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*

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

Variational Autoencoder : Variational Inference

Maximize

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

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

Variational Autoencoder : Variational Inference

Maximize

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

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

Variational Autoencoder : Variational Inference

Maximize

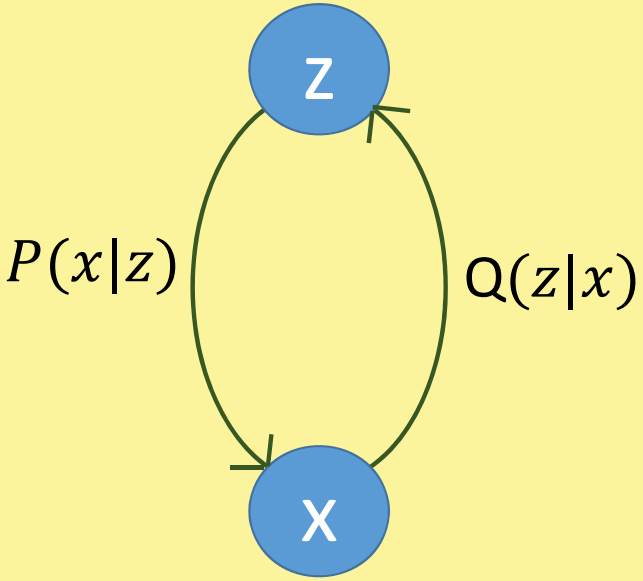
$$\mathrm{L} = \sum_z Q(z|x) \log \frac{P(x,z)}{Q(z|x)} = \sum_z Q(z|x) \log \frac{P(x|Z)P(z)}{Q(z|x)}$$

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

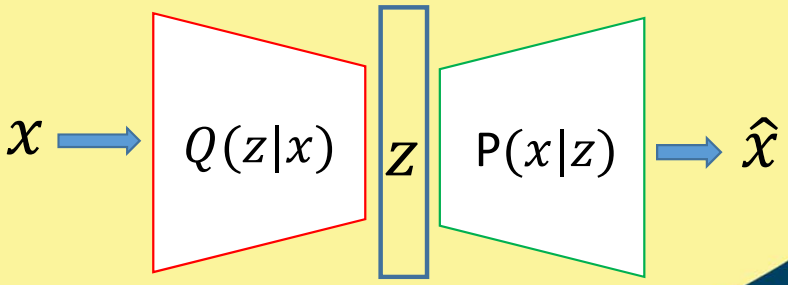
- Translate the loss functions into an auto-encoder architecture.


Variational Autoencoder : Network Realization

- We have the following graphical model



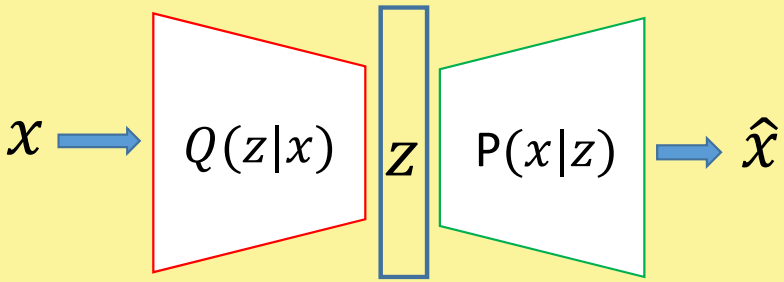
- Realize both $P(\cdot)$ and $Q(\cdot)$ with neural networks



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 $P(z)$.

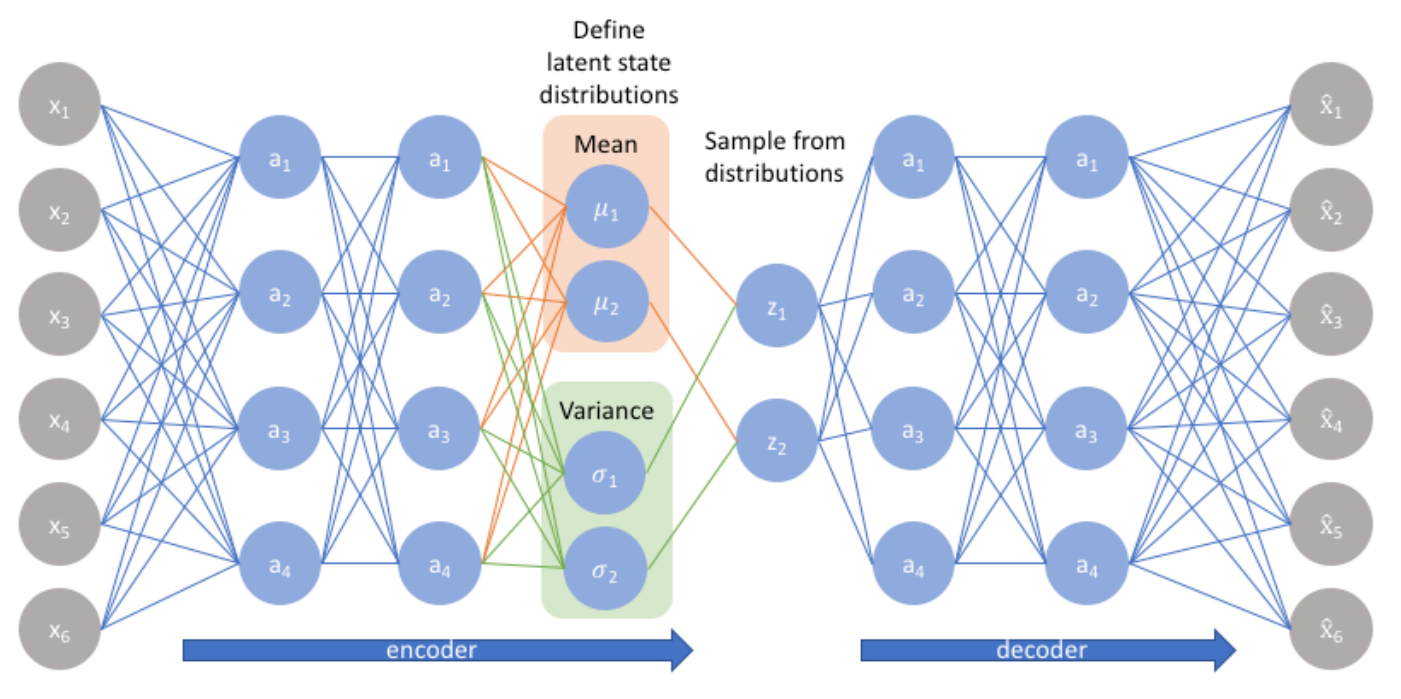
- Usual practice is to select a Normal distribution $N(0, I)$ for the prior.



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



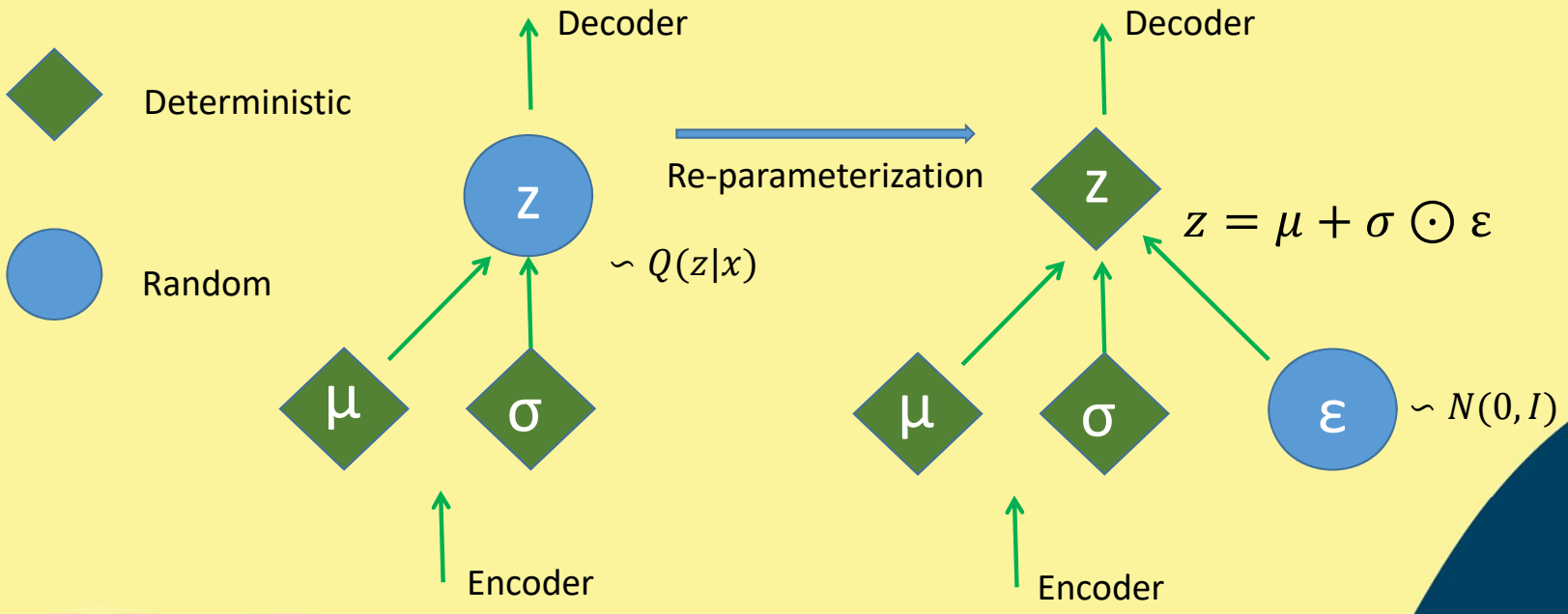
https://www.jeremyjordan.me/variational-autoencoders/

Variational Autoencoder : Network Realization

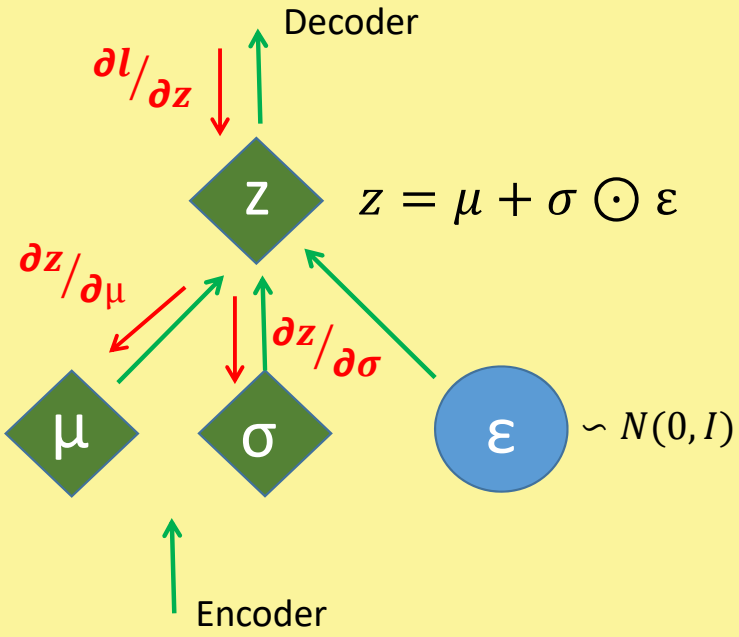
Sampling breaks computational graph
and
hinders Gradient Descent based
optimization

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

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

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

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

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

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

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

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

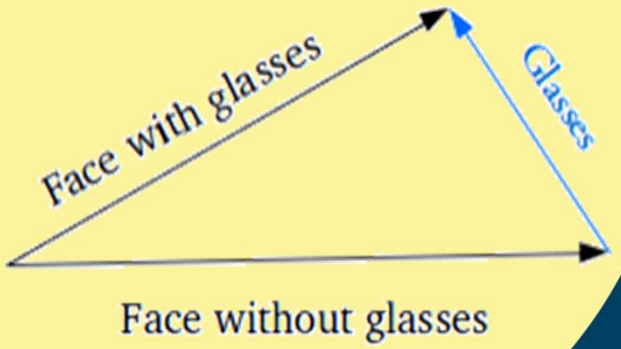
Variational Autoencoder : Generative Model

VAE generating novel faces after trained on CelebA dataset

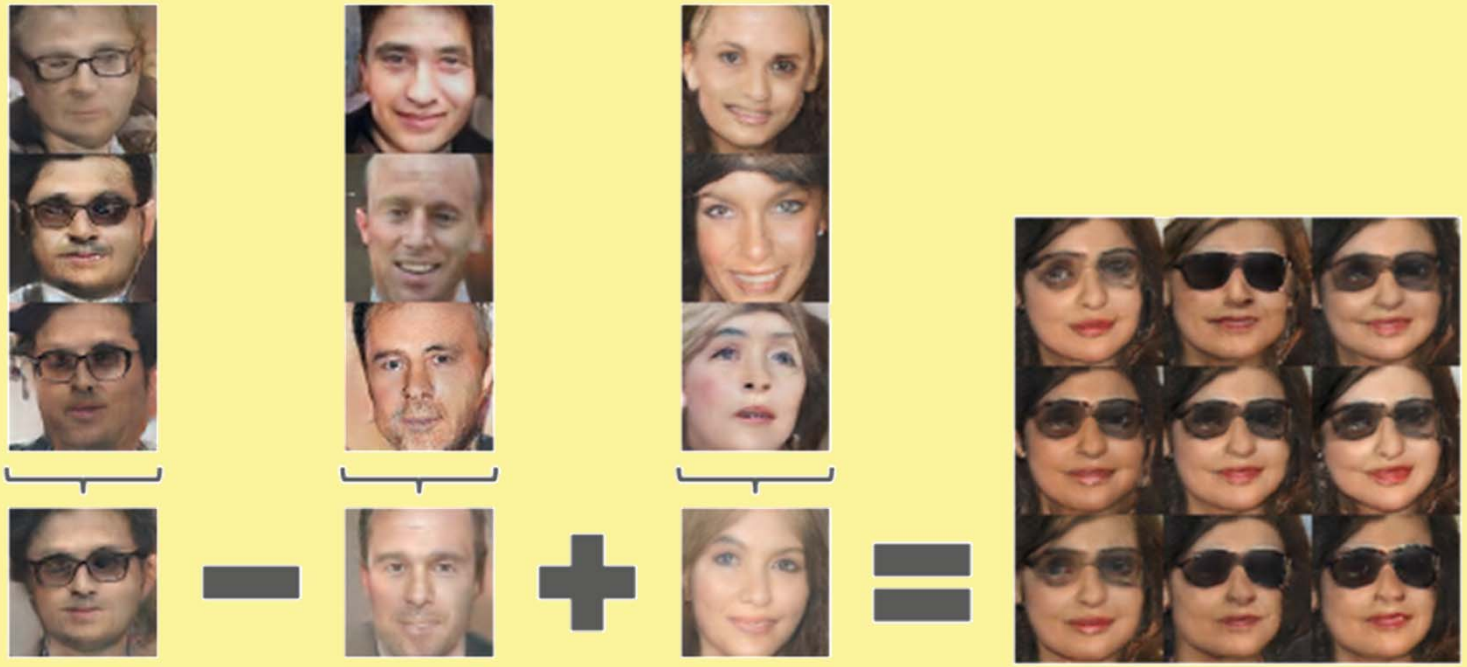
Variational Autoencoder : Vector Arithmetic

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

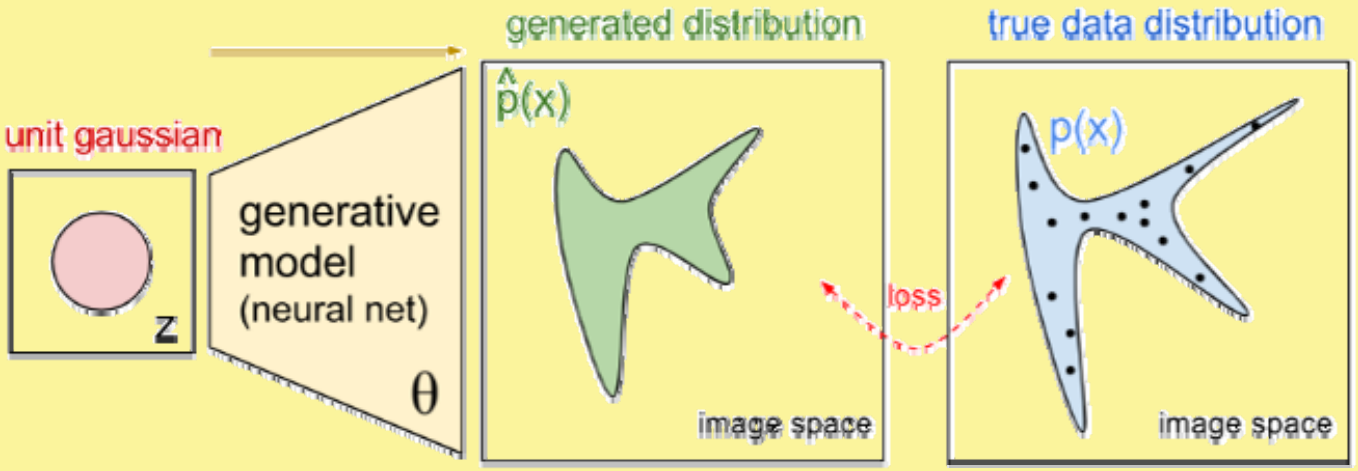


Generative Adversarial Network (GAN)

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

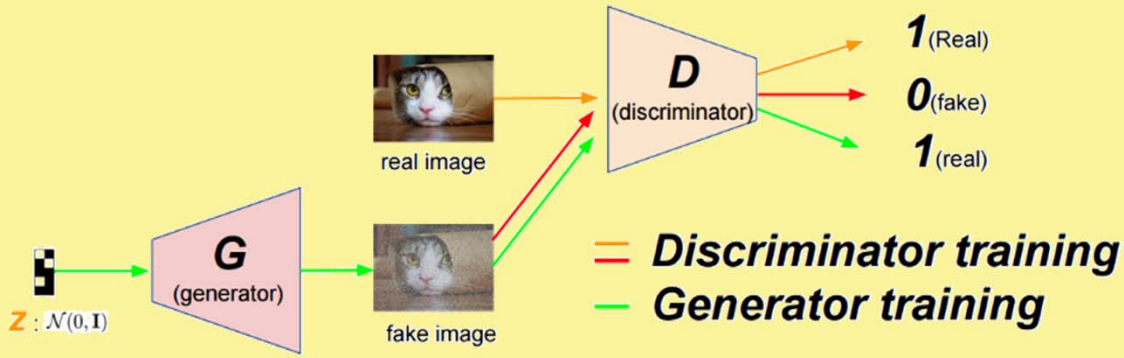

https://openai.com/blog/generative-models/

GAN Overview

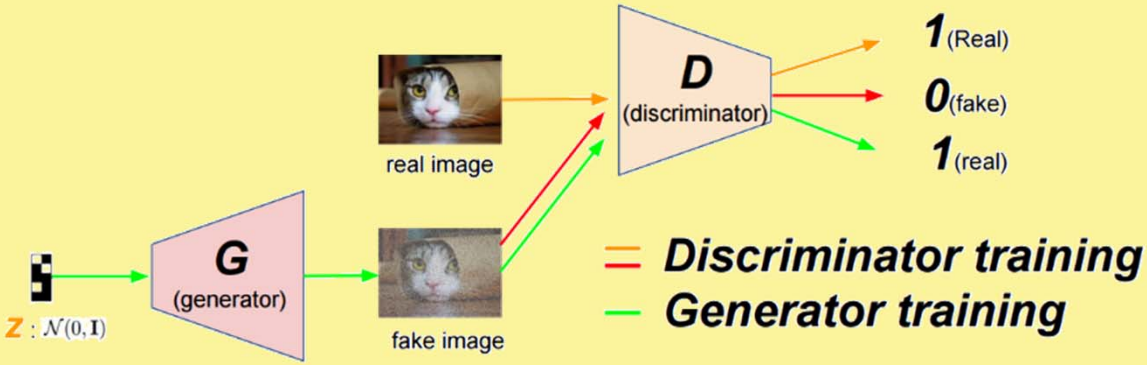
- In GAN the main idea is to have two neural networks compete with each other.
- Its Game Theoretic Approach.
  - **Generator** network samples a z vector and tries to produce realistic samples.
  - **Discriminator** network tries to distinguish fake samples (from Generator) and real samples.


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



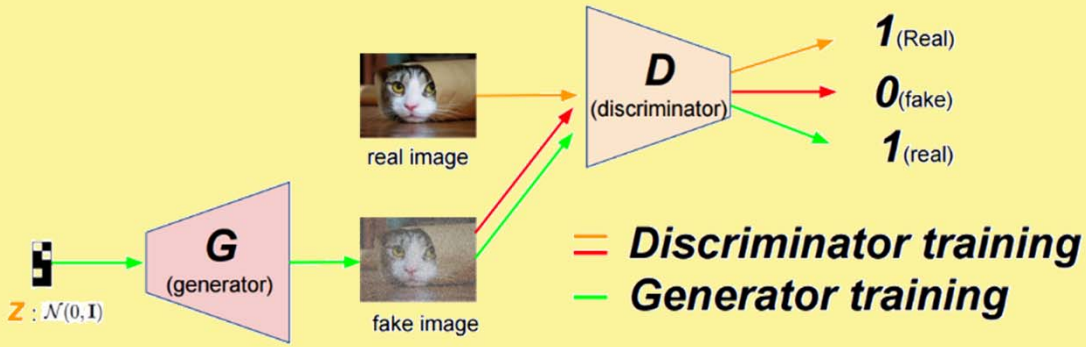
GAN Overview



Training the Discriminator

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

GAN Overview



Training the Generator

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

$$\equiv \max_G E_{z\sim p_z(z)}[\log D(G(z))]$$

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:

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

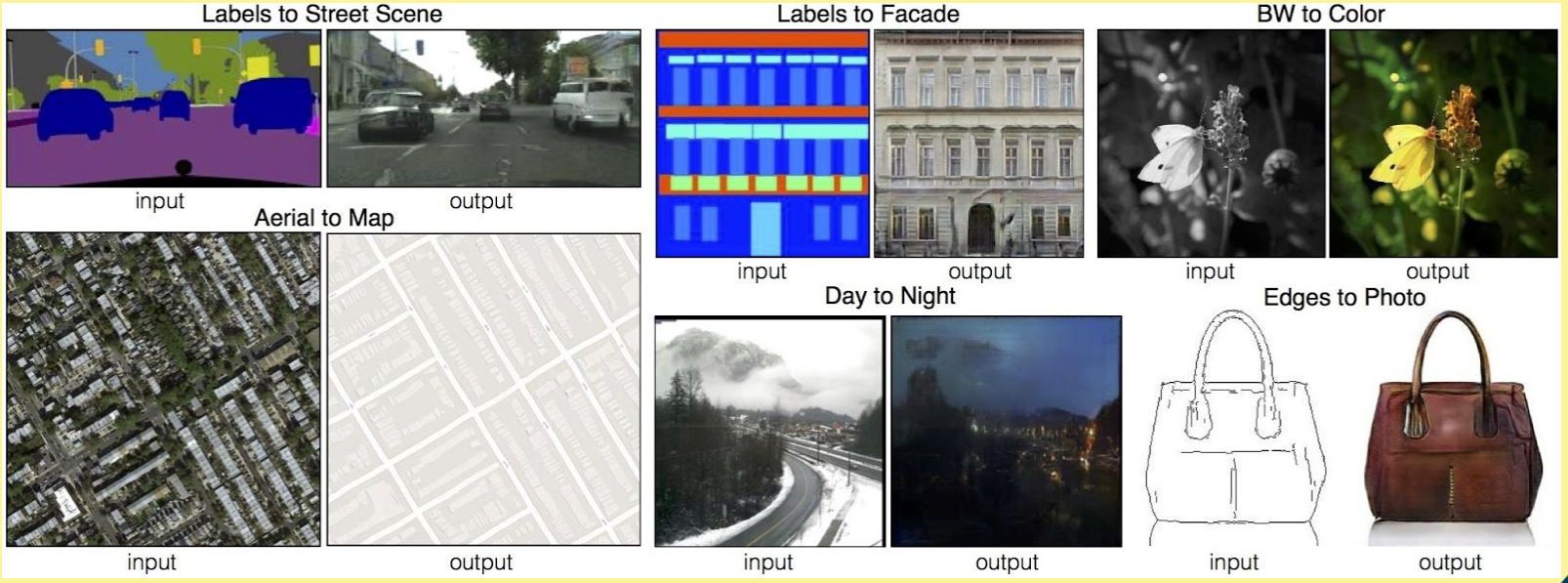
end for

Goodfellow et al. "Generative Adversarial Networks", NeuIPS, 2014

GAN Applications: High Resolution Image Synthesis

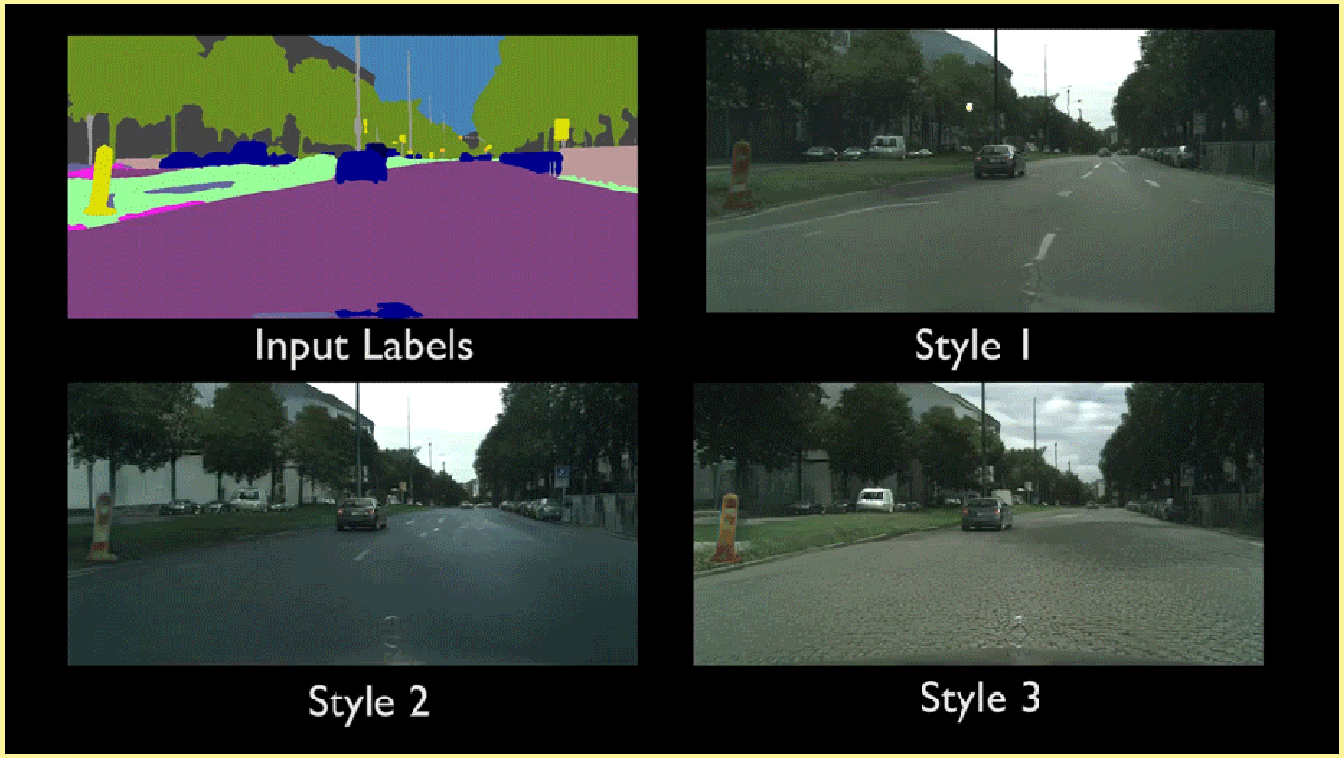
Karras, Tero, Timo Aila, Samuli Laine, and Jaakko Lehtinen. "Progressive growing of gans for improved quality, stability, and variation." ICLR, 2018.

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*

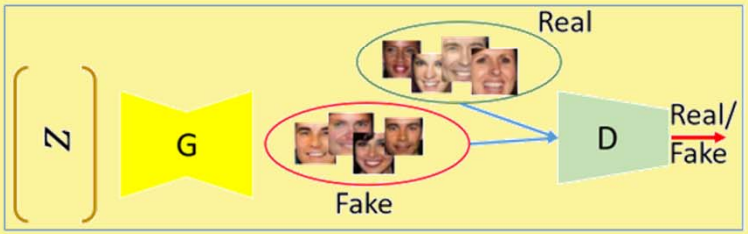
GAN Applications: Image Inpainting

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

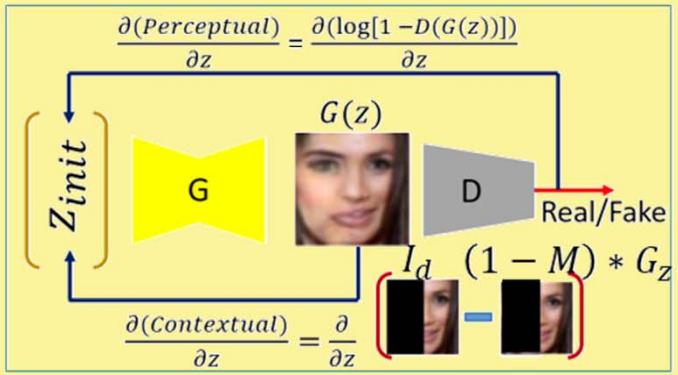
Intermediate Output : Image, I_G, after iterative optimization for z

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: $z_{init} = \underset{z^i}{argmin}\, L_{nn}\left(I_d, G(z^i)\right)$

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

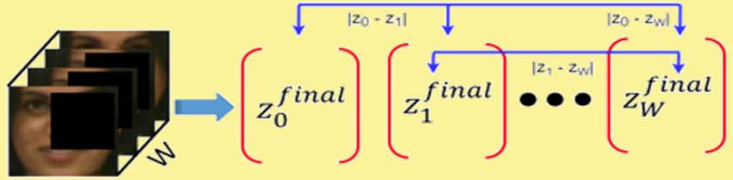
Lahiri, A., Jain, A. K., Nadendla, D., & Biswas, P. K., "Faster Unsupervised Semantic Inpainting: A GAN Based Approach", ICIP 2019.

GAN Applications: Video Inpainting

PROPOSED VIDEO INPAINTING (80X SPEEDUP)

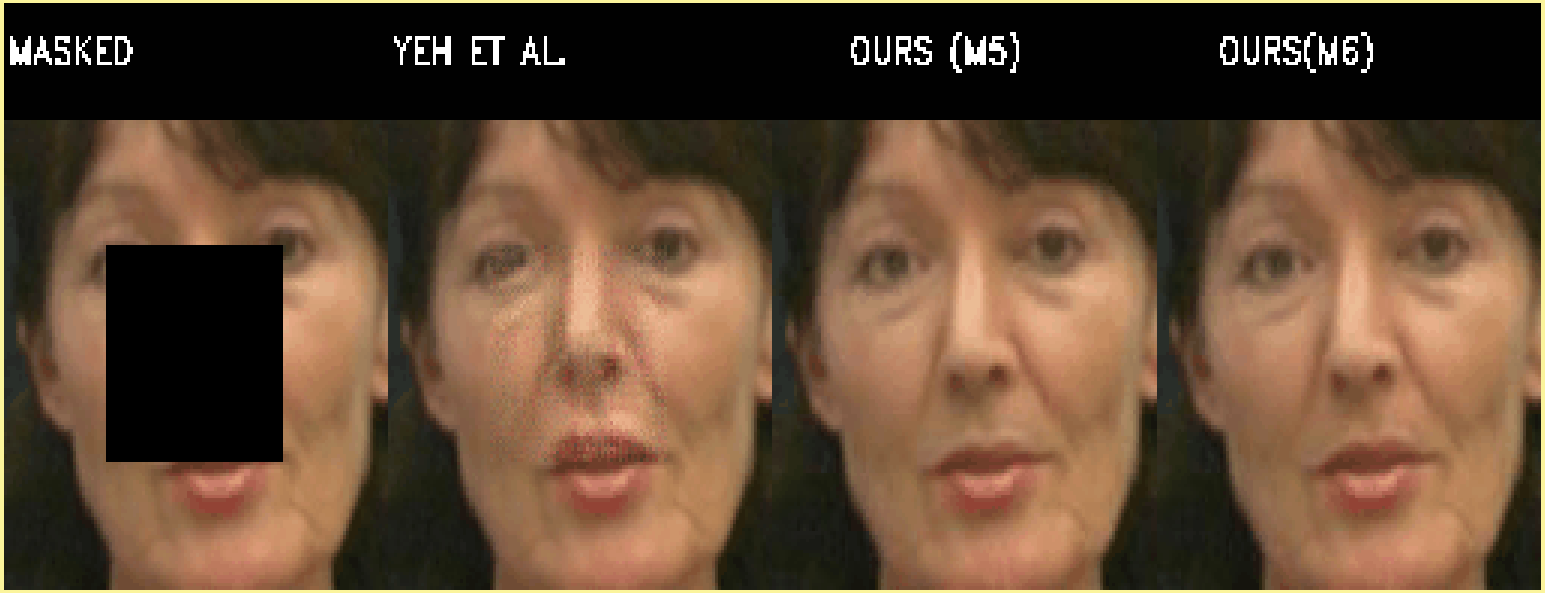
- **Reuse Noise Priors**
  - Exploit temporal redundancy
  - Temporal neighbours should have close z representaitons

- **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