Generative models

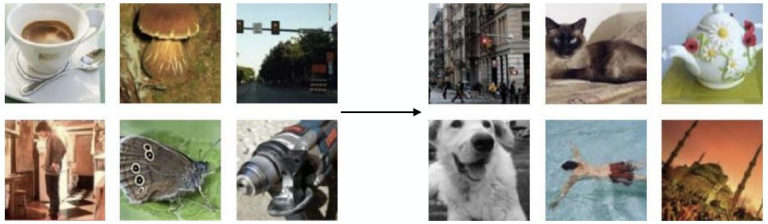
Mohammad Havaei

July 5, 2022

Why Generative models?

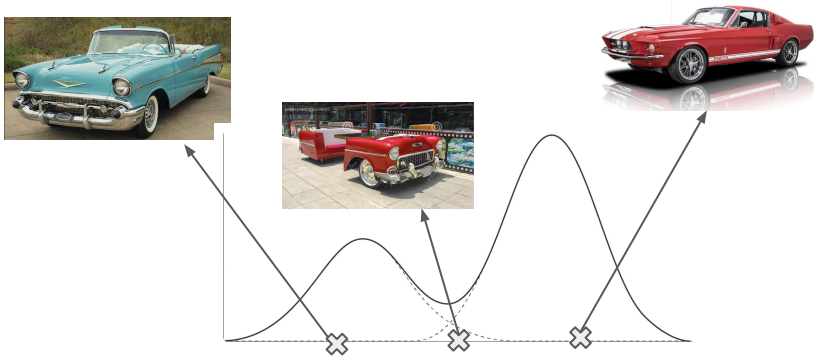
- Density estimation

- Samples from data



images taken from Goodfellow (2017)

- Outlier detection (learning a prior model over data)



- Simulation (Learning on synthetic data)

Generative Models

- Variational Autoencoders
- Generative Adversarial Networks
- Autoregressive models
- Diffusion Models

Latent Variable Models

learn a mapping from some low dimensional latent variable **z** to a complicated distribution on **x**.

z is simple, you can easily sample from it and encodes the information you care about in a disentangled way.

g is the function that maps samples from that simple space to the complicated data space in a way that is coherent

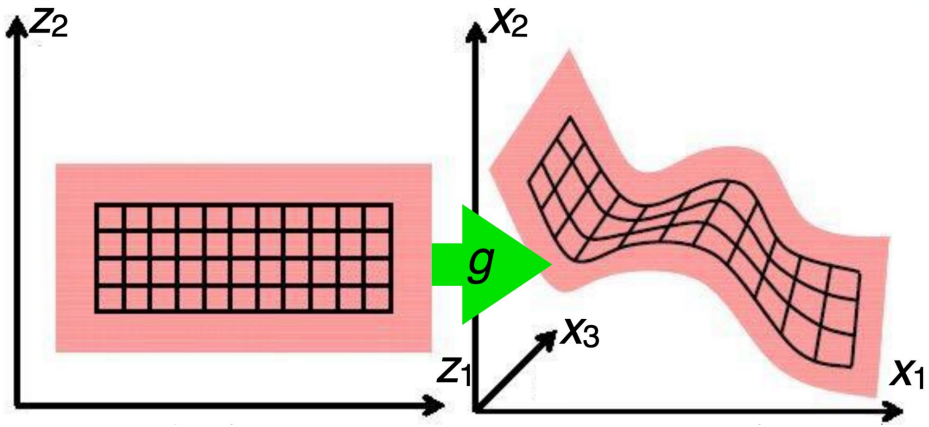
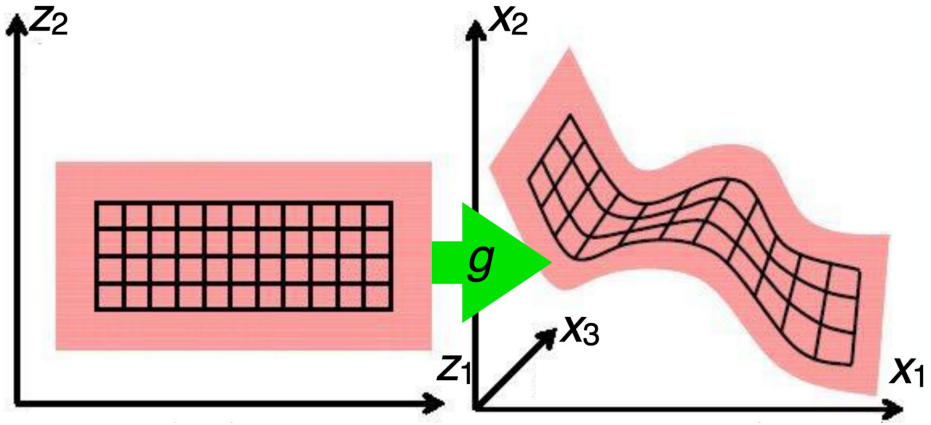


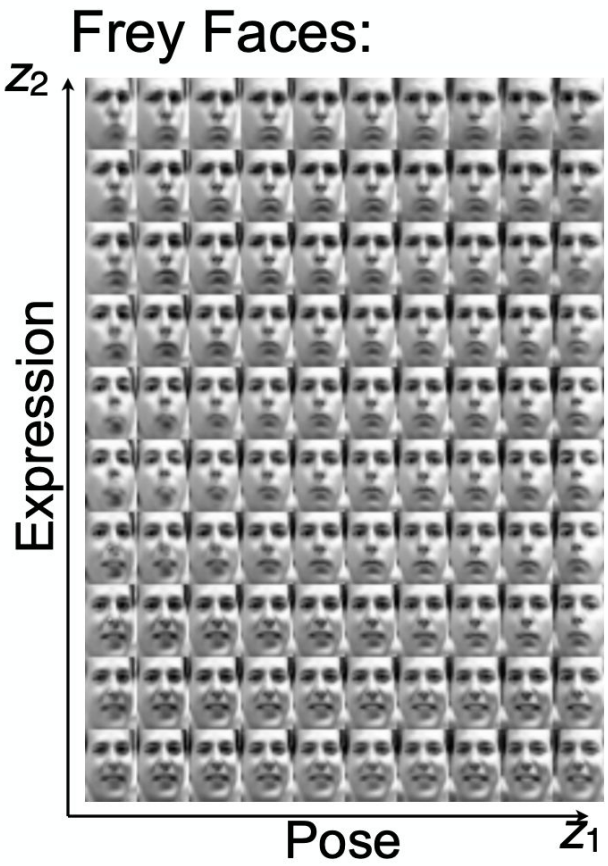
image from: [3D Surface Parameterization Using Manifold Learning for Medial Shape Representation]

Latent Variable Models

2 factors of variation captured in the latent space: *Pose & Expression*

Using latent variable model we have learned to ***disentangle*** these features





Latent Variable Models

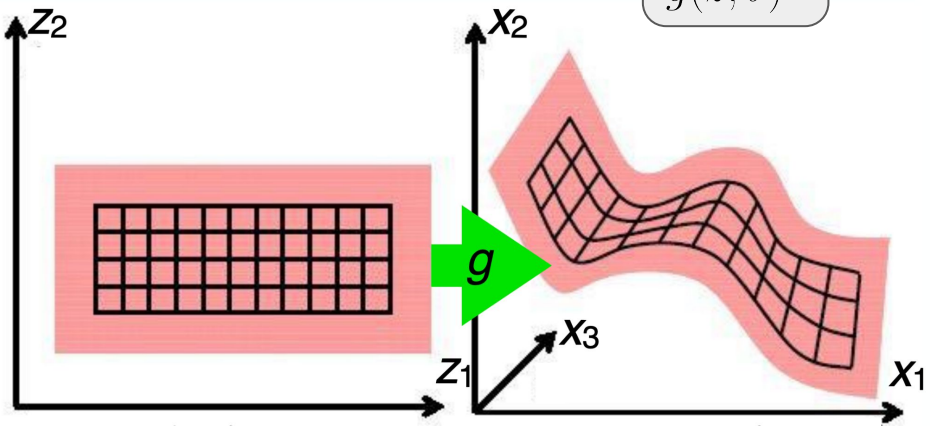
We are interested in modeling $p(x)$, we would like to use a set of latent variables z to represent factors of variation in the data distribution.

$$p(x) = \int p(x, z)dz = \int p(x|z)p(z)dz$$

maximize the marginal likelihood of x

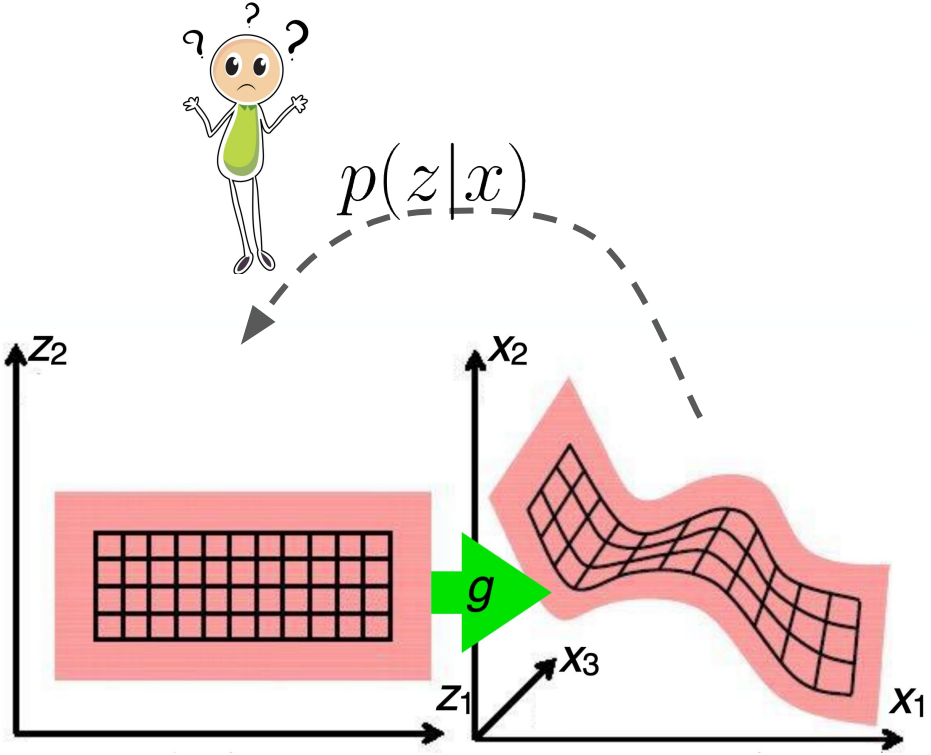
likelihood $g(z; \theta)$

the **prior** over the latent space
simple distribution (e.g. unit gaussian)



Variational Auto-Encoder (VAE) (Kingma et al 2013)

To discover z's we need the posterior $p(z|x)$ which is intractable.



$$p(x) = \int p(x,z)dz = \int p(x|z)p(z)$$

$$p(z|x) = \underbrace{\frac{p(x|z)p(z)}{\int p(x|z)p(z)dz}}$$

very expensive

Variational Auto-Encoder (VAE) (Kingma et al 2013)

To discover z's we need the posterior $p(z|x)$ which is intractable.

- VAE solution is to approximate the posterior with another function $q_\phi(z|x)$

  .

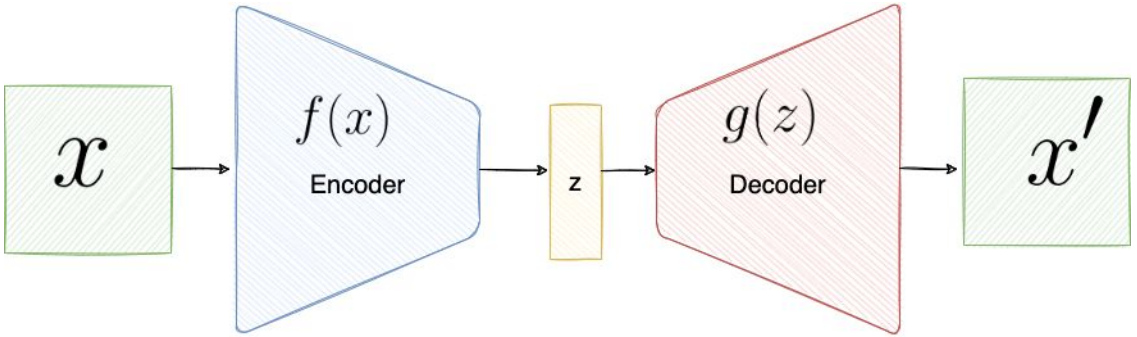
- Which mean use a neural network with parameters ϕ to infer an approximate posterior distribution

- How to train $q_\phi(z|x)$? Based on the math (explained later) we can use **Autoencoders**.

Auto-Encoders

Reconstruct **high dimensional data** passed through a **low dimensional bottleneck**

The model learns interesting features of the data manifold in an **unsupervised** way



High dimensional input image

Low dimensional *bottleneck*

High dimensional reconstructed image

Binary Cross Entropy

$$\mathcal{L} = -\frac{1}{K}\sum_k (x_k \log(x'_k) + (1 - x_k)\log(1 - x'_k))$$

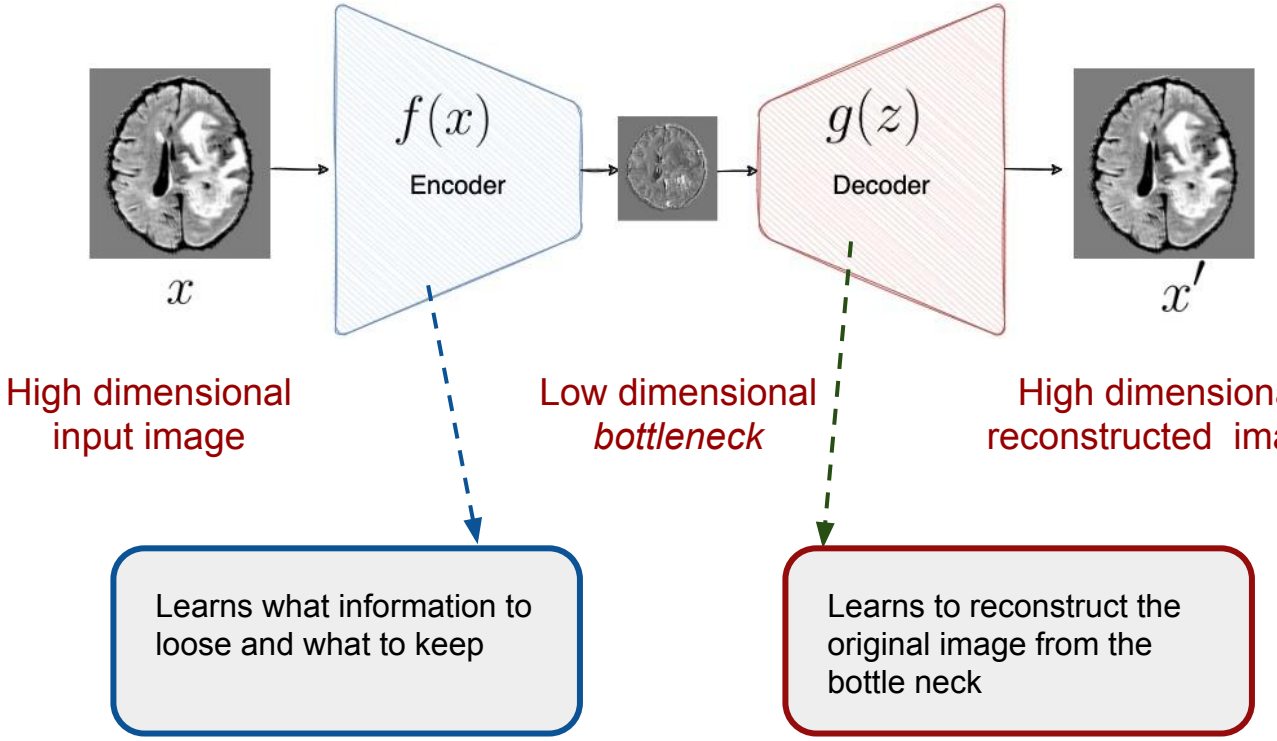
Mean Squared Error

$$\mathcal{L} = -\frac{1}{K}\sum_k (x_k - x'_k)^2$$

Auto-Encoders

Encoder and Decoder learn **input agnostic** information (common across dataset)

The bottleneck represents **input specific** information (specific to data point x)

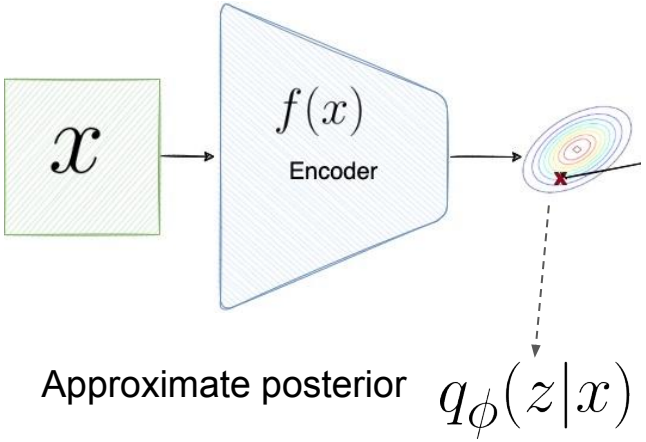


VAE architecture

We said that we need to know which z corresponds to which x to formulate $p_\theta(x|z)$

VAE train a function q to approximate the posterior $q_\phi(z|x)$

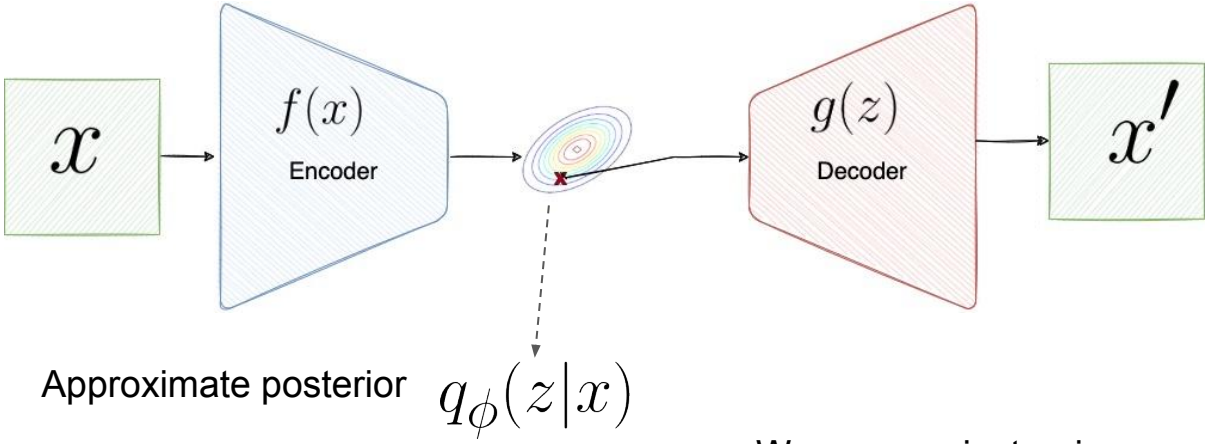
This is to uncover which z, corresponds to which x.



VAE architecture

VAE uses an encoder to construct the distribution $q_\phi(z|x)$ and a decoder for $p_\theta(x|z)$.

Instead of inferring the code **z,** like AE does, infer a gaussian distribution.



- We assume isotropic gaussian to parameterize q.
- The dimensionality is the length of the hidden variables

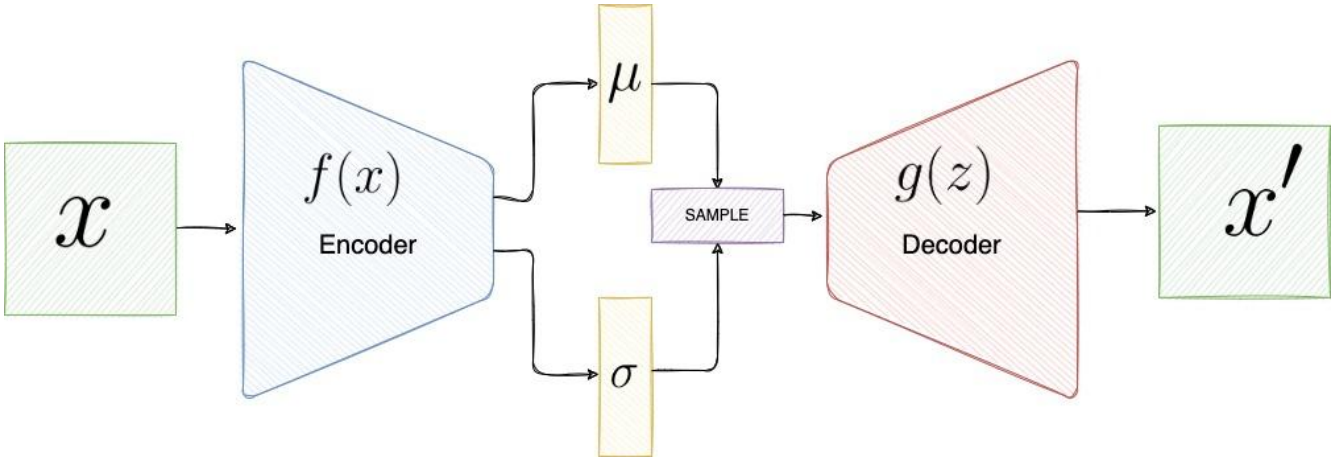
But how does that look like in code?

VAE architecture

VAE uses an encoder to construct the distribution $q_\phi(z|x)$ and decoder for $p_\theta(x|z)$

To parameterize $q_\phi(z|x)$ the encoder outputs two set of vectors for $\mu_z(x), \sigma_z(x)$

$$q_\phi(z|x) = \mathcal{N}(z; \mu_z(x), \sigma_z^2(x))$$

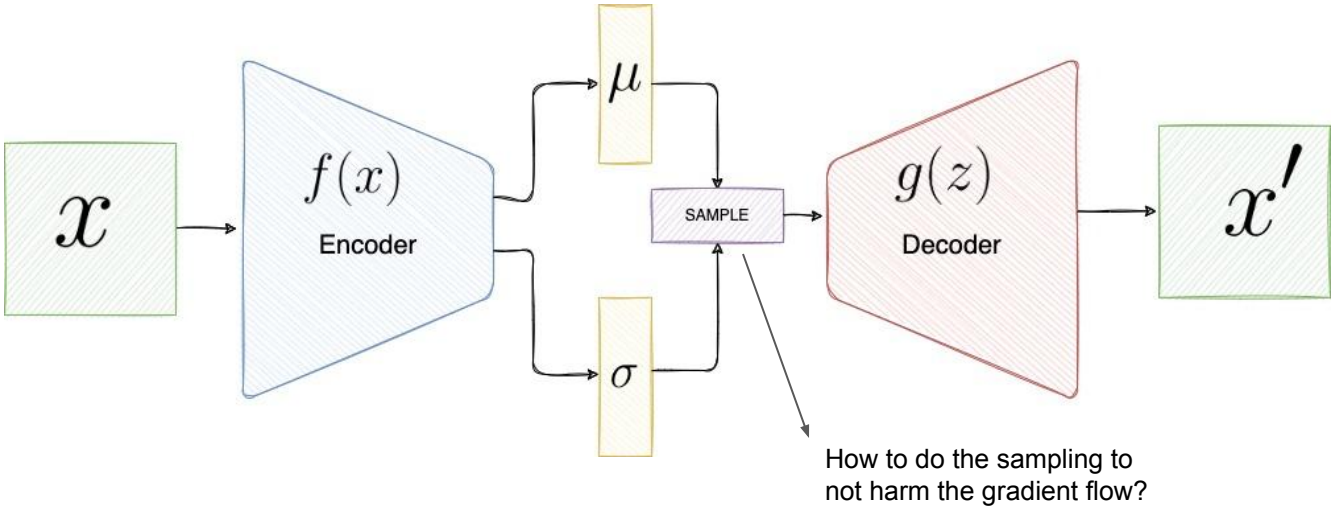


VAE architecture

VAE uses an encoder to construct the distribution $q_\phi(z|x)$ and decoder for $p_\theta(x|z)$

To parameterize $q_\phi(z|x)$ the encoder outputs two set of vectors for $\mu_z(x), \sigma_z(x)$

$$q_\phi(z|x) = \mathcal{N}(z; \mu_z(x), \sigma_z^2(x))$$



VAE architecture

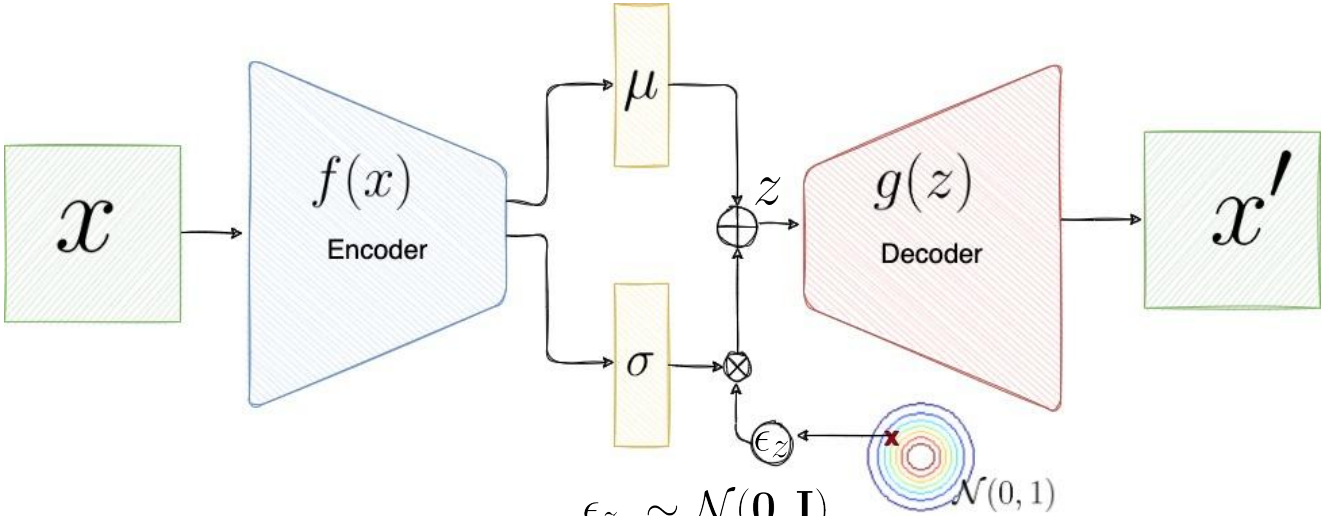
Reparameterization trick

separates the stochasticity from the graph

which makes backpropagation possible

$$\mu_z(x), \sigma_z(x) = f(x)$$
$$\epsilon_z \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$
$$z = \mu_z(x) + \sigma_z(x)\epsilon_z$$

$$q_\phi(z|x) = \mathcal{N}(z; \mu_z(x), \sigma_z(x))$$



$$\epsilon_z \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

The prior over latent space

VAE architecture

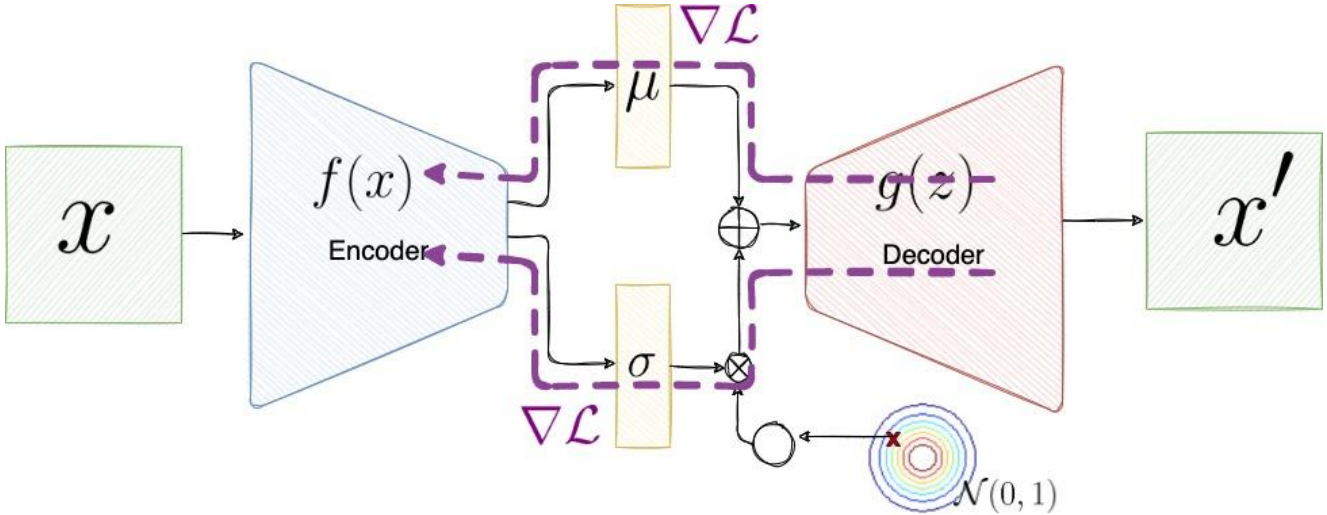
Reparameterization trick

separates the stochasticity from the graph

which makes backpropagation possible

$$\mu_z(x), \sigma_z(x) = f(x)$$
$$\epsilon_z \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$
$$z = \mu_z(x) + \sigma_z(x)\epsilon_z$$

$$q_\phi(z|x) = \mathcal{N}(z; \mu_z(x), \sigma_z(x))$$



VAE architecture

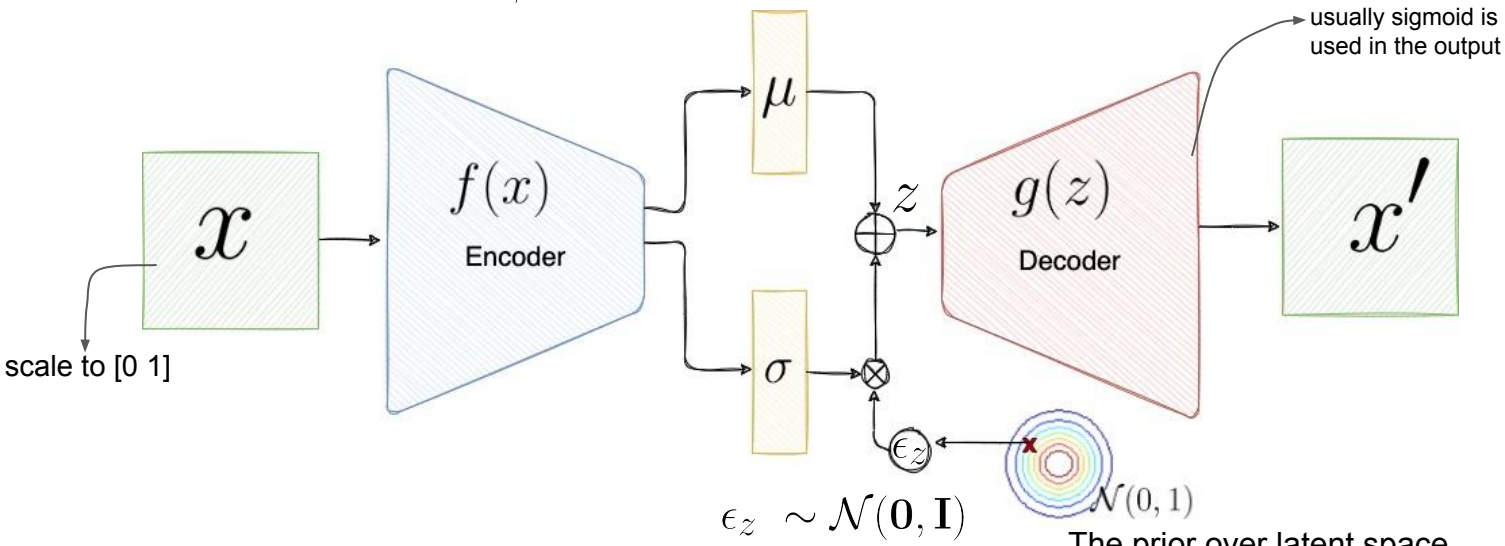
Reparameterization trick

separates the stochasticity from the graph

which makes backpropagation possible

$$\mu_z(x), \sigma_z(x) = f(x)$$
$$\epsilon_z \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$
$$z = \mu_z(x) + \sigma_z(x)\epsilon_z$$

$$q_\phi(z|x) = \mathcal{N}(z; \mu_z(x), \sigma_z(x))$$



$$\epsilon_z \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

The prior over latent space

Variational Auto-Encoder (VAE) (Kingma et al 2013)

To discover z's we need the posterior $p(z|x)$ which is intractable. VAE introduce an inference function $q_\phi(z|x)$ that learn to approximate the true posterior.

$$\log p(x) \geq \mathbb{E}_{q(z|x)}[\log p(x,z) - \log q(z|x)] = \mathcal{L}(x;\theta)$$

Variational Lower bound (ELBO)

$$\mathcal{L}(x;\theta) = \log p(x) - D_{KL}(q(z|x)||p(z|x))$$

$$= -D_{KL}(q_\phi(z|x)||p_\theta(z)) + \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]$$

Regularization

The prior, Normal distribution

Reconstruction

Variational Auto-Encoder (VAE) (Kingma et al 2013)

To discover z's we need the posterior $p(z|x)$ which is intractable. VAE introduce an inference function $q_\phi(z|x)$ that learn to approximate the true posterior.

$$\log p(x) \geq \mathbb{E}_{q(z|x)}[\log p(x,z) - \log q(z|x)] = \mathcal{L}(x;\theta)$$

Variational Lower bound (ELBO)

$$\mathcal{L}(x;\theta) = \log p(x) - D_{KL}(q(z|x)||p(z|x))$$

$$= -D_{KL}(q_\phi(z|x)||p_\theta(z)) + \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]$$

Regularization

KL between two gaussians is easy to compute

The prior, Normal distribution

Reconstruction

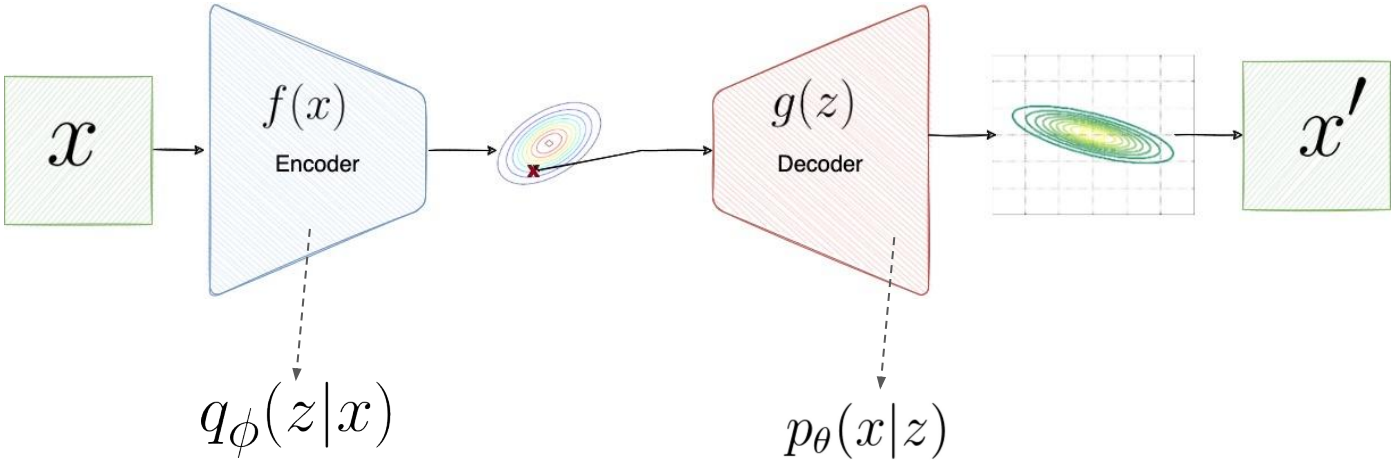
Reduces to MSE (could also use BCE)

VAE architecture

Theoretically the decoder should also output a distribution $p_\theta(x|z)$.

This is done by decoder outputting $\mu_x(z), \sigma_x(z) = g(z)$ and repeating the reparametrization trick in the output as well. $\epsilon_x \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ $\tilde{x} = \mu_x(z) + \sigma_x(z)\epsilon_x$

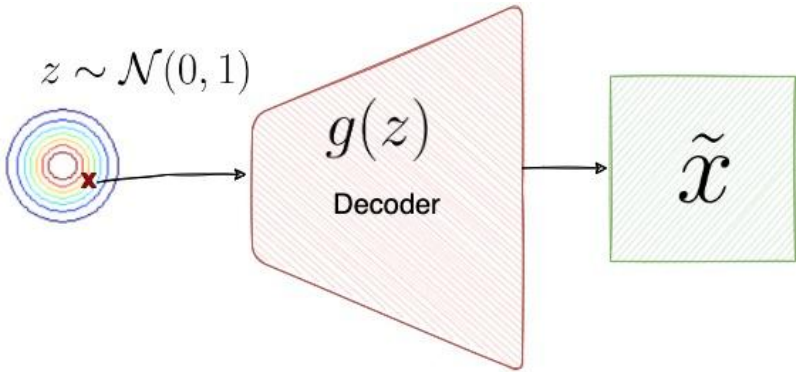
In many implementations decoder directly outputs the reconstructed image $x' = g(z)$



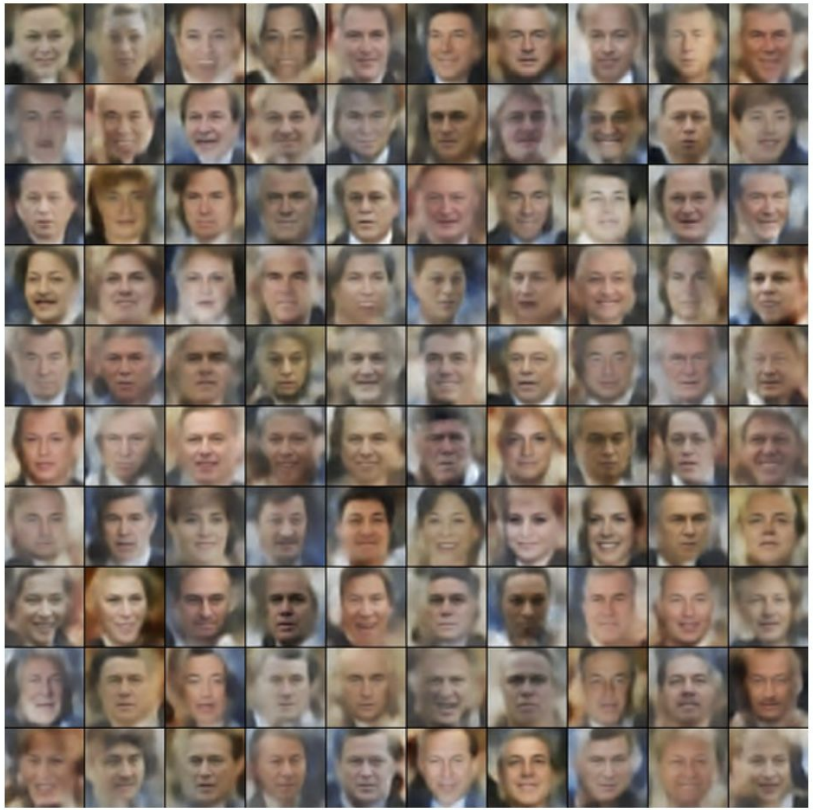
Sampling from VAE

The KL divergence in VAE loss, forces the distribution of the latent space to look like the prior i.e. Normal distribution.

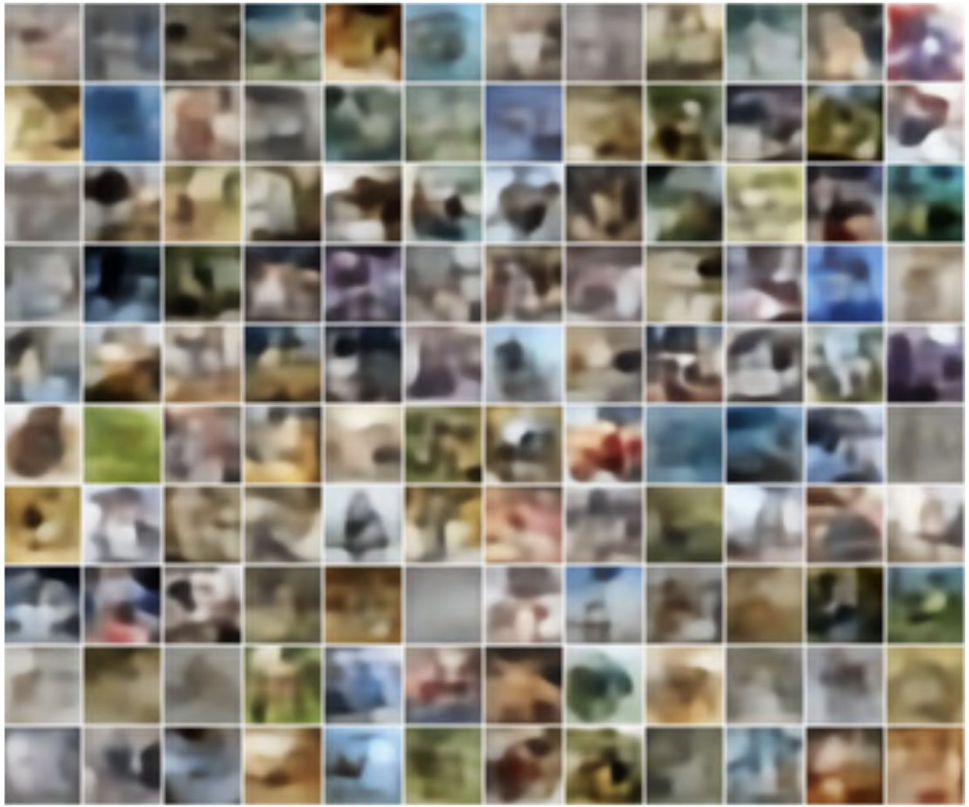
At sampling time we remove the inference machine (encoder) and sample from the prior to the generator $\tilde{x} = g(z)$ where $z \sim \mathcal{N}(0, 1)$



VAE samples



Labelled Faces in the Wild (LFW)



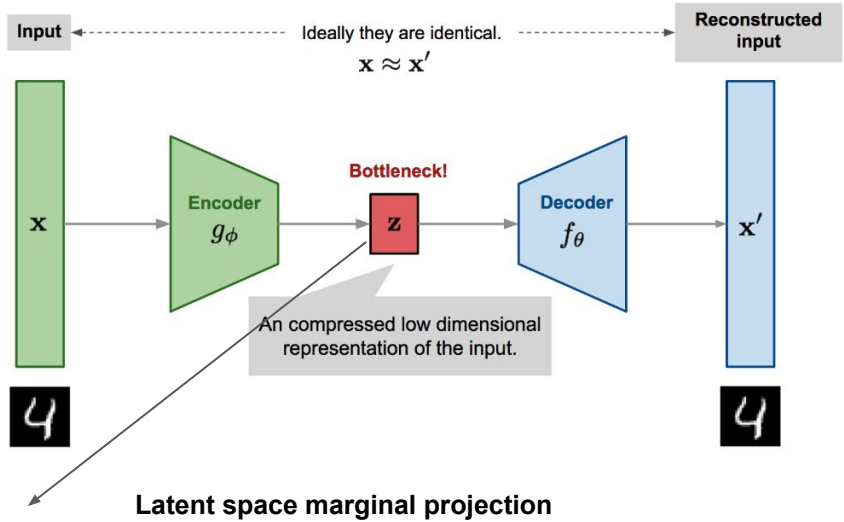
ImageNet (small)

Images from Aaaron courville slides 2017

Latent space in VAE and AE

Autoencoder also creates a compressed code of the input. However, it does not give a posterior distribution

We don't know what the latent space of Auto-encoder looks like. We cannot sample from it.



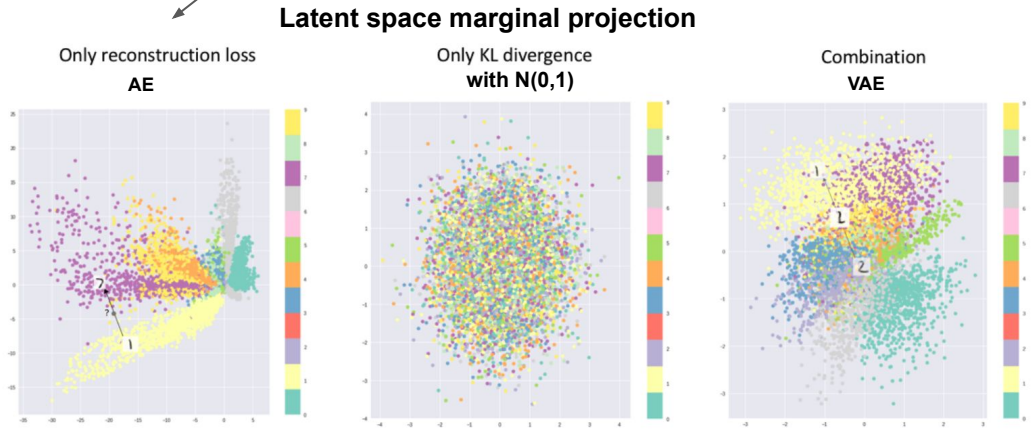


image sources: 1, 2

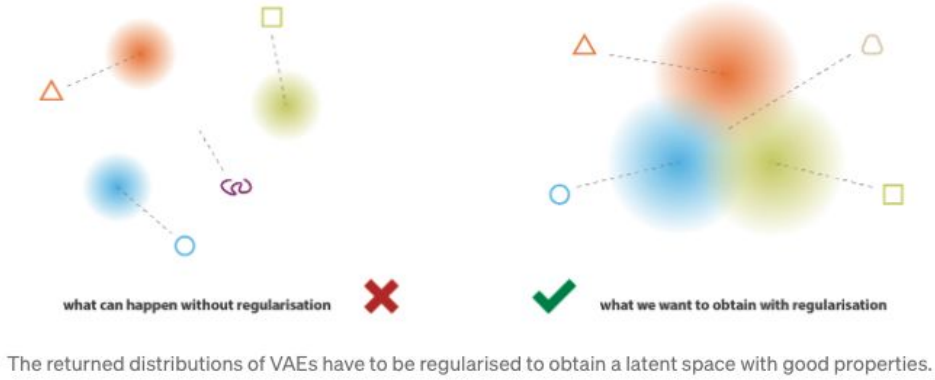
Intuition about regularization

KL regularization encourages smooth latent space (continuous posterior)

Distributions overlap

The more smooth the latent space, less gaps are in the space. The marginal posterior looks more like the prior

This is important because we sample from the prior.



The returned distributions of VAEs have to be regularised to obtain a latent space with good properties.

β-VAE (Higgins et al 2016)

VAE have difficulty learning disentangled factors of variation for complex datasets.

There is a trade-off between the reconstruction error and the regularization

ELBO: $$-D_{KL}(q_\phi(z|x)||p_\theta(z)) + \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]$$

Tries to match q towards a Normal distribution

Tries to match q to a position best suited for reconstruction
(most probably away from Normal distribution)

β-VAE (Higgins et al 2016)

VAE have difficulty learning disentangled factors of variation for complex datasets.

There is a trade-off between the reconstruction error and the regularization

ELBO: $-D_{KL}(q_\phi(z|x)||p_\theta(z)) + \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]$

Tries to match q towards a Normal distribution

Tries to match q to a position best suited for reconstruction (most probably away from Normal distribution)

beta-VAE tries to enforce disentanglement by making sure KL objective is lower than a threshold epsilon

$$\max_{\phi,\theta} \mathbb{E}_{x\sim\mathbf{D}}\left[\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})]\right] \quad \text{subject to} \quad D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})) < \epsilon$$

$$\mathcal{L}(\theta,\phi;\mathbf{x},\mathbf{z},\beta) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \beta\, D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))$$

β-VAE (Higgins et al 2016)

VAE have difficulty learning disentangled factors of variation for complex datasets.

There is a trade-off between the reconstruction error and the regularization

ELBO: $-D_{KL}(q_\phi(z|x)||p_\theta(z)) + \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)]$

Tries to match q towards a Normal distribution

Tries to match q to a position best suited for reconstruction (most probably away from Normal distribution)

beta-VAE tries to enforce disentanglement by making sure KL objective is lower than a threshold epsilon

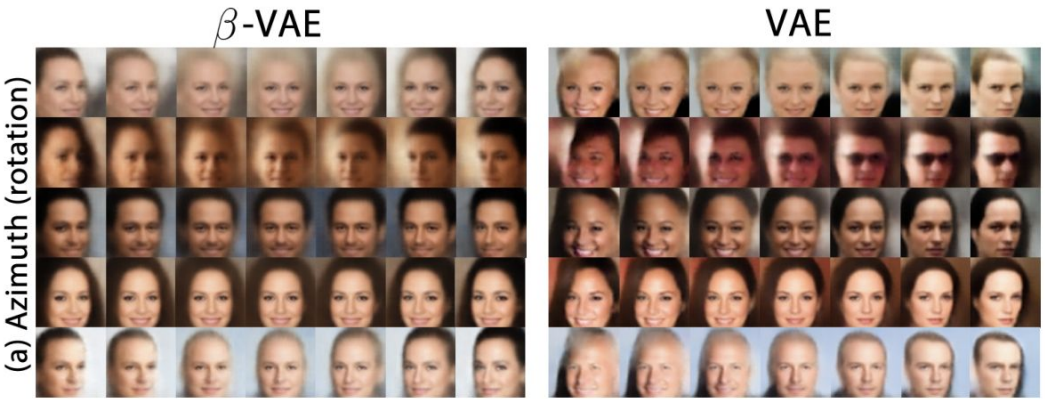
$$\max_{\phi,\theta} \mathbb{E}_{x\sim\mathbf{D}}\left[\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})]\right] \quad \text{subject to} \quad D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})) < \epsilon$$

$$\mathcal{L}(\theta,\phi;\mathbf{x},\mathbf{z},\beta) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{x}|\mathbf{z})] - \beta\, D_{KL}(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z}))$$

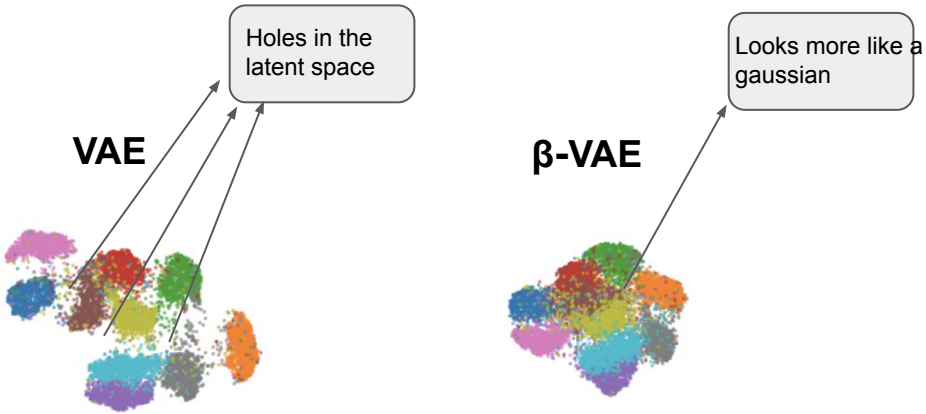
$\beta = 1$ corresponds to original VAE

$\beta > 1$ puts stronger emphasis on KL, improving disentanglement

β-VAE (Higgins et al 2016)



Marginal posterior (z space)



The holes in the vae latent space makes interpolations not very smooth

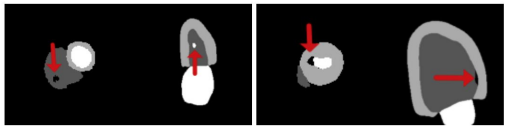
image from link

VAE as Prior over data

(Painchaud et al, Jodoin 2020)

Goal: Anatomical Guarantees for cardiac segmentation

Segmentation models often results in artifacts which violate the anatomical constraints. for example:



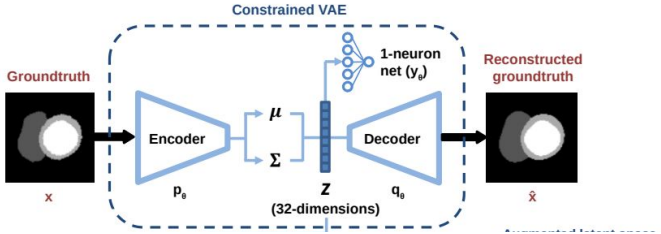
(a) Intra-structure holes (b) Inter-structure holes

(c) Connectivity and LV cavity touching background

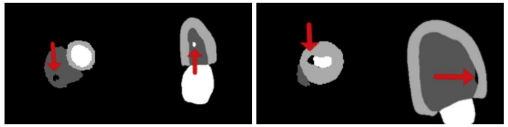
VAE as Prior over data

(Painchaud et al, Jodoin 2020)

Goal: Anatomical Guarantees for cardiac segmentation



Segmentation models often results in artifacts which violate the anatomical constraints. for example:



(a) Intra-structure holes (b) Inter-structure holes

(c) Connectivity and LV cavity touching background

Model Stages:

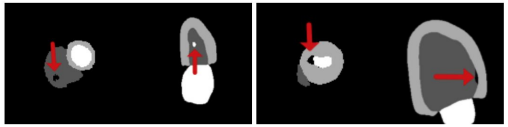
1) Constrained VAE to learn the latent representation of cardiac shapes. constrain the encoder to learn a linear manifold.

VAE as Prior over data

(Painchaud et al, Jodoin 2020)

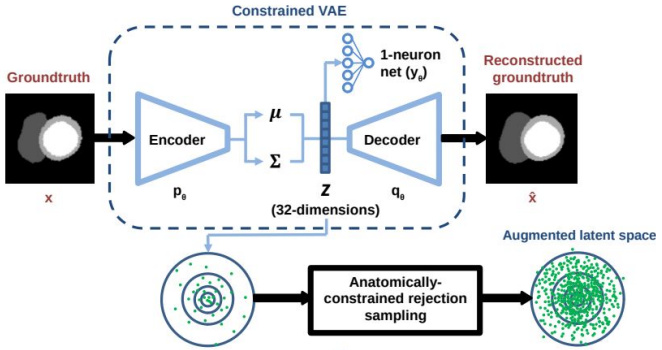
Anatomical Guarantees for cardiac segmentation

Segmentation models often results in artifacts which violate the anatomical constraints. for example:



(a) Intra-structure holes (b) Inter-structure holes

(c) Connectivity and LV cavity touching background



Model Stages:

1) Constrained VAE to learn the latent representation of cardiac shapes. constrain the encoder to learn a linear manifold.

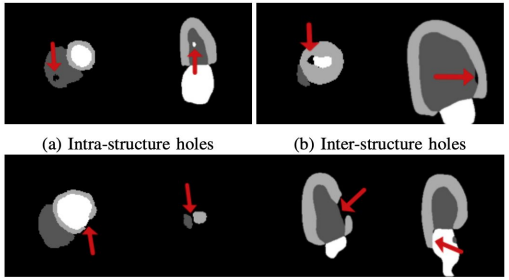
2) an *anatomically-constrained rejection sampling* procedure to augment the number of latent vectors and

VAE as Prior over data

(Painchaud et al, Jodoin 2020)

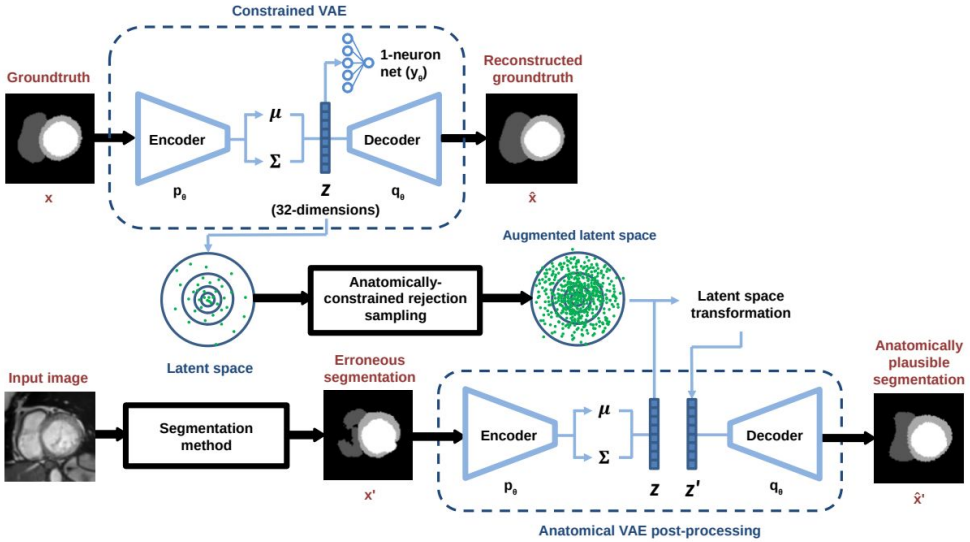
Anatomical Guarantees for cardiac segmentation

Segmentation models often results in artifacts which violate the anatomical constraints. for example:



(a) Intra-structure holes (b) Inter-structure holes

(c) Connectivity and LV cavity touching background



Model Stages:

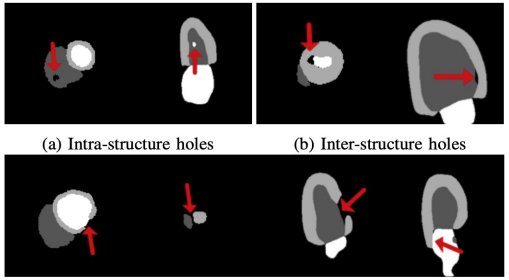
1) Constrained VAE to learn the latent representation of cardiac shapes. constrain the encoder to learn a linear manifold.
2) an *anatomically-constrained rejection sampling* procedure to augment the number of latent vectors and
3) a post-processing VAE that warps anatomically invalid shapes toward the closest valid ones.

VAE as Prior over data

(Painchaud et al, Jodoin 2020)

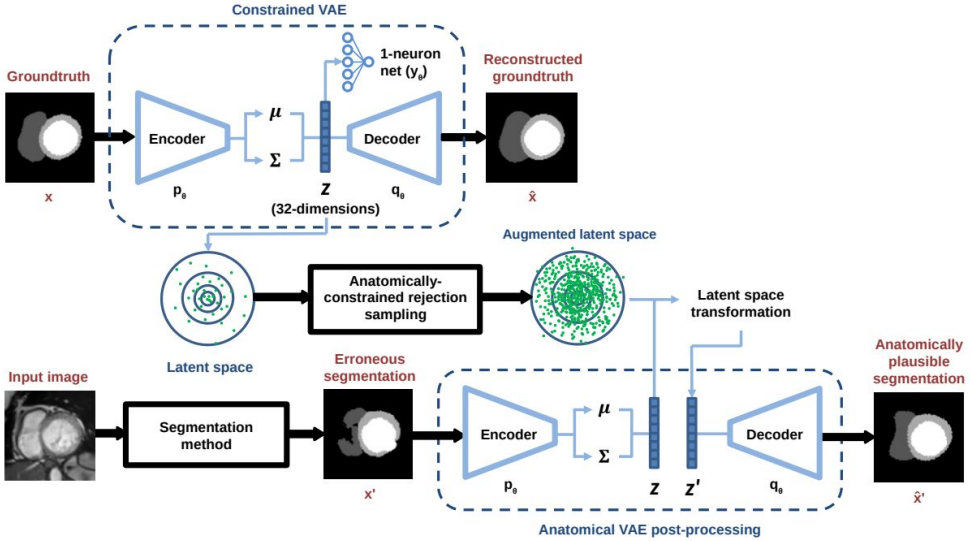
Anatomical Guarantees for cardiac segmentation

Segmentation models often results in artifacts which violate the anatomical constraints. for example:



(a) Intra-structure holes (b) Inter-structure holes

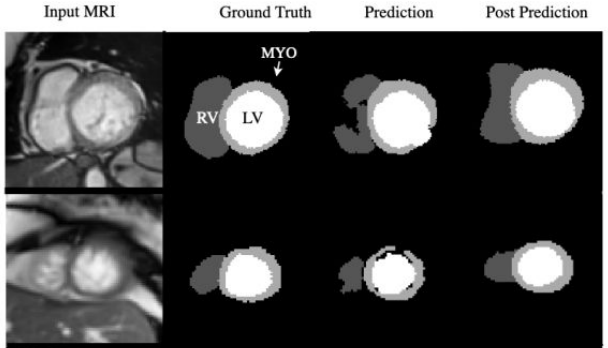
(c) Connectivity and LV cavity touching background



Model Stages:

1) Constrained VAE to learn the latent representation of cardiac shapes. constrain the encoder to learn a linear manifold.
2) an *anatomically-constrained rejection sampling* procedure to augment the number of latent vectors and
3) a post-processing VAE that warps anatomically invalid shapes toward the closest valid ones.

Results



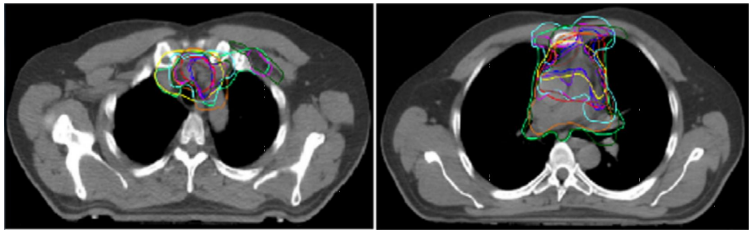
Probabilistic UNet for segmentation (Kohl, et al 2018)

- **Ambiguous labels** : intraobserver and interobserver variability

common Solution is Label overlap (reaching a consensus)

- Its an easy solution
- But Loses the variability and label uncertainty.

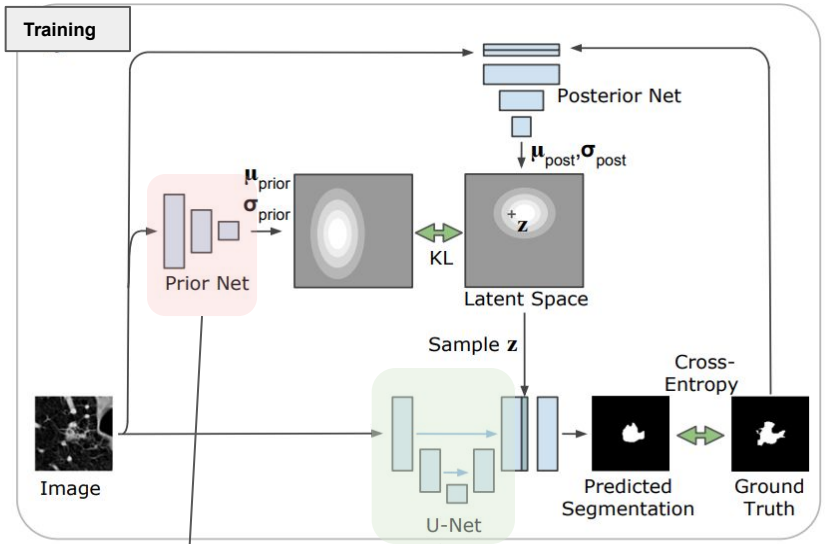
Images Ground truth segmentation by 4 graders (LIDC)



(Reference)

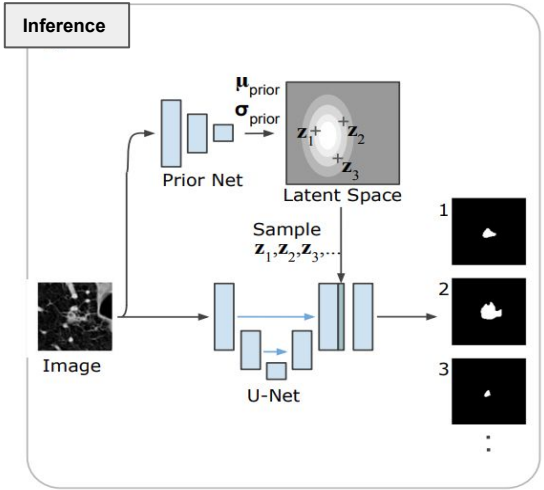
Probabilistic UNet for segmentation (Kohl, et al 2018)

Learn a distribution of plausible outcomes.



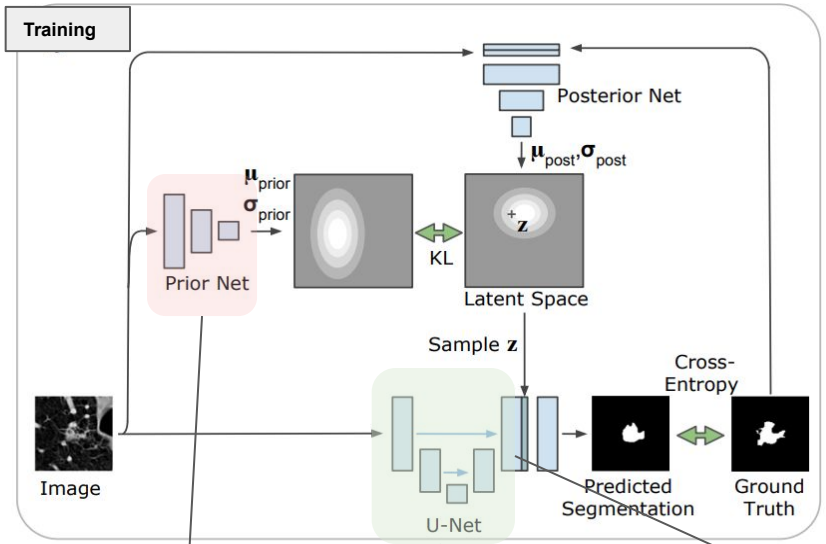
Conditional
prior network

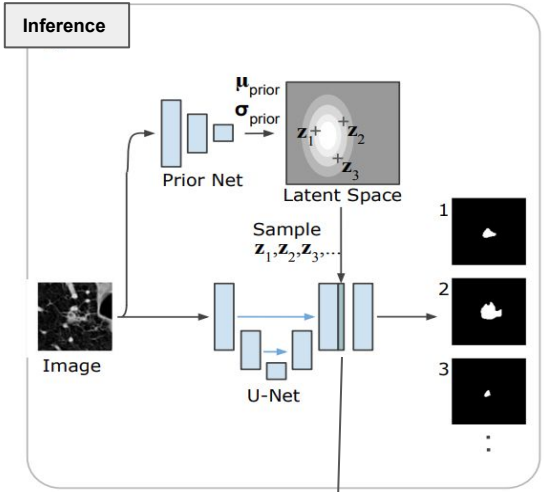
Segmentation module



Probabilistic UNet for segmentation (Kohl, et al 2018)

Learn a distribution of plausible outcomes.





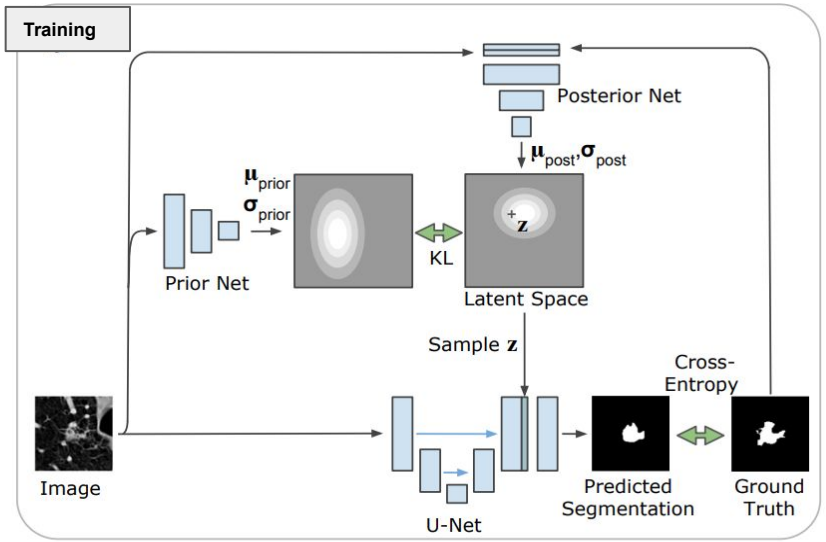
Conditional prior network

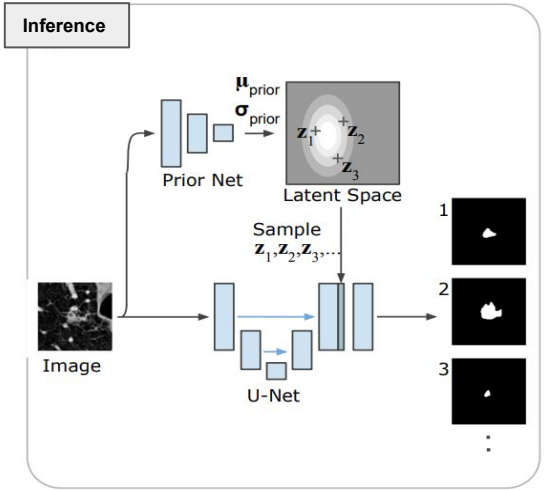
Segmentation module

By sampling from the latent space we add **stochasticity** to the Unet which results in one to many mappings

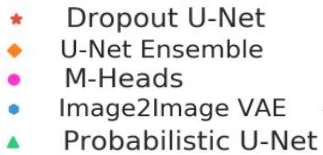
Probabilistic UNet for segmentation (Kohl, et al 2018)

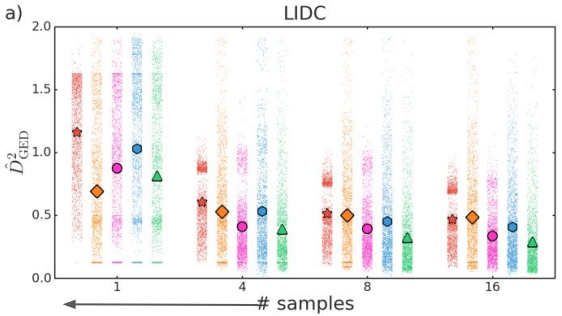
Learn a distribution of plausible outcomes.





Generate a distribution of plausible segmentation results, compare squared energy distance with ground truth



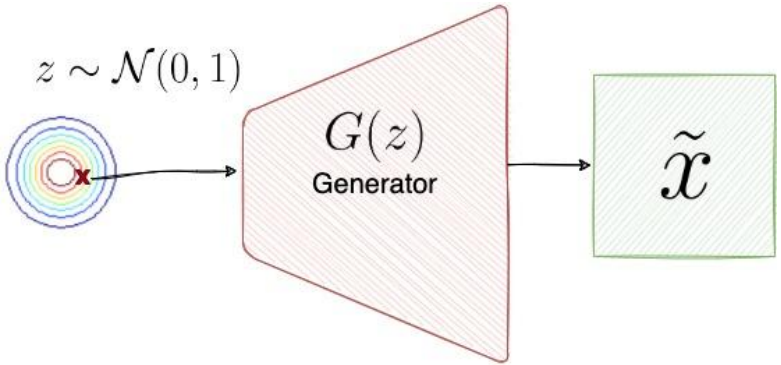


PUNet gets better as MC samples increase

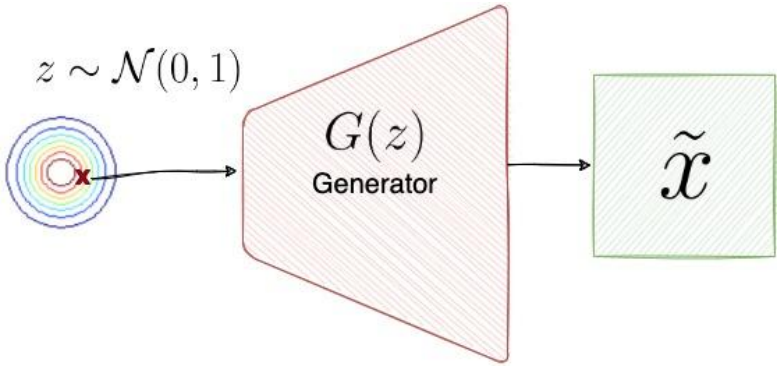
Generative Adversarial Networks (Goodfellow et al 2014)

We saw that VAEs take a probabilistic approach for learning x = g(z) through maximizing a lower bound through an inference machine

GANs learn x = g(z) by not explicitly modeling the density function. They learn a transformation function that assigns samples from a simple noise distribution to samples of the data distribution.



Generative Adversarial Networks (Goodfellow et al 2014)



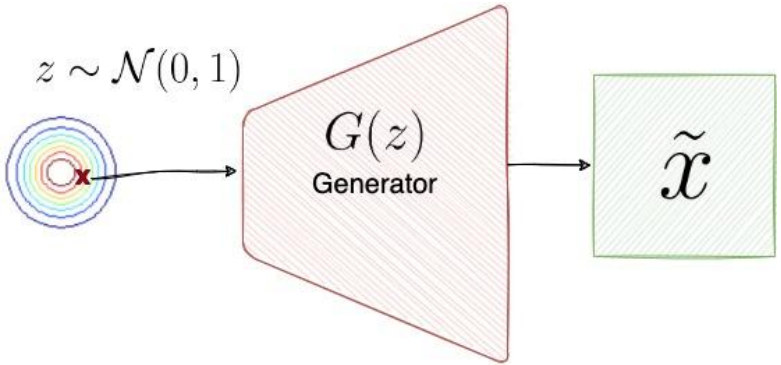
How do we train this?

What loss function do we provide?

We don't know how P(X) looks like.

Generative Adversarial Networks (Goodfellow et al 2014)

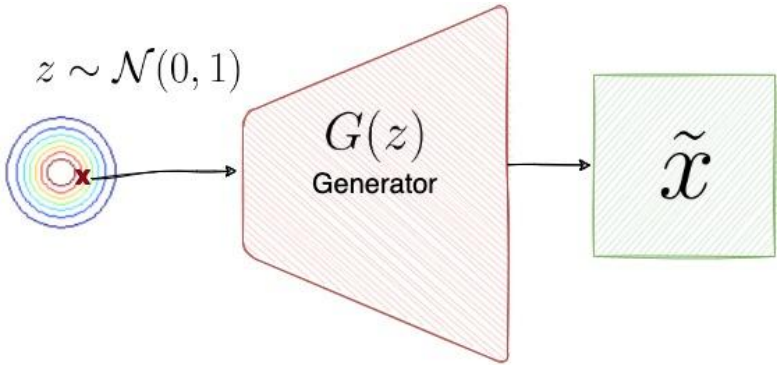
We saw that VAEs take a probabilistic approach for learning x = g(z) through maximizing a lower bound through an inference machine



We know that we want to train the generator such that the generated samples resemble data samples . For example in CelebA:

Generative Adversarial Networks (Goodfellow et al 2014)

We saw that VAEs take a probabilistic approach for learning x = g(z) through maximizing a lower bound through an inference machine



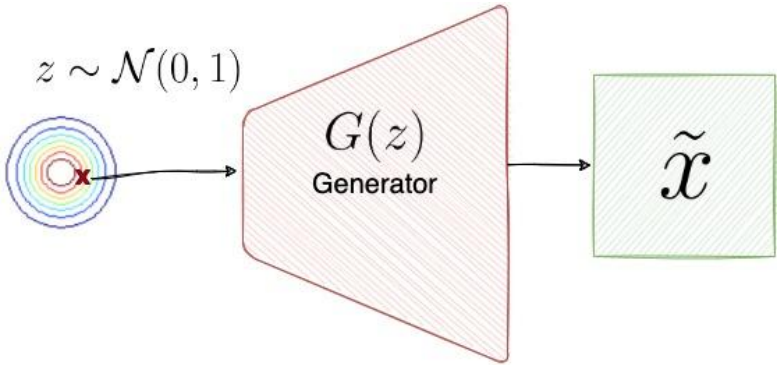
Can we use a neural network to check if our generated images look like real images?

Real

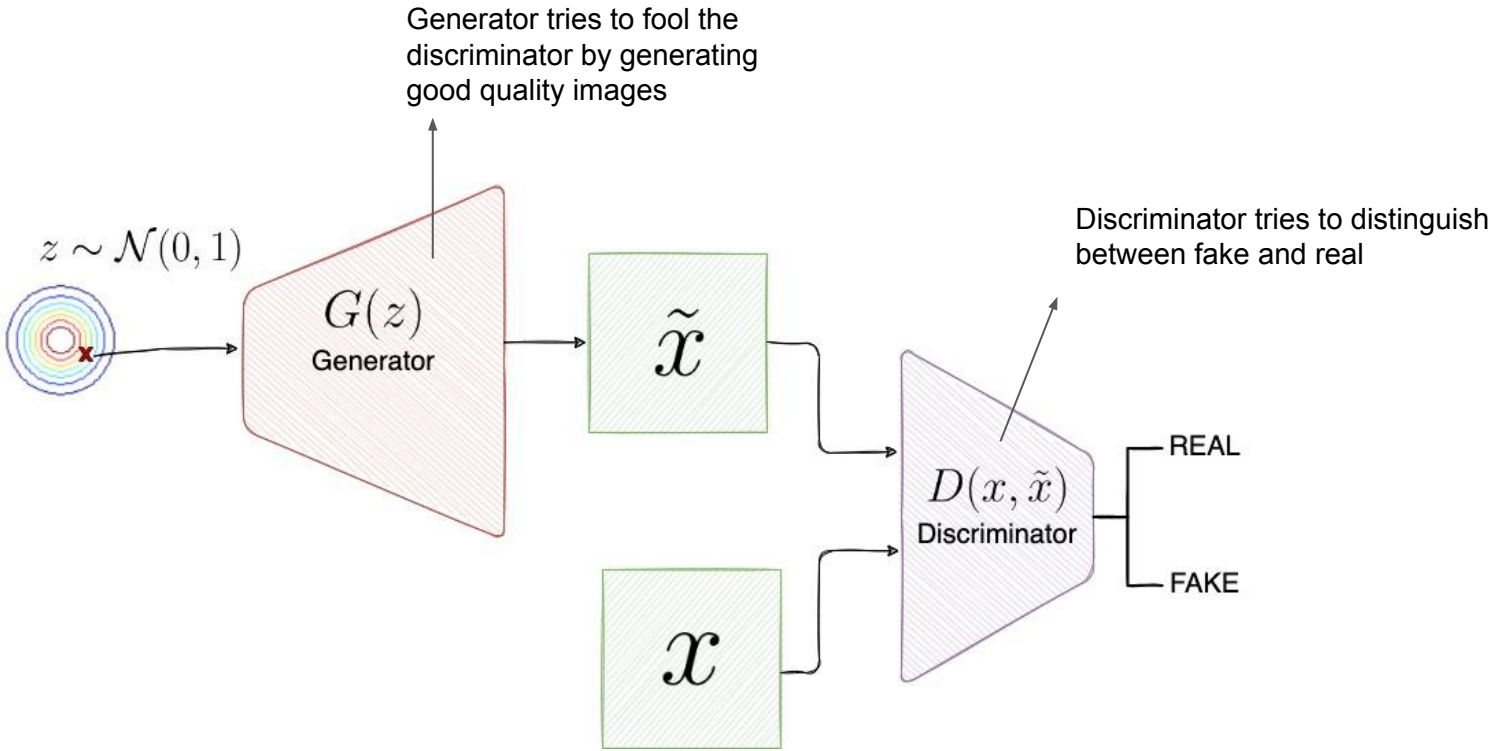
Generated

Generative Adversarial Networks (Goodfellow et al 2014)

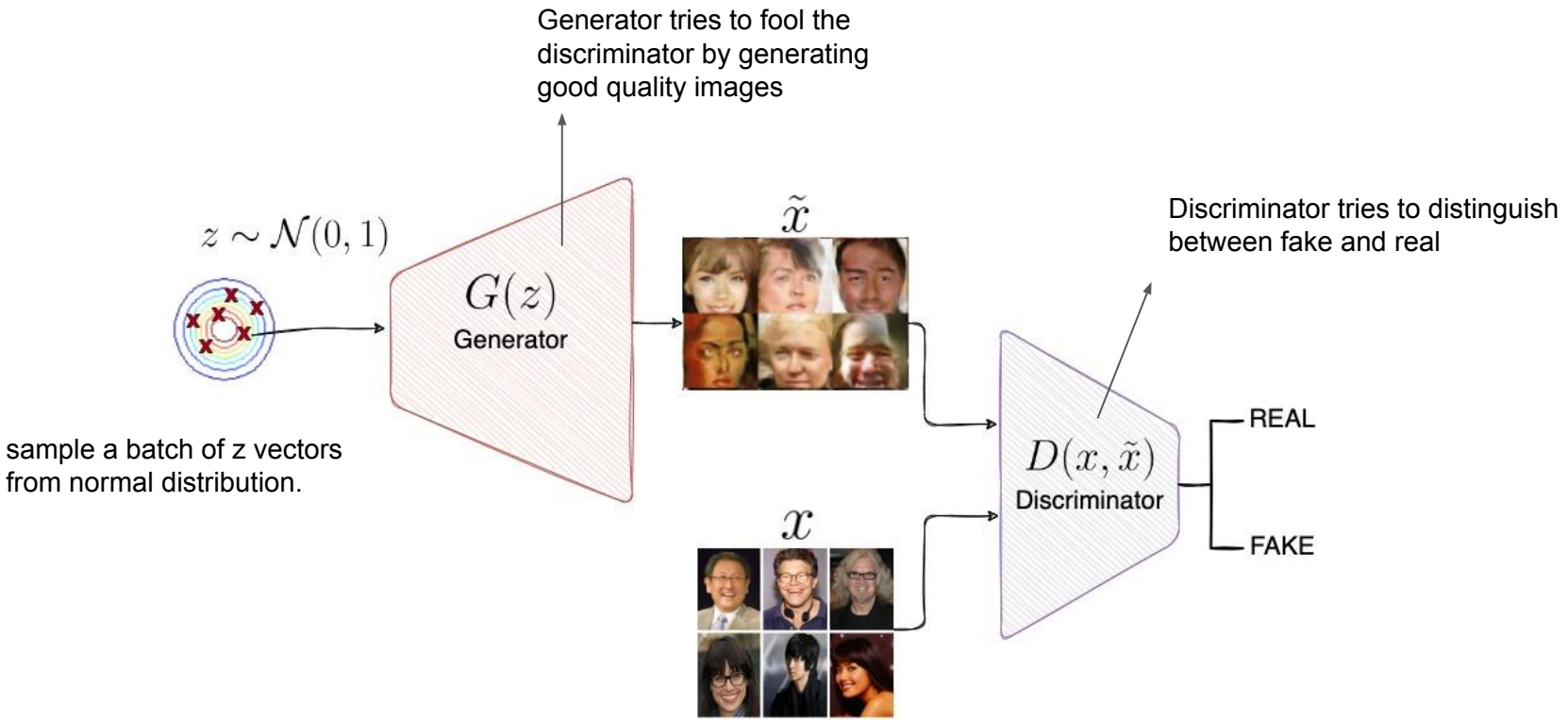
- GANs learn x = g(z) by not explicitly modeling the density function.
- GANs use a second model (discriminator) to see whether the generated samples look like real ones.
- Learn g, through a 2-player game



Generative Adversarial Networks (Goodfellow et al 2014)



Generative Adversarial Networks (Goodfellow et al 2014)



GANs: Adversarial Game

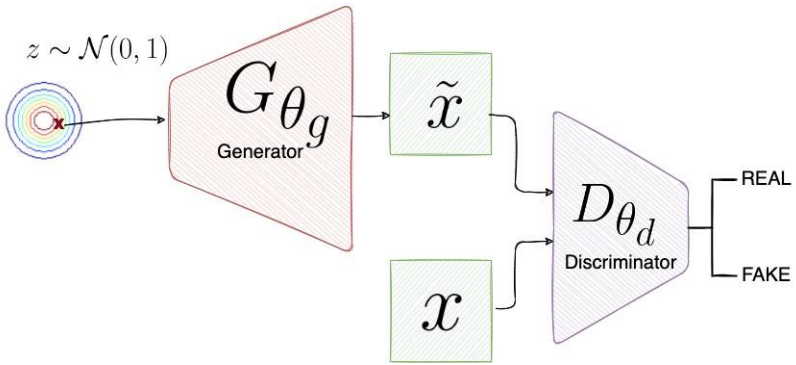
How do we update the model parameters?

Discriminator network: Try to distinguish between fake and real images

Generator network: Try to fool the discriminator by generating real-looking images

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log \underbrace{D_{\theta_d}(x)}_{\text{discriminator tries to assign prob 1 to real data}} + \mathbb{E}_{z \sim p(z)} \log \underbrace{(1 - D_{\theta_d}(G_{\theta_g}(z)))}_{\text{discriminator tries to assign prob 0 generated data}} \right]$$

z: noise vector
G(z) is the generated image



Generative Adversarial Networks

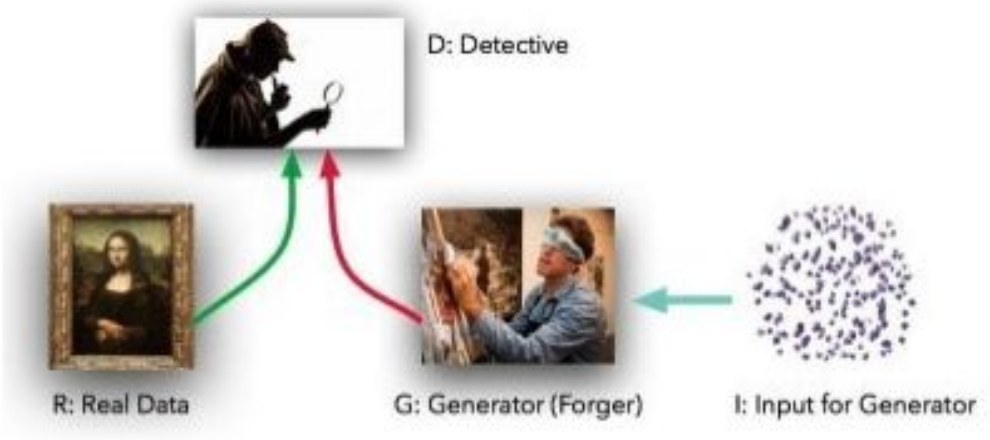
Think of the discriminator as a **detective**.

The generator as a **Forger.**

Detectives job is to distinguish what is **real art** and what is **fake art**.

In order to **fool** the detective, the forger has to become very good at mimicking real art.

- Initially, the detective gets the lead because the forger has no idea what the real art looks like
- As time goes by (longer training), the forger uses the **feedback (gradients)** from the detective to get better and better



https://pt.slideshare.net/ShamaneSiriwardhana/generative-adversarial-networks-slides-auckland-ai-ml-meetup

GANs: Adversarial Game

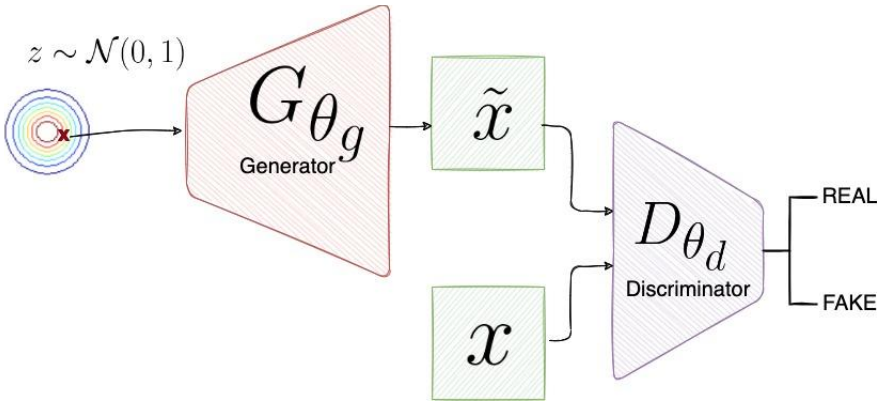
Discriminator network: Try to distinguish between fake and real images

Generator network: Try to fool the discriminator by generating real-looking images

Training Gans:

1. $$\max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

2. $$\min_{\theta_g} \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z)))$$



GANs: Adversarial Game

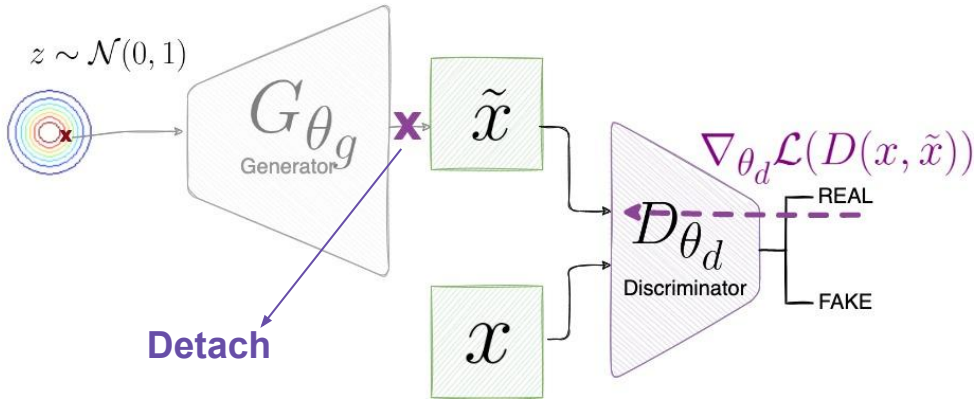
Discriminator network: Try to distinguish between fake and real images

Generator network: Try to fool the discriminator by generating real-looking images

Training Gans:

1. $$\max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$ ⇒ **Update Discriminator**

2. $$\min_{\theta_g} \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z)))$$



GANs: Adversarial Game

Discriminator network: Try to distinguish between fake and real images

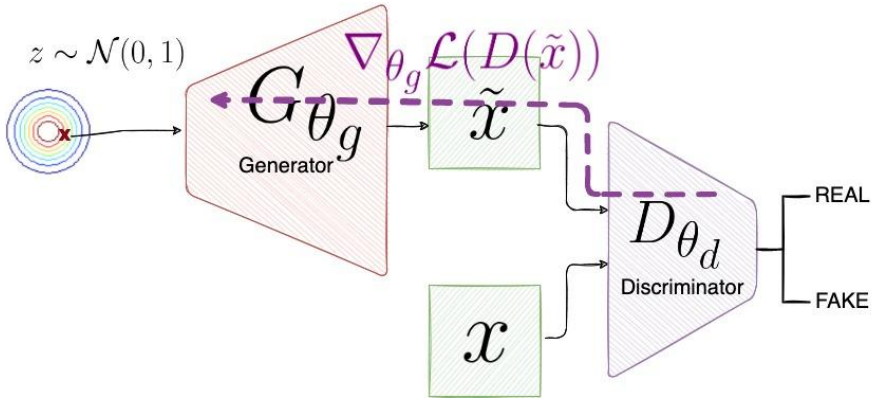
Generator network: Try to fool the discriminator by generating real-looking images

Training Gans:

1. $$\max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

2. $$\min_{\theta_g} \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z)))$$ ⇒ **Update Generator**

we update both discriminator and generator using the signal (losses) we get from the discriminator.



GANs: Adversarial Game

Discriminator network: Try to distinguish between fake and real images

Generator network: Try to fool the discriminator by generating real-looking images

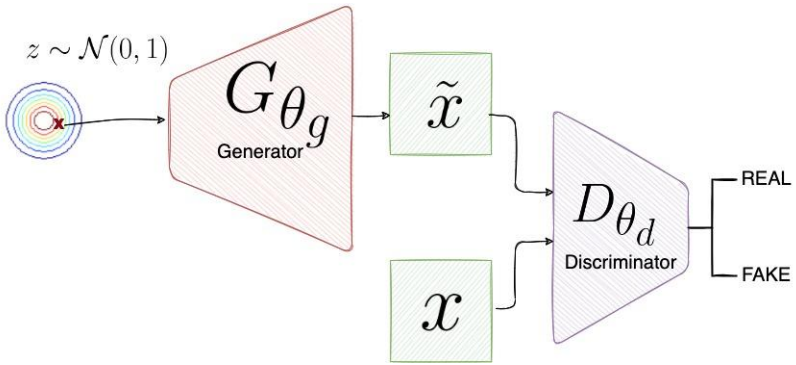
$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log \underbrace{D_{\theta_d}(x)}_{\text{discriminator tries to assign prob 1 to real data}} + \mathbb{E}_{z \sim p(z)} \log \underbrace{(1 - D_{\theta_d}(G_{\theta_g}(z)))}_{\text{discriminator tries to assign prob 0 generated data}} \right]$$

Training Gans:

1. $$\max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

2. $$\min_{\theta_g} \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z)))$$

   $$\max_{\theta_g} \mathbb{E}_{z \sim p(z)} \log(D_{\theta_d}(G_{\theta_g}(z)))$$



Improving GAN training

Issues with GANs:

- Unstability

  The generators fluctuation of gradients. One reason is because of the generators modified objective function see the wasserstein GAN paper (Arjovsky et al 2017) for details.

  Many research is has been trying to address this issue.

Improving GAN training

Issues with GANs:

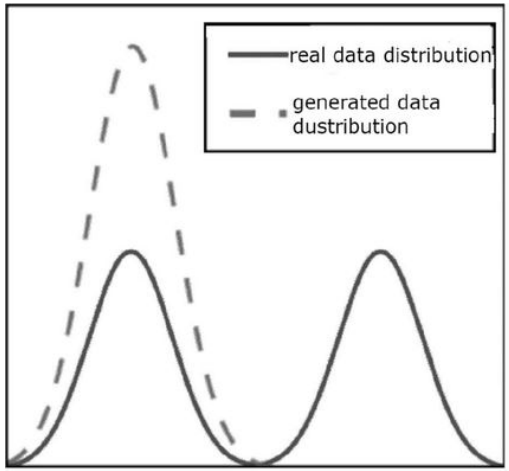
- Mode collapse

G collapses providing limited sample variety

Reason:

- The discriminator is stuck in a local minima. Generator can easily fool discriminator with a single type of image

- This happens usually because of ***vanishing gradients*** due to discriminators objective function



(a)

(b)

Improving GAN training

Issues with GANs:

- Slow training

  Because of the vanishing gradients resulting from GAN objective, the generator could learn extremely slow

Improving GAN training (Radford et al 2016)

- Feature matching : Change the cost function for the generator

$$||\mathbb{E}_{\boldsymbol{x} \sim p_{\text{data}}} \mathbf{f}(\boldsymbol{x}) - \mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{z}}(\boldsymbol{z})} \mathbf{f}(G(\boldsymbol{z}))||_2^2$$

- Minibatch discrimination
- One-sided label smoothing
- Historical averaging
- Using labels (conditional Generation)
- Cost functions (see zoogan)
- Spectral Normalization
- Discriminator and generator capacity control
- Virtual batch Normalization
- Adam optimizer
- scaling the image pixel value between -1 and 1
- use strided conv instead of max pooling

For discussion on each item see this link

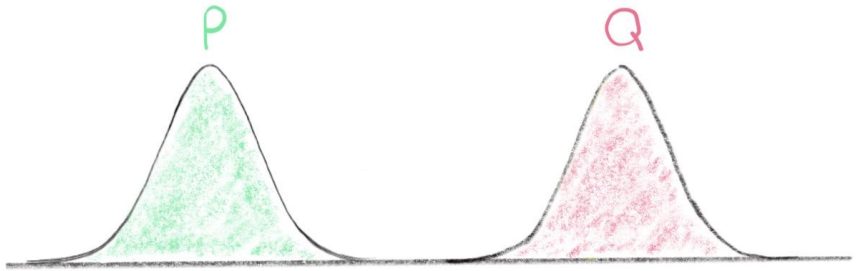
Wasserstein GAN (Arjovsky et al 2017)

We can show that the discriminator in GAN (Goodfellow 2014) is optimizing the Jensen-Shannon divergence:

$$\min_{\theta} JS(p||q_\theta)$$

$$JS(p\|q) = \frac{1}{2} KL(p\|\frac{p+q}{2}) + \frac{1}{2} KL(q\|\frac{p+q}{2})$$

is this a good divergence function for GANs?



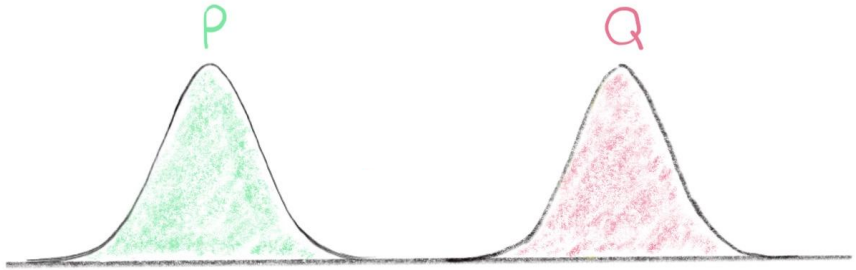
Wasserstein GAN (Arjovsky et al 2017)

Why can't we use KL measure for GAN?

$$\min_{\theta} KL(p||q_\theta)?$$

KL compares the mass of p and q vertically

KL explodes when p and q do not have any overlap.



$$KL(p||q_\theta) = \sum p \log \frac{p}{q}$$

sum over all values

We can only compute KL in areas where q has mass.

If p and q do not have overlap , then

KL = ∞

Wasserstein GAN (Arjovsky et al 2017)

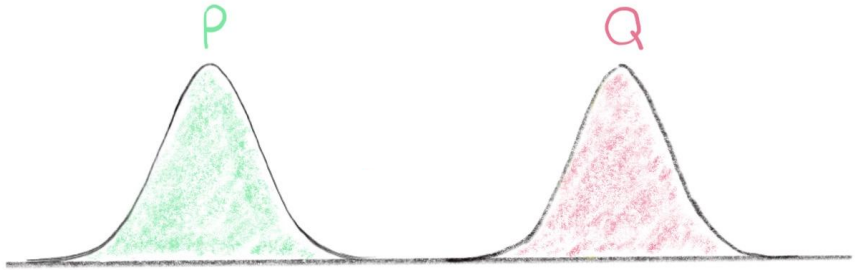
We can show Jensen-Shannon divergence is also a not good measure for GANs.

$$\min_{\theta} \; JS(P_X \| P_\theta)$$

$$JS(p\|q) = \frac{1}{2} KL(p\|\frac{p+q}{2}) + \frac{1}{2} KL(q\|\frac{p+q}{2})$$

$$= \underbrace{\frac{1}{2}\sum p \log 2\frac{p}{p+q} + \frac{1}{2}\sum q \log 2\frac{q}{q+p}}$$

Constant if no overlap

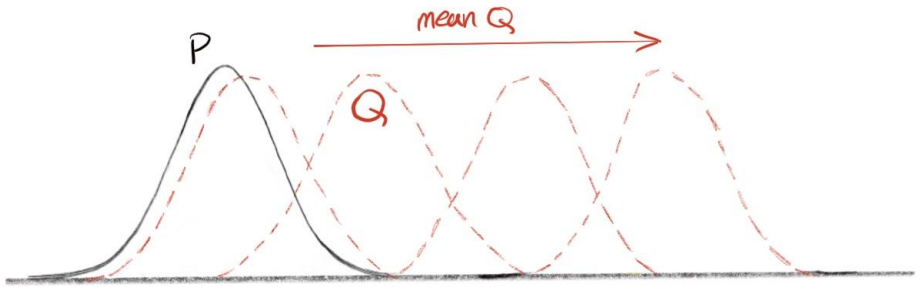


Wasserstein GAN (Arjovsky et al 2017)

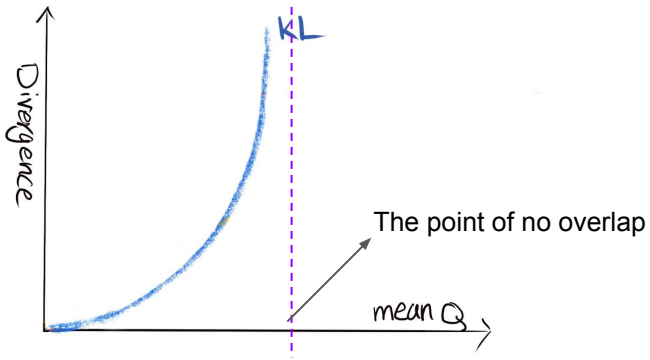
We can show Jensen-Shannon divergence is also a not good measure for GANs.

To see the effect of these divergence measures we do a little experiment

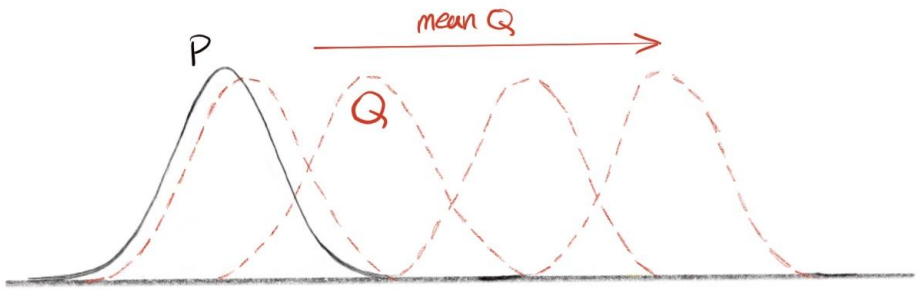
We overlap p and q and gradually increase their distance (by moving mean q) and see how the divergence looks like.



KL goes to infinity when no overlap :(

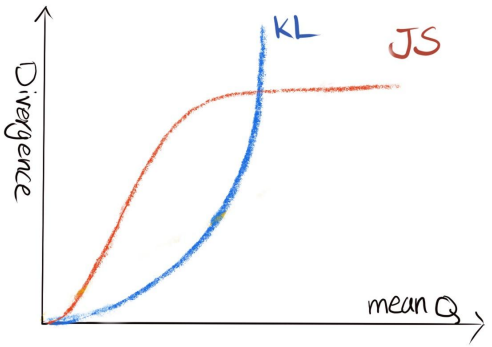


Wasserstein GAN (Arjovsky et al 2017)



JS goes to a constant value when no overlap :(

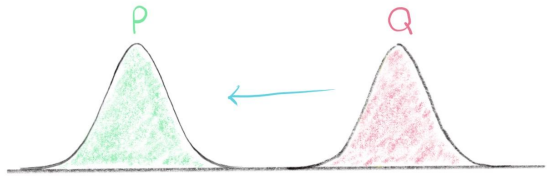
Causes vanishing gradients



Wasserstein GAN

Wasserstein distance: Minimum effort requires to transport mass from Q to P

$$W(p,q) = \inf_{\gamma \sim \prod(p,q)} \mathbb{E}_{(x,y)\sim\gamma}[\|x-y\|]$$

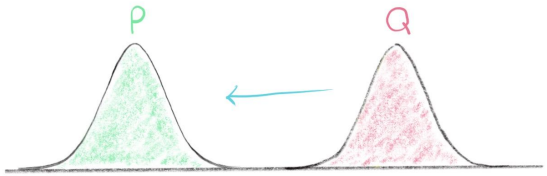


Wasserstein GAN

Wasserstein distance: Minimum effort requires to transport mass from Q to P

$$W(p,q) = \inf_{\gamma \sim \prod(p,q)} \mathbb{E}_{(x,y)\sim\gamma}[\|x-y\|]$$

$$W(p,q) = \frac{1}{K} \sup_{\|f\|_L \leq K} \mathbb{E}_{x\sim p}[f(x)] - \mathbb{E}_{x\sim q}[f(x)]$$ (Dual form)



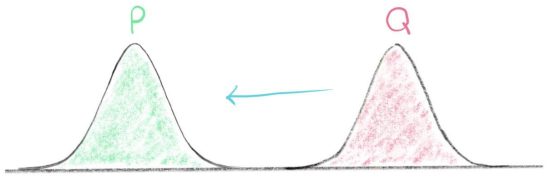
Provided f is K lipschitz

Wasserstein GAN

Wasserstein distance: Minimum effort requires to transport mass from Q to P

$$W(p,q) = \inf_{\gamma \sim \prod(p,q)} \mathbb{E}_{(x,y)\sim\gamma}[\|x-y\|]$$

$$W(p,q) = \frac{1}{K} \sup_{\|f\|_L \leq K} \mathbb{E}_{x\sim p}[f(x)] - \mathbb{E}_{x\sim q}[f(x)] \quad \text{(Dual form)}$$



If p is the data dist. and q is the distribution of generated images:

$$W(p,q) = \max_{w\in W} \mathbb{E}_{x\sim p}[f_w(x)] - \mathbb{E}_{x\sim q}[f_w(g_\theta(z))]$$
$$s.t \quad \|f_w\|_L \leq k$$

↳ **Discriminator**

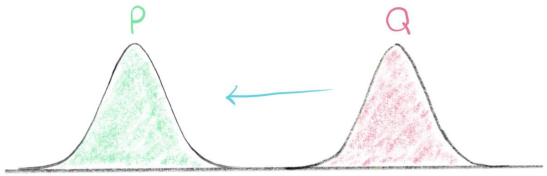
Wasserstein GAN

Wasserstein distance: Minimum effort requires to transport mass from Q to P

If p is the data dist. and q is the distribution of generated images:

$$W(p,q) = \max_{w \in W} \mathbb{E}_{x \sim p}[f_w(x)] - \mathbb{E}_{x \sim q}[f_w(g_\theta(z))]$$
$$s.t \quad \|f_w\|_L \leq k$$

Discriminator



Full GAN objective with Wasserstein distance (WGAN)

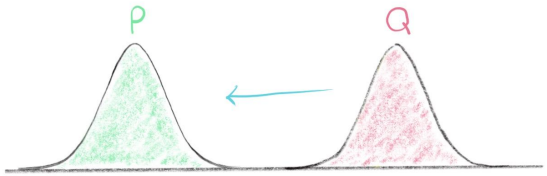
$$V(f,g) = \min_{\theta} \max_{w \in W} \mathbb{E}_{x \sim p_x}[f_w(x)] - \mathbb{E}_{z \sim p_z}[f_w(g_\theta(z))]$$
$$s.t \quad \underbrace{\|f_w\|_L \leq k}$$

We enforce it by weight clipping on discriminator

Wasserstein GAN

Wasserstein distance: Minimum effort requires to transport mass from Q to P

If p is the data dist. and q is the distribution of generated images:



$$W(p,q) = \max_{w \in W} \mathbb{E}_{x \sim p}[f_w(x)] - \mathbb{E}_{x \sim q}[f_w(g_\theta(z))]$$
$$s.t \quad \|f_w\|_L \leq k$$

↳ **Discriminator**

Full GAN objective with Wasserstein distance (WGAN)

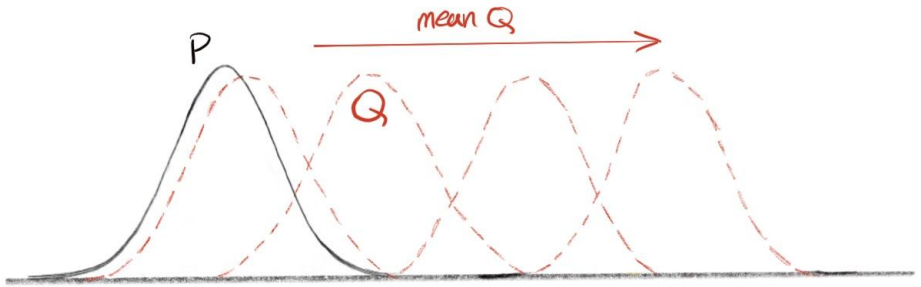
$$V(f,g) = \min_{\theta} \max_{w \in W} \mathbb{E}_{x \sim p_x}[f_w(x)] - \mathbb{E}_{z \sim p_z}[f_w(g_\theta(z))]$$
$$s.t \quad \|f_w\|_L \leq k$$

gradient Vanishing or exploding

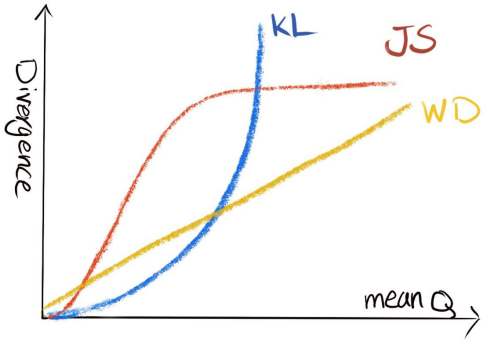
~~weight clipping~~ → gradient penalty

Wasserstein GAN (Arjovsky et al 2017)

KL compares distributions ***vertically*** , wasserstein compares them ***horizontally***.

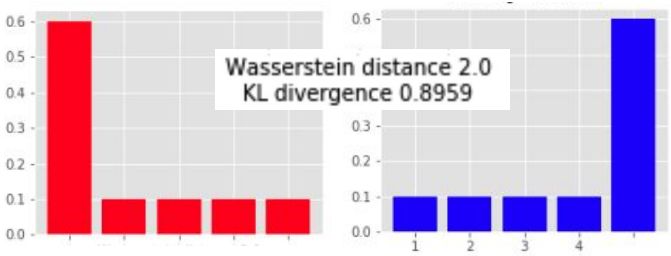


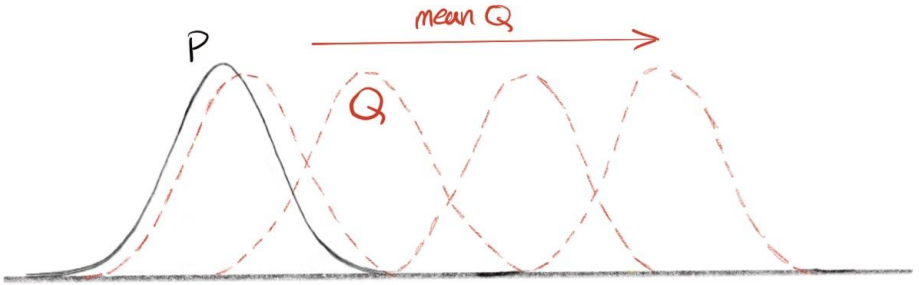
With WD, we have a linearly increasing distance :)

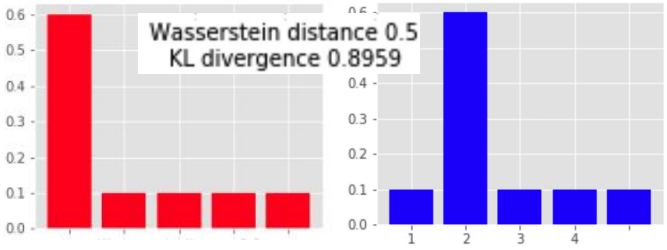


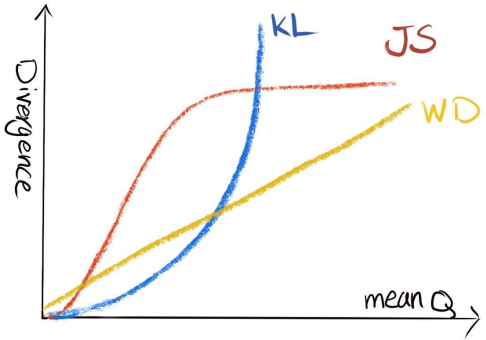
Wasserstein GAN (Arjovsky et al 2017)

KL compares distributions ***vertically*** , wasserstein compares them ***horizontally***.









Adversarial Inference (Dumoulin et al 2016) (Donahue et al 2016)

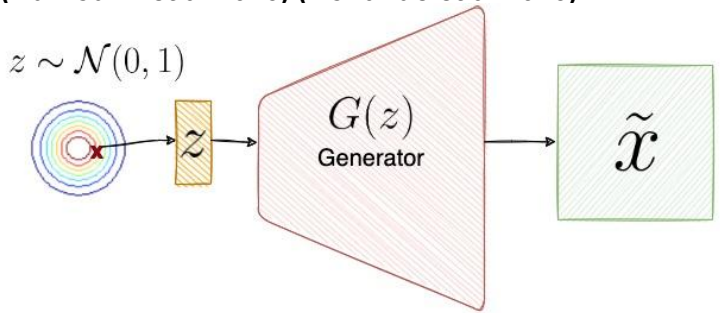
GANs don't provide Inference

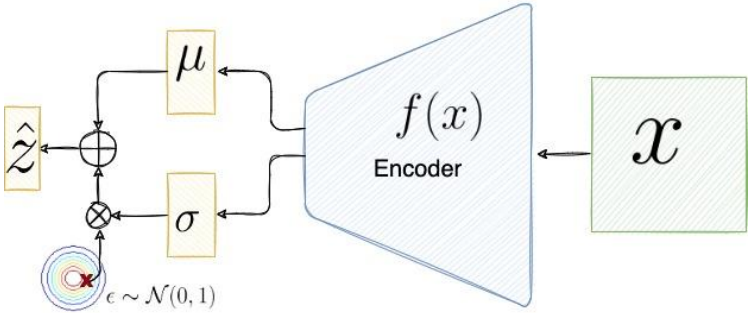
the *encoder* $q(\boldsymbol{x}, \boldsymbol{z}) = q(\boldsymbol{x})q(\boldsymbol{z} \mid \boldsymbol{x})$

the *decoder* $p(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{z})p(\boldsymbol{x} \mid \boldsymbol{z})$

strategy: Match the joint distributions, and then the marginals and conditionals will match. In particular $q(\boldsymbol{z} \mid \boldsymbol{x})$tches $p(\boldsymbol{z} \mid \boldsymbol{x})$

How do we match the joint distributions?





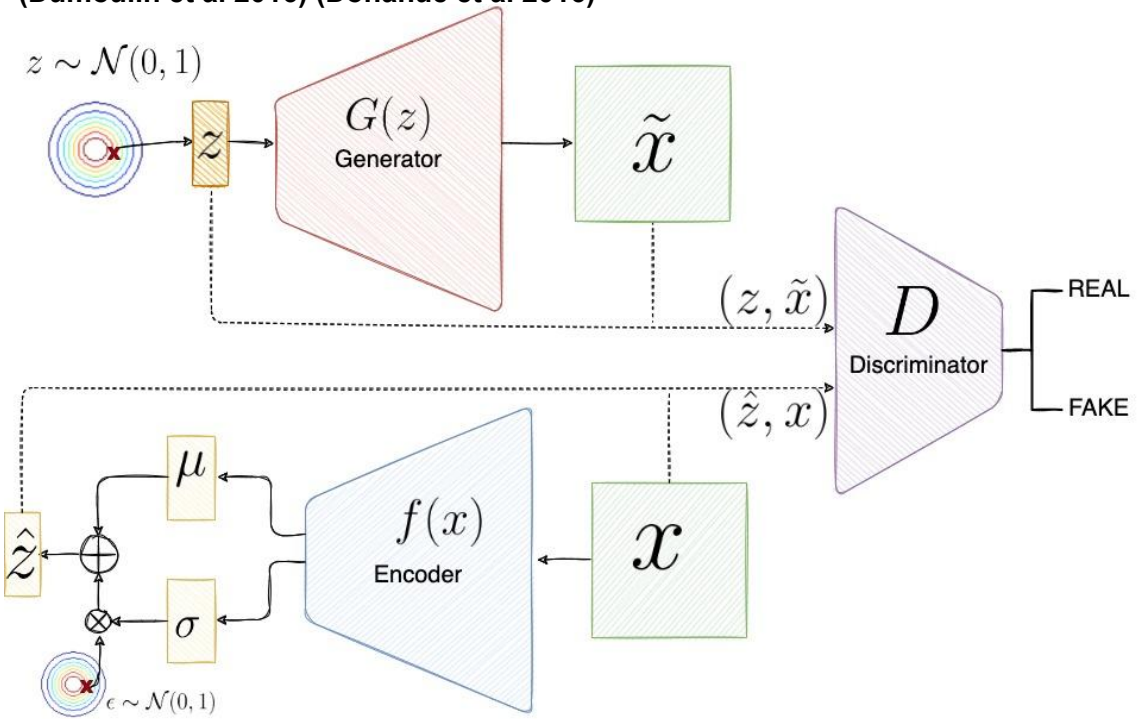
Adversarial Inference (Dumoulin et al 2016) (Donahue et al 2016)

GANs don't provide Inference

the *encoder* $q(\boldsymbol{x}, \boldsymbol{z}) = q(\boldsymbol{x})q(\boldsymbol{z} \mid \boldsymbol{x})$

the *decoder* $p(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{z})p(\boldsymbol{x} \mid \boldsymbol{z})$

strategy: Match the joint distributions, and then the marginals and conditionals will match

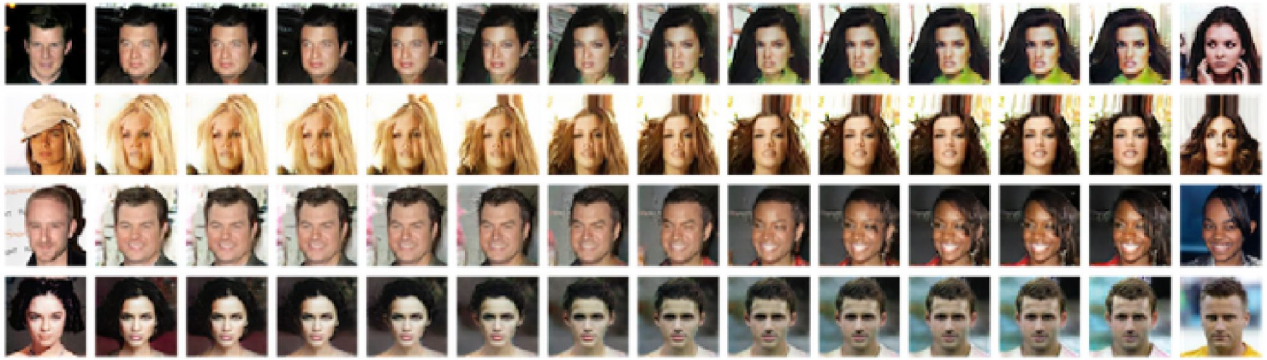


ALI's objective

$$\min_G \max_D V(D, G) = \mathbb{E}_{q(\boldsymbol{x})}[\log(D(\boldsymbol{x}, G_z(\boldsymbol{x})))] + \mathbb{E}_{p(\boldsymbol{z})}[\log(1 - D(G_x(\boldsymbol{z}), \boldsymbol{z}))]$$

ALI samples

(a) CelebA samples.



Latent space interpolations on the CelebA validation set.

image source

image target

Adversarial loss as Implicit loss function

We can categorize loss functions as:

- **Explicit loss functions:** Have a closed mathematical form. e.g. MSE, Cross-Entropy, KL divergence, …

- **GANs provide a procedure to learn a loss function (or similarity function) implicitly**. The discriminator acts as a loss function for the generator.

Adversarial loss as Implicit loss function

We can categorize loss functions as:

- **Explicit loss functions:** Have a closed mathematical form. e.g. MSE, Cross-Entropy, KL divergence, …

- **GANs provide a procedure to learn a loss function (or similarity function) implicitly**. The discriminator acts as a loss function for the generator.

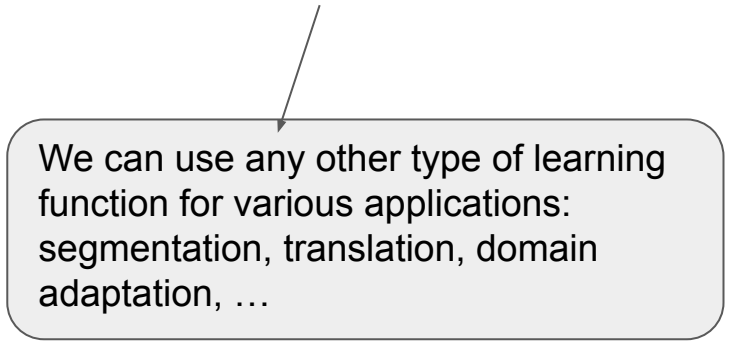
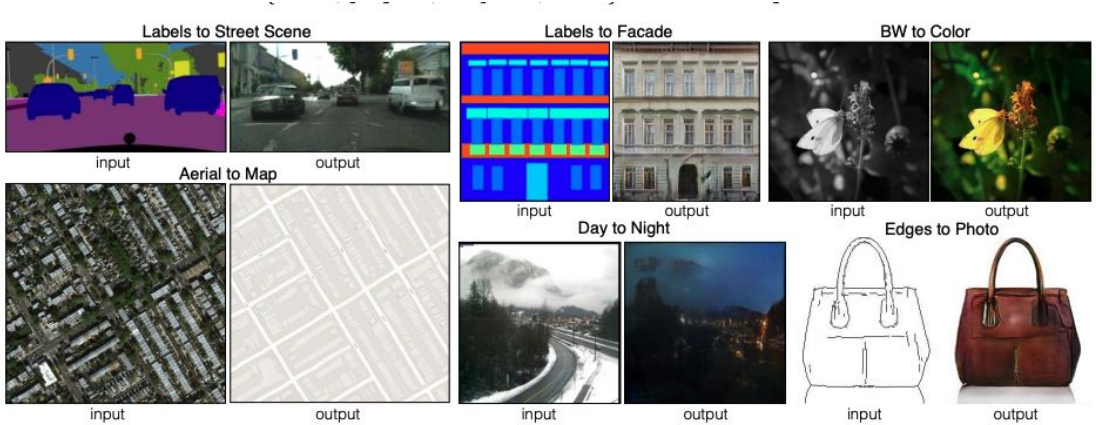


Image translation (Isola et al 2018)

pix2pix

- Paired training data available. x in one domain, y in another domain.
- The discriminator learns a prior over target distribution
- the discriminator observes the pair (x,y) as real and (x, G(x)) as fake
- Important: One to one mapping



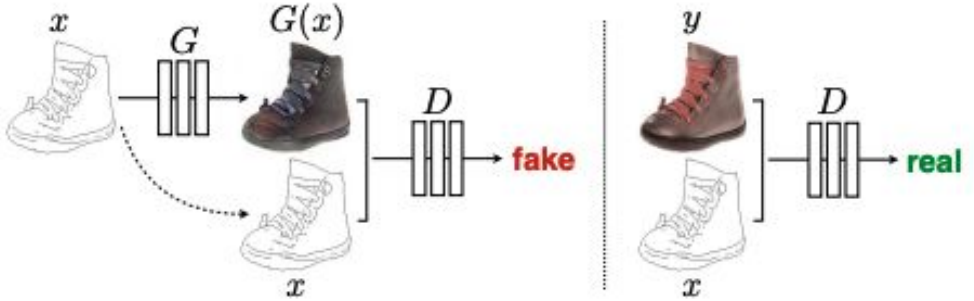
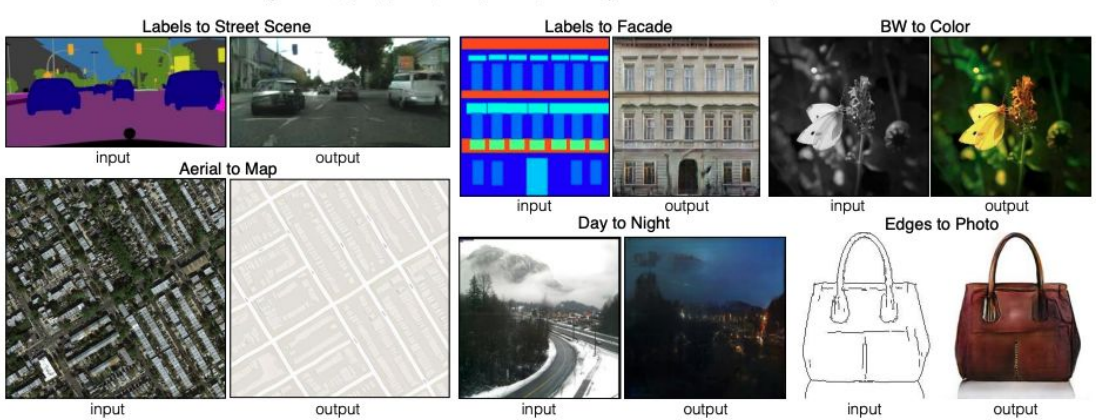


Image translation (Isola et al 2018)

pix2pix

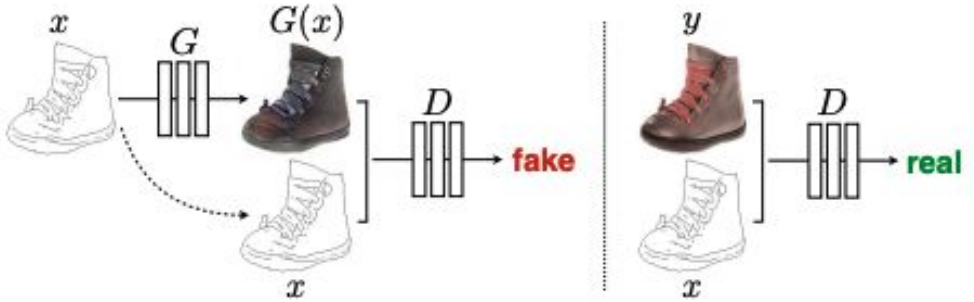
- Paired training data available. x in one domain, y in another domain.
- The discriminator learns a prior over target distribution
- the discriminator observes the pair (x,y) as real and (x, G(x)) as fake
- Important: One to one mapping



$$\mathcal{L}_{cGAN}(G,D) = \mathbb{E}_{x,y}[\log D(x,y)] + \mathbb{E}_{x,z}[\log(1 - D(x, G(x,z))]$$

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x,z)\|_1]$$

$$G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G,D) + \lambda \mathcal{L}_{L1}(G)$$



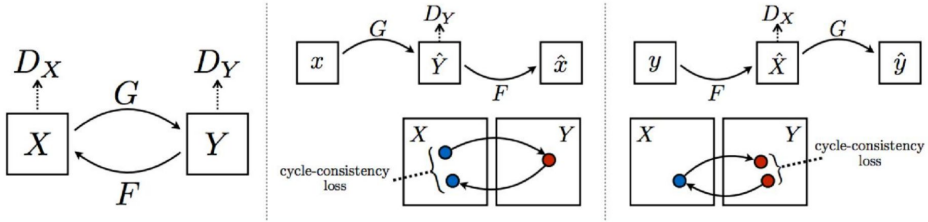
We can view image segmentation as a type of translation and extend this method for segmentation (use Dice loss instead of L1)

CycleGAN (Zhu et al 2017)

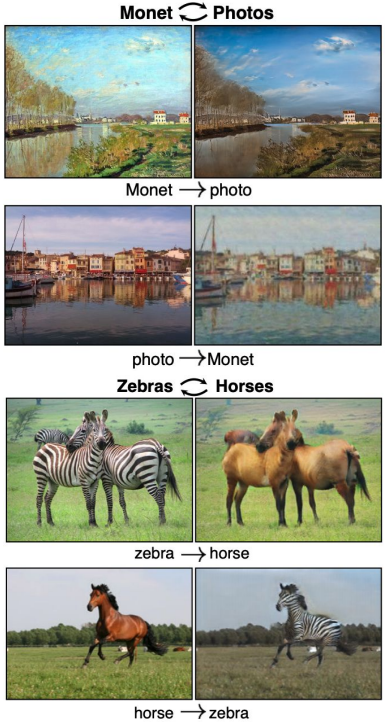
-**What if we dont have any paired data?**

- Learns transformations across domain pairs with unpaired data

-combines GAN loss with **cycle-consistency loss**



Uses two discriminators (Dx and Dy), one for each domain

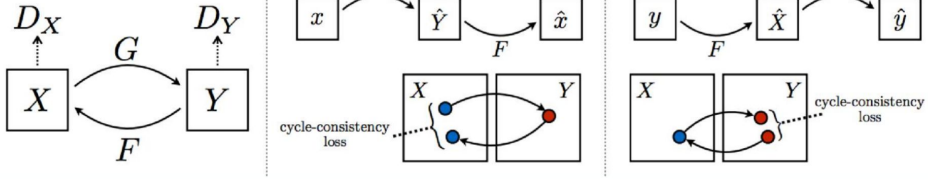


CycleGAN (Zhu et al 2017)

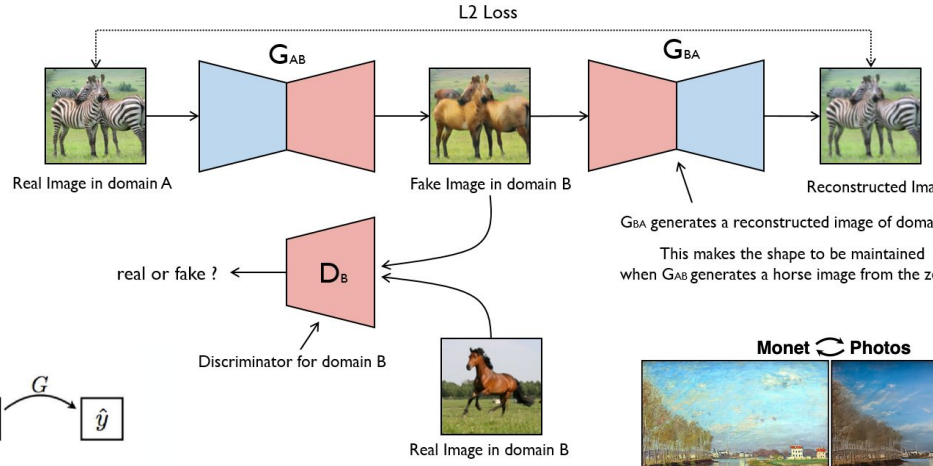
-**What if we dont have any paired data?**

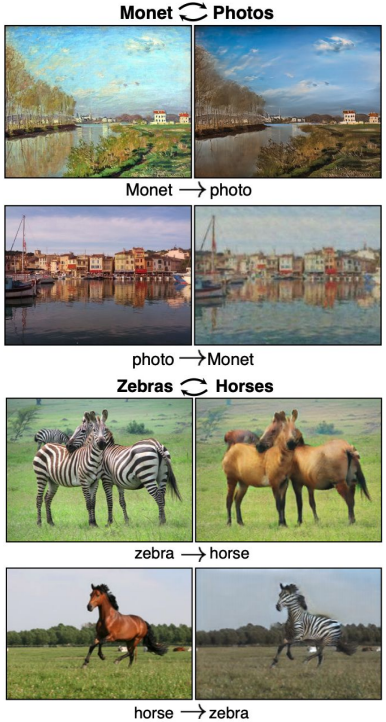
- Learns transformations across domain pairs with unpaired data

-combines GAN loss with **cycle-consistency loss**



Uses two discriminators (Dx and Dy), one for each domain



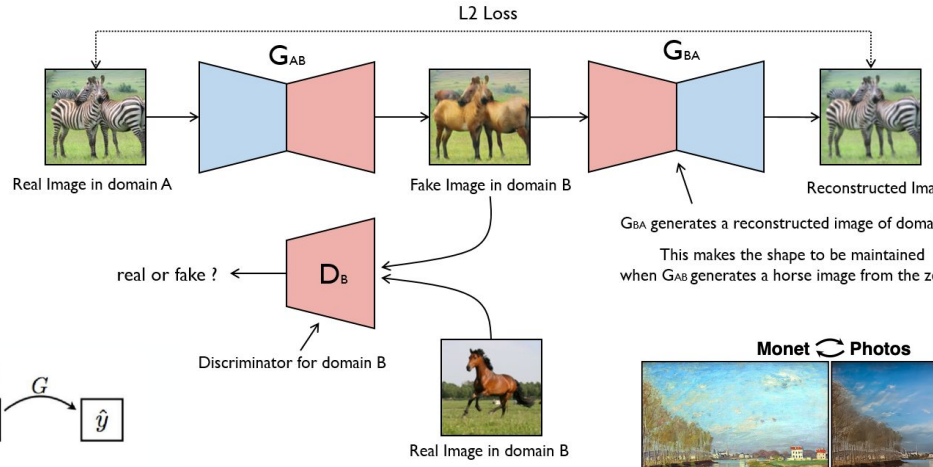


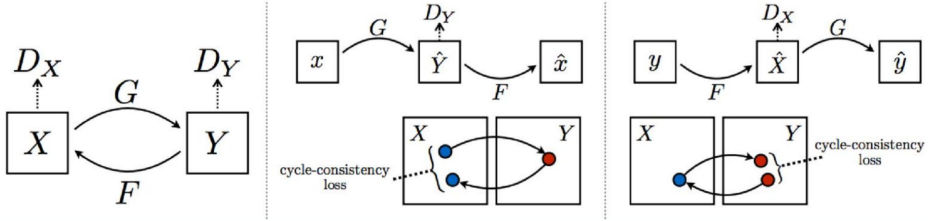
CycleGAN (Zhu et al 2017)

-**What if we dont have any paired data?**

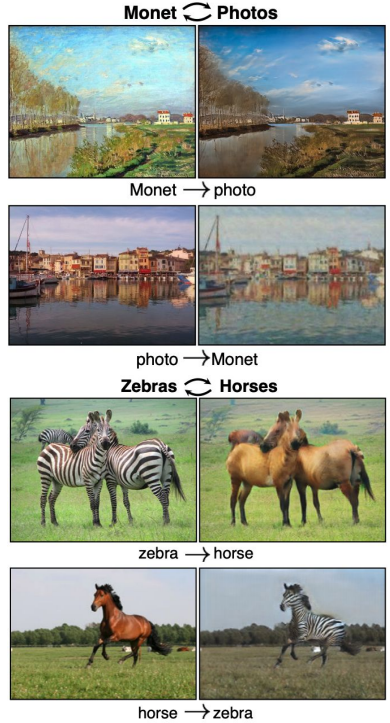
- Learns transformations across domain pairs with unpaired data

-combines GAN loss with **cycle-consistency loss**





$$\begin{aligned}\mathcal{L}(G, F, D_X, D_Y) =& \mathcal{L}_{\text{GAN}}(G, D_Y, X, Y) \\ &+ \mathcal{L}_{\text{GAN}}(F, D_X, Y, X) \\ &+ \lambda \mathcal{L}_{\text{cyc}}(G, F),\end{aligned}$$



Explicit cycle consistency loss using L1

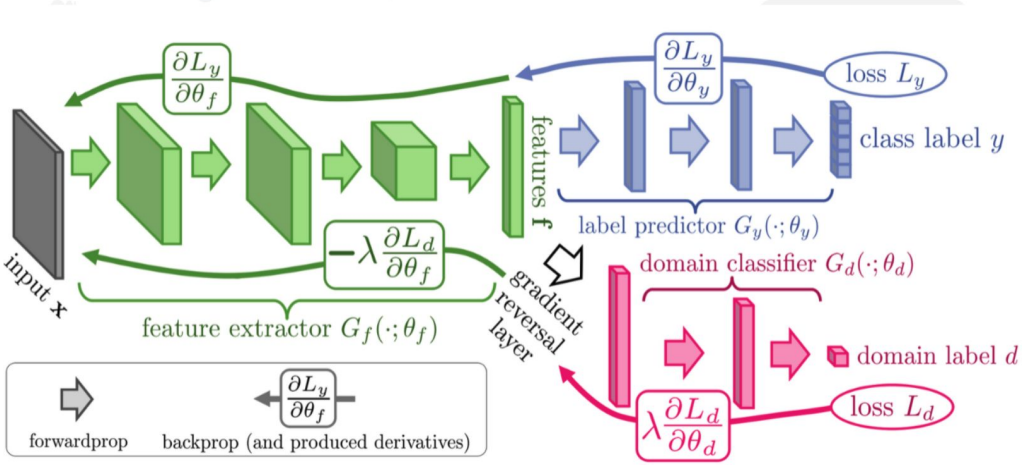
$$\begin{aligned}\mathcal{L}_{\text{cyc}}(G, F) = &\mathbb{E}_{x \sim p_{\text{data}}(x)}[\|F(G(x)) - x\|_1] \\ &+ \mathbb{E}_{y \sim p_{\text{data}}(y)}[\|G(F(y)) - y\|_1]\end{aligned}$$

Implicit cycle consistency loss

$$\begin{aligned}\min_{\boldsymbol{\theta}, \boldsymbol{\phi}} \max_{\boldsymbol{\eta}} \mathcal{L}^{\text{A}}_{\text{Cycle}}(\boldsymbol{\theta}, \boldsymbol{\phi}, \boldsymbol{\eta}) = &\mathbb{E}_{\boldsymbol{x} \sim q(\boldsymbol{x})}[\log \sigma(f_{\boldsymbol{\eta}}(\boldsymbol{x}, \boldsymbol{x}))] \\ &+ \mathbb{E}_{\hat{\boldsymbol{x}} \sim p_{\boldsymbol{\theta}}(\hat{\boldsymbol{x}}|\boldsymbol{z}), \boldsymbol{z} \sim q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{x})} \log(1 - \sigma(f_{\boldsymbol{\eta}}(\boldsymbol{x}, \hat{\boldsymbol{x}})))].\end{aligned}$$

Domain Adaptation (Ganin et al 2016)

- $\mathcal{D}_s = \{(x_i^S, y_i^S)\}_{i=1}^{N_s}$ is the labeled source data
- $\mathcal{D}_t = \{(x_i^T)\}_{i=1}^{N_t}$ is the unlabeled target data
- Task on source and target is the same



The goal is to learn domain invariant features.

- The domain classifier (discriminator) tries to classify source and target domains
- The feature extractor tries to fool

Conditioning GANs

GANs can be extended to a conditional model if both the generator and the discriminator are conditioned on some information (Mirza et al 2014).

$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x}|\boldsymbol{y})] + \mathbb{E}_{\boldsymbol{z}\sim p_z(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z}|\boldsymbol{y})))]$$

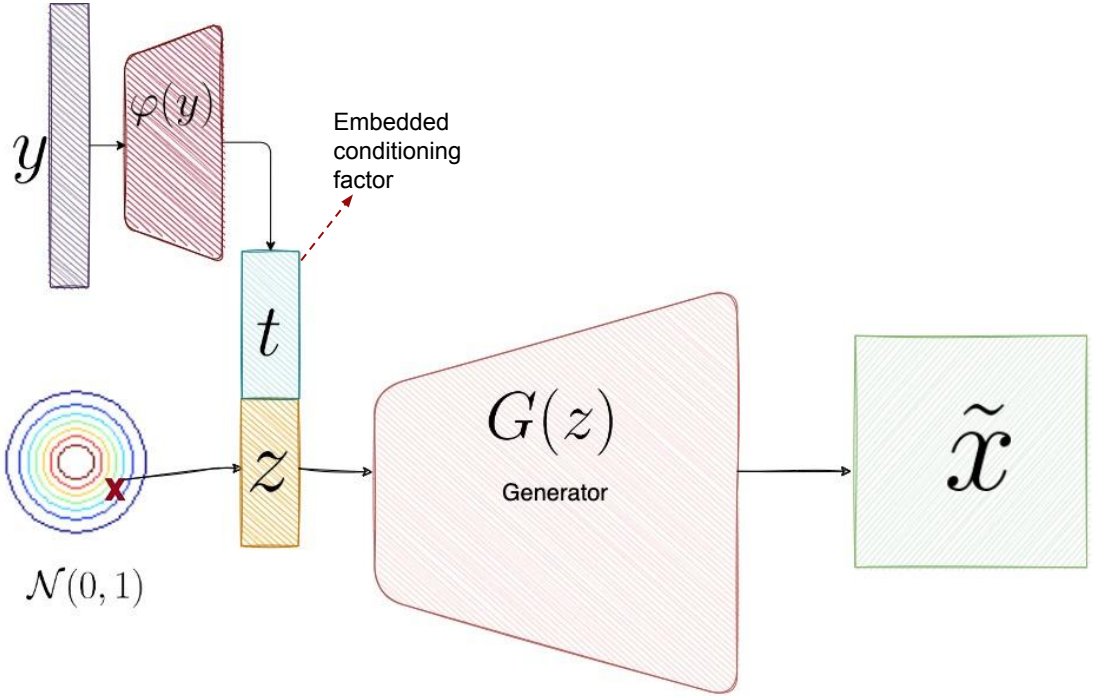
- **Conditioning the Generator:**
  - concatenation
    - Use embedding layer
    - same size as the noise vector
  - conditional batch norm

- **Conditioning the discriminator:**
  - concatenation
    - Use embedding layer
    - Same size as the image
  - using auxiliary classifier (ACGAN, InfoGAN)
  - Projection

Conditioning GANs

Conditioning the Generator: Concatenation (Mirza et al 2014)

- Use an embedding layer
- Balance size of the embedding with size of noise. Both variables should be used equally in the generator.

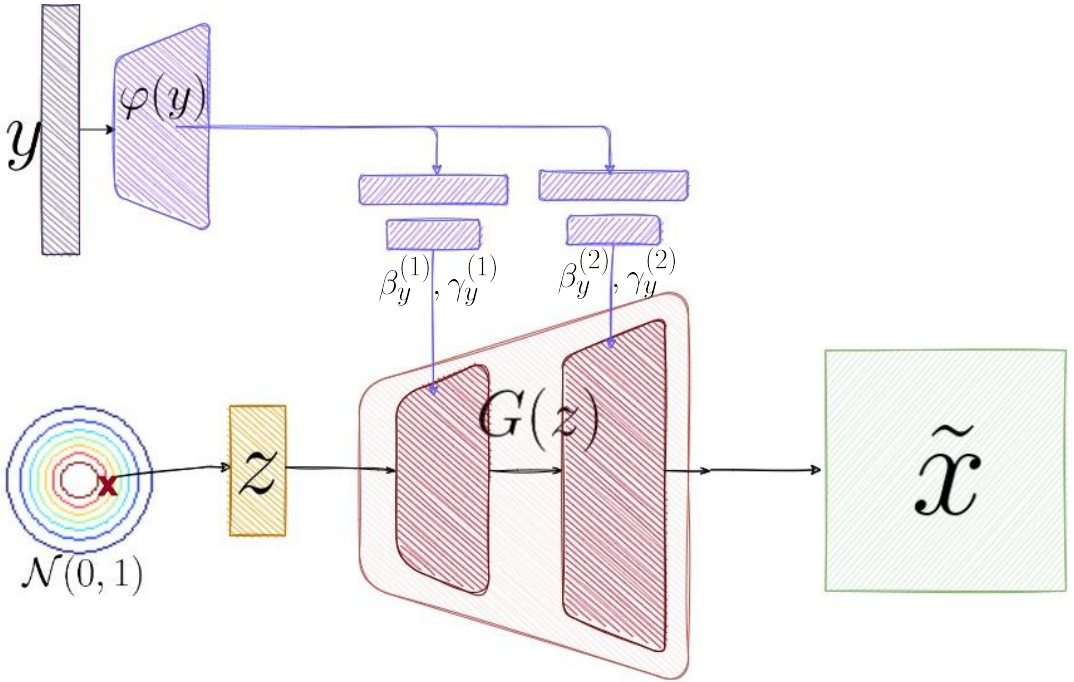


Conditioning GANs

Conditioning the Generator: Conditional Batch Norm (Zhang et al., 2019; Brock et al.,2018)

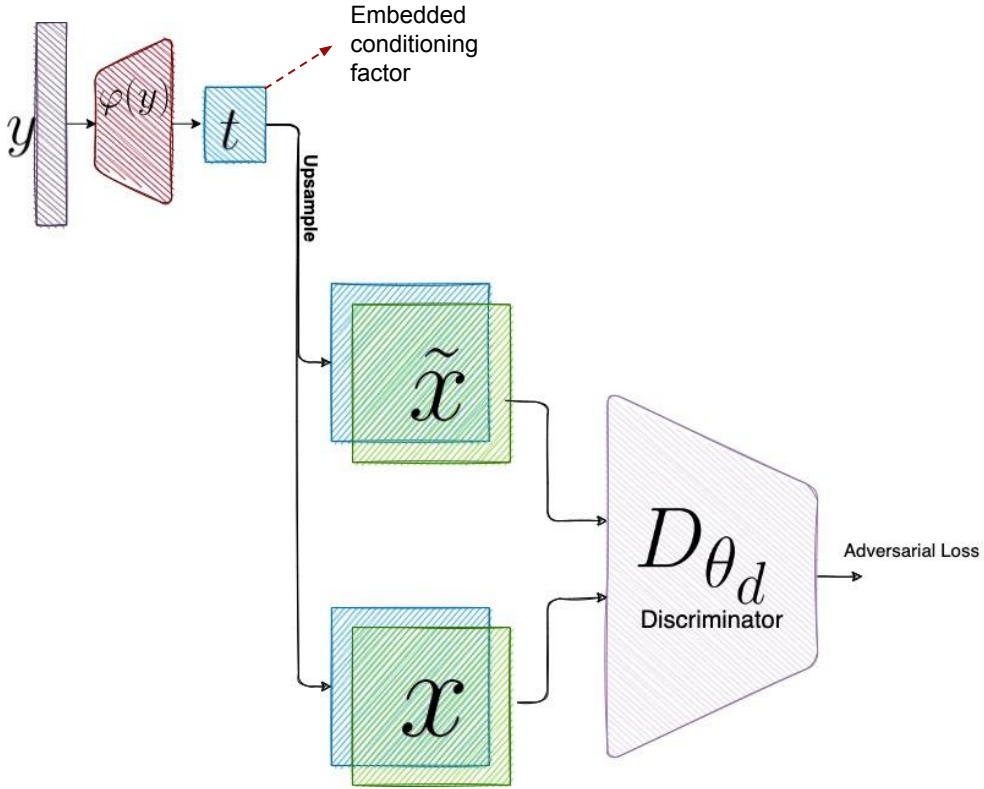
First, the activations are normalized using the mean and the variance calculated over the mini-batch instances and then scaled and shifted by β_y, γ_y

$$\hat{h} = \gamma_y \frac{h-\mu}{\sigma} + \beta_y$$



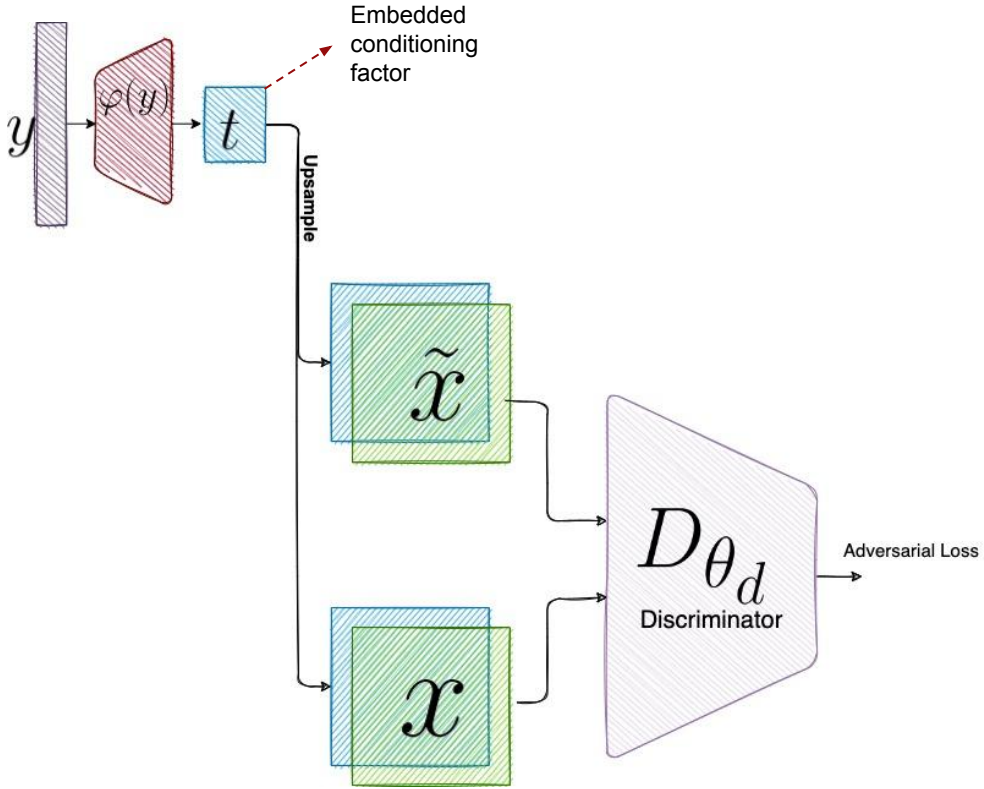
Conditioning GANs

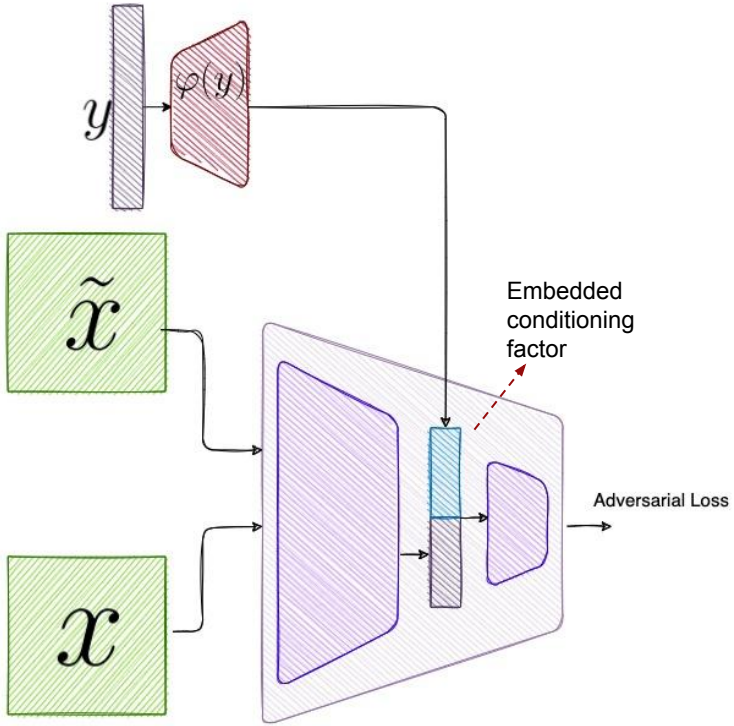
Conditioning the Discriminator: Concatenation (Mirza et al 2014)



Conditioning GANs

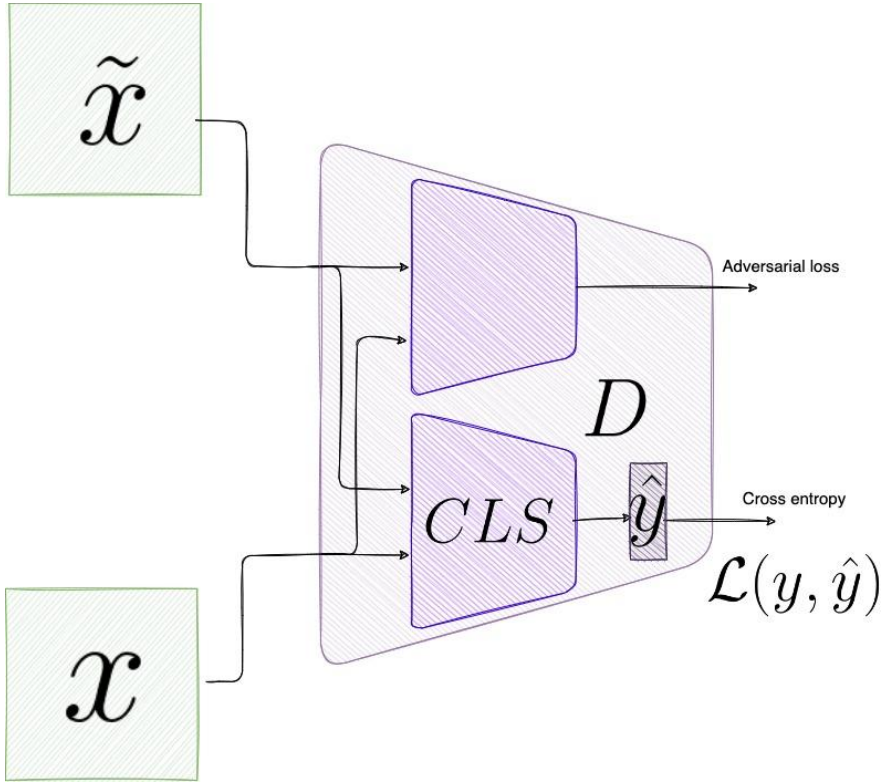
Conditioning the Discriminator: Concatenation (Mirza et al 2014)





Conditioning GANs

Conditioning the Discriminator: Auxiliary Classifier ACGan (Odena et al 2017)

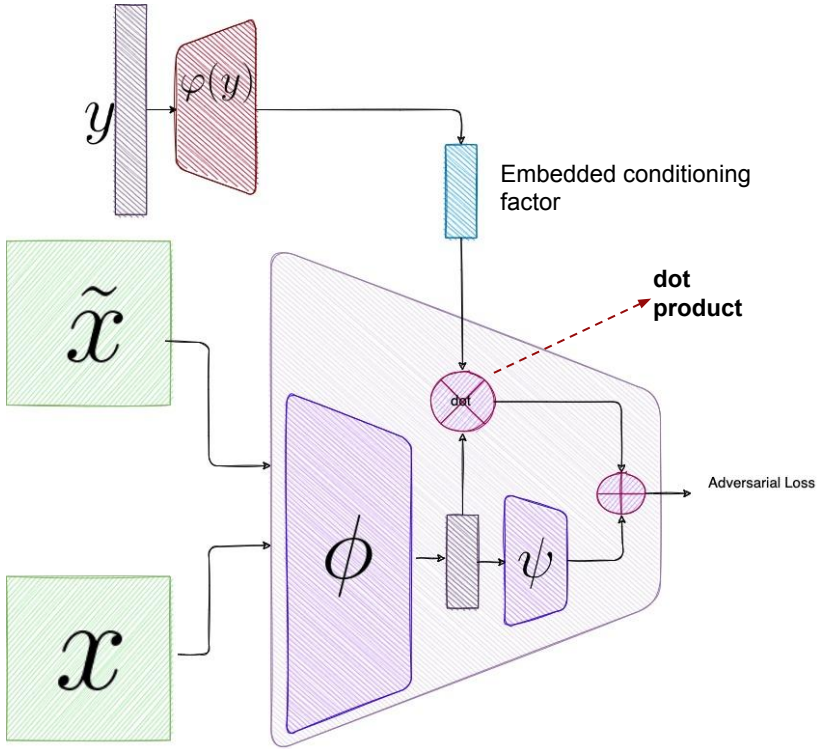


Conditioning GANs

Conditioning the Discriminator: Projection (Miyato et al 2018)

$$D(x, y) = \varphi(y)\phi(x) + \psi(\phi(x))$$

Using a dot product measure the similarity between the condition and the image. This is added to the discriminator logits



On Evaluation of Generative Models

- Likelihood (VAEs, Autoregressive models) $\log p(x) = \log \sum_k p(x|z_k)p(z_k)$

On Evaluation of Generative Models

- Likelihood (VAEs, Autoregressive models) $\log p(x) = \log \sum_k p(x|z_k)p(z_k)$
- sample based
  - **qualitative:**
    - Nearest neighbor
    - Human rating via crowdsourcing
    - Evaluating Mode Drop and Mode Collapse.

On Evaluation of Generative Models

- Likelihood (VAEs, Autoregressive models) $\log p(x) = \log \sum_k p(x|z_k)p(z_k)$
- sample based
  - **qualitative:**
    - Nearest neighbor
    - Human rating via crowdsourcing
    - Evaluating Mode Drop and Mode Collapse.
  - **quantitative:**
    - Inception score:
    - Frechet Inception Distance (FID)
    - KID (https://arxiv.org/abs/1801.01401) [uses MMD as distance measure]

On Evaluation of Generative Models

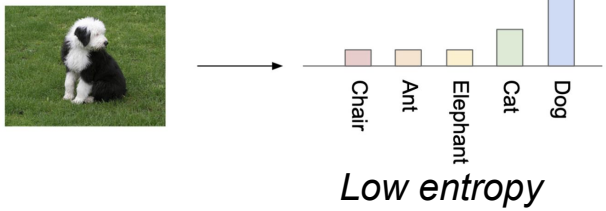
- Likelihood (VAEs, Autoregressive models) $\log p(x) = \log \sum_k p(x|z_k)p(z_k)$
- sample based
  - **qualitative:**
    - Nearest neighbor
    - Human rating via crowdsourcing
    - Evaluating Mode Drop and Mode Collapse.
  - **quantitative:**
    - Inception score:
    - Frechet Inception Distance (FID)
    - KID (https://arxiv.org/abs/1801.01401)
- Conditional Generation:
  - Conditional consistency (in the case of Conditional Models): use a pretrained model to evaluate how the generated samples respect the conditioning factor:
    - For colorization: evaluate on a model trained on color images (Zhang et al. [84])
    - For bounding boxes, evaluate on object detection,
    - For class conditioning, classification …

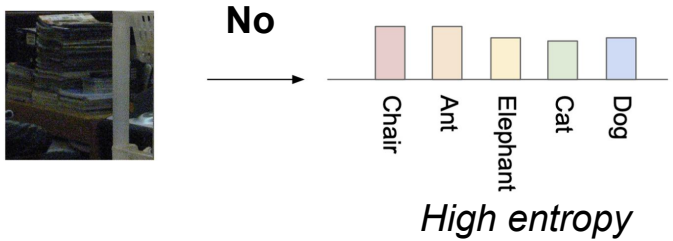
On Inception score

- **Image Quality**. Do images look like a specific object?
- **Image Diversity**. Is a wide range of objects generated?

A good blogpost

We can use the classifier to assess if the image has a distinct object.
This measures the **quality** of **each** generated image



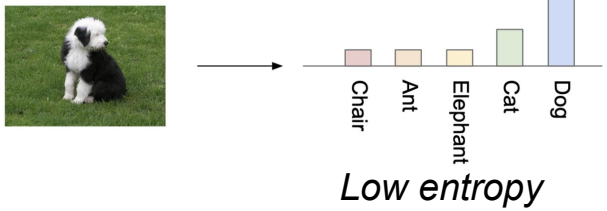


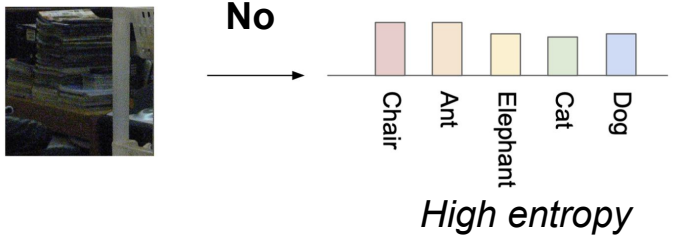
images from: https://medium.com/octavian-ai/a-simple-explanation-of-the-inception-score-372dff6a8c7a

On Inception score

- **Image Quality**. Do images look like a specific object?
- **Image Diversity**. Is a wide range of objects generated?

We can use the classifier to assess if the image has a distinct object.
This measures the **quality** of **each** generated image





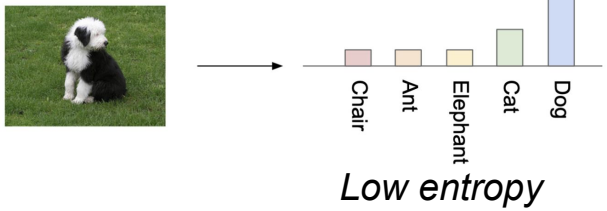
For each generated example Measure the conditional entropy

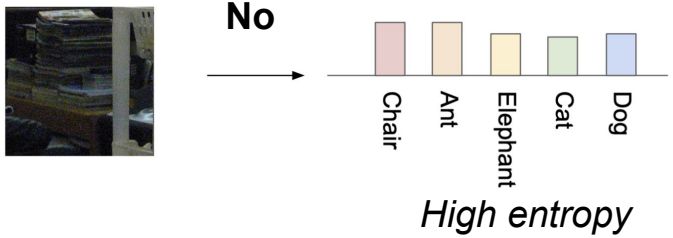
$$H(y|\tilde{x}_i) = -\sum_y p(y|\tilde{x}_i) \log p(y|\tilde{x}_i)$$

On Inception score

- **Image Quality**. Do images look like a specific object?
- **Image Diversity**. Is a wide range of objects generated?

We can use the classifier to assess if the image has a distinct object.
This measures the **quality** of **each** generated image





For each generated example Measure the conditional entropy

$$H(y|\tilde{x}_i) = -\sum_y p(y|\tilde{x}_i) \log p(y|\tilde{x}_i)$$

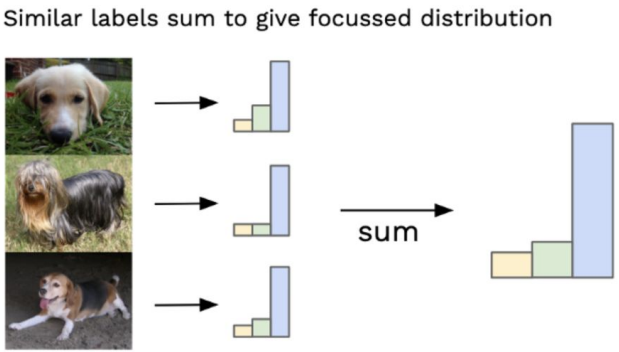
Take the expectation over many generations

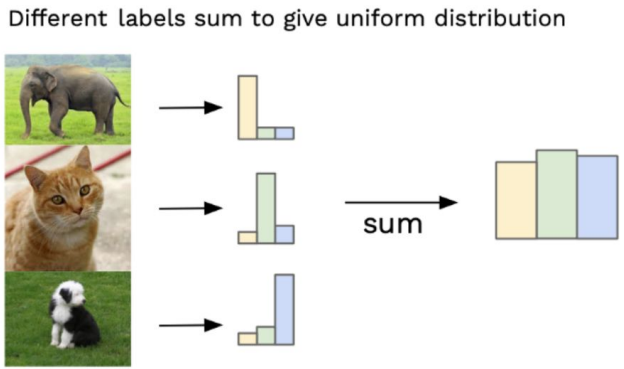
$$H(y|\tilde{X}) = \frac{1}{N} \sum_{i=1}^{N} H(y|\tilde{x}_i)$$

On Inception score

- **Image Quality**. Do images look like a specific object?
- **Image Diversity**. Is a wide range of objects generated?

we can combine the label probability distributions for many of of our generated images. (50,000 generated images)
This measures the **diversity** over **all** generated images

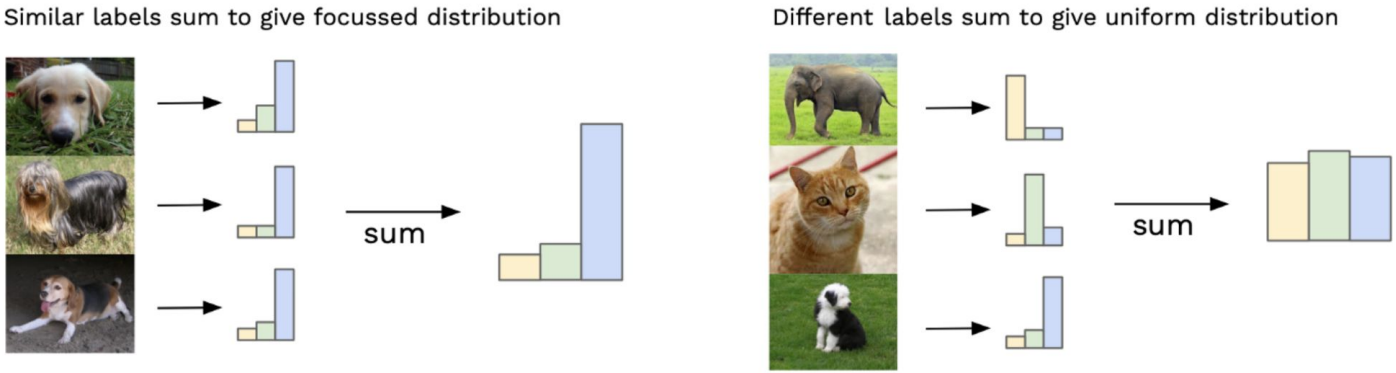




images from: https://medium.com/octavian-ai/a-simple-explanation-of-the-inception-score-372dff6a8c7a

On Inception score

we can combine the label probability distributions for many of of our generated images. (50,000 generated images)
This measures the **diversity** over **all** generated images

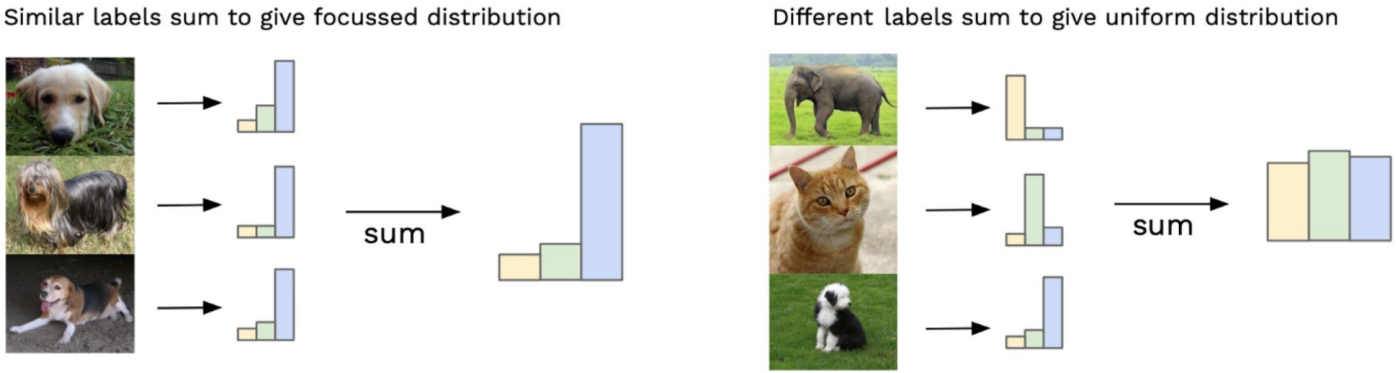


Compute expectation over class prob. of many generated images

$$p(y) = \frac{1}{N}\sum_{i=1}^{N} P(y|\tilde{x}_i)$$

On Inception score

we can combine the label probability distributions for many of of our generated images. (50,000 generated images)
This measures the **diversity** over **all** generated images



Compute expectation over class prob. of many generated images

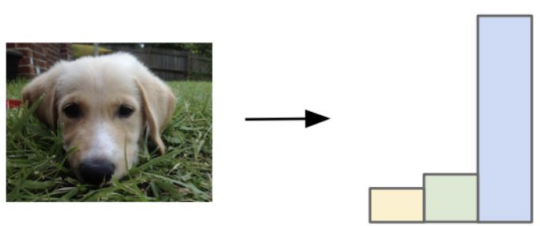
$$p(y) = \frac{1}{N}\sum_{i=1}^{N} P(y|\tilde{x}_i)$$

Compute entropy over this marginal expectation

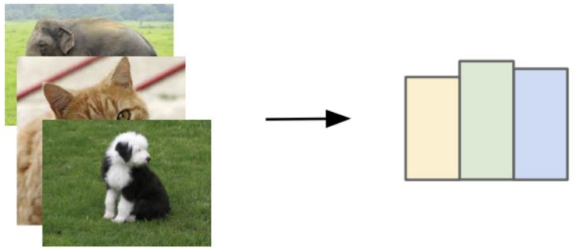
$$H(y) = -\sum_{y} p(y)\log p(y)$$

On Inception score

- **Image Quality**. Do images look like a specific object?
- **Image Diversity**. Is a wide range of objects generated?



Ideal label distribution



Ideal marginal distribution

We want each image to each be distinct (above figure, left)
and to collectively have variety (above figure, right)

$$I(y; \tilde{X}) = H(y) - H(y|\tilde{X})$$

On Fréchet Inception Distance (FID)

- FID summarizes how similar two groups of images are in terms of statistics on features of images calculated using the inception v3 model used for image classification.
- It is used to assess the quality of generated images. i.e. it measures the distance of the generated images to real images in feature space.
- The FID score is calculated by first loading a pre-trained Inception v3 model.
- The output layer of the model is removed and the output is taken as the activations from the last pooling layer
- Using this pretrained model, Feature vectors is then predicted for a collection of real images from the problem domain to provide a reference for how real images are represented. Feature vectors are also computed for generated images
- Compute FID using the following formula $$\mathrm{FID} = \|\mu - \mu_w\|_2^2 + \mathrm{tr}\left(\Sigma + \Sigma_w - 2\left(\Sigma^{\frac{1}{2}} \cdot \Sigma_w \cdot \Sigma^{\frac{1}{2}}\right)^{\frac{1}{2}}\right)$$

On Fréchet Inception Distance (FID)

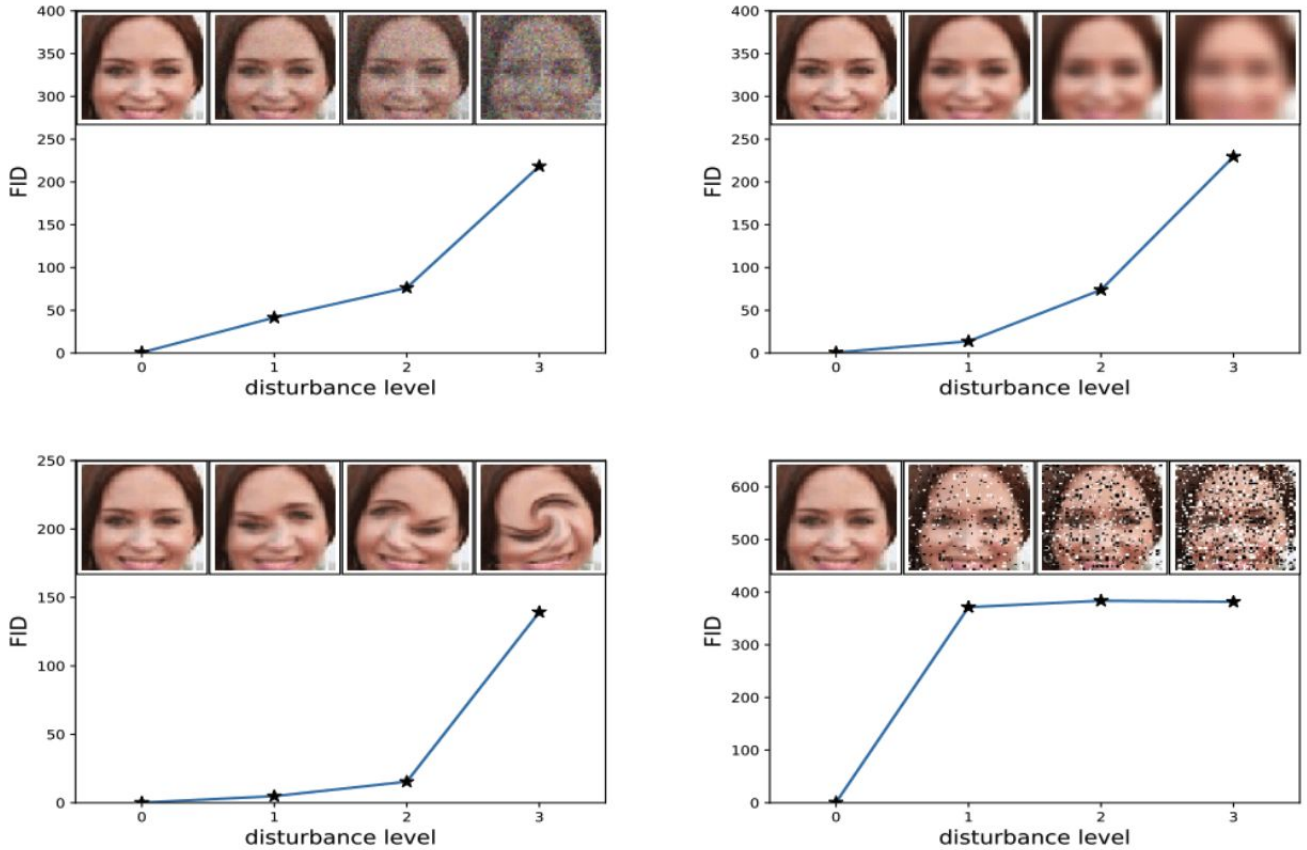
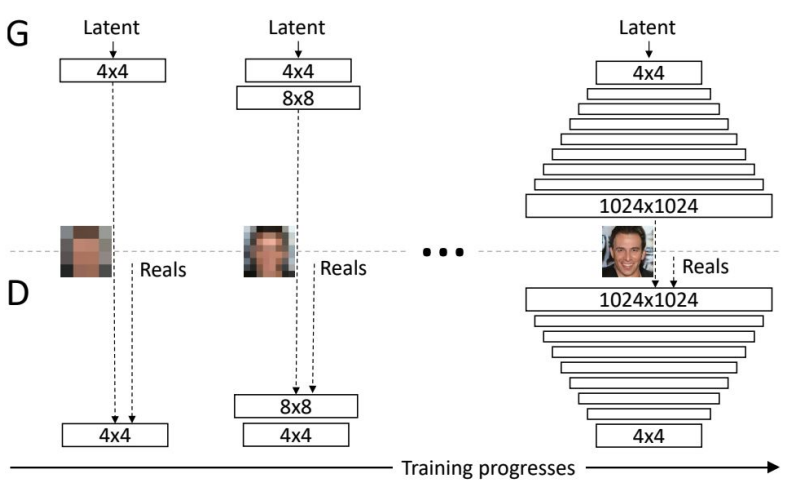


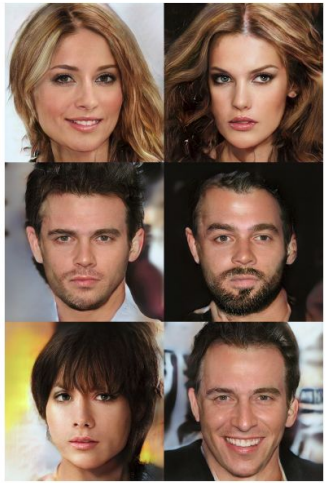
image from https://machinelearningmastery.com/how-to-implement-the-frechet-inception-distance-fid-from-scratch/

Progressive Growing of GANs (Karras et al 2017)

key idea: grow both the generator and discriminator progressively

starting from a low resolution, we add new layers and model increasingly adds details as training progresses.

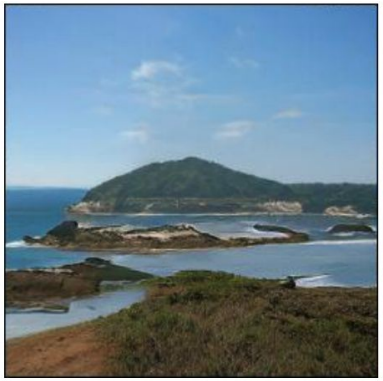




Benefits:

- Early in training generation of smaller images is stable. No details in images.
- Iterative training reduces complexity at each iteration
- Training on smaller images is very fast. 2-6 times faster training

BigGAN (Brock et al 2018)


Better conditional GANs by scaling up

- **More parameters (increasing width)**
- **Larger batch size**
- Change in architecture
  - Self-attention GAN (SAGAN)
  - Using conditional instance normalization for conditioning G
  - Using HingLoss
  - Skip connections from z to further layers in G
  - using projection discriminator for conditioning D
- Spectral Normalization
- Truncation trick
- Orthogonal regularization
- ….

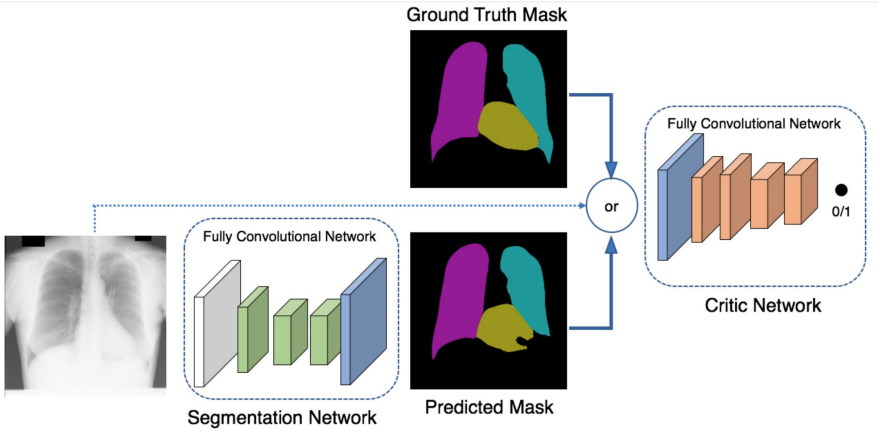
Applications in Medical Imaging

- Generative: one to many mapping
  - unconditional generation
  - conditional generation
- Deterministic: one to one mapping
  - Image segmentation
  - Reconstruction
  - Domain Transfer
  - Domain adaptation

Segmentation (Yang et al 2017) (Dai et al 2018)

The adversarial loss imposes on the segmentation network, the structural regularities emerging from human physiology

Segmentation of the lung fields and the heart in Chest X-Ray **(Yang et al 2017)**

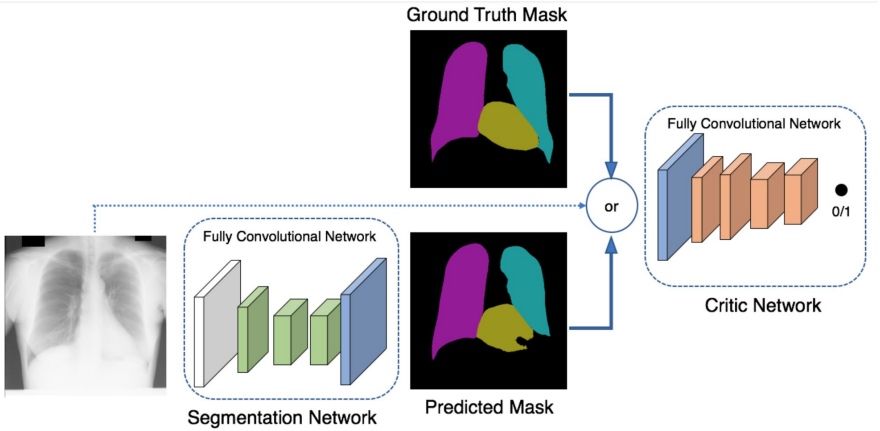


$$\min_S \max_D \Big\{ J(S,D) := \sum_{i=1}^{N} J_s(S(\boldsymbol{x}_i), \boldsymbol{y}_i) - \lambda \Big[J_d(D(\boldsymbol{x}_i, \boldsymbol{y}_i), 1) + J_d(D(\boldsymbol{x}_i, S(\boldsymbol{x}_i)), 0) \Big] \Big\}$$

Segmentation (Yang et al 2017) (Dai et al 2018)

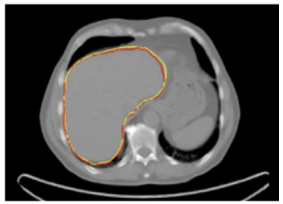
The adversarial loss imposes on the segmentation network, the structural regularities emerging from human physiology

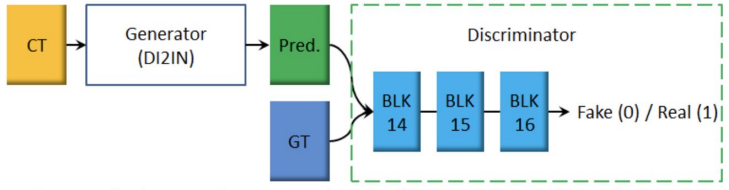
Segmentation of the lung fields and the heart in Chest X-Ray



$$\min_S \max_D \Big\{ J(S,D) := \sum_{i=1}^{N} J_s(S(\boldsymbol{x}_i), \boldsymbol{y}_i) - \lambda \Big[J_d(D(\boldsymbol{x}_i, \boldsymbol{y}_i), 1) + J_d(D(\boldsymbol{x}_i, S(\boldsymbol{x}_i)), 0) \Big] \Big\}$$

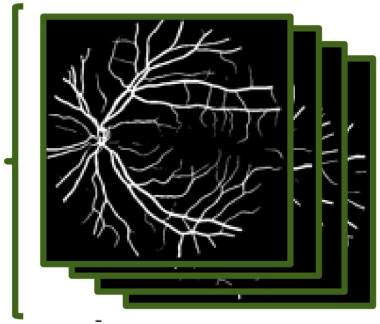
Liver segmentation in 3D CT **(Dai et al 2018)**



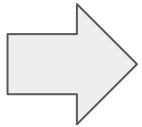


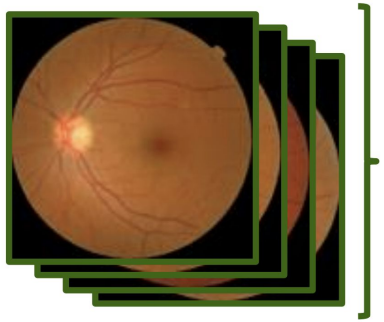
$$G^* = \arg\min_G \max_D L_{GAN}(G,D) + \lambda L_{seg}$$

Conditional synthesis (Costa et al 2017)



vessel tree segmentation



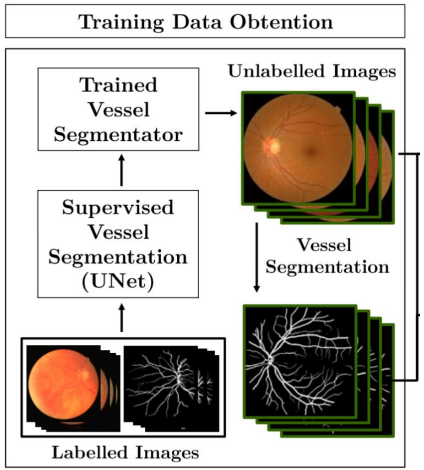


Retina Images

Conditional synthesis (Costa et al 2017)

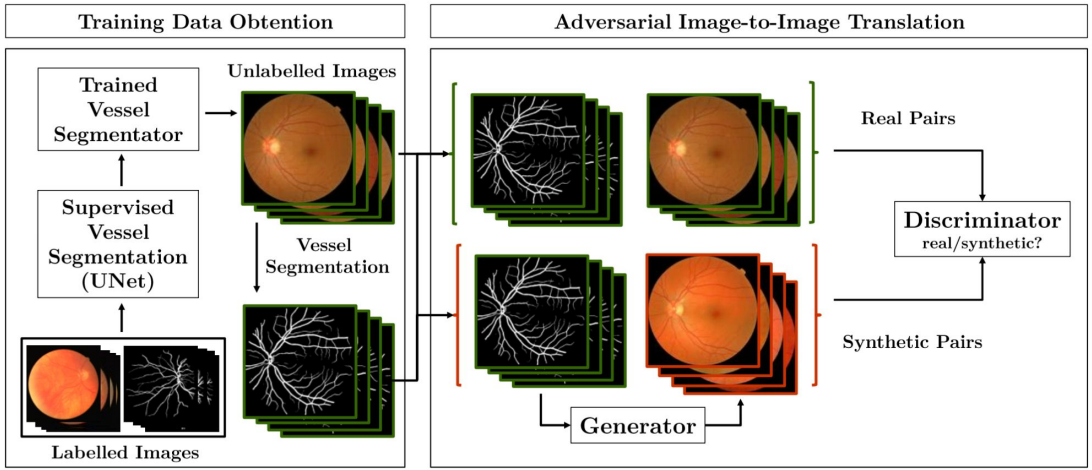
Phase 1:
Acquire training data for training the generation process.

It’s easier to go from Retina images to vessel tree (Needs less data)



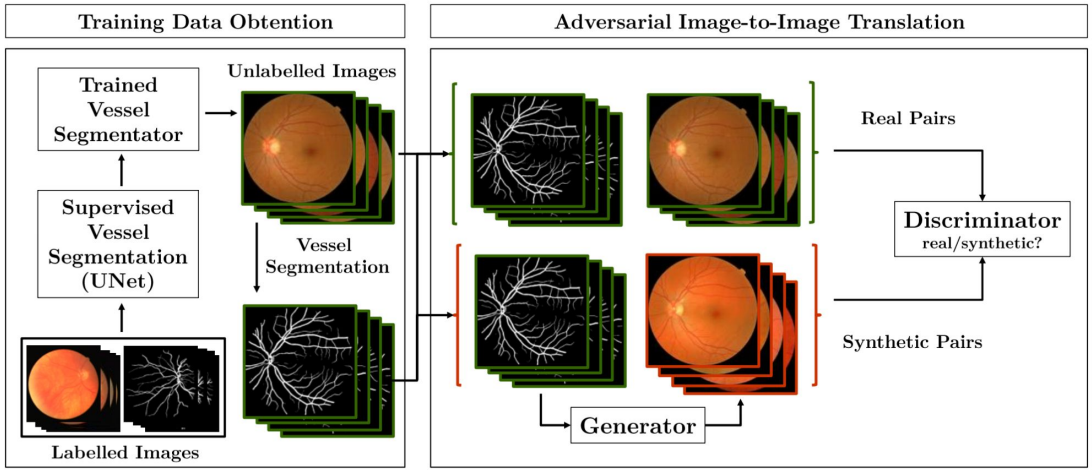
Conditional synthesis (Costa et al 2017)

Phase 2
Use the paired vessel segmentation maps together with Retina images to train the generation process.



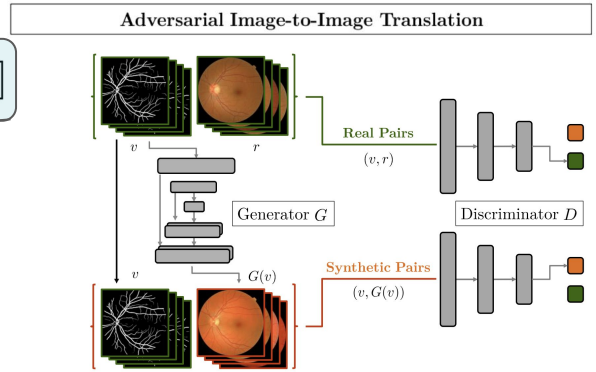
Conditional synthesis (Costa et al 2017)

Phase 2
Use the paired vessel segmentation maps together with Retina images to train the generation process.



$$\mathcal{L}_{adv}(G,D) = \mathbb{E}_{v,r\sim p_{data}(v,r)}[logD(v,r)] + \mathbb{E}_{v\sim p_{data}(v)}[log(1-D(v,G(v)))]$$

$$\mathcal{L}(G,D) = \mathcal{L}_{adv}(G,D) + \lambda\mathbb{E}_{v,r\sim p_{data}(v,r)}\left(||r-G(v)||_1\right)$$



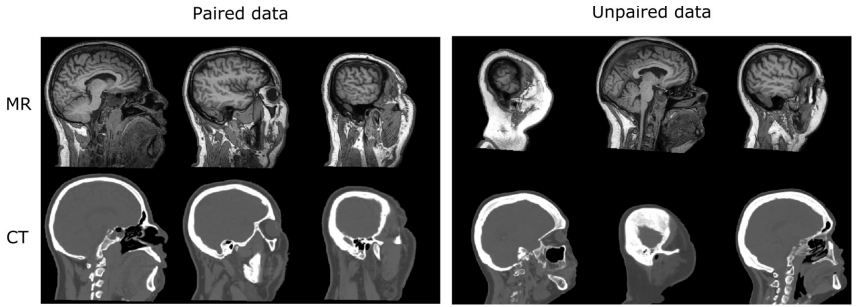
Cross modality synthesis

(Wolterink et al 2017)

Learn a modality translation function from MR to CT and from CT to MR

Evaluated both with paired data using pix-to-pix and unpaired data using CycleGAN

Due to noisy registration in paired data results on unpaired data with CycleGAN was better



When training with paired data, MR and CT slices that are simultaneously provided to the network correspond to the same patient at the same anatomical location.

When training with unpaired data, MR and CT slices that are simultaneously provided to the network belong to different patients at different locations in the brain

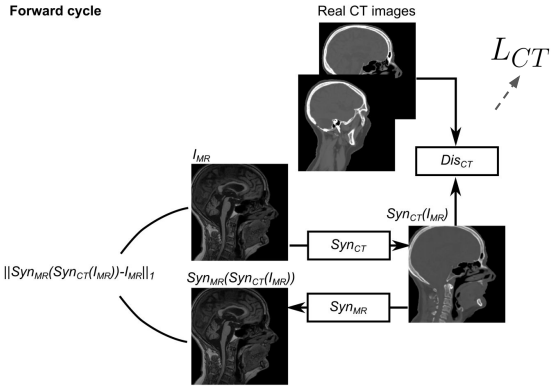
Cross modality synthesis

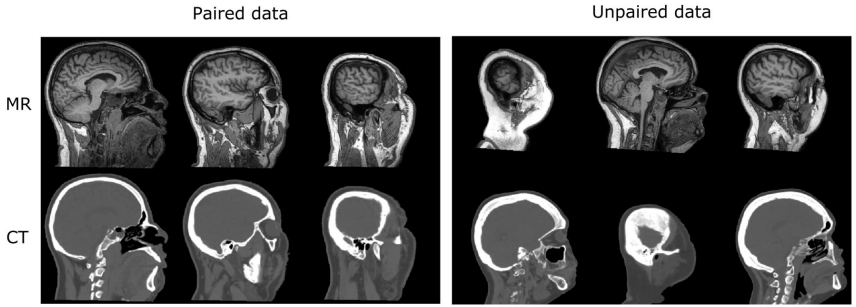
(Wolterink et al 2017)

Learn a modality translation function from MR to CT and from CT to MR

Evaluated both with paired data using pix-to-pix and unpaired data using CycleGAN

Due to noisy registration in paired data results on unpaired data with CycleGAN was better





When training with paired data, MR and CT slices that are simultaneously provided to the network correspond to the same patient at the same anatomical location.

When training with unpaired data, MR and CT slices that are simultaneously provided to the network belong to different patients at different locations in the brain

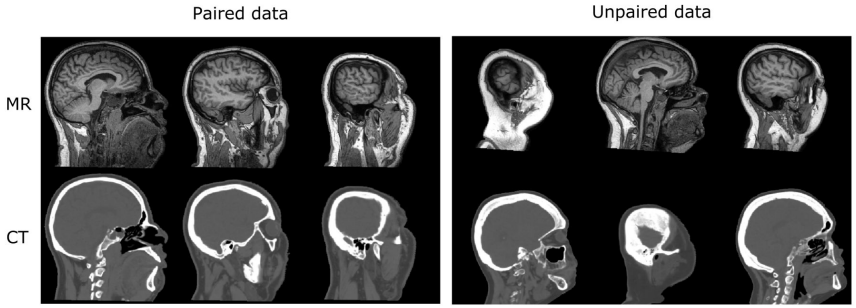
Cross modality synthesis

(Wolterink et al 2017)

Learn a modality translation function from MR to CT and from CT to MR

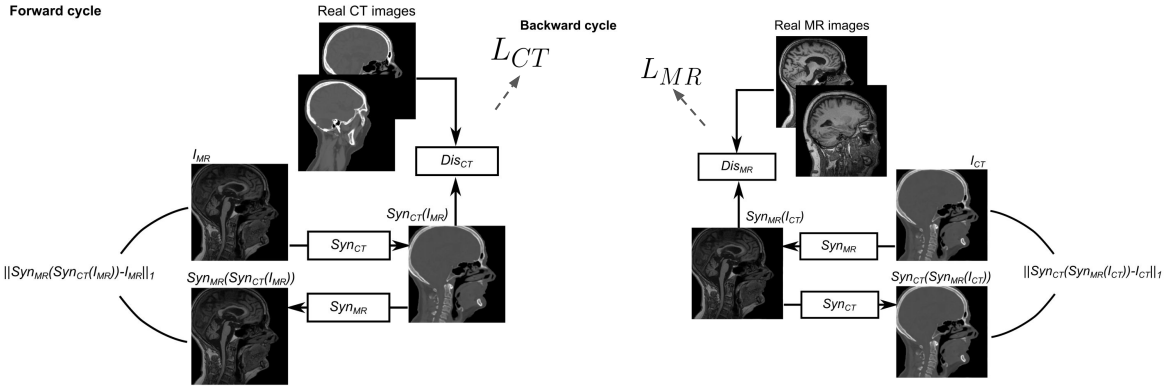
Evaluated both with paired data using pix-to-pix and unpaired data using CycleGAN

Due to noisy registration in paired data results on unpaired data with CycleGAN was better



When training with paired data, MR and CT slices that are simultaneously provided to the network correspond to the same patient at the same anatomical location.

When training with unpaired data, MR and CT slices that are simultaneously provided to the network belong to different patients at different locations in the brain



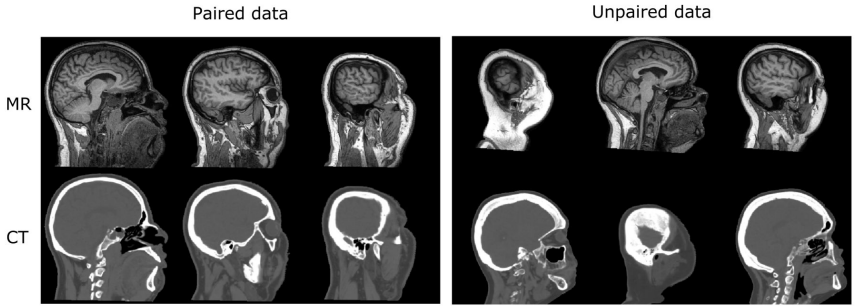
Cross modality synthesis

(Wolterink et al 2017)

Learn a modality translation function from MR to CT and from CT to MR

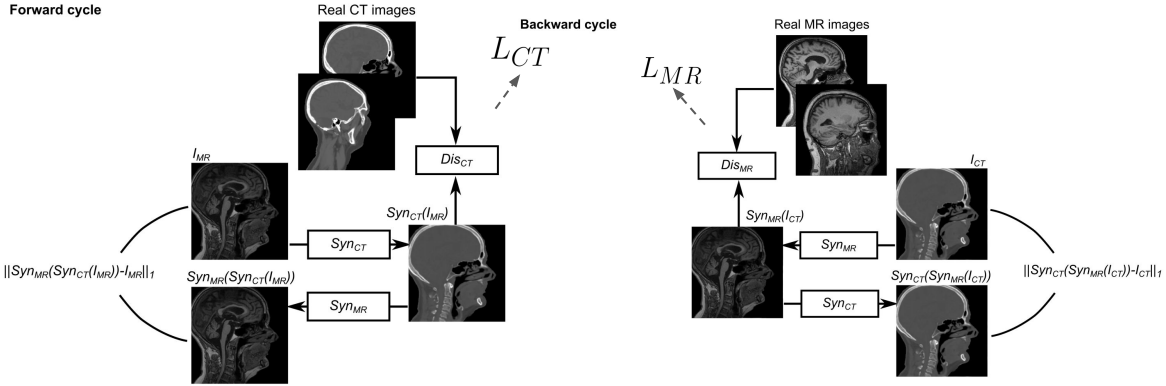
Evaluated both with paired data using pix-to-pix and unpaired data using CycleGAN

Due to noisy registration in paired data results on unpaired data with CycleGAN was better



When training with paired data, MR and CT slices that are simultaneously provided to the network correspond to the same patient at the same anatomical location.

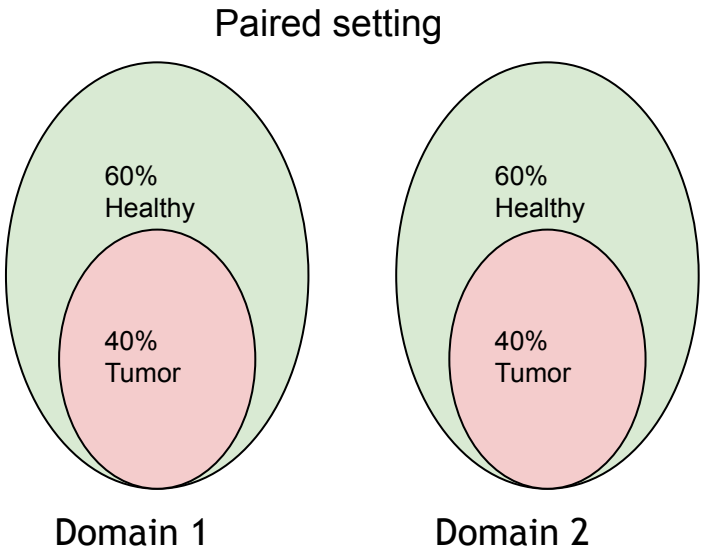
When training with unpaired data, MR and CT slices that are simultaneously provided to the network belong to different patients at different locations in the brain



$$\mathcal{L}_{Cycle} = ||Syn_{MR}(Syn_{CT}(I_{MR})) - I_{MR}||_1 + ||Syn_{CT}(Syn_{MR}(I_{CT})) - I_{CT}||_1$$

Cross modality synthesis

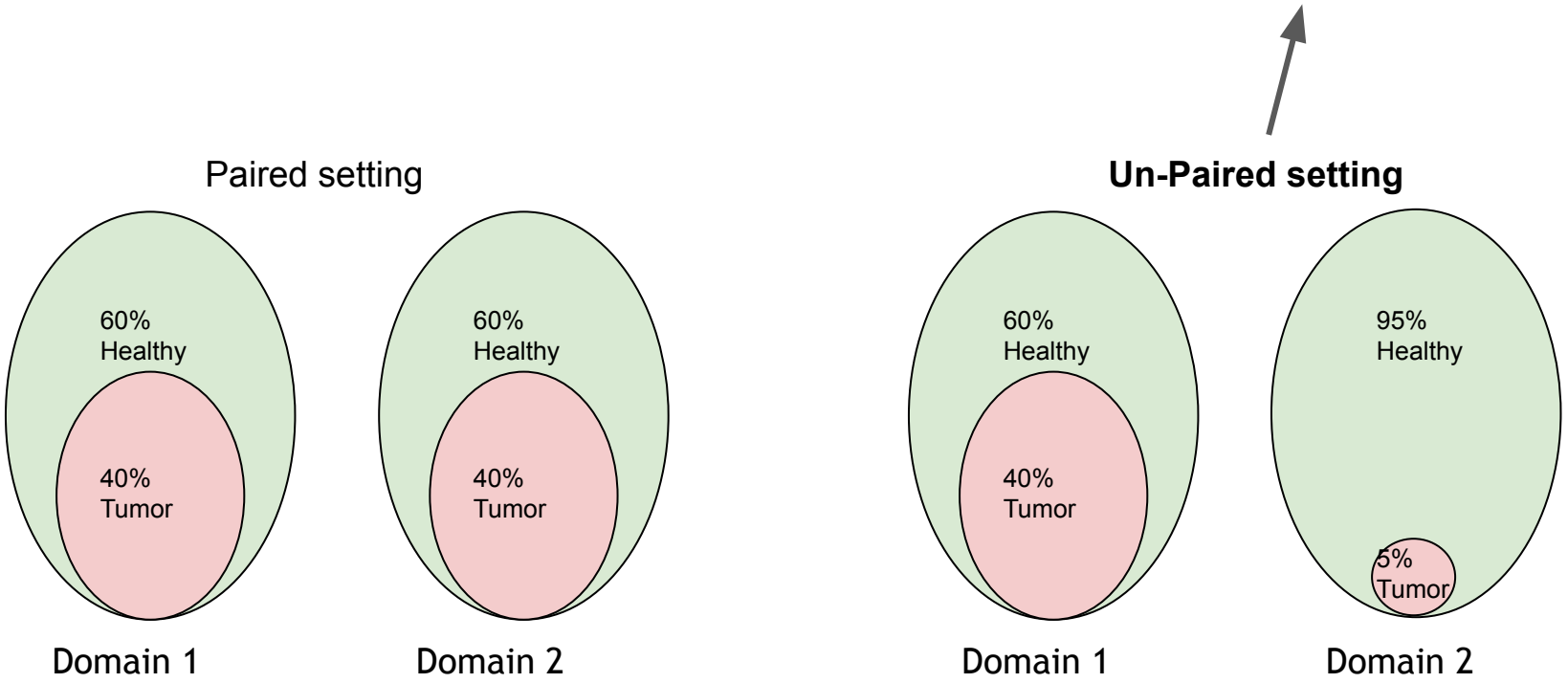
Implicit distribution matching may cause feature hallucination

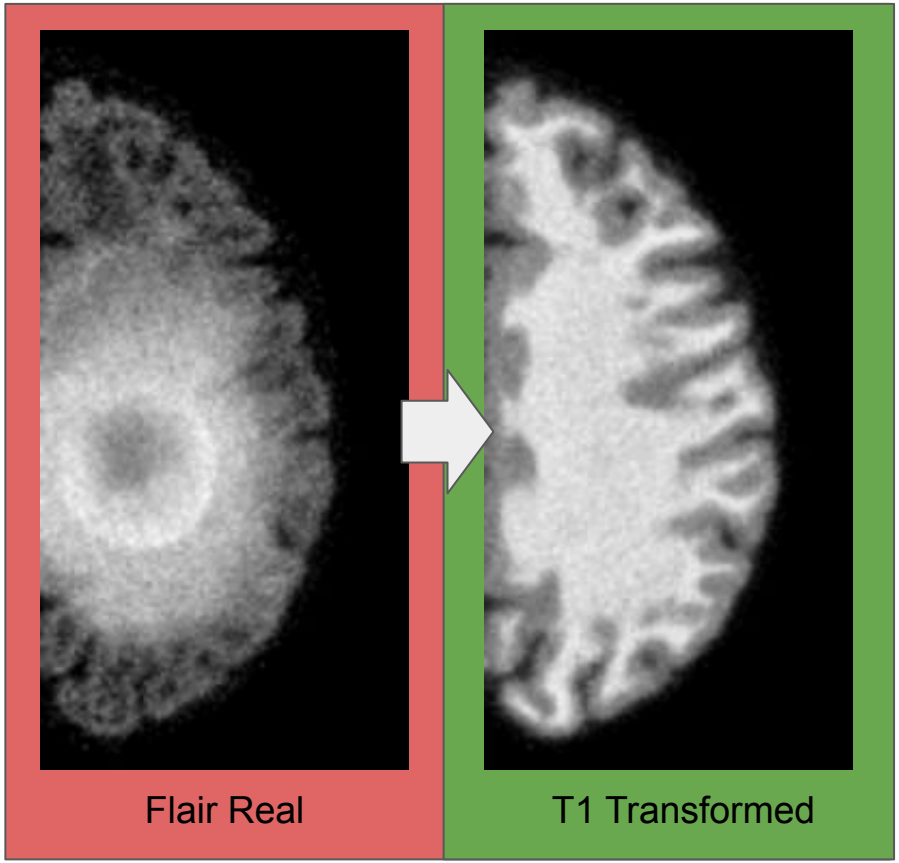


Cross modality synthesis

Implicit distribution matching may cause feature hallucination

Could feature hallucination
or feature removal



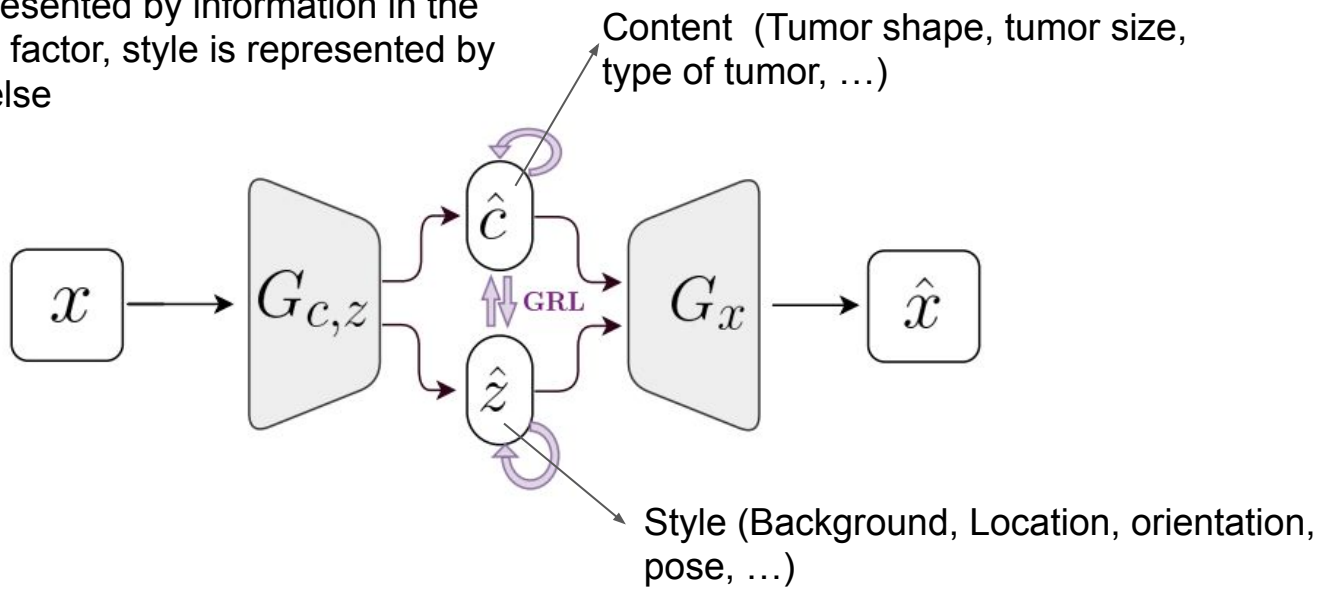


Slide from Joseph Cohen, MILA (source)

Adversarial Inference

DRAI (Havaei et al 2021)

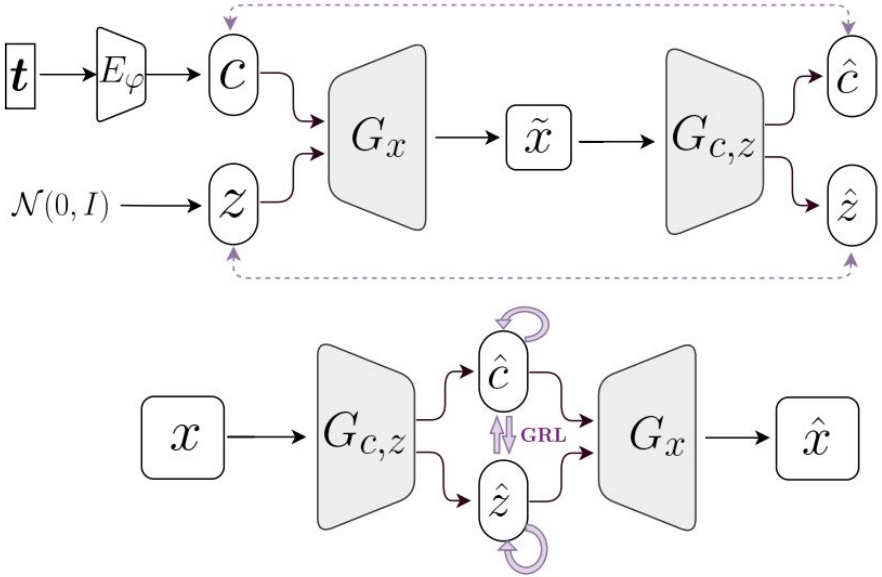
- Both conditional generator and inference machine
- **inferes style and content in a disentangled way**
- content represented by information in the conditioning factor, style is represented by everything else



Adversarial Inference

DRAI (Havaei et al 2021)

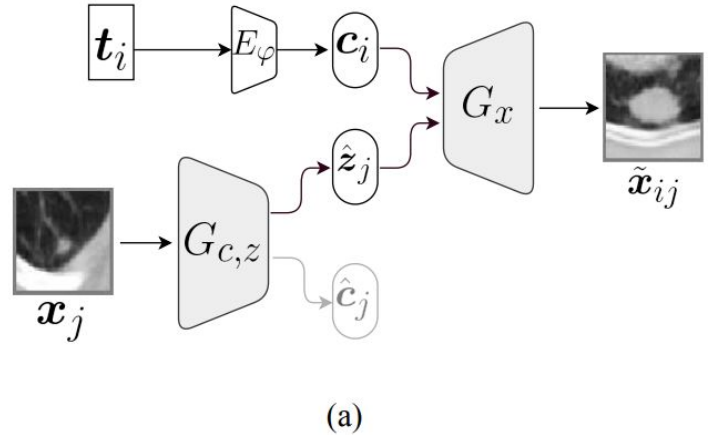
- To train we use a method inspired by ALI.
- We use dual random variables instead
- We use disentanglement constraints

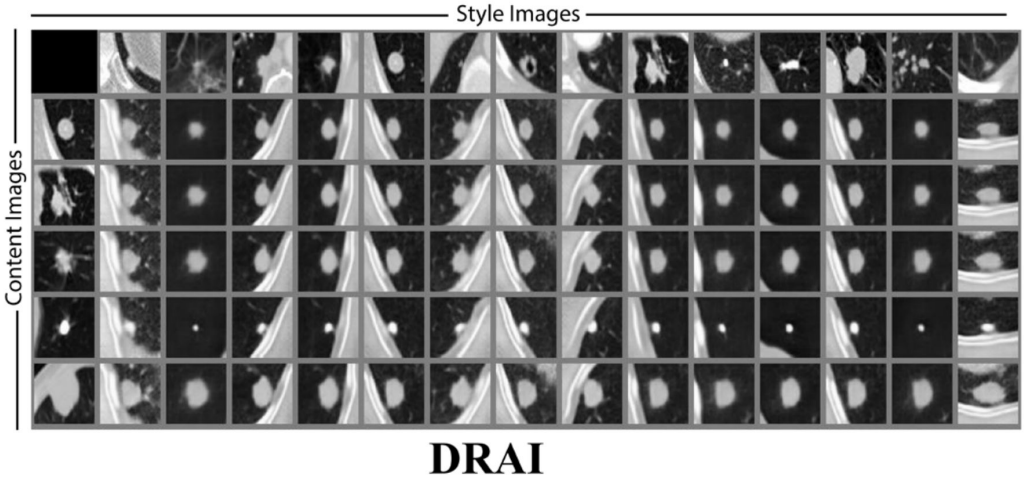


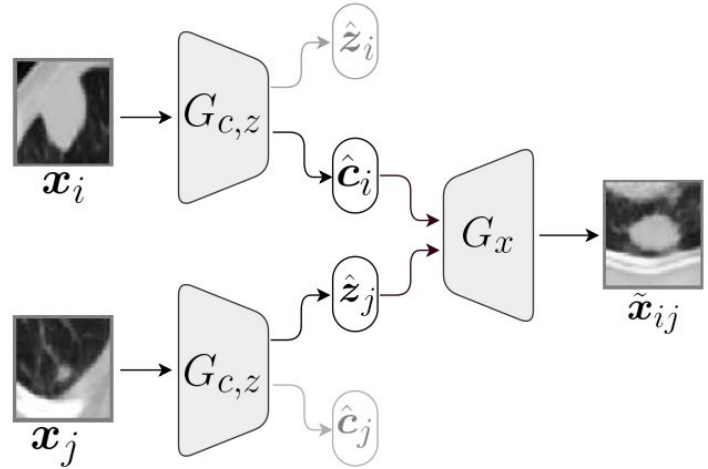
Adversarial Inference

DRAI (Havaei et al 2021)

- One trained it has interesting capabilities.
- In addition to conditional generation using attributes t, we can control the generation with style images or content images.

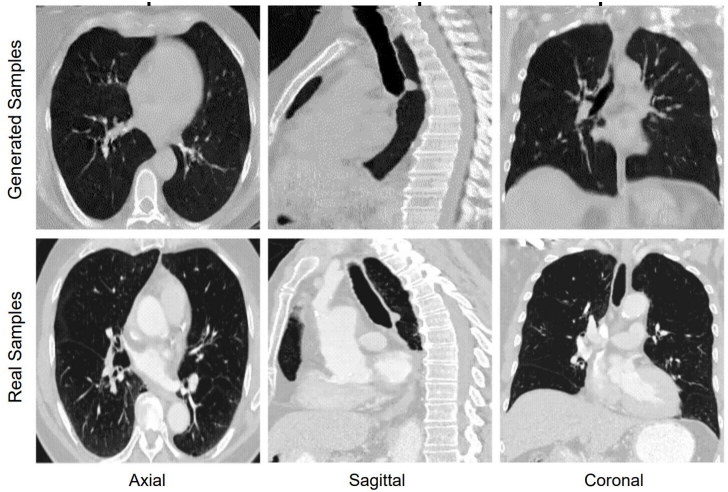






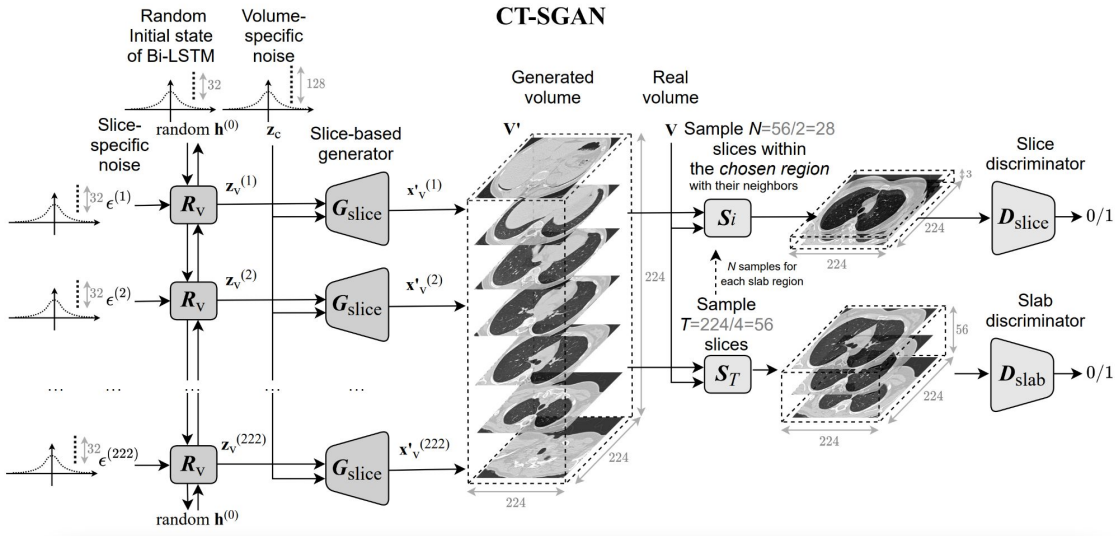
CTSGAN (Pesaranghader et al Havaei 2021)

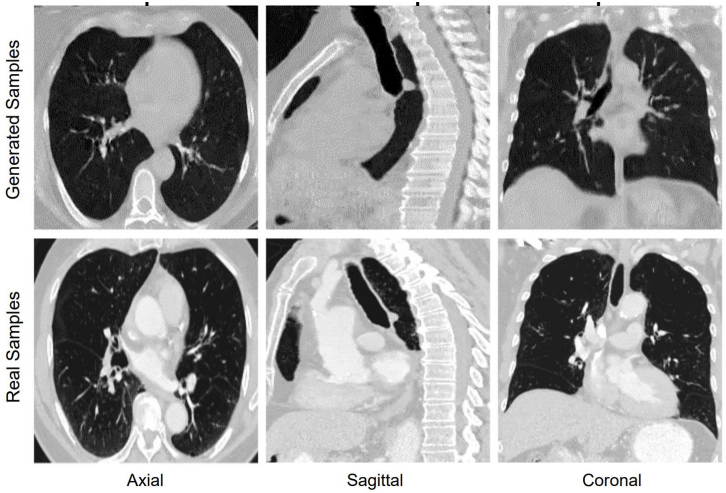
Generate high resolution (224x224x224) lung CT volumes using Bi-LSTM and GAN



CTSGAN (Pesaranghader et al Havaei 2021)

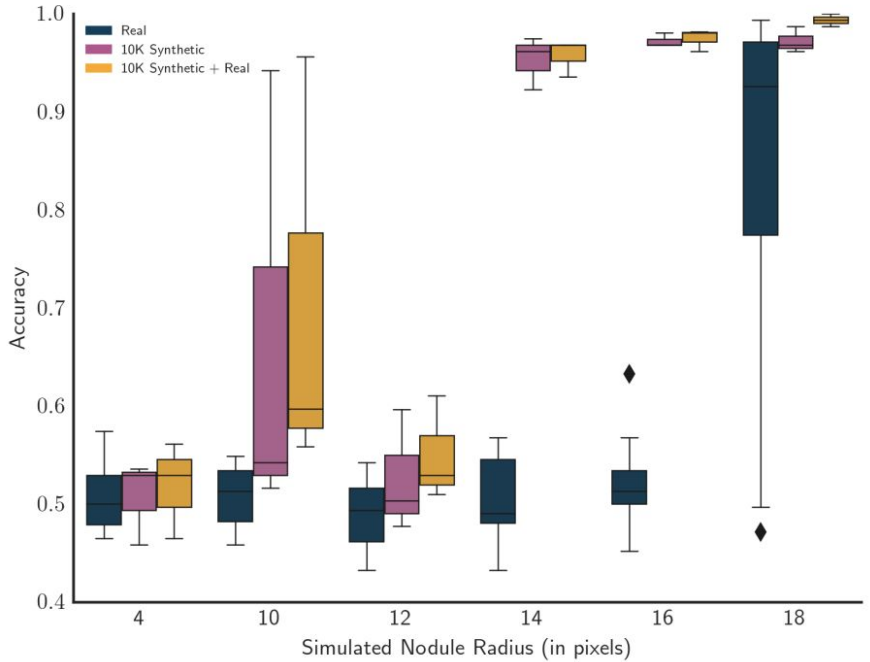
Generate high resolution (224x224x224) lung CT volumes using Bi-LSTM and GAN

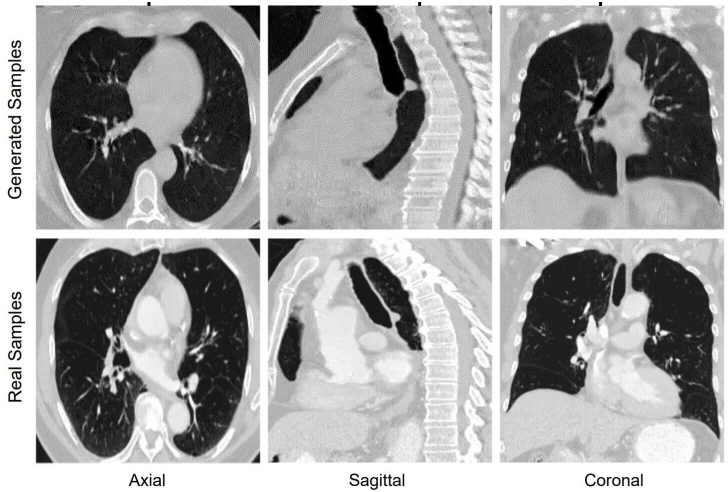




CTSGAN (Pesaranghader et al Havaei 2021)

Evaluate whether pre-training on synthetic data can improve nodule detection task.





Conclusion

- VAEs are a probabilistic model trains by maximizing ELBO
- GANs train by implicitly matching distributions
- Evaluating generative models is not very simple
    - Use both qualitative and quantitative measures.
- VAEs provide inference out of the box, while vanilla GANs dont
- Training VAEs is generally more stable
- GANs provide better looking samples.

References

- Kingma, D.P. and Welling, M., 2013. Auto-encoding variational bayes. arXiv. arXiv preprint arXiv:1312.6114.
- Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S. and Lerchner, A., 2016. beta-vae: Learning basic visual concepts with a constrained variational framework.
- Van Den Oord, A. and Vinyals, O., 2017. Neural discrete representation learning. Advances in neural information processing systems, 30.
- Painchaud, N., Skandarani, Y., Judge, T., Bernard, O., Lalande, A. and Jodoin, P.M., 2020. Cardiac segmentation with strong anatomical guarantees. IEEE transactions on medical imaging, 39(11), pp.3703-3713.
- Kohl, S., Romera-Paredes, B., Meyer, C., De Fauw, J., Ledsam, J.R., Maier-Hein, K., Eslami, S.M., Jimenez Rezende, D. and Ronneberger, O., 2018. A probabilistic u-net for segmentation of ambiguous images. Advances in neural information processing systems, 31.
- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A. and Bengio, Y., 2014. Generative adversarial nets. Advances in neural information processing systems, 27.
- Radford, A., Metz, L. and Chintala, S., 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434
- Arjovsky, M., Chintala, S. and Bottou, L., 2017, July. Wasserstein generative adversarial networks. In International conference on machine learning (pp. 214-223). PMLR.
- Dumoulin, V., Belghazi, I., Poole, B., Mastropietro, O., Lamb, A., Arjovsky, M. and Courville, A., 2016. Adversarially learned inference. arXiv preprint arXiv:1606.00704.
- Donahue, J., Krähenbühl, P. and Darrell, T., 2016. Adversarial feature learning. arXiv preprint arXiv:1605.09782.
- Isola, P., Zhu, J.Y., Zhou, T. and Efros, A.A., 2017. Image-to-image translation with conditional adversarial networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1125-1134).
- Zhu, J.Y., Park, T., Isola, P. and Efros, A.A., 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In Proceedings of the IEEE international conference on computer vision (pp. 2223-2232).
- Pathak, D., Krahenbuhl, P., Donahue, J., Darrell, T. and Efros, A.A., 2016. Context encoders: Feature learning by inpainting. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2536-2544).
- Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M. and Lempitsky, V., 2016. Domain-adversarial training of neural networks. The journal of machine learning research, 17(1), pp.2096-2030.
- Mirza, M. and Osindero, S., 2014. Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784.
- Karras, T., Aila, T., Laine, S. and Lehtinen, J., 2017. Progressive growing of gans for improved quality, stability, and variation. arXiv preprint arXiv:1710.10196.
- Karras, T., Laine, S. and Aila, T., 2019. A style-based generator architecture for generative adversarial networks. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 4401-4410).

References

- Odena, A., Olah, C. and Shlens, J., 2017, July. Conditional image synthesis with auxiliary classifier gans. In *International conference on machine learning* (pp. 2642-2651). PMLR.
- Miyato, T. and Koyama, M., 2018. cGANs with projection discriminator. *arXiv preprint arXiv:1802.05637*.
- Zhang, H., Goodfellow, I., Metaxas, D. and Odena, A., 2019, May. Self-attention generative adversarial networks. In *International conference on machine learning* (pp. 7354-7363). PMLR.
- Brock, A., Donahue, J. and Simonyan, K., 2018. Large scale GAN training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*.
- Brock, A., Donahue, J. and Simonyan, K., 2018. Large scale GAN training for high fidelity natural image synthesis. arXiv preprint arXiv:1809.11096.
- Schlegl, T., Seeböck, P., Waldstein, S.M., Schmidt-Erfurth, U. and Langs, G., 2017, June. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In International conference on information processing in medical imaging (pp. 146-157). Springer, Cham.
- Yang, D., Xu, D., Zhou, S.K., Georgescu, B., Chen, M., Grbic, S., Metaxas, D. and Comaniciu, D., 2017, September. Automatic liver segmentation using an adversarial image-to-image network. In International conference on medical image computing and computer-assisted intervention (pp. 507-515). Springer, Cham.
- Dai, W., Dong, N., Wang, Z., Liang, X., Zhang, H. and Xing, E.P., 2018. Scan: Structure correcting adversarial network for organ segmentation in chest x-rays. In Deep learning in medical image analysis and multimodal learning for clinical decision support (pp. 263-273). Springer, Cham.
- Costa, P., Galdran, A., Meyer, M.I., Abramoff, M.D., Niemeijer, M., Mendonça, A.M. and Campilho, A., 2017. Towards adversarial retinal image synthesis. arXiv preprint arXiv:1701.08974.
- Wolterink, J.M., Dinkla, A.M., Savenije, M.H., Seevinck, P.R., van den Berg, C.A. and Išgum, I., 2017, September. Deep MR to CT synthesis using unpaired data. In International workshop on simulation and synthesis in medical imaging (pp. 14-23). Springer, Cham.
- Havaei, M., Mao, X., Wang, Y. and Lao, Q., 2021. Conditional generation of medical images via disentangled adversarial inference. Medical Image Analysis, 72, p.102106.