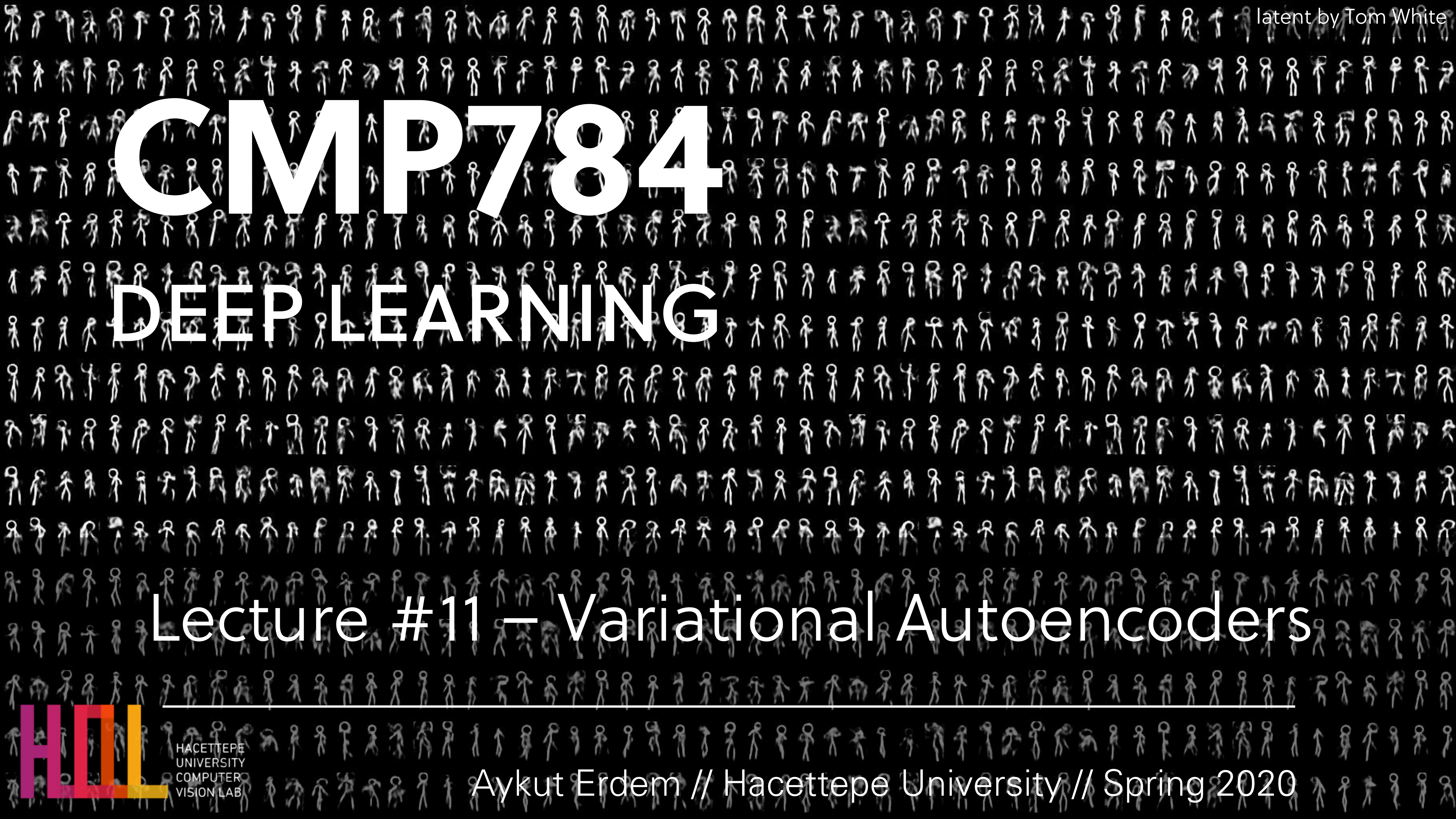


latent by Tom White

CMP784

DEEP LEARNING

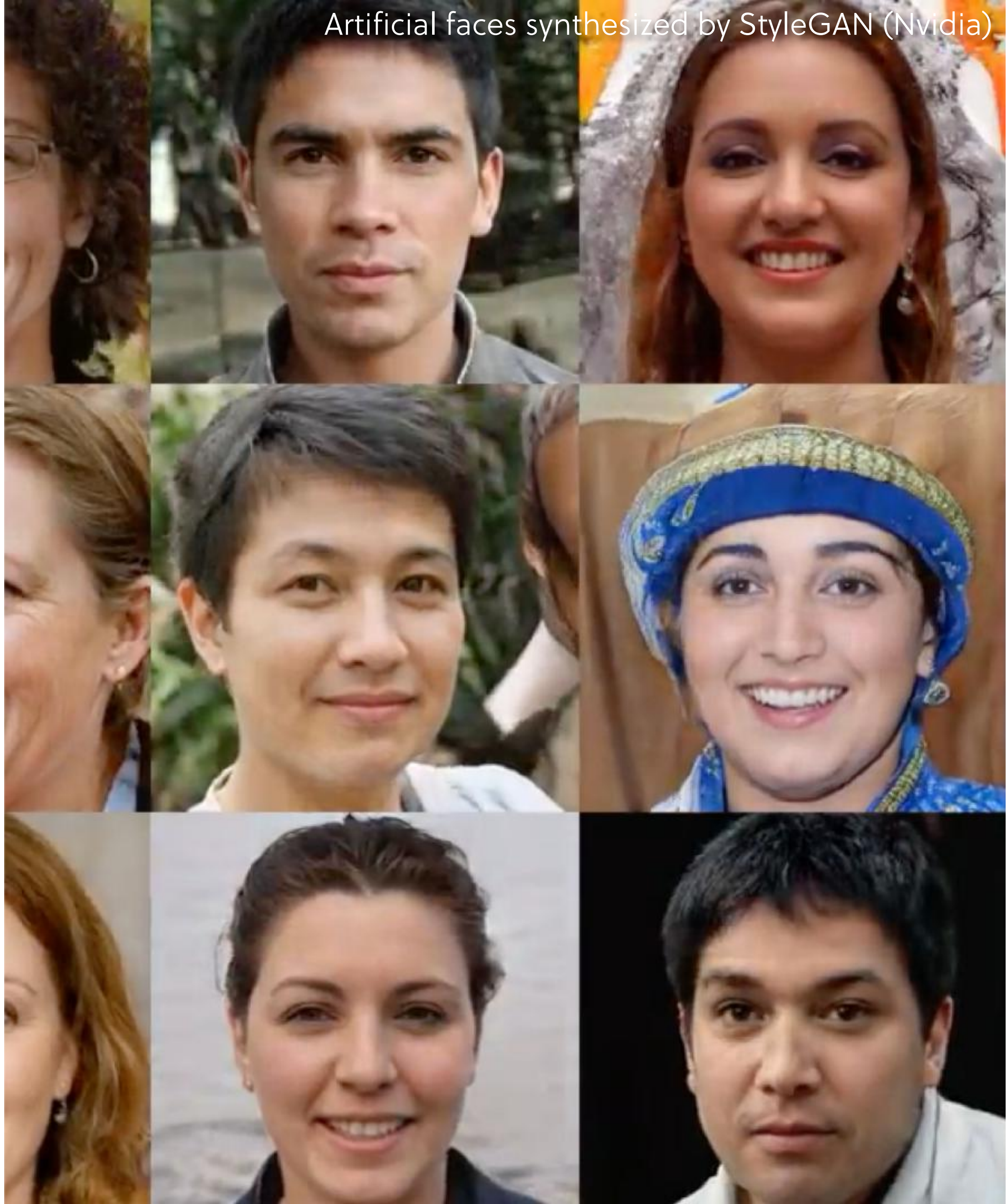
Lecture #11 – Variational Autoencoders

HACETTEPE
UNIVERSITY
COMPUTER
VISION LAB

Aykut Erdem // Hacettepe University // Spring 2020

Previously on CMP784

- Supervised vs. Unsupervised Representation Learning
- Sparse Coding
- Autoencoders
- Autoregressive Generative Models



Artificial faces synthesized by StyleGAN (Nvidia)

Lecture overview

- Motivation for Variational Autoencoders (VAEs)
- Mechanics of VAEs
- Separatibility of VAEs
- Training of VAEs
- Evaluating representations
- Vector Quantized Variational Autoencoders (VQ-VAEs)

Disclaimer: Much of the material and slides for this lecture were borrowed from

—Pavlov Protopapas, Mark Glickman and Chris Tanner's Harvard CS109B class

—Andrej Risteski's CMU 10707 class

—David McAllester's TTIC 31230 class

Lecture overview

- **Motivation for Variational Autoencoders (VAEs)**
- Mechanics of VAEs
- Separatibility of VAEs
- Training of VAEs
- Evaluating representations
- Vector Quantized Variational Autoencoders (VQ-VAEs)

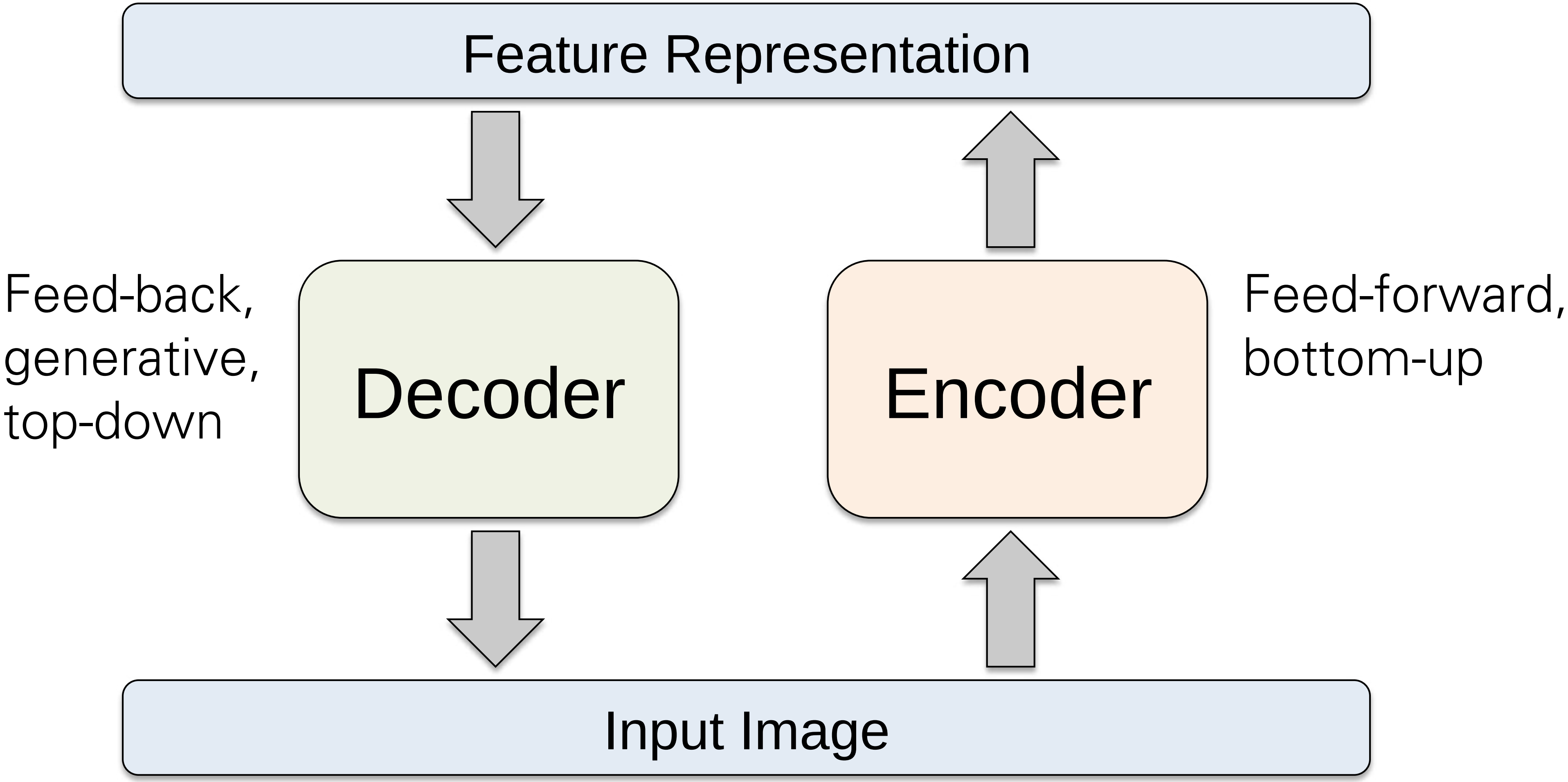
Disclaimer: Much of the material and slides for this lecture were borrowed from

—Pavlov Protopapas, Mark Glickman and Chris Tanner's Harvard CS109B class

—Andrej Risteski's CMU 10707 class

—David McAllester's TTIC 31230 class

Recap: Autoencoders



- Details of what goes insider the encoder and decoder matter!
- Need constraints to avoid learning an identity.

Parameter space of autoencoder

- Let’s examine the latent space of an AE.
- Is there any separation of the different classes? If the AE learned the “essence” of the MNIST images, similar images should be close to each other.
- Plot the latent space and examine the separation.
- Here we plot the 2 PCA components of the latent space.

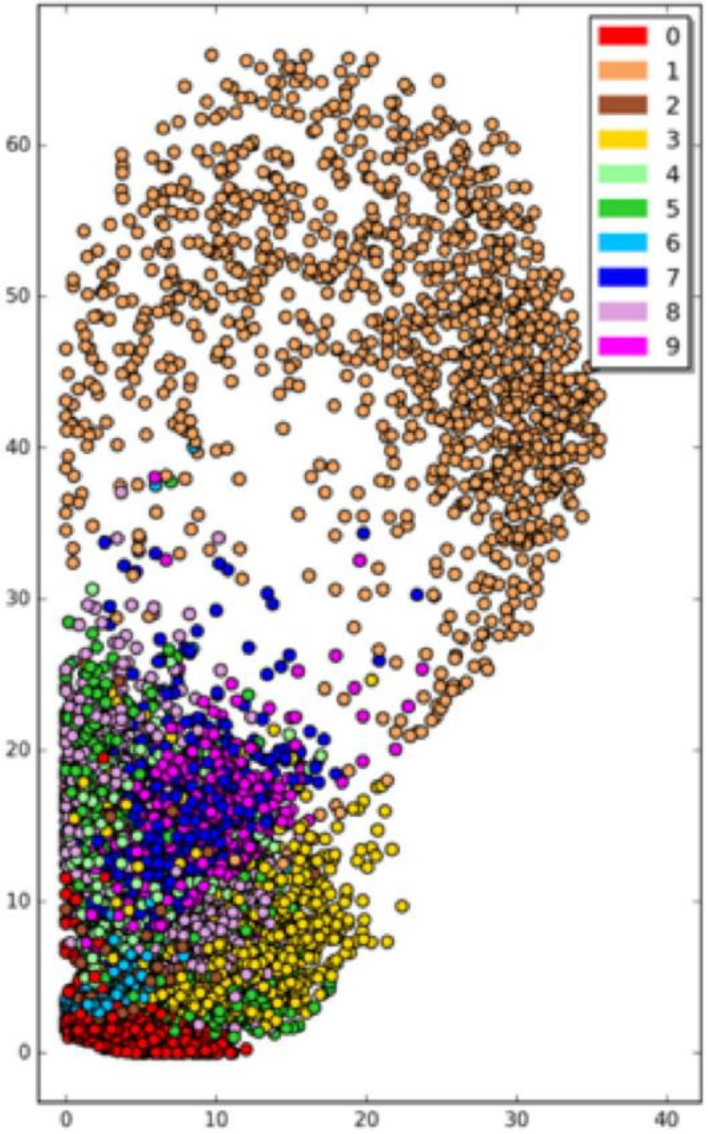
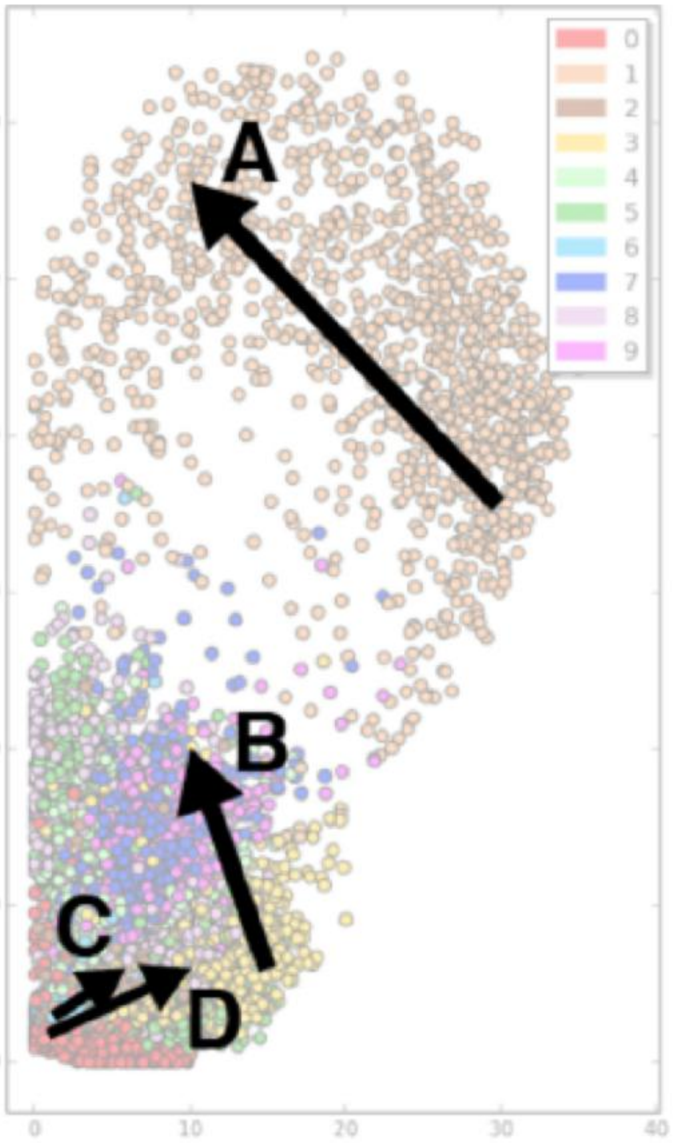


Image taken from A. Glassner, Deep Learning, Vol. 2: From Basics to Practice

Traversing the latent space



- We start at the start of the arrows in latent space and then move to end of the arrow in 7 steps.
- For each value of z we use the already trained decoder to produce an image.

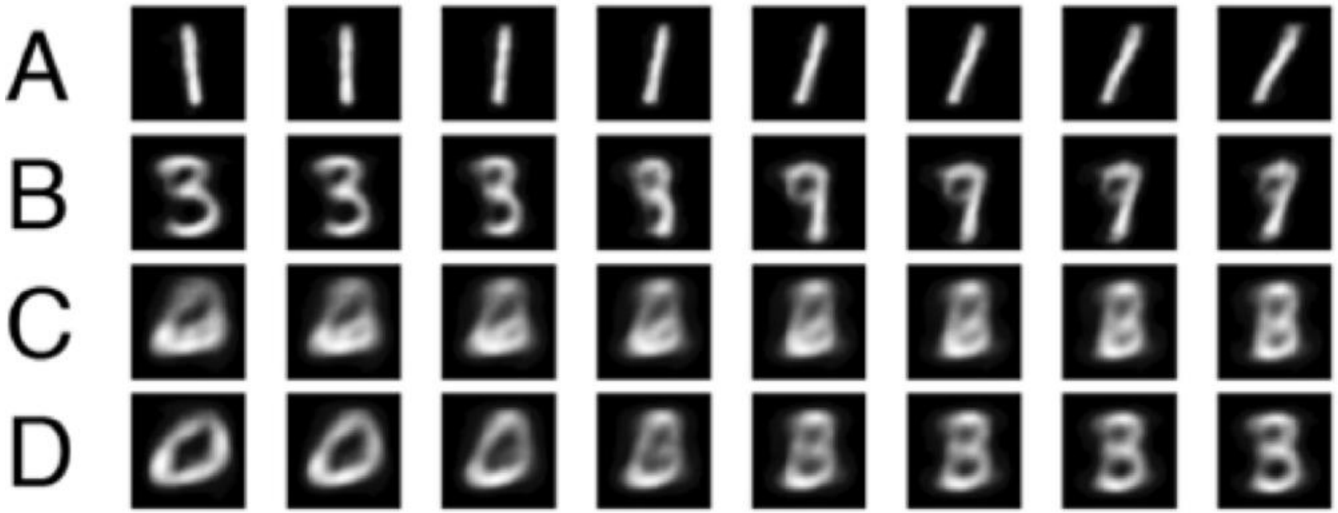


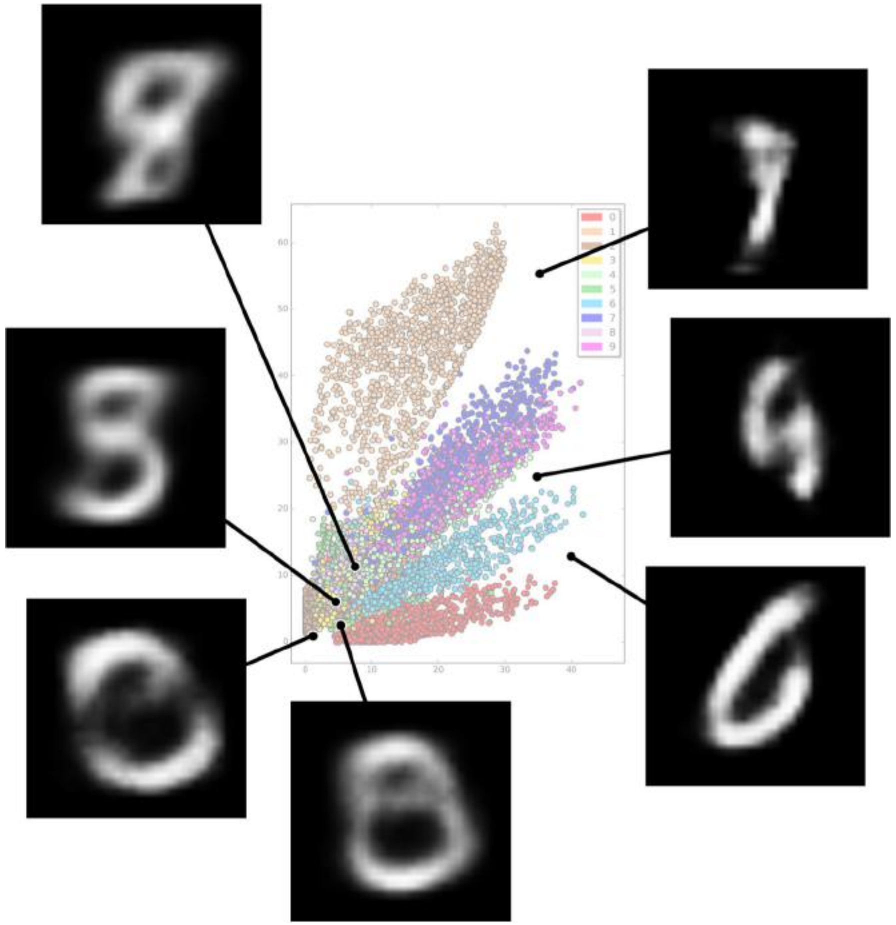
Image taken from A. Glassner, Deep Learning, Vol. 2: From Basics to Practice

Problems with Autoencoders

- Gaps in the latent space

- Discrete latent space

- Separability in the latent space



Lecture overview

- Motivation for Variational Autoencoders (VAEs)
- **Mechanics of VAEs**
- Separatibility of VAEs
- Training of VAEs
- Evaluating representations
- Vector Quantized Variational Autoencoders (VQ-VAEs)

Disclaimer: Much of the material and slides for this lecture were borrowed from

—Pavlov Protopapas, Mark Glickman and Chris Tanner's Harvard CS109B class

—Andrej Risteski's CMU 10707 class

—David McAllester's TTIC 31230 class

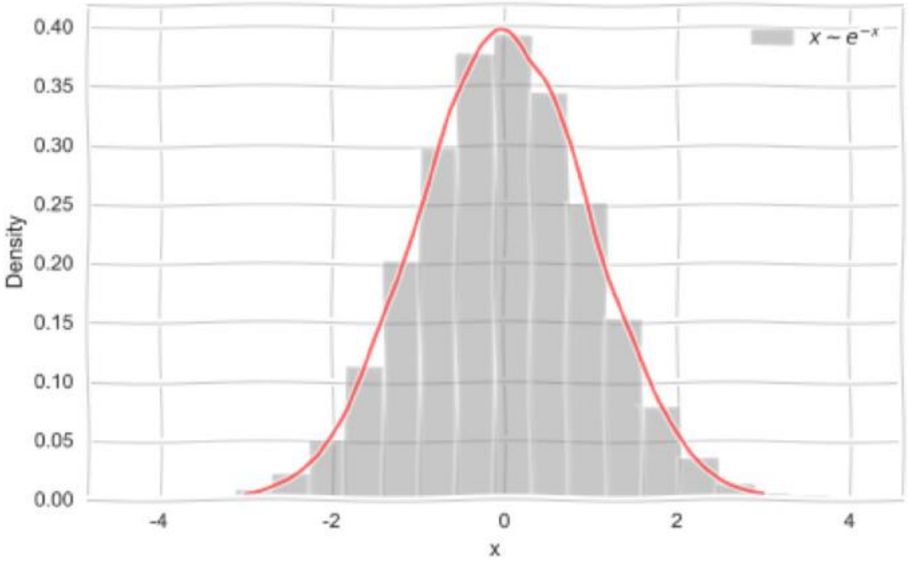
Generative models

- Imagine we want to generate data from a distribution,

$$x \sim p(x)$$

- e.g.

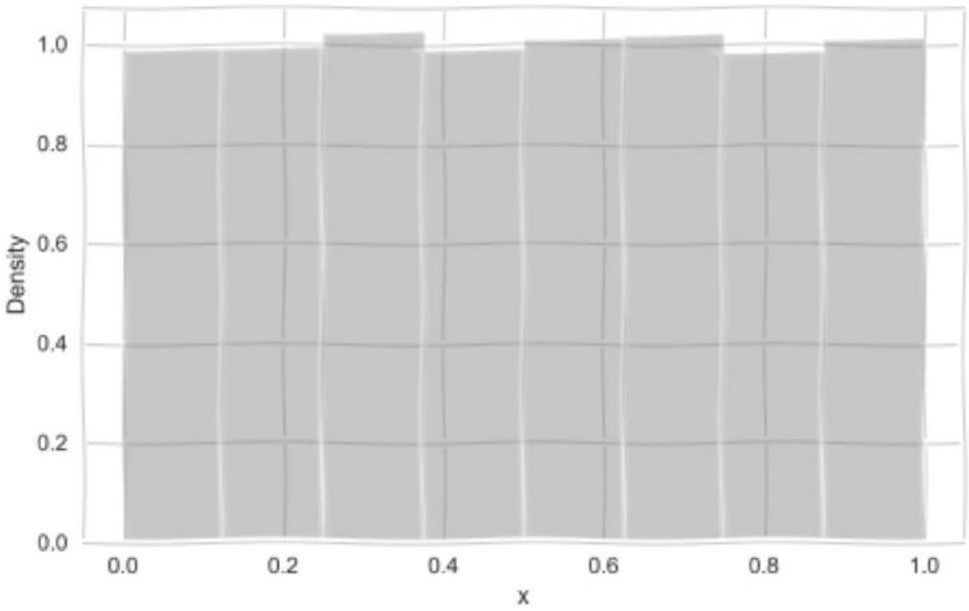
$$x \sim \mathcal{N}(\mu, \sigma)$$



Generative models

- But how do we generate such samples?

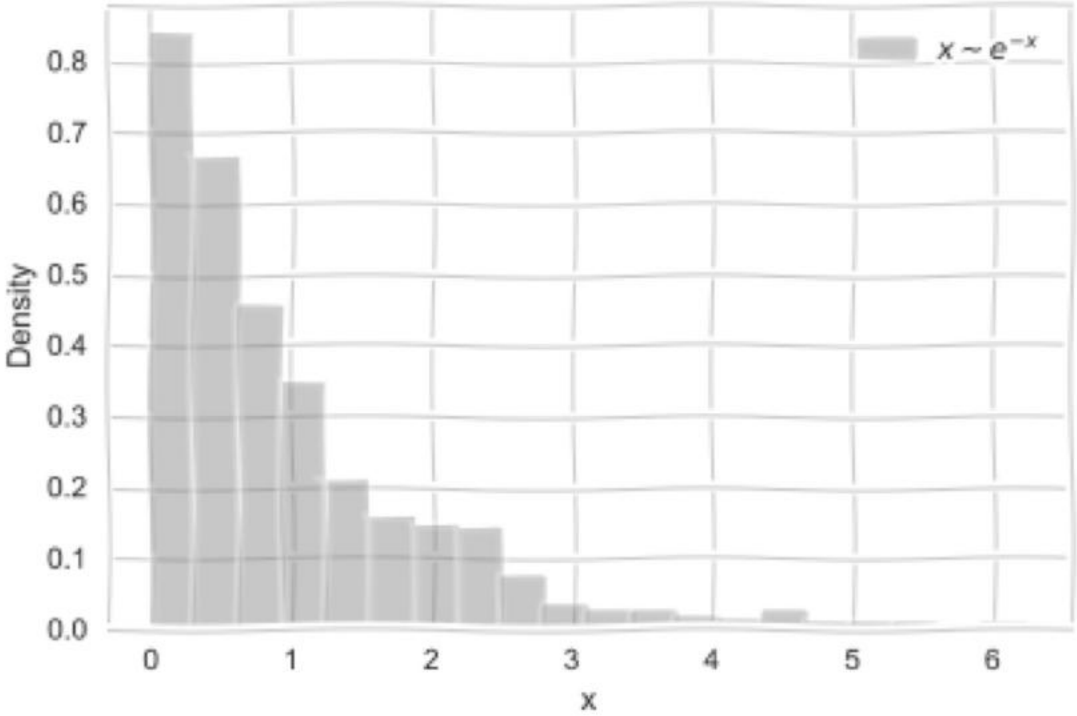
$$z \sim \mathrm{Unif}(0, 1)$$



Generative models

- But how do we generate such samples?

$$z \sim \mathrm{Unif}(0,1) \quad \mathrm{x} = \ln \mathrm{z}$$



Generative models

- In other words we can think that if we choose ***z*~Uniform** then there is a mapping:

$$x = f(z)$$

such as:

$$x \sim p(x)$$

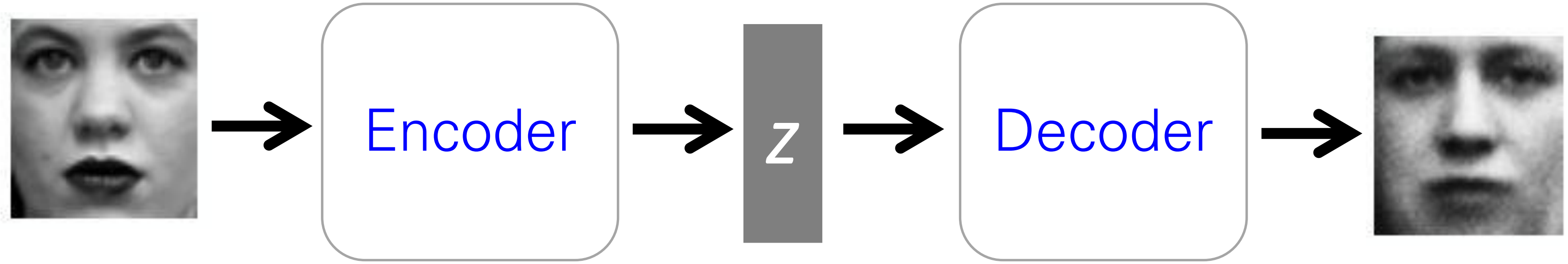
where in general f is some complicated function.

- We already know that **Neural Networks are great in learning complex functions.**

$$\begin{tikzcd} \boxed{z \sim g(z)} \arrow[r] & \boxed{x = f(z)} \arrow[r] & \boxed{x \sim p(x)} \end{tikzcd}$$

Traditional Autoencoders

- In traditional autoencoders, we can think of encoder and decoders as some function mapping.

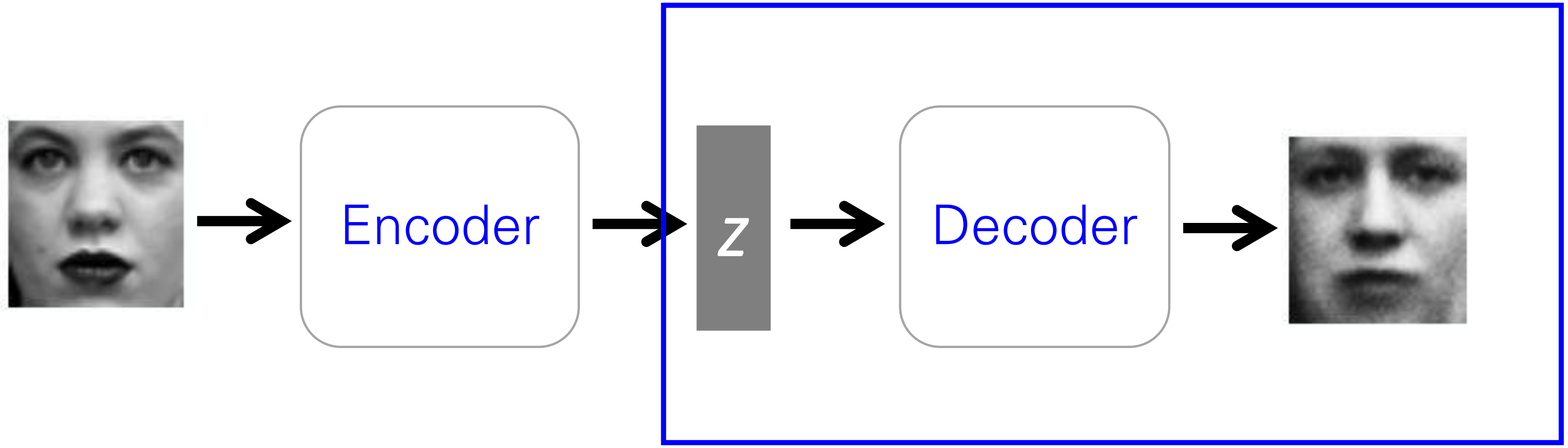


$$\boxed{z = h(x)}$$

$$\boxed{\hat{x} = f(z)}$$

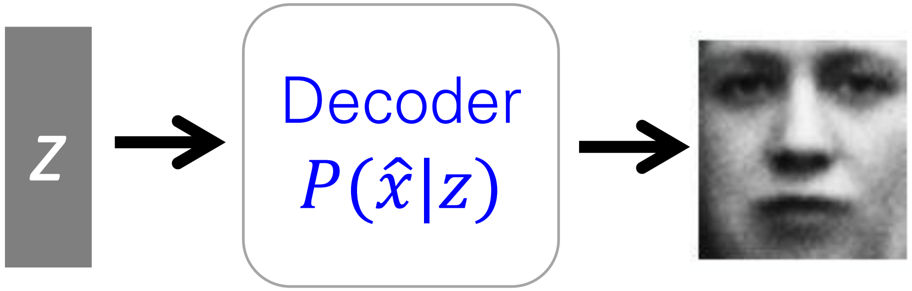
Variational Autoencoders

- To go to variational autoencoders, we need to first add some **stochasticity** and think of it as a probabilistic modeling.



Variational Autoencoders

Sample from g(z)
e.g. Standard
Gaussian

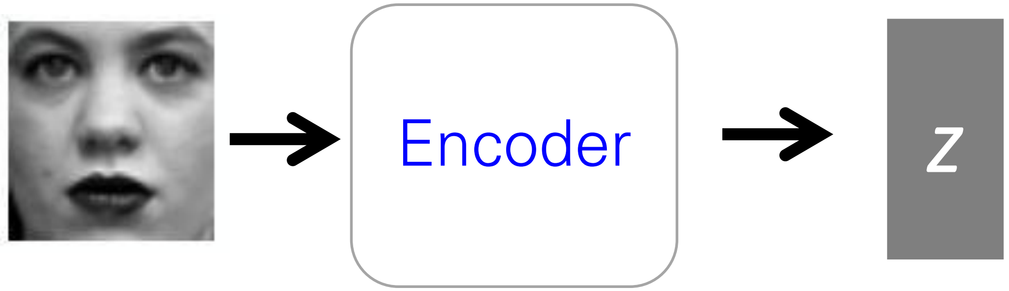


z

Decoder
$P(\hat{x}|z)$

$\boxed{z \sim g(z)}$ $\boxed{\hat{x} = f(z)}$ $\boxed{\hat{x} \sim P(x|z)}$

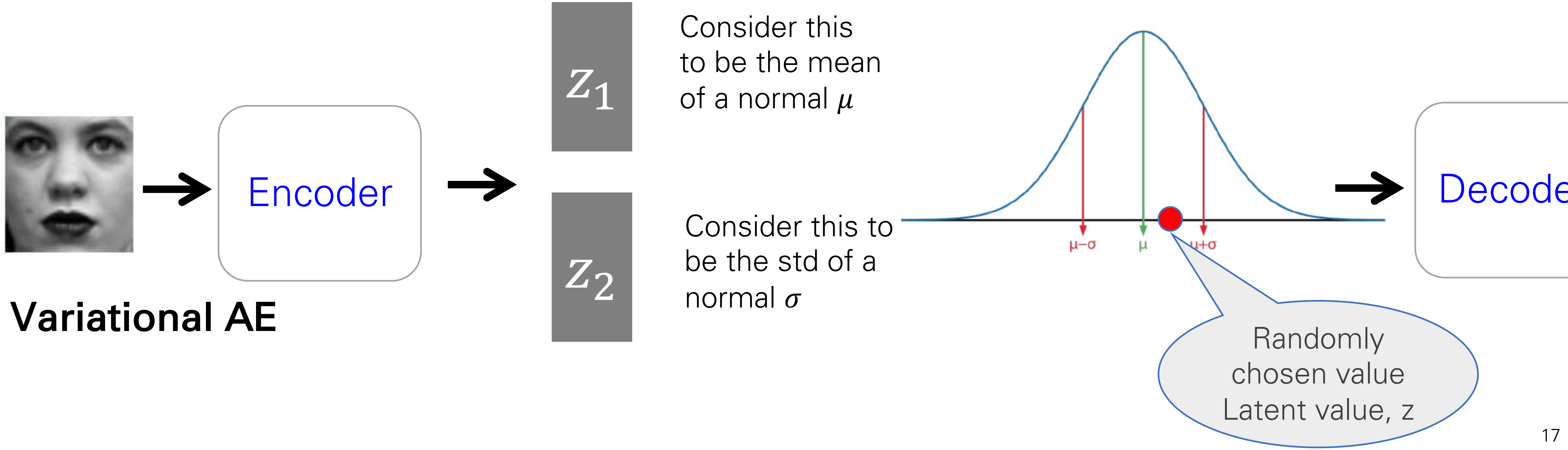
Variational Autoencoders



Encoder

z

Traditional AE



Encoder

z_1

Consider this to be the mean of a normal μ

z_2

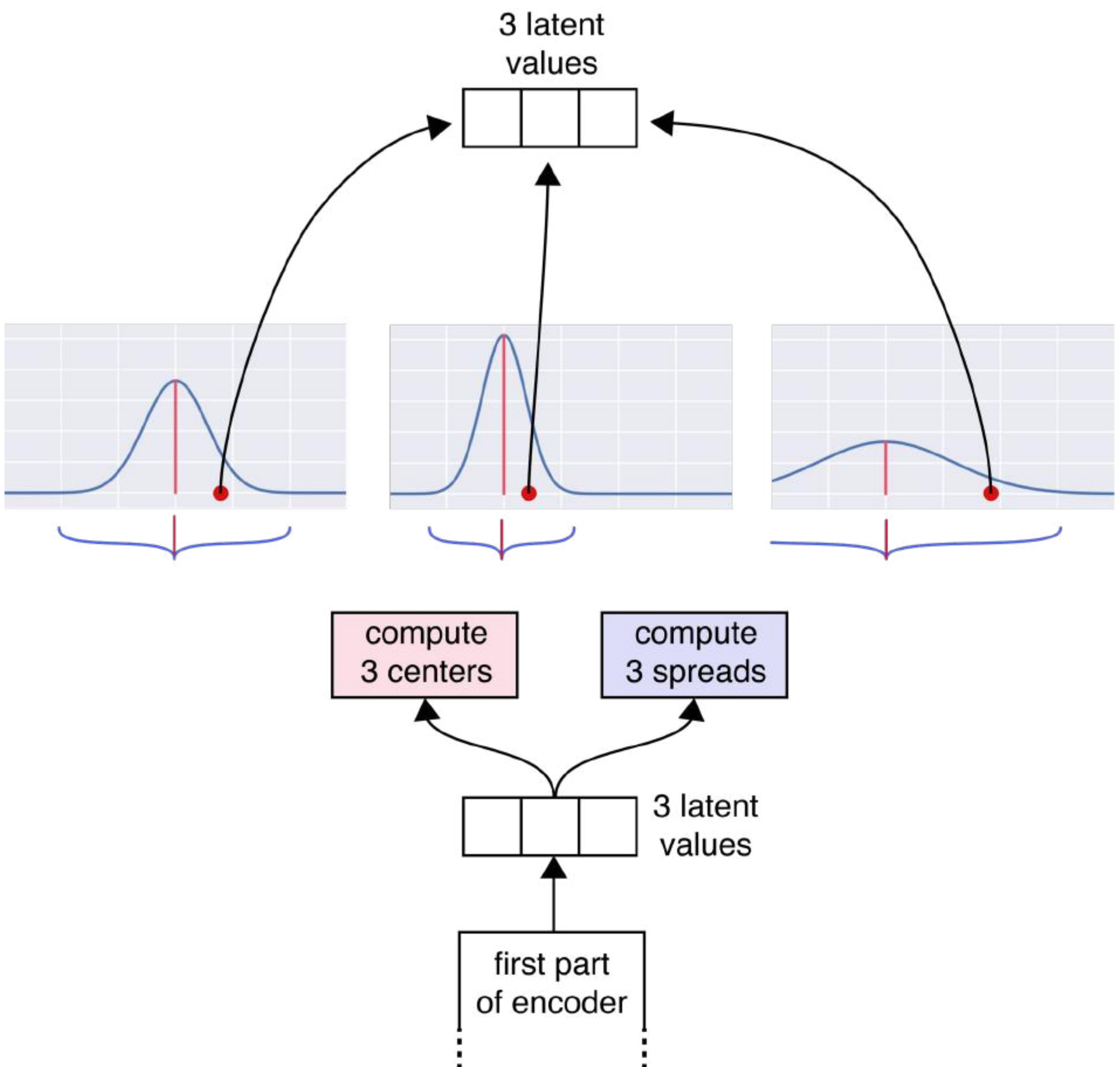
Consider this to be the std of a normal σ

Decode

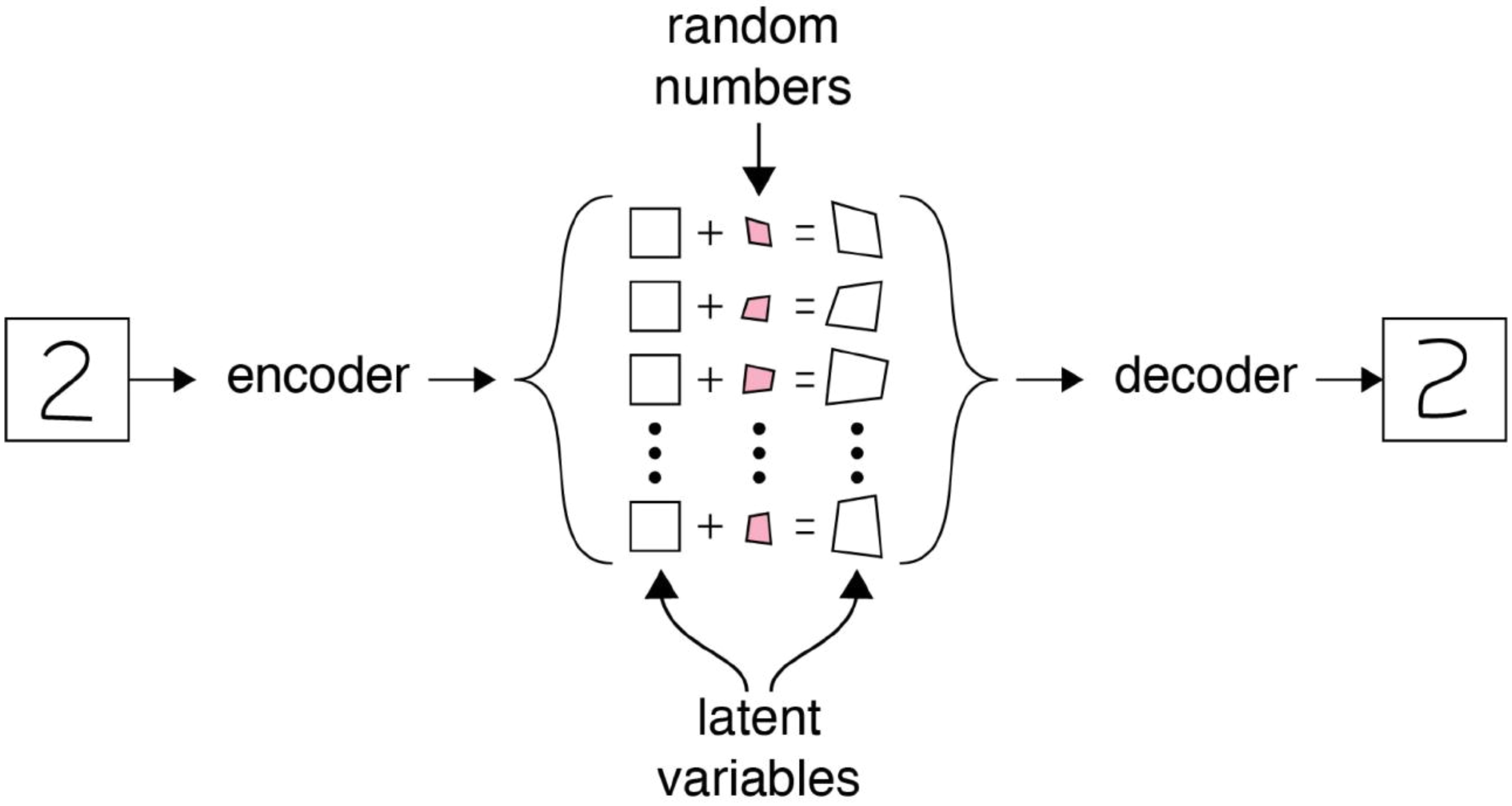
Randomly chosen value Latent value, z

Variational AE

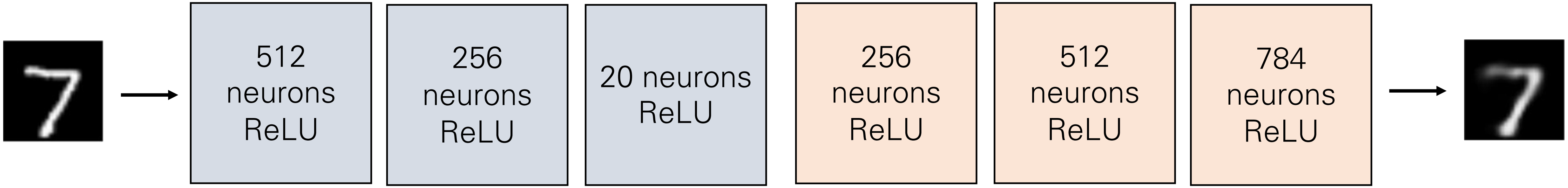
Variational Autoencoders



Variational Autoencoders



Variational Autoencoders



512 neurons ReLU

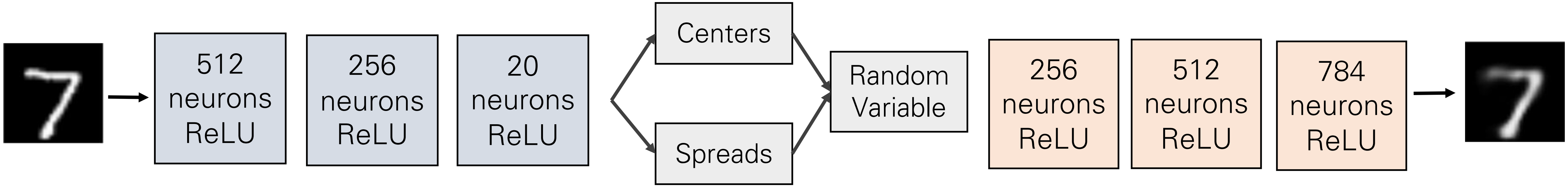
256 neurons ReLU

20 neurons ReLU

256 neurons ReLU

512 neurons ReLU

784 neurons ReLU



512 neurons ReLU

256 neurons ReLU

20 neurons ReLU

Centers

Spreads

Random Variable

256 neurons ReLU

512 neurons ReLU

784 neurons ReLU

Lecture overview

- Motivation for Variational Autoencoders (VAEs)
- Mechanics of VAEs
- **Separatibility of VAEs**
- Training of VAEs
- Evaluating representations
- Vector Quantized Variational Autoencoders (VQ-VAEs)

Disclaimer: Much of the material and slides for this lecture were borrowed from

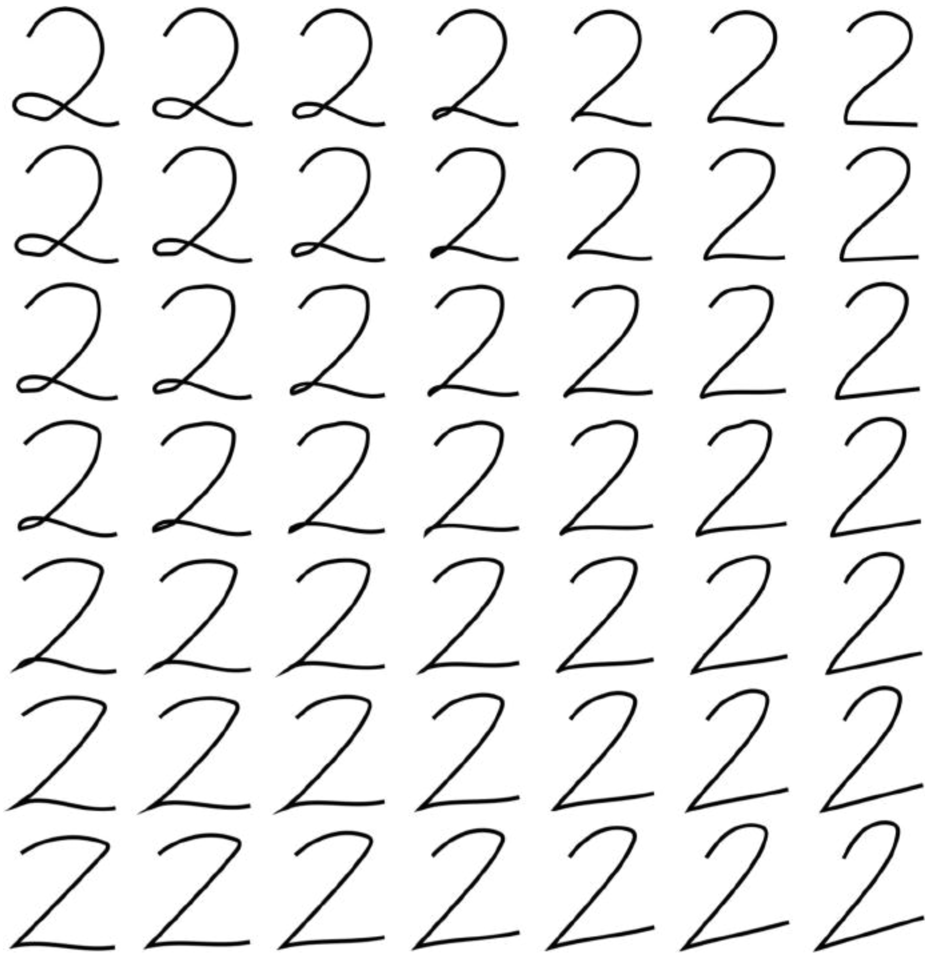
—Pavlov Protopapas, Mark Glickman and Chris Tanner's Harvard CS109B class

—Andrej Risteski's CMU 10707 class

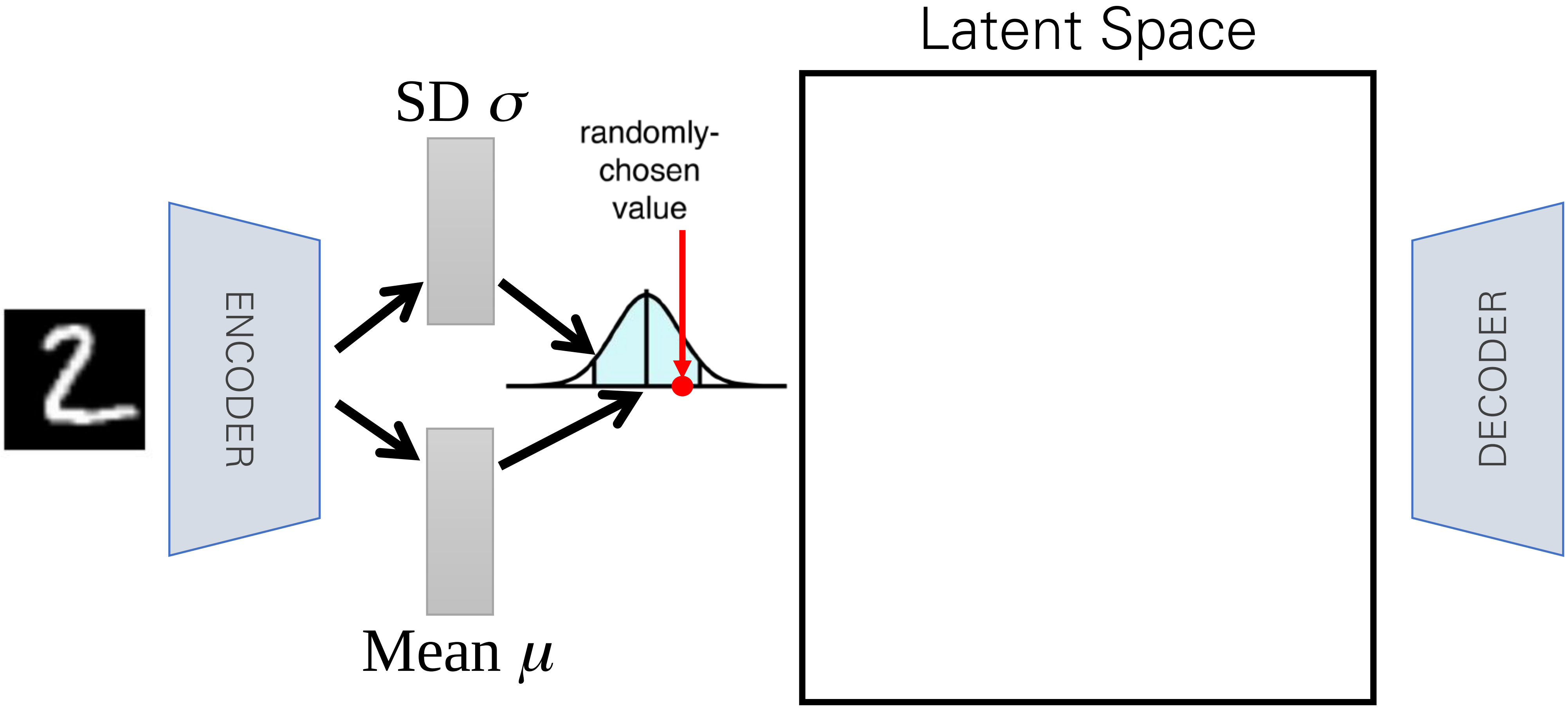
—David McAllester's TTIC 31230 class

Separability in Variational Autoencoders

- Separability is not only between classes but we also want similar items in the same class to be near each other.
- For example, there are different ways of writing “2”, we want similar styles to end up near each other.
- Let’s examine VAE, there is something magic happening once we add stochasticity in the latent space.

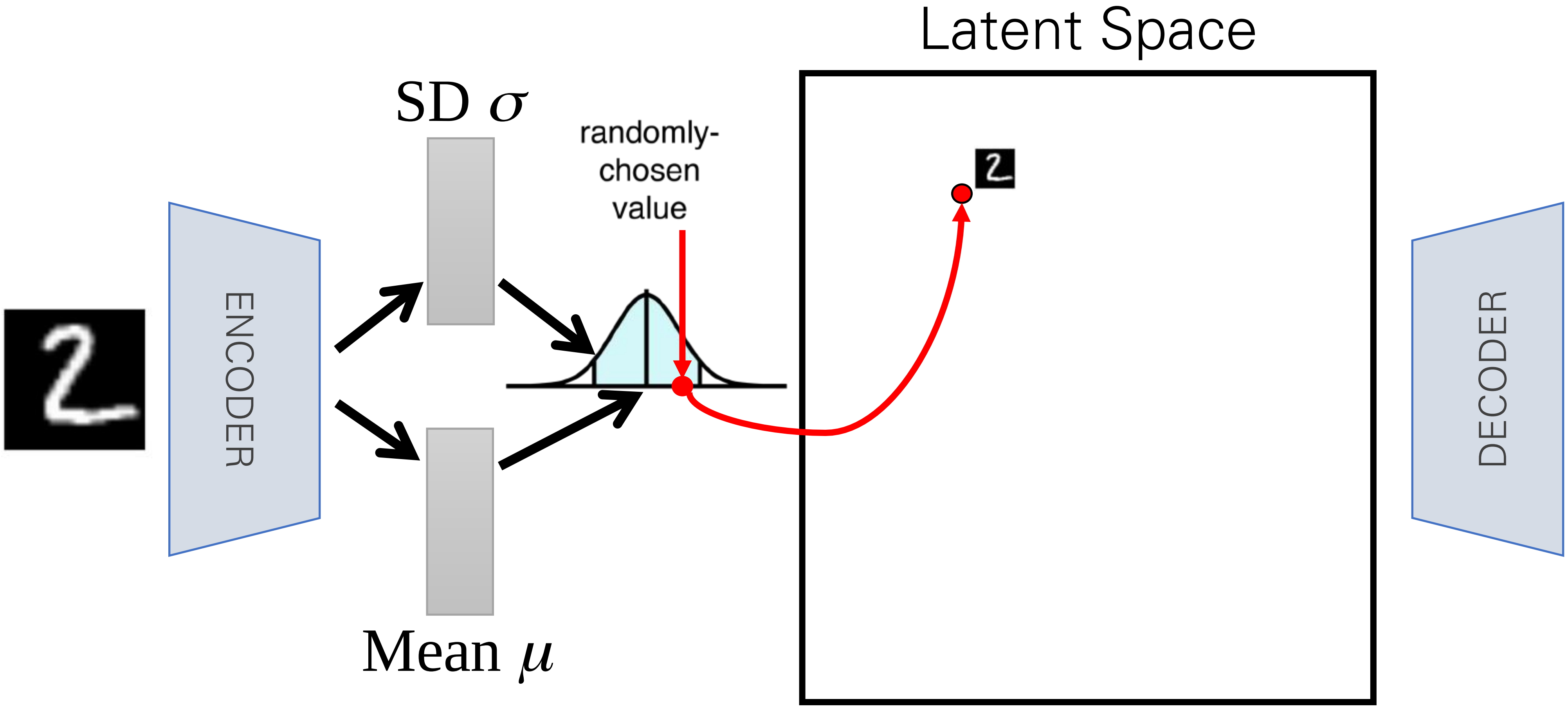


Separability in Variational Autoencoders



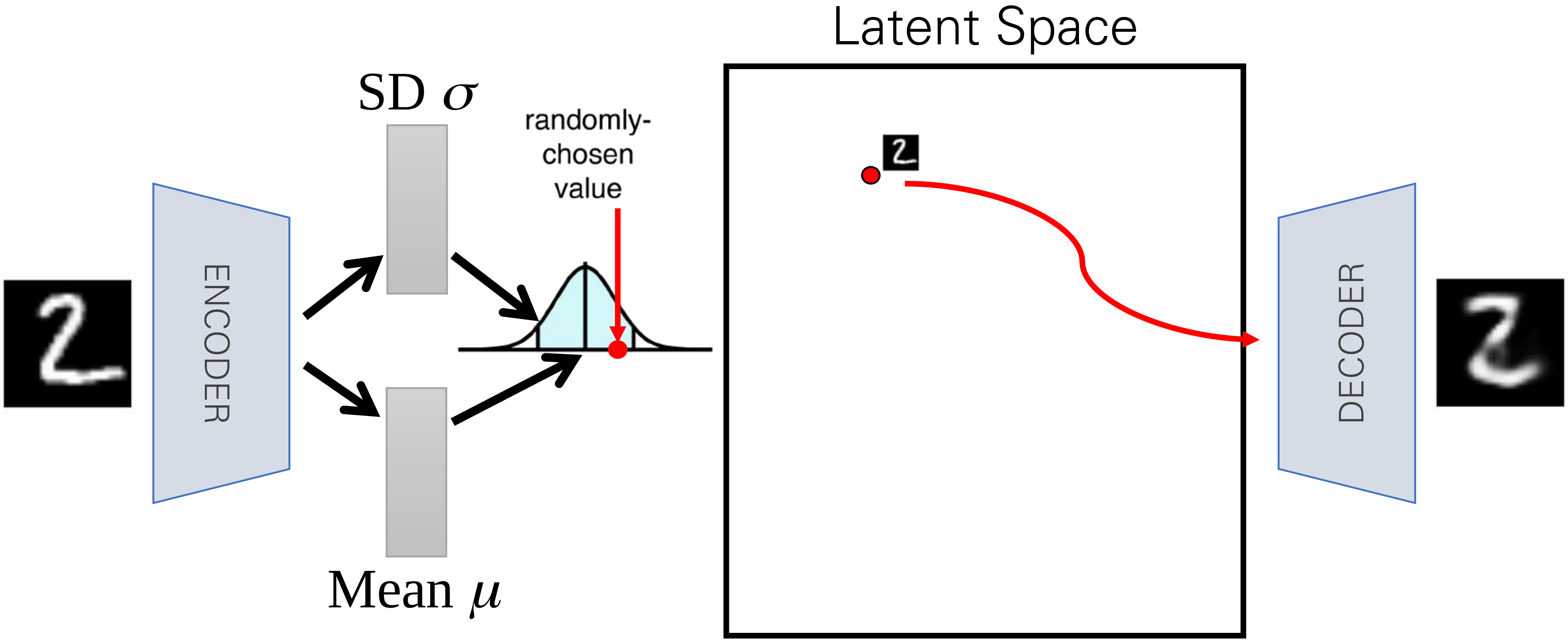
Encode the first sample (a “2”) and find μ_1, σ_1

Separability in Variational Autoencoders



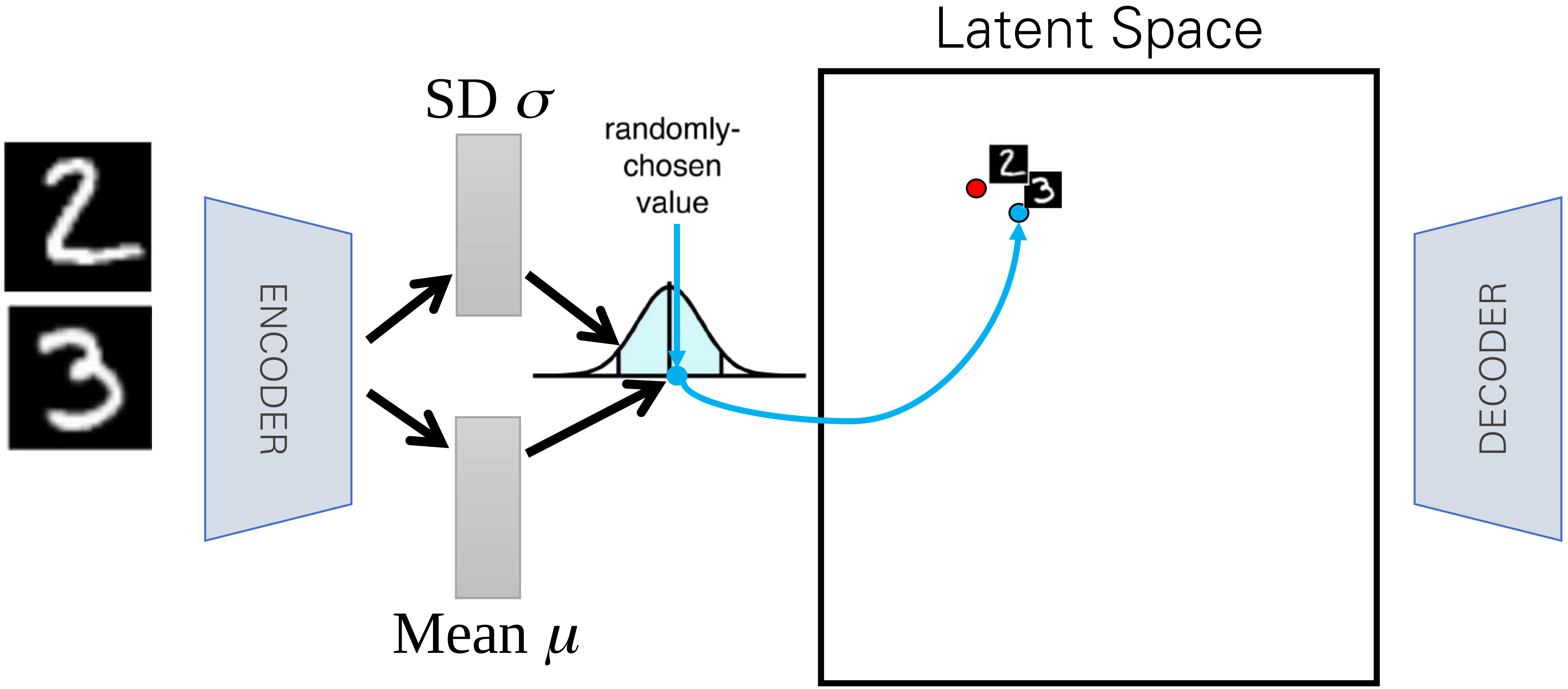
Sample $z_1 \sim N(\mu_1, \sigma_1)$

Blending Latent Variables



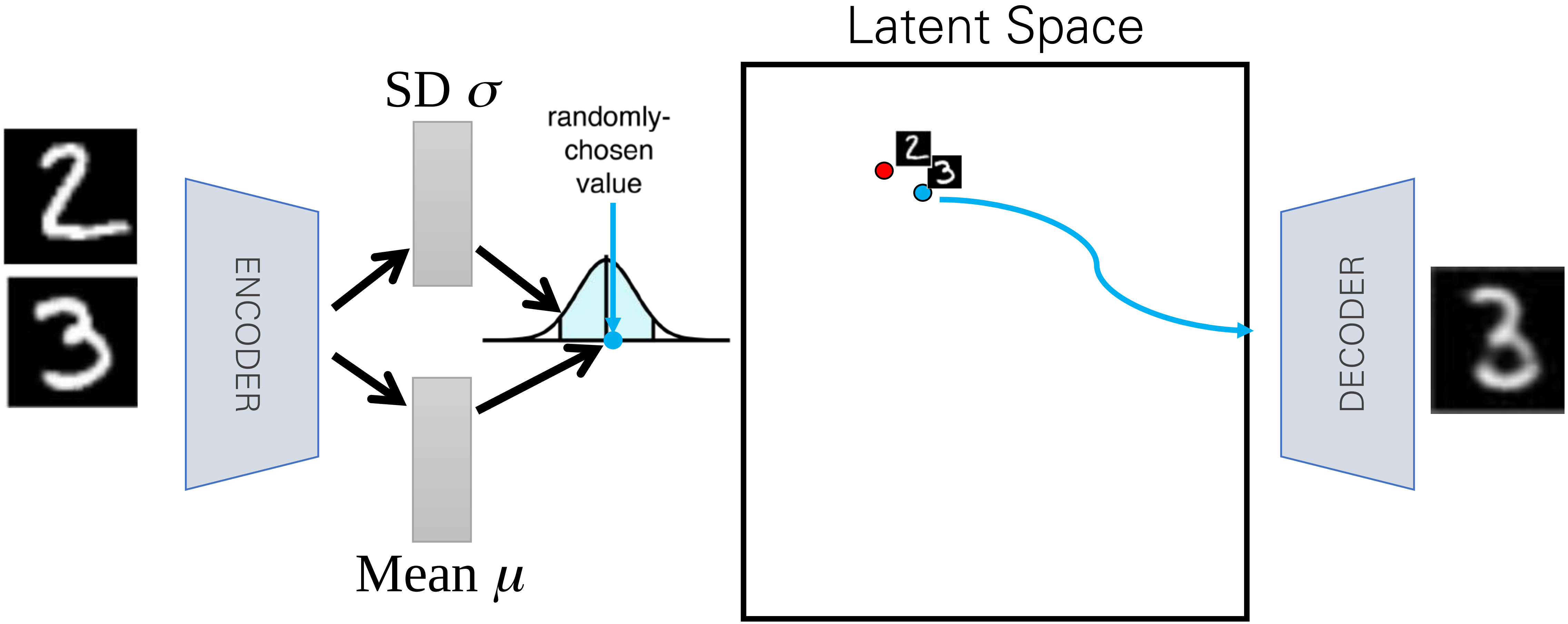
Decode to $\hat{x}_1$

Separability in Variational Autoencoders



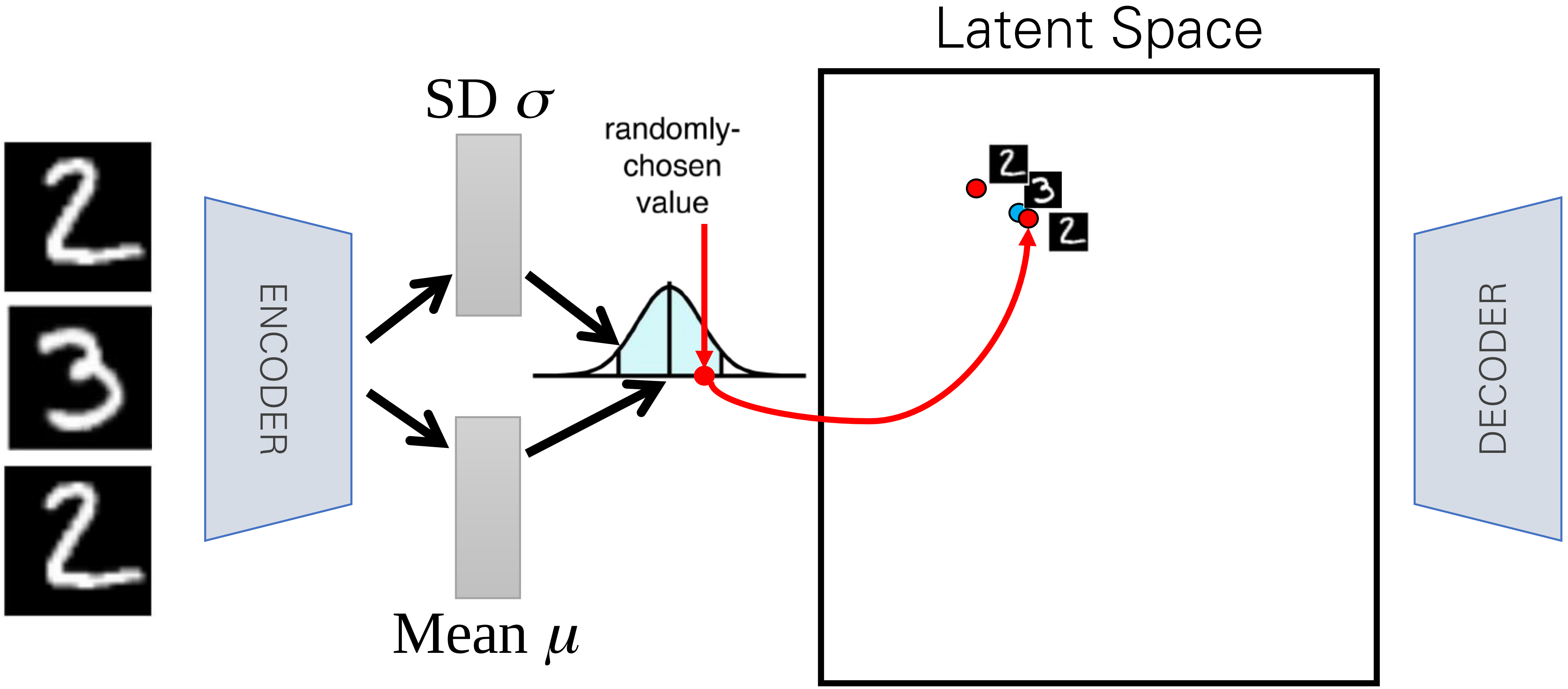
Encode the second sample (a "3") find μ_2, σ_2. Sample $z_2 \sim N(\mu_2, \sigma_2)$

Separability in Variational Autoencoders



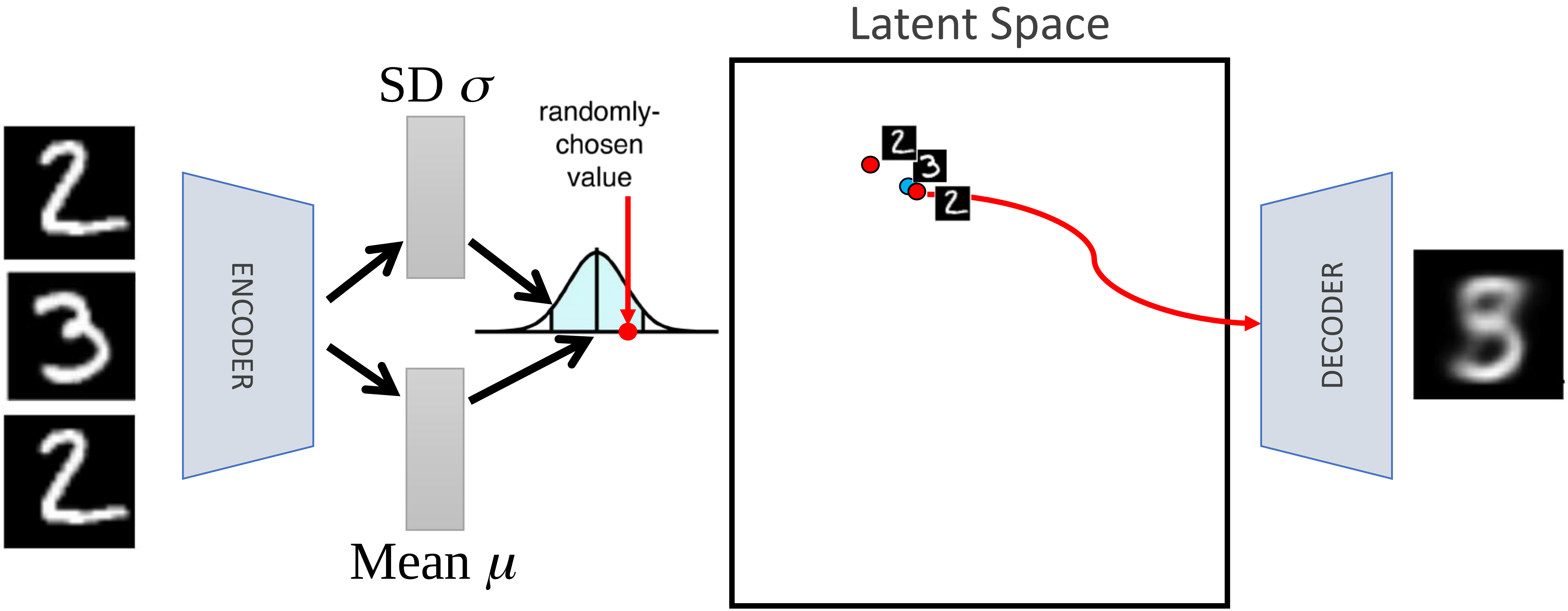
Decode to $\hat{x}_2$

Separability in Variational Autoencoders



Train with the first sample (a “2”) again and find μ_1, σ_1. However $z_1 \sim N(\mu_1, \sigma_1)$ **will not be the same.** It can happen to be close to the “3” in latent space.

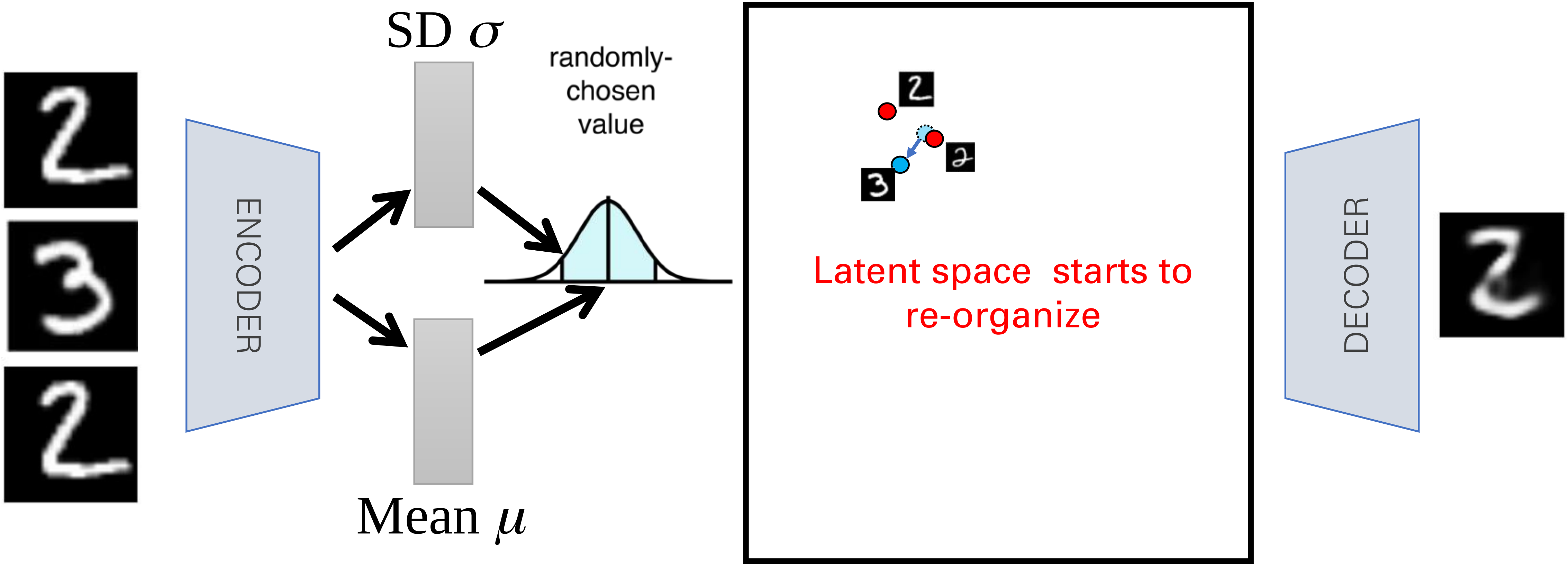
Separability in Variational Autoencoders



Decode to $\hat{x}_1$. Since the decoder only knows how to map from latent space to $\hat{x}$ space, it will return a "3".

Separability in Variational Autoencoders

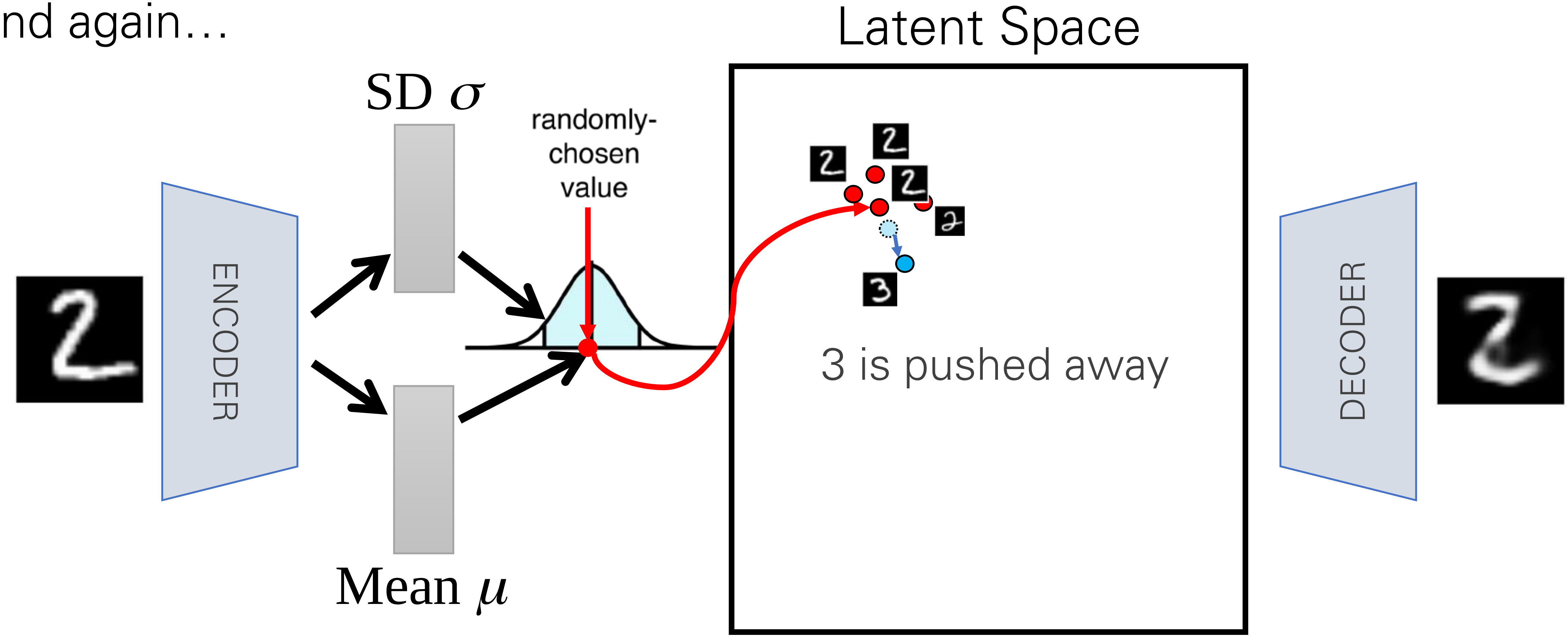
Train with 1st sample again



Latent space starts to re-organize

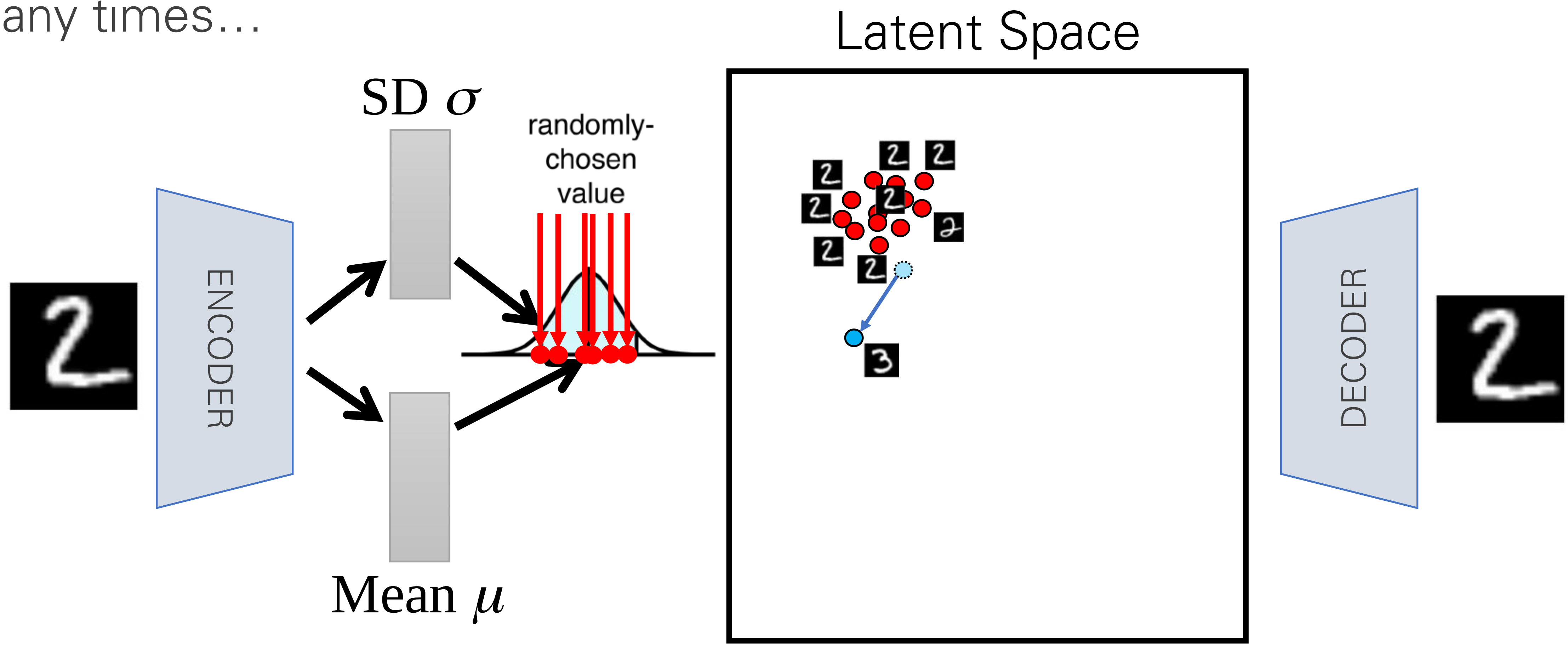
Separability in Variational Autoencoders

And again…



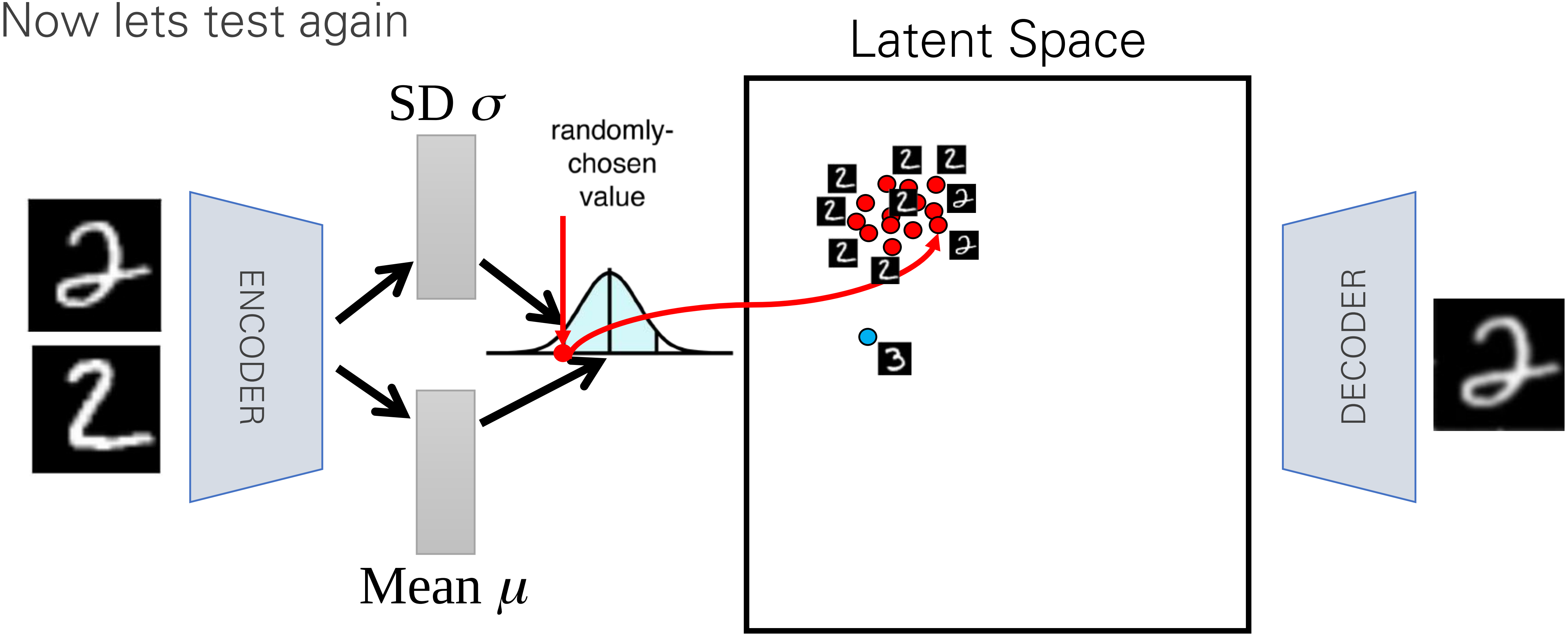
Separability in Variational Autoencoders

Many times…



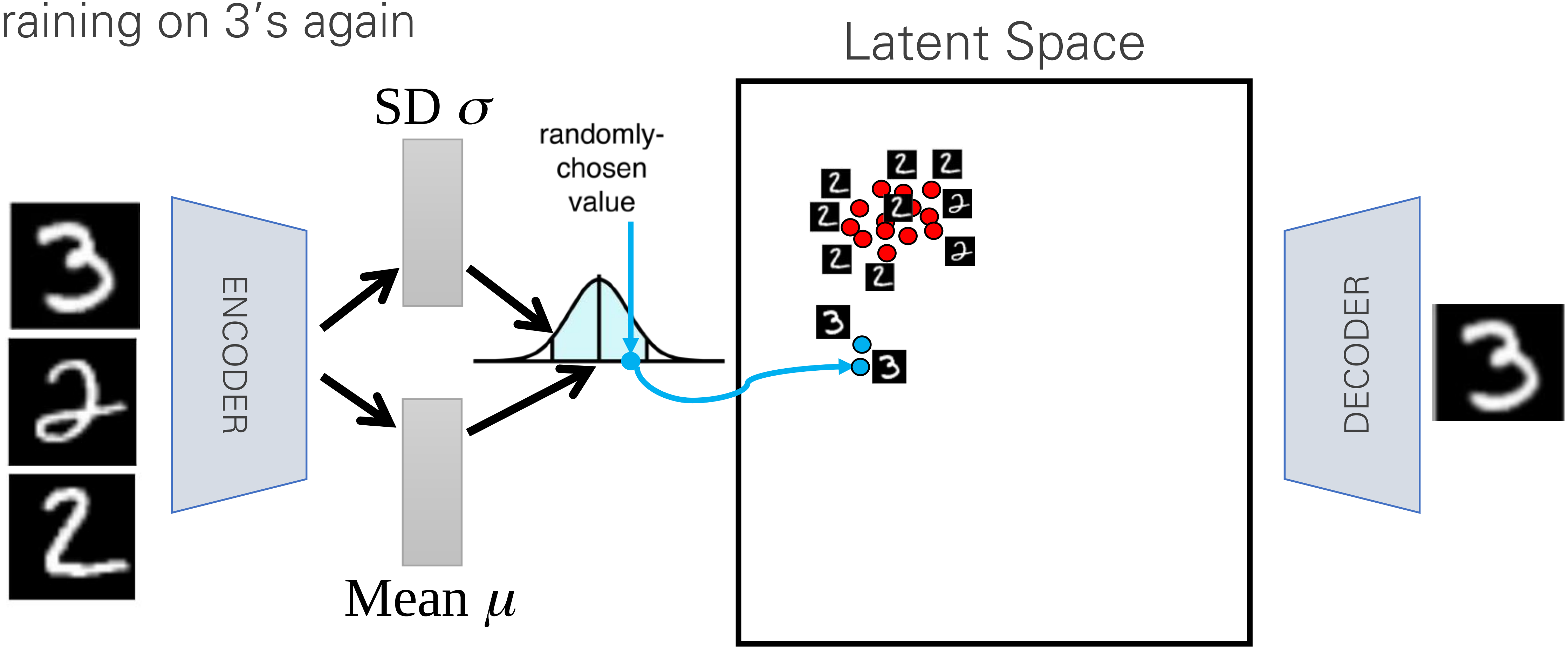
Separability in Variational Autoencoders

Now lets test again



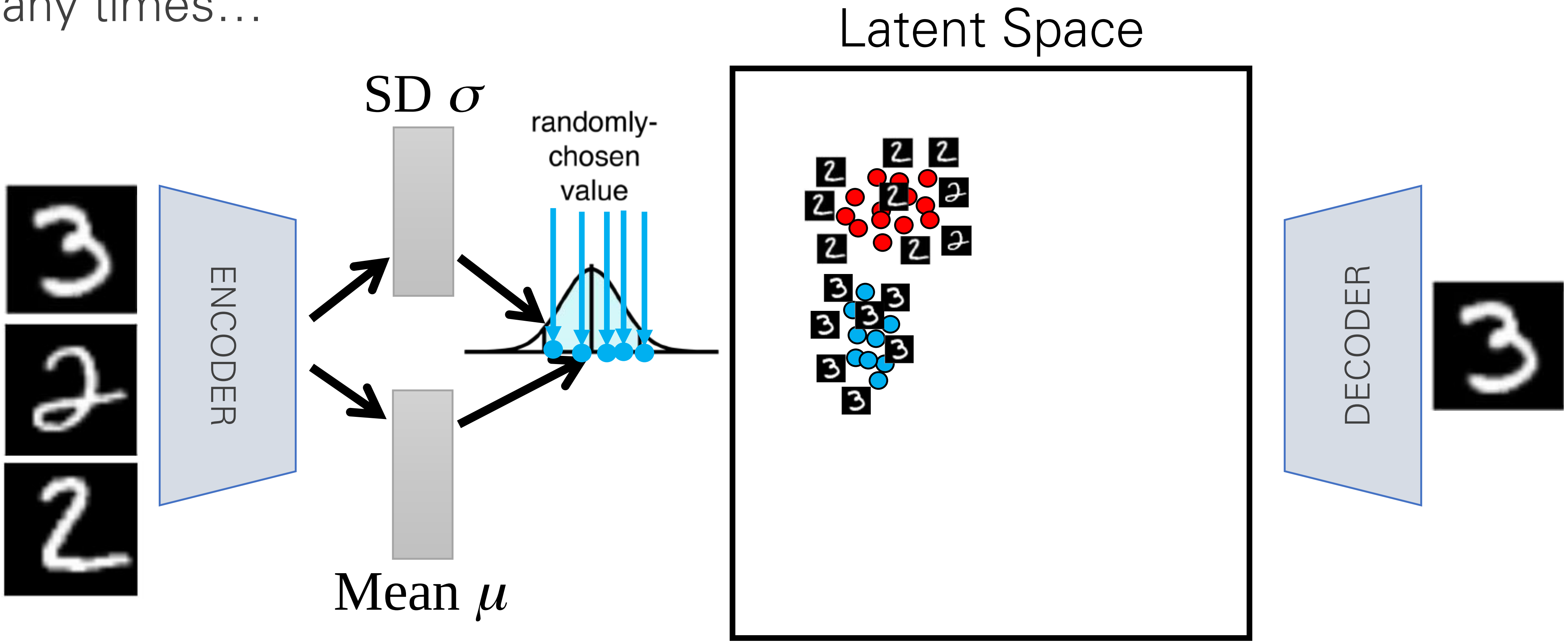
Separability in Variational Autoencoders

Training on 3's again



Separability in Variational Autoencoders

Many times…



Lecture overview

- Motivation for Variational Autoencoders (VAEs)
- Mechanics of VAEs
- Separatibility of VAEs
- **Training of VAEs**
- Evaluating representations
- Vector Quantized Variational Autoencoders (VQ-VAEs)

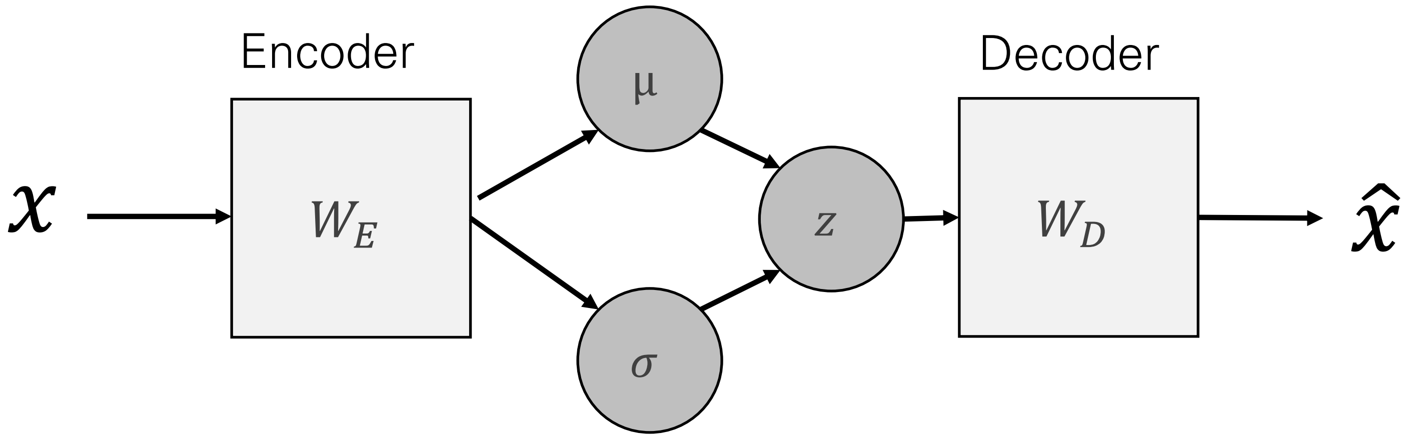
Disclaimer: Much of the material and slides for this lecture were borrowed from

—Pavlov Protopapas, Mark Glickman and Chris Tanner's Harvard CS109B class

—Andrej Risteski's CMU 10707 class

—David McAllester's TTIC 31230 class

Training



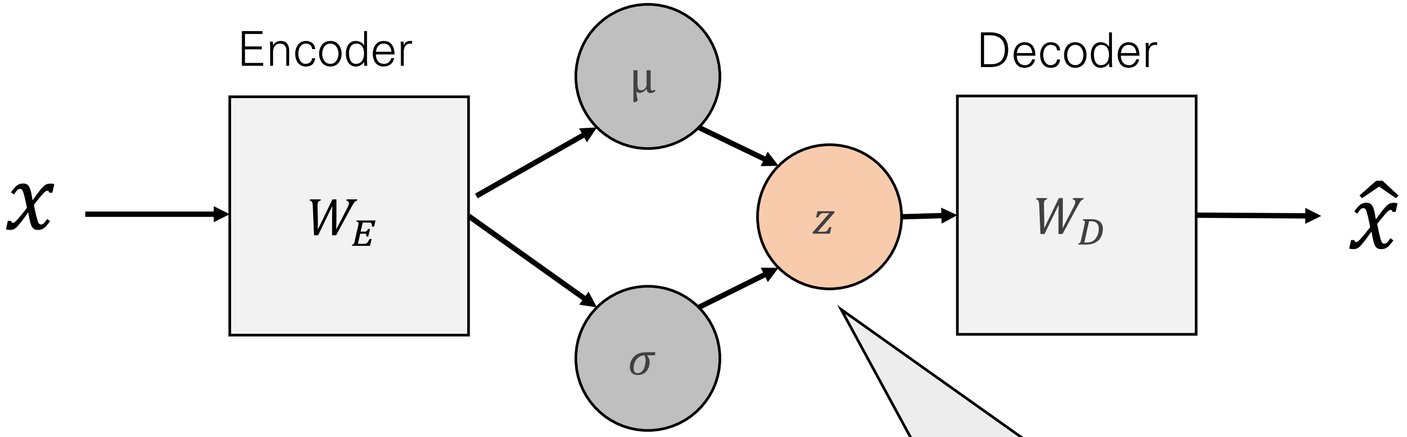
Training means learning W_E and W_D.

- Define a loss function $\mathcal{L}$
- Use stochastic gradient descent (or Adam) to minimize $\mathcal{L}$

The Loss function:

- Reconstruction error: $\mathcal{L}_R = \frac{1}{n}\sum_i (x_i - \hat{x}_i)^2$
- Similarity between the probability of z given x, $\mathrm{p}(z|x)$, and some predefined probability distribution $\mathrm{p(z)}$, which can be computed by Kullback-Leibler divergence (KL): $KL(p(z|x)||p(z))$

Bayesian AE



Parameters of the model (θ is z)

Bayes rule:

$$p(\theta|D) \propto p(D|\theta)p(\theta)$$

Posterior for our parameters, z is:

$$p(z|x,\hat{x}) \propto p(\hat{x}|z,x)p(z)$$

Posterior predictive, probability to see $\hat{x}$ given x; **this is INFERENCE**:

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

Decoder: NN

Posterior

Bayesian AE

The posterior, $P(z|x, \hat{x})$, can be sampled with MCMC, i.e. no minimization of Loss function. How?

1. Set the priors, $p(z)$
2. Define the likelihood, $P(\hat{x}|z, x)$
3. Propose a new z* and:
    a. check if $P(z^*|x, \hat{x})/P(z|x, \hat{x})$ >1: accept, z^*
    b. If $P(z^*|x, \hat{x})/P(z|x, \hat{x})$ <1 throw a random coin and accept/reject z^*
4. This will converge to true $P(z|x, \hat{x})$!
5. Calculate $P(\hat{x}|x) = \int P(\hat{x}|z, x)P(z|x)dz$ (Note: this is easily done with sample from z and re-weight given the likelihood)

DOABLE!

Variational AE

Problem: z is the dimensionality of your latent space, which can be too large. In other words this $\int p(\hat{x}|z,x)p(z|x)dz$ becomes intractable.

Instead we turn this into a minimization problem – Variational Calculus
Find a q$(z|x)$ that is similar to $p(z|x)$ by minimizing their difference.

After some math:

Reconstruction Loss

Proposal distribution should resemble a Gaussian

$$-\mathbf{E}_{z\sim q_\phi(z|x)} \log\left(p_\theta\left(x|z\right)\right) + KL\left(q_\phi\left(z|x\right)\middle\| p_\theta(z)\right)$$

Evidence Lower BOund (ELBO)

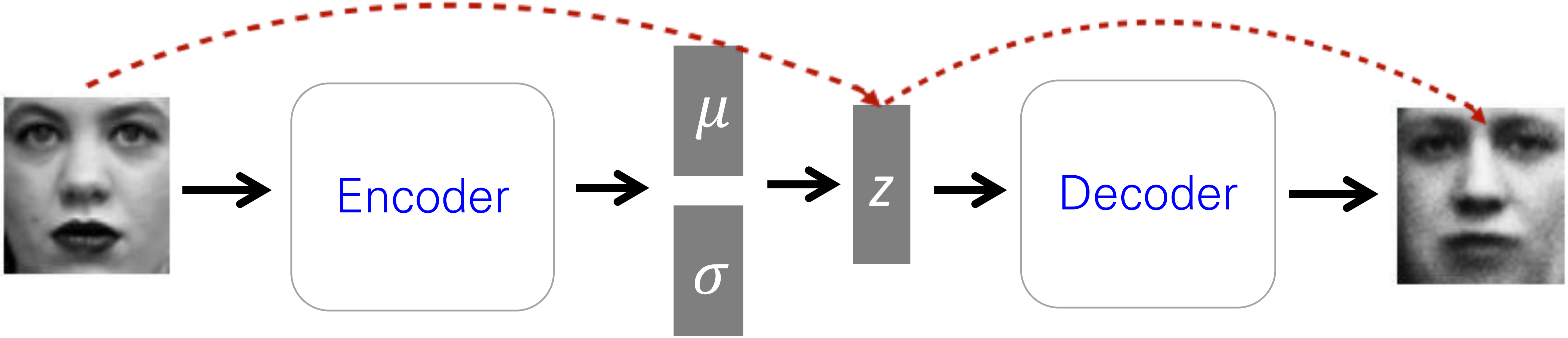
Training VAE

- Apply stochastic gradient descent (SGD)

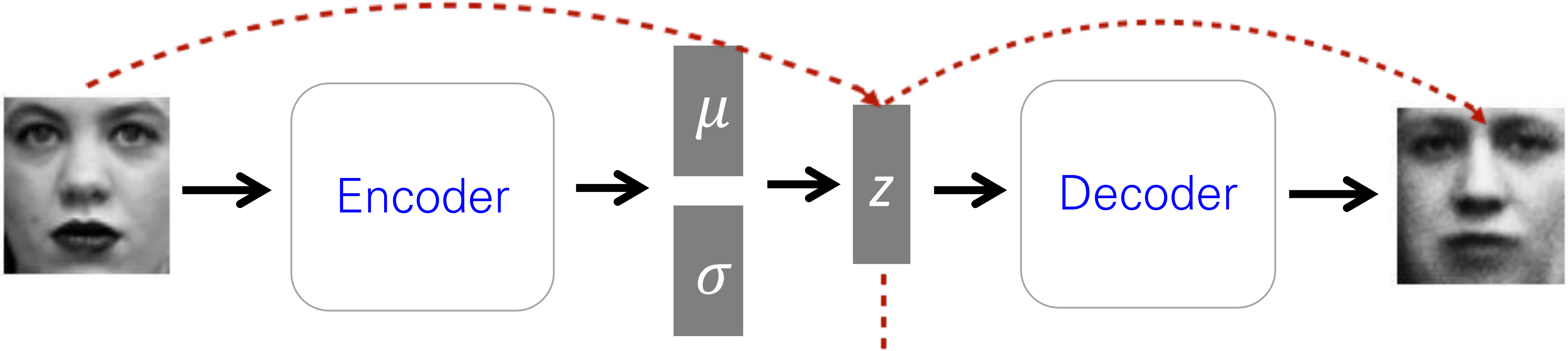
Problem:

- Sampling step not differentiable
- Use a re-parameterization trick
  - Move sampling to input layer, so that the sampling step is independent of the model

Reparametrization Trick

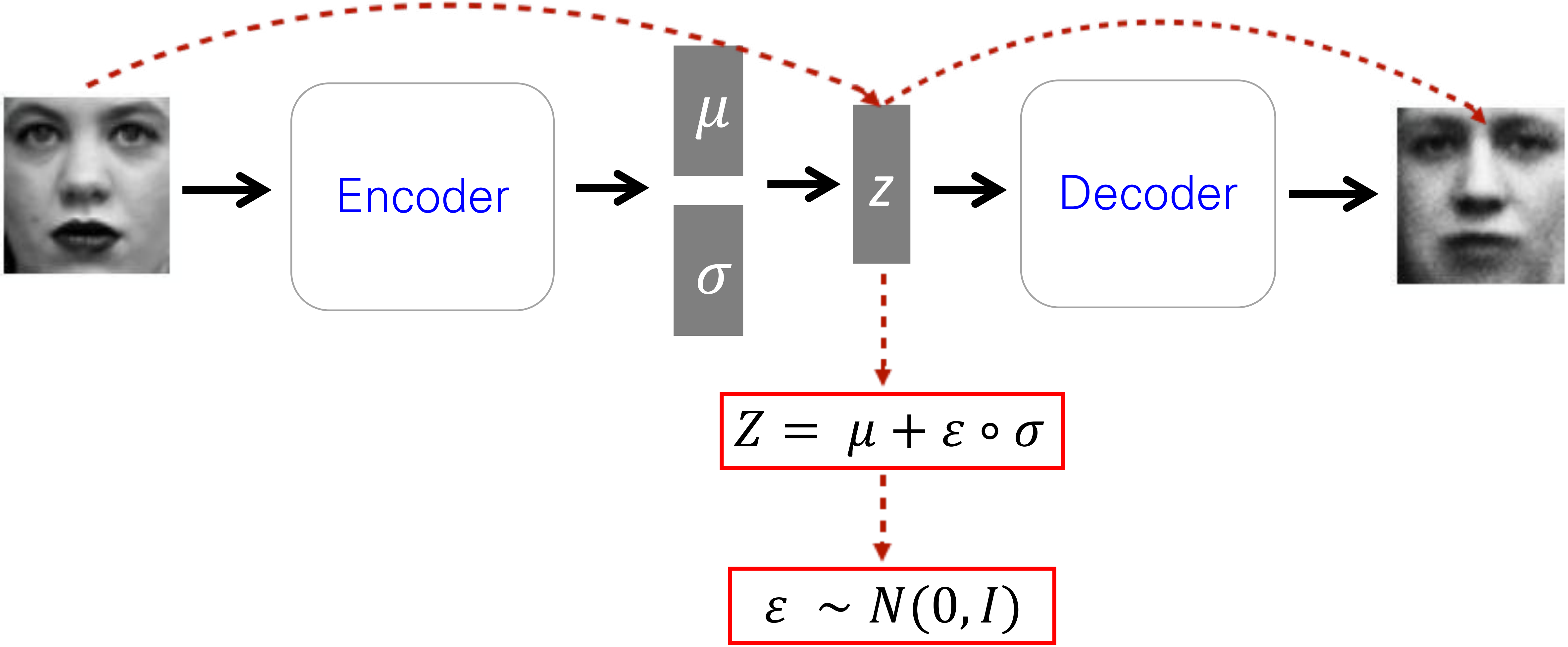


Reparametrization Trick



$$\boxed{Z = \mu + \varepsilon \circ \sigma}$$

Reparametrization Trick



$$Z = \mu + \varepsilon \circ \sigma$$

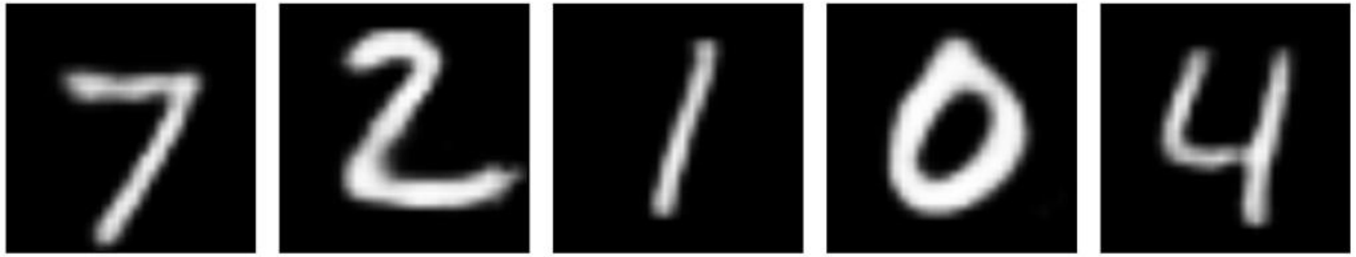
$$\varepsilon \sim N(0, I)$$

Training VAE

Traditional AE:

Input Image:

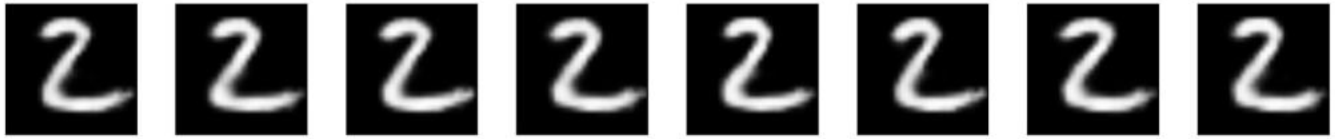
Output Images:



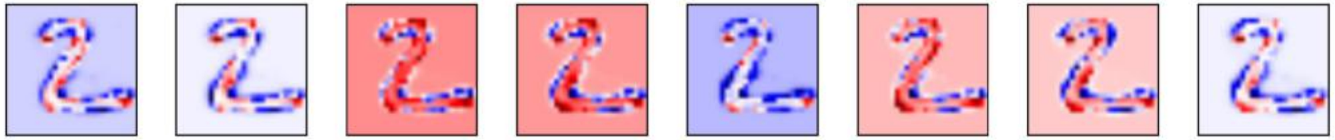
Variational AE:

Input Image:

Output Images:

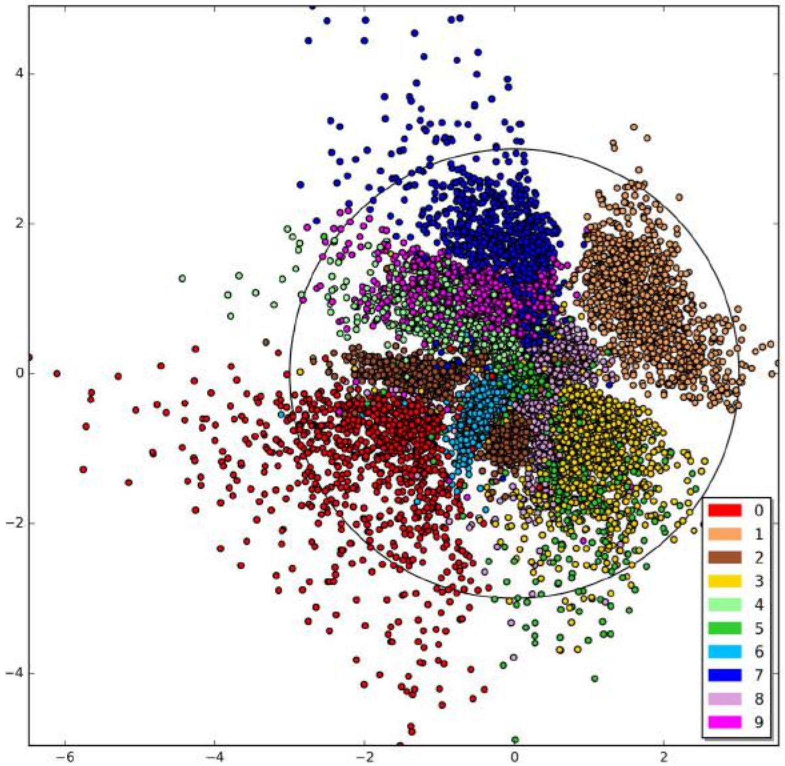


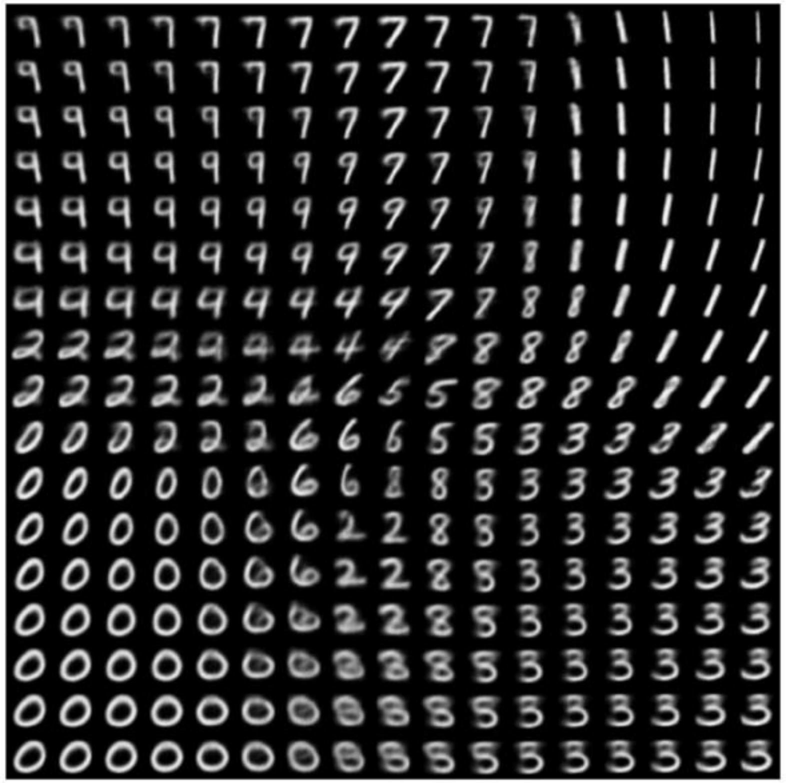
Difference:



Latent space of VAE

- More separable than AE
- Because of the prior N(0,1) everything is center at (0,0) with spread of approx 1.





Lecture overview

- Motivation for Variational Autoencoders (VAEs)
- Mechanics of VAEs
- Separatibility of VAEs
- Training of VAEs
- **Evaluating representations**
- Vector Quantized Variational Autoencoders (VQ-VAEs)

Disclaimer: Much of the material and slides for this lecture were borrowed from

—Pavlov Protopapas, Mark Glickman and Chris Tanner's Harvard CS109B class

—Andrej Risteski's CMU 10707 class

—David McAllester's TTIC 31230 class

Desiderata for representations

What do we want out a representation?

Many possible answers here. First, a few uncontroversial desiderata:

- **Interpretability**: if the derived features are semantically meaningful, and interpretable by a human, they can be easily evaluated.
  (e.g. noisy-OR: "features" are diseases a patient has)

  Sparsity of a representation is an important subcase: "explanatory" features for sample can be examined if there are a small number of them.

- **Downstream usability:** the features are "useful" for downstream tasks. Some examples:

  Improving label efficiency: if, for a task, a linear (or otherwise "simple") classifier can be trained on features and it works well, smaller # of labeled samples are needed.

Desiderate for representations

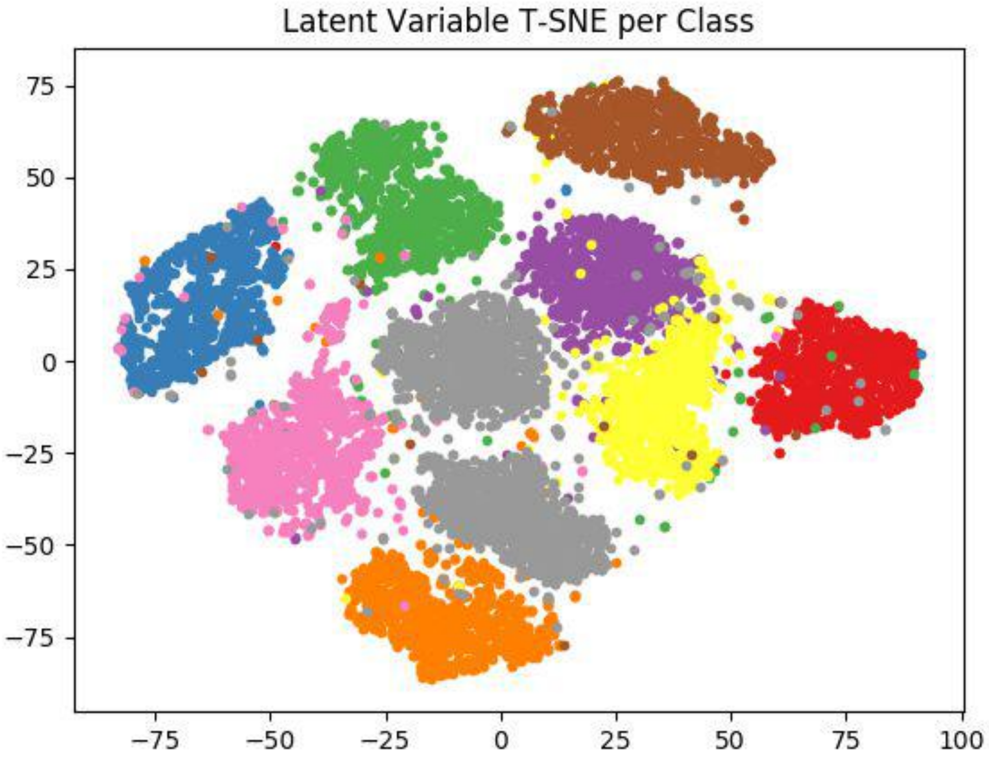
- **Obvious issue:** interpretability and “usefulness” are not easily mathematically expressed. We need some “proxies” that induce such properties.

  This is a lot more contraversial – here we survey some general desiderata, proposed as early as Bengio-Courville-Vincent ’14:

- **Hierarchy/compositionality**: video/images/text/ are expected to have hierarchical structure – depth helps induce such structure.
- **Semantic clusterability**: features of the same ”semantic class” (e.g. images in the same category) are clustered.
- **Linear interpolation**: in representation space, linear interpolations produce meaningful data points (i.e. ”latent space is convex”). Sometimes called manifold flattening.
- **Disentangling**: features capture “independent factors of variation” of data. (Bengio-Courville-Vincent ’14). Has been very popular in modern unsupervised learning, though many potential issues with it.

Semantic clustering

- **Semantic clusterability:** features of the same "semantic class" (e.g. images in the same category) are clustered together.



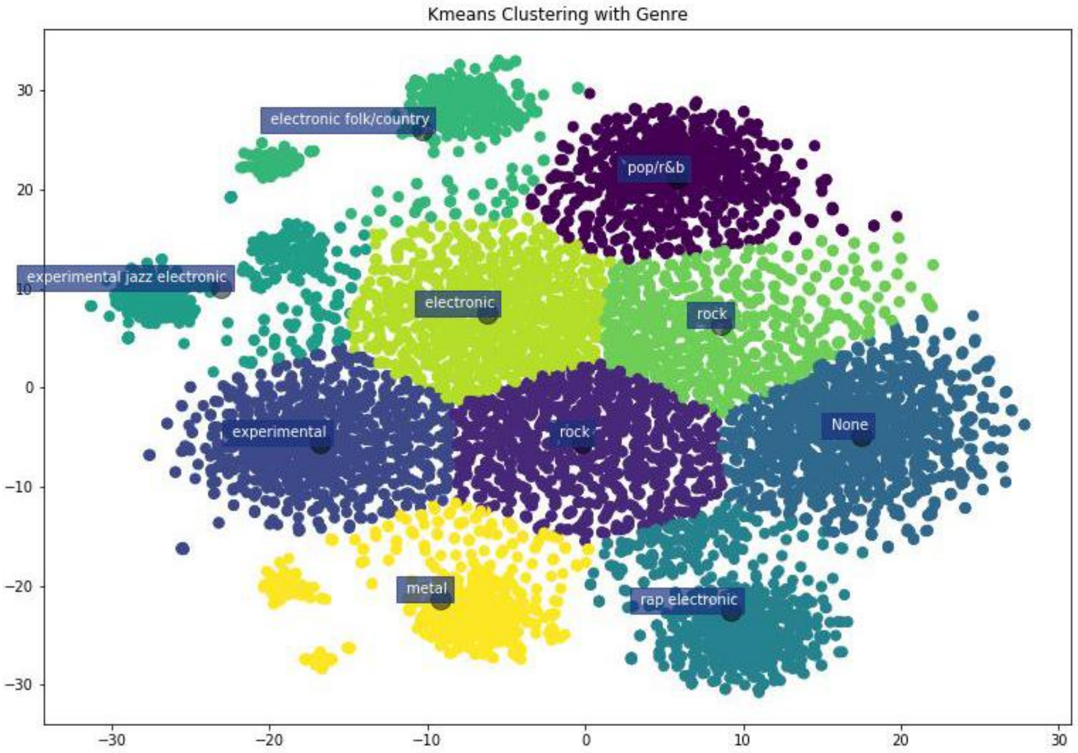
The intuition:

If semantic classes are linearly (or other simple function) separable, and labels on downstream tasks depend linearly on semantic classes – can afford to learn a simple classifier!!

t-SNE projection of VAE-learned features of the 10 MNIST classes.
Image from https://pyro.ai/examples/vae.html

Semantic clustering

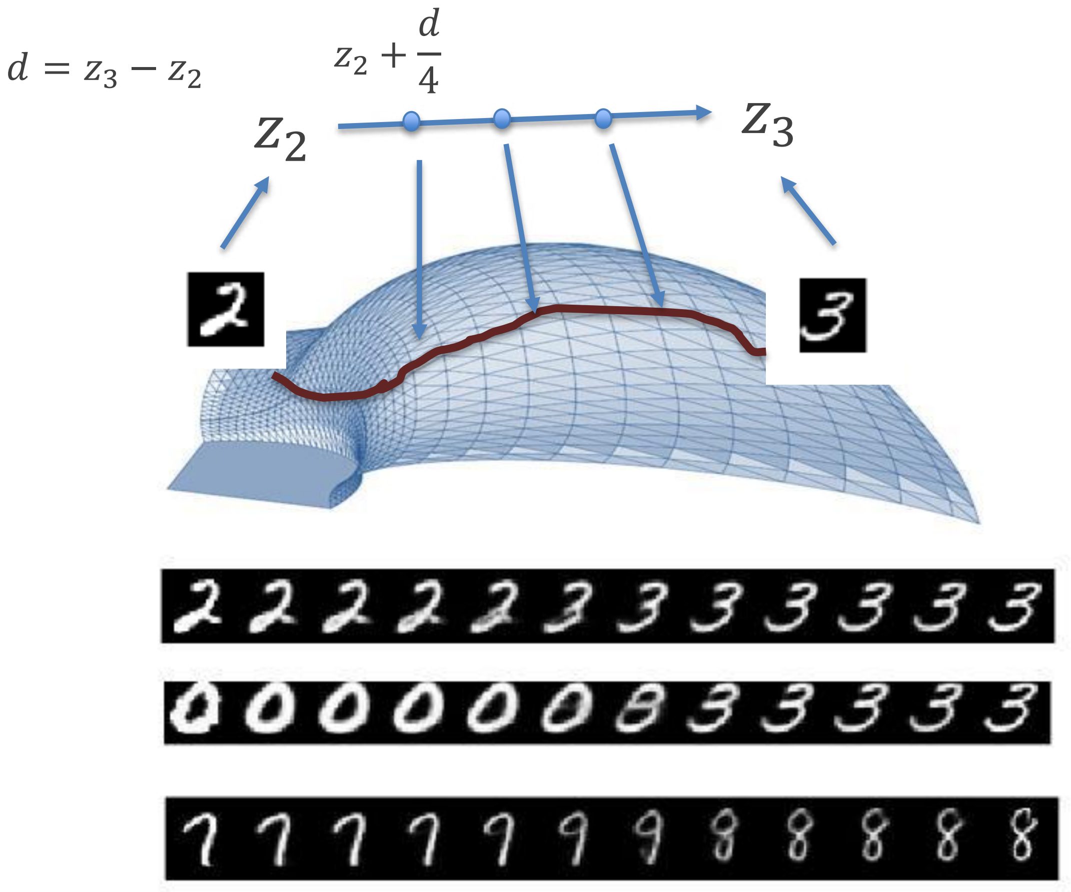
- **Semantic clusterability:** features of the same "semantic class" (e.g. images in the same category) are clustered together.



t-SNE projection of word embeddings for artists (clustered by genre). Image from https://medium.com/free-code-camp/learn-tensorflow-the- word2vec-model-and-the-tsne-algorithm-using-rock-bands-97c99b5dcb3a

Linear interpolation

- **Linear interpolation:** in representation space, linear interpolations produce meaningful data points. (i.e. “latent space is convex”)



Interpolations for a VAE trained on MNIST.

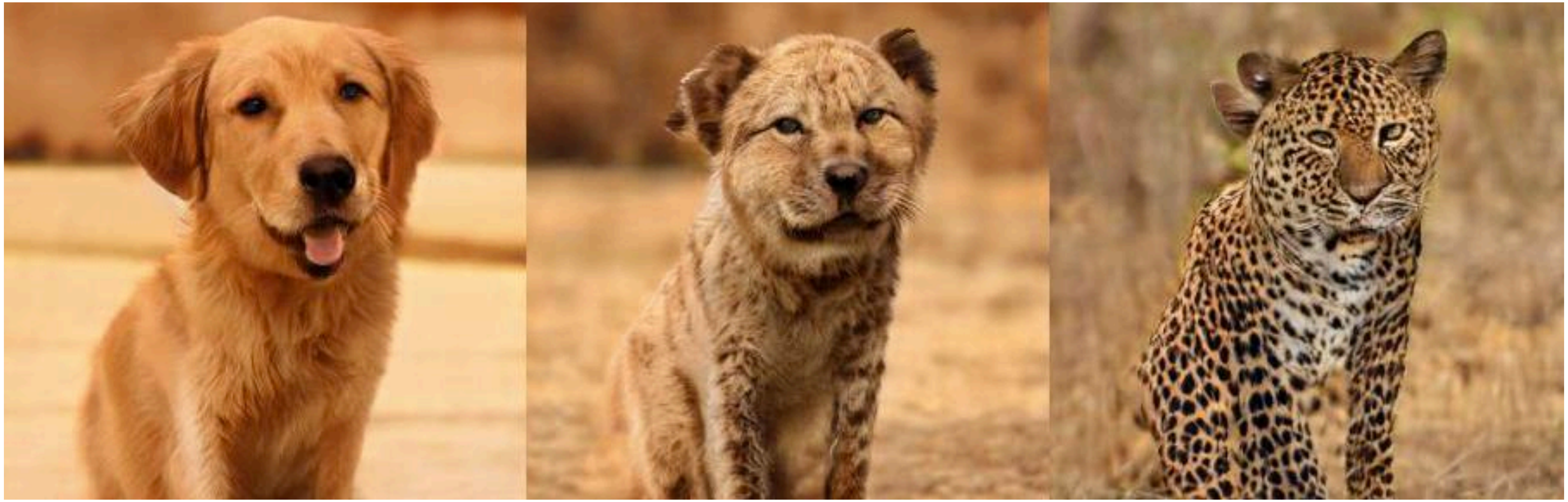
The intuition:

The data manifold is complicated/curved.

The latent variable manifold is a convex set – moving in straight lines keeps us on it.

Linear interpolation

- **Linear interpolation:** in representation space, linear interpolations produce meaningful data points. (i.e. “latent space is convex”)



Interpolations for a BigGAN, image from https://thegradient.pub/bigganex-a-dive-into- the-latent-space-of-biggan/

Disentangled representations

- **Disentangling**: features capture "independent factors of variation" of data. (Bengio-Courville-Vincent '14).
- For concreteness, let's assume that we have a latent variable model for data with latent variables $\mathbf{z}$, observables $\mathbf{x}$, and joint distribution $p_\theta(\mathbf{z}, \mathbf{x})$
- There are (at least) two ways to formalize this.

Prior disentangling: is a product distribution, i.e. $\boldsymbol{p_\theta(\mathbf{z}) = \Pi_i p_\theta(z_i)}$

Classical example: ICA (independent component analysis)

Posterior disentangling: fit a variational posterior $\boldsymbol{q_\theta}$ s.t. $\boldsymbol{q_\theta(\mathbf{z}|x)}$ is (on average over $\mathbf{x}$) a product distribution

In other words $\displaystyle\int_x \boldsymbol{q_\theta(z|x)p(x)dx}$ -- usually called the aggregate posterior – is close to a product distribution.

Disentangled representations

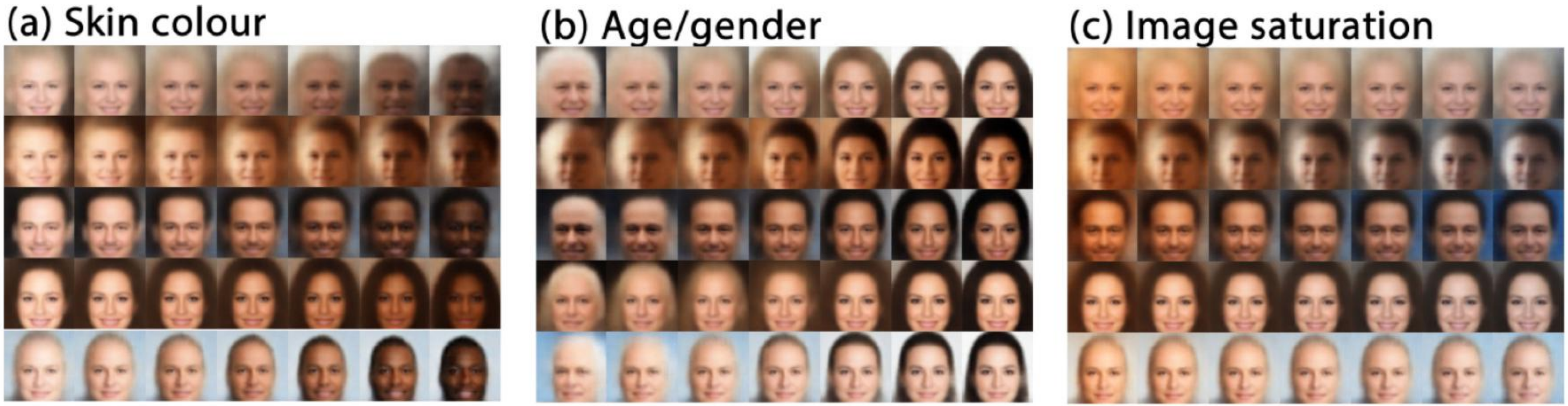


Figure 4: **Latent factors learnt by β-VAE on celebA:** traversal of individual latents demonstrates that β-VAE discovered in an unsupervised manner factors that encode skin colour, transition from an elderly male to younger female, and image saturation.

- Posterior disentangling in β-VAE. To produce plots, infer latent variable for an image, then change a single latent variable gradually.

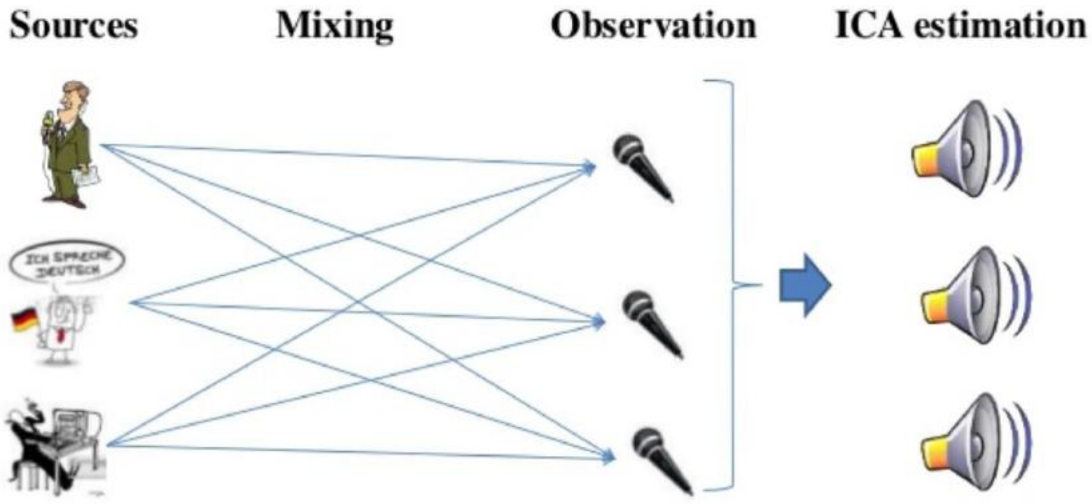
Irina Higgins et al. β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. ICLR 2017.

Prior disentangling

- **Prior disentangling:** $p_{\boldsymbol{\theta}}(\boldsymbol{z})$ is a product distribution, i.e. $p_{\boldsymbol{\theta}}(\boldsymbol{z}) = \Pi_i p_{\boldsymbol{\theta}}(\boldsymbol{z_i})$

Classical example: ICA (independent component analysis), also called the "cocktail party problem".

Assume data is generated as



If z has an independent, non-Gaussian prior, model is identifiable and efficiently learnable. (See, e.g. Frieze-Jerum-Kannan '96, Anandkumar et al '12)

Other examples: noisy-OR networks (diseases are independent), general Bayesian nets, viewing top variables as z's, GANs, …

Posterior disentanglement in VAEs

- Recall the “regularization” view of the VAEs objective:

$$\underbrace{\Sigma_x \mathbb{E}_{q(h^L|x)} \log p(x|h^L)}_{\text{"Reconstruction" error}} - \underbrace{KL(q(h^L|x)||p(h^L))}_{\text{"Regularization towards prior"}}$$

- Consider a prior which is a product distribution (e.g. standard Gaussian): The KL term implicitly penalizes distributions for which

$$\sum_x KL(q(h^L|x)||p(h^L)) \approx \mathbb{E}_{x\sim p^*} KL(q(h^L|x)||p(h^L))$$

is large – i.e. the aggregated posterior is far from a product distribution

Posterior disentanglement in VAEs

- Recall the “regularization” view of the VAEs objective:

$$\underbrace{\Sigma_x \mathbb{E}_{q(h^L|x)} \log p(x|h^L)}_{\text{"Reconstruction" error}} - \underbrace{KL(q(h^L|x)||p(h^L))}_{\text{"Regularization towards prior"}}$$

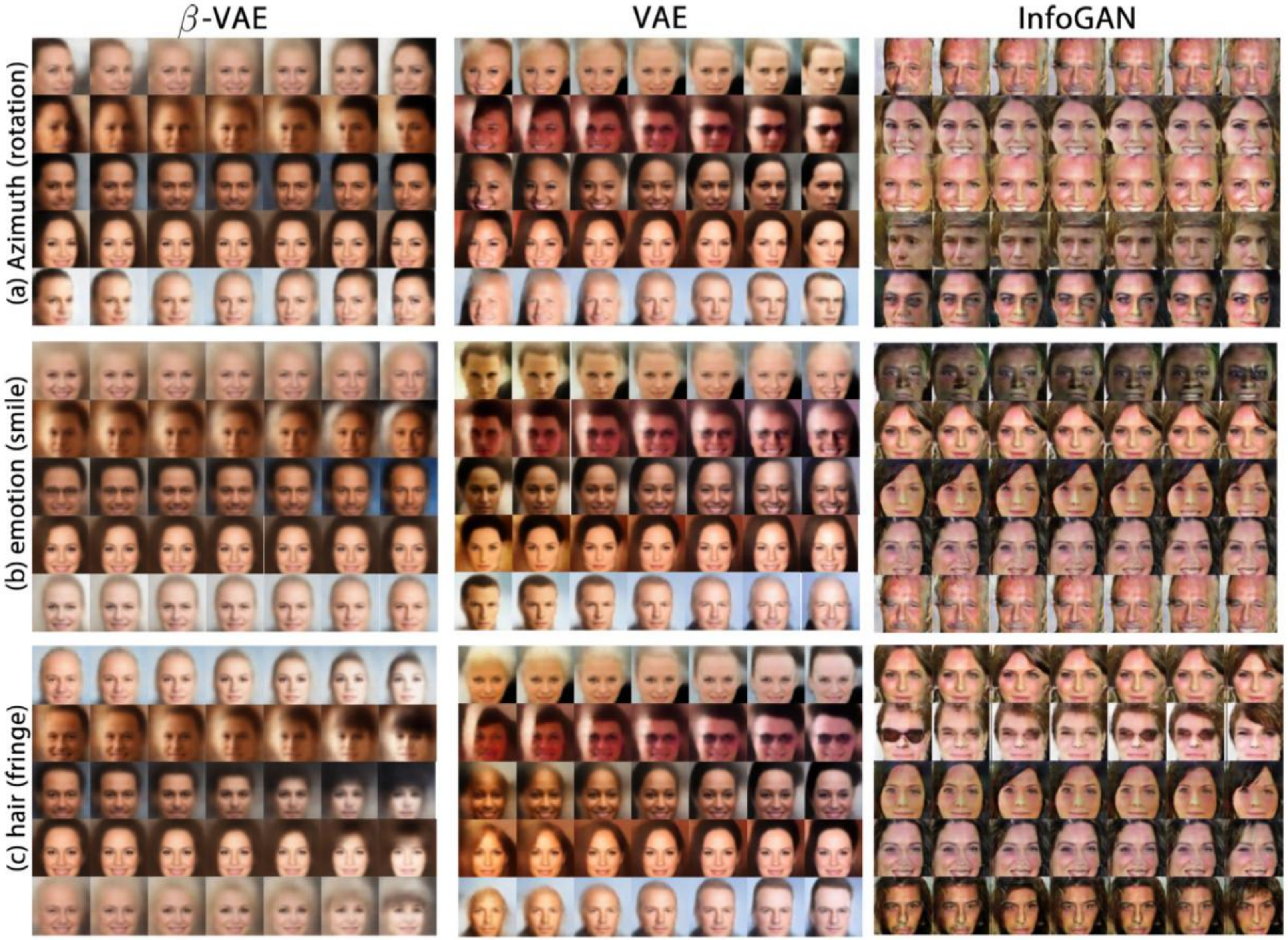
The KL term implicitly penalizes distributions for which

$$\sum_x KL(q(h^L|x)||p(h^L)) \approx \mathbb{E}_{x\sim p^*} KL(q(h^L|x)||p(h^L))$$

The idea of Higgins et al ’17 introduce a “weighting” factor to put more weight on reconstruction or disentanglement:

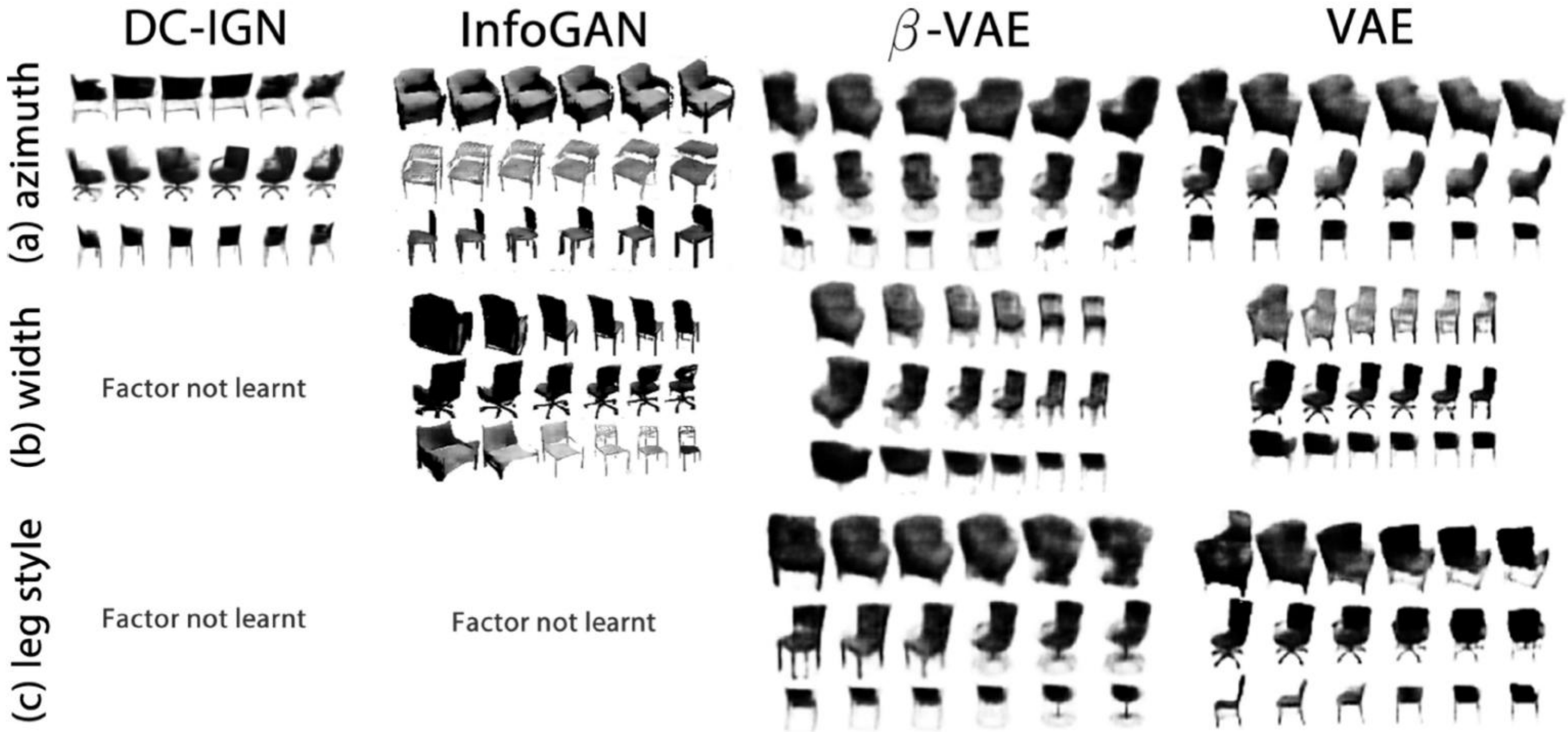
β-VAE objective: $$\sum_x \mathbb{E}_{q(h^L|x)} \log p(x|h^L) - \beta KL(q(h^L|x)||p(h^L))$$

Posterior disentanglement in VAEs



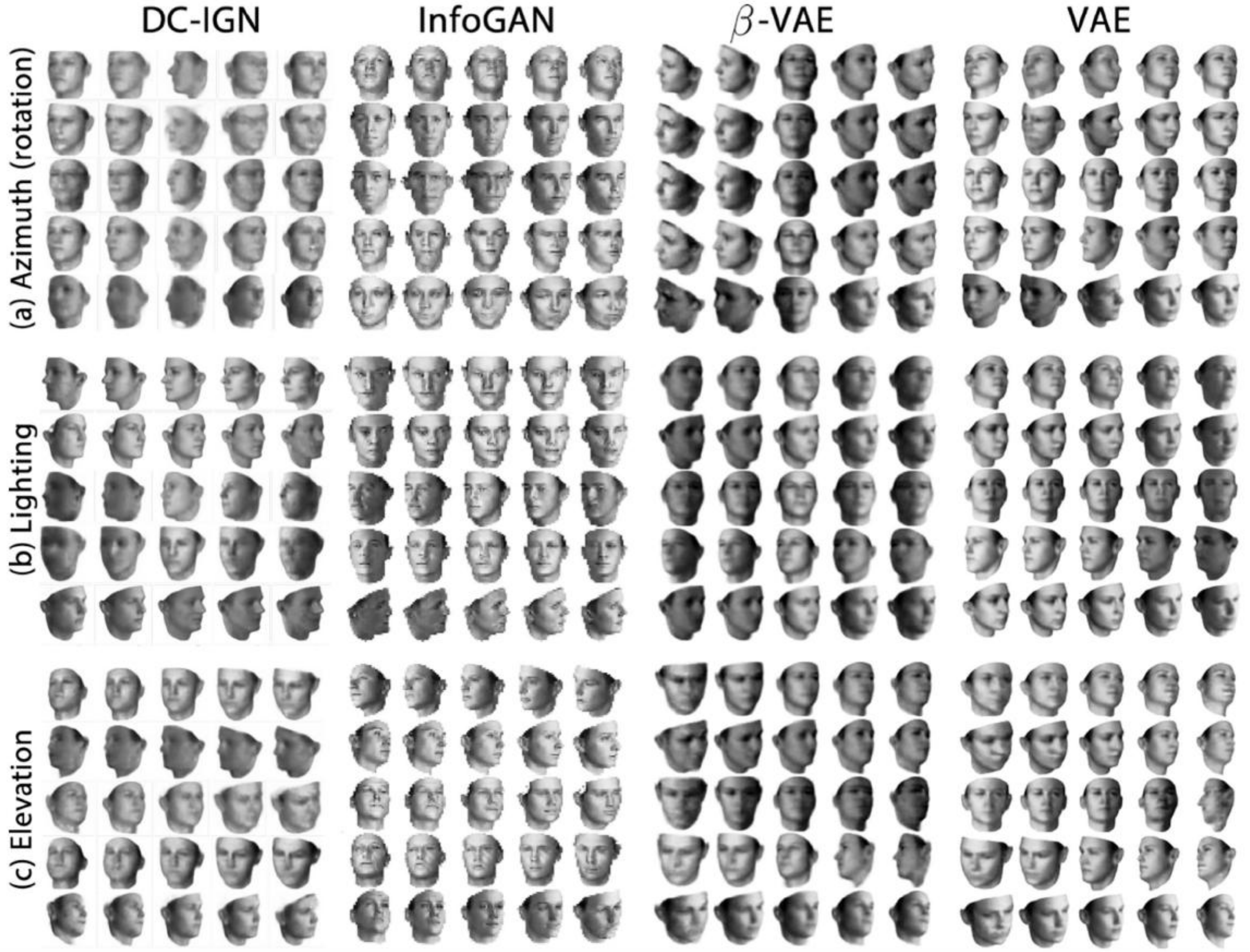
Irina Higgins et al. β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. ICLR 2017.

Posterior disentanglement in VAEs



Irina Higgins et al. β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. ICLR 2017.

Posterior disentanglement in VAEs



Irina Higgins et al. β-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework. ICLR 2017.

Measuring disentanglement

- Metrics are typically defined assuming access to a dataset with K “ground-truth” variation factors.

BetaVAE metric: based on "linear separability" of factors

```
Generate a training set of samples as follows:
  Sample a batch of B samples as follows:
    Pick a ground-truth variation factor k uniformly at random from [K].
    Generate two sets of “ground truth” latent factors, $\mathbf{v}_1, \mathbf{v}_2 \in R^K$, s.t.
    $(\mathbf{v}_1)_k = (\mathbf{v}_2)_k$ , and other coords are independently, randomly sampled.
    Generate images $\mathbf{x}_1, \mathbf{x}_2$ from $\mathbf{v}_1, \mathbf{v}_2$.
    Infer latent vars $\mathbf{z}_1, \mathbf{z}_2$ using model we are evaluating. (e.g. encoder in VAE)
Calculate average $\mathbf{z}_{avg}$ of $|\mathbf{z}_1 - \mathbf{z}_2|$ in batch, add $(\mathbf{z}_{avg}, k)$ to training set.
Train linear predictor on training set, evaluate it’s test performance.
```

Measuring disentanglement

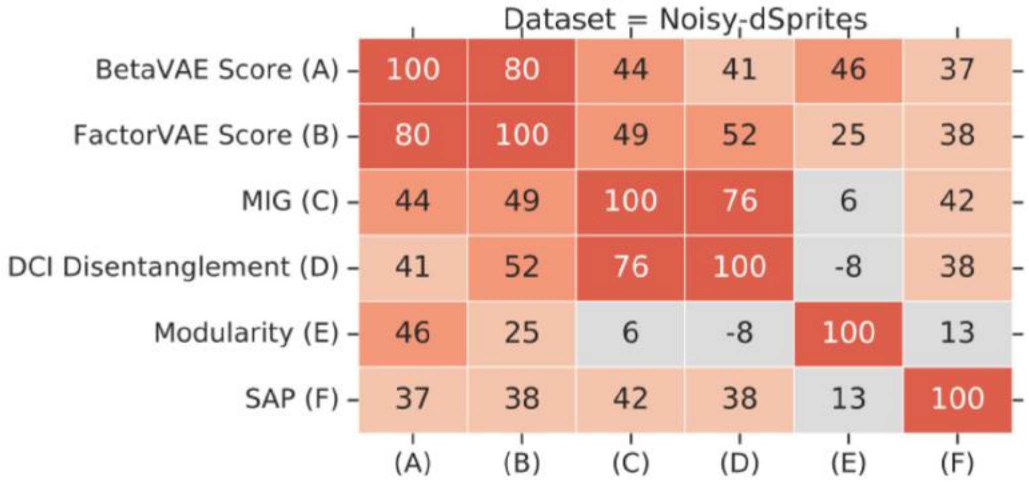
BetaVAE metric: based on "linear separability" of factors

```
Generate a **training set** of samples as follows:
  Sample a **batch** of B samples as follows:
    Pick a **ground-truth variation factor k** uniformly at random from [K].
    Generate two sets of “ground truth” latent factors, $\mathbf{v}_1, \mathbf{v}_2 \in R^K$, s.t.
    $(\mathbf{v}_1)_k = (\mathbf{v}_2)_k$ , and other coords are independently, randomly sampled.
    Generate **images** $\mathbf{x}_1, \mathbf{x}_2$ from $\mathbf{v}_1, \mathbf{v}_2$.
    Infer latent vars $\mathbf{z}_1, \mathbf{z}_2$ using model we are evaluating. (e.g. encoder in VAE)
Calculate average $\mathbf{z}_{avg}$ of $|\mathbf{z}_1 - \mathbf{z}_2|$ in batch, add $(\mathbf{z}_{avg}, k)$ to training set.
Train linear predictor on training set, evaluate it’s test performance.
```

- Intuition: averaging should make coords in $\mathbf{z}_{avg}$ different from k smaller, thus linear classifier should “focus” on k.
- Many variants of this exist. (e.g. FactorVAE, mutual information gap, etc.)

Measuring disentanglement

- Locatello et al ’19, “Challenging Common Assumptions in the Unsupervised Learning of Disentangled Representations” (Best paper award ar ICML’19): A large-scale study of disentanglement measures, as well as gen. models.



Dataset = Noisy-dSprites

	(A)	(B)	(C)	(D)	(E)	(F)
BetaVAE Score (A)	100	80	44	41	46	37
FactorVAE Score (B)	80	100	49	52	25	38
MIG (C)	44	49	100	76	6	42
DCI Disentanglement (D)	41	52	76	100	-8	38
Modularity (E)	46	25	6	-8	100	13
SAP (F)	37	38	42	38	13	100

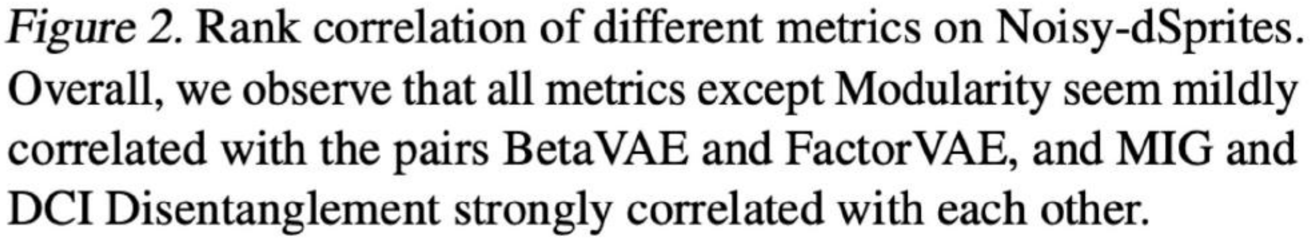
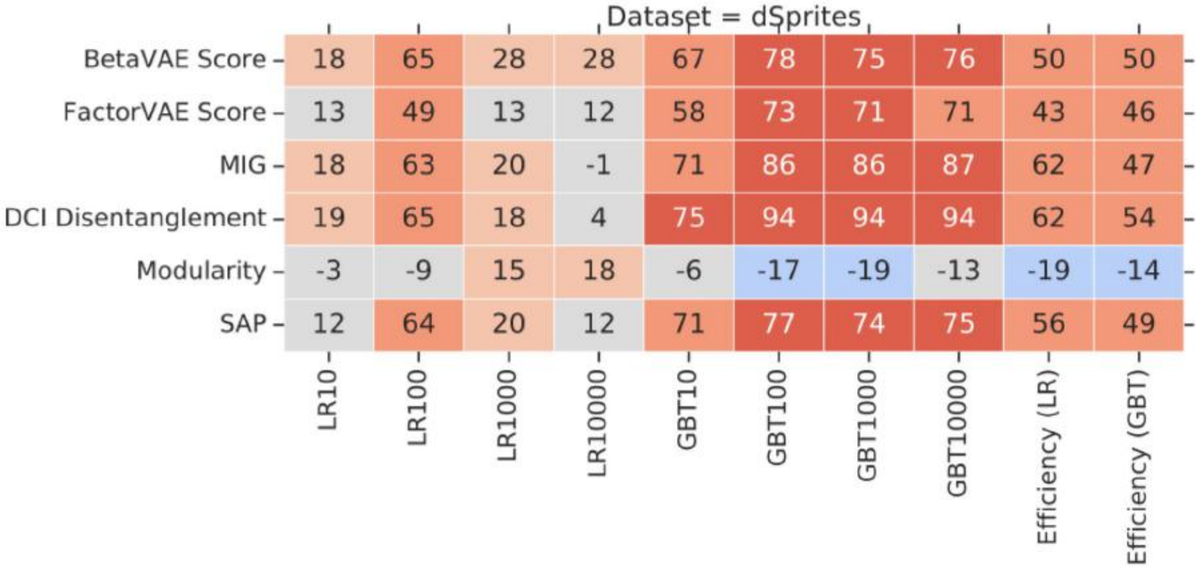


Figure 2. Rank correlation of different metrics on Noisy-dSprites. Overall, we observe that all metrics except Modularity seem mildly correlated with the pairs BetaVAE and FactorVAE, and MIG and DCI Disentanglement strongly correlated with each other.

Usefulness of disentanglement?

- Downstream classification task: predict true ground-truth factors (w/ multiclass logistic regression)
- Careful to extrapolate too much – task/setup is a little contrived.



Dataset = dSprites

	LR10	LR100	LR1000	LR10000	GBT10	GBT100	GBT1000	GBT10000	Efficiency (LR)	Efficiency (GBT)
BetaVAE Score	18	65	28	28	67	78	75	76	50	50
FactorVAE Score	13	49	13	12	58	73	71	71	43	46
MIG	18	63	20	-1	71	86	86	87	62	47
DCI Disentanglement	19	65	18	4	75	94	94	94	62	54
Modularity	-3	-9	15	18	-6	-17	-19	-13	-19	-14
SAP	12	64	20	12	71	77	74	75	56	49

Figure 5. Rank correlations between disentanglement metrics and downstream performance (accuracy and efficiency) on dSprites.

Locatello et al. Challenging Common Assumptions in the Unsupervised Learning of Disentangled Representations. ICML 2019.

Usefulness of disentanglement?

- Statistical efficiency measure: average accuracy based on 100 samples divided by the average accuracy based on 10,000 samples

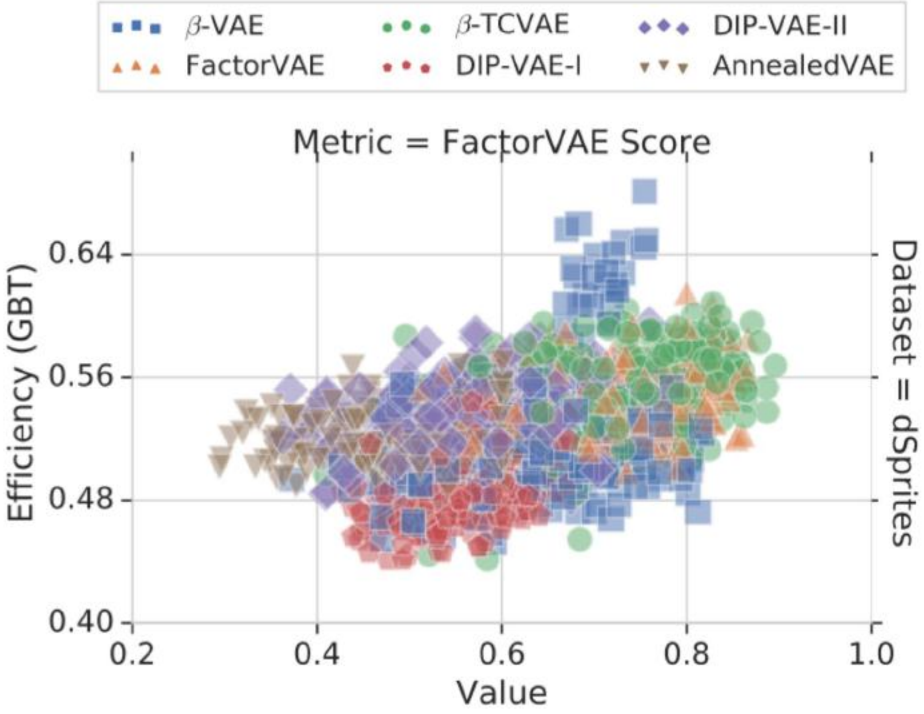


Figure 6. Statistical efficiency of the FactorVAE Score for learning a GBT downstream task on dSprites.

Locatello et al. Challenging Common Assumptions in the Unsupervised Learning of Disentangled Representations. ICML 2019.

Issue of ill-posedness?

- Similar issues plague disentangling that do "flat minima": a model can be re-parametrized, s.t. the distribution over the data is unchanged, but it can be arbitrarily more "entangled".
- Thus, **some kind of inductive bias both on model class and data seems necessary**.
- As a simple example: consider. $\mathbf{z} \sim \mathcal{N}(0, \boldsymbol{I})$, let $\mathbf{z}' = \boldsymbol{U}\mathbf{z}$, for any non-identity orthogonal matrix U.
- Then, under any "intuitive" understanding of entangling, $\mathbf{z}'$ seems entangled with $\mathbf{z}$ – small changes of coordinates of z cause global changes in $\mathbf{z}'$.

Locatello et al. Challenging Common Assumptions in the Unsupervised Learning of Disentangled Representations. ICML 2019.

Lecture overview

- Motivation for Variational Autoencoders (VAEs)
- Mechanics of VAEs
- Separatibility of VAEs
- Training of VAEs
- Evaluating representations
- **Vector Quantized Variational Autoencoders (VQ-VAEs)**

Disclaimer: Much of the material and slides for this lecture were borrowed from

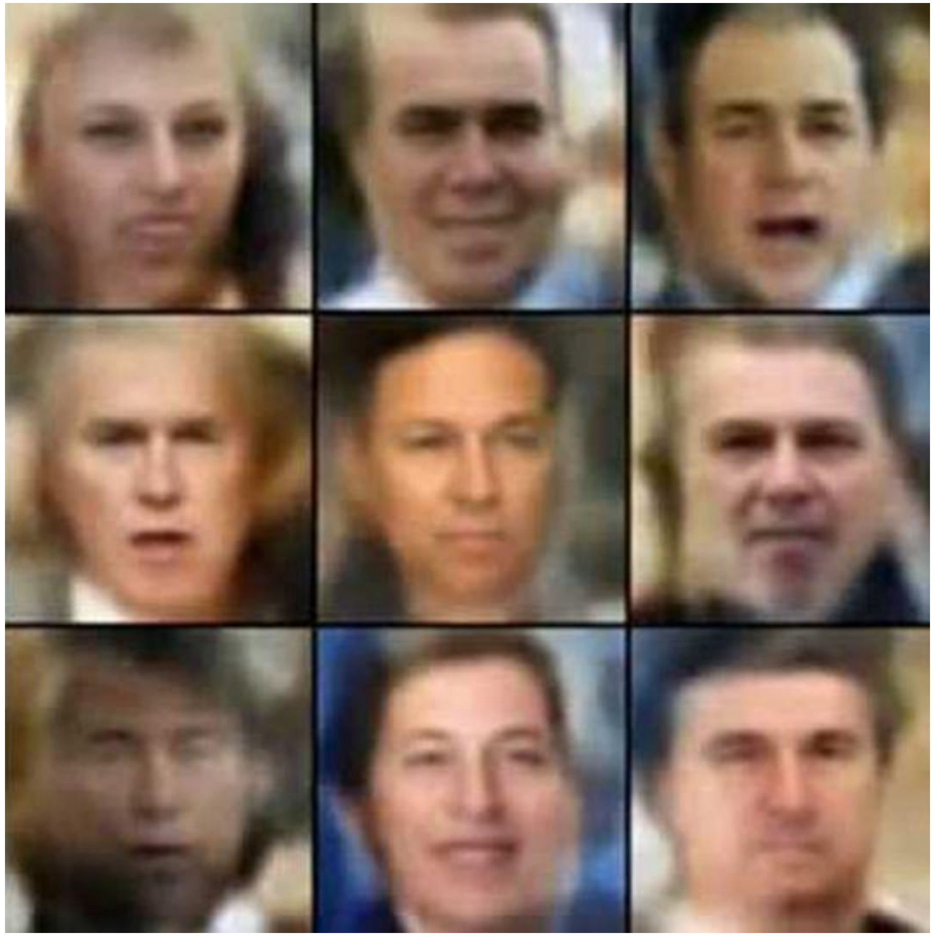
—Pavlov Protopapas, Mark Glickman and Chris Tanner's Harvard CS109B class

—Andrej Risteski's CMU 10707 class

—David McAllester's TTIC 31230 class

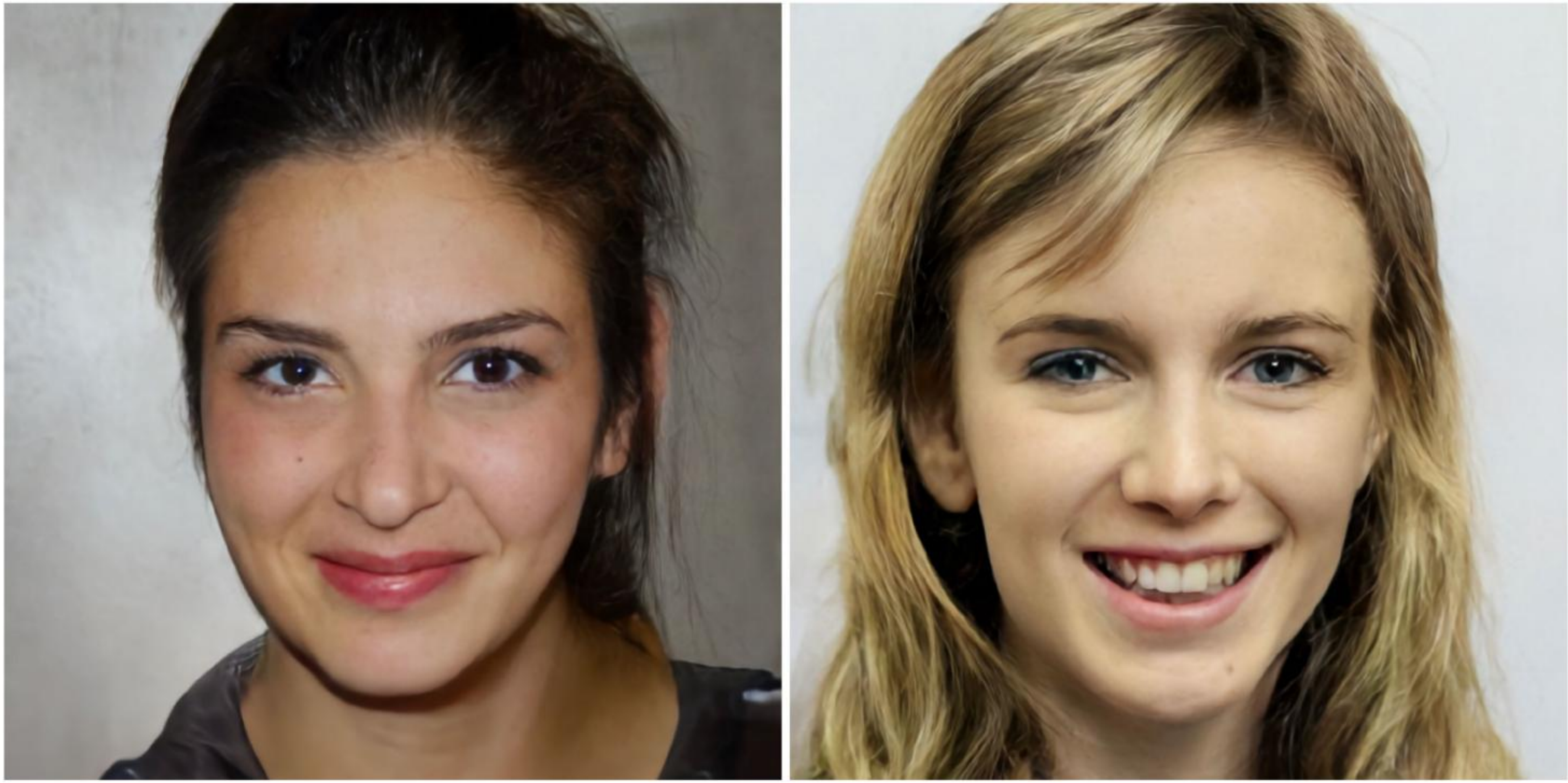
Gaussian VAEs 2013

Sample $z \sim \mathcal{N}(0, I)$ and compute $y_\Phi(z)$



[Alec Radford]

Vector Quantized VAEs (VQ-VAE) 2019



VQ-VAE-2, Razavi et al., NeurIPS 2019

Vector Quantized VAEs (VQ-VAE) 2019

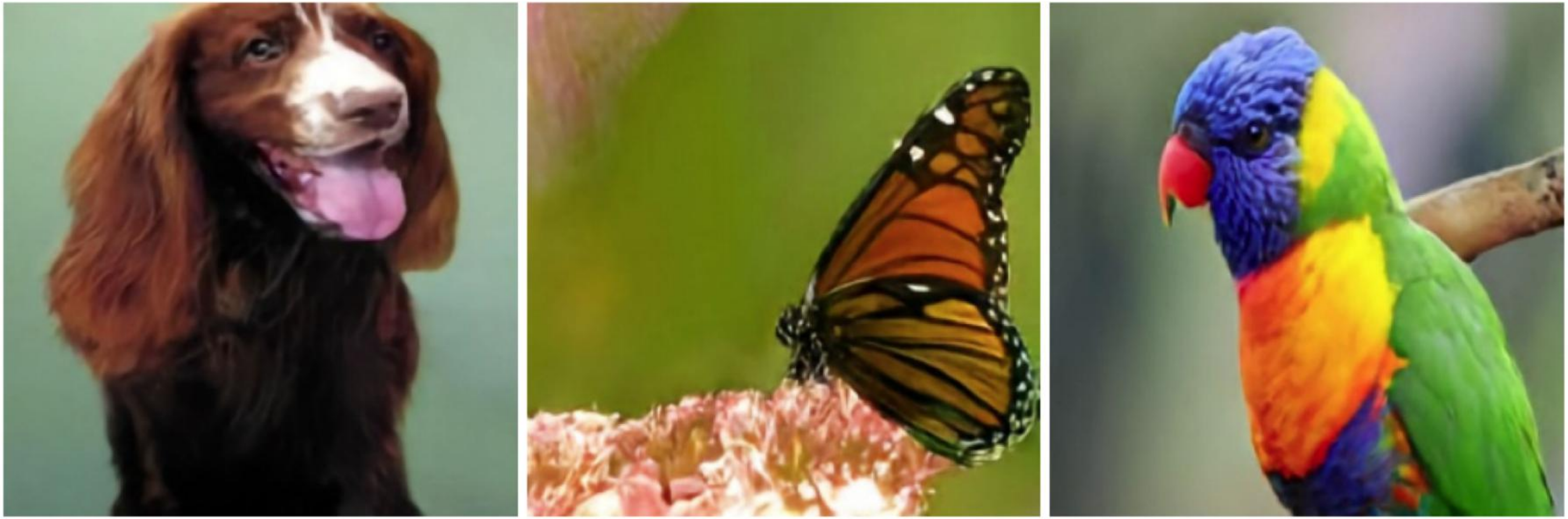


Figure 1: Class-conditional 256x256 image samples from a two-level model trained on ImageNet.

VQ-VAE-2, Razavi et al., NeurIPS 2019

Vector Quantized VAEs (VQ-VAE)

- VQ-VAEs effectively perform k-means on vectors in the model so as to represent vectors by discrete cluster centers.

- For concreteness we will consider VQ-VAEs on images with a single layer of quantization.

- We use x and y for spatial image coordinates and use s (for signal) to denote images.

VQ-VAE Encoder-Decoder

- We train a dictionary $C[K,I]$ where $C[k,I]$ is the center vector of cluster k.

$$L[X,Y,I] = \mathrm{Enc}_\Phi(s)$$

$$z[x,y] = \operatorname*{argmin}_k \; ||L[x,y,I] - C[k,I]||$$

$$\hat{L}[x,y,I] = C[z[x,y],I]$$

$$\hat{s} = \mathrm{Dec}_\Phi(\hat{L}[X,Y,I])$$

- The “symbolic image” z[X, Y] is the latent variable.

VQ-VAE Training Loss

- We preserve information about the image *s* by minimizing the distortion between $L[X,Y,I]$ and its reconstruction $\hat{L}[X,Y,I]$

$$\Phi^* = \underset{\Phi}{\operatorname{argmin}} \; E_s \, \beta||L[X,Y,I] - \hat{L}[X,Y,I]||^2 + ||s - \hat{s}||^2$$

Parameter-Specific Learning Rates

$$||L[X,Y,I] - \hat{L}[X,Y,I]||^2 = \sum_{x,y} ||L[x,y,I] - C[z[x,y],I]||^2$$

- For the gradient of this they use

$$\text{for } x,y \;\; L[x,y,I].\text{grad} \mathrel{+}= 2\beta(L[x,y,I] - C[z[x,y],I])$$
$$\text{for } x,y \;\; C[z[x,y],I].\text{grad} \mathrel{+}= 2(C[z[x,y],I] - L[x,y,I])$$

- This gives a parameter-specific learning rate for $C[K,I]$.
- Parameter-specific learning rates do not change the stationary points (the points where the gradients are zero).

The Relationship to K-means

$$\text{for } x, y \;\; C[z[x,y], I].\text{grad} \mathrel{+}= 2(C[z[x,y], I] - L[x,y,I])$$

- At a stationary point we get that $C[k, I]$ is the mean of the set of vectors $L[x, y, I]$ with $z[x, y] = k$ (as in K-means).

Straight Through Gradients

- The latent variables are discrete so some approximation to SGD must be used.

- The authors use “straight-through” gradients.

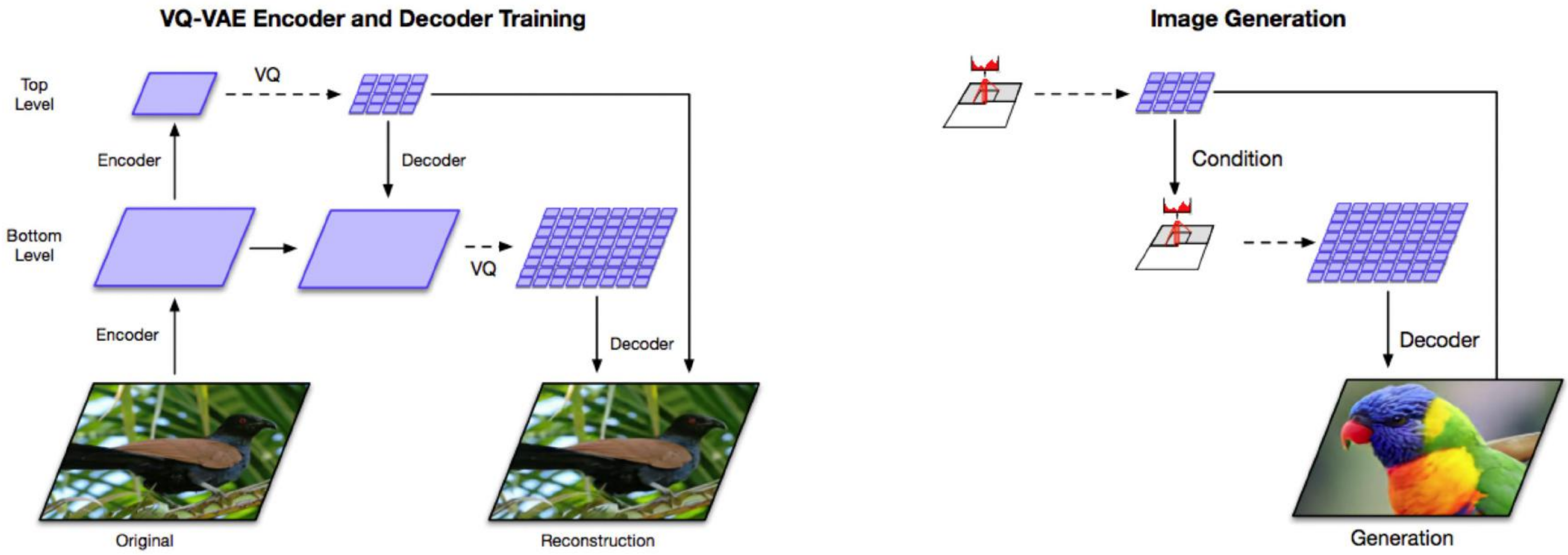
$$\text{for } x, y \;\; L[x, y, I].\text{grad} \mathrel{+}= \hat{L}[x, y, I].\text{grad}$$

- This assumes low distortion between $L[X, Y, I]$ and $\hat{L}[X, Y, I]$.

Training Phase II

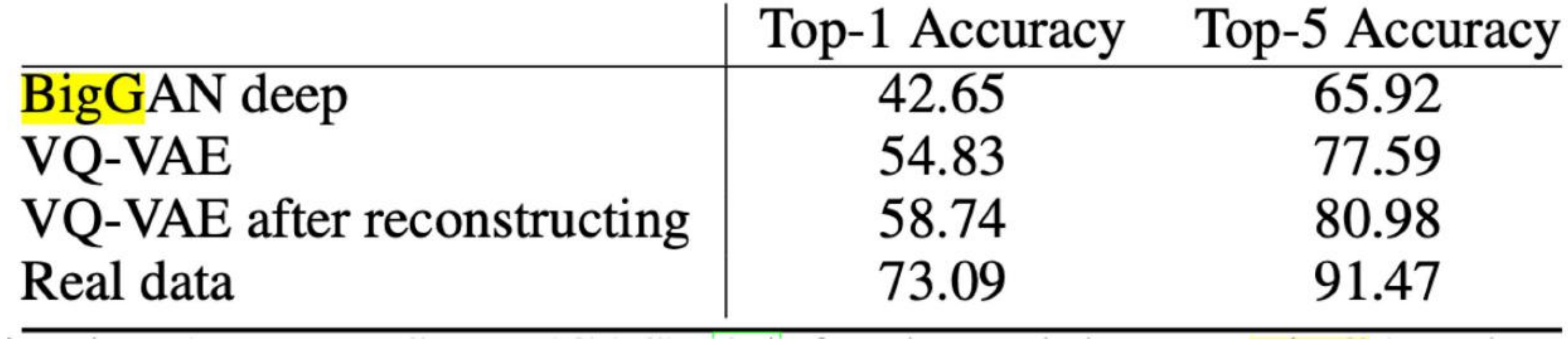
- Once the model is trained we can sample images s and compute the “symbolic image” $z[X,Y]$.
- Given samples of symbolic images $z[X,Y]$ we can learn an auto-regressive model of these symbolic images using a pixel- CNN.
- This yields a prior probability distribution $P_\Phi(z[X,Y])$ which provides a tighter upper bound on the rate.
- We can then measure compression and distortion for test images. This is something GANs cannot do.

Multi-Layer Vector Quantized VAEs



Quantitative Evaluation

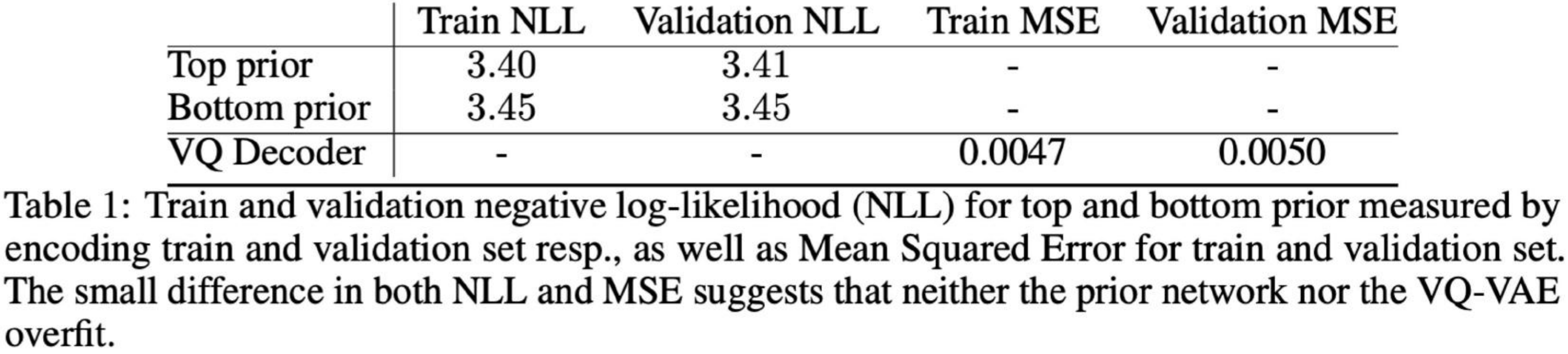
- The VQ-VAE2 paper reports a classification accuracy score (CAS) for class-conditional image generation.
- We generate image-class pairs from the generative model trained on the ImageNet training data.
- We then train an image classifier from the generated pairs and measure its accuracy on the ImageNet test set.



	Top-1 Accuracy	Top-5 Accuracy
BigGAN deep	42.65	65.92
VQ-VAE	54.83	77.59
VQ-VAE after reconstructing	58.74	80.98
Real data	73.09	91.47

Direct Rate-Distortion Evaluation

- Rate-distortion metrics for image compression to discrete representations support unambiguous rate-distortion evaluation.
- Rate-distortion metrics also allow one to explore the rate-distortion trade-off.



	Train NLL	Validation NLL	Train MSE	Validation MSE
Top prior	3.40	3.41	-	-
Bottom prior	3.45	3.45	-	-
VQ Decoder	-	-	0.0047	0.0050

Table 1: Train and validation negative log-likelihood (NLL) for top and bottom prior measured by encoding train and validation set resp., as well as Mean Squared Error for train and validation set. The small difference in both NLL and MSE suggests that neither the prior network nor the VQ-VAE overfit.

Image Compression

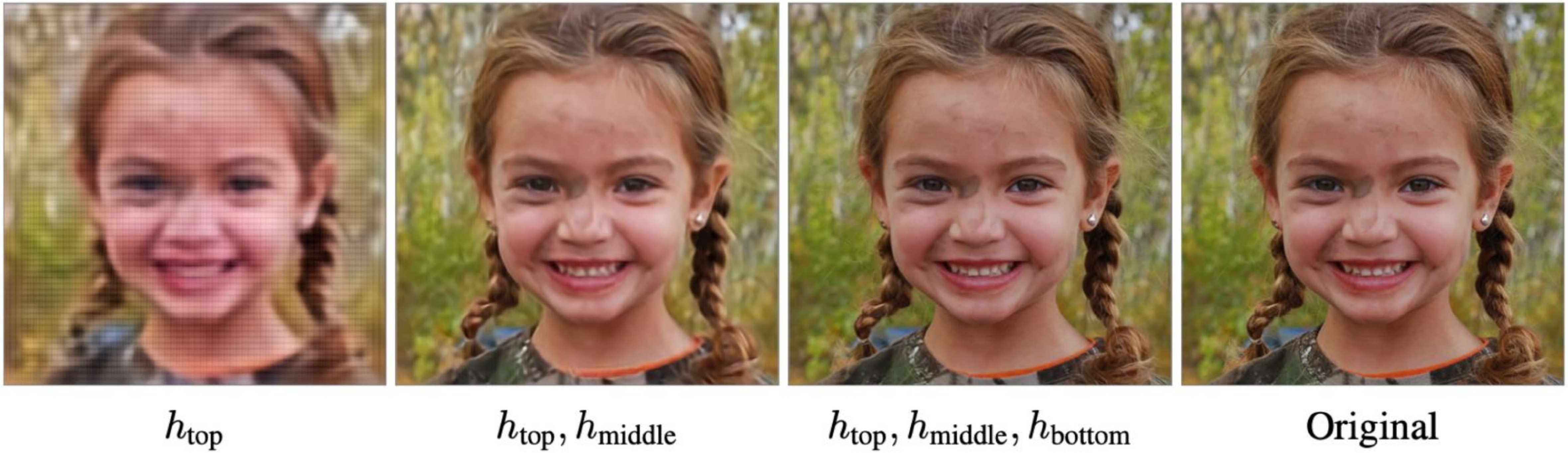

h_{top}
$h_{\text{top}}, h_{\text{middle}}$
$h_{\text{top}}, h_{\text{middle}}, h_{\text{bottom}}$
Original

Figure 3: Reconstructions from a hierarchical VQ-VAE with three latent maps (top, middle, bottom). The rightmost image is the original. Each latent map adds extra detail to the reconstruction. These latent maps are approximately 3072x, 768x, 192x times smaller than the original image (respectively).

Vector Quantization (Emergent Symbols)

- Vector quantization represents a distribution (or density) on vectors with a discrete set of embedded symbols.
- Vector quantization optimizes a rate-distortion tradeoff for vector compression.
- The VQ-VAE uses vector quantization to construct a discrete representation of images and hence a measurable image compression rate-distortion trade-off.

Symbols: A Better Learning Bias

- Do the objects of reality fall into categories?

- If so, shouldn’t a learning architecture be designed to categorize?

- Whole image symbols would yield emergent whole image classification.

Symbols: Improved Interpretability

- Vector quantization shifts interpretation from linear threshold units to the emergent symbols.

- This seems related to the use of t-SNE as a tool in interpretation.

Symbols: Unifying Vision and Language

- Modern language models use word vectors.

- Word vectors are embedded symbols.

- Vector quantization also results in models based on embedded symbols.

Symbols: Addressing the “Forgetting” Problem

- When we learn to ski we do not forget how to ride a bicycle.
- However, when a model is trained on a first task, retraining on a second tasks degrades performance on the first (the model “forgets”).
- But embedded symbols can be task specific.
- The embedding of a task-specific symbol will not change when training on a different task.

Symbols: Improved Transfer Learning

- Embedded symbols can be domain specific.

- Separating domain-general parameters from domain-specific parameters may improve transfer between domains.

Next lecture:
Self-Supervised Learning