Representation Learning
Pre-training

Contrastive learning

Idea: if features are “semantically” relevant, a “distortion” of an image should produce similar features.

Framework:

- For every training sample, produce multiple *augmented* samples by applying various transformations.
- Train an encoder ***E*** to predict whether two samples are augmentations of the same base sample.
- A common way is train $\langle E(x), E(x') \rangle$ big if x, x' are two augmentations of the same sample:

$$\ell_{x,x'} = -\log\left(\frac{\exp(\tau\langle E(x), E(x')\rangle)}{\sum_{\tilde{x}} \exp(\tau\langle E(x), E(\tilde{x})\rangle)}\right)$$

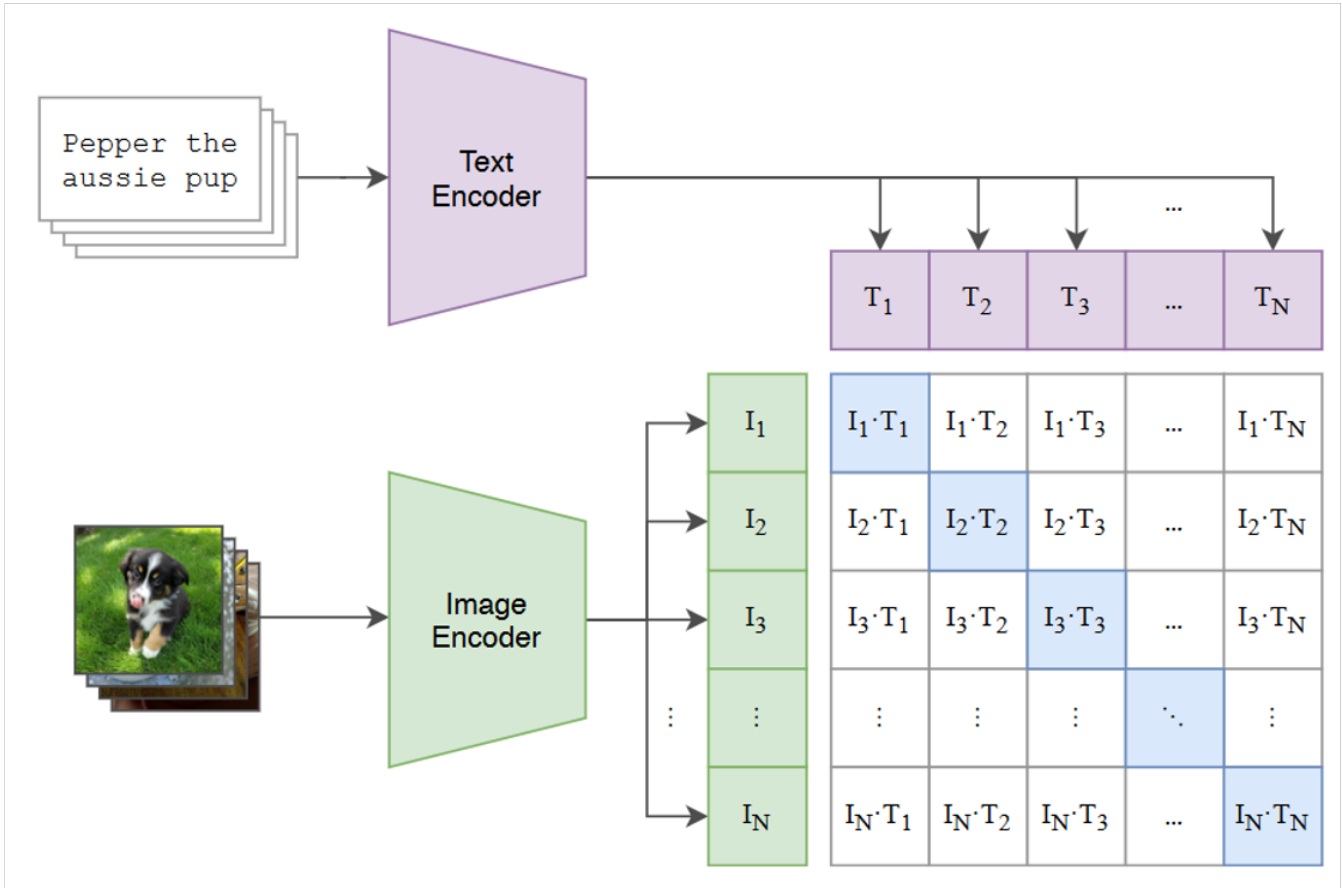
$$\min \sum_{x,x' \text{ augments of each other}} \ell_{x,x'}$$

Multimodal Contrastive Learning

(image, text)

$\{(image_i, text_i)\}_{i=1}^{N}$

Contrastive Pretraining:
Train image and text representation together



Multimodal Contrastive Learning

Loss function

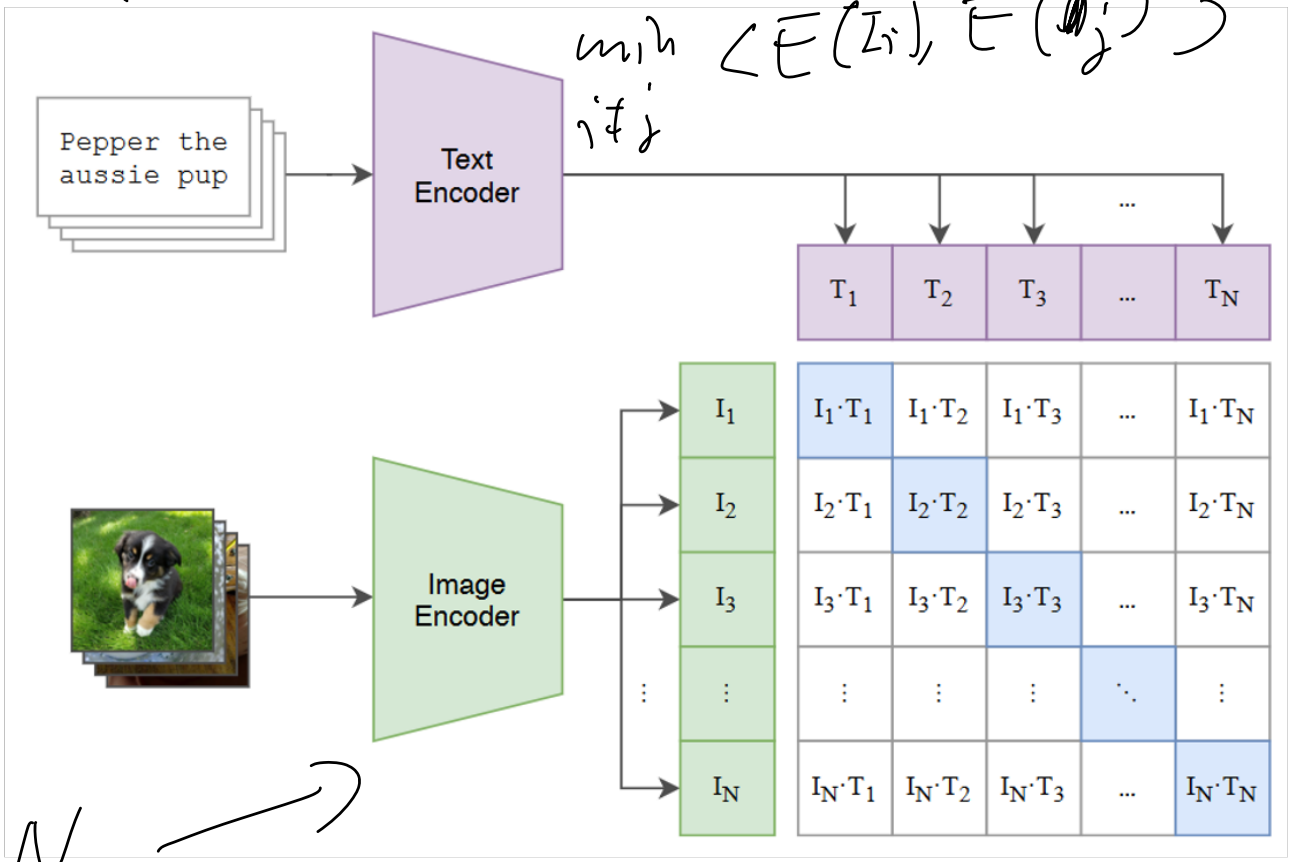
Let $q_{ij} := I_i^\top T_j$ (normalized embeddings: $||I_i||_2 = ||T_j||_2 = 1$),

$$\text{loss} = \frac{\text{loss}_I + \text{loss}_T}{2}$$

where,

$$\text{loss}_I = -\sum_{i=1}^{N} \log \frac{\exp(q_{ii})}{\sum_j \exp(q_{ij})}$$

$$\text{loss}_T = -\sum_{j=1}^{N} \log \frac{\exp(q_{jj})}{\sum_i \exp(q_{ij})}$$

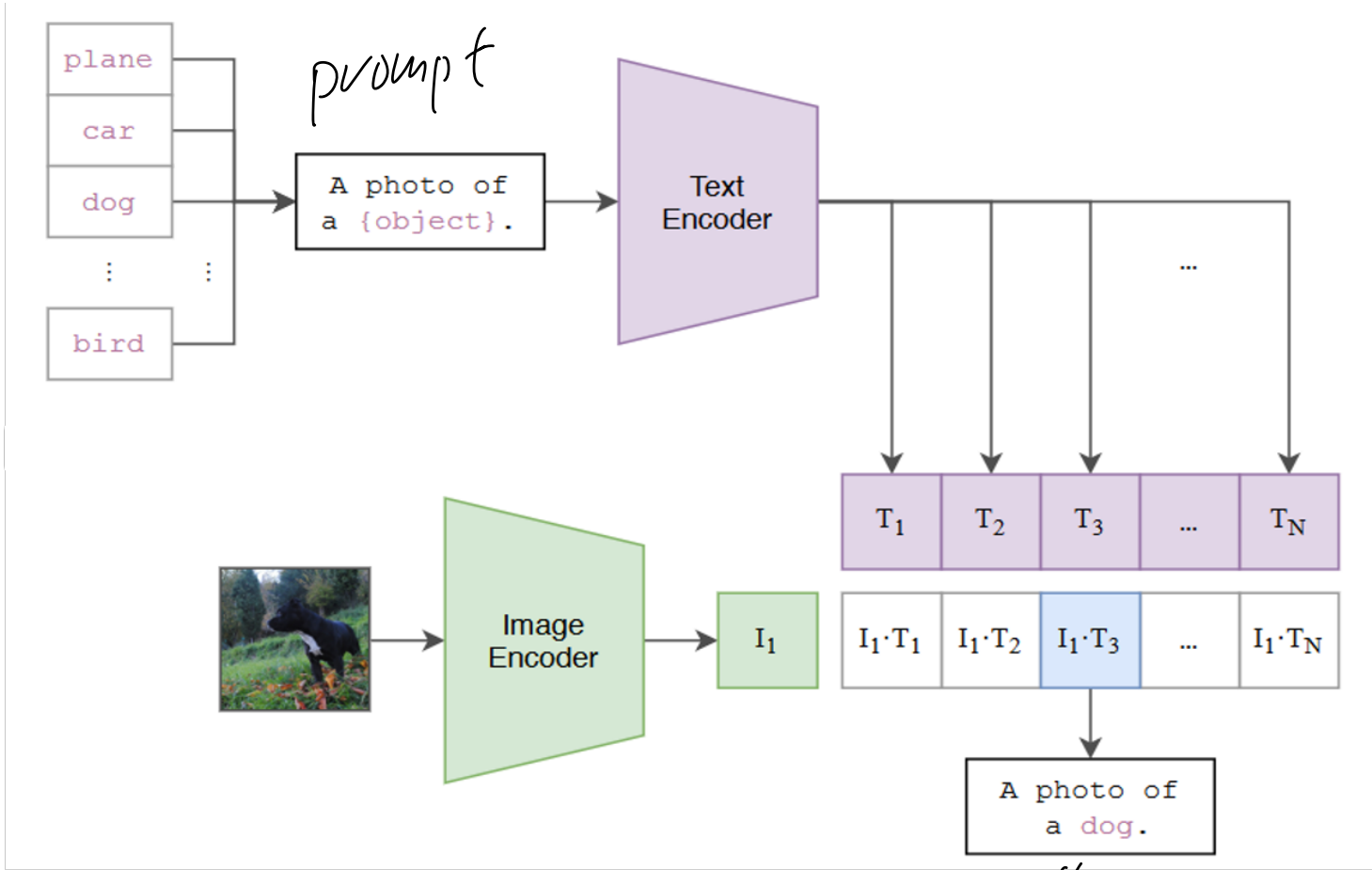


Multimodal Contrastive Learning

CLIP

Zero-Shot Classification:

- Generate a prompt for each class



/K classes

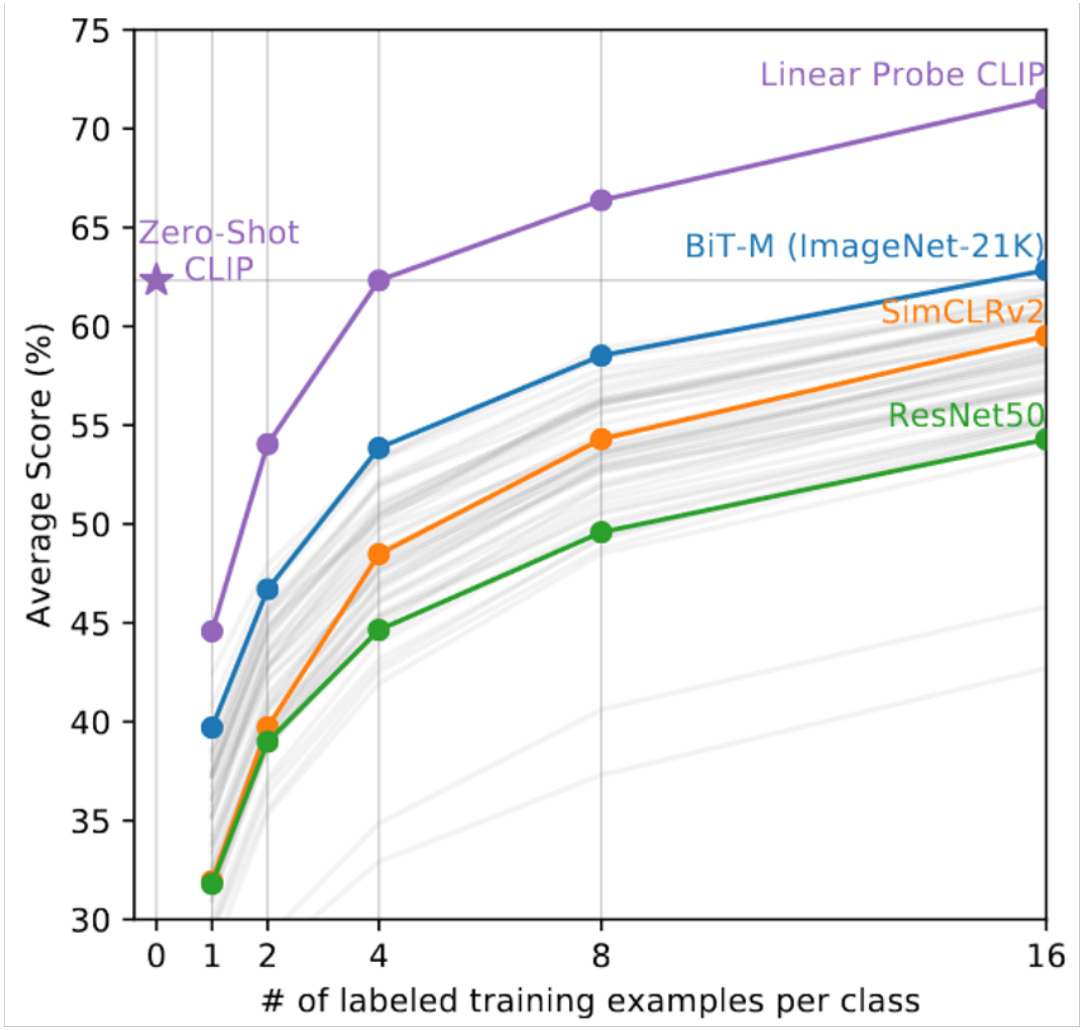
prompt

Classification

Multimodal Contrastive Learning

Results

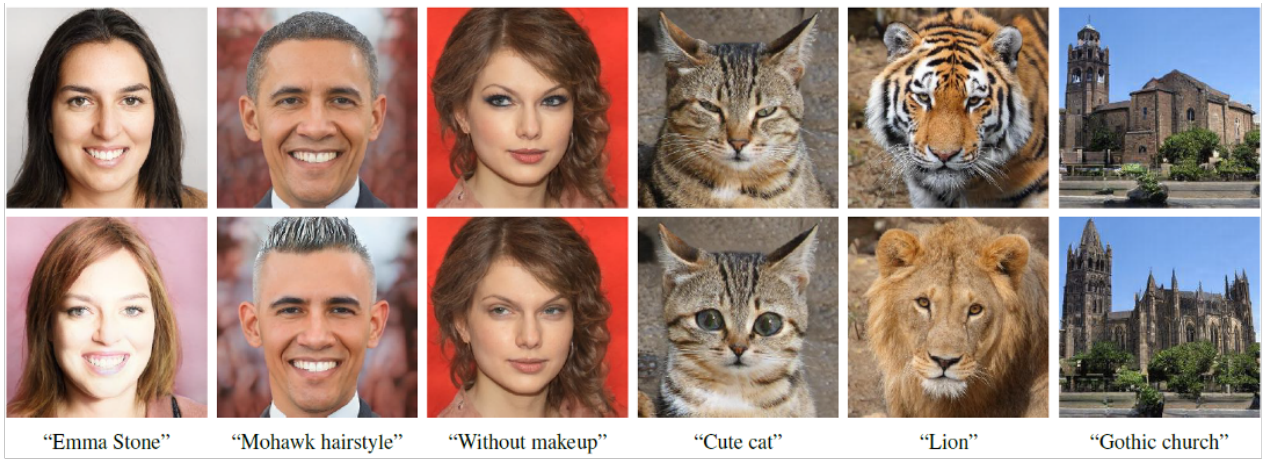
- Strong zero-shot and few-shot performance compared with other models.
- Zero-shot performance on **ImageNet**: CLIP ≈ fully supervised ResNet50!



Applications of CLIP

Image Generation

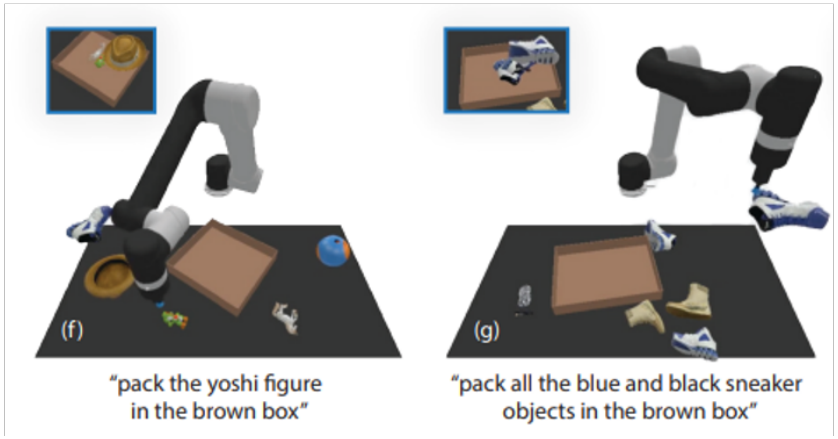
(StyleCLIP [Patashnik et al. 2021])



Robotics

(CLIPort [Shridhar et al. 2021])

...



Problems about Training CLIP

Require large amount of ***carefully curated*** image-text pairs
4 Billion closed-source data used for OpenAI's CLIP

Q: How to obtain lots of high-quality data?

One choice: Web-curated data pairs + data filtering

DataComp

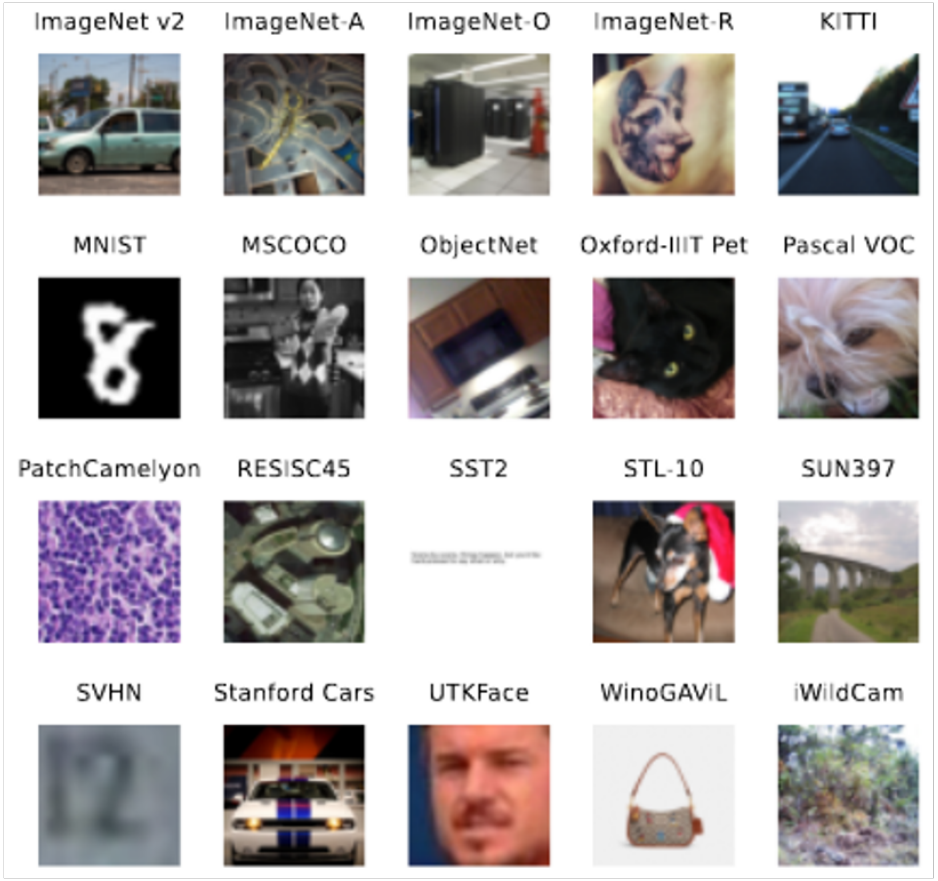
A benchmark standardize the training configuration

Training Process:

- Filtering data from a pool of ***low-quality*** data pairs
- Train a CLIP model with a **fixed architecture** and **hyperparameters**
- Fix **total number of training data seen** (1 pass of 4B data = 4 passes of 1B data)

Evaluation:

- 38 Zero-shot downstream tasks



Data Filtering

Distribution-agnostic methods

Image-based filtering

- Cluster the image embeddings (from a **pre-trained** CLIP model) of training data, and select the groups that contain at least one embedding from ImageNet-1k

CLIP score filtering

- Filter the data with low CLIP similarity assigned by a **pre-trained** CLIP model.

$$\text{CLIP score} = \bar{f}_{\text{image}}^{T} \bar{f}_{\text{text}}$$

Data Filtering

Setup:

Total number of training sample seen = 12.8M

Filtering Strategy	Dataset Size	ImageNet (1 sub-task)	ImageNet Dist. Shift (5)	VTAB (11)	Retrieval (3)	Average (38)
No filtering	12.8M	2.5	3.3	14.5	10.5	13.2
CLIP score (30%, reproduced)	3.8M	4.8	5.3	17.1	11.5	15.8
Image-based ∩ CLIP score (45%)	1.9M	4.2	4.6	17.4	10.8	15.5
$\mathbb{D}^2$ Pruning (image+text, reproduced)	3.8M	4.6	5.2	18.5	11.1	16.1
CLIP score (45%)	5.8M	4.5	5.1	17.9	**12.3**	16.1

Filtering significantly improves the performance!

auto regressive

Generative Models

Distribution learning

Training Data(CelebA)

Model Samples (Karras et.al., 2018)

4 years of progression on Faces

Brundage et al., 2017

Image credits to Andrej Risteski

Distribution learning

BigGAN, Brock et al '18

Distribution learning

Conditional generative model P(zebra images| horse images)

Style Transfer

Input Image

Monet

Van Gogh

Image credits to Andrej Risteski

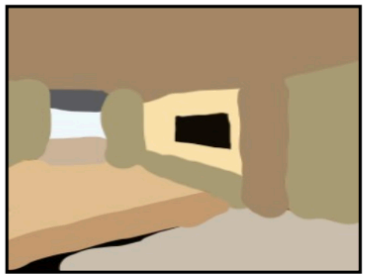
Distribution learning

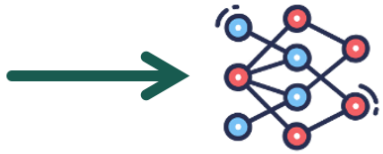
Source actor

Target actor

Real-time reenactment

Generative model





Generate

Generative model
of realistic images

Stroke paintings to realistic images

[Meng, He, Song, et al., ICLR 2022]

"Ace of Pentacles"

Generate

Generative model
of paintings

Language-guided artwork creation

https://chainbreakers.kath.io @RiversHaveWings

Slides credit to Yang Song

Generative model

High probability

Low probability

Generative model of traffic signs

Outlier detection

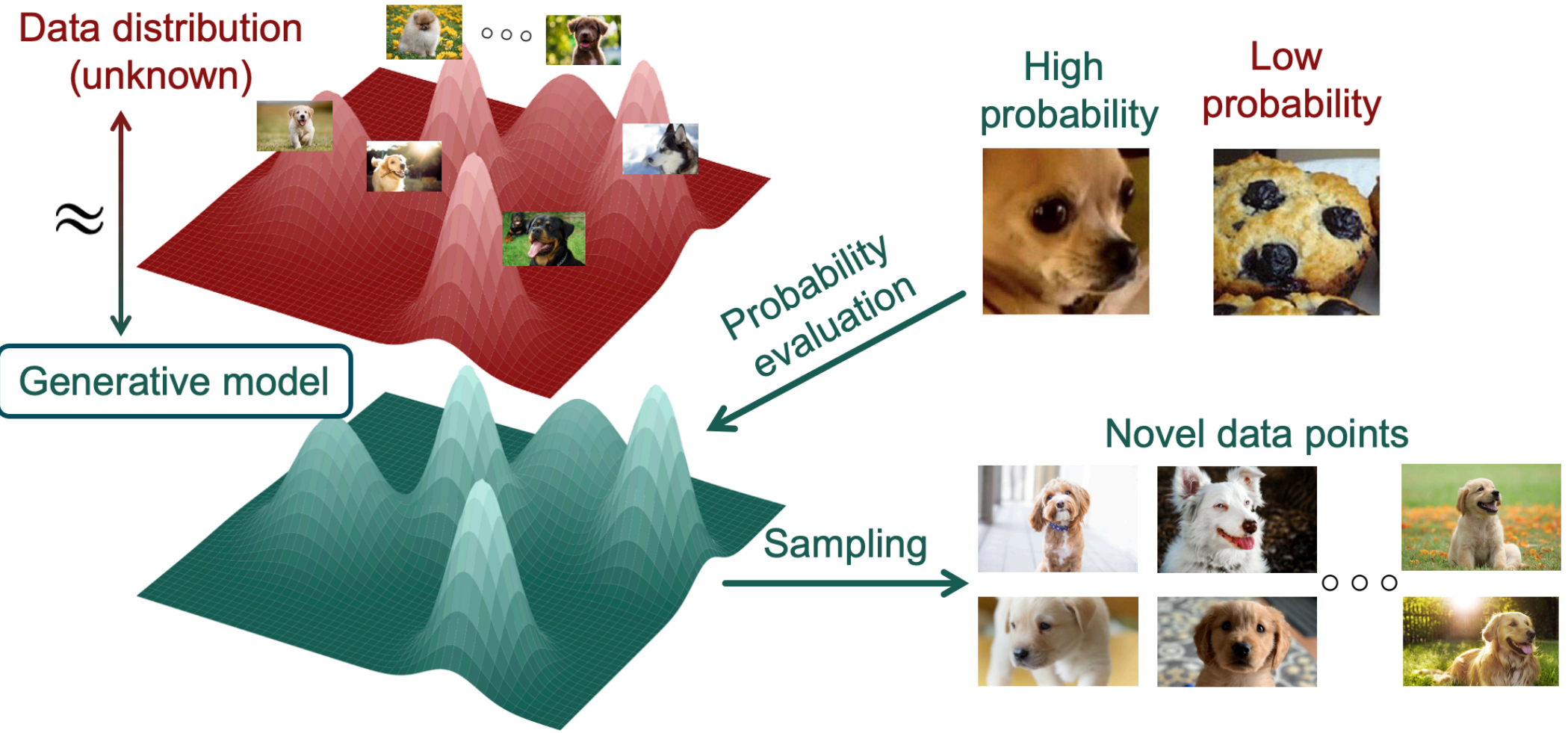
[Song et al., ICLR 2018]

Slide credit to Yang Song

Desiderata for generative models

- **Probability evaluation**: given a sample, it is computationally efficient to evaluate the probability of this sample.
- **Flexible model family:** it is easy to incorporate any neural network models.
- **Easy sampling:** it is computationally efficient to sample a data from the probabilistic model.

Desiderata for generative models



Slide credit to Yang Song

Taxonomy of generative models

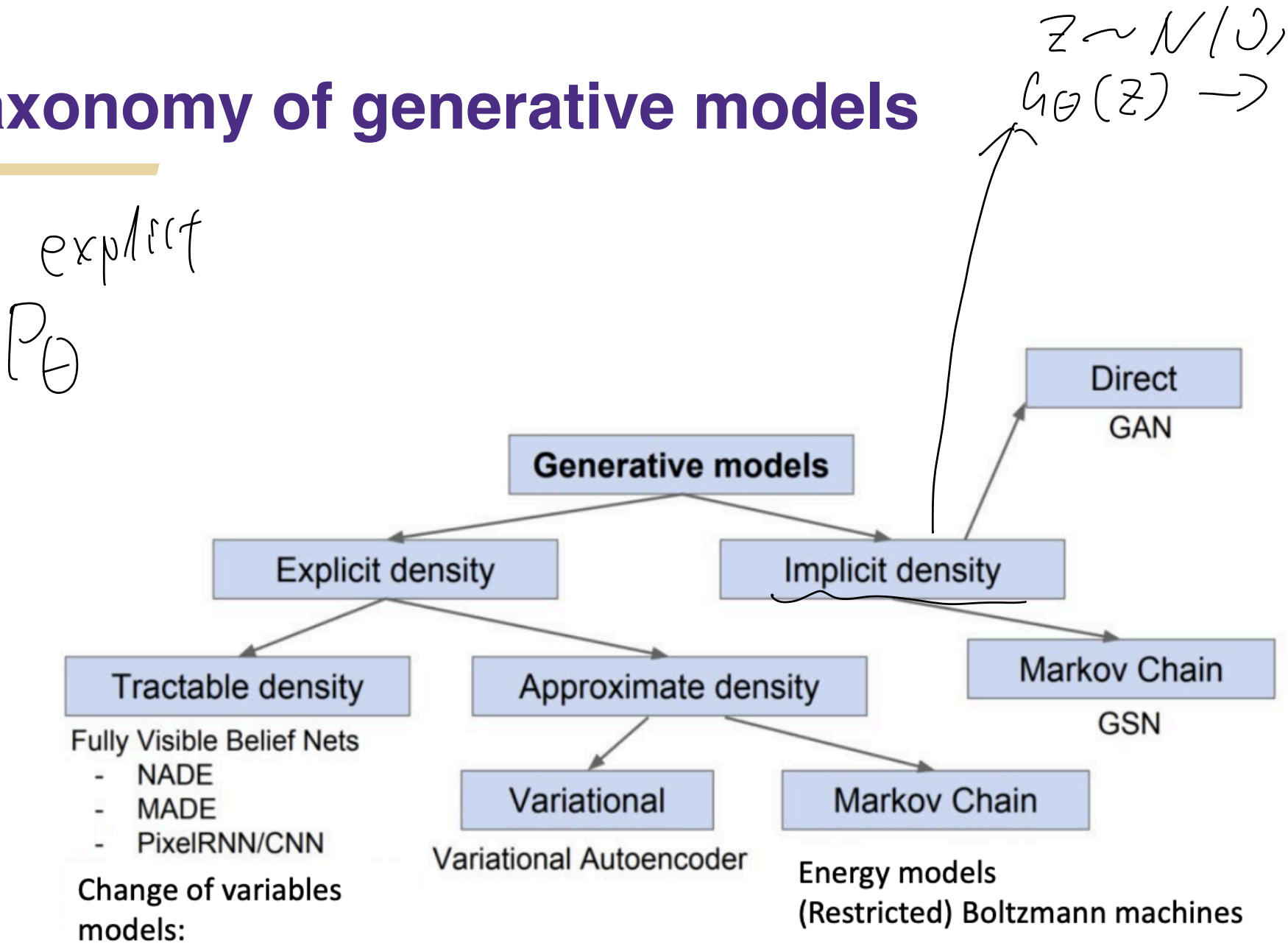
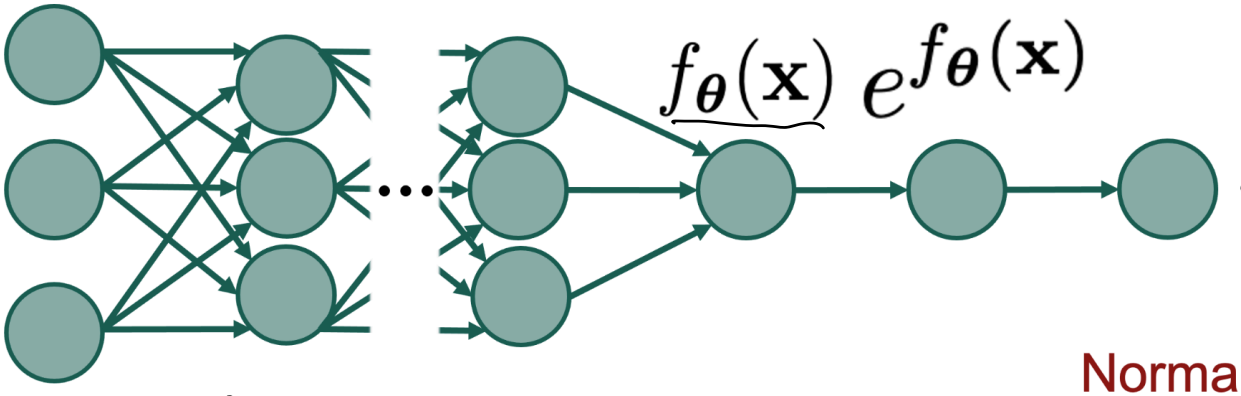


Image credits to Andrej Risteski

Key challenge for building generative models

distribution $\begin{cases} p_\theta(x) \geq 0 \\ \int_x p_\theta(x)\, dx = 1 \end{cases}$

$\mathbf{x}$



auto-regressive

$Z_\theta = \sum_{i=1}^{N} e^{f_\theta(x)}$

N: # tokens

$Z_\theta = \int_x e^{f_\theta(x)}\, dx$

Slide credit to Yang Song

Key challenge for building generative models

Approximating the normalizing constant

- Variational auto-encoders [Kingma & Welling 2014, Rezende et al. 2014]
- Energy-based models [Ackley et al. 1985, LeCun et al. 2006]

✗ Inaccurate probability evaluation

Using restricted neural network models

- Autoregressive models [Bengio & Bengio 2000, van den Oord et al. 2016]
- Normalizing flow models [Dinh et al. 2014, Rezende & Mohamed 2015]

✗ Restricted model family

Generative adversarial networks (GANs)

- Model the generation process, not the probability distribution [Goodfellow et al. 2014]

✗ Cannot evaluate probabilities

Slide credit to Yang Song

Training generative models

- **Likelihood-based:** maximize the likelihood of the data under the model (possibly using advanced techniques such as variational method or MCMC):

$$\max_{\theta} \sum_{i=1}^{n} \log p_\theta(x_i)$$

- Pros:
  - **Easy training**: can just maximize via SGD.
  - **Evaluation**: evaluating the fit of the model can be done by evaluating the likelihood (on test data).

- Cons:
  - **Large models needed**: likelihood objectve is hard, to fit well need very big model.
  - **Likelihood entourages averaging:** produced samples tend to be blurrier, as likelihood encourages “coverage” of training data.

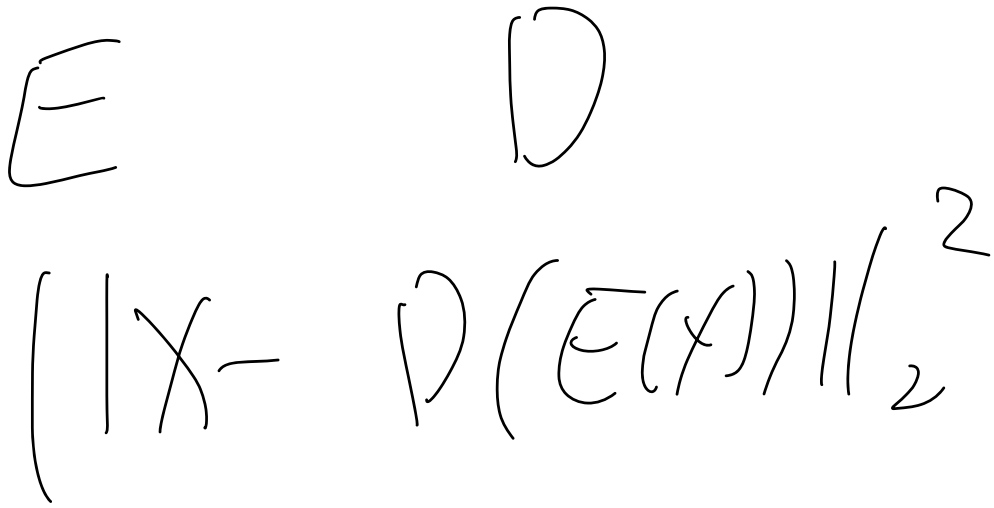
Training generative models

- **Likelihood-free:** use a **surrogate loss** (e.g., GAN) to train a discriminator to differentiate real and generated samples.

- Pros:
  - **Better objective, smaller models needed:** objective itself is learned - can result in visually better images with smaller models.

- Cons:
  - **Unstable training:** typically min-max (saddle point) problems.
  - **Evaluation:** no way to evaluate the quality of fit.

Variational Autoencoder



Architecture

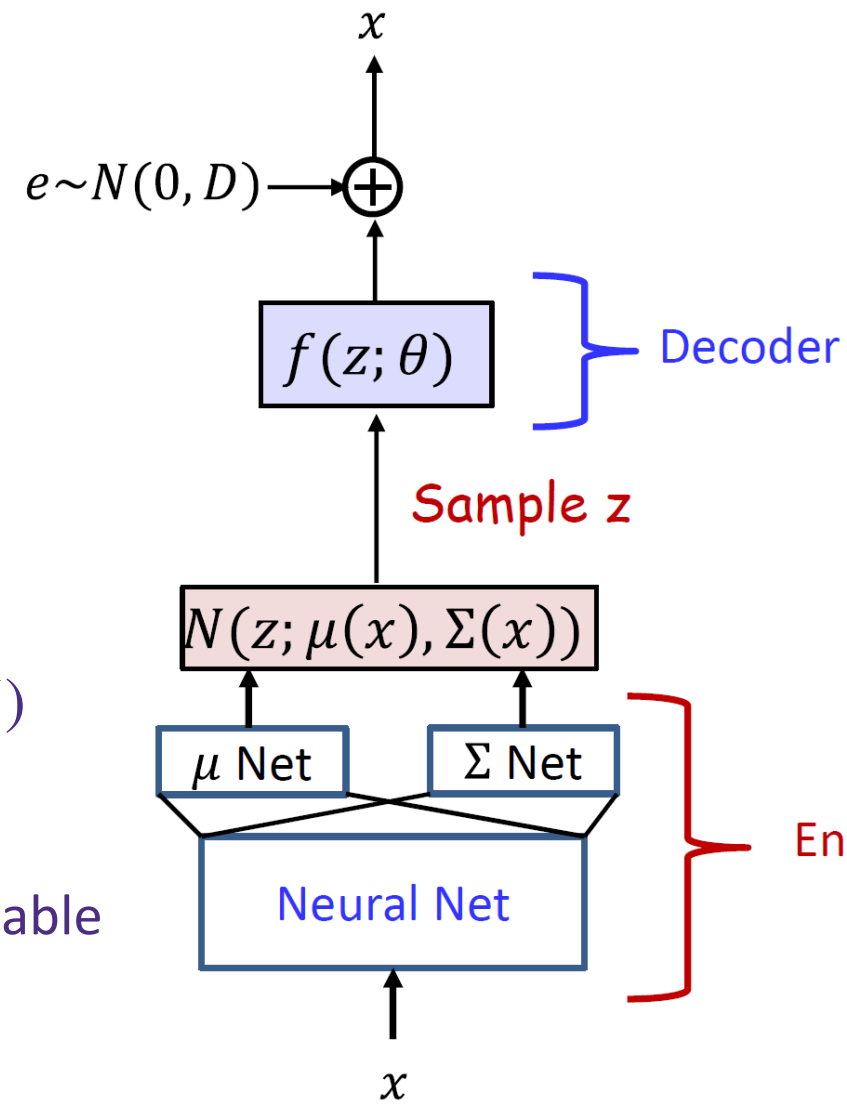
- Auto-encoder: $x \rightarrow z \rightarrow x$
- Encoder: $q(z|x;\phi) : x \rightarrow z$
- Decoder: $p(x|z;\theta) : z \rightarrow x$

- Isomorphic Gaussian:

$$q(z|x;\phi) = N(\mu(x;\phi), \text{diag}(\exp(\sigma(x;\phi))))$$

- Gaussian prior: $p(z) = N(0,I)$
- Gaussian likelihood: $p(x|z;\theta) \sim N(f(z;\theta), I)$

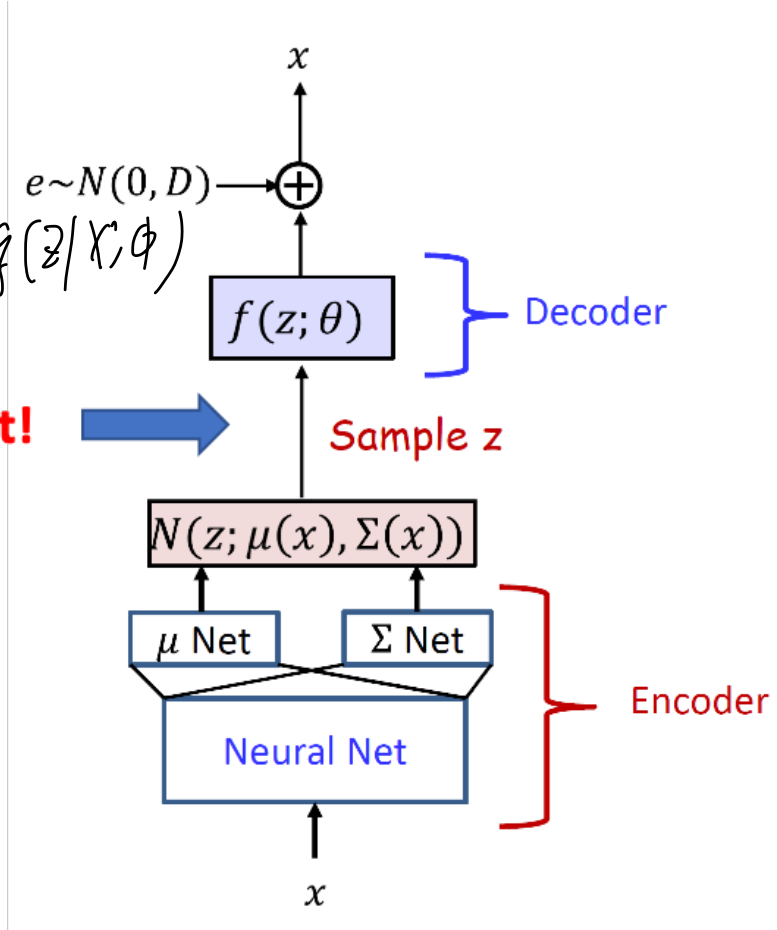
- Probabilistic model interpretation: latent variable model.



VAE Training

- Training via optimizing ELBO
  - $L(\phi, \theta; x) = \mathbb{E}_{z \sim q(z|x;\phi)}[\log p(z|x;\theta)] - KL\left(q(z|x;\phi)||p(z)\right)$
  - Likelihood term + KL penalty

- KL penalty for Gaussians has closed form.
- Likelihood term (reconstruction loss):
  - Monte-Carlo estimation
  - Draw samples from $q(z|x;\phi)$
  - Compute gradient of θ:
    - $x \sim N(f(z;\theta); I)$
    - $p(x) = \frac{1}{\sqrt{2\pi}} \exp(-\frac{1}{2}\|x - f(z;\theta)\|_2^2)$



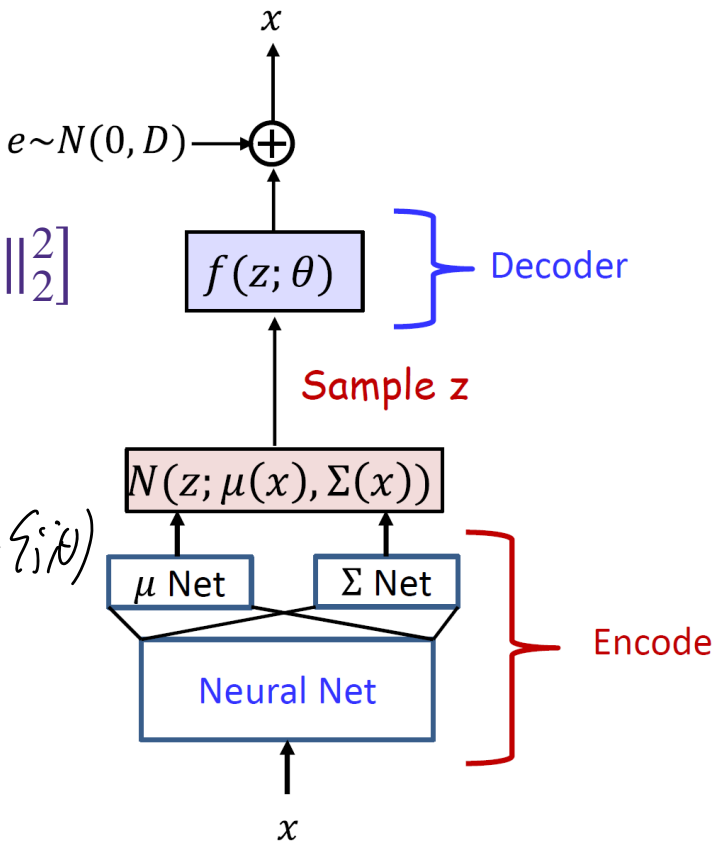
VAE Training

- Likelihood term (reconstruction loss):
  - Gradient for ϕ . Loss: $L(\phi) = \mathbb{E}_{z\sim q(z;\phi)}\left[\log p(x\,|\,z)\right]$
  - Reparameterization trick:
    - $z \sim N(\mu, \Sigma) \Leftrightarrow z = \mu + \epsilon, \epsilon \sim N(0,\Sigma)$
  - $L(\phi) \propto \mathbb{E}_{z\sim q(z|\phi)}\left[\|f(z;\theta) - x\|_2^2\right]$
    $\propto \mathbb{E}_{\epsilon\sim N(0,I)}\left[\|f(\mu(x;\phi) + \sigma(x;\phi)\cdot\epsilon;\theta) - x\|_2^2\right]$
  - Monte-Carlo estimate for $\nabla L(\phi)$

sample $\epsilon_1, \ldots, \epsilon_N \sim N$

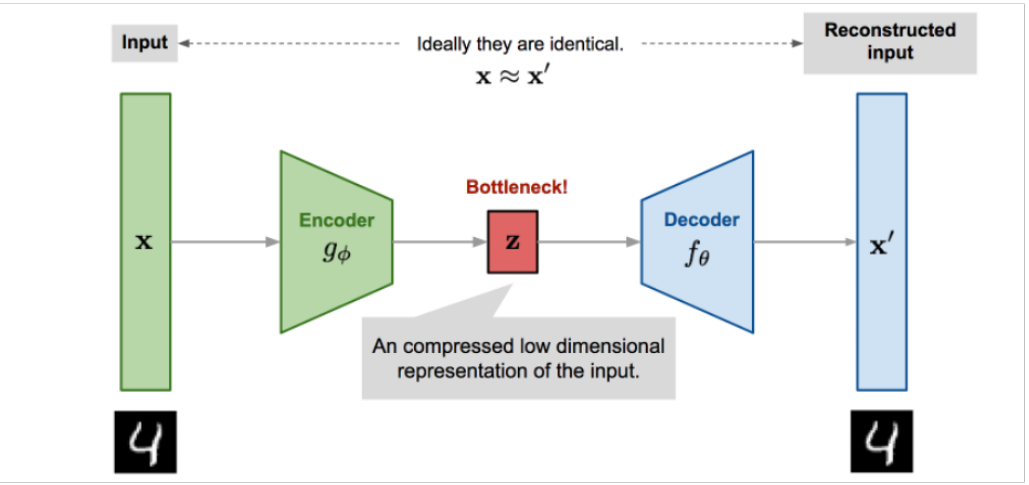
- End-to-end training

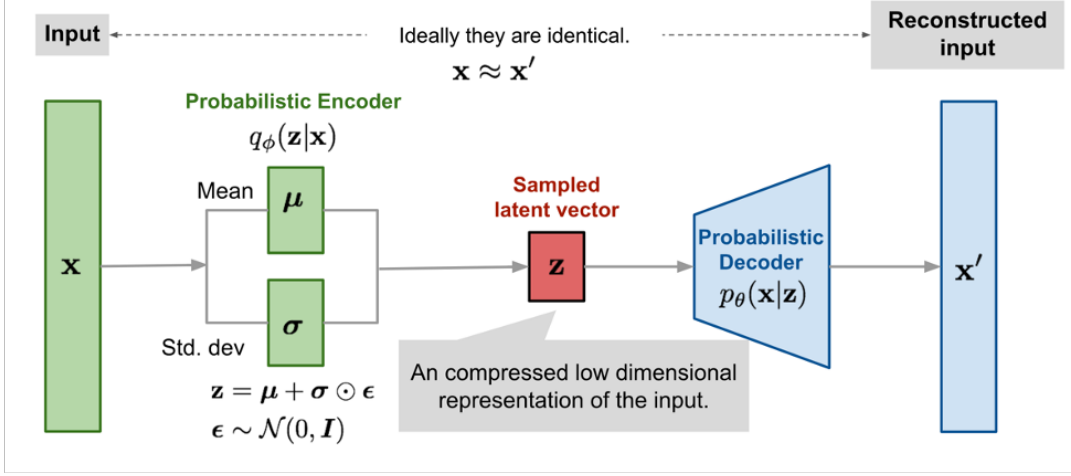
$\| f(\mu(x;\phi) + \sigma(x;\phi)\cdot\epsilon_i;\theta) - x \|_2^2$



VAE vs. AE

- AE: classical unsupervised representation learning method.
- VAE: a probabilistic model of AE
  - AE + Gaussian noise on z
  - KL penalty: L_2 constraint on the latent vector z





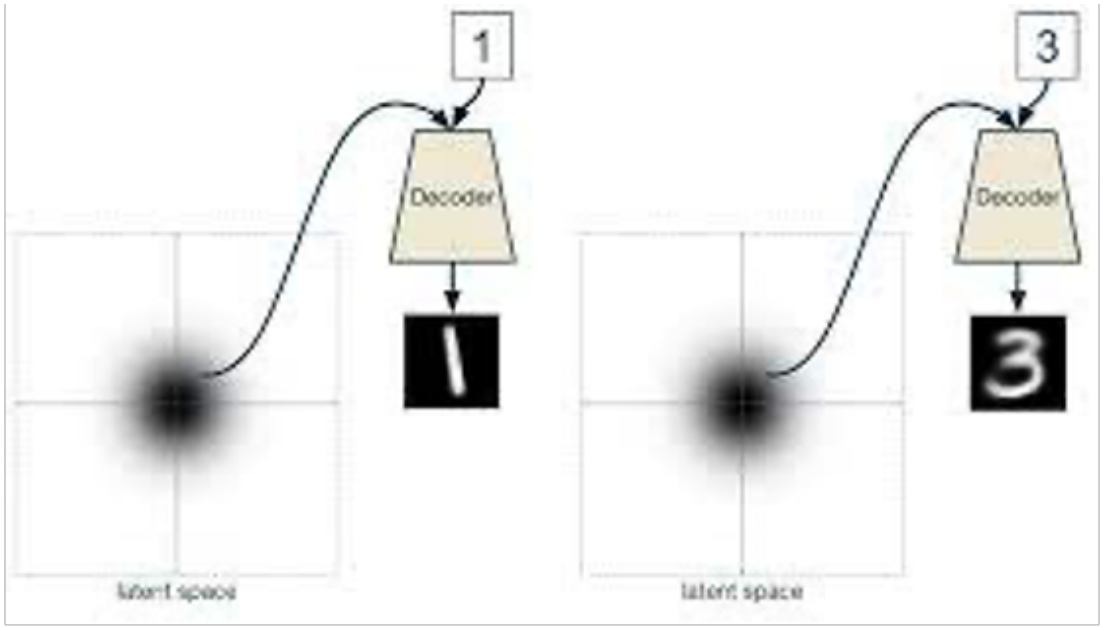
Conditioned VAE

$p(x|z;\theta;y)$

- Semi-supervised learning: some labels are also available

y: label

$p(x|z;\theta)$



conditioned generation

Comments on VAE

- Pros:
  - Flexible architecture
  - Stable training

- Cons:
  - Inaccurate probability evaluation (approximate inference)