Variational AutoEncoder

& Generative Models

By: Shai Harel, structured data vision team

What we’ll see today

- Generative vs. Discriminative models [1]
- VAE Algorithm Overview [2]
- Putting it to work - Semi-supervised [3]

[1] Deep Neural Networks are Easily Fooled
[2] Auto-Encoding Variational Bayes
[3] Semi-Supervised Learning with Deep Generative Models

What we'll *NOT* see today

$$\log p(\mathbf{X}) = \sum_{i=1}^{N} \log p(\mathbf{x}^{(i)}), \tag{1}$$

but in general this marginal likelihood is intractable to compute or differentiate directly for flexible generative models that have high-dimensional latent variables and flexible priors and likelihoods. A solution is to introduce $q(\mathbf{z}|\mathbf{x})$, a parametric *inference model* defined over the latent variables, and optimize the *variational lower bound* on the marginal log-likelihood of each observation $\mathbf{x}$:

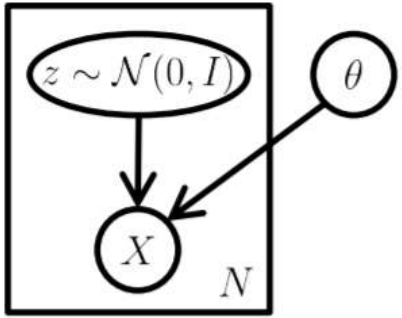
$$\log p(\mathbf{x}) \geq \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \left[\log p(\mathbf{x}, \mathbf{z}) - \log q(\mathbf{z}|\mathbf{x})\right] = \mathcal{L}(\mathbf{x}; \theta) \tag{2}$$

where θ indicates the parameters of p and q models.

There are various ways to optimize the lower bound $\mathcal{L}(\mathbf{x}; \theta)$; for continuous $\mathbf{z}$ it can be done efficiently through a re-parameterization of $q(\mathbf{z}|\mathbf{x})$ (Kingma & Welling, 2013; Rezende et al., 2014).

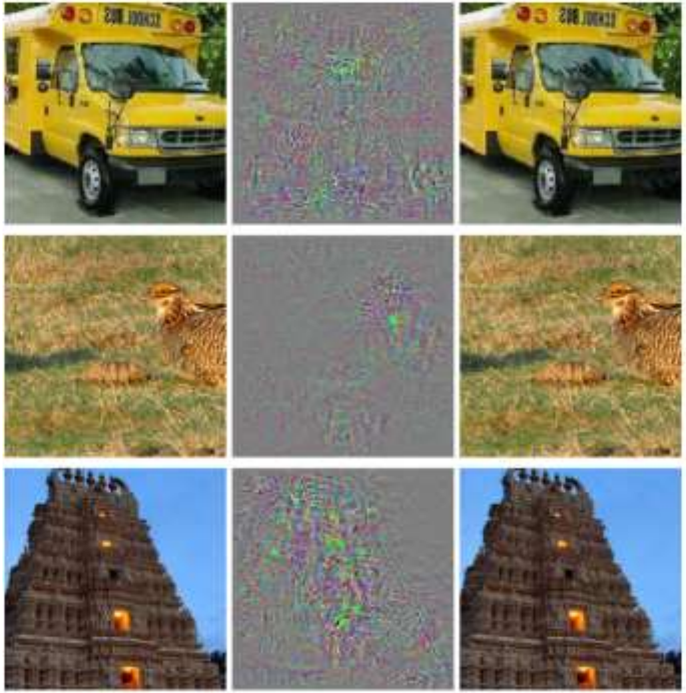
This way of optimizing the variational lower bound with a parametric inference network and re-parameterization of continuous latent variables is usually called VAE. The "autoencoding" terminology comes from the fact that the lower bound $\mathcal{L}(\mathbf{x}; \theta)$ can be re-arranged:

$$\begin{aligned} \mathcal{L}(\mathbf{x}; \theta) &= \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \left[\log p(\mathbf{x}, \mathbf{z}) - \log q(\mathbf{z}|\mathbf{x})\right] && (3) \\ &= \mathbb{E}_{q(\mathbf{z}|\mathbf{x})} \left[\log p(\mathbf{x}|\mathbf{z})\right] - D_{KL}(q(\mathbf{z}|\mathbf{x}) || p(\mathbf{z})) && (4) \end{aligned}$$



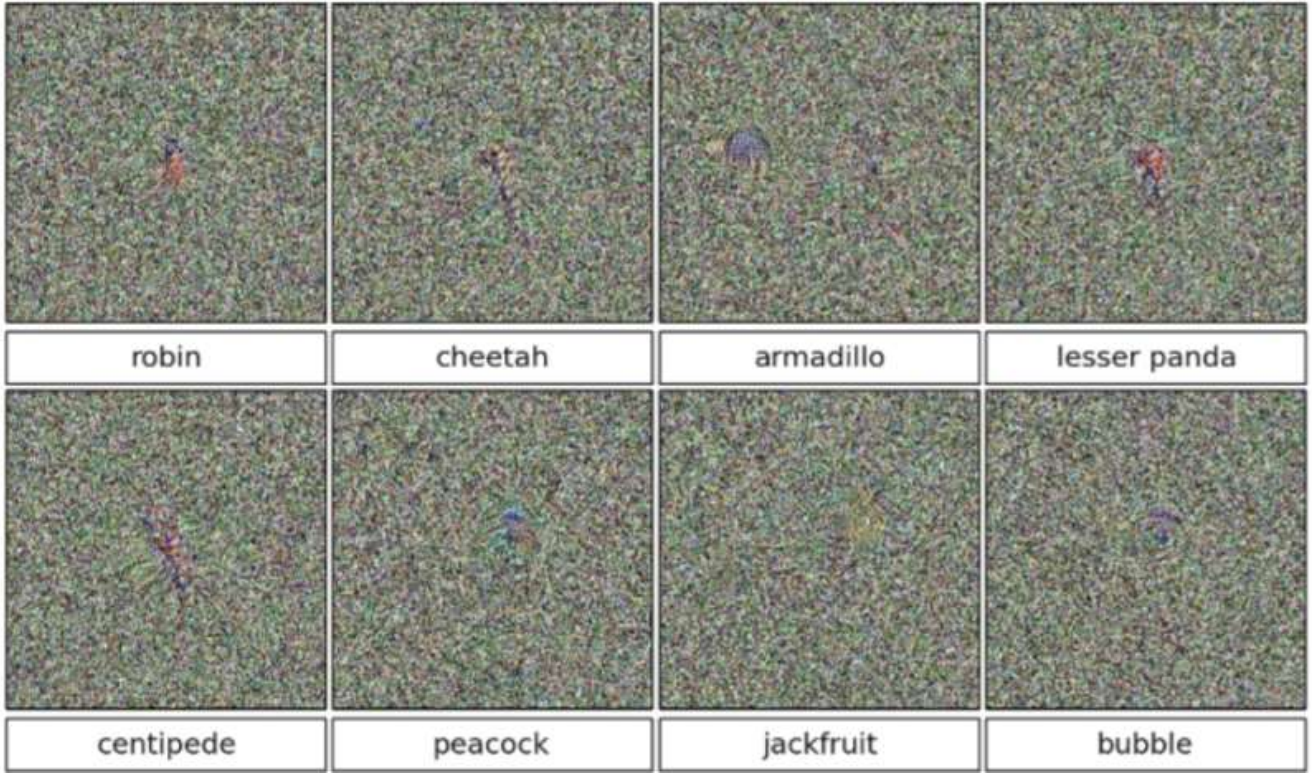
$$\begin{aligned} L &= \log(p(x)) \\ &= \sum_z q(z|x) \log(p(x)) \\ &= \sum_z q(z|x) \log\left(\frac{p(z,x)}{p(z|x)}\right) \\ &= \sum_z q(z|x) \log\left(\frac{p(z,x)}{q(z|x)} \frac{q(z|x)}{p(z|x)}\right) \\ &= \sum_z q(z|x) \log\left(\frac{p(z,x)}{q(z|x)}\right) + \sum_z q(z|x) \log\left(\frac{q(z|x)}{p(z|x)}\right) \\ &= L^v + D_{KL}(q(z|x) \| p(z|x)) \\ &\geq L^v \end{aligned}$$

Why Gen. Models?

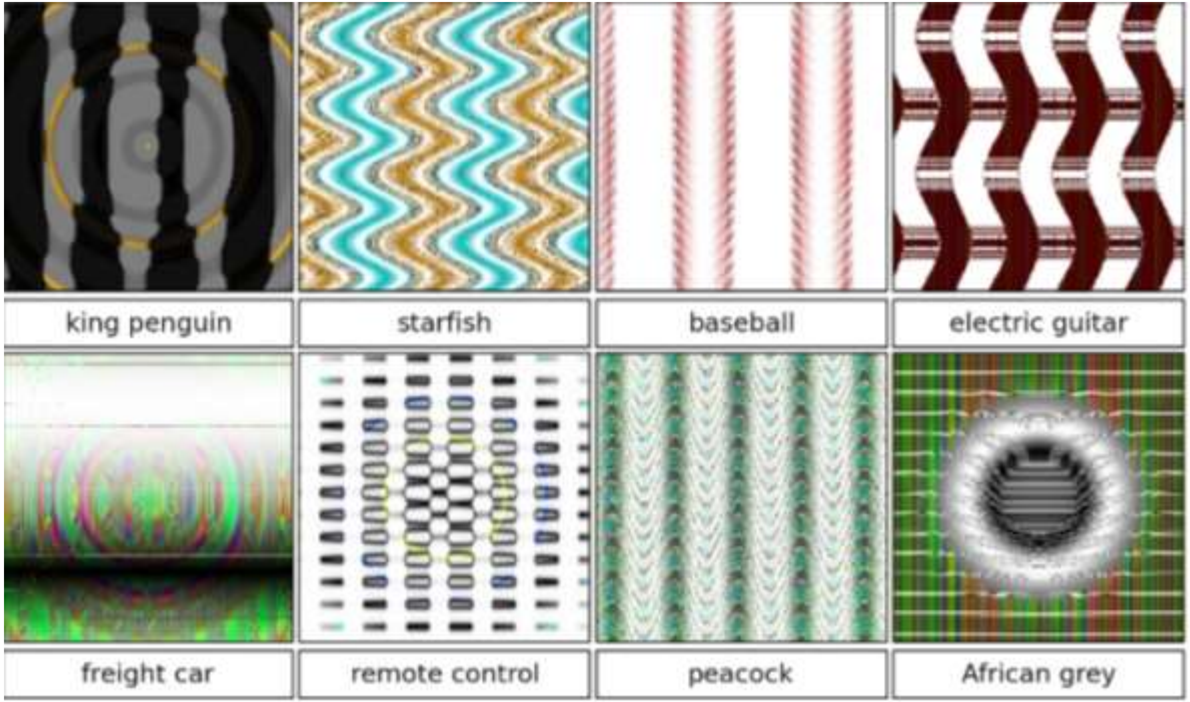


99.99%
ostrich, Struthio camelus

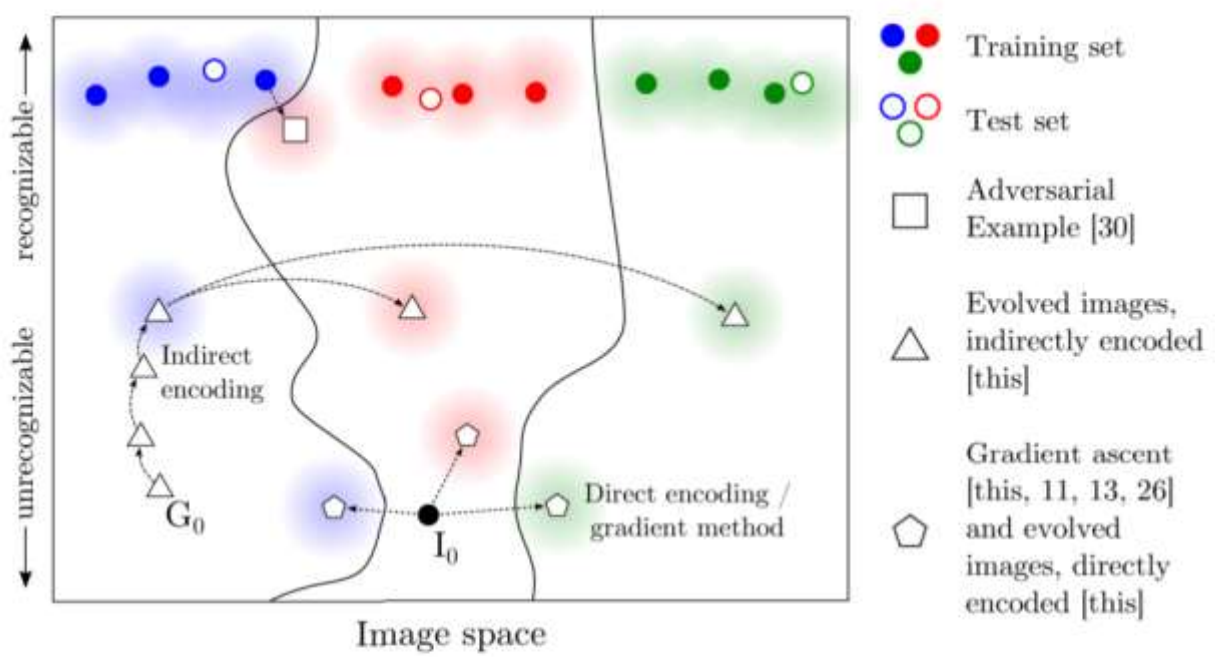
Why Gen. Models?



Why Gen. Models?



Why Gen. Models?

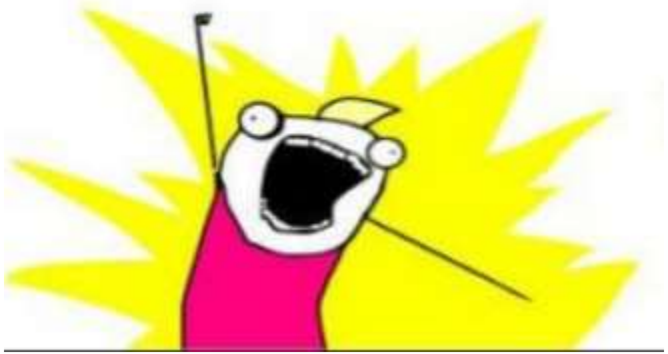


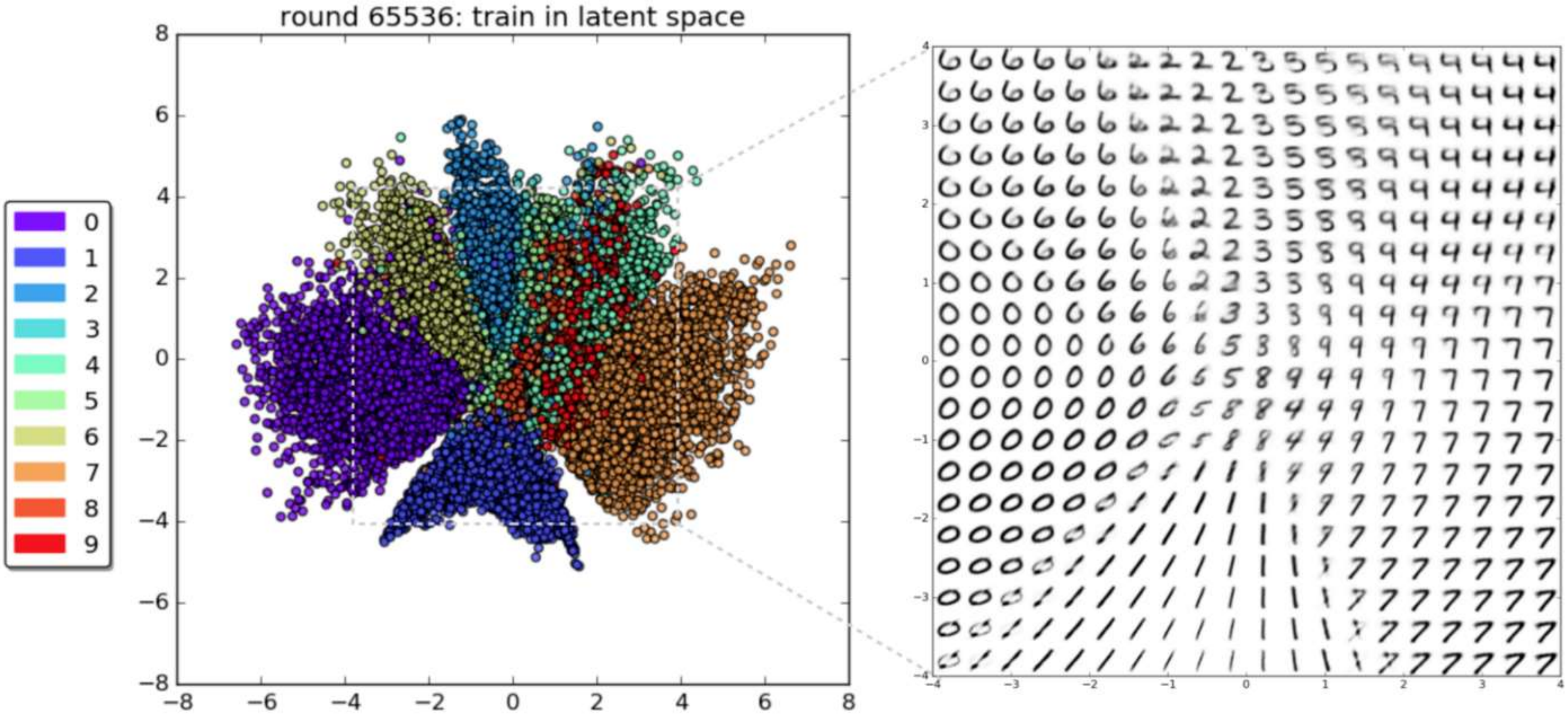
What do we want?

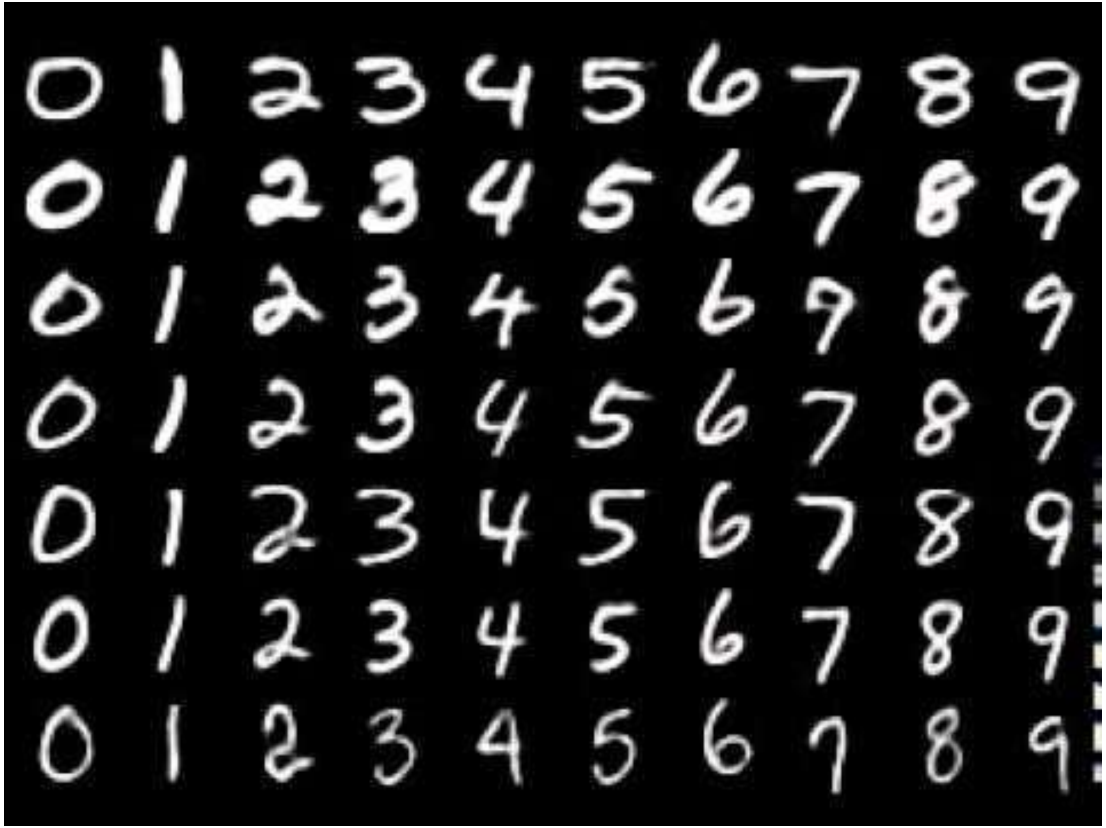
- Generative model
- “Structure constraint” on latent space

Why?

- Semi-Supervised learning
- Visualize z-space
- Not so easily fooled
- More...




round 65536: train in latent space
0
1
2
3
4
5
6
7
8
9

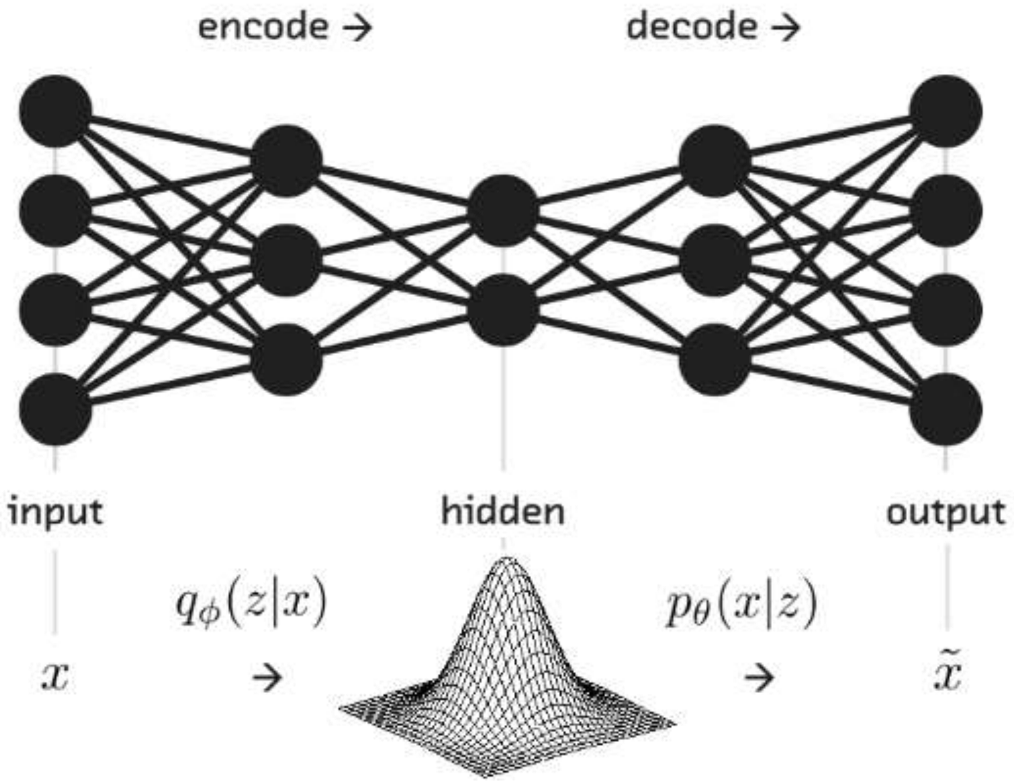

0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9

AutoEncoder Attempt #1

- Encoder
  q(z|x): get z given x

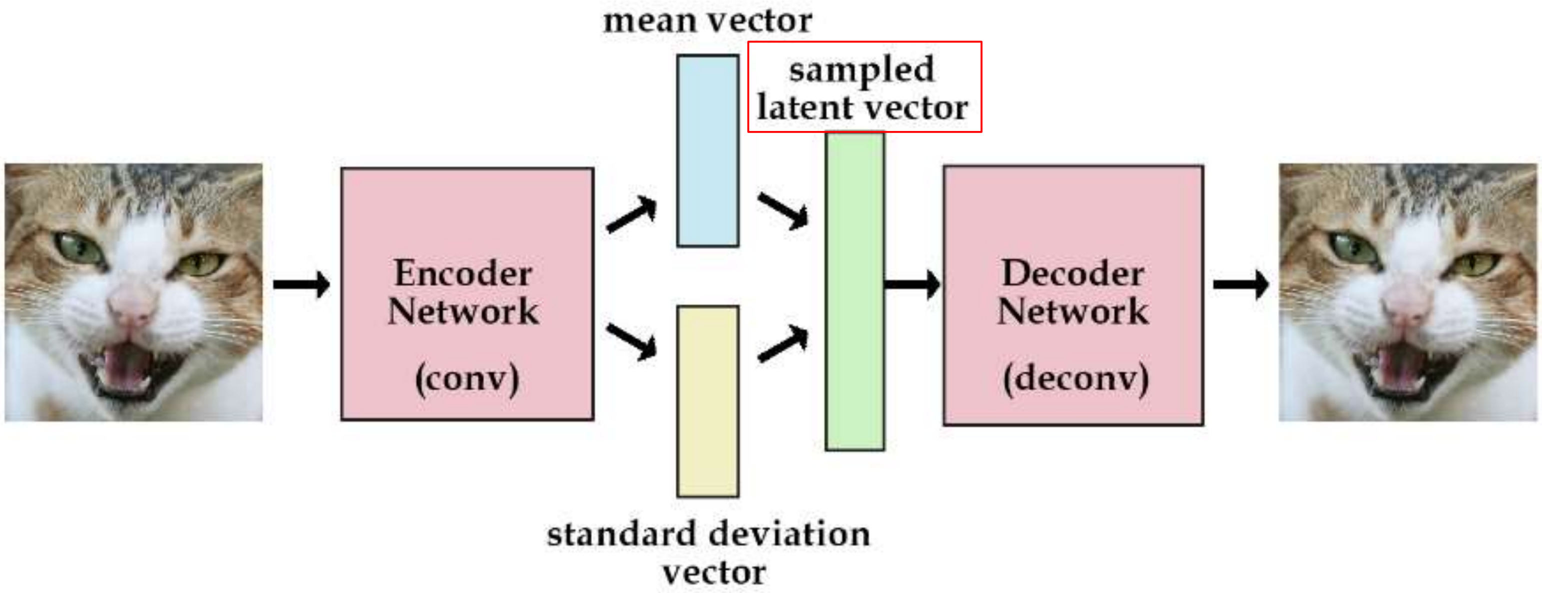
- Decoder
  p(x|z): get x given z

- What's the difference?



AutoEncoder Attempt #1

Problems?

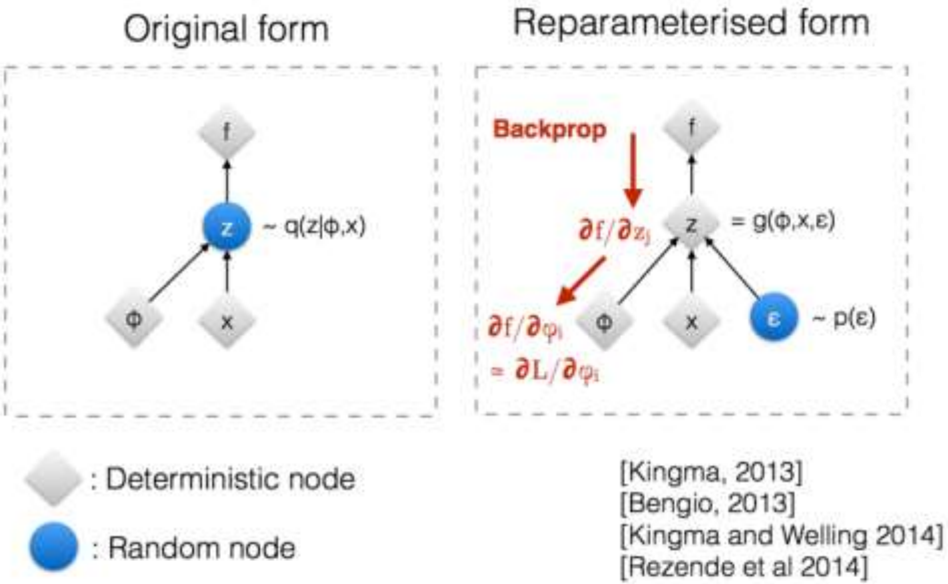


Re-Parameterization Trick

Backpropagation not possible through random sampling!

$$z^{(i,l)} = \mu^{(i)} + \sigma^{(i)} \odot \varepsilon_i$$

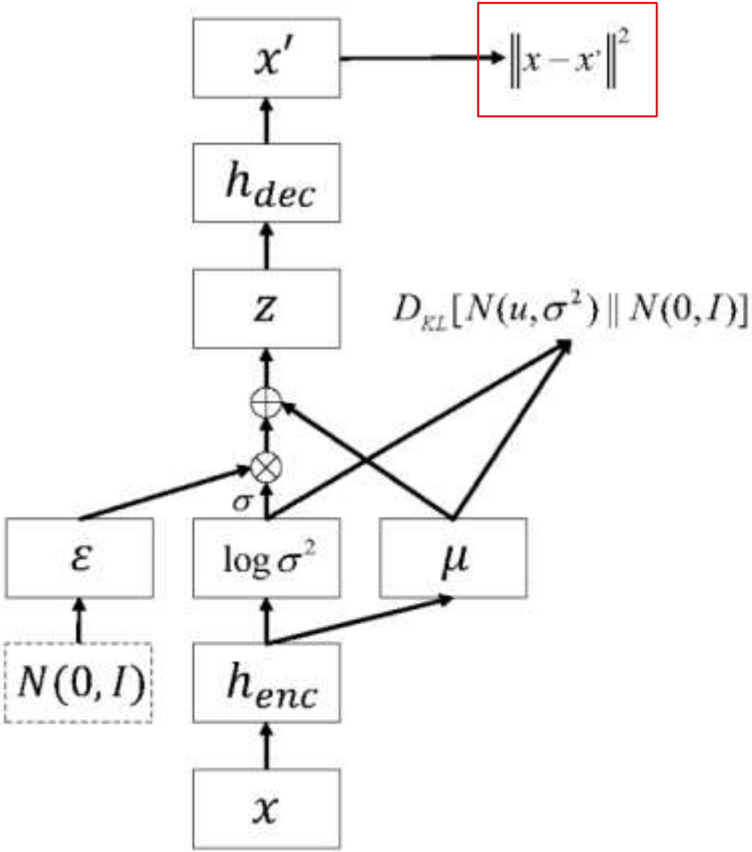
$$\varepsilon_i \sim N(0,1)$$



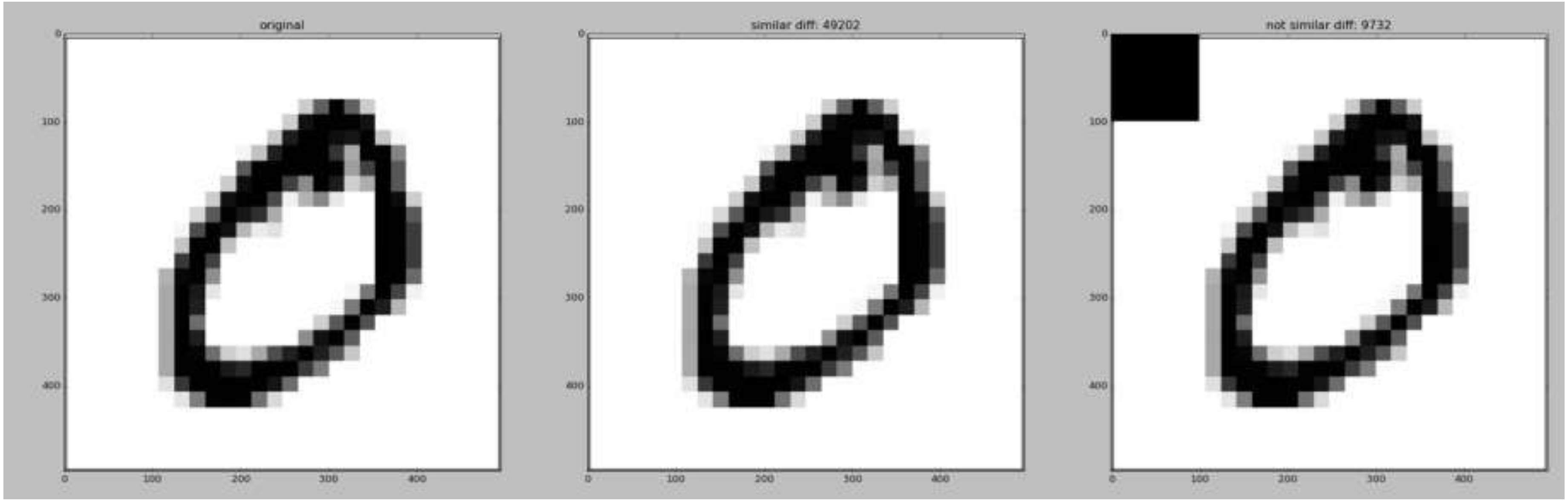
[https://arxiv.org/abs/1609.04468]

Net Structure

- More problems?



L2 distance in pixel space

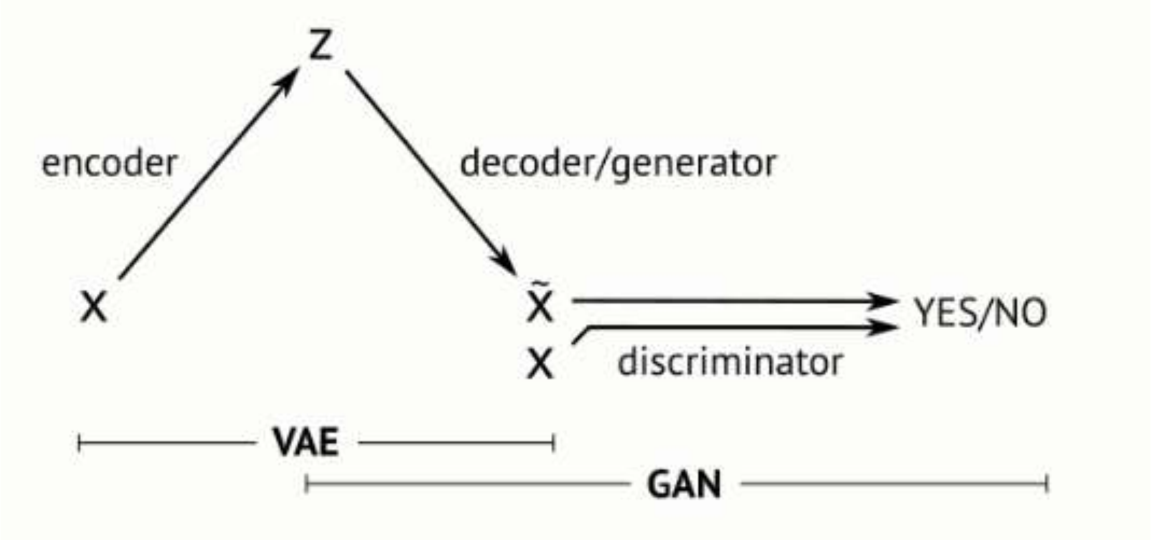


GAN - Generative Adversarial Networks

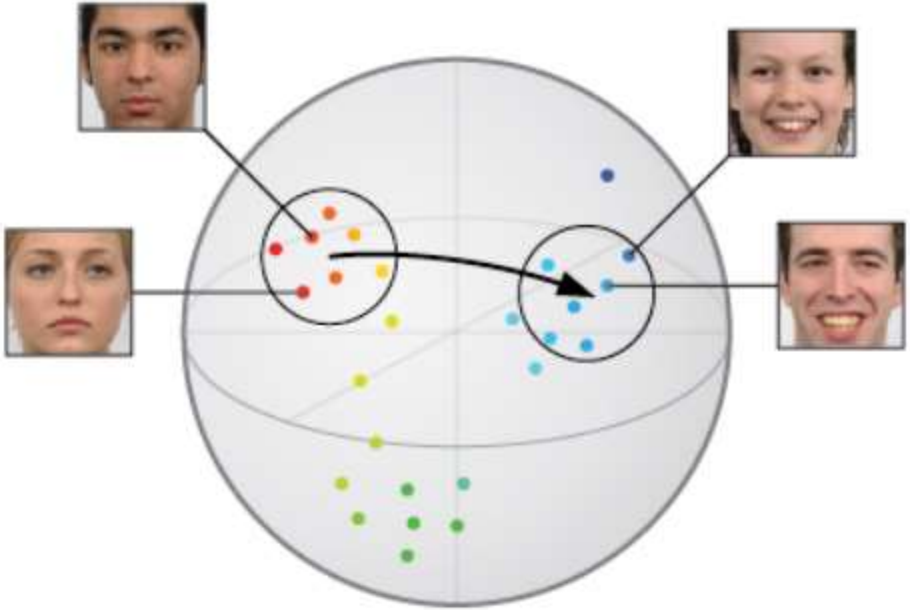
“You pit a generative (G) machine against a discriminative (D) machine and make them fight.”

http://soumith.ch/eyescream/

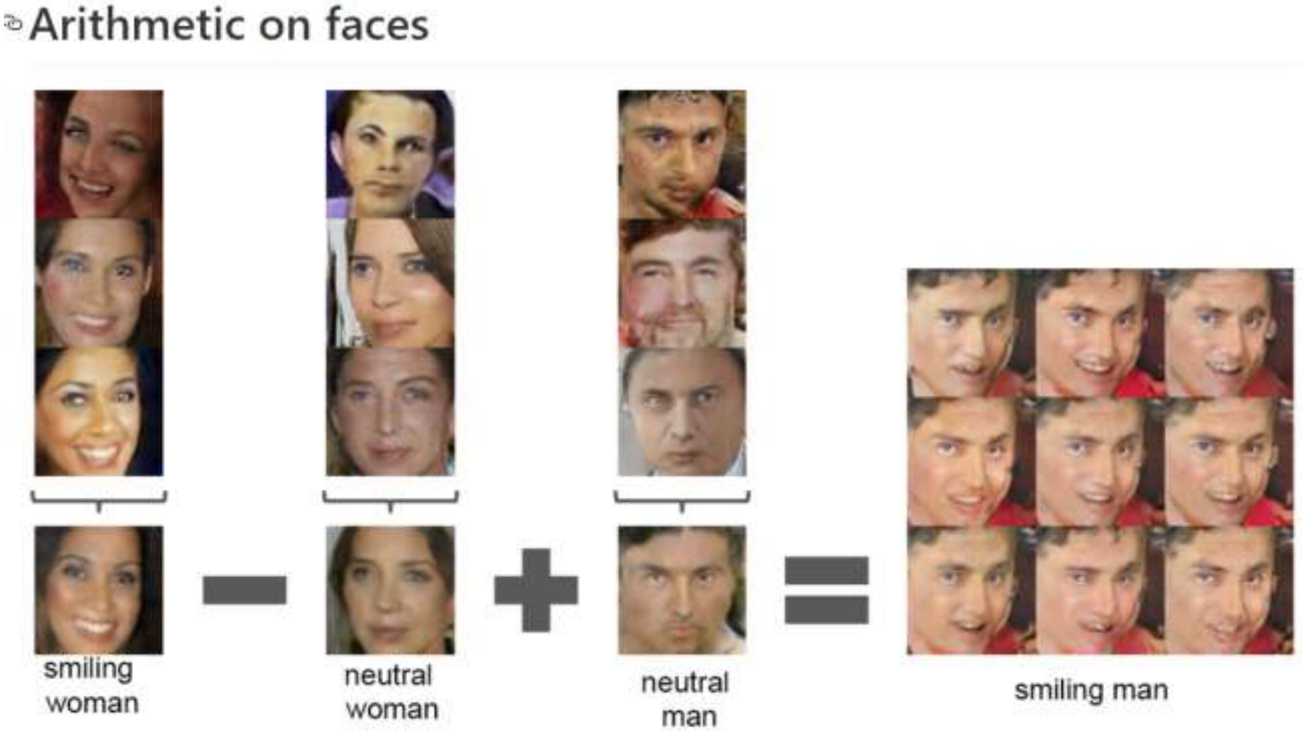
GAN - learn the loss



If you do it right!

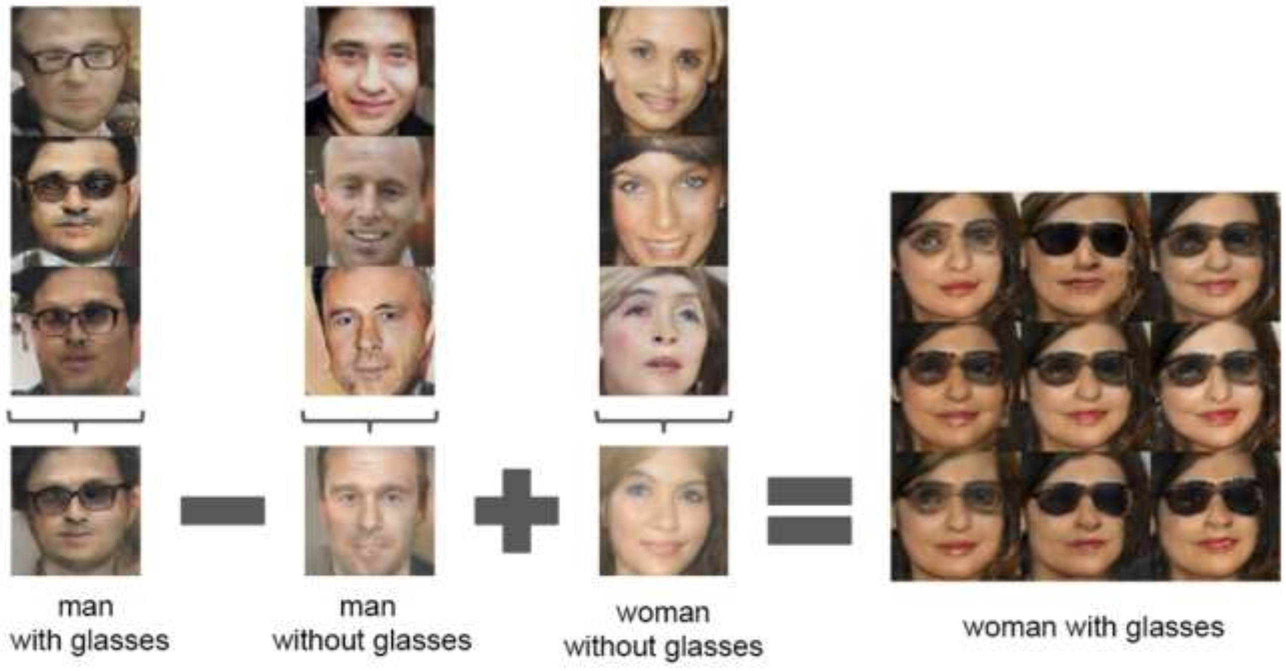


If you do it right!



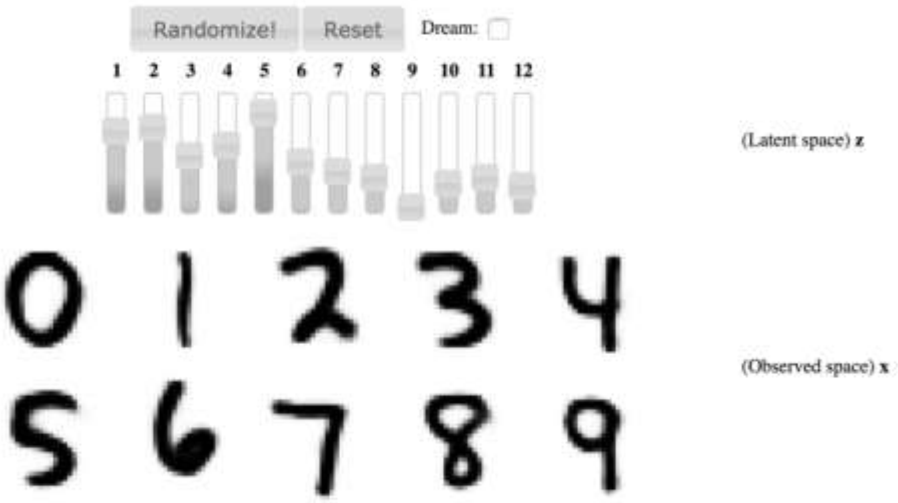
If you do it right!

Arithmetic on faces



If you do it right!

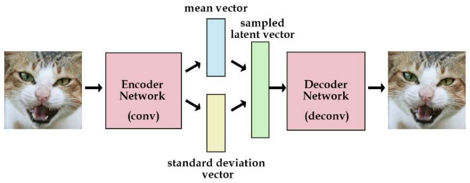
http://www.dpkingma.com/sgvb_mnist_demo/demo.html



Take Aways

- Employ Structure, It’s cool
- GAN may be your next loss function
  - Super Res
  - Pixel Level Seg.
  - AutoEncoder (we saw it today)
- Re-Parameterization Trick

  ----Personal takeaw



- Don’t give up when r 1000)

References

- https://github.com/oduerr/dl_tutorial/blob/master/tensorflow/vae/vae_demo-2D.ipynb
- https://home.zhaw.ch/~dueo/bbs/files/vae.pdf
- http://kvfrans.com/variational-autoencoders-explained/
- http://blog.fastforwardlabs.com/2016/08/12/introducing-variational-autoencoders-in-prose-and.html
- http://blog.fastforwardlabs.com/2016/08/22/under-the-hood-of-the-variational-autoencoder-in.html
- https://github.com/Newmu/dcgan_code
- http://torch.ch/blog/2015/11/13/gan.html
- https://github.com/soumith/ganhacks
- https://research.fb.com/wp-content/uploads/2016/11/luc16wat.pdf
- https://arxiv.org/pdf/1412.1897.pdf

Re-Parameterization Trick

Sampling (reparametrization trick)

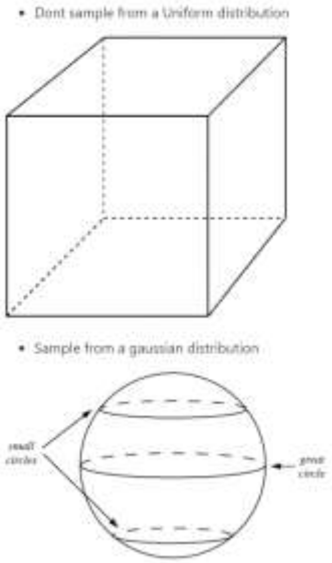
$$z^{(i,l)} \sim N(\mu^{(i)}, \sigma^{2(i)})$$

$$z^{(i,l)} = \mu^{(i)} + \sigma^{(i)} \odot \varepsilon_i \quad \varepsilon_i \sim N(0,1)$$

Writing z in this form, results in a deterministic part and noise.

Cannot back propagate through a random drawn number

z has the same distribution, but now one can back propagate.



[https://arxiv.org/abs/1609.04468]