CS167: Machine Learning

Generative Modeling:
Variational Autoencoder (VAE)

November, 11th, 2024

Drake
UNIVERSITY

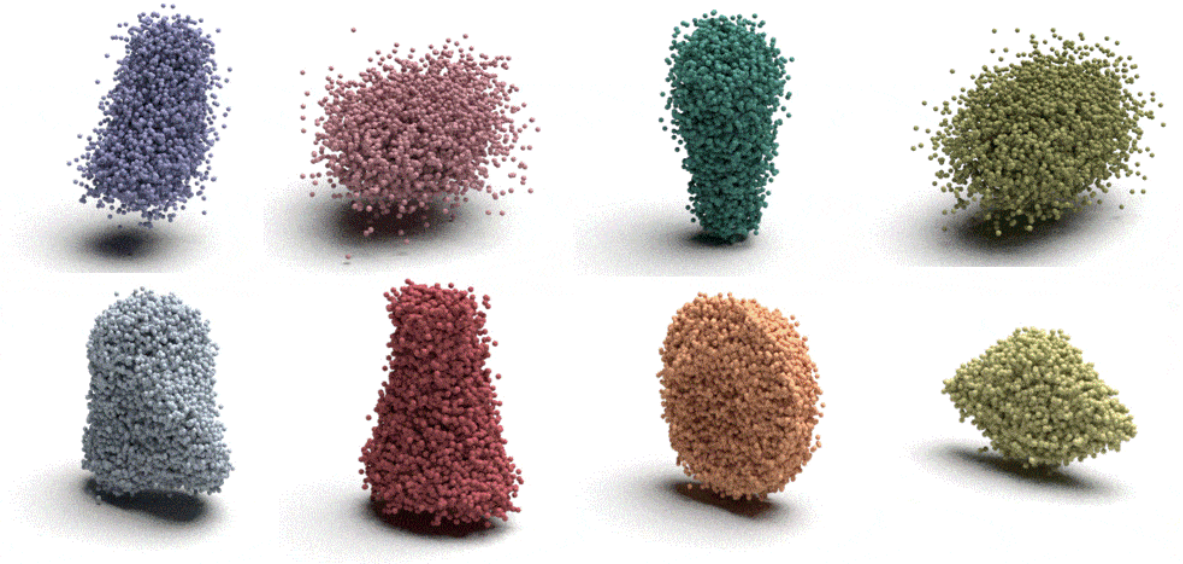
Today's Topics

- **Generative Modeling**
- Latent Space
- Autoencoder
- Variational Autoencoder (VAE)

Generative Model

- Generative modeling have achieved state-of-the-art performance on many downstream tasks and applications

- Shape generation: generate shapes from 3D point-cloud



Learning Gradient Fields for Shape Generation - ECCV'20

Generative Model

- Generative modeling can be used on many downstream tasks and applications

- Audio synthesis: generate audio that sounds realistic

DiffWave: A Versatile Diffusion Model for Audio Synthesis - ICML'21

- Music generation: generate new music

Symbolic Music Generation with Diffusion Models

Types of Generative Models

- Generative modeling is based on representation of probability distribution
- Existing generative modeling techniques can be classified into three broad categories
  - Implicit generative models (eg, GAN)
  - Likelihood-based models (eg, VAE)
  - Score-based models (eg, Diffusion process)

Generative Model: Likelihood-Based Model

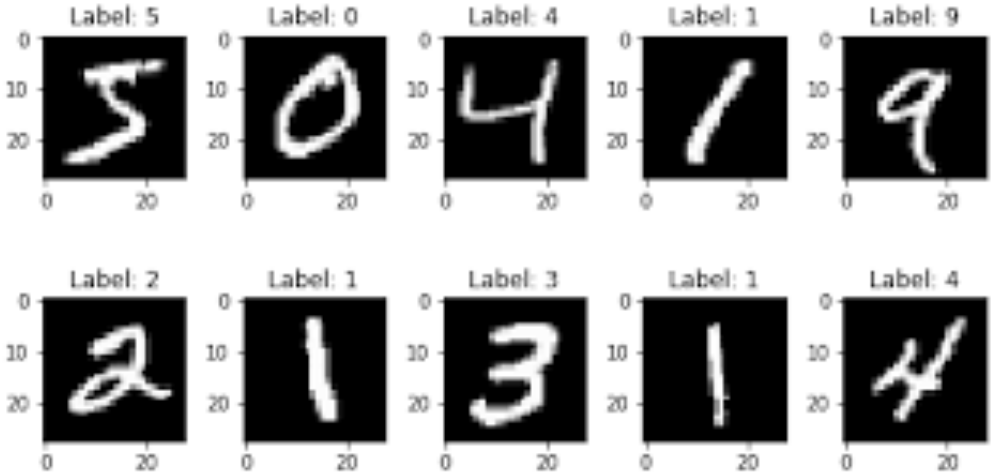
- Generative modeling is based on representation of probability distribution
- Likelihood-based models:
  - directly learns the distribution's probability density function (PDF) via maximum likelihood estimation method
  - Variational Autoencoder (VAE) is an example of likelihood-based generative model
  - Limitations
    - rely on surrogate objectives to approximate maximum likelihood training
    - require strong restriction on the model architecture for tractable normalization

Today's Topics

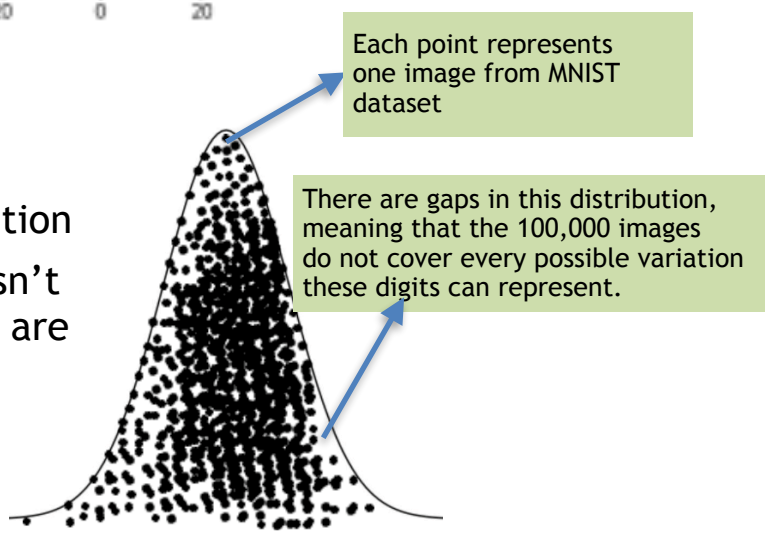
- Generative Modeling
- **Latent Space**
- Autoencoder
- Variational Autoencoder (VAE)

Dataset Distribution vs. True Distribution

- Let's consider the MNIST dataset, which is a collection of 28x28 pixel images. Each image represents one of 10 possible digits as follows:

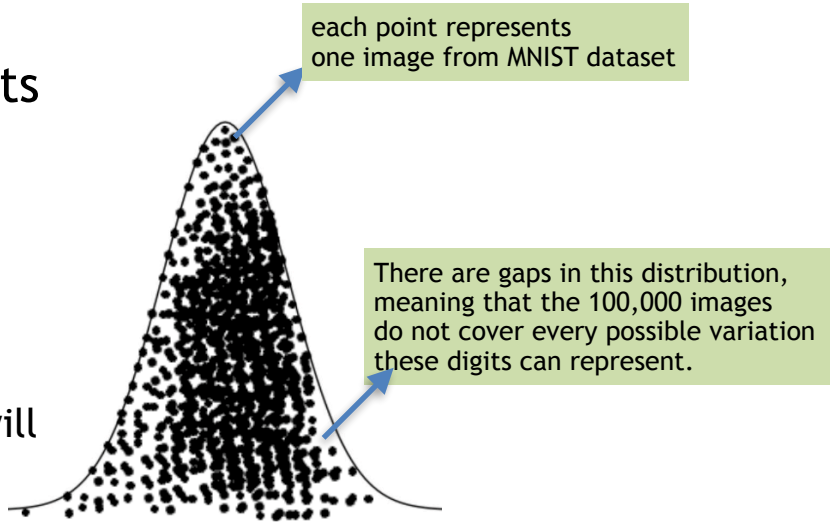


- MNIST dataset:
  - 100k images sampled from the true data distribution
  - This subset follows the true distribution but doesn't capture all possible variations. Some data points are missing, creating gaps in the distribution



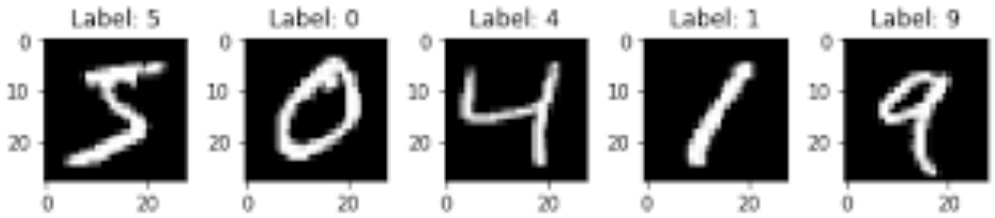
Dataset Distribution vs. True Distribution

- What can say about the missing data points
  - We don't know the nature of the missing data points
- Consequence
  - Model trained on MNIST with perfect accuracy will make mistake if tested on missing data points



Data Distribution: Latent Space

- Let's consider the MNIST dataset, which is a collection of 28x28 pixel images. Each image represents one of 10 possible digits

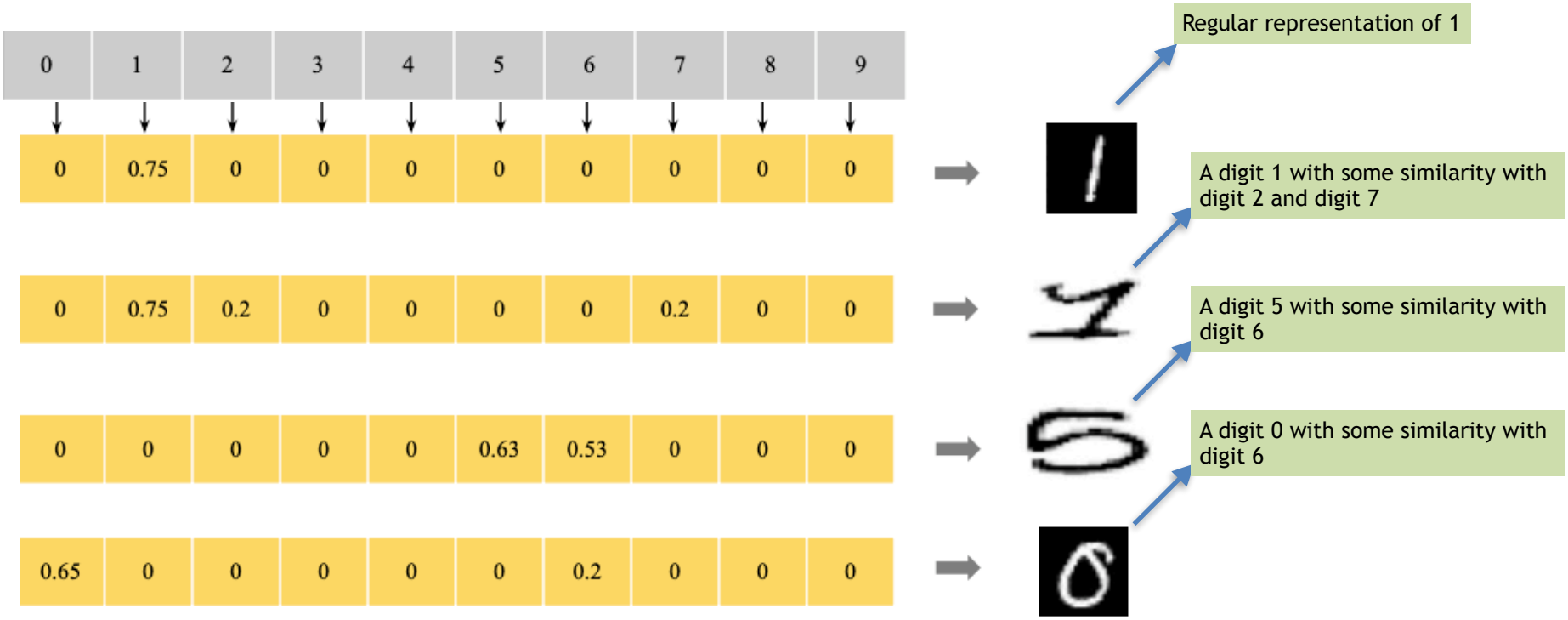


- Images are represented in a 28*28 = 768 dimensional pixel space
  - Most of the pixels are not useful (eg, a lot of black region in each image with no information)
  - Hence, most of the dimensions are redundant

- Such images can be represented in a much smaller dimensions
  - significantly less than 768 dimensional pixel space
  - Can you think of any 10 dimensional or 15 dimensional feature space?

10 Dimensional Latent Space

- 10 dimensional latent representation:
  - 1 dimension to represent each digit
  - It could contain a floating point number signifying the variations



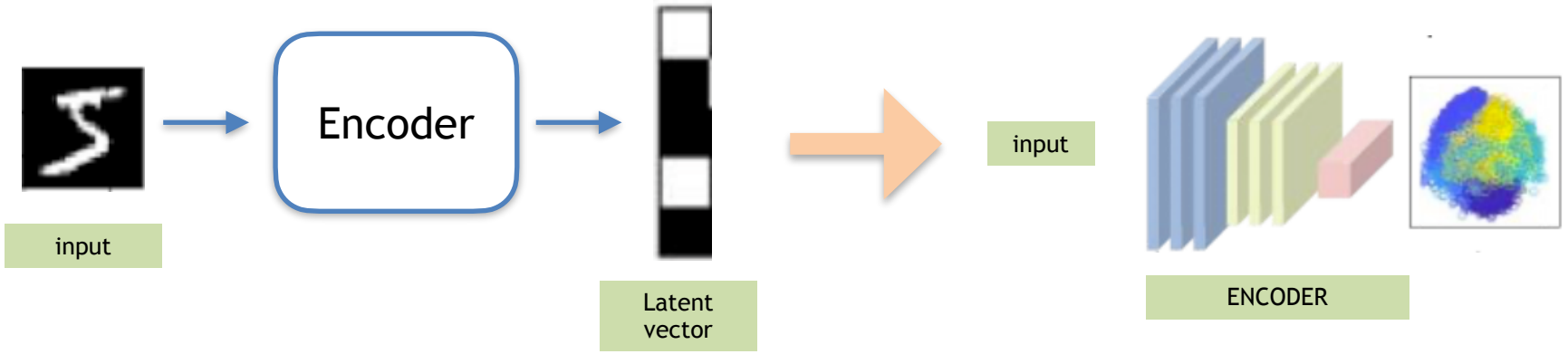
- This 10 dimensional latent representation is known as latent vector (LV)
- The size of the latent vector is 10, which is a hyper-parameter
- How can you generate this?

Today's Topics

- Generative Modeling
- Latent Space
- **Autoencoder**
- Variational Autoencoder (VAE)

Encoder for Learning Latent Representation

- A neural network (based on either MLP or CNN) to generate the latent vector



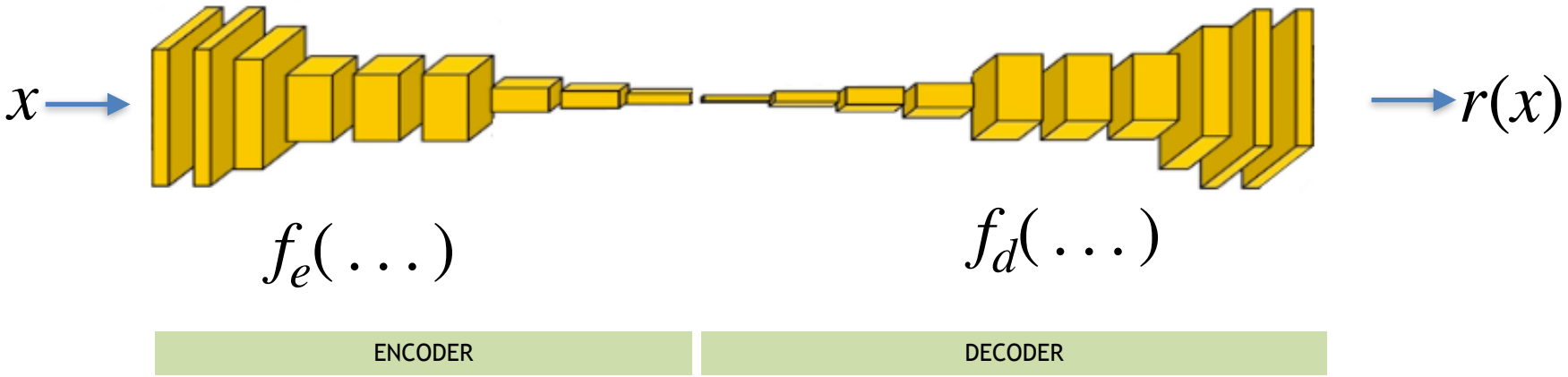
- Why do need compress the image?
  - Most of the dimensions are redundant
  - Efficient representation if we can compress it down to 10, 20, 100
  - Useful for followup task such as classification, decoding, etc

- These compressed latent representation can be learned with the help of another decoder sub-network jointly. Collectively, this network can be called Autoencoder.

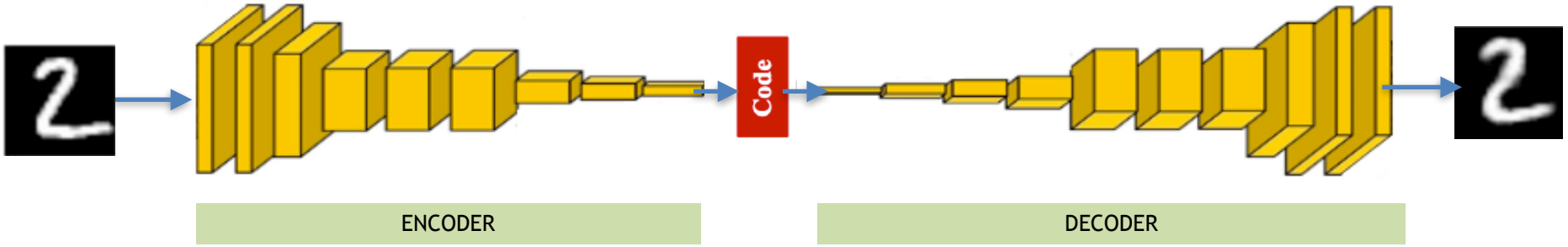
Learning Latent Representation via Autoencoder

- An Autoencoder is a type of neural network consisting of an Encoder and a Decoder that jointly learns the non-linear mapping between input ***x*** and output ***r(x)***

$$r(x) = f_d(f_e(x))$$



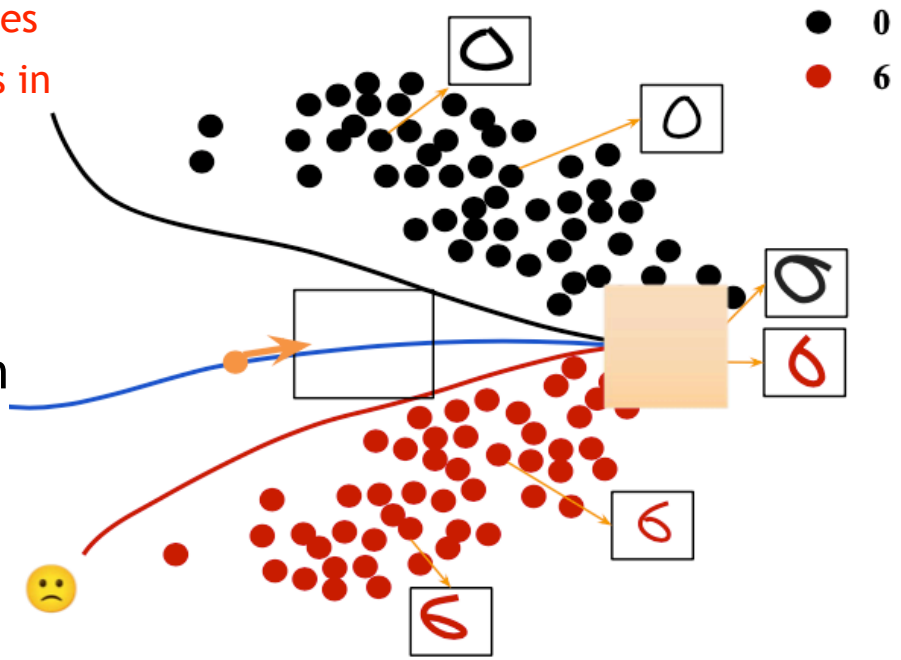
Vanilla Autoencoder



- Autoencoder input image is coded and decoded in one go
- After training similar examples share similar latent vector or code
- Since the network weights are fixed; after training if we input the same image we will always get the same latent vector
  - Can not generate images using unseen code
  - Latent space is discrete and not continuous

Vanilla Autoencoder

- Autoencoder do not have control over latent vectors (LV) or code
  - Only generate latent variable for seen images
  - Autoencoder is trained using discrete points in the latent space



- What will the model generate for a sample in the latent space in between two classes (eg, 0 and 6)?

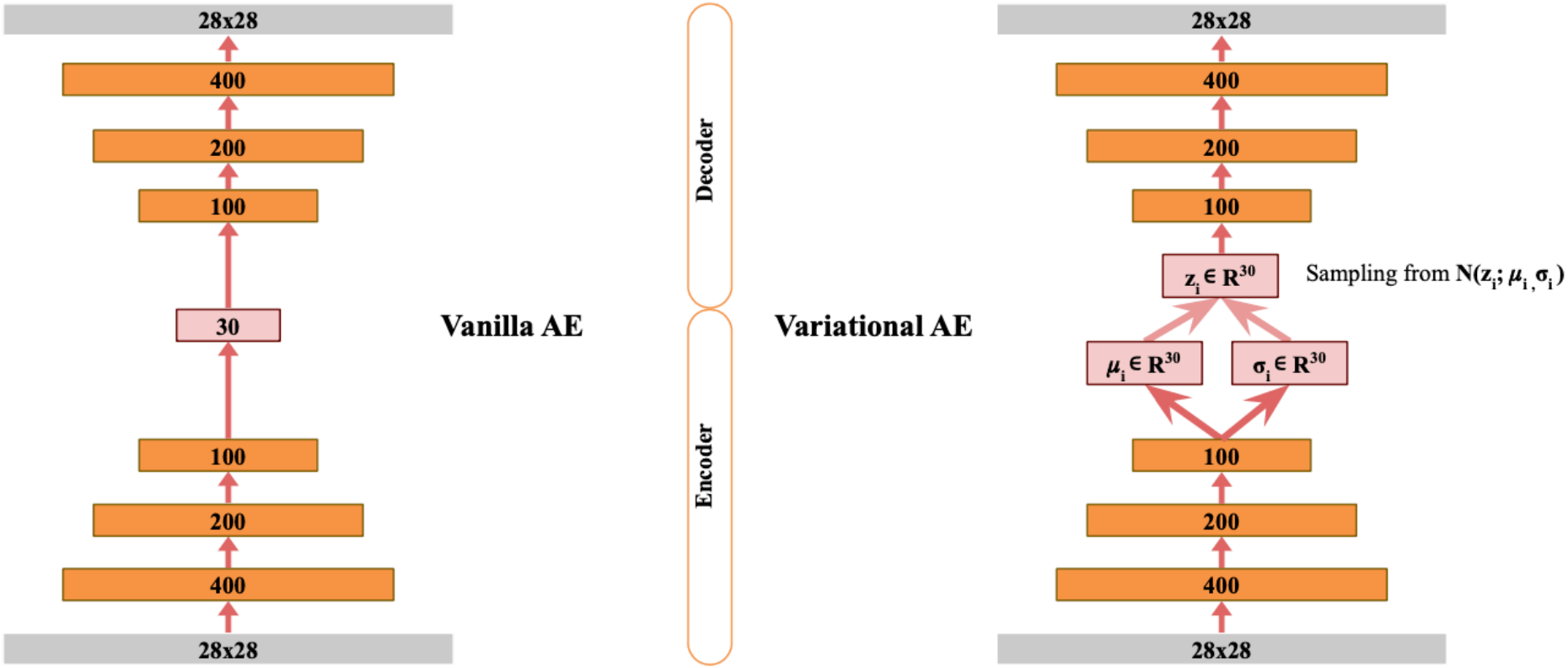
- Need to make generation process smooth or continuous in the latent space
- Variational Autoencoder (VAE) can achieve this goal.

Today's Topics

- Generative Modeling
- Latent Space
- Autoencoder
- **Variational Autoencoder (VAE)**

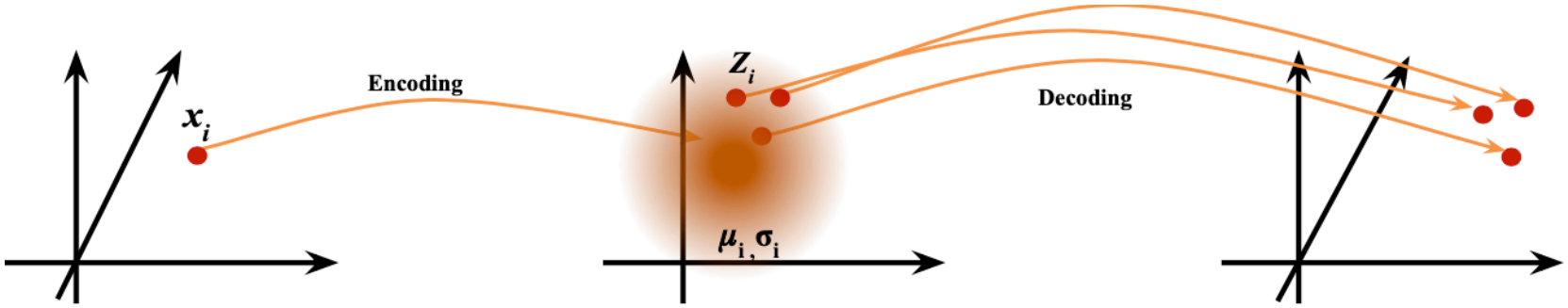
Variational Autoencoder (VAE)

- Variational Autoencoder (VAE) try to sample from a continuous distribution in the latent space



Variational Autoencoder (VAE)

Sampling enables to learn from multiple samples of the latent distribution



- From one example, we got a distribution with all nearby variations → making latent space continuous

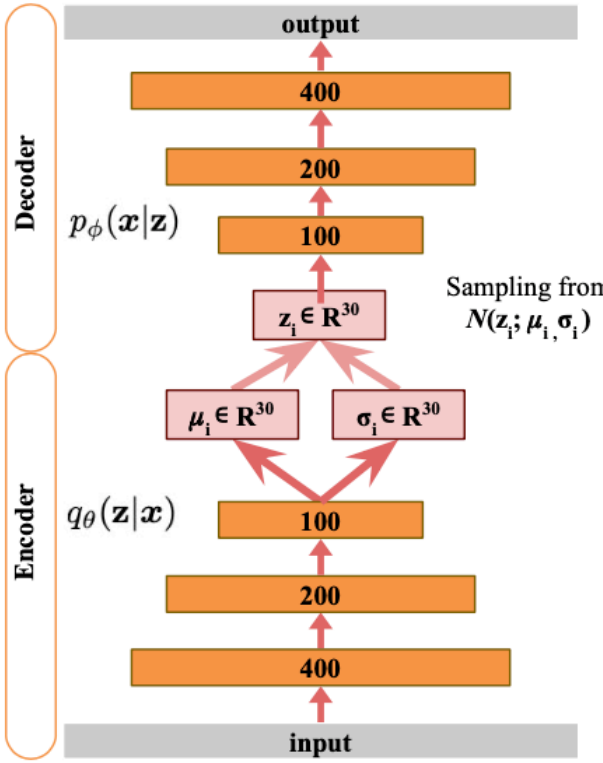
- During training
  - The reconstructed output is from a sampled version of the latent variable
  - The training algorithm is exposed to the continuous distribution of the latent space distribution (which was not possible with Vanilla Autoencoder)

- During testing
  - it produces stochastic predictions (randomly generate new output) due to the sampling process

Training Loss Function of VAE

- Encoder
  - posterior distribution over latent variable given the data samples
  - parameterized by the weights of the network

- Decoder
  - distribution over the multivariate observations (pixels) given the latent variable
  - parameterized by the weights of the network

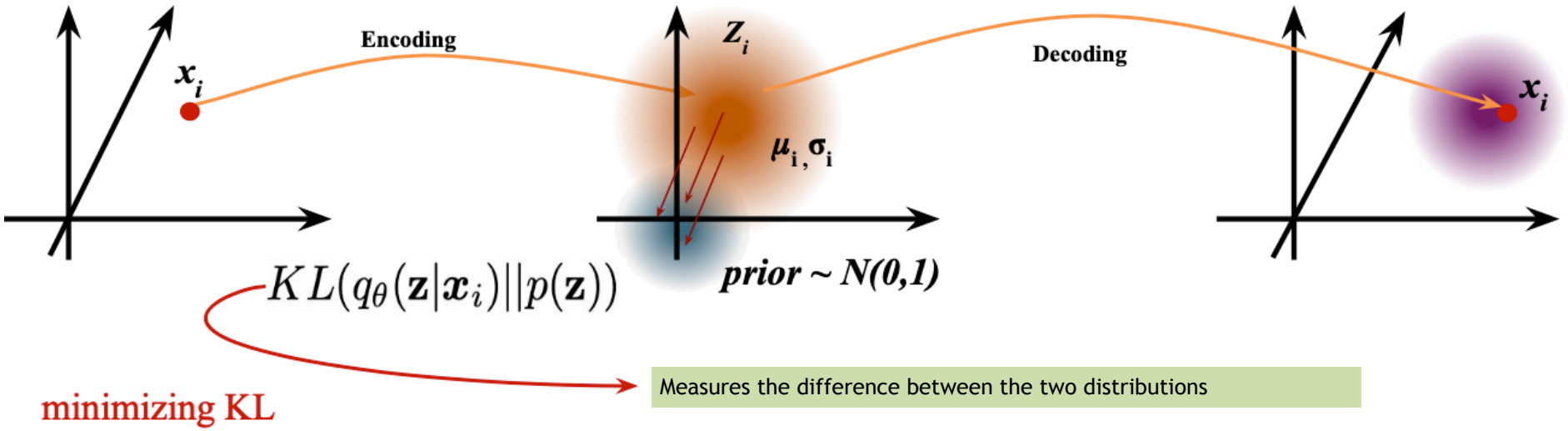


- Loss function is an expectation function over latent variable

$$\mathcal{L}_i(\theta, \phi) = -\mathbb{E}_{\mathbf{z} \sim q_\theta(\mathbf{z}|\boldsymbol{x}_i)}[\log p_\phi(\boldsymbol{x}_i|\mathbf{z})] + KL(q_\theta(\mathbf{z}|\boldsymbol{x}_i)||p(\mathbf{z}))$$

- Two terms inside the expectation function
  - Likelihood term: likelihood of observing the input given the latent variable
  - Regularization term: the latent variable should be close to a prior Gaussian distribution

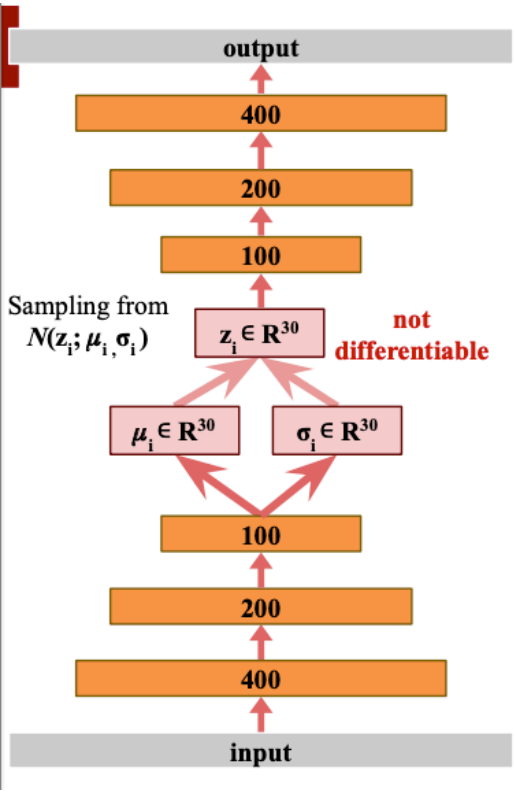
Role of the Prior in VAE

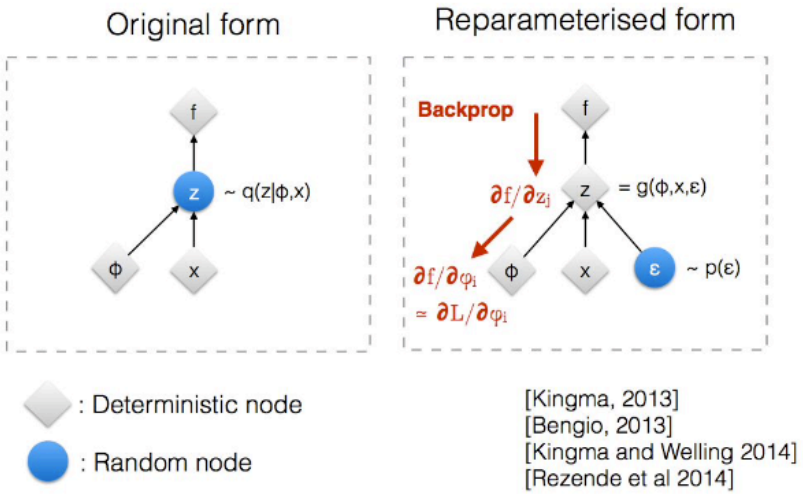


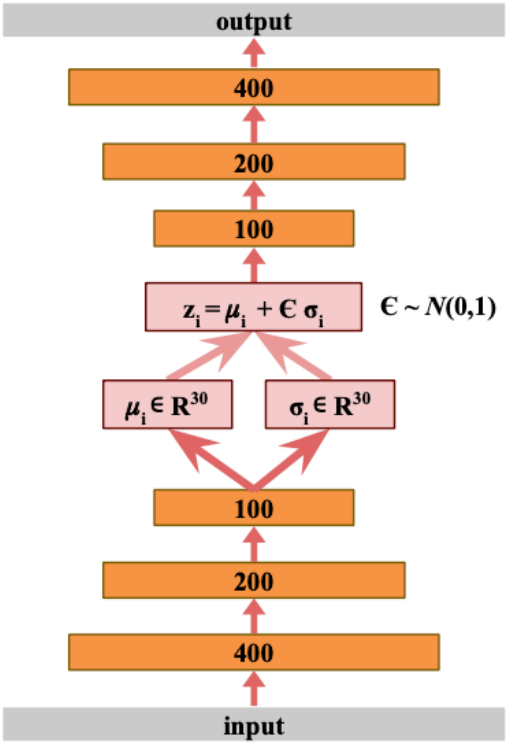
- We want latent variable distribution to look like a standard normal distribution
  - it will provide control over the latent space distribution

- Apply a KL-divergence loss function b/w LV distribution and standard normal prior
  - at each step, the prior will nudge the latent variable distribution towards standard normal

Reparameterization Trick for Backpropagation







- After reparameterization, Z_i is differentiable w.r.t. both its parameters
- Why multiplying with a normal distribution?
  - adds randomness around the mean value of Z_i for continuous variations

VAE Image Generation

- Images with continuous variations are generated with VAE
  - Some digits looks like perfect fusion of two digits
  - Many digits do not make any sense but latent space is continuous