NPTEL ONLINE CERTIFICATION COURSES

Course Name: Deep Learning

Faculty Name: Prof. P. K. Biswas

Department : E & ECE, IIT Kharagpur

Topic

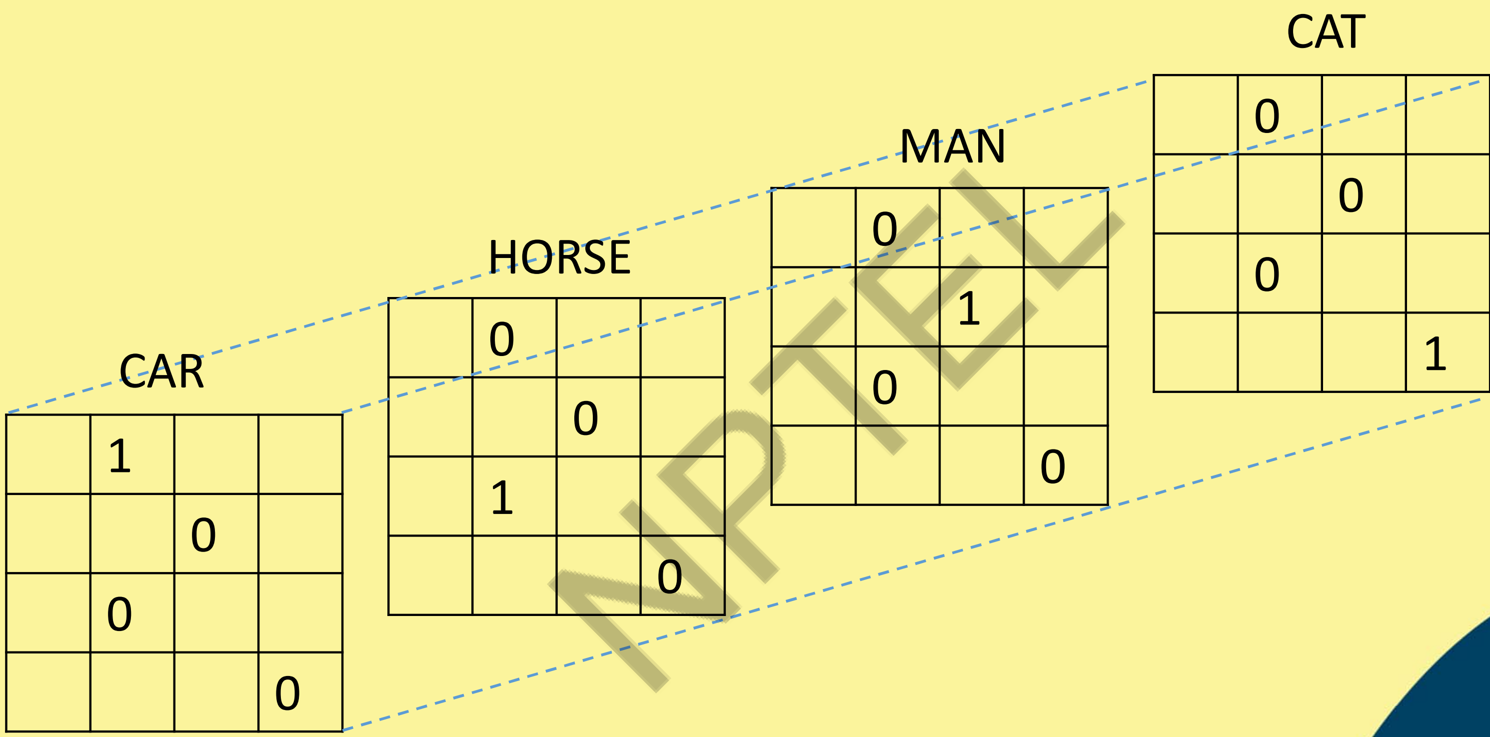
Lecture 56: Image Denoising

CONCEPTS COVERED

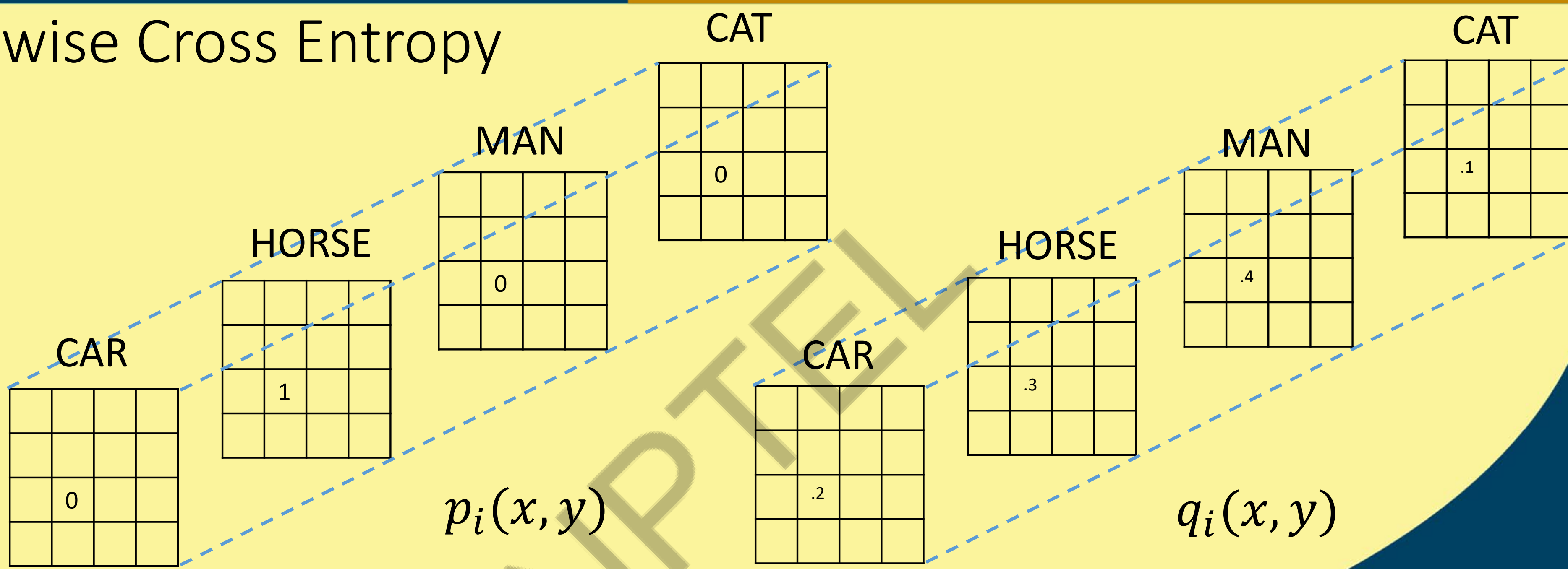
Concepts Covered:

- FCN/Deconv NN Training
- Pixelwise Entropy Loss
- Dice Loss
- Image Restoration
- Image Restoration Network
- Low dose C.T. denoising

Training for Sem Segmentation



Pixel wise Cross Entropy



$$L = -\frac{1}{N}\sum_{N}\sum_{x,y} p_i(x,y).\,log q_i(x,y)$$

Dice Loss

- Another popular loss function for image segmentation tasks is based on the Dice coefficient.
- A measure of overlap between two samples.
- This measure ranges from 0 to 1 where a Dice coefficient of 1 denotes perfect and complete overlap.

$$Dice = \frac{2|A \cap B|}{|A| + |B|}$$

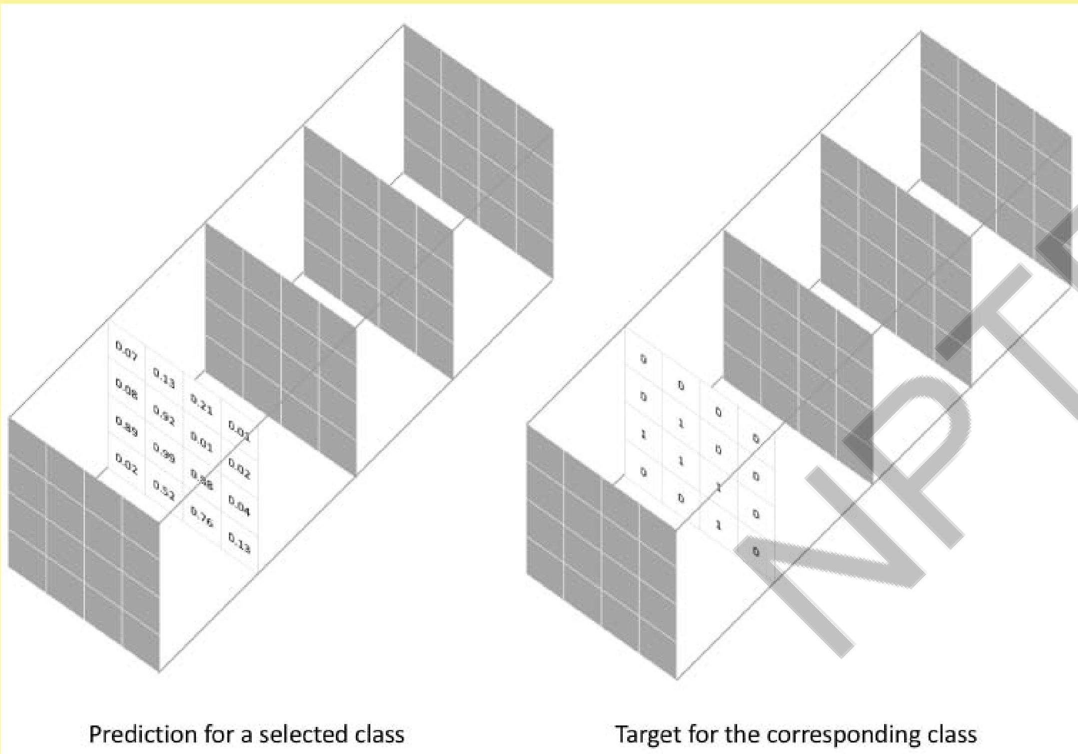
- |A∩B| represents the common elements between sets A and B
- |A| represents the number of elements in set A (and likewise for set B)
- |A∩B| is the element-wise multiplication between the prediction and target mask, and then sum the resulting matrix

Dice Loss

$$|A \cap B| = \underbrace{\begin{bmatrix} 0.01 & 0.03 & 0.02 & 0.02 \\ 0.05 & 0.12 & 0.09 & 0.07 \\ 0.89 & 0.85 & 0.88 & 0.91 \\ 0.99 & 0.97 & 0.95 & 0.97 \end{bmatrix}}_{\text{prediction}} * \underbrace{\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}}_{\text{target}} \xrightarrow{\text{element-wise multiply}} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0.89 & 0.85 & 0.88 & 0.91 \\ 0.99 & 0.97 & 0.95 & 0.97 \end{bmatrix}$$

Image Source : https://www.jeremyjordan.me/semantic-segmentation/

Dice Loss



$$L(class) = 1 - \frac{2\sum_{\forall x,y} t(x,y).p(x,y)}{\sum_{\forall x,y} t(x,y)^2 + \sum_{\forall x,y} p(x,y)^2}$$

$$L = \sum_{\forall class} L(class)$$

Image Source : https://www.jeremyjordan.me/semantic-segmentation/

Image Restoration

- A general Image degradation operation consists of a degradation operator followed by additive noise.
- Image restoration is fundamental problem in image processing research.
- There are different type of restoration process like: deblurring, denoising, super resolution, inpainting etc depending on the degradation function H.
- Image restoration becomes a problem of image denoising if degradation operator is an identity matrix.

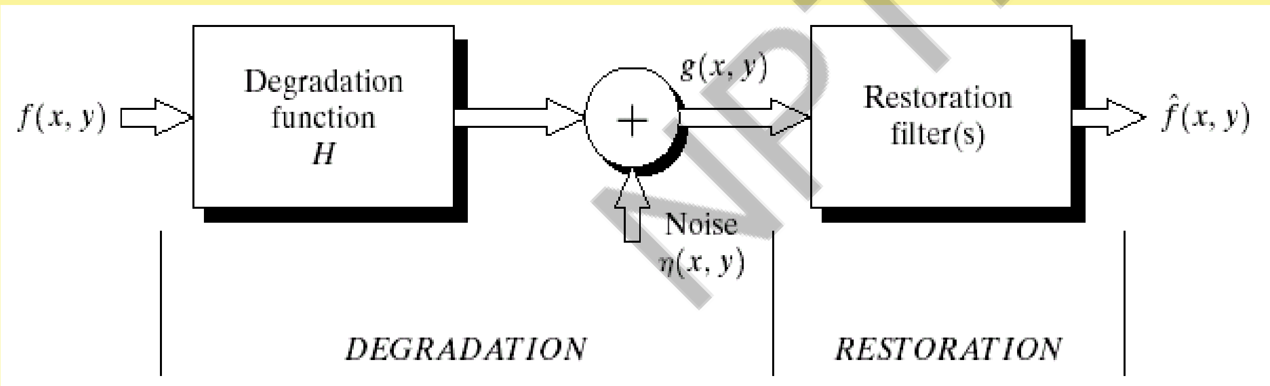
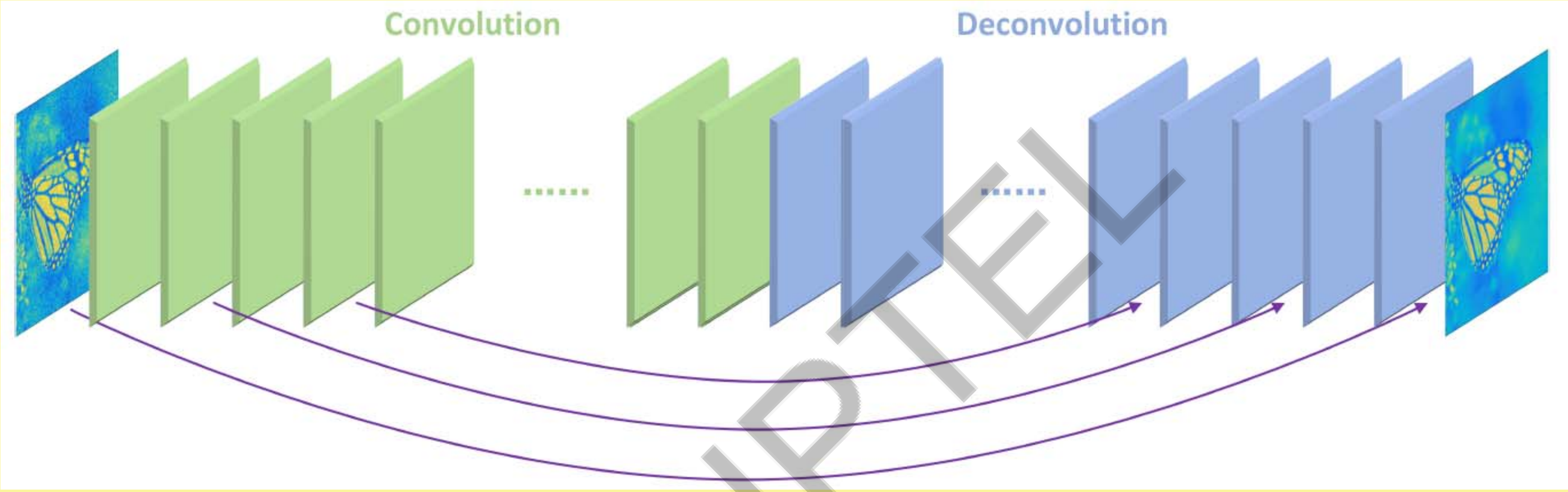


Image Denoising

Image effected with white Gaussian noise

Clean Image

Image Restoration Network



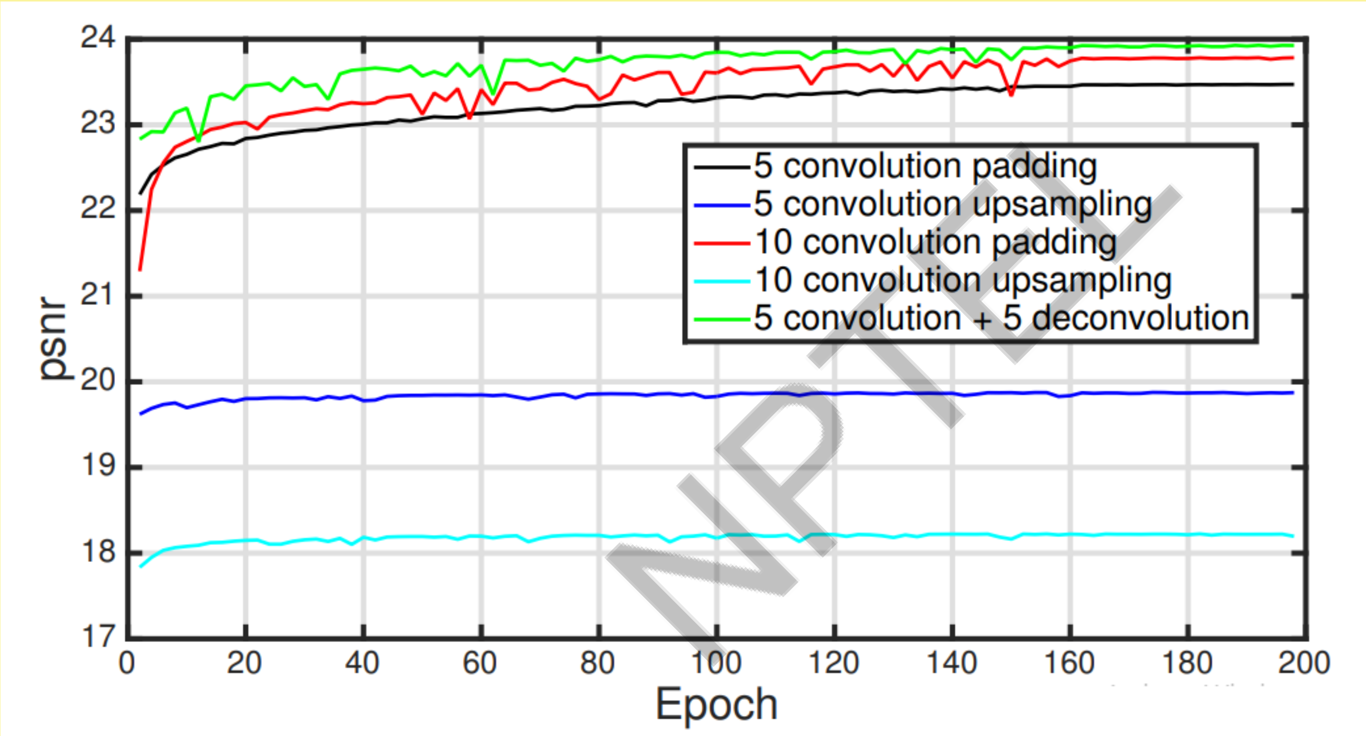
Source : Mao, Xiao-Jiao, Chunhua Shen, and Yu-Bin Yang. "Image restoration using convolutional auto-encoders with symmetric skip connections." *arXiv preprint arXiv:1606.08921* (2016).

Image Restoration Network

- The network contains layers of symmetric convolution (encoder) and deconvolution (decoder).
- Convolutional layers successively down-sample the input image content into a small size abstraction.
- Deconvolutional layers then up-sample the abstraction back into its original resolution.
- The convolutional layers act as the feature extractor, which capture the abstraction of image contents while eliminating noises/corruptions
- The deconvolutional layers are then combined to recover the details of image contents.
- Deconvolutional layers associate a single input activation with multiple outputs.
- Deconvolution is usually used as learnable up-sampling layers.

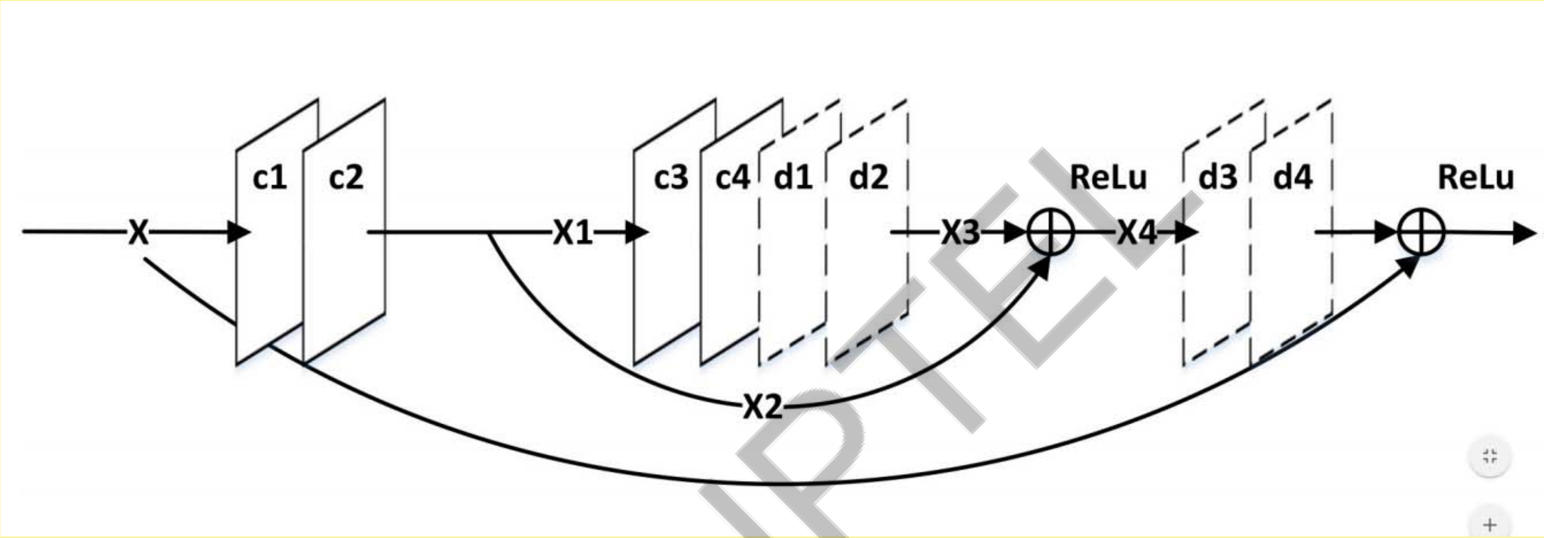
Source : Mao, Xiao-Jiao, Chunhua Shen, and Yu-Bin Yang. "Image restoration using convolutional auto-encoders with symmetric skip connections." *arXiv preprint arXiv:1606.08921* (2016).

Comparison with Fully Convolutional Network



Source : Mao, Xiao-Jiao, Chunhua Shen, and Yu-Bin Yang. "Image restoration using convolutional auto-encoders with symmetric skip connections." *arXiv preprint arXiv:1606.08921* (2016).

Image Restoration Network



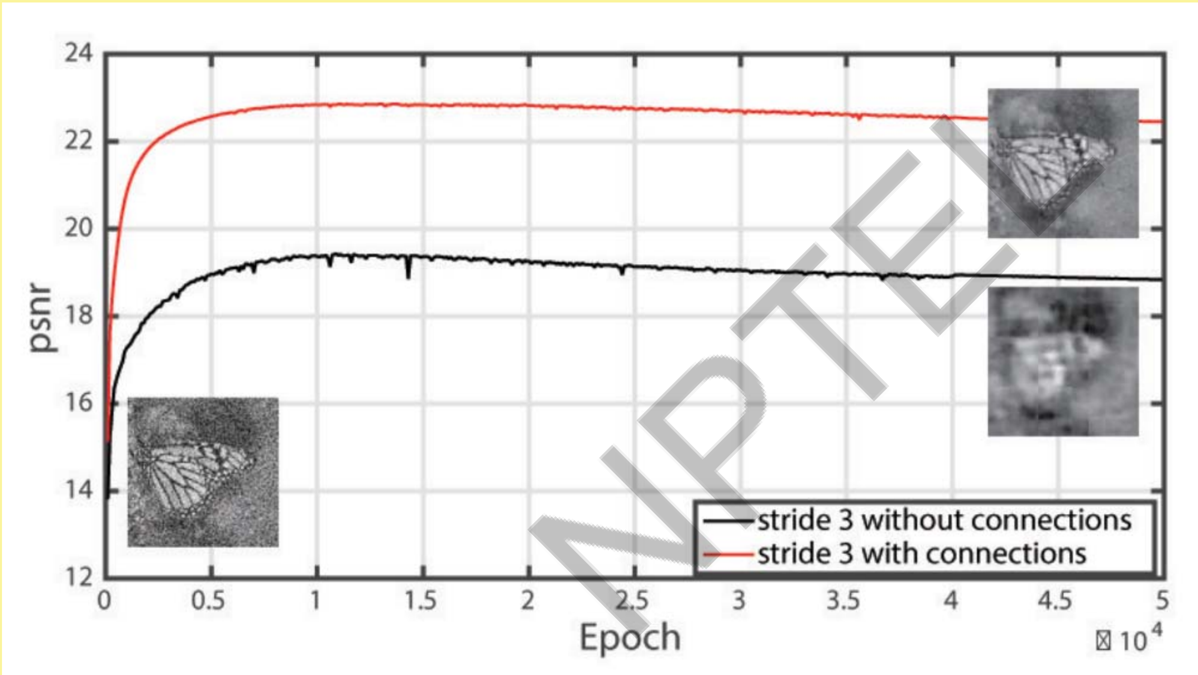
Source : Mao, Xiao-Jiao, Chunhua Shen, and Yu-Bin Yang. "Image restoration using convolutional auto-encoders with symmetric skip connections." *arXiv preprint arXiv:1606.08921* (2016).

Why Skip Connections?

- As the network goes deeper image details are lost, making it difficult for deconvolution recovering them.
- The feature maps passed by skip connections carry much image detail, which helps deconvolution to recover an improved clean version of the image.
- The skip connections also achieve benefits on back-propagating the gradient to bottom layers, which makes training deeper network much easier.

Source : Mao, Xiao-Jiao, Chunhua Shen, and Yu-Bin Yang. "Image restoration using convolutional auto-encoders with symmetric skip connections." *arXiv preprint arXiv:1606.08921* (2016).

Why Skip Connections?



Source : Mao, Xiao-Jiao, Chunhua Shen, and Yu-Bin Yang. "Image restoration using convolutional auto-encoders with symmetric skip connections." *arXiv preprint arXiv:1606.08921* (2016).

Training the Restoration Network

- Learning the end-to-end mapping from corrupted images to clean images needs to estimate the weights Θ represented by the convolutional and deconvolutional kernels.
- Specifically, given a collection of N training sample pairs $\{Xi\ , Y_i\}$, where X_i is a noisy image and Y_i is the clean version as the ground truth. We can minimize the following Mean Squared Error (MSE):

$$L(\Theta) = \frac{1}{N}\sum_{i=1}^{N} \|F(X_i;\Theta) - Y_i\|_F^2$$

- Traditionally, a network can learn the mapping from the corrupted image to the clean version directly.
- However, it has been reported the if the network learns for the additive corruption from the input image then the network converges fast to a minima.

Low Dose CT denoising

- X-RAY computed tomography (CT) has been widely utilized in clinical, industrial and other applications.
- Due to the increasing use of medical CT, concerns have been expressed on the overall radiation dose to a patient.
- We can lower the radiation dose of a CT image by lowering the operating current, or shortening the exposure time.
- This type of lower dose CT image is known as Low dose CT images.
- However doing so results in distorting the image.
- A example of low dose CT image distorted with photon noise is given.

Image Source:
https://www.aapm.org/GrandChallenge/LowDoseCT/

Low Dose CT Denoising

Low dose CT image

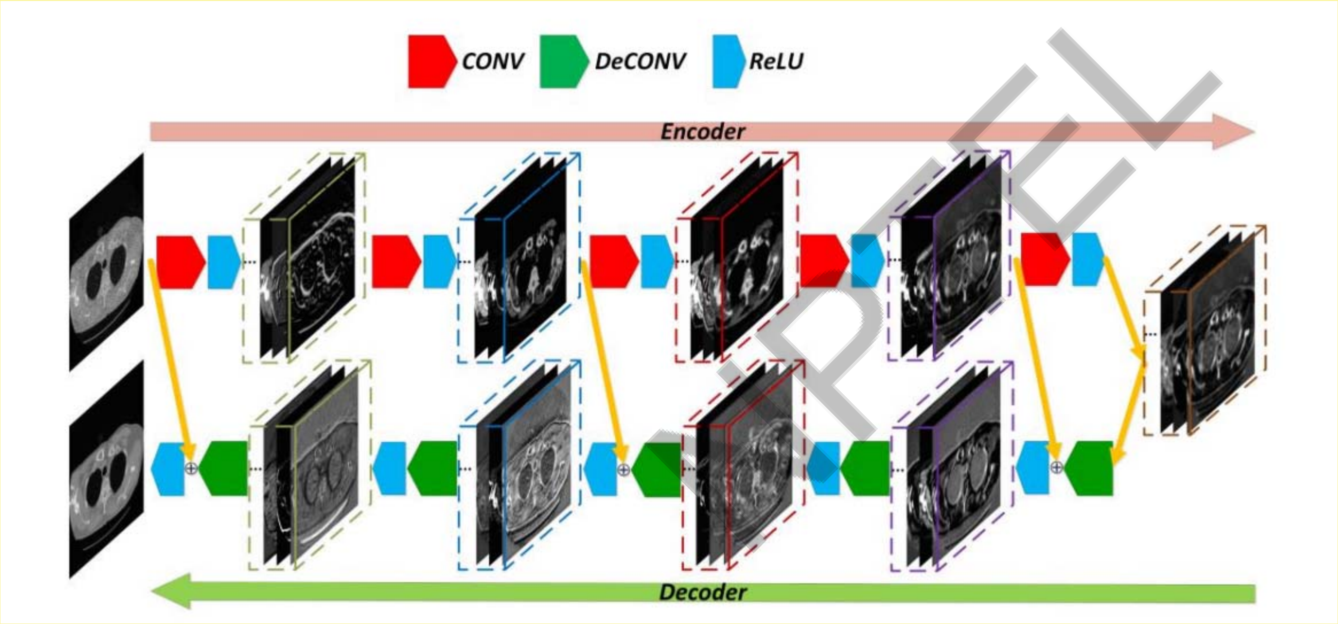
Normal dose CT image

- ❑ Due to presence of noise low dose CT images some time loose their diagnosis value
- ❑ Many important nodules are no more visible in Low dose CT image.

Image Source:
https://www.aapm.org/GrandChallenge/LowDoseCT/

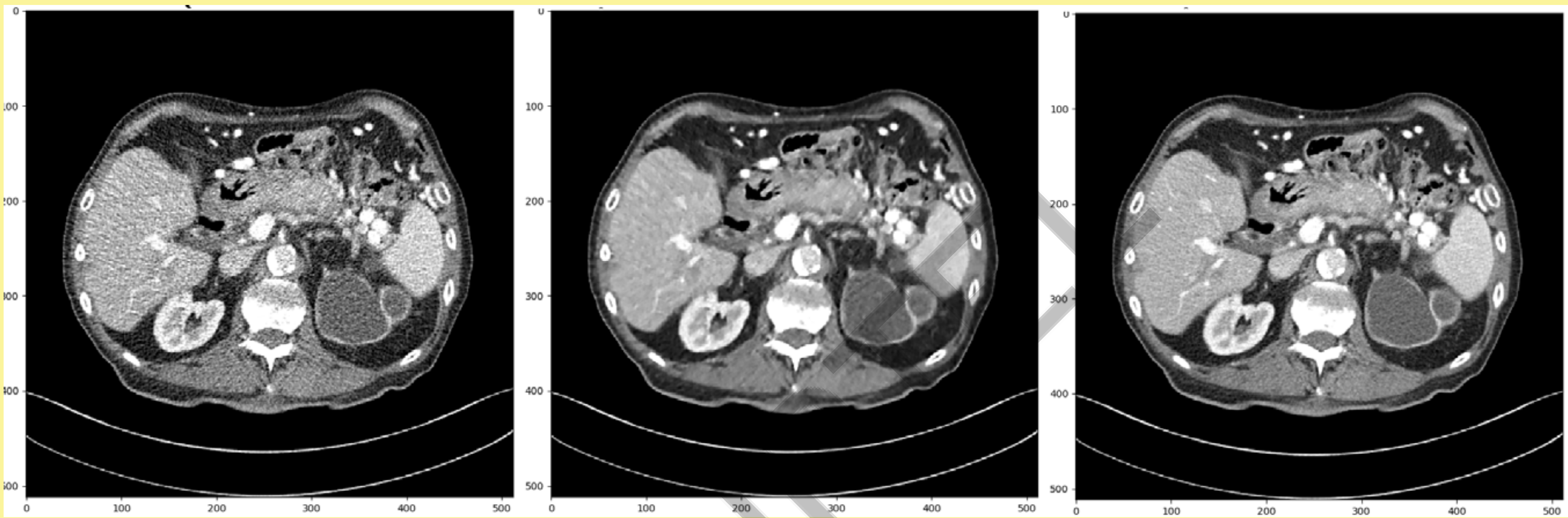
Low Dose CT Denoising

Deep Learning network can be applied to solve this real life crucial problem. A network with architecture of previous network can effectively remove noise from this low dose CT images and can recover the visibility.



Source : Chen, Hu, Yi Zhang, Mannudeep K. Kalra, Feng Lin, Yang Chen, Peixi Liao, Jiliu Zhou, and Ge Wang. "Low-dose CT with a residual encoder-decoder convolutional neural network." *IEEE transactions on medical imaging* 36, no. 12 (2017): 2524-2535.

Low Dose CT Denoising



Low Dose | Restored | Normal Dose

NPTEL ONLINE CERTIFICATION COURSES

Thank you

NPTEL ONLINE CERTIFICATION COURSES

Course Name: Deep Learning

Faculty Name: Prof. P. K. Biswas

Department : E & ECE, IIT Kharagpur

Topic

Lecture 57: Variational Autoencoder

CONCEPTS COVERED

Concepts Covered

- Generative Model
- Limitations of usual auto-encoder
- Intuitions behind VAE
- Variational Inference
- Practical Realization of VAE

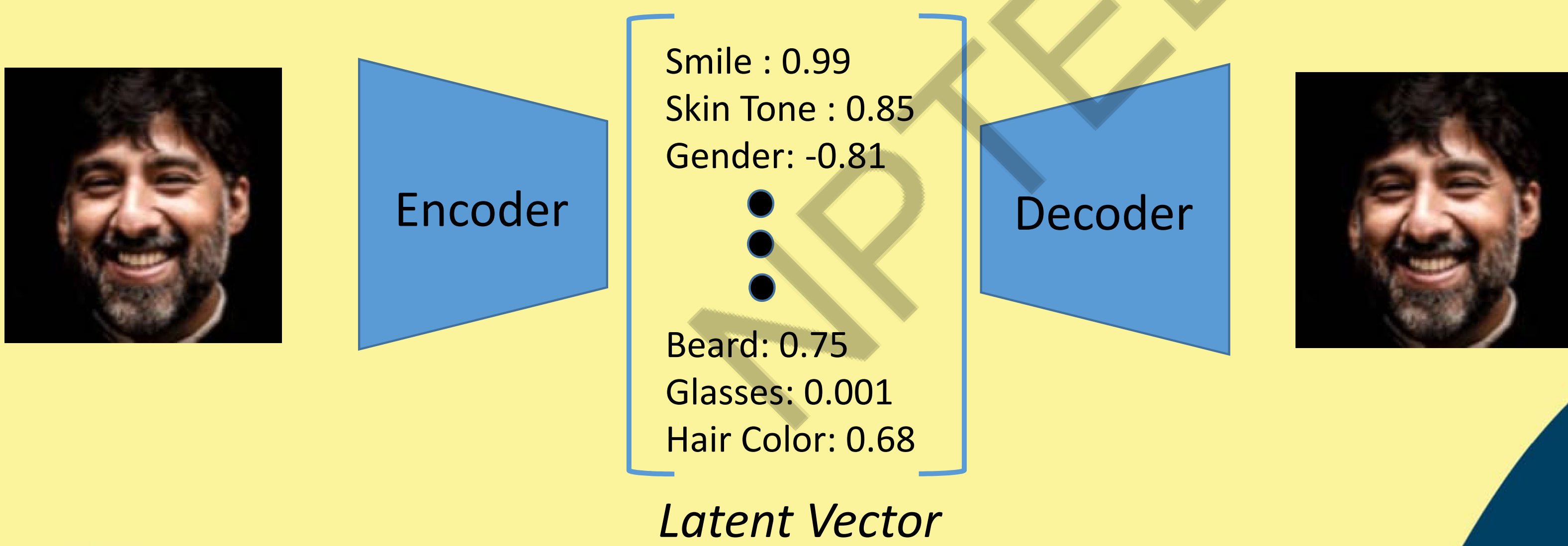
Generative Model

- Big Animal.
- Has four legs.
- Big ears.
- Long trunk.
- A pair of tusks
- …..

Latent Variables

Traditional Autoencoder

- Maps an input image via an encoder to a deterministic latent code
- Decoder maps the latent code to reconstruct the input image

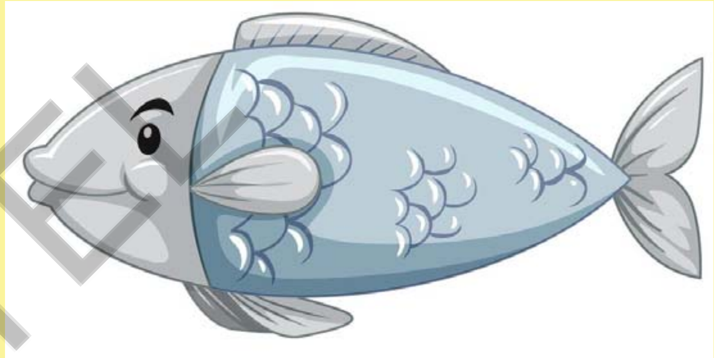


https://www.jeremyjordan.me/variational-autoencoders/

Traditional Autoencoder : Limitations

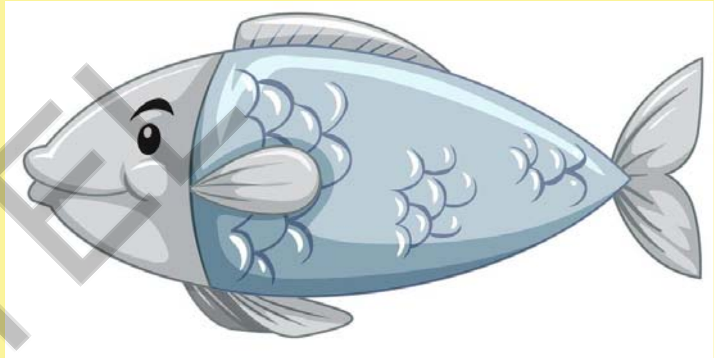
- In pursuit of compact representations, auto-encoders tends to create a latent space which is not continuous
- As a generative model, we need a latent space from which we can smoothly sample and yet get realistic reconstructions
- Auto-encoders do not allow such easy interpolations in latent space

Traditional Autoencoder : Limitations

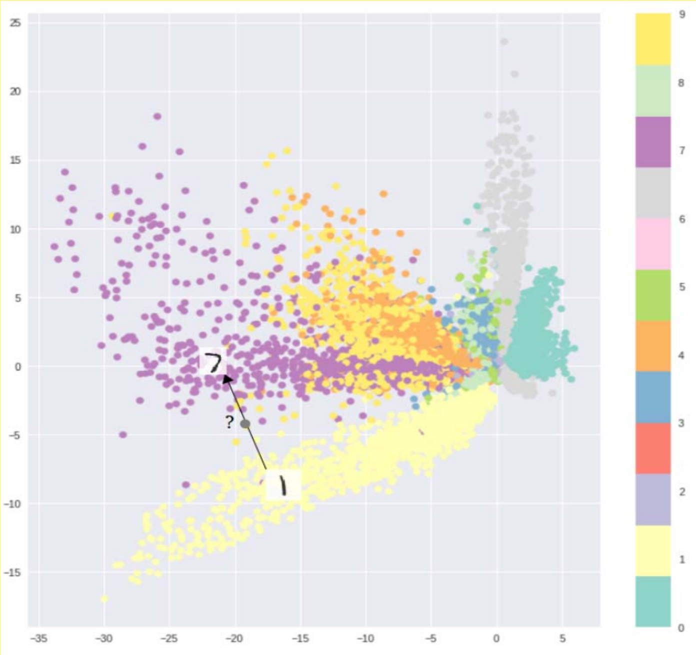


?

Traditional Autoencoder : Limitations



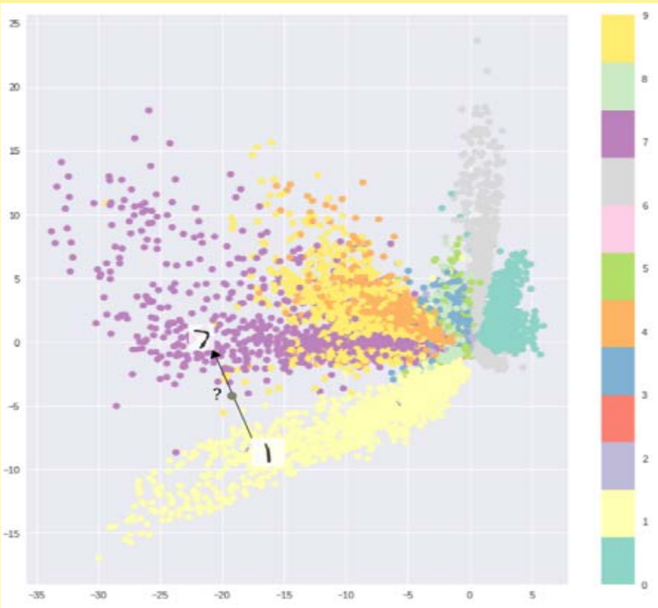
Traditional Autoencoder : Limitations



- Distinct cluster for each class
- Not easy for decoder to reconstruct since we need different distinct codes for each image

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

Traditional Autoencoder : Limitations



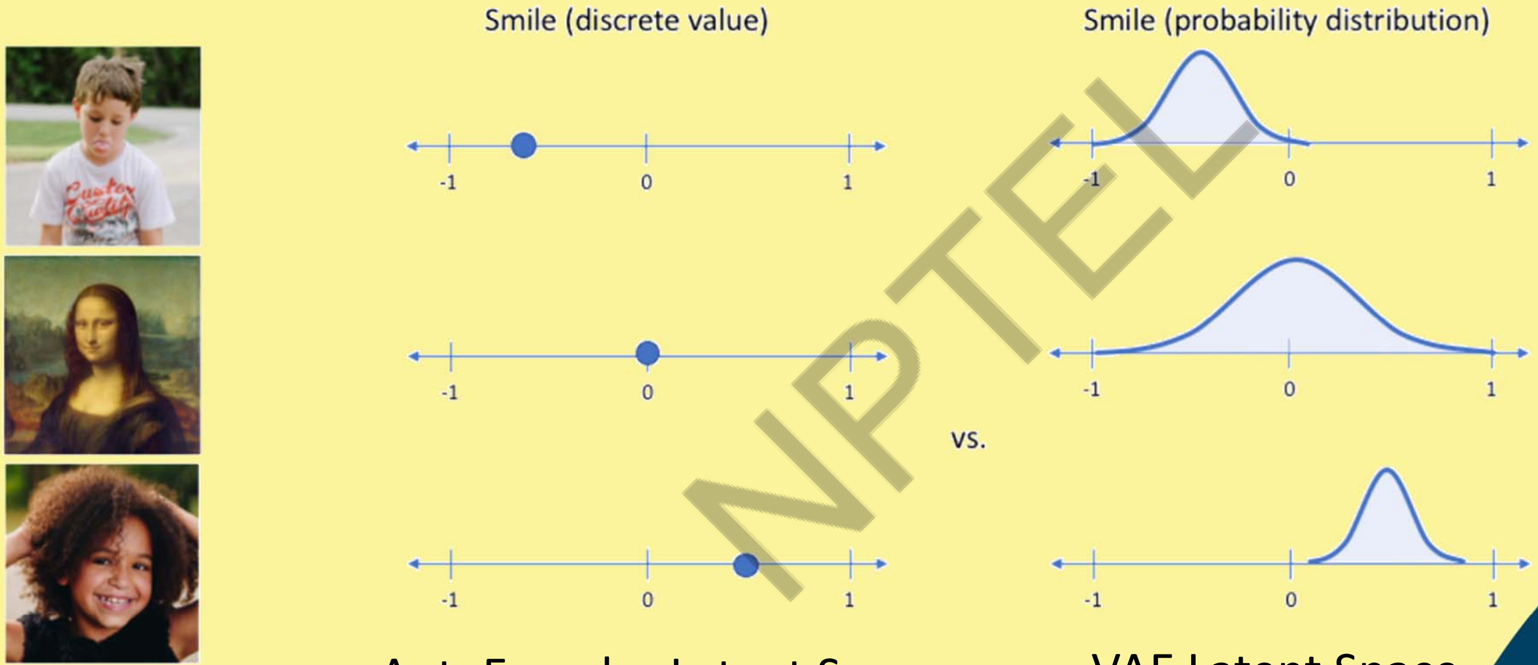
- Discontinuous latent space means decoder never reconstructed from such unexplored points
- If we sample from such points, decoder will give unrealistic output
- **Aim:** Try to make latent space continuous yet maintain the class specific compactness

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

Variational Autoencoder Intuition

- Instead of deterministic latent code we might be interested to learn a distribution over the latent code
- For example, it is more intuitive to determine a range of “smile” value for a face instead of an absolute “smile” value
- Instead of deterministic code, we will now output the mean and standard deviation of each component of the vector (assuming each component is independent of each other)

Autoencoder Intuition vs. VAE Latent Space



AutoEncoder Latent Space

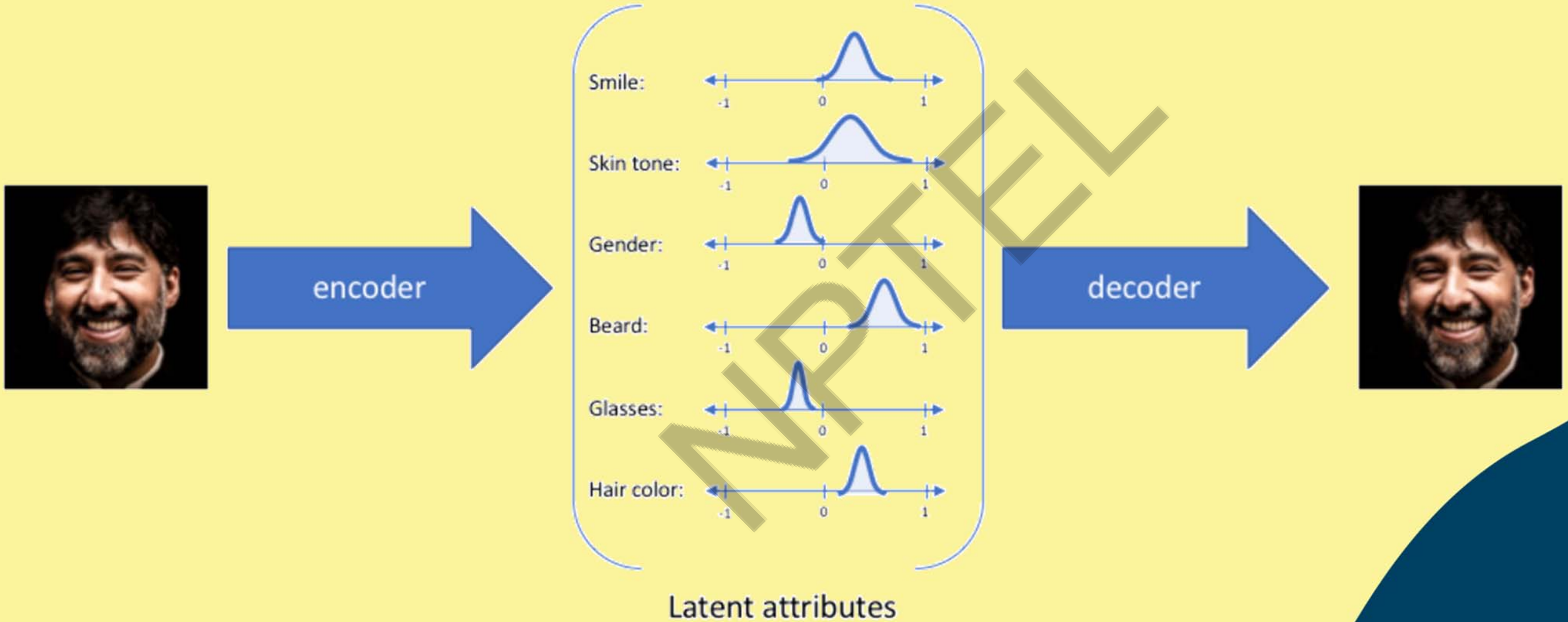
VAE Latent Space

https://www.jeremyjordan.me/variational-autoencoders/

Variational Autoencoder Intuition

- With this setup we can represent each latent factor as a probability distribution
- We can sample from such distribution
- Then the sampled vector can be passed through Decoder (Generator) to generate an image

Variational Autoencoder Intuition



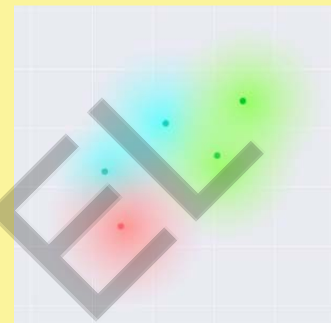
https://www.jeremyjordan.me/variational-autoencoders/

Variational Autoencoder Intuition

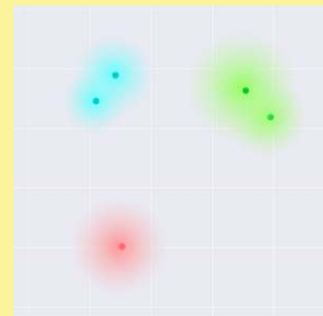
- Mean vector controls where the encoding of an input should be centered around
- Standard deviation controls the "area", how much from the mean the encoding can vary
- As encodings are generated at random from inside a hyper-sphere (distribution) decoder learns that not only is a single point in latent space referring to a sample of that class, but all nearby points refer to the same as well

Variational Autoencoder Intuition

- For smooth interpolations, ideally, we want overlap between samples that are not very similar too, in order to interpolate between classes.
- However μ and σ can take any value and learn to cluster the mean vectors of different classes far apart (and minimize σ) to reduce uncertainty for the Decoder



Our goal



Network might converge to

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

Variational Autoencoder Intuition

- In order to enforce smooth transition we will apply Kullback–Leibler divergence (KL divergence) between the distribution of encoded vectors and a prior distribution asserted on latent distribution space
- KL divergence between two probability distributions simply measures how much they diverge from each other.
- Minimizing the KL divergence here means optimizing the probability distribution parameters (μ and σ) to closely resemble that of the target distribution.

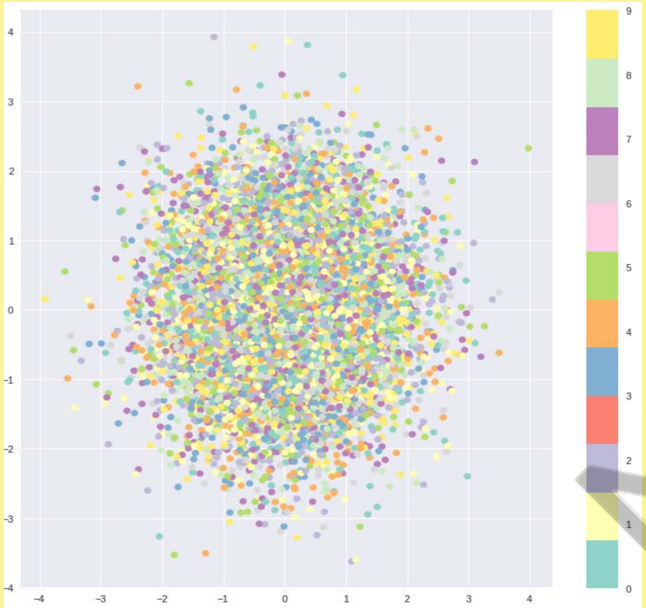
Variational Autoencoder Intuition

- In VAE, it is usually assumed that the distribution of the latent space follows a zero mean Normal distribution with diagonal covariance matrix (each component is independent of the other)
- KL divergence loss will encourage encodings from different inputs to be clustered about the center of the latent space
- If network creates clusters in specific regions then KL divergence loss will penalize such clusters formation

Variational Autoencoder Intuition

- But, only KL loss results in a latent space encodings densely placed randomly, near the center of the target distribution, with little regard for similarity/dis-similarity of input samples.
- The decoder finds it impossible to decode anything meaningful from this space, simply because there really isn't any structure/context specific meaning.

Variational Autoencoder Intuition



Latent space after training on MNIST when only optimized with KL loss

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

Variational Autoencoder Intuition

- Optimizing reconstruction loss + KL divergence loss results in the generation of a latent space which maintains the similarity of nearby encodings on the local scale via clustering
- Yet globally, is very densely packed near the latent space origin

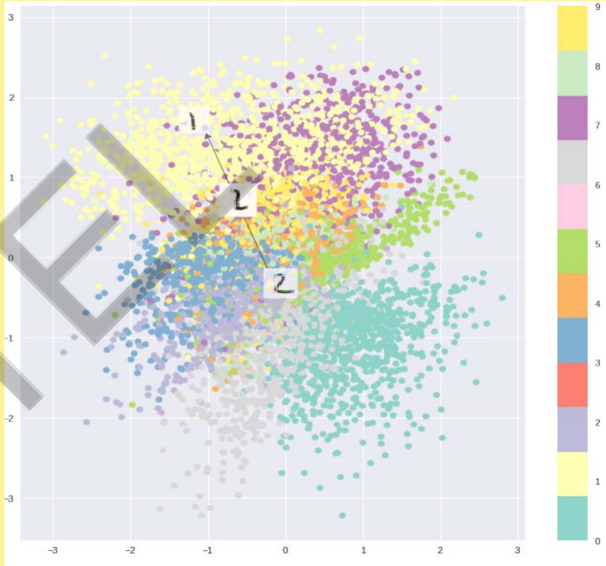
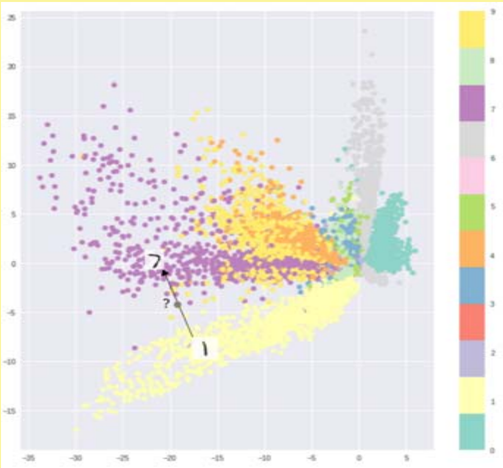
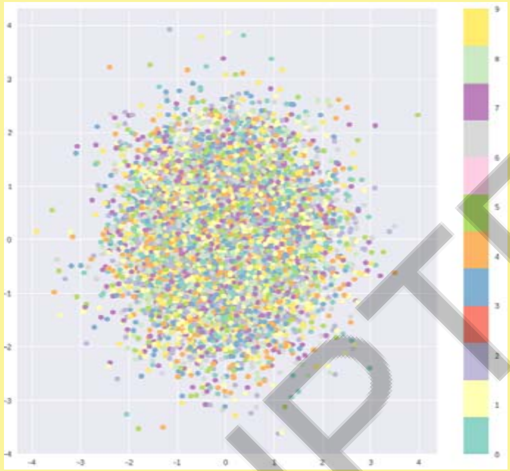


Image Source: https://towardsdatascience.com/intuitively-understanding-variational-autoencoders-1bfe67eb5daf

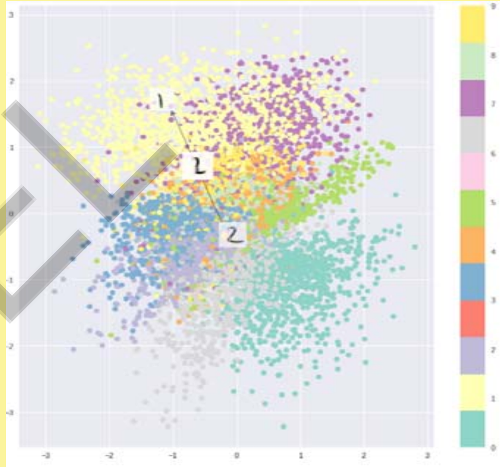
Variational Autoencoder Intuition



Reconstruction Loss



KL Divergence Loss



KL Divergence +
Reconstruction Loss

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

NPTEL ONLINE CERTIFICATION COURSES

Thank you

NPTEL ONLINE CERTIFICATION COURSES

Course Name: Deep Learning

Faculty Name: Prof. P. K. Biswas

Department : E & ECE, IIT Kharagpur

Topic

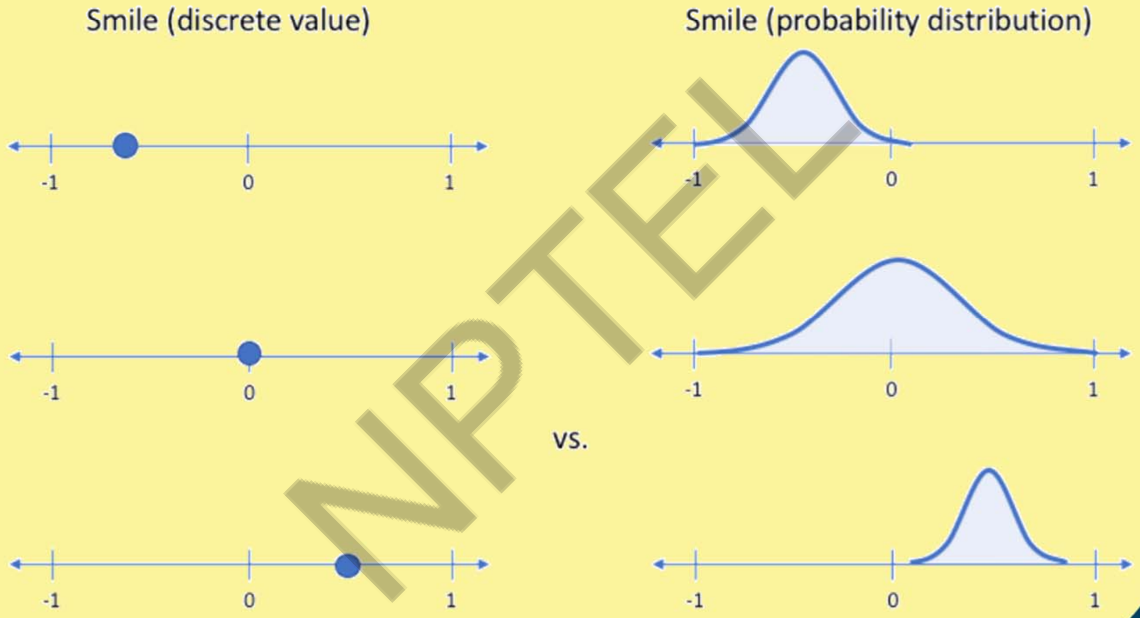
Lecture 58: Variational Autoencoder - II

CONCEPTS COVERED

Concepts Covered

- ❑ Generative Model
- ❑ Limitations of usual auto-encoder
- ❑ Intuitions behind VAE
- ❑ Variational Inference
- ❑ Practical Realization of VAE

Autoencoder Intuition vs. VAE Latent Space



AutoEncoder Latent Space

VAE Latent Space

https://www.jeremyjordan.me/variational-autoencoders/

Variational Autoencoder Intuition

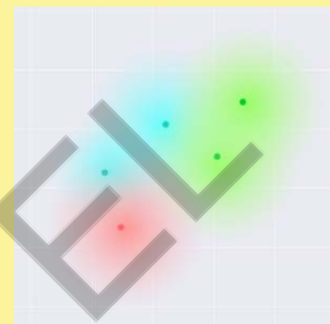
- Instead of deterministic latent code we might be interested to learn a distribution over the latent code
- For example, it is more intuitive to determine a range of “smile” value for a face instead of an absolute “smile” value
- Instead of deterministic code, we will now output the mean and standard deviation of each component of the vector (assuming each component is independent of each other)

Variational Autoencoder Intuition

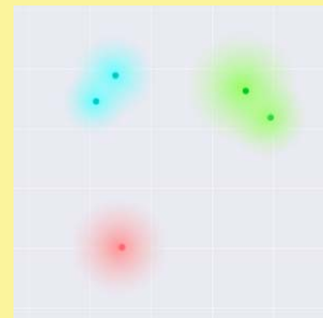
- With this setup we can represent each latent factor as a probability distribution
- We can sample from such distribution
- Then the sampled vector can be passed through Decoder (Generator) to generate an image

Variational Autoencoder Intuition

- For smooth interpolations, ideally, we want overlap between samples that are not very similar too, in order to interpolate between classes.
- However μ and σ can take any value and learn to cluster the mean vectors of different classes far apart (and minimize σ) to reduce uncertainty for the Decoder



Our goal



Network might converge to

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

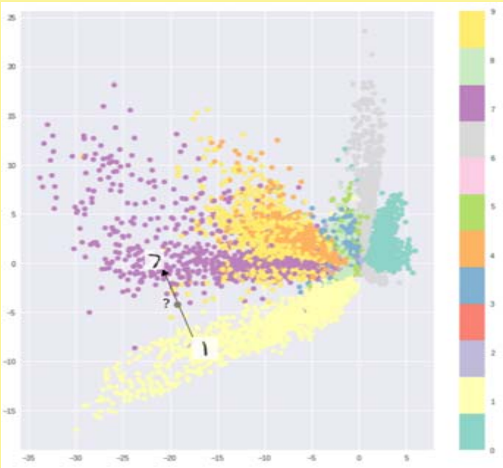
Variational Autoencoder Intuition

- In order to enforce smooth transition we will apply Kullback–Leibler divergence (KL divergence) between the distribution of encoded vectors and a prior distribution asserted on latent distribution space
- KL divergence between two probability distributions simply measures how much they diverge from each other.
- Minimizing the KL divergence here means optimizing the probability distribution parameters (μ and σ) to closely resemble that of the target distribution.

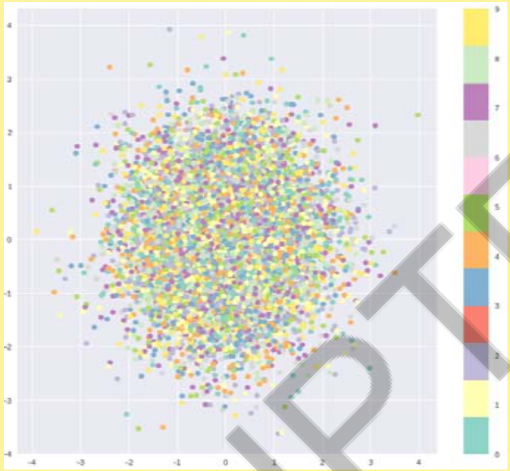
Variational Autoencoder Intuition

- In VAE, it is usually assumed that the distribution of the latent space follows a zero mean Normal distribution with diagonal covariance matrix (each component is independent of the other)
- KL divergence loss will encourage encodings from different inputs to be clustered about the center of the latent space
- If network creates clusters in specific regions then KL divergence loss will penalize such clusters formation

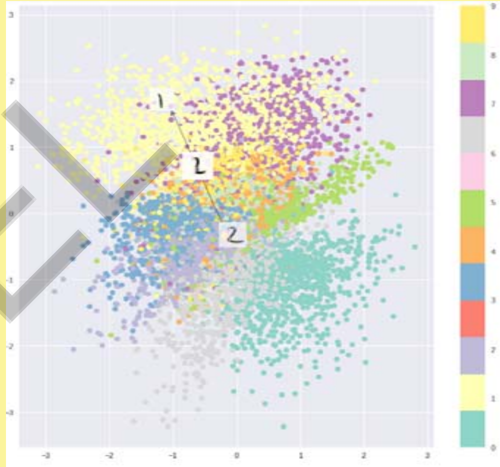
Variational Autoencoder Intuition



Reconstruction Loss



KL Divergence Loss



KL Divergence + Reconstruction Loss

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

Variational Autoencoder Intuition

- This equilibrium is attributed to cluster-forming nature of the reconstruction loss, and the dense packing nature of the KL loss
- It means when randomly generating, if you sample a vector from the prior distribution, $P(z)$ of latent space, the Decoder will successfully decode it.
- For interpolation, since there is no sudden gap between clusters, but a smooth mix of features, a Decoder can understand.

Variational Autoencoder : Variational Inference

- In VAE, we assume that there is a latent (unobserved) variable, z, generating our observed random variable, x.

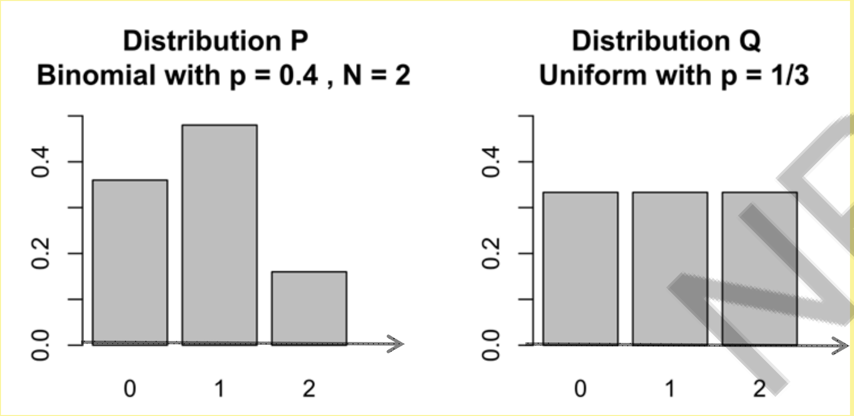
- Our aim: To compute the posterior $P(z|x) = \frac{P(X|Z)P(z)}{P(x)}$
- $P(x) = \int P(x|z)P(z)dz$ $\longrightarrow$ Intractable

Variational Autoencoder : Variational Inference

- Let's assume there is a tractable distribution Q, such that $P(z|x) \approx Q(z|x)$
- We want $Q(\cdot)$ to be in the family of tractable distributions (Gaussian for example) such that we can play around with its parameters to match P$(z|x)$
- So, we will aim towards minimizing KL divergence of P$(z|x)$ with respect to $Q(z|x)$
- Our objective: minimize KL($Q(z|x)$ || P$(z|x)$)

KL Divergence

$$\text{KL}(Q(z|x) \;||\; \text{P}(z|x)) = \sum_x Q(x) log \frac{Q(x)}{P(x)}$$



x	0	1	2
P(x)	0.36	0.48	0.16
Q(x)	0.33	0.33	0.33

KL Divergence

$$KL(P||Q) = \sum_x P(x) log \frac{P(x)}{Q(x)}$$

$$= 0.36 \log\left(\frac{0.36}{0.33}\right) + 0.48\, log\left(\frac{0.48}{0.33}\right) + 0.16\, log\left(\frac{0.16}{0.33}\right) = 0.0414$$

$$KL(Q||P) = \sum_x Q(x) log \frac{Q(x)}{P(x)}$$

$$= 0.33 \log\left(\frac{0.33}{0.36}\right) + 0.33\, log\left(\frac{0.33}{0.48}\right) + 0.33\, log\left(\frac{0.33}{0.16}\right) = 0.0375$$

x	0	1	2
P(x)	0.36	0.48	0.16
Q(x)	0.33	0.33	0.33

KL Divergence

Minimize

$$KL(Q(z|x)||P(z|x))$$

KL Divergence

$$KL(Q(z|x)||P(z|x))$$

$$= -\sum_{z} Q(z|x) \log \frac{P(z|x)}{Q(z|x)}$$

$$= -\sum_{z} Q(z|x) \log \frac{P(x,z)}{P(x) * \ Q(z|x)}$$

KL Divergence

$$= -\sum_{z} Q(z|x)\left\{log\frac{P(x,z)}{Q(z|x)} - logP(x)\right\}$$

$$= -\sum_{z} Q(z|x)\log\frac{P(x,z)}{Q(z|x)} + \sum_{z} Q(z|x)\log P(x)$$

$$= -\sum_{z} Q(z|x)\log\frac{P(x,z)}{Q(z|x)} + logP(x)$$

KL Divergence

$$KL(Q(z|x)||P(z|x)) = -\sum_z Q(z|x)\log\frac{P(x,z)}{Q(z|x)} + \ logP(x)$$

$$\downarrow$$

$$\log P(x) = KL(Q(z|x)\|P(z|x)) + \sum_z Q(z|x)\log\frac{P(x,z)}{Q(z|x)}$$

KL Divergence

$$\log P(x) = KL(Q(z|x)||P(z|x)) + \sum_z Q(z|x) \log \frac{P(x,z)}{Q(z|x)}$$

- Since, x is given, LHS is constant.
- Aim is to minimize $KL(Q(z|x)||P(z|x))$
- This is same as maximizing $\sum_z Q(z|x) \log \frac{P(x,z)}{Q(z|x)}$

NPTEL ONLINE CERTIFICATION COURSES

Thank you

NPTEL ONLINE CERTIFICATION COURSES

Course Name: Deep Learning

Faculty Name: Prof. P. K. Biswas
Department : E & ECE, IIT Kharagpur

Topic

Lecture 59: Variational Autoencoder - III

CONCEPTS COVERED

Concepts Covered

- ❑ Generative Model
- ❑ Limitations of usual auto-encoder
- ❑ Intuitions behind VAE
- ❑ Variational Inference
- ❑ Practical Realization of VAE

Variational Autoencoder : Variational Inference

- In VAE, we assume that there is a latent (unobserved) variable, z, generating our observed random variable, x.

- Our aim: To compute the posterior $P(z|x) = \frac{P(X|Z)P(z)}{P(x)}$
- $P(x) = \int P(x|z)P(z)dz \longrightarrow$ Intractable

Variational Autoencoder : Variational Inference

- Let's assume there is a tractable distribution Q, such that $P(z|x) \approx Q(z|x)$
- We want $Q(\cdot)$ to be in the family of tractable distributions (Gaussian for example) such that we can play around with its parameters to match P$(z|x)$
- So, we will aim towards minimizing KL divergence of P$(z|x)$ with respect to $Q(z|x)$
- Our objective: minimize KL($Q(z|x)$ || P$(z|x)$)

KL Divergence

Minimize

$$KL(Q(z|x)||P(z|x))$$

KL Divergence

$$KL(Q(z|x)||P(z|x)) = -\sum_z Q(z|x)\log\frac{P(x,z)}{Q(z|x)} + \; logP(x)$$

$$\log P(x) = KL(Q(z|x)||P(z|x)) + \sum_z Q(z|x)\log\frac{P(x,z)}{Q(z|x)}$$

KL Divergence

$$\log P(x) = KL(Q(z|x)||P(z|x)) + \sum_z Q(z|x) \log \frac{P(x,z)}{Q(z|x)}$$

- Since, x is given, LHS is constant.
- Aim is to minimize $KL(Q(z|x)||P(z|x))$
- This is same as maximizing $\sum_z Q(z|x) \log \frac{P(x,z)}{Q(z|x)}$

KL Divergence

$$\log P(x) = KL(Q(z|x)||P(z|x)) + \sum_z Q(z|x)\log\frac{P(x,z)}{Q(z|x)}$$

- Since, x is given, LHS is constant.
- Aim is to minimize $KL(Q(z|x)||P(z|x))$
- This is same as maximizing $\sum_z Q(z|x)\log\frac{P(x,z)}{Q(z|x)}$

Variational Lower Bound

Variational Lower Bound

$$\log P(x) = KL\big(Q(z|x)\,||\,P(z|x)\big) + \sum_{z} Q(z|x) \log \frac{P(x,z)}{Q(z|x)}$$

$$KL\big(Q(z|x)\,||\,P(z|x)\big) \geq 0$$

$$\sum_{z} Q(z|x) \log \frac{P(x,z)}{Q(z|x)} \leq logP(x)$$

Variational Autoencoder : Variational Inference

- Our initial objective: minimize KL($Q(z|x)$ || P$(z|x)$)
- Which is same as maximizing $\sum_z Q(z|x) \log \frac{P(x,z)}{Q(z|x)}$

Variational Lower Bound

- So, aim now is: *maximize*

$$L = \sum_z Q(z|x) \log \frac{P(x,z)}{Q(z|x)} = \sum_z Q(z|x) \log \frac{P(x|z)P(z)}{Q(z|x)}$$

Variational Autoencoder : Variational Inference

Maximize

Variational Autoencoder : Variational Inference

Maximize

$$\mathrm{L} = \sum_z Q(z|x) \log \frac{P(x,z)}{Q(z|x)} = \sum_z Q(z|x) \log \frac{P(x|Z)P(z)}{Q(z|x)}$$

Variational Autoencoder : Variational Inference

Maximize

$$L = \sum_z Q(z|x) \log \frac{P(x,z)}{Q(z|x)} = \sum_z Q(z|x) \log \frac{P(x|Z)P(z)}{Q(z|x)}$$

$$= \sum Q(z|x) \log P(x|z) + \sum Q(z|x) \log \frac{P(z)}{Q(z|x)}$$

Variational Autoencoder : Variational Inference

Maximize

$$L = \sum_z Q(z|x) \log \frac{P(x,z)}{Q(z|x)} = \sum_z Q(z|x) \log \frac{P(x|Z)P(z)}{Q(z|x)}$$

$$= \underbrace{\sum Q(z|x) \log P(x|z)}_{E_{Q(z|x)} \log P(x|z)} + \underbrace{\sum Q(z|x) \log \frac{P(z)}{Q(z|x)}}_{-KL(Q(z|x) \,||\, P(z))}$$

Variational Autoencoder : Variational Inference

Maximize

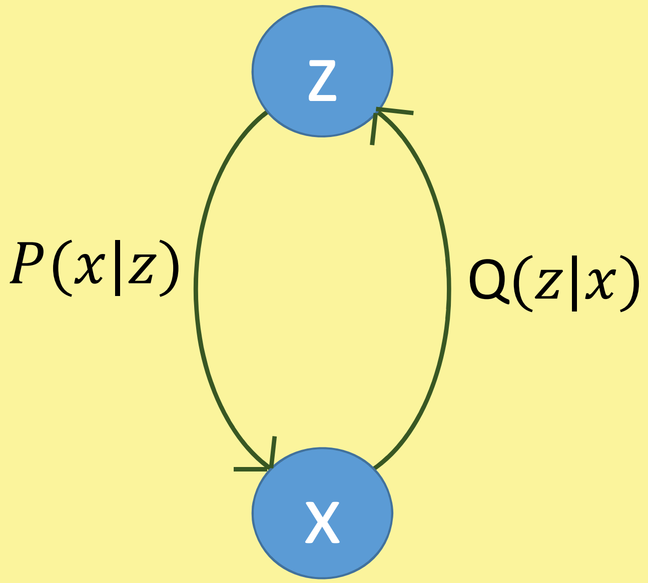
$$L = \sum_z Q(z|x) \log \frac{P(x,z)}{Q(z|x)} = \sum_z Q(z|x) \log \frac{P(x|Z)P(z)}{Q(z|x)}$$

$$= \underbrace{\sum Q(z|x) \log P(x|z)}_{E_{Q(z|x)} \log P(x|z)} + \underbrace{\sum Q(z|x) \log \frac{P(z)}{Q(z|x)}}_{-KL(Q(z|x) \,||\, P(z))}$$

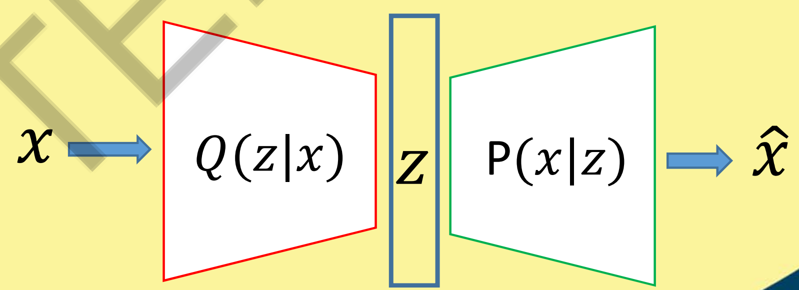
- Translate the loss functions into an auto-encoder architecture.

Variational Autoencoder : Network Realization

- We have the following graphical model



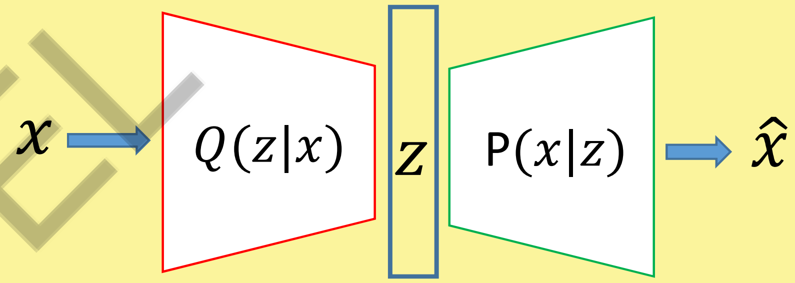
- Realize both $P(\cdot)$ and $Q(\cdot)$ with neural networks



Variational Autoencoder : Network Realization

- The z codes we get here should match with the distribution of $P(z)$ and we can decide what prior distribution to choose for $P(z)$.

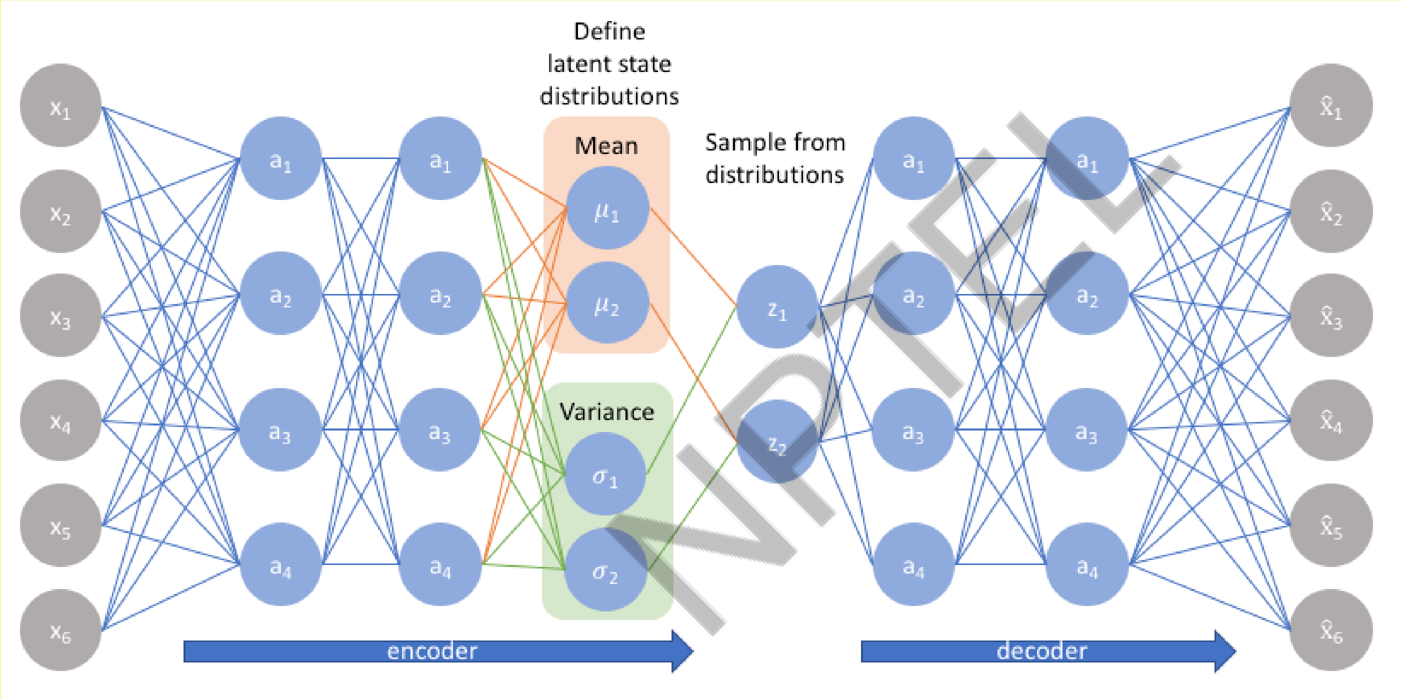
- Usual practice is to select a Normal distribution $N(0, I)$ for the prior.



Variational Autoencoder : Network Realization

- Instead of generating a fixed code for an input, Encoder now gives parameters of the distribution of the latent code.
- For a given input x, we need to generate mean vector $\mu(x)$ and diagonal covariance matrix, $\Sigma(x)$.
- We need to SAMPLE a code from that latent distribution and pass forward to the Decoder.

Variational Autoencoder : Network Realization



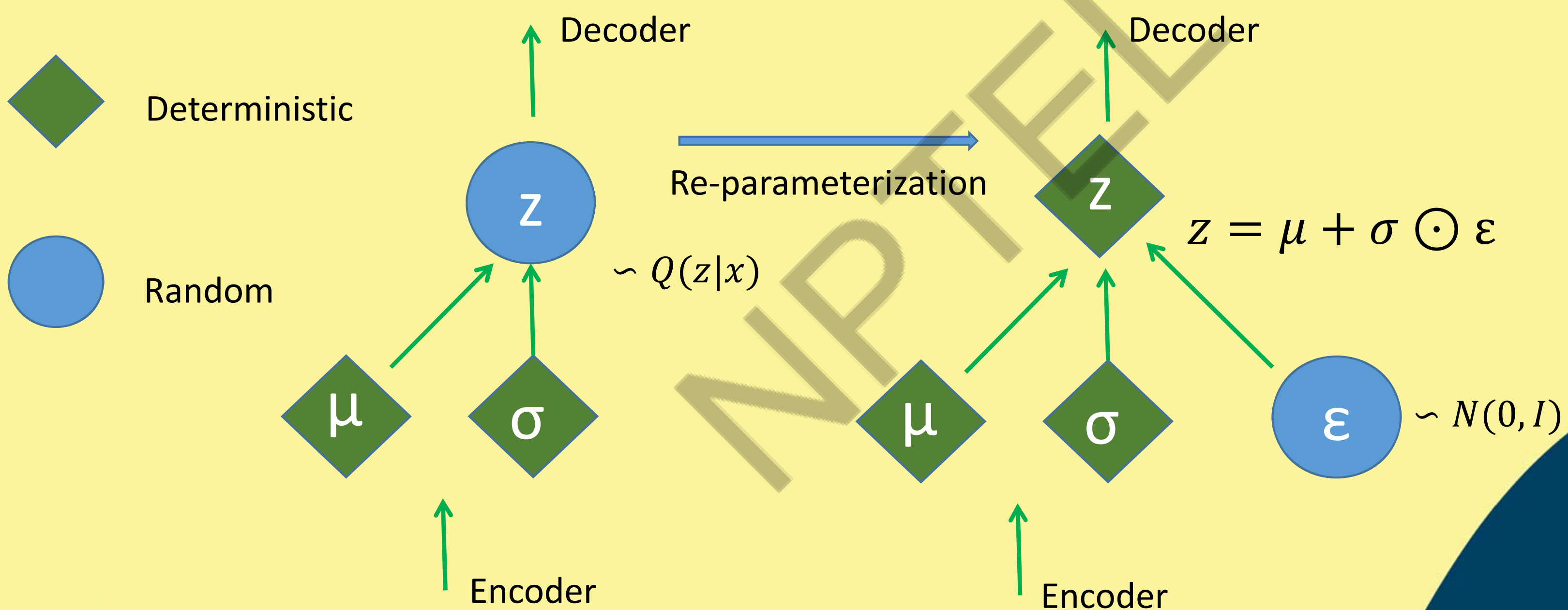
https://www.jeremyjordan.me/variational-autoencoders/

Variational Autoencoder : Network Realization

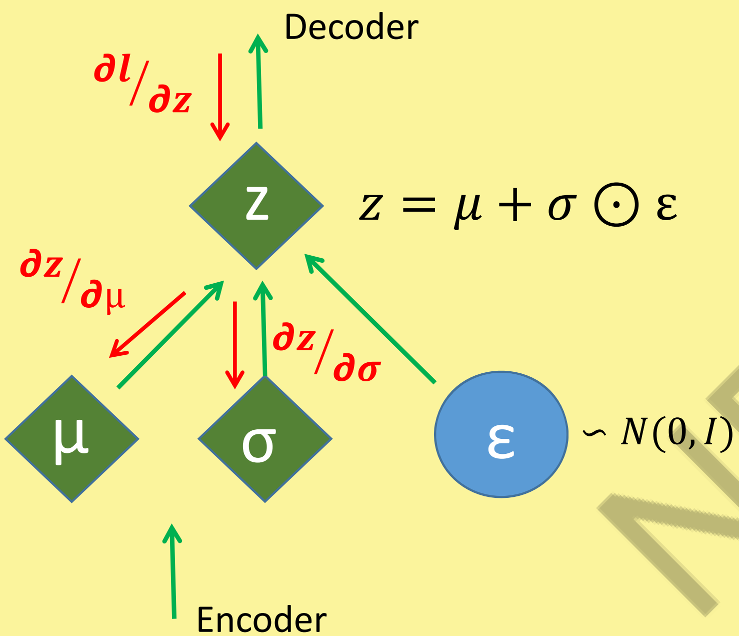
Sampling breaks computational graph
and
hinders Gradient Descent based
optimization

Variational Autoencoder : Reparameterization Trick

- We randomly sample ε from a unit Gaussian, and then shift the randomly sampled ε by the latent distribution's mean μ and scale it by the latent distribution's variance σ.



Variational Autoencoder : Reparameterization Trick



Re-parameterization enables

- ❑ Optimization of the parameters of the distribution.
- ❑ Still maintaining the ability to randomly sample from that distribution.

Variational Autoencoder : Coding the Cost Functions

$$E_{Q(z|x)} \log P(x|z) - KL(Q(z|x)\,||P(z))$$

Maximize

Minimize

Variational Autoencoder : Coding the Cost Functions

- Maximizing $E_{Q(z|x)} \log P(x|z)$ is a maximum likelihood estimation. It is observed all the time in discriminative supervised model, for example Logistic Regression, SVM, or Linear Regression.
- In the other words, given an input z and an output x, we want to maximize the conditional distribution $P(x|z)$ under some model parameters.
- So we could implement it by using any classifier with input z and output x, then optimize the objective function by using for example log loss or regression loss.

Variational Autoencoder : Coding the Cost Functions

- We want to minimize the second component of the loss, $KL((Q(z|x) \ || \ P(z))$
- We assumed that $P(z)$ follows $N(0, I)$, so we have to push $Q(z|x)$ towards $N(0, I)$

Assuming $P(z)$ to be $N(0, I)$ has 2 advantages:

- Easy to sample latent vectors from $N(0, I)$ when we want to generate samples.
- Assuming $Q(z|x)$ to be a Gaussian distribution with parameters, $\mu(x)$ and $\Sigma(x)$ allows $KL(Q(z|x) \ || \ P(z))$ to be in a closed form and easy for optimization.

NPTEL ONLINE CERTIFICATION COURSES

Thank you

NPTEL ONLINE CERTIFICATION COURSES

Course Name: Deep Learning

Faculty Name: Prof. P. K. Biswas
Department : E & ECE, IIT Kharagpur

Topic

Lecture 60: Generative Adversarial Network

CONCEPTS COVERED

Concepts Covered

- ❑ Generative Model
- ❑ Intuitions behind VAE
- ❑ Variational Inference
- ❑ Practical Realization of VAE
- ❑ Generative Adversarial Network
- ❑ Applications of GAN

Variational Autoencoder : Variational Inference

- ❑ Our initial objective: minimize KL($Q(z|x)$ || P($z|x$))
- ❑ Which is same as maximizing $\sum_z Q(z|x) \log \frac{P(x,z)}{Q(z|x)}$

Variational Lower Bound

➢ So, aim now is: *maximize*

$$L = \sum_z Q(z|x) \log \frac{P(x,z)}{Q(z|x)} = \sum_z Q(z|x) \log \frac{P(x|z)P(z)}{Q(z|x)}$$

Variational Autoencoder : Variational Inference

Maximize

$$\mathrm{L} = \sum_z Q(z|x) \log \frac{P(x,z)}{Q(z|x)} = \sum_z Q(z|x) \log \frac{P(x|Z)P(z)}{Q(z|x)}$$

$$= \sum Q(z|x) \log P(x|z) + \sum Q(z|x) \log \frac{P(z)}{Q(z|x)}$$

Variational Autoencoder : Variational Inference

Maximize

$$L = \sum_z Q(z|x) \log \frac{P(x,z)}{Q(z|x)} = \sum_z Q(z|x) \log \frac{P(x|Z)P(z)}{Q(z|x)}$$

$$= \underbrace{\sum Q(z|x) \log P(x|z)}_{E_{Q(z|x)} \log P(x|z)} + \underbrace{\sum Q(z|x) \log \frac{P(z)}{Q(z|x)}}_{-KL(Q(z|x)\, ||\, P(z))}$$

Variational Autoencoder : Variational Inference

Maximize

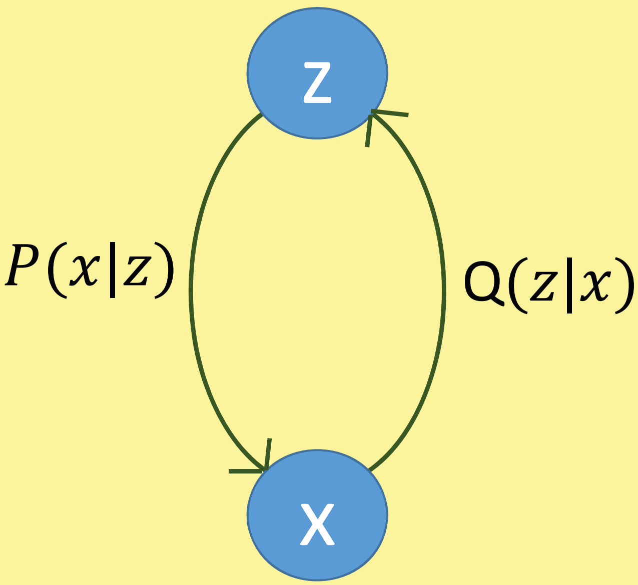
$$L = \sum_z Q(z|x) \log \frac{P(x,z)}{Q(z|x)} = \sum_z Q(z|x) \log \frac{P(x|Z)P(z)}{Q(z|x)}$$

$$= \underbrace{\sum Q(z|x) \log P(x|z)}_{E_{Q(z|x)} \log P(x|z)} + \underbrace{\sum Q(z|x) \log \frac{P(z)}{Q(z|x)}}_{-KL(Q(z|x)\,||\,P(z))}$$

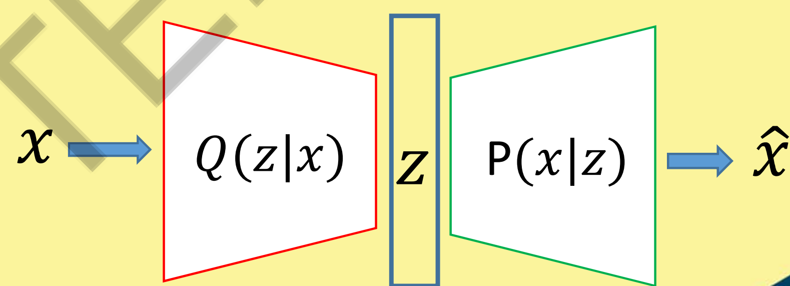
- Translate the loss functions into an auto-encoder architecture.

Variational Autoencoder : Network Realization

- We have the following graphical model



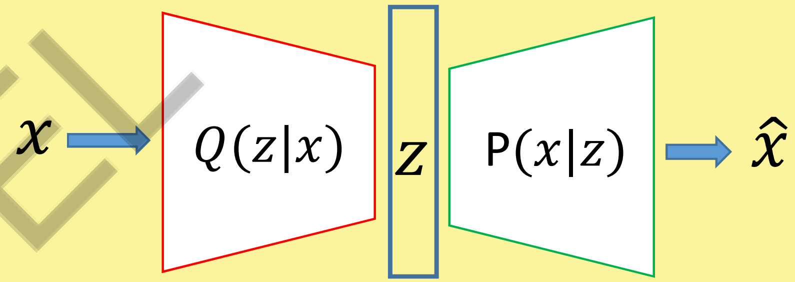
- Realize both $P(\cdot)$ and $Q(\cdot)$ with neural networks



Variational Autoencoder : Network Realization

- The z codes we get here should match with the distribution of $P(z)$ and we can decide what prior distribution to choose for $P(z)$.

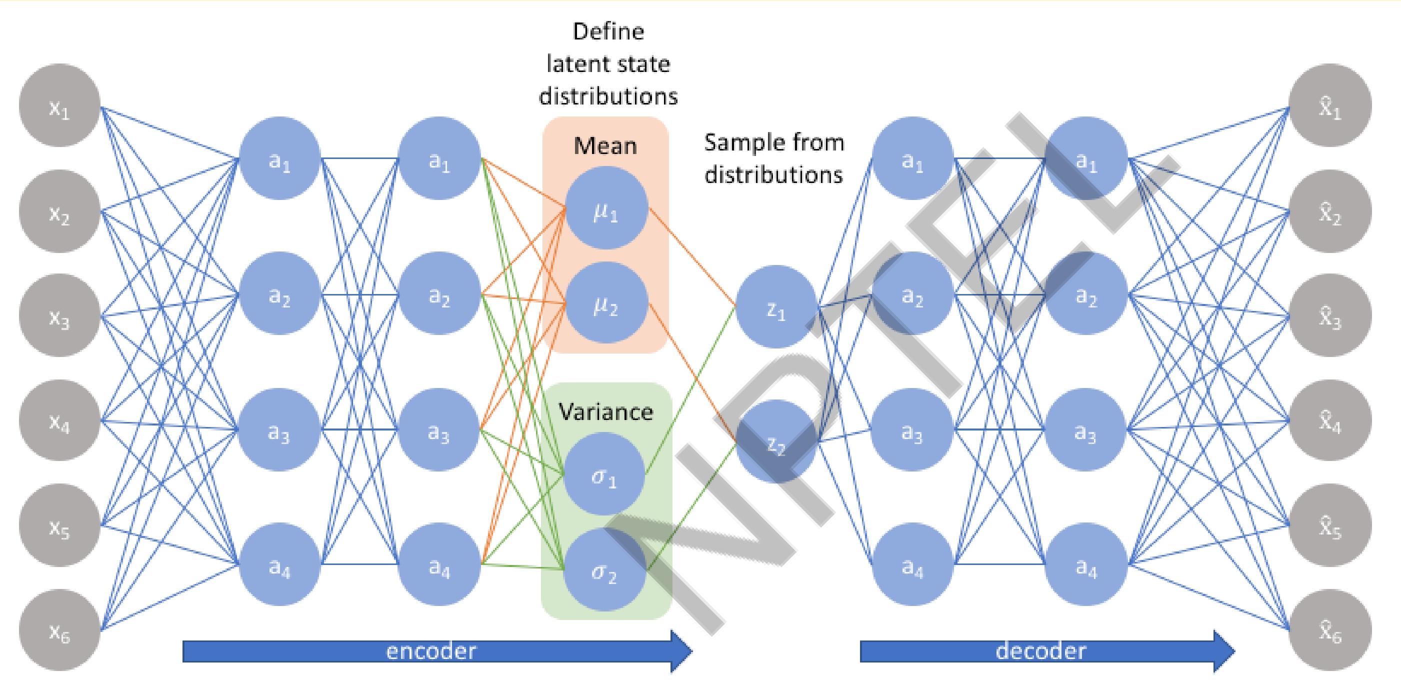
- Usual practice is to select a Normal distribution $N(0, I)$ for the prior.



Variational Autoencoder : Network Realization

- Instead of generating a fixed code for an input, Encoder now gives parameters of the distribution of the latent code.
- For a given input x, we need to generate mean vector $\mu(x)$ and diagonal covariance matrix, $\Sigma(x)$.
- We need to SAMPLE a code from that latent distribution and pass forward to the Decoder.

Variational Autoencoder : Network Realization



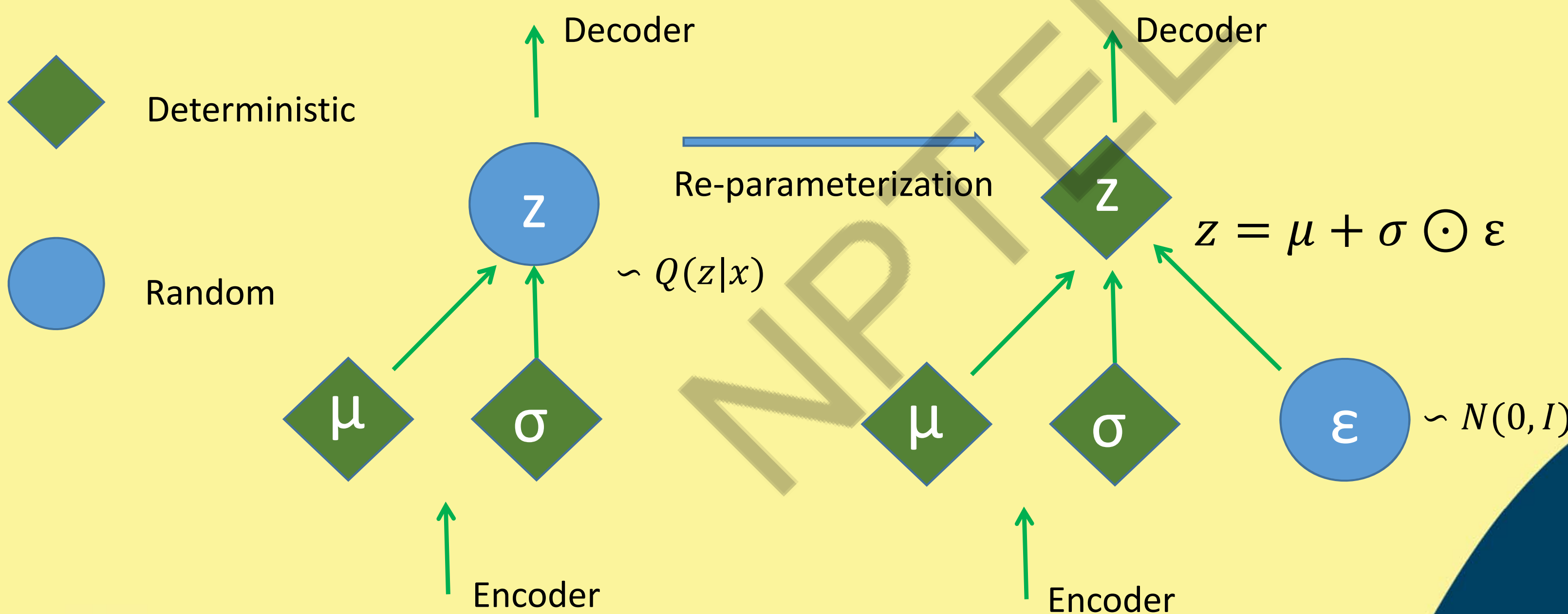
https://www.jeremyjordan.me/variational-autoencoders/

Variational Autoencoder : Network Realization

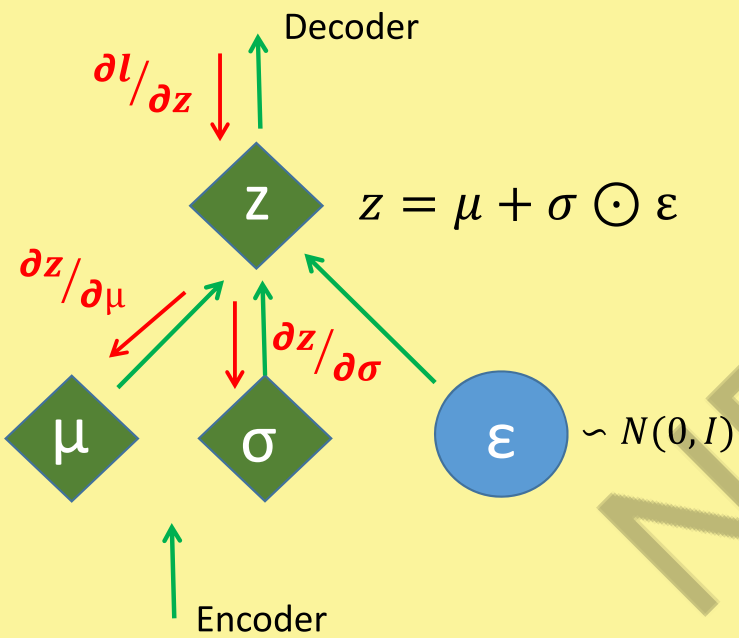
Sampling breaks computational graph
and
hinders Gradient Descent based
optimization

Variational Autoencoder : Reparameterization Trick

- We randomly sample ε from a unit Gaussian, and then shift the randomly sampled ε by the latent distribution's mean μ and scale it by the latent distribution's variance σ.



Variational Autoencoder : Reparameterization Trick



Re-parameterization enables

- ❑ Optimization of the parameters of the distribution.
- ❑ Still maintaining the ability to randomly sample from that distribution.

Variational Autoencoder : Coding the Cost Functions

$$E_{Q(z|x)} \log P(x|z) - KL(Q(z|x) \,||P(z))$$

Maximize

Minimize

Variational Autoencoder : Coding the Cost Functions

- Maximizing $E_{Q(z|x)} \log P(x|z)$ is a maximum likelihood estimation. It is observed all the time in discriminative supervised model, for example Logistic Regression, SVM, or Linear Regression.
- In the other words, given an input z and an output x, we want to maximize the conditional distribution $P(x|z)$ under some model parameters.
- So we could implement it by using any classifier with input z and output x, then optimize the objective function by using for example log loss or regression loss.

Variational Autoencoder : Coding the Cost Functions

- We want to minimize the second component of the loss, $KL((Q(z|x) \,||\, P(z))$
- We assumed that $P(z)$ follows $N(0, I)$, so we have to push $Q(z|x)$ towards $N(0, I)$

Assuming $P(z)$ to be $N(0, I)$ has 2 advantages:

- Easy to sample latent vectors from $N(0, I)$ when we want to generate samples.
- Assuming $Q(z|x)$ to be a Gaussian distribution with parameters, $\mu(x)$ and $\Sigma(x)$ allows $KL(Q(z|x) \,||\, P(z))$ to be in a closed form and easy for optimization.

Variational Autoencoder : Coding the Cost Functions

$$KL(N(\mu(x),\Sigma(x)) \,||\, N(0,I)) = 0.5 \, * [tr(\Sigma(x)) \;+\; \mu(x)^T \mu(x) \;-k \;-\log\det(\Sigma(x)))]$$

- k is dimension of the latent code
- $tr(\sum(x))$ is trace of a covariance matrix
- $\Sigma(x)$ is the diagonal covariance matrix. So, its determinant can be computed as product of its diagonal entries.
- In practice $\Sigma(x)$ can be predicted only as vector containing the diagonal entries

Variational Autoencoder : Coding the Cost Functions

$$KL(N(\mu(x), \Sigma(x)) \,||\, N(0, I) = 0.5 \, * [tr(\Sigma(x) \; + \; \mu(x)^T \, \mu(x) \; - k \; - \log\det(\Sigma(x)))]$$

$$= 0.5 * \left[\sum_k \Sigma(x)_k + \sum_k (\mu(x)_k)^2 + \sum_k 1 - \log \prod\nolimits_k \Sigma(x)_k\right]$$

$$= 0.5 * \left[\sum_k \Sigma(x)_k + \sum_k (\mu(x)_k)^2 + \sum_k 1 - \sum_k \log \Sigma(x)_k\right]$$

$$= 0.5 \, * \sum_k [\Sigma(x)_k + (\mu(x)_k)^2 + 1 + \log \Sigma(x)_k]$$

Variational Autoencoder : Coding the Cost Functions

In practice, we predict $\log \Sigma(x)$ instead of only $\Sigma(x)$ since it is numerically better to exponentiate a value during run time rather than taking log.

$$KL(N(\mu(x), \Sigma(x)) \,||\, N(0, I))$$

$$= 0.5 \ * \sum_{k} [\exp(\Sigma(x)_k) + (\mu(x)_k)^2 + 1 + \log \Sigma(x)_k]$$

Variational Autoencoder :After training

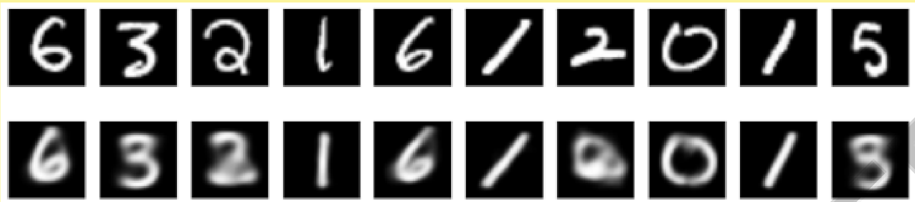
Visualizing Reconstructions:

Input

Reconstructions

Variational Autoencoder :After training

Visualizing Reconstructions:

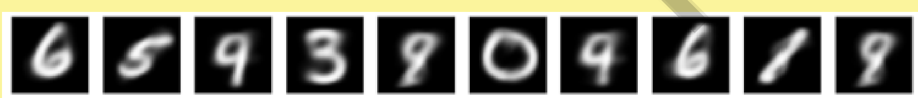


Input

Reconstructions

Using as Generative Model

- ❑ Sample a random vector from $N(0, I)$
- ❑ Feed forward the vector through the pre-trained Decoder



Generated Samples

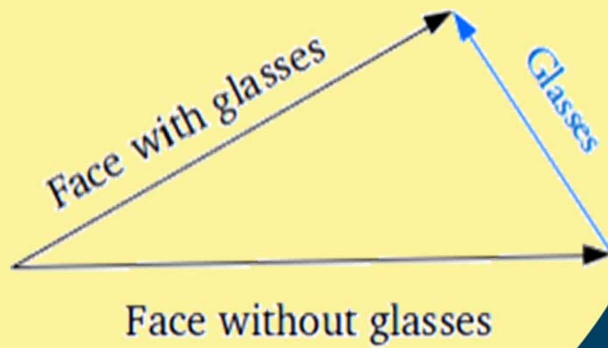
Variational Autoencoder : Generative Model

VAE generating novel faces after trained
on CelebA dataset

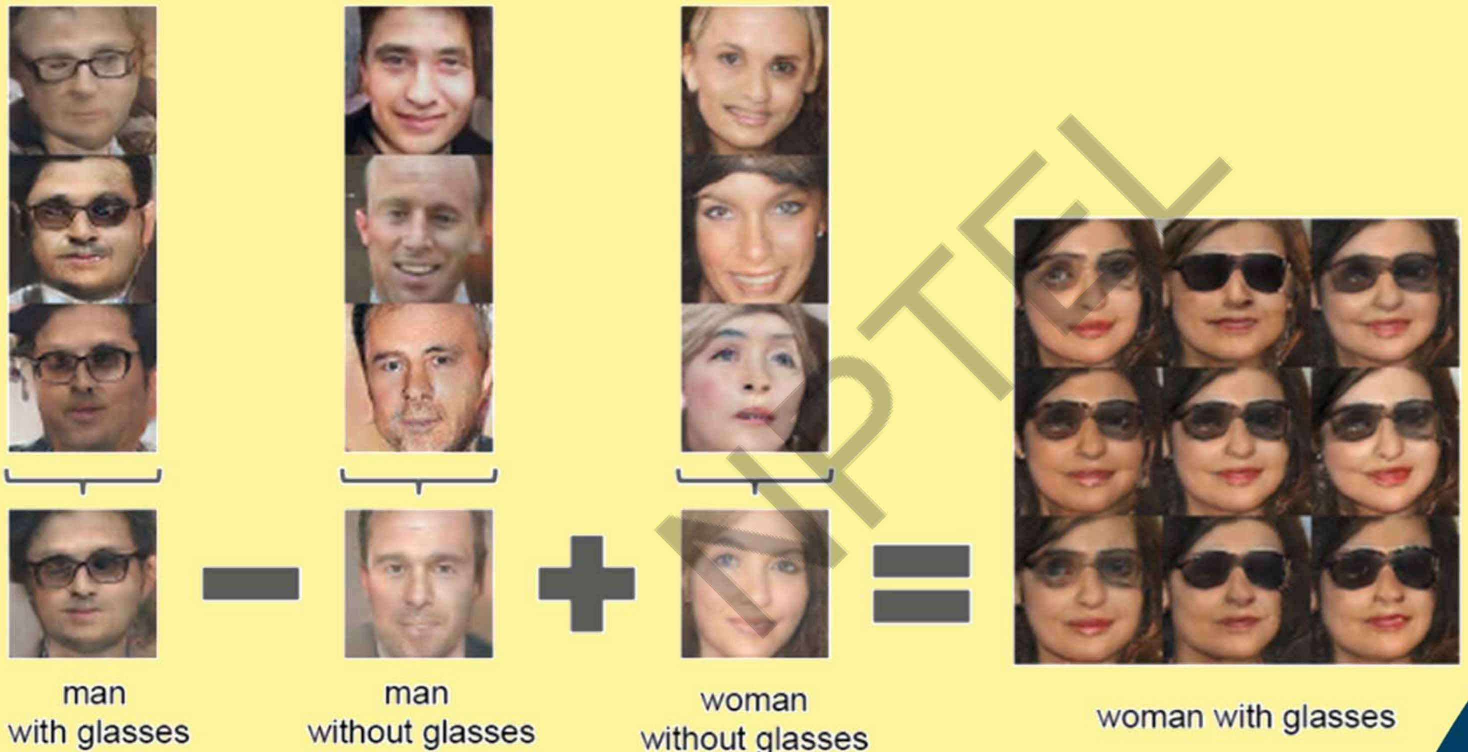
Variational Autoencoder : Vector Arithmetic

How do you interpolate between two samples ?

- Take a face image with glasses and find the latent code (C_1)
- Take another face without glasses and find latent code (C_2)
- $C_3 = C_1 - C_2$ gives code for glasses
- Take a new face without glasses and find latent code (C_4)
- $C_3 + C_4$ will overlay glasses on this new image
- Such transitions are possible only if the latent space is continuous instead of clusters



Variational Autoencoder : Generative Model

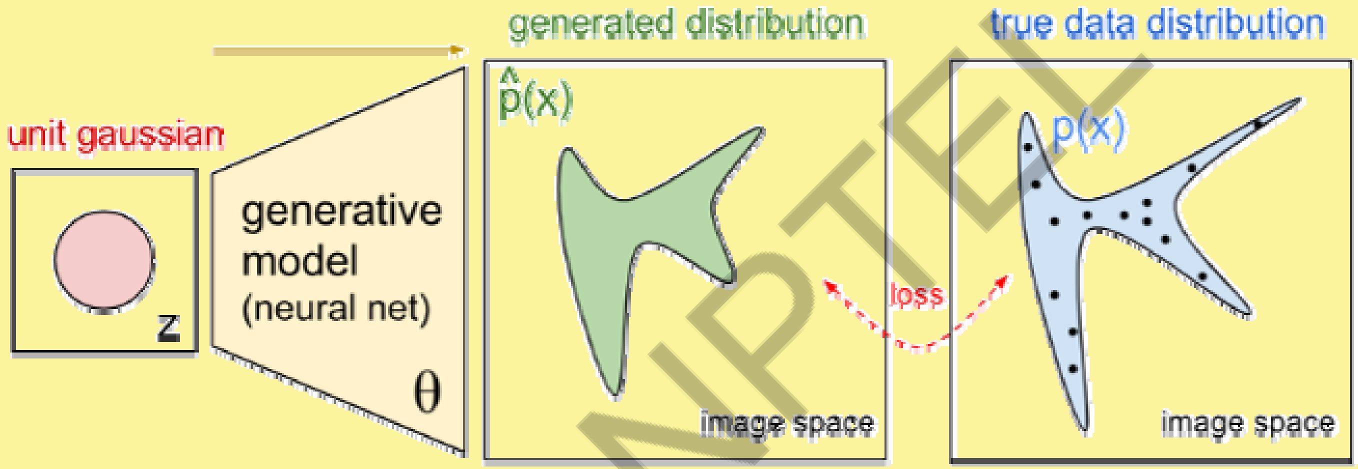


Generative Adversarial Network
(GAN)

Implicit Generative Models

- Implicitly defines a probability distribution.
- Sample code vector, z, from a simple and fixed distribution (e.g. spherical Gaussian or Uniform).
- A generator network is trained as a differentiable network to map z to a data point x.

Implicit Generative Models



https://openai.com/blog/generative-models/

Implicit Generative Models

- Blue Region shows areas with high probability of real image.
- Black dots represent actual images from true distribution $p(x)$.
- Generative model (parameterized by θ) also describes a function $\hat{p}$(x)
  - Takes points (latent codes) from an unit Gaussian distribution.
  - Maps those points to a generator distribution.
  - θ can be optimized to reduce $KL(p(x)||\hat{p}(x))$
  - Green distribution starts randomly then aligns with blue distribution

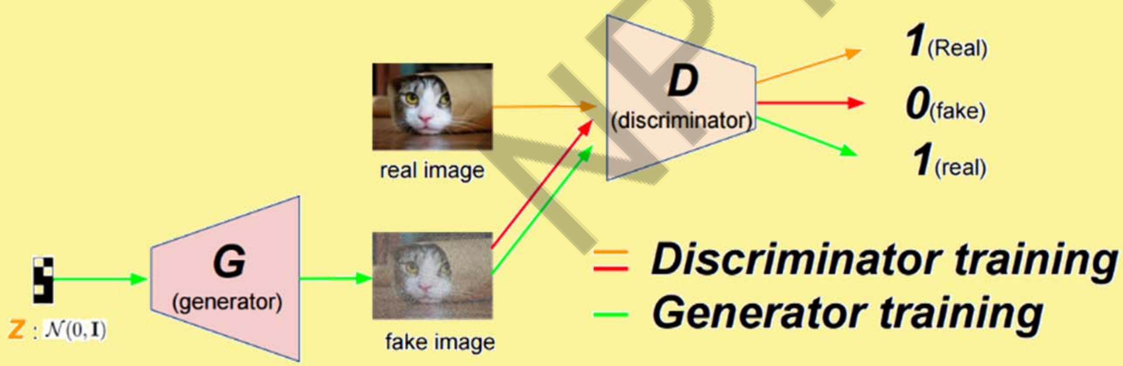
https://openai.com/blog/generative-models/

GAN Overview

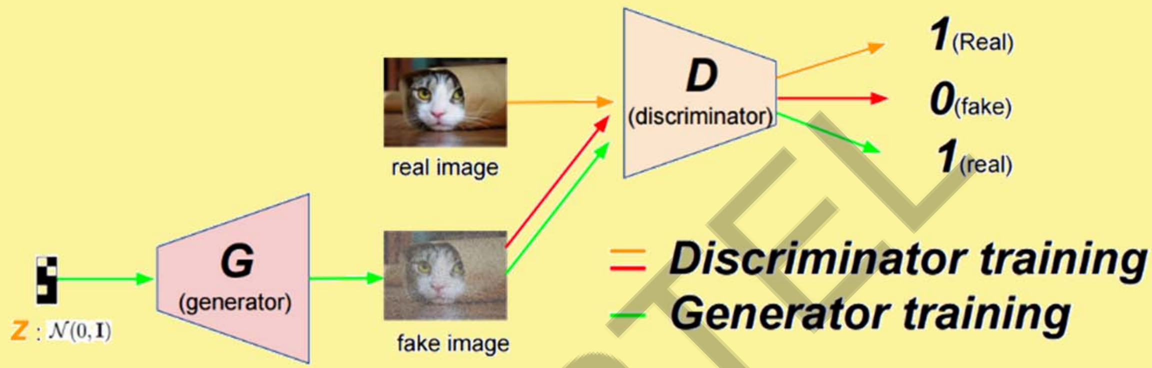
- In GAN the main idea is to have two neural networks compete with each other.
- Its Game Theoretic Approach.
  - **Generator** network samples a z vector and tries to produce realistic samples.
  - **Discriminator** network tries to distinguish fake samples (from Generator) and real samples.

GAN Overview

- Assume $D(x)$ represents probability of belonging to real class for a given sample, x
- Discriminator will try to increase $D(x)$ for real samples and decrease $D(x)$ for fake/generated samples
- Generator will try to increase $D(x)$ for generated samples



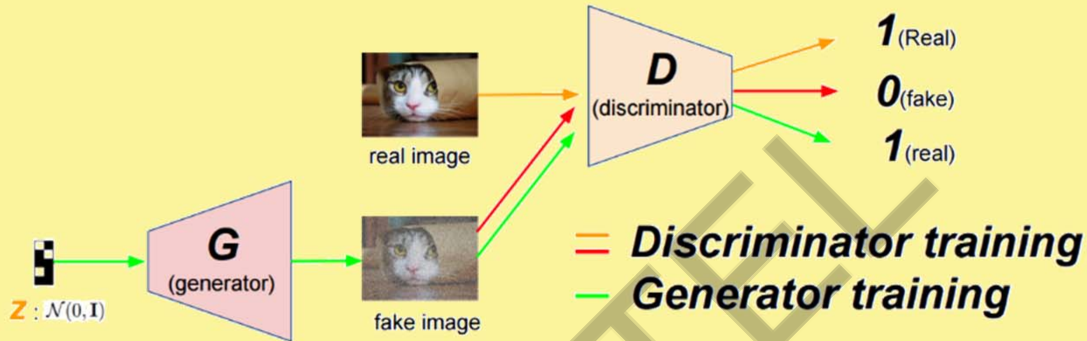
GAN Overview



Training the Discriminator

$$\max_D V(D,G) = \underbrace{E_{x\sim p_{data}(x)} \log D(x)}_{\text{Maximize probability for real}} + \underbrace{E_{z\sim p_z(z)}[\log\{1 - D(G(z))\}]}_{\text{Minimize probability for generated}}$$

GAN Overview



Training the Generator

$$\min_G V(D,G) = \quad E_{z\sim p_z(z)}[\log\{1 - D(G(z))\}]$$

$$\equiv \underbrace{\max_G \quad E_{z\sim p_z(z)}[\log D(G(z))]}_{\text{Maximize probability for generated}}$$

GAN Training : Alternate updates of D and G

for number of training iterations **do**

for k steps **do**

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right)\right].$$

end for

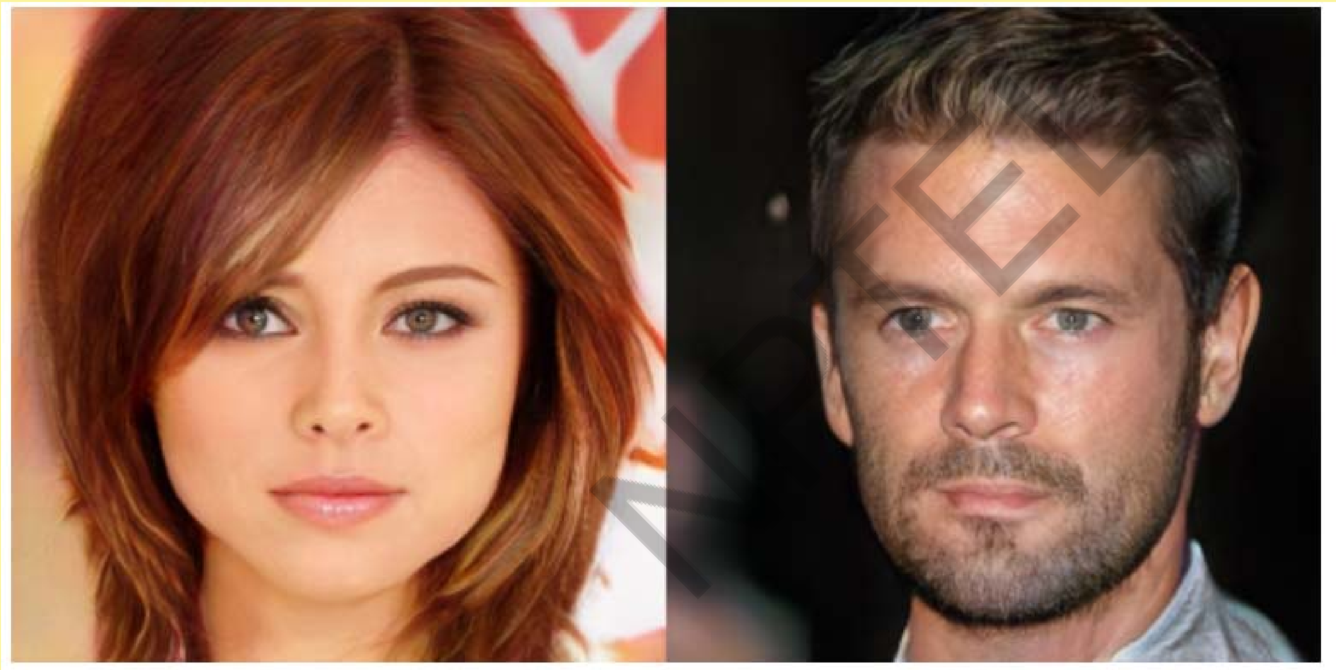
- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

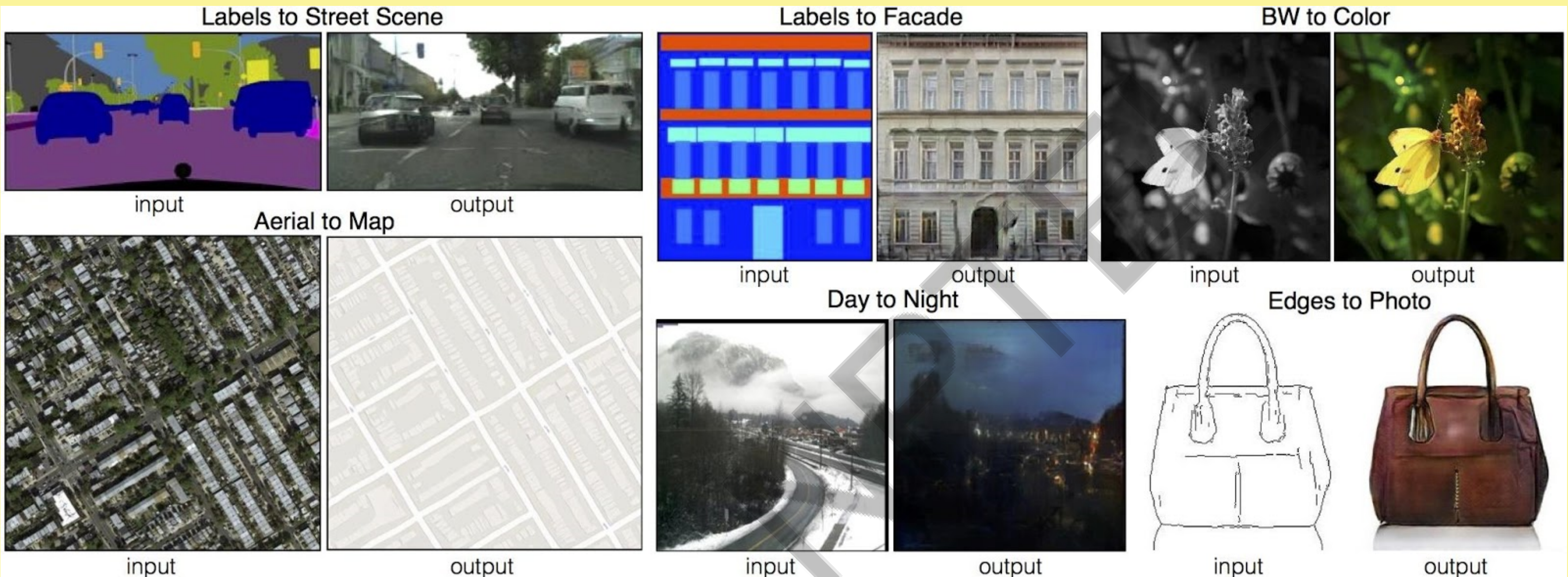
Goodfellow et al. “Generative Adversarial Networks”, NeuIPS, 2014

GAN Applications: High Resolution Image Synthesis



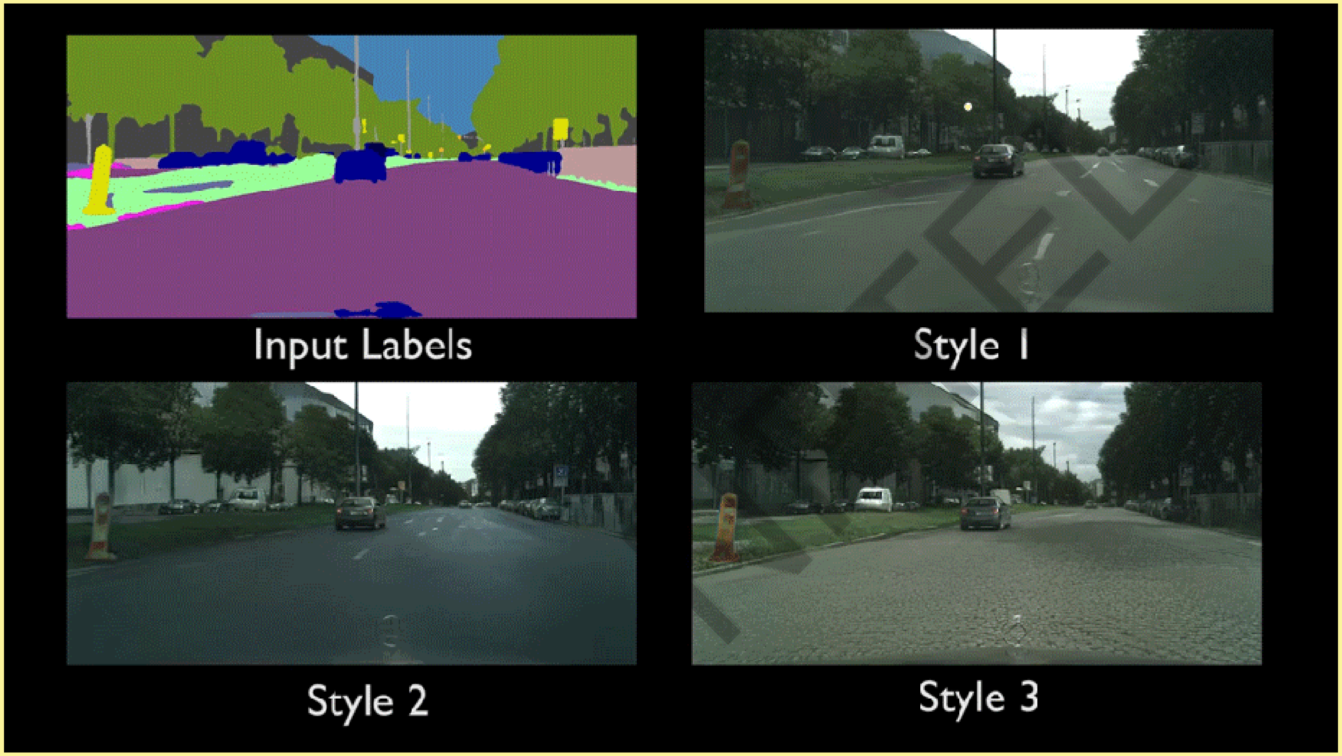
Karras, Tero, Timo Aila, Samuli Laine, and Jaakko Lehtinen. "Progressive growing of gans for improved quality, stability, and variation." ICLR, 2018.

GAN Applications: Image to Image Translation



*Isola, Phillip, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. "Image-to-image translation with conditional adversarial networks." *CVPR, 2017*

GAN Applications: Video to Video Translations



Wang, Ting-Chun, Ming-Yu Liu, Jun-Yan Zhu, Guilin Liu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. "Video-to-video synthesis." *NeurIPS, 2018*

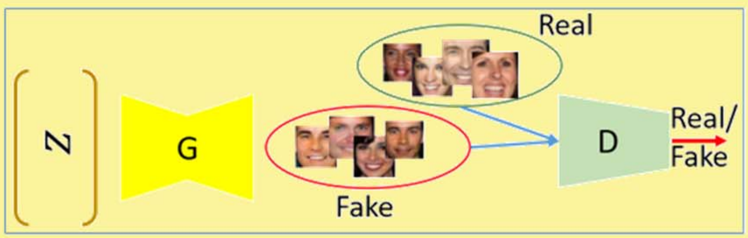
GAN Applications: Image Inpainting

Input: Masked/damaged image, I_d, with binary mask, M

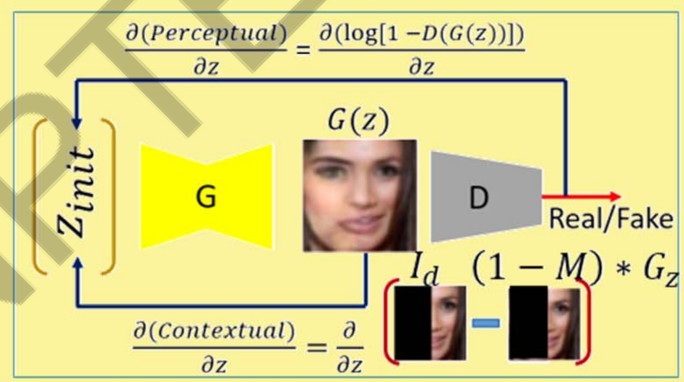
Intermediate Output : Image, I_G, after iterative optimization for z

Final Output: Inpainted image, $\widehat{I_d} = M * I_d + (1 - M) * I_G$

Stage 1: Pre-training a GAN



Stage 2: Iterative search for z



Yeh, R. A., Chen, C., Yian Lim, T., Schwing, A. G., Hasegawa-Johnson, M., & Do, M. N. (2017). Semantic image inpainting with deep generative models. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

GAN Applications: Image Inpainting

PROPOSED IMAGE INPAINTING (5X SPEEDUP)

Nearest Neighbour Search for Better z Initialization

- Sample a pool of z vectors and pass through pre-trained G
- Data + Structure loss = $L_{nn}(\cdot)$ between masked, I_d & pooled, I_p^i
- Select z_{init} as initial solution, s.t: $z_{init} = \underset{z^i}{argmin}\, L_{nn}\left(I_d, G\left(z^i\right)\right)$

Data Loss, L_D

$$L_D^i = |I_d - M * p_i|$$

Structure Loss, L_S

$$L_s^i = \left|\Delta_x I_d - \Delta_x M * I_p^i\right| + \left|\Delta_y I_d - \Delta_y M * I_p^i\right|$$

Both uses only unmasked pixels info

Lahiri, A., Jain, A. K., Nadendla, D., & Biswas, P. K., "Faster Unsupervised Semantic Inpainting: A GAN Based Approach", ICIP 2019.

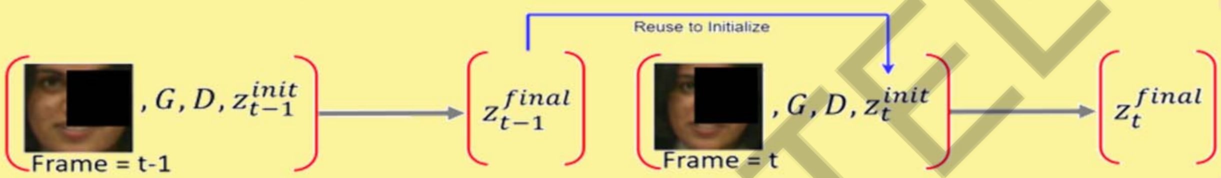
GAN Applications: Image Inpainting

Lahiri, A., Jain, A. K., Nadendla, D., & Biswas, P. K., "Faster Unsupervised Semantic Inpainting: A GAN Based Approach", ICIP 2019.

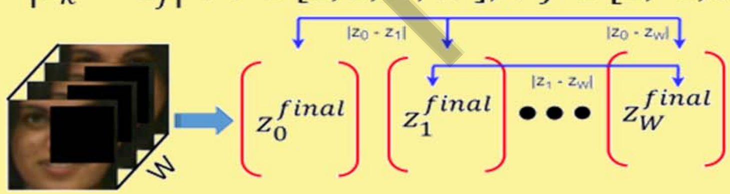
GAN Applications: Video Inpainting

PROPOSED VIDEO INPAINTING (80X SPEEDUP)

- **Reuse Noise Priors**
  - Exploit temporal redundancy
  - Temporal neighbours should have close z representaitons



- **Group Consistency Loss**
  - Penalize if a local temporal neighbourhood of W frames differ
  - Helps in reducing sudden flickering effects across frames
  - $Loss = |z_k - z_j| \; \forall\, i \in [1, 2, .., W]; \forall\, j \in [1, 2, .., W]$



Lahiri, A., Jain, A. K., Nadendla, D., & Biswas, P. K., "Faster Unsupervised Semantic Inpainting: A GAN Based Approach", ICIP 2019.

GAN Applications: Video Inpainting

Lahiri, A., Jain, A. K., Nadendla, D., & Biswas, P. K., “Faster Unsupervised Semantic Inpainting: A GAN Based Approach”, ICIP 2019.

NPTEL ONLINE CERTIFICATION COURSES

Thank you