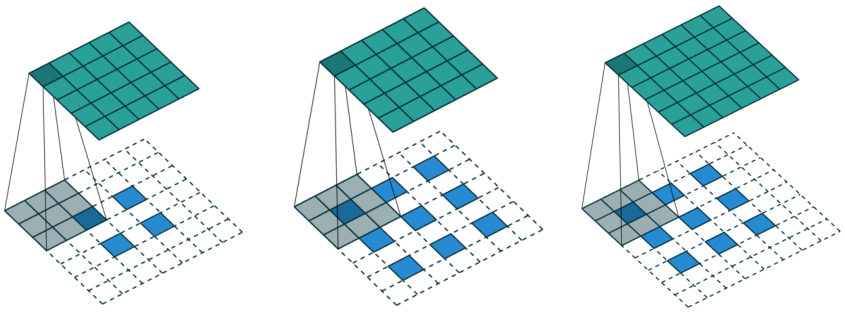
Blue : input image
Green: output image

See afternoon exercise

https://github.com/vdumoulin/conv_arithmetic

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

Imperial College London

Which Approach to Identify Transposed Convolution Operator?

1. Identify first 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

Imperial College London

Introduction to Generative Models

1. **PCA and Autoencoders**
2. **Transposed Convolution**
3. **Variational Autoencoders**
4. **Generative Models Generalization**

Imperial College London

Which Use for Autoencoders?

W_1 W_2 W_3 W_4

784 → 1000 → 500 → 250 → 30 features

Throw away the calibrated Decoder and use the Encoder for Dimensionality Reduction!

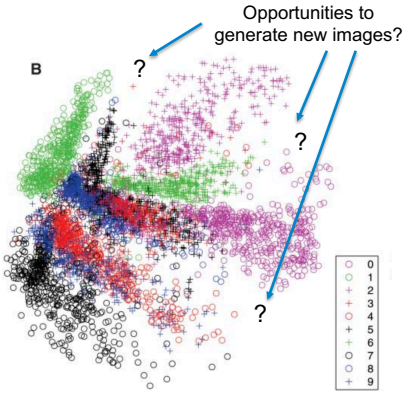
Every Image of MNIST Dataset now represented by only 30 features instead of by 784 pixel values.

Possible Application: a Supervised Learning exercise on the MNIST Dataset can use the 30 features as input features, instead of the 784 pixel values. Can avoid overfitting in case there are many input images but not many of them are labelled.

From Hinton and Salakhutdinov, Science, Science, July 2006
https://www.youtube.com/watch?v=_Ex1Ur85AVs

Imperial College London

Autoencoders can Only Encode and Decode Existing Images

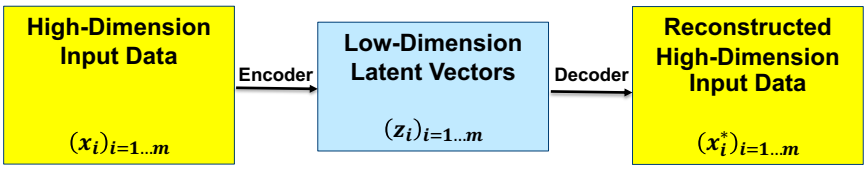


… we have not used the decoder yet. Why not make it generate more random images than just the existing ones?

https://www.youtube.com/watch?v=5WoItGTWV54 From Hinton and Salakhutdinov, Science, Science, July 2006

Imperial College London

Variational Autoencoders (VAEs): Generate New Images which were not in Training Dataset



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

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

Imperial College London

The Variational Autoencoder Model (in case of images)

Training Data

The set of images x (each image of dimension n) result from the application of a (neural network) function to a "latent vector" of much smaller dimension p than n . This latent vector is assumed to follow a standardized MultiGaussian distribution $N(z; 0, I_{p\times p})$.

How can we derive the parameters of this neural network?

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

Imperial College London

How Variational Autoencoders Work

The Encoding Step

Input Image x

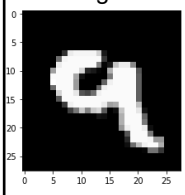


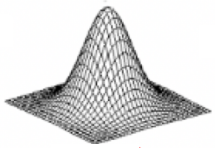
Image is of dimension n

Neural Network

$\mu(x)$

$\sigma^2(x)$

$\mu(x)$ and $\sigma^2(x)$ of dimension p



pdf of Latent Vector z

$\mathrm{N}(z; \mu(x), \sigma^2(x) I_{p\times p}))$

z is of dimension p

z

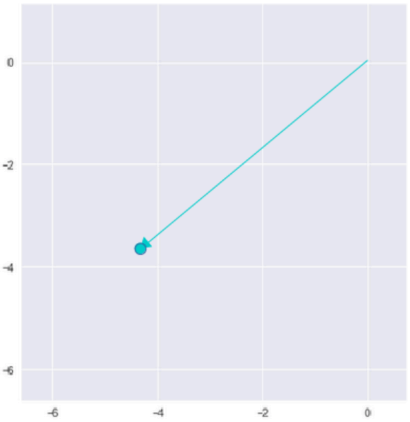
Value of z sampled by Monte-Carlo

z is of dimension p

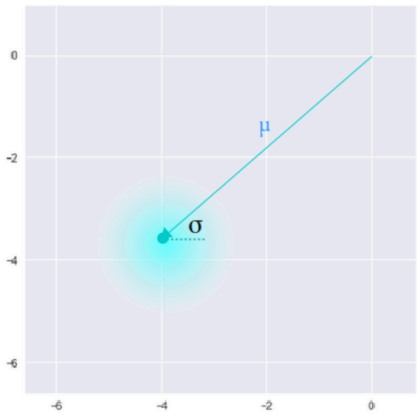
https://www.kaggle.com/rvislaywade/visualizing-mnist-using-a-variational-autoencoder#

Imperial College London

From Deterministic to Probabilistic Latent Coordinates



Basic Autoencoder (deterministic encoding coordinates z)



Variational Autoencoder ($\mu(x)$ and $\sigma(x)$ now define pdf $\mathrm{N}(z; \mu(x), \sigma^2(x) I_{p\times p}))$

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

Imperial College London

How Variational Autoencoders Work

The Decoding Step

Latent Vector z

z

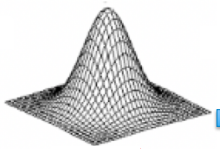
z is of dimension p

Neural Network

$\mu(z)$

$\sigma^2(z)$

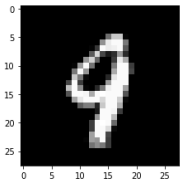
$\mu(z)$ and $\sigma^2(z)$ of dimension n



pdf of Generated Image x

$\mathrm{N}(x; \mu(z), \sigma^2(z) I_{n \times n}))$

x is of dimension n



Value of x sampled by Monte-Carlo

https://www.kaggle.com/rvislaywade/visualizing-mnist-using-a-variational-autoencoder#

Imperial College London

Some Important Definitions

The encoder network is called the ***recognition*** or ***inferenc***e network.

The decoder network is called the ***generation*** network.

Imperial College London

How Variational Autoencoders Work

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

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

$\mu(x)$ 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 $\phi's$ of $q_\phi(z/x)$.
$\mu(z)$ 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 $\theta's$ of $p_\theta(x/z)$.

These parameters must be calibrated using the m data $x^{(i)}$ by maximization of a likelihood function.

Imperial College London

How Variational Autoencoders Work (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

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

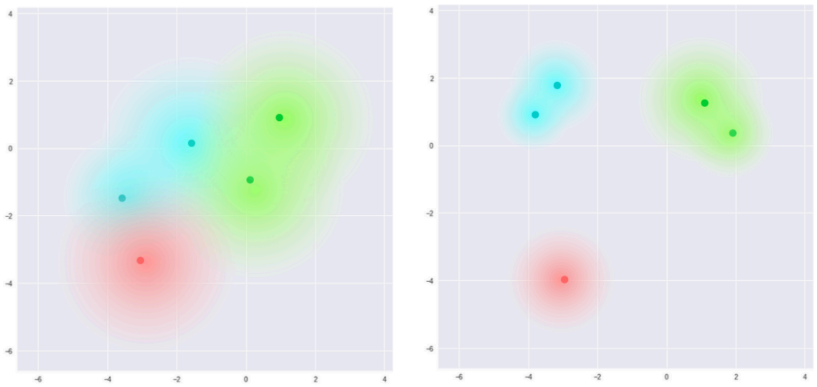
The right-hand side is called the Evidence Lower Bound (ELBO).
The ELBO is negative, as the sum of two negative terms.

The first term of the ELBO is a **reconstruction likelihood**, quantifying the quality of the « round trip » between the training image $x^{(i)}$ and its associated latent vector pdf. The second one is a **KL divergence** that aims to « spread » the range of latent vectors associated to the image $x^{(i)}$.

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

Imperial College London

If Only First (Likelihood) Term of Objective Function is Used

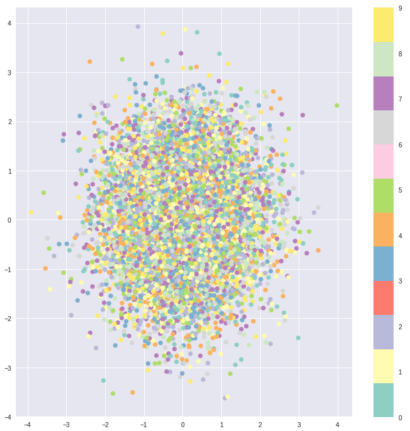


We want a continuous latent space……..……But the Likelihood Term tends to create clusters

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

Imperial College London

If Only KL Divergence 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

Imperial College London

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

Imperial College London

How Variational Autoencoders Work (Kingma & Welling, 2014)

Calculating $-\mathbf{KL}(\boldsymbol{q_\phi}(\boldsymbol{z}/\boldsymbol{x^{(i)}})\|\boldsymbol{q}(\boldsymbol{z}))$

This is the KL divergence between two Gaussian distributions, each of dimension p:

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

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

Using the formula giving the KL divergence of two multivariate Gaussians we get

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

Where p is the dimension of the latent vector

Imperial College London

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:

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

In our case the first distribution is that of the conditional Gaussian of mean $\mu = \begin{pmatrix}\mu_1 \\ \vdots \\ \mu_p\end{pmatrix}$ and diagonal 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 the standardized multivariate normal: $\mu_2 = 0$ and $\Sigma_2 = I_{p\times p}$. Thus, in our case:

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

Imperial College London

How Variational Autoencoders Work (Kingma & Welling, 2014)

Calculating $E_{z\sim q_\phi(z/x^{(i)})}\left(log\, p_\theta\left(x^{(i)}/z\right)\right)$ 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 space of associated latent vectors z. This is an expectation, which can be calculated by Monte Carlo sampling and averaging over a large set of L simulated values of z.

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

In order to express this as a function of parameters $\mu_i = \mu(x_i)$ and $\sigma_i^2 = \sigma^2(x_i)$, we write:

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

Imperial College London

Likelihood of Uncorrelated Multivariate 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: $f(x) = (2\pi)^{-\frac{n}{2}}\frac{1}{\prod_{j=1}^{j=n}\sigma_j}\exp\left[-\frac{1}{2}\sum_{i=1}^{n}\left(\frac{x_j-\mu_j}{\sigma_j}\right)^2\right]$

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

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

Imperial College London

Summarizing how we calculate a VAE

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

$$\log p\left(x^{(i)}\right) \geq \boldsymbol{E_{q(z)}\left(\log p_\theta\left(x^{(i)}/z\right)\right) - \mathrm{KL}\left(q_\phi\left(z/x^{(i)}\right)\|q(z)\right)}$$

$E_{q(z)}\left(\log p_\theta\left(x^{(i)}/z\right)\right)$ *is a function of the vectors* $\mu(z_i), \sigma^2(z_i), \mu(x_i), \sigma^2(x_i)$

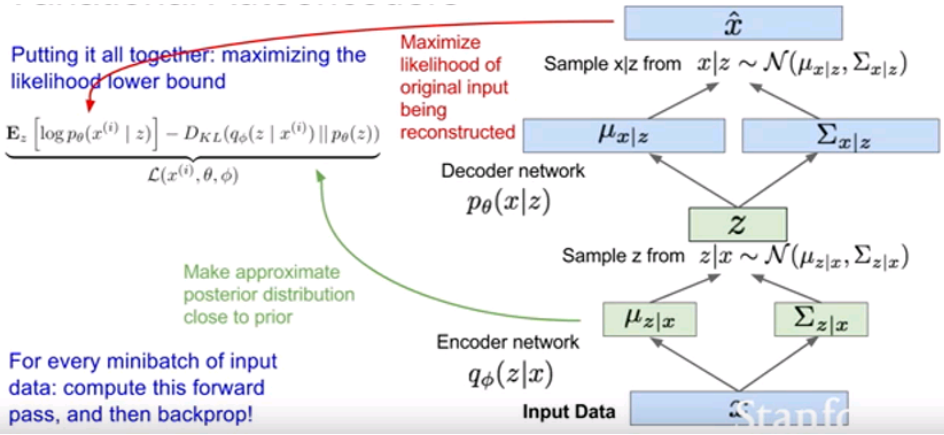
$-KL\left(q_\phi\left(z/x^{(i)}\right)\|q(z)\right)$ *is a function of the vectors* $\mu(x_i), \sigma^2(x_i)$

The ELBO, sum of these two terms, is thus a function of $\mu(z_i), \sigma^2(z_i), \mu(x_i), \sigma^2(x_i)$.

The parameters ϕ *and* θ *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 *are obtained by maximizing the ELBO on the training set* $(x_i)_{i=1\ldots m}$.

Imperial College London

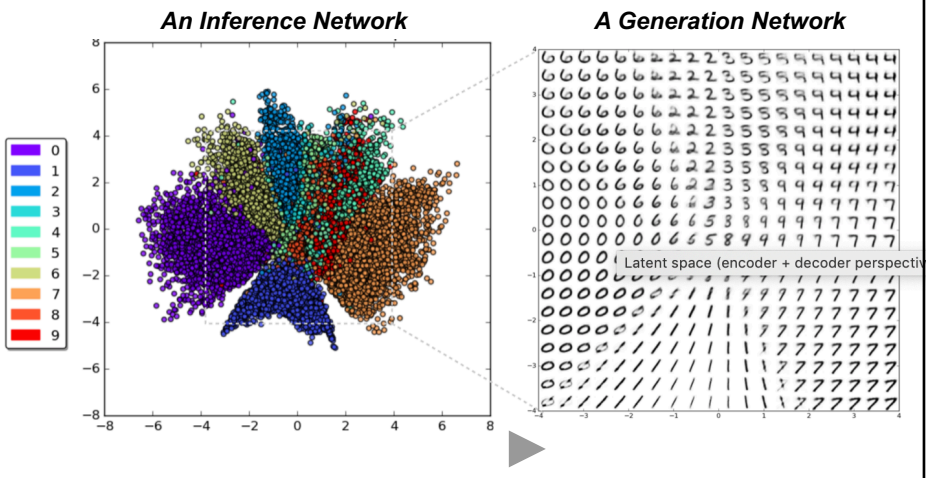
A Synthesis of how Variational Autoencoders Work.



https://www.youtube.com/watch?v=5WoItGTWV54

Imperial College London

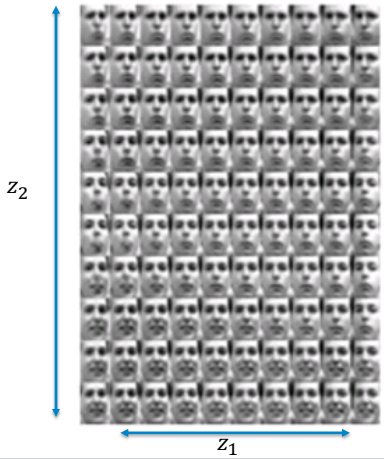
Variational Autoencoders: Two Results – MNIST Example



https://blog.fastforwardlabs.com/2016/08/12/introducing-variational-autoencoders-in-prose-and.html

Imperial College London

Interpreting the Latent Vector Features or Coordinates



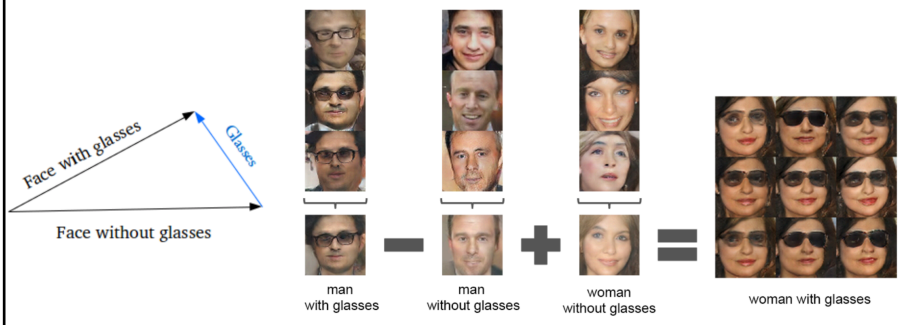
z_1: Head Pose

z_2: Degree of Smile

Kingma and Welling, 2014

Imperial College London

Arithmetic Operations on Latent Vectors



Arithmetic operations on images (from willwhitney.github.io/gan-article/)

Imperial College London

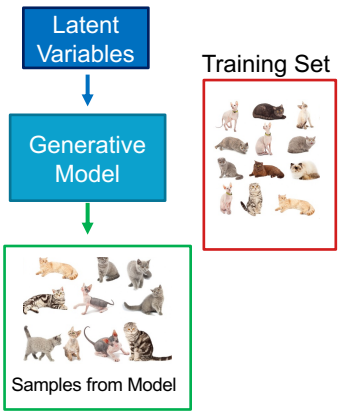
Introduction to Generative Models

1. **PCA and Autoencoders**
2. **Transposed Convolution**
3. **Variational Autoencoders**
4. **Generative Models Generalization**

Imperial College London

(Deep) Generative Methods

- Task: Draw samples from unknown pdf given a set of samples
- Variational Autoencoders (VAE)
- Generative Adversarial Networks (GAN)
  - Many more ...



Imperial College London

Why Generative Models?

Generic Synthetic Images, Music, Text….

Training Data

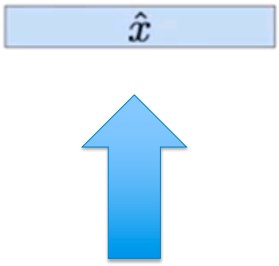
Samples from Model

Samples from $p_{data}(x)$ Samples from $p_{model}(x)$

Goal: learn $p_{model}(x)$ from $p_{data}(x)$ in order to generate more samples "looking like" the data

Imperial College London

VAEs: Decoder for Generating Samples



Sample z from $z \sim \mathcal{N}(0, I)$

https://www.youtube.com/watch?v=5WoItGTWV54

Imperial College London

Objective of Generative Networks Based on Latent Vectors

Apply a transformation G to random latent vector z (usually a multivariate Gaussian $N(0, I)$), such that the pdf $p_{model}(x)$ of $x = G(z)$ is close to $p_{data}(x)$

Will need tools to compare pdfs: KL Divergence, Maximum Mean Discrepancy, Discriminator…

Imperial College London

Compare two pdfs: the Kullback-Leibler (KL) Divergence

If the distributions $p(x)$ and $q(x)$ are continuous:

$$D_{KL}(p\|q) = \int_{-\infty}^{+\infty} p(x) \log p(x)\, dx - \int_{-\infty}^{+\infty} p(x) \log q(x)\, dx$$

Two Fundamental Properties

- $D_{KL}(p\|q)$ is positive, and 0 if $p(x)$ and $q(x)$ identical
- Asymmetry: $D_{KL}(p\|q) \neq D_{KL}(q\|p)$

Imperial College London

Compare two pdfs: the MMD

Consider two sample datasets:

$X = (x^{(1)}, x^{(2)}, \ldots\ldots, x^{(m)})$ and $X' = (x'^{(1)}, x'^{(2)}, \ldots\ldots, x'^{(m')})$

The maximum mean discrepancy (MMD) compares the two samples by comparing the sample means in a so-called feature space:

$$MMD(X, X') = \left\| \sum_{i=1}^{m} \phi(x^{(i)}) - \sum_{i'=1}^{m'} \phi(x'^{(i')}) \right\|^2$$

which can be written as a function of a kernel function

$$MMD(X, X') = \frac{1}{m^2} \sum_{i=1}^{m} \sum_{j=1}^{m} k(x^{(i)}, x^{(j)}) + \frac{1}{m'^2} \sum_{i'=1}^{m'} \sum_{j'=1}^{m'} k(x'^{(i')}, x'^{(j')}) - \frac{2}{mm'} \sum_{i=1}^{i=m} \sum_{j'=1}^{j'=m'} k(x^{(i)}, x'^{(j')})$$

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 closest $D(x)$ is to 1, the more likely is x to be also a sample of $p_{data}(x)$.
Both $D(x)$ and $G(z)$ are neural networks.

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
- Introduction to other Generative Neural Networks and the general issue of approximating one pdf $p_{data}(x)$ by another pdf $p_{model}(x)$