UNIVERSITY OF MINNESOTA
Driven to Discover℠

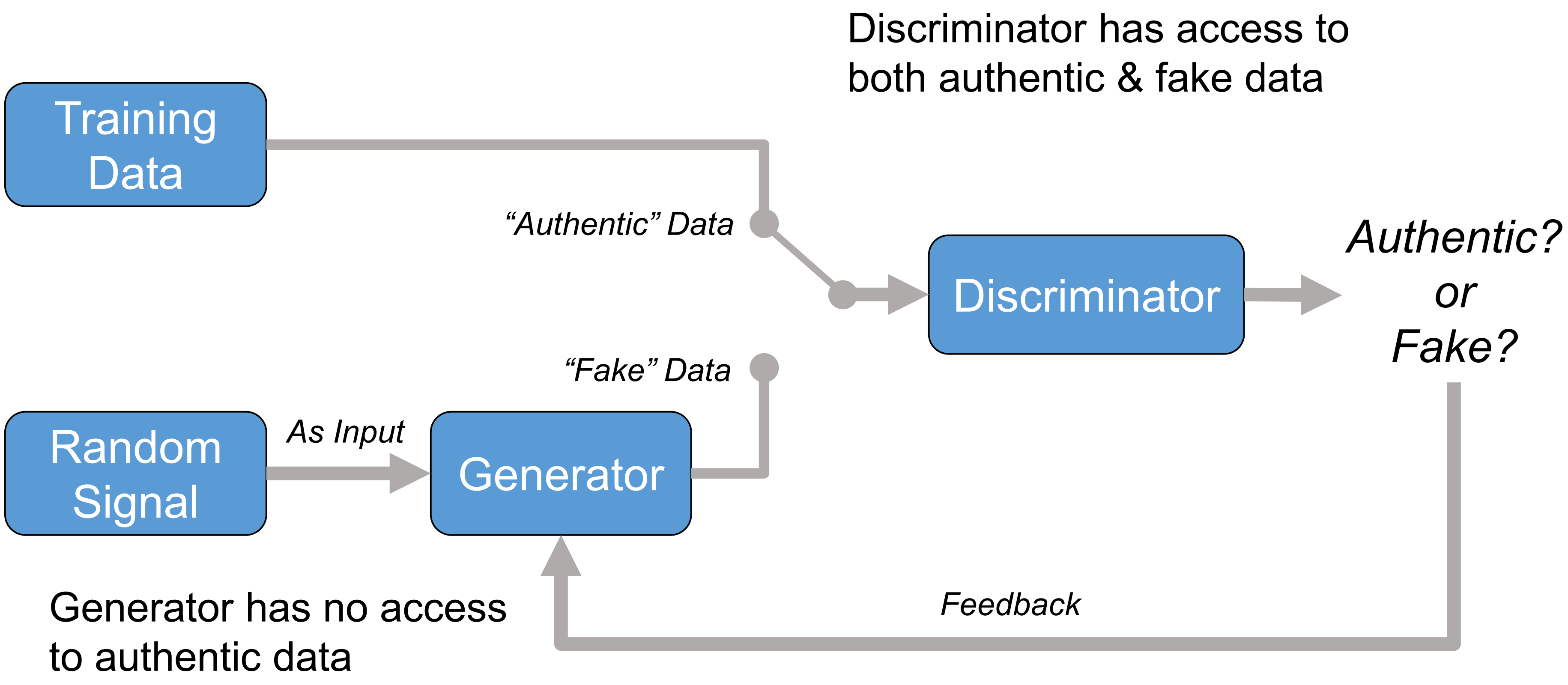
EE 5561: Image Processing and Applications

Lecture 26

Mehmet Akçakaya

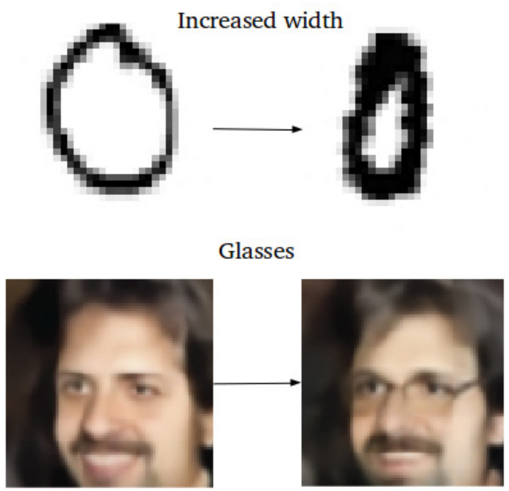
Recap of Last Lecture

- Generative Adversarial Networks
  - Generate random samples **x** by exciting the generator network with random noise samples **z**



Another Generative Model

- Today: Another generative model
  - VAEs
  - Generate random samples **x** by exciting the generator network with random variables *whose PDF has been learned from data*
  - Allows you to *alter/create variations* of data you already have



https://arxiv.org/pdf/1711.05772.pdf

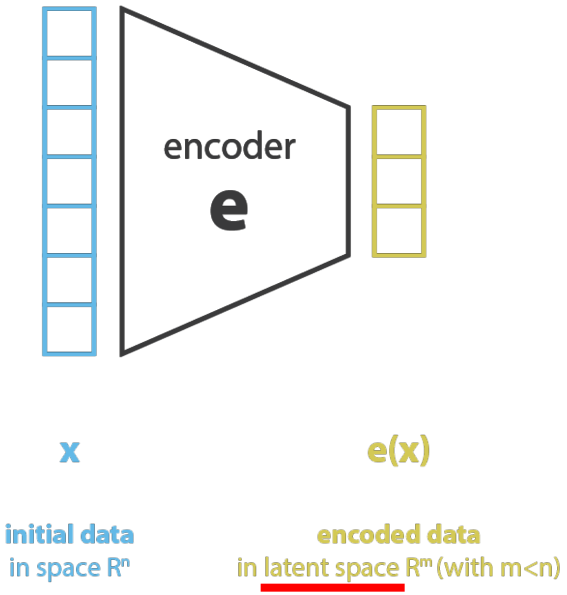
Background: Dimensionality Reduction

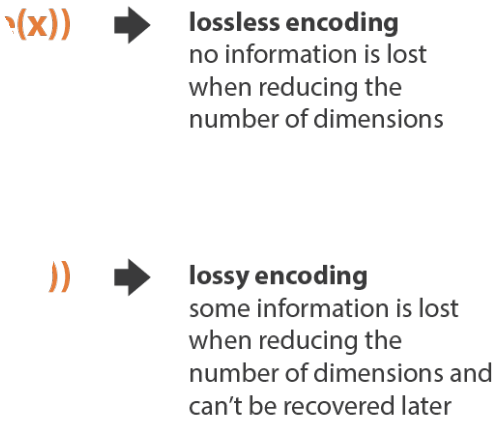
- Dimensionality reduction
  - Process of reducing the number of features that describe some data
  - Selection → Only some existing features are conserved
  - Extraction → Reduced number of new features created
  - Useful in visualization, storage, …

- General idea:
  - Encoder: Produce “new features” from data
  - Decoder: Reverse the process

Background: Dimensionality Reduction

- Encoder – decoder setup





https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

Background: Dimensionality Reduction

- Encoder – decoder setup
  - Find the best encoder-decoder pair among a class of such pairs

$$(e^*, d^*) = \underset{(e,d)\in E\times D}{\arg\min}\ \epsilon(x, d(e(x)))$$

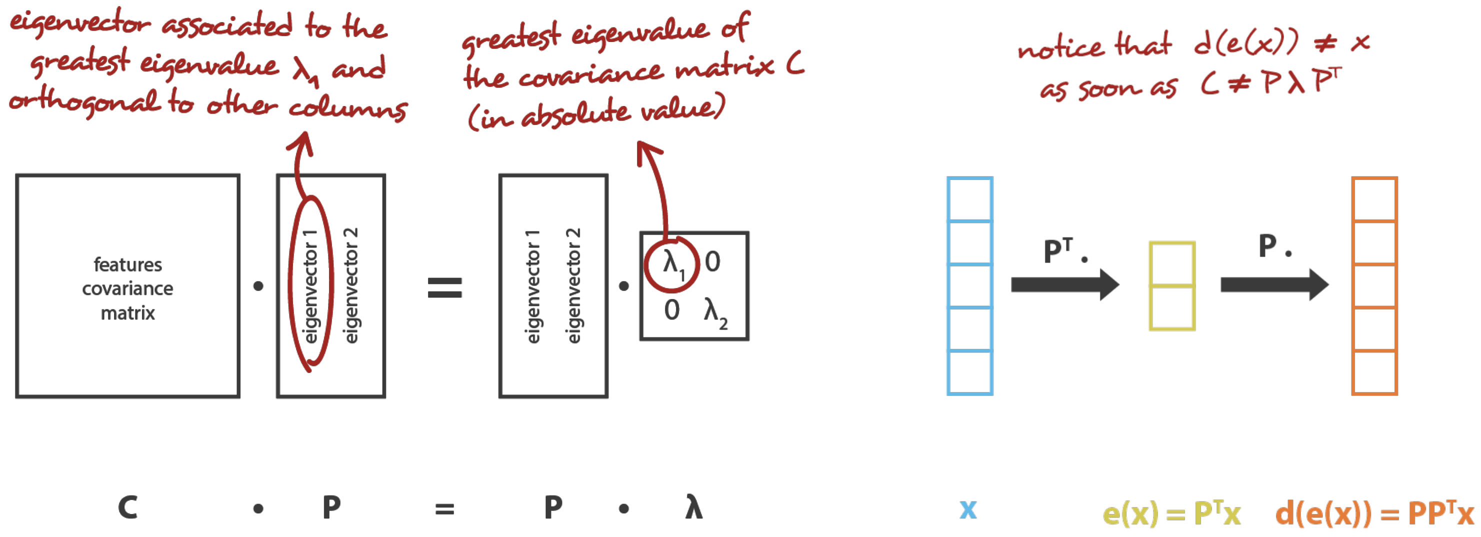
some error metric

Background: Dimensionality Reduction

- Principal Components Analysis (PCA)
  - Arguably the most commonly used dimensionality reduction technique
  - Build new *independent* features that are *linear combinations* of the old features
  - Projections of data on the *subspace* defined by new features are close to original data
  - Find the best linear subspace of the initial space

Background: Dimensionality Reduction

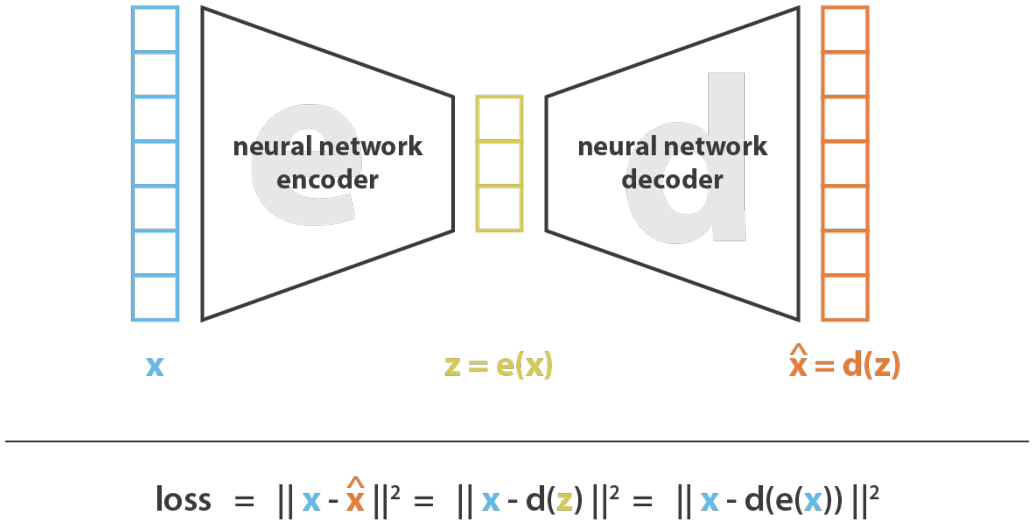
- Principal Components Analysis (PCA)
  - Encoder: matrix (linear transformation) with orthonormal rows
  - Solution: Orthonormal eigenvector corresponding to *m* largest eigenvalues of the covariance features matrix → best subspace



https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

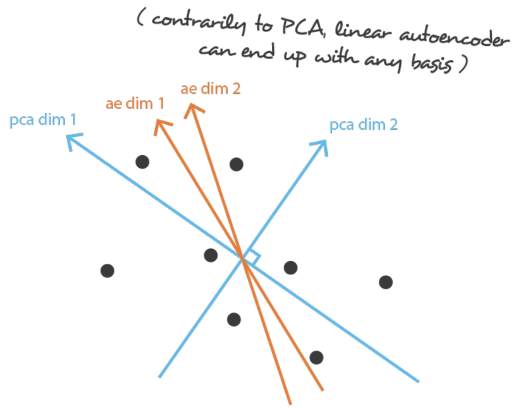
Background: Dimensionality Reduction

- Autoencoders
  - Same idea of encoder-decoder
  - But use neural networks for encoder & decoder
  - Train with backpropagation etc



Background: Dimensionality Reduction

- Autoencoders
  - If encoder & decoder are 1-layer with no non-linearity, then we have the same objective as PCA
  - Unlike PCA, the solution that we get via gradient descent does not have to have orthogonal components
  - It could be another arbitrary basis for the same subspace



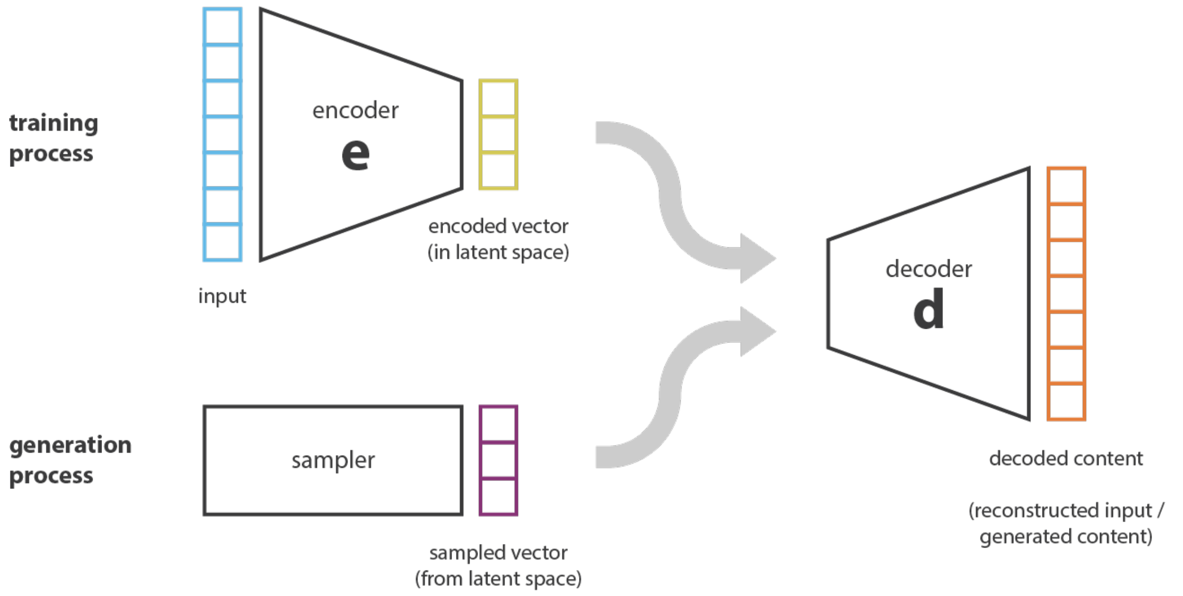
https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

Background: Dimensionality Reduction

- Autoencoders
  - If encoder & decoder are deep, then we need to be careful about “overfitting”
  - If there are enough parameters we can represent things in 1 dimensions
  - But in general we want to have some sort of interpretability/ structure in the “latent” space
  - Need to be careful about dimensions & # parameters

Variational Autoencoders

- Can we use autoencoders for generating content?
- How would we even do this?
  - Once the encoder-decoder is fixed, take a new sample in the latent space
  - Run it through the decoder



https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

Variational Autoencoders

- But when we use just MSE loss between input x and d(e(x)), we do not guarantee any structure on the latent space
  - For instance, are distances near-preserved?
  - Does interpolating two points in the latent space make sense?
- Consider the extreme case
  - Overfitting so that latent space is 1-dimensional
  - Points from the original data are placed on a line
  - Hard to organize anything on this line…

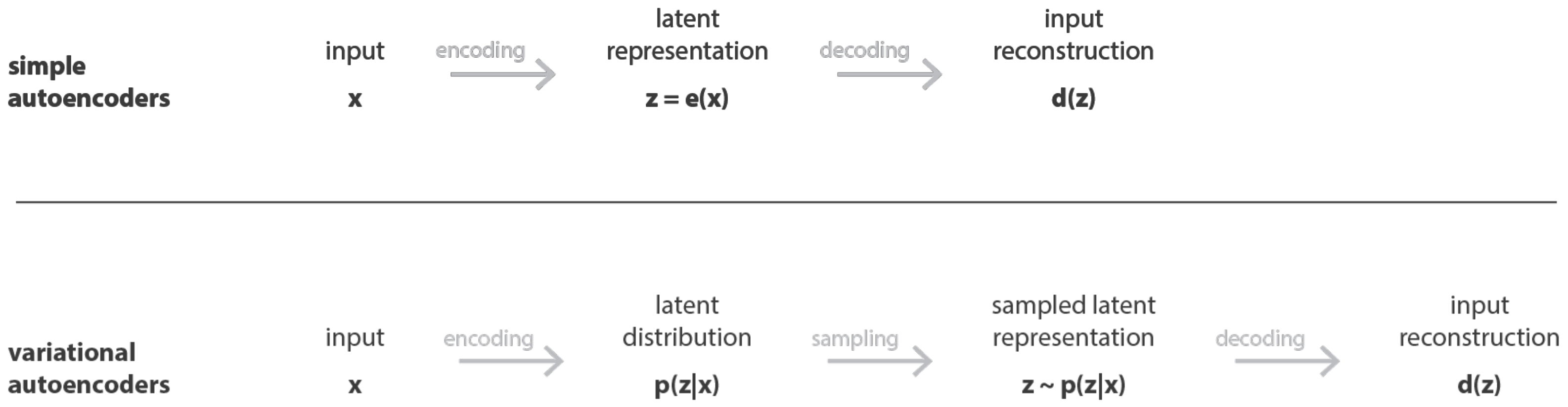
Variational Autoencoders

- This is where VAEs come in!
  - Essentially an autoencoder with regularized training
  - To avoid overfitting
  - To ensure latent space has “nice” properties

- Instead of encoding an input as a single point, VAEs encode it as a distribution over the latent space
  - Bayesian in nature

Variational Autoencoders

- VAE training (high-level)
  - Encode input as distribution over latent space
  - Sample a point in latent space with this distribution
  - Decode sampled point, calculate reconstruction error
  - Backpropagate



https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

Variational Autoencoders

- The ideal process:
  - We are given observations $\{\boldsymbol{x_k}\}_{k=1}^{N}$ which are samples of random vector $\boldsymbol{X}$
  - There is a distribution $p(\boldsymbol{z})$ on the latent space
  - New sample $\widehat{\boldsymbol{x}}$ is generated from a conditional PDF $p(\boldsymbol{x}|\boldsymbol{z_n})$
  - PDFs are parametrized by some parameters
  - $\boldsymbol{z_n}$ and these parameters are unknown
  - Learn these from observations $\{\boldsymbol{x_k}\}_{k=1}^{N}$

Variational Autoencoders

- Here the decoder is “probabilistic”
  - Naturally define as $p(\boldsymbol{x}|\boldsymbol{z})$

- The probabilistic encoder is similarly defined as $p(\boldsymbol{z}|\boldsymbol{x})$
- These are obviously related (Bayes’ rule)

$$p(\boldsymbol{z}|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|\boldsymbol{z})p(\boldsymbol{z})}{p(\boldsymbol{x})}$$

- To make things tractable, we will assume these distributions are Gaussian
  - Can be described fully by mean & covariance

Variational Autoencoders

- For the prior, $p(\boldsymbol{z})$, choose a standard Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$
  - No parameters to optimize
  - Without any info on $\boldsymbol{x}$, ~ treat latent space as noise (GANs)
- For $p(\boldsymbol{x}|\boldsymbol{z})$, choose another Gaussian distribution $\mathcal{N}(f(\mathbf{z}), c\mathbf{I})$
  - $f(\cdot)$ is a deterministic function (from a family of functions to be determined later)
  - c controls the variance
  - This is the probabilistic decoder
- Now we need $p(\boldsymbol{z}|\boldsymbol{x})$, but intractable via Bayes' rule

Variational Autoencoders

- Variational inference to the rescue!
  - Technique to approximate complex distributions
- VAEs approximate the posterior by another pdf $q(\boldsymbol{z}|\boldsymbol{x})$
  - Also a Gaussian distribution
  - $\mathcal{N}(g(\mathbf{x}), h(\mathbf{x}))$
  - Covariance matrix assumed to be diagonal
- How to choose the best such distribution?
  - Minimize KL divergence between pdfs from this family and the true distribution $p(\boldsymbol{z}|\boldsymbol{x})$

Variational Autoencoders

$$
\begin{aligned}
(g^*, h^*) &= \arg\min_{(g,h)} KL(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z}|\mathbf{x})) \\
&= \arg\min_{(g,h)} \int q(\mathbf{z}|\mathbf{x}) \log \frac{q(\mathbf{z}|\mathbf{x})}{p(\mathbf{z}|\mathbf{x})} d\mathbf{z} \\
&= \arg\min_{(g,h)} \int q(\mathbf{z}|\mathbf{x}) \log \frac{q(\mathbf{z}|\mathbf{x})p(\mathbf{x})}{p(\mathbf{x}|\mathbf{z})p(\mathbf{z})} d\mathbf{z} \\
&= \arg\min_{(g,h)} \int q(\mathbf{z}|\mathbf{x}) \log \frac{q(\mathbf{z}|\mathbf{x})}{p(\mathbf{x}|\mathbf{z})p(\mathbf{z})} d\mathbf{z} \\
&= \arg\min_{(g,h)} \int -q(\mathbf{z}|\mathbf{x}) \log p(\mathbf{x}|\mathbf{z}) d\mathbf{z} + \int q(\mathbf{z}|\mathbf{x}) \log \frac{q(\mathbf{z}|\mathbf{x})}{p(\mathbf{z})} d\mathbf{z} \\
&= \arg\min_{(g,h)} \int -q(\mathbf{z}|\mathbf{x}) \frac{-||\mathbf{x} - f(\mathbf{z})||_2^2}{2c} d\mathbf{z} + KL\big(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\big) d\mathbf{z} \\
&= \arg\min_{(g,h)} \int q(\mathbf{z}|\mathbf{x}) \frac{||\mathbf{x} - f(\mathbf{z})||_2^2}{2c} d\mathbf{z} + KL\big(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\big)
\end{aligned}
$$

Bayes rule

log $p(\mathbf{x})$ does not depend on $q(\mathbf{z}|\mathbf{x})$

- Related to evidence lower bound (ELBO): The final loss is negative of ELBO, which is a lower bound for log-evidence, log $p(\mathbf{x})$

Variational Autoencoders

$$\arg\min_{(g,h)} \int q(\mathbf{z}|\mathbf{x}) \frac{||\mathbf{x} - f(\mathbf{z})||_2^2}{2c} d\mathbf{z} + KL\big(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\big)$$

- First term: Likelihood of “observations” (expected log-likelihood)
- Second term: Stay close to prior distribution (regularization)
- So far *f*(·) assumed known & fixed → not true in practice
- $q(\boldsymbol{z}|\boldsymbol{x})$ depends on *f*(·) (through *g* and *h*)
- “Best” *f*(·) depends on $q(\boldsymbol{z}|\boldsymbol{x})$ (minimize first term for fixed *q*)
- So we actually have

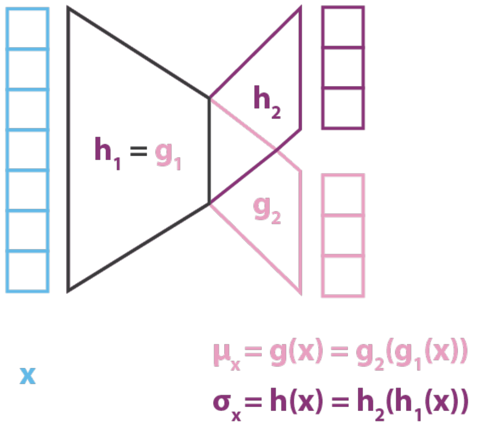
$$(f^*, g^*, h^*) = \arg\min_{(f,g,h)} \int q(\mathbf{z}|\mathbf{x}) \frac{||\mathbf{x} - f(\mathbf{z})||_2^2}{2c} d\mathbf{z} + KL\big(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\big)$$

Practical Implementation

- So far *f*, *g*, *h* are arbitrary functions
- Constrain them to be functions defined by neural networks
- In practice, have *g* and *h* share part of the architecture & weights

$$g(\mathbf{x}) = g_2(g_1(\mathbf{x})) \qquad h(\mathbf{x}) = h_2(h_1(\mathbf{x})) \qquad \text{where} \quad g_1(\mathbf{x}) = h_1(\mathbf{x})$$

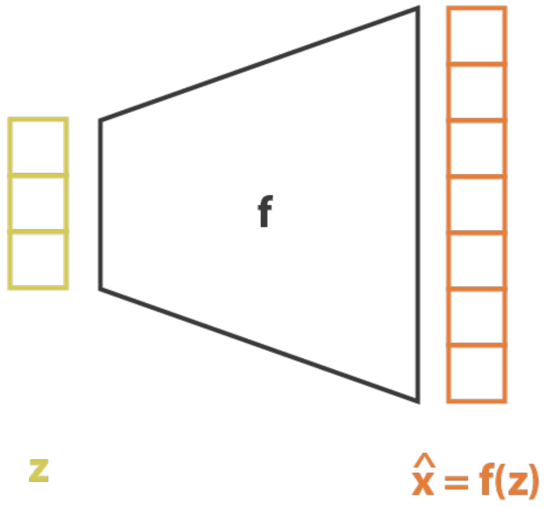
- Recall $h(\cdot)$ describes the diagonals of covariance matrix
- Encoder, $p(\boldsymbol{z}|\boldsymbol{x})$:



https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

Practical Implementation

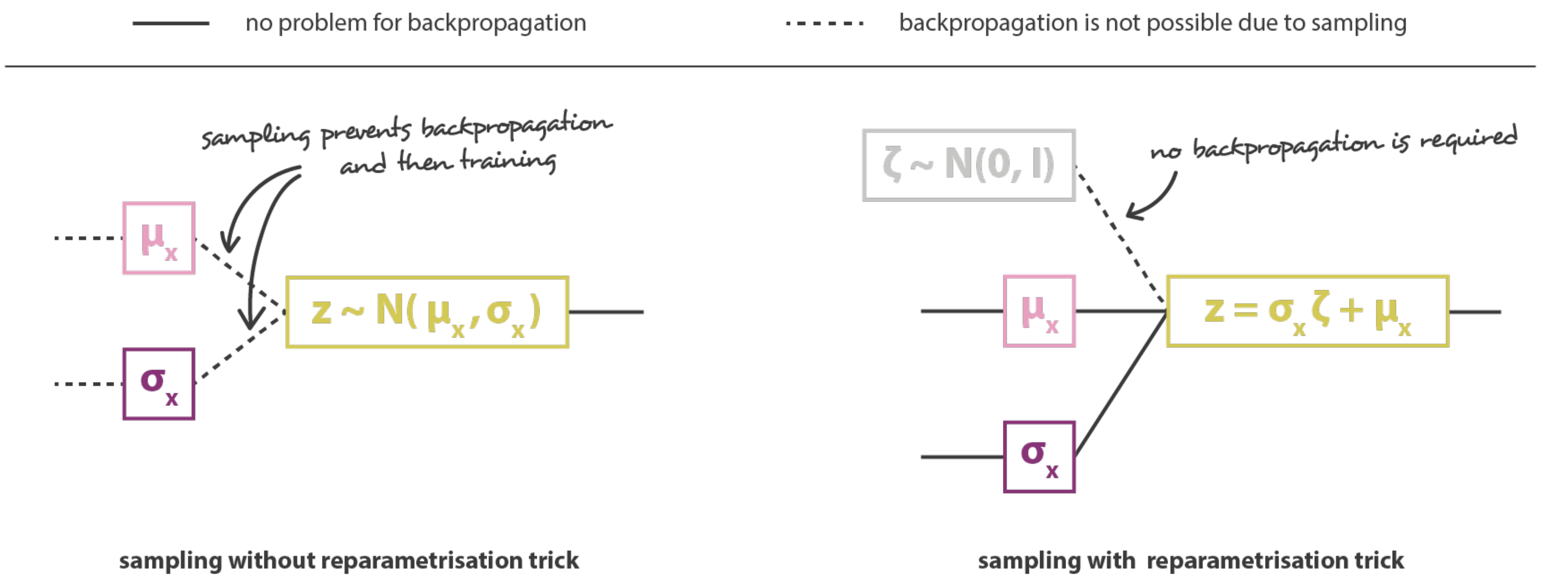
- Decoder, $p(\boldsymbol{x}|\boldsymbol{z})$ has fixed covariance, $\mathcal{N}(f(\mathbf{z}), c\mathbf{I})$
- The decoder defines this mean



Practical Implementation

- Still need one more ingredient: How do we backpropagate the sampling process in the latent space?
- The solution is the "reparatmetrization trick"

$$\mathbf{z} = h(\mathbf{x}) \odot \zeta + g(\mathbf{x}) \quad \text{where} \quad \zeta \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$



https://towardsdatascience.com/understanding-variational-autoencoders-vaes-f70510919f73

Practical Implementation

- Loss as an expectation in it (first term)

$$(f^*, g^*, h^*) = \arg \min_{(f,g,h)} \int q(\mathbf{z}|\mathbf{x}) \frac{||\mathbf{x} - f(\mathbf{z})||_2^2}{2c} d\mathbf{z} + KL\big(q(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\big)$$

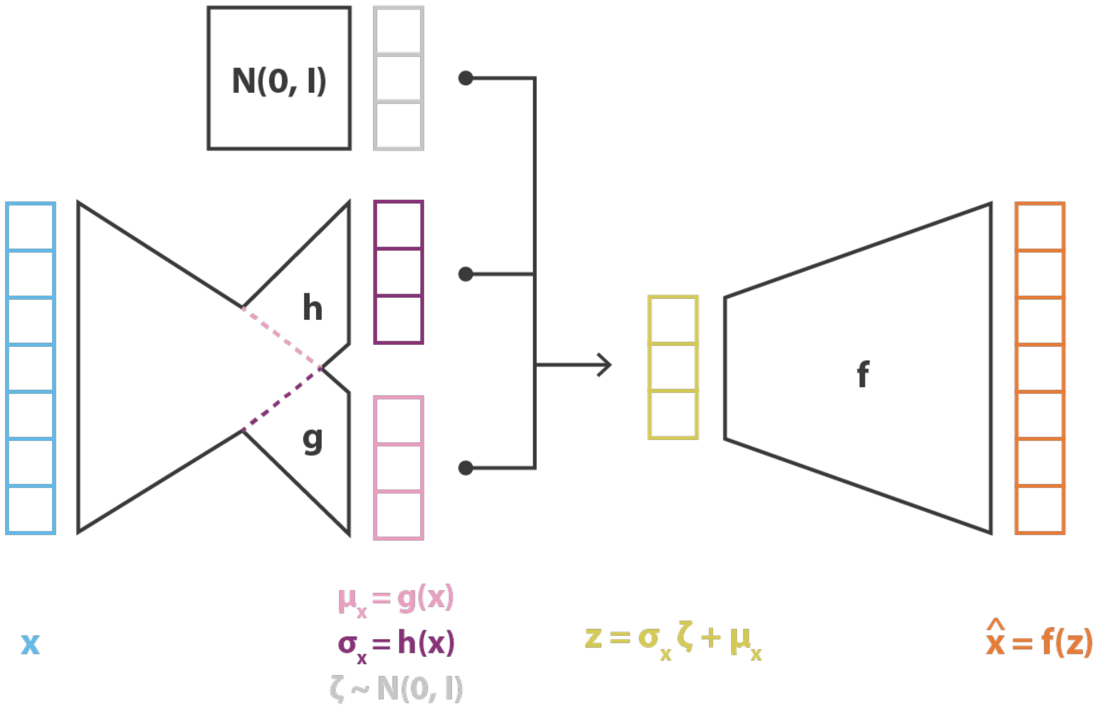
- Replace this with a Monte-Carlo approximation (~single draw)
- Also let $C = 1/(2c)$ for ease of notation
- We have our training loss function

$$(f^*, g^*, h^*) = \arg \min_{(f,g,h)} \frac{1}{N} \sum_{k=1}^{N} C||\mathbf{x}^n - f(\mathbf{z}^n)||_2^2 + KL\big(\mathcal{N}(g(\mathbf{x}^n), h(\mathbf{x}^n))||\mathcal{N}(\mathbf{0}, \mathbf{I})\big)$$

  - First term: reconstruction, second term: regularization, C: relative weights

Practical Implementation

- Final product



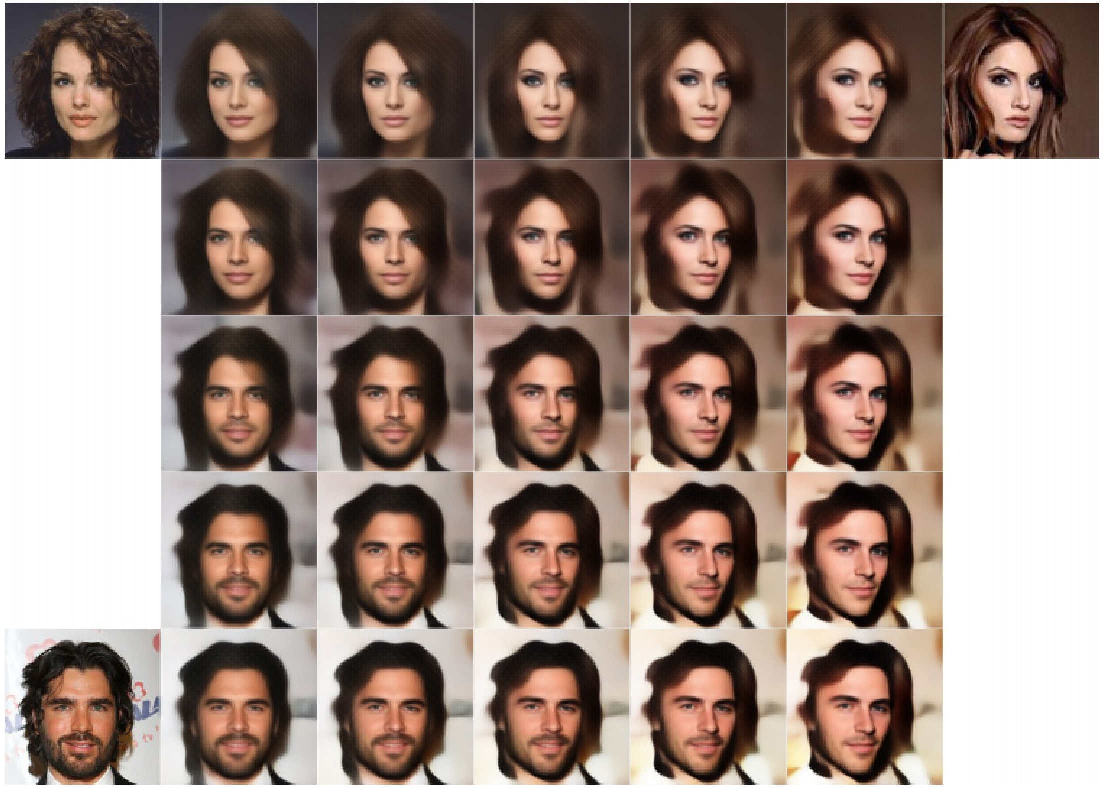
$$\text{loss} = C\,\|x - \hat{x}\|^2 + \mathrm{KL}[\,N(\mu_x, \sigma_x), N(0, I)\,] = C\,\|x - f(z)\|^2 + \mathrm{KL}[\,N(g(x), h(x)), N(0, I)\,]$$

Practical Implementation

- What happens at test time?
- For image generation: No test-image for generation
  - So no need for encoder
- Sample $\boldsymbol{z}$ from $\mathcal{N}(\mathbf{0}, \mathbf{I})$ and run it through the decoder

Some Applications

- Interpolation in latent space



https://arxiv.org/pdf/1609.04468.pdf

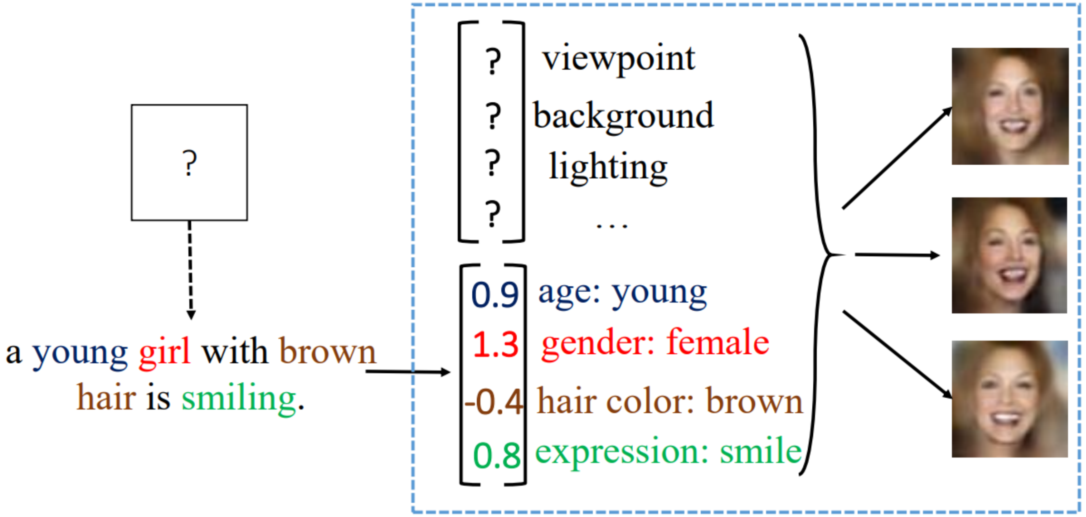
Some Applications

- Conditional VAEs
  - We can have other labels (**y**) to condition on
  - e.g. hair color, age, etc
  - Derivation stays the same
  - Replace all p(**x**|**z**) with p(**x**|**z**,**y**) and same for q(.|.)

https://arxiv.org/abs/1606.05908

Some Applications

- Conditional VAEs
  - One can condition on attributes

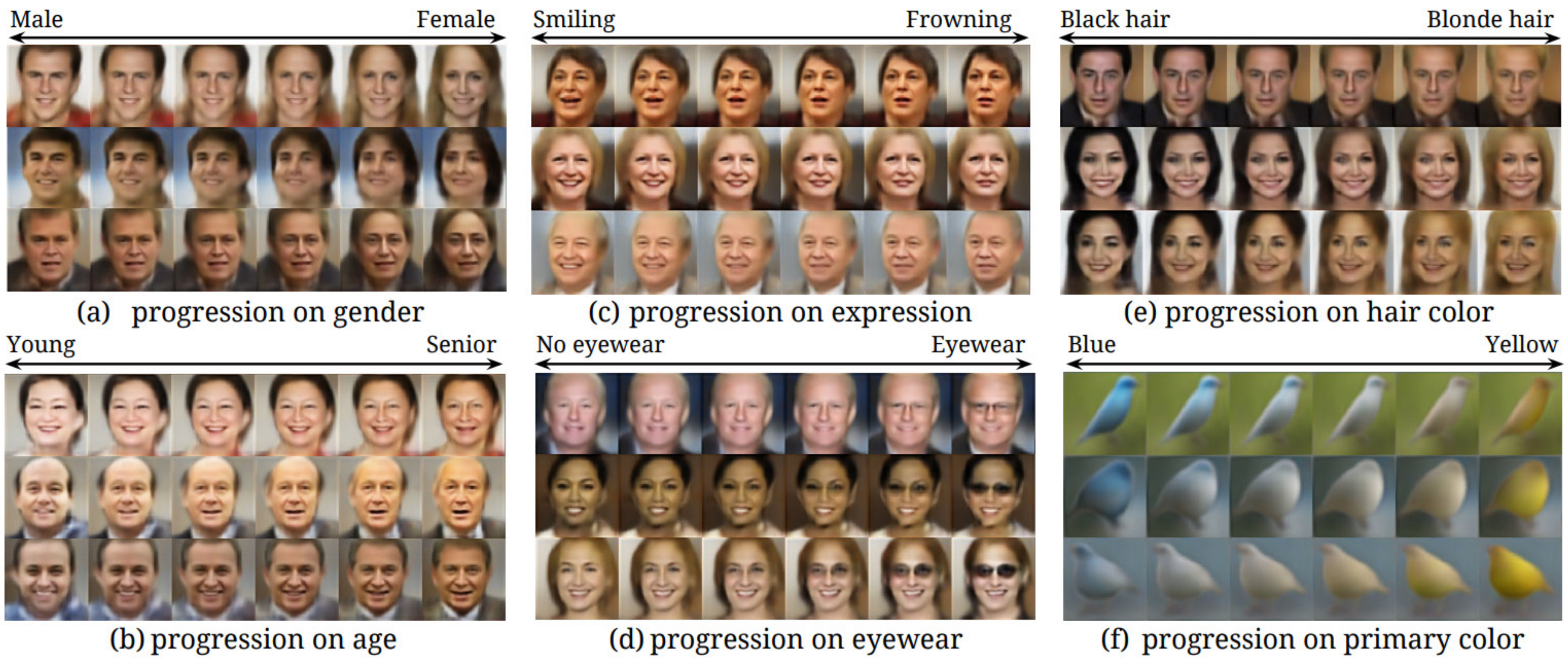


Attribute-conditioned Image Generation

https://arxiv.org/abs/1512.00570

Some Applications

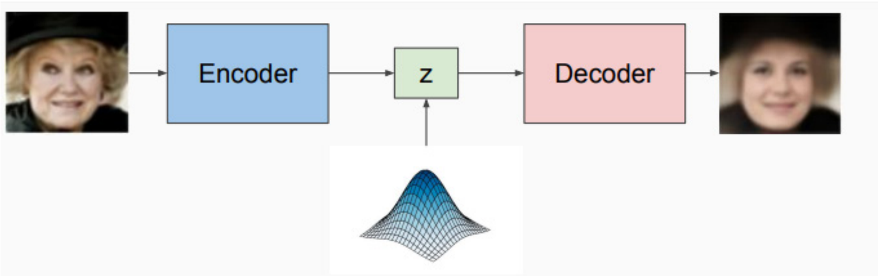
- Conditional VAEs
  - One can condition on attributes



https://arxiv.org/abs/1512.00570

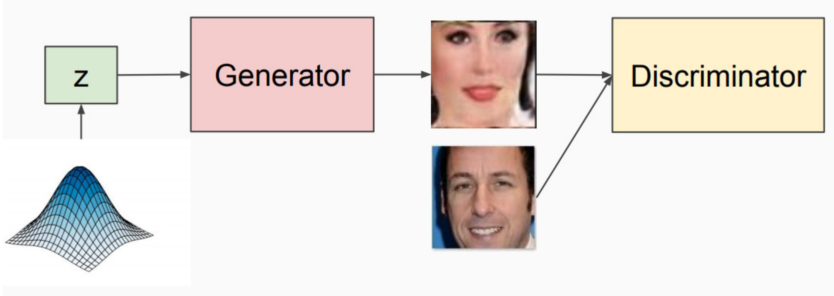
VAEs vs. GANs

VAE



- Advantages
  - Elegant theory
  - State-of-the-art results
  - Interpretable probabilities
- Disadvantages
  - Images may be blurry (loss function has MSE + KL)
  - Caveat: These are the mean p(**x**|**z**) images
  - Individual samples may have salt-and-pepper noise

GAN



- Advantages
  - Sharp images
- Disadvantages
  - No explicit probability

Can combine the two: AE architecture + adversarial losses etc

https://arxiv.org/pdf/1909.11573

More on VAEs

- Vector Quantized VAEs (VQ-VAE)
  - Part of DALL-E (creates images from textual descriptions)

  - DALL-E 3 was released a few weeks ago → may have seen in the news
  - Main idea: VAEs have a continuous latent space (**z**), whereas VQ-VAE learns a discrete latent representation
  - Sometimes natural to work with discrete representations (e.g. speech)
  - Also we have algorithms that work on discrete data (e.g. transformers)

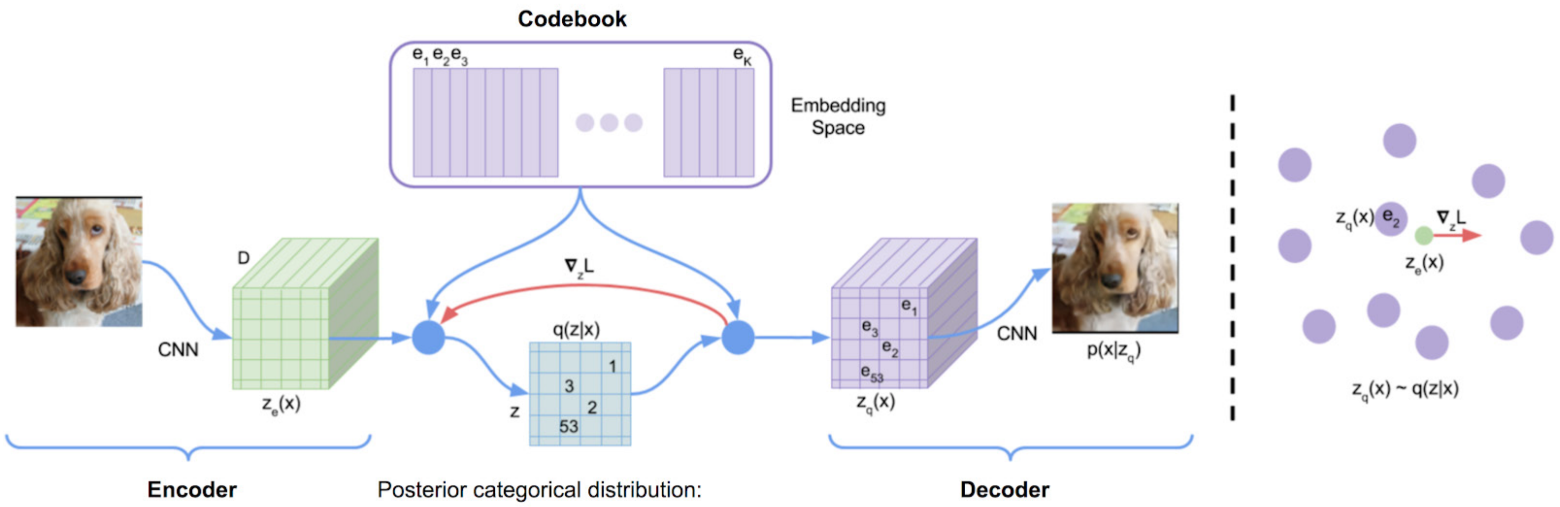
https://ml.berkeley.edu/blog/posts/vq-vae/

VQ-VAEs

- Quantizing autoencoders
  - A discrete codebook is added to quantize the latent space

$$z_q(x) = \arg\min_{k \in \{1,\dots,K\}} ||z_e(x) - e_k||_2$$

$z_e(x)$: encoding for some input x, e_k: k^{th} codebook vector, $z_q(x)$: resulting quantized vector (goes into decoder)



$$q(\mathbf{z} = \mathbf{e}_k|\mathbf{x}) = \begin{cases} 1 & \text{if } k = \arg\min_i ||\mathbf{z}_e(\mathbf{x}) - \mathbf{e}_i||_2 \\ 0 & \text{otherwise.} \end{cases}$$

https://arxiv.org/pdf/1711.00937.pdf

VQ-VAEs

- Quantizing autoencoders
  - A discrete codebook is added to quantize the latent space

$$z_q(x) = \arg \min_{k \in \{1,\ldots,K\}} ||z_e(x) - e_k||_2$$

  - Note arg min is not differentiable with respect to encoder
    - In practice: Set gradient to 1 with respect to encoder and quantized codebook vector, 0 with respect to all other codebook vectors
    - This works fine
  - How to build a codebook to avoid "memorization"?
    - K = 512 or so in practice
    - But the encoder output is 32 × 32
    - So decoder can output $512^{32\times 32} = 2^{9216}$ possible images
    - Huge discrete space

https://arxiv.org/pdf/1711.00937.pdf

VQ-VAEs

- Learning the codebook
  - Learned via gradient descent
  - Need to learn both the codebook (aligning with encoder outputs) and the encoder (whose outputs align with the codebook)
  - Solved with the loss function

stop gradient operator: identity at forward computation and has zero partial derivatives (as described in previous slide)
i.e. its operand remains constant (not updated)

$$\log\Big(p(x|q(x)\Big) + \Big|\Big|sg[z_e(x)] - e\Big|\Big|_2^2 + \beta\Big|\Big|z_e(x) - sg[e]\Big|\Big|_2^2$$

standard loss from before

codebook alignment loss
sg applied to encoder output → this term only updates the codebook

codebook commitment loss
sg applied to codebook → get the encoder output to commit to the closest codeword

https://arxiv.org/pdf/1711.00937.pdf

VQ-VAEs

- Can learn images close to original images (dimensionality reduction)

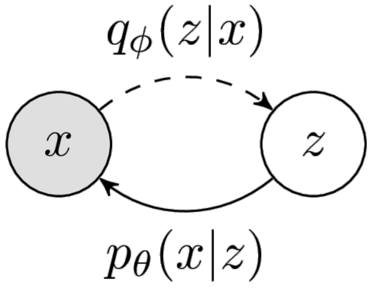
Figure 2: Left: ImageNet 128x128x3 images, right: reconstructions from a VQ-VAE with a 32x32x1 latent space, with K=512.

- In standard VAEs, we assume a prior on latent space p(z), encoder learns p(z|x), decoder learns p(x|z)
- Here, VQ-VAE assumes uniform prior over latent space during training
- For image generation → Abandon the uniform prior, and learn a new prior on the latent space

https://arxiv.org/pdf/1711.00937.pdf

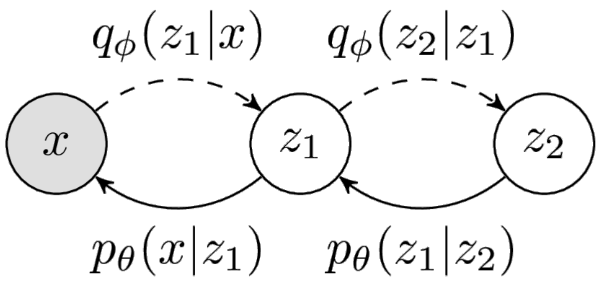
Hierarchical VAEs

- Recall our VAE setup
  - We assume a distribution on the latent space, *p*(z), and we have *p*(x|z)
  - We need *p*(z|x) for the encoder, but instead approximate this by learning *q*(z|x)
  - Graphically



Note both the decoder *p*(x|z) and the encoder *q*(z|x) are parametrized by learnable parameters θ and ϕ

- One extension of this is hierarchical VAEs
  - We consider a VAE with two latent spaces, graphically



  - The loss function for this is given in a similar manner

$$\mathbb{E}_{q(\boldsymbol{z}_1|\boldsymbol{z}_2)}[\log p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z}_1)] - KL\Big(q_{\boldsymbol{\phi}}(\boldsymbol{z}_1|\boldsymbol{x})||p_{\boldsymbol{\theta}}(\boldsymbol{z}_1|\boldsymbol{x})\Big) - KL\Big(q_{\boldsymbol{\phi}}(\boldsymbol{z}_2|\boldsymbol{z}_1)||p_{\boldsymbol{\theta}}(\boldsymbol{z}_2)\Big)$$

  - First term: "reconstruction", the other two terms are KL divergences between inference layers and corresponding priors
  - Sets us up for diffusion models (next lecture)

https://angusturner.github.io/generative_models/2021/06/29/diffusion-probabilistic-models-I.html

Recap

- VAEs
  - Dimensionality reduction
  - Autoencoders
  - VAEs
  - Implementation Details
- Variations
  - Conditional VAEs
  - VQ-VAEs
  - Hierarchical VAEs
    - Revisit this in the next lecture
- Next lecture: Diffusion/score-based models