CS 229br Lecture 3: Unsupervised / Self-Supervised Learning

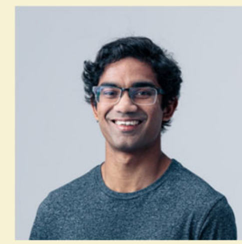
Boaz Barak

Ankur Moitra
MIT 18.408

Yamini Bansal
Official TF

Dimitris Kalimeris
Unofficial TF

Gal Kaplun
Unofficial TF



Preetum Nakkiran
Unofficial TF

#lectures | #qanda | #sys-help | #admin | #hw0 | #project | #papers

Unsupervised and semi-supervised learning

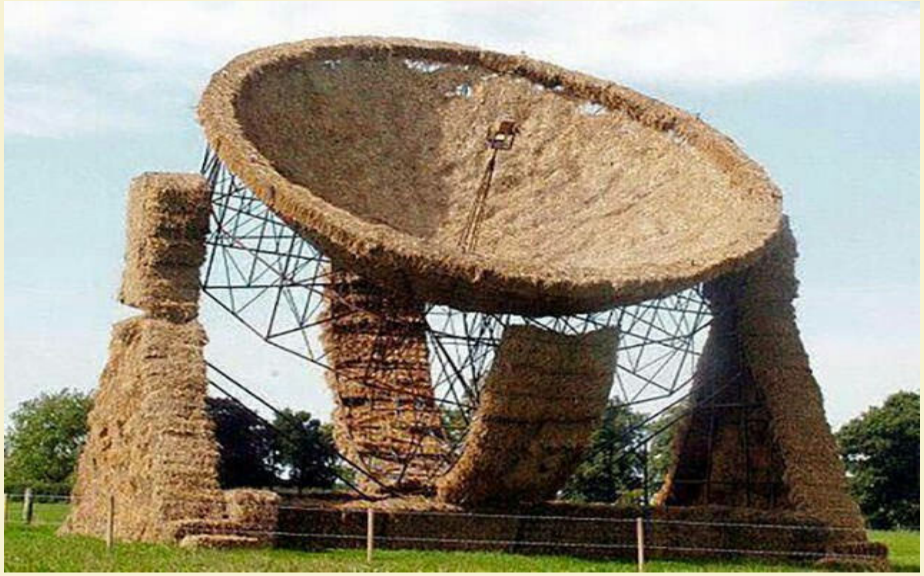
"No more y's!"

Input: $x_1, x_2, \dots, x_n \sim p \subseteq \mathbb{R}^d$

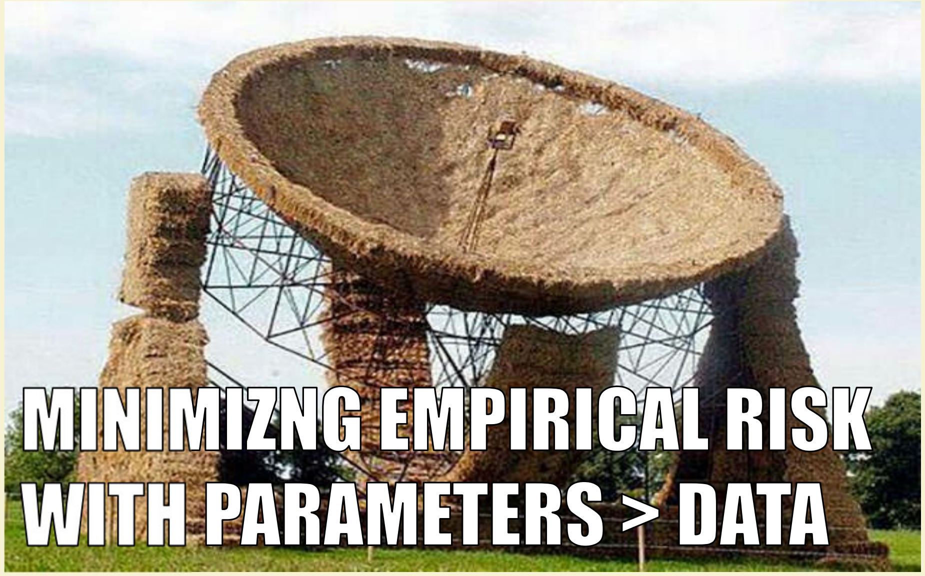
Goal: "understand" p

- Compute/approximate $x \mapsto p(x)$
- Sample fresh $x \sim p$
- Predict x_A from x_B
- Find "good" representation $r: \mathbb{R}^d \to \mathbb{R}^r$

Digressions



Is deep learning a cargo cult?



Two scenarios

Murphy's Law: *"Anything that can go wrong will go wrong"*

Marley's Law: *"Every little thing gonna be alright"*

Two technical digressions

1) Distance between distributions

2) Optimizing multiple objectives

Distances between probability distributions

p, q probability distributions over some domain D

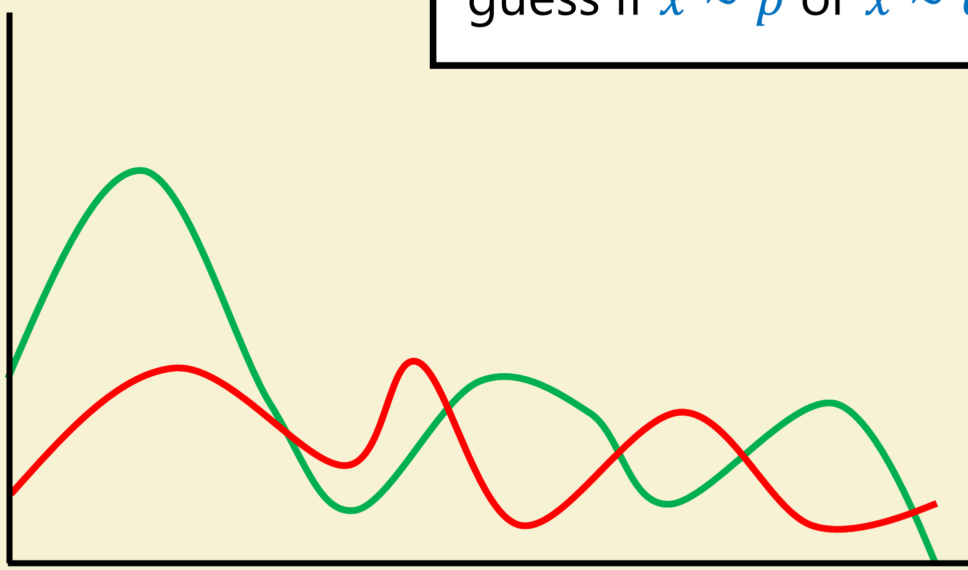
$$\Delta_{TV}(p,q) = \frac{1}{2}\sum_{x\in D}|p(x)-q(x)| = \max_{f:D\to\{0,1\}}\left|\mathbb{E}_p f - \mathbb{E}_q f\right|$$

Advantage over $\frac{1}{2}$ to guess if $x \sim p$ or $x \sim q$

$$\Delta_{KL}(p \parallel q) = \mathbb{E}_{x\sim p}\left[\log\frac{p(x)}{q(x)}\right] \geq 0$$

If $\Delta_{KL}(p \parallel q) = \delta$,
$\approx 1/\delta$ samples from p to rule out q

If $\Delta_{KL}(p \parallel q) = k$, k bits of “surprise”
$q \approx p$ after revealing k bits



Distances between probability distributions

p, q probability dis

$\Delta_{TV}(p, q) = \frac{1}{2}\sum_{x \in D} |p($

$\Delta_{KL}(p \parallel q) = \mathbb{E}_{x\sim p}\left[\log \frac{}{q(x)}\right] \geq 0$

Example:

- p dist over documents
- q dist over documents with topic y

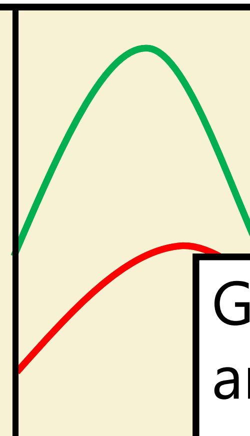
$$\Delta_{KL}(p \parallel q) \approx H(y)$$

to

$\sim q$

If $\Delta_{KL}(p \parallel q) = \delta$,
$\approx 1/\delta$ samples from p to rule out q

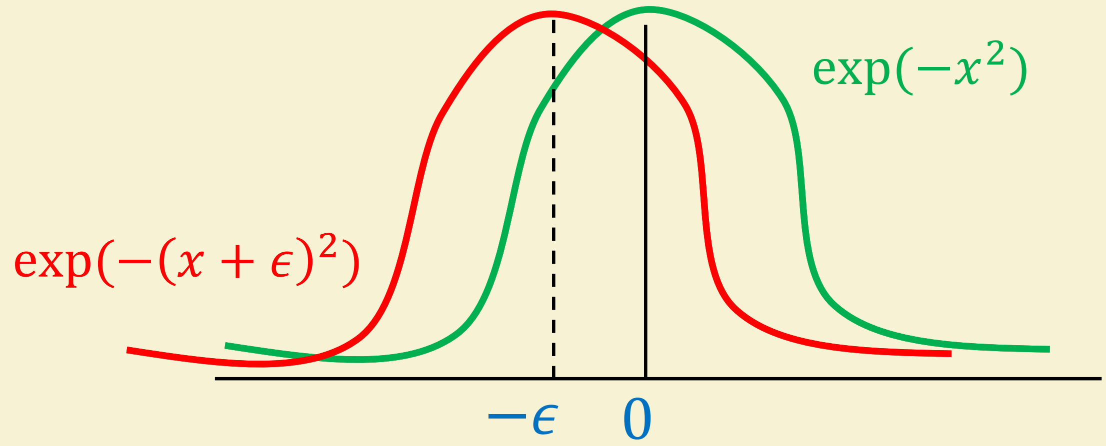
If $\Delta_{KL}(p \parallel q) = k$, k bits of "surprise"
$q \approx p$ after revealing k bits



Generalize KL to f-divergences and TV to integral probability metrics (IPM)

Normal Distribution

$p = N(0,1)$, $q = N(-\epsilon, 1)$



For const $x > 0$, $\frac{p(x)}{q(x)} \approx \frac{\exp(-x^2)}{\exp(-(x+\epsilon)^2)} \approx \exp(2\epsilon x) \approx (1 + c \cdot \epsilon)$

TV: With prob $\frac{1}{2}$, $p(x) \geq (1 + c \cdot \epsilon) \cdot q(x) \Rightarrow \Delta_{TV}(p,q) \approx \epsilon$

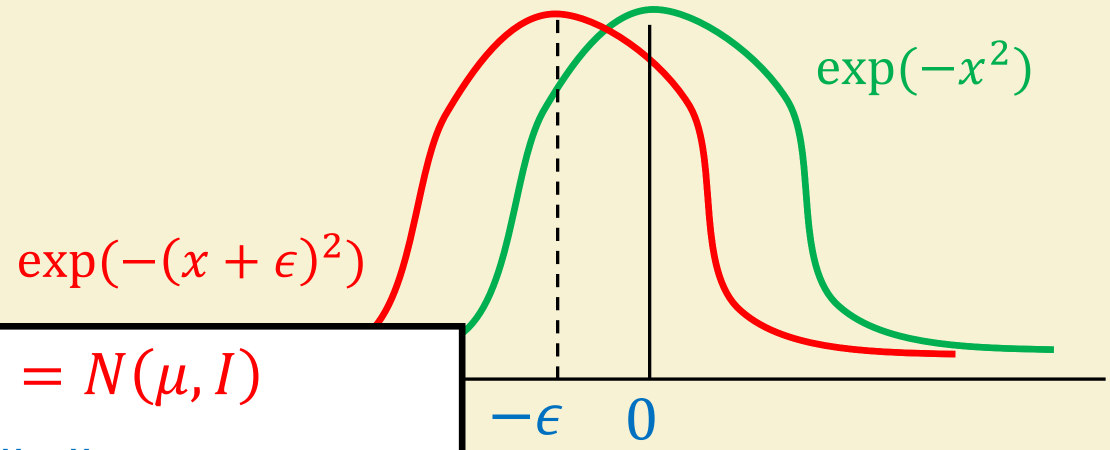
KL: For $x \sim p$, w.p. $\frac{1}{2} + \epsilon$, $\frac{p(x)}{q(x)} \approx 1 + \epsilon$, $\log \frac{p(x)}{q(x)} \approx \epsilon$

w.p. $\frac{1}{2} - \epsilon$, $\frac{p(x)}{q(x)} \approx 1 - \epsilon$, $\log \frac{p(x)}{q(x)} \approx -\epsilon$

$\Rightarrow \Delta_{KL}(p \parallel q) \approx \epsilon^2$

Normal Distribution

$p = N(0,1)$, $q = N(-\epsilon, 1)$



High dim case: $p = N(0, I)$, $q = N(\mu, I)$

- $\Delta_{TV}(p, q) \approx \|\mu\|$ (for small $\|\mu\|$)
- $\Delta_{KL}(p \parallel q) \approx \|\mu\|^2$

For … $1 + c \cdot \epsilon)$

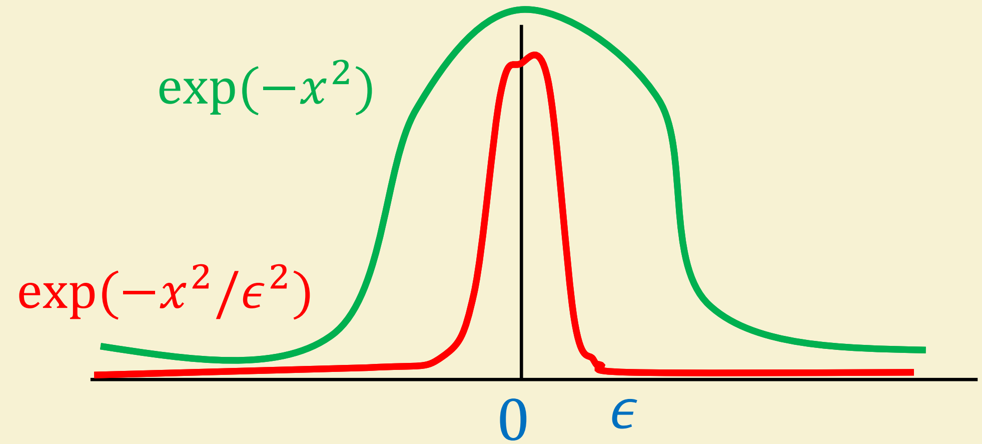
TV: With prob $\frac{1}{2}$, $p(x) \geq (1 + c \cdot \epsilon) \cdot q(x) \Rightarrow \Delta_{TV}(p, q) \approx \epsilon$

KL: For $x \sim p$, w.p. $\frac{1}{2} + \epsilon$, $\frac{p(x)}{q(x)} \approx 1 + \epsilon$, $\log \frac{p(x)}{q(x)} \approx \epsilon$

$\Rightarrow \Delta_{KL}(p \parallel q) \approx \epsilon^2$

w.p. $\frac{1}{2} - \epsilon$, $\frac{p(x)}{q(x)} \approx 1 - \epsilon$, $\log \frac{p(x)}{q(x)} \approx -\epsilon$

Normal Distribution II



$p = N(0,1)$, $q = N(0,\epsilon^2)$

TV: $\Delta_{TV}(p,q) \approx 1$

KL: With const prob, $\log \frac{p(x)}{q(x)} \approx \log \frac{\exp(-x^2)}{\exp(-x^2/\epsilon^2)} = \frac{x^2}{\epsilon^2} - x^2 \Rightarrow \Delta_{KL}(p \parallel q) \approx \frac{1}{\epsilon^2}$

High dim case: $p = N(0, I_d)$, $q = N(0, \mathrm{V})$

$$\Delta_{KL}(p \parallel q) \approx Tr\left(\mathrm{V}^{-1}\right) - d + \ln \det \mathrm{V}$$

$$= \sum \lambda_i^{-1} - d + \sum \ln \lambda_i$$

Example: $V = \epsilon^2 I \Rightarrow \Delta_{KL}(p \parallel q) \approx d/\epsilon^2 - d - 2d \ln 1/\epsilon$

If q discrete then $\Delta_{KL}(p \parallel q) = \infty$

Matching Distributions

If p is given distribution, and g is candidate generator, then

$$\Delta_{KL}(p \parallel g) = \mathbb{E}_{x\sim p}\left[\log\frac{p(x)}{g(x)}\right] = \underbrace{\mathbb{E}_{x\sim p}[\log p(x)]}_{-H(p)} - \underbrace{\mathbb{E}_{x\sim p}[\log g(x)]}_{H(p,g)}$$

Minimizing KL = Maximizing $\mathbb{E}_{x\sim p}[\log g(x)]$

Log likelihood / neg cross entropy

Want model g such that typical $x \sim p$ are likely under g

Can evaluate with samples from p and g's density map $x \mapsto g(x)$

Matching Distributions

If p is given distribution, and g is candidate generator, then

Minimizing KL = Maximizing $\mathbb{E}_{x \sim p}[\log g(x)]$

Log likelihood / neg cross entropy

Want model g such that typical $x \sim p$ are likely under g

Can evaluate with samples from p and g's density map $x \mapsto g(x)$

Memorizing model: Given $x_1, \ldots, x_n$ output $\mathrm{g} = U(\{x_1, \ldots, x_n\})$

For train $g(x_i) = \frac{1}{n}$: huge!

Useless for test

Two technical digressions

1) Distance between distributions

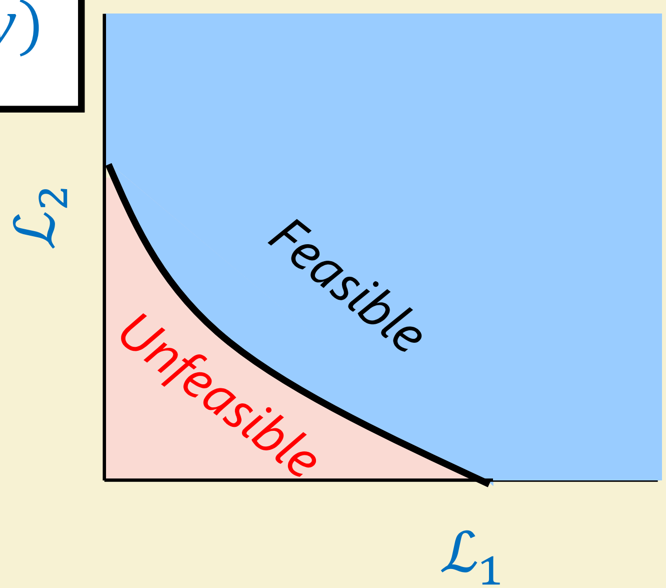
2) Optimizing multiple objectives

Multiple objectives

Want $\mathcal{L}_1(w)$ and $\mathcal{L}_2(w)$ to be small.

Pareto curve: $\mathcal{P} = \{(a,b) \in Im(\mathcal{L}_1, \mathcal{L}_2){:}\ \forall w \in \mathcal{W}\ , \mathcal{L}_1(w) \geq a\ \vee \mathcal{L}_2(w) \geq b\ \}$

THM: If $\mathcal{L}_1, \mathcal{L}_2$ convex, $\forall(a,b) \in \mathcal{P}\ \exists \lambda \geq 0$ s.t.
$a, b = \mathcal{L}_1(w), \mathcal{L}_2(w)$ for $\mathrm{w} = \arg\min \mathcal{L}_1(w) + \lambda \mathcal{L}_2(w)$



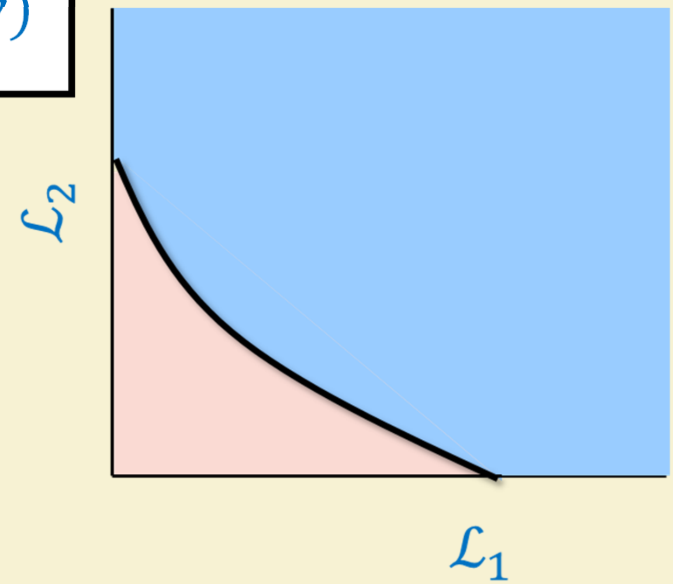
Multiple objectives

Want $\mathcal{L}_1(w)$ and $\mathcal{L}_2(w)$ to be small.

Pareto curve: $\mathcal{P} = \{(a,b) \in Im(\mathcal{L}_1, \mathcal{L}_2)\colon \forall w \in \mathcal{W}, \mathcal{L}_1(w) \geq a \vee \mathcal{L}_2(w) \geq b\}$

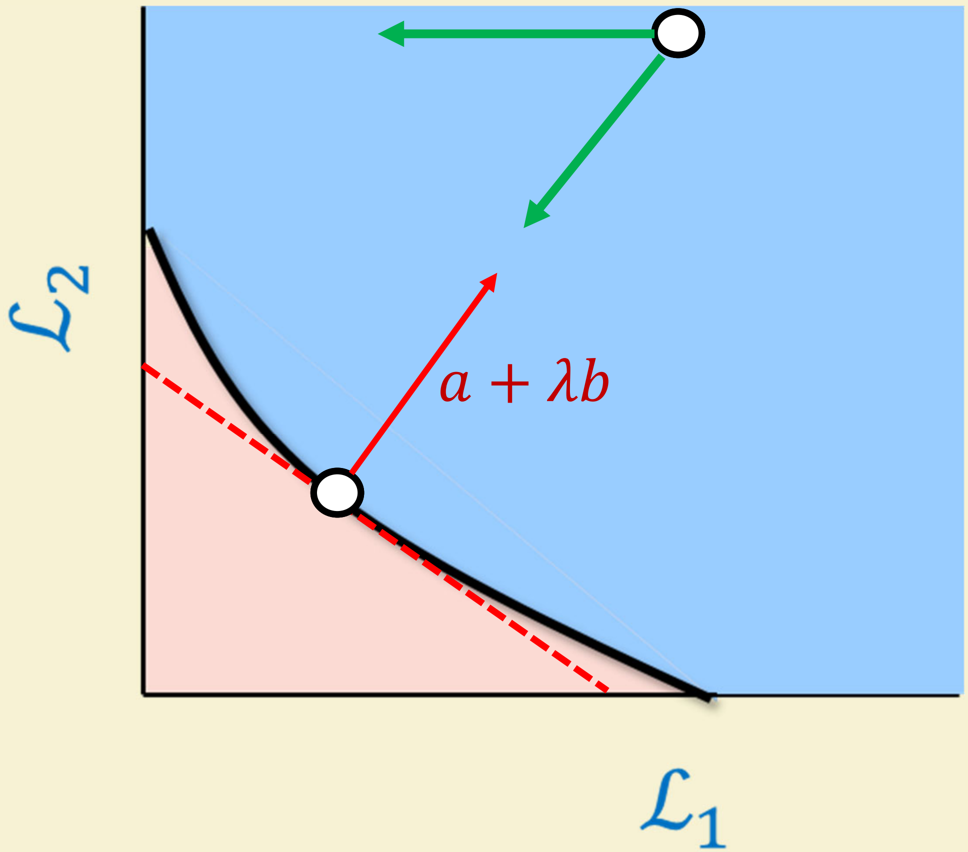
THM: If $\mathcal{L}_1, \mathcal{L}_2$ convex, $\forall (a,b) \in \mathcal{P}\ \exists \lambda \geq 0$ s.t.
$a, b = \mathcal{L}_1(w), \mathcal{L}_2(w)$ for $\mathrm{w} = \arg\min \mathcal{L}_1(w) + \lambda \mathcal{L}_2(w)$

Proof by picture



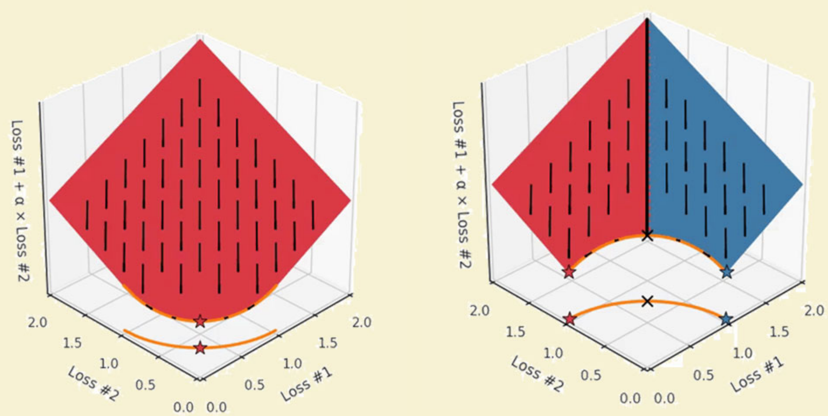
THM: If $\mathcal{L}_1, \mathcal{L}_2$ convex, $\forall (a,b) \in \mathcal{P}\ \exists \lambda \geq 0$ s.t.
$a, b = \mathcal{L}_1(w), \mathcal{L}_2(w)$ for $\mathrm{w} = \arg\min \mathcal{L}_1(w) + \lambda \mathcal{L}_2(w)$

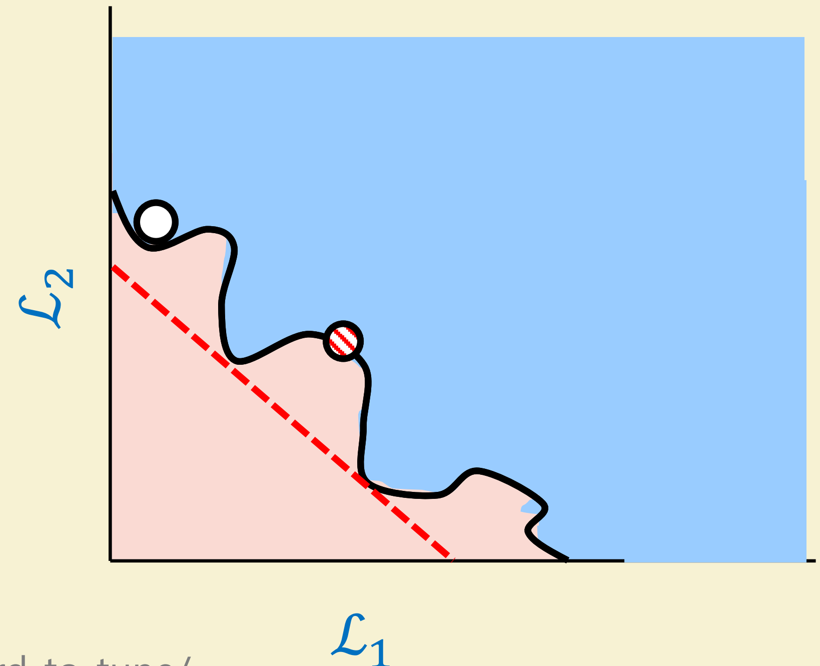
Proof by picture



Non convex case

- Some points on $\mathcal{P}$ not minima of any $\mathcal{L}_1 + \lambda\mathcal{L}_2$
- $\mathcal{L}_1 + \lambda\mathcal{L}_2$ can have multiple minima
- Depending on path, could get stuck in local minima





https://engraved.ghost.io/why-machine-learning-algorithms-are-hard-to-tune/

End of digressions

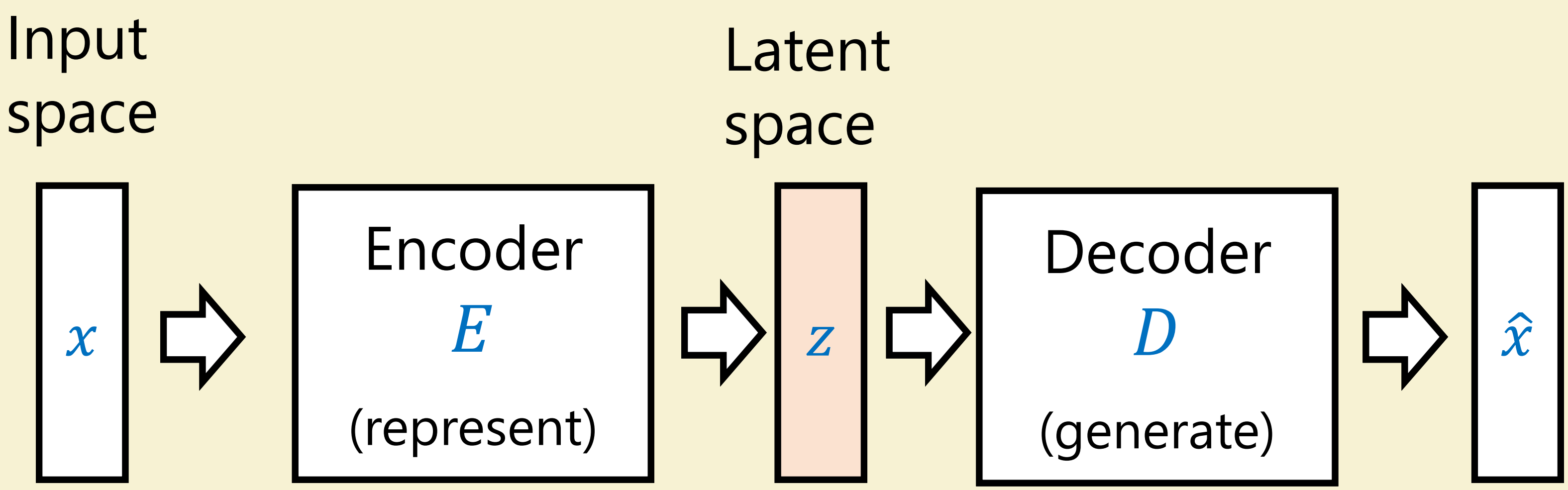
Unsupervised and semi-supervised learning

Input: $x_1, x_2, \dots, x_n \sim p \subseteq \mathbb{R}^d$

Goal: “understand” p

- Compute/approximate $x \mapsto p(x)$
- Sample fresh $x \sim p$
- Predict x_A from x_B
- Find “good” representation $r: \mathbb{R}^d \to \mathbb{R}^r$

Dream: Solve all via



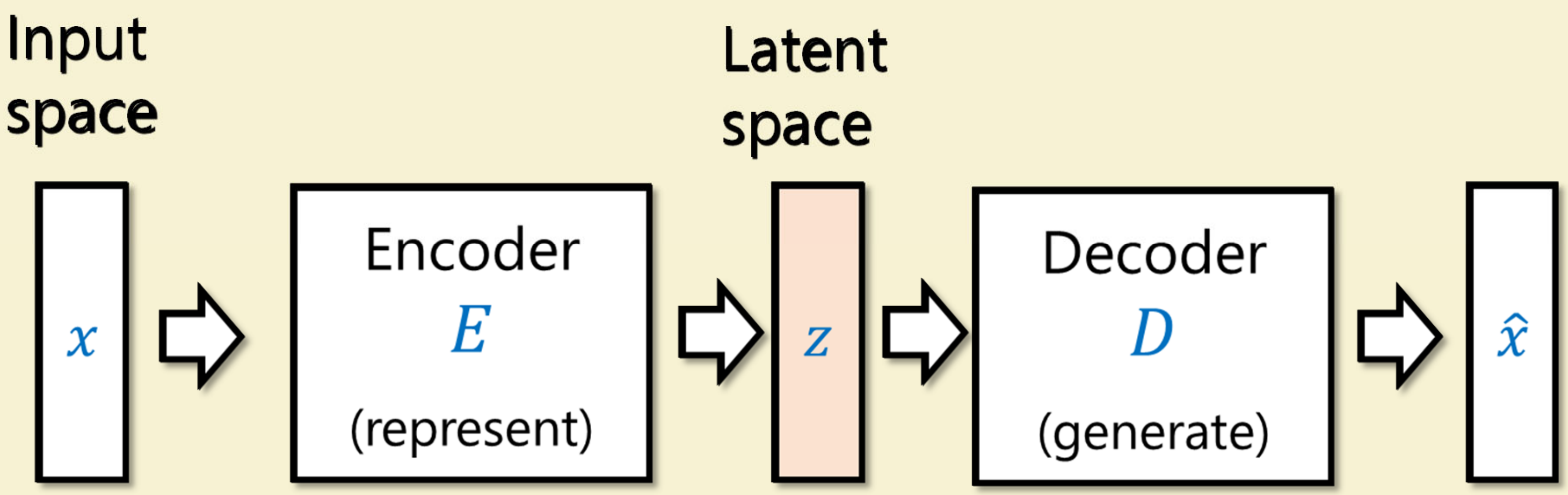
Unsupervised and semi-supervised learning

Input: $x_1, x_2, \dots, x_n \sim p \subseteq \mathbb{R}^d$

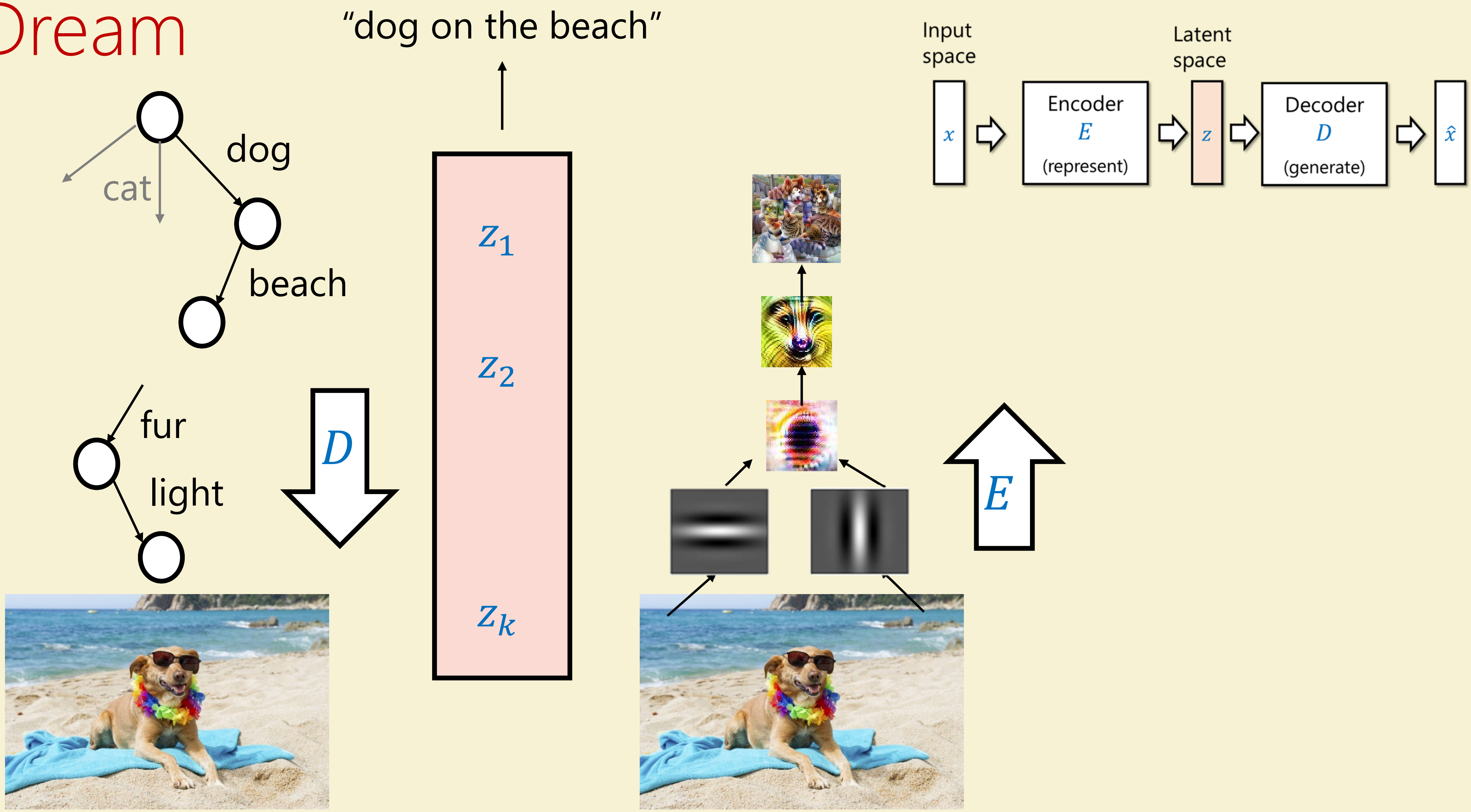
Goal: "understand" p

- Compute/approximate $x \mapsto p(x)$
- Sample fresh $x \sim p$
- Predict x_A from x_B
- Find "good" representation $r: \mathbb{R}^d \to \mathbb{R}^r$

Dream: Solve all via

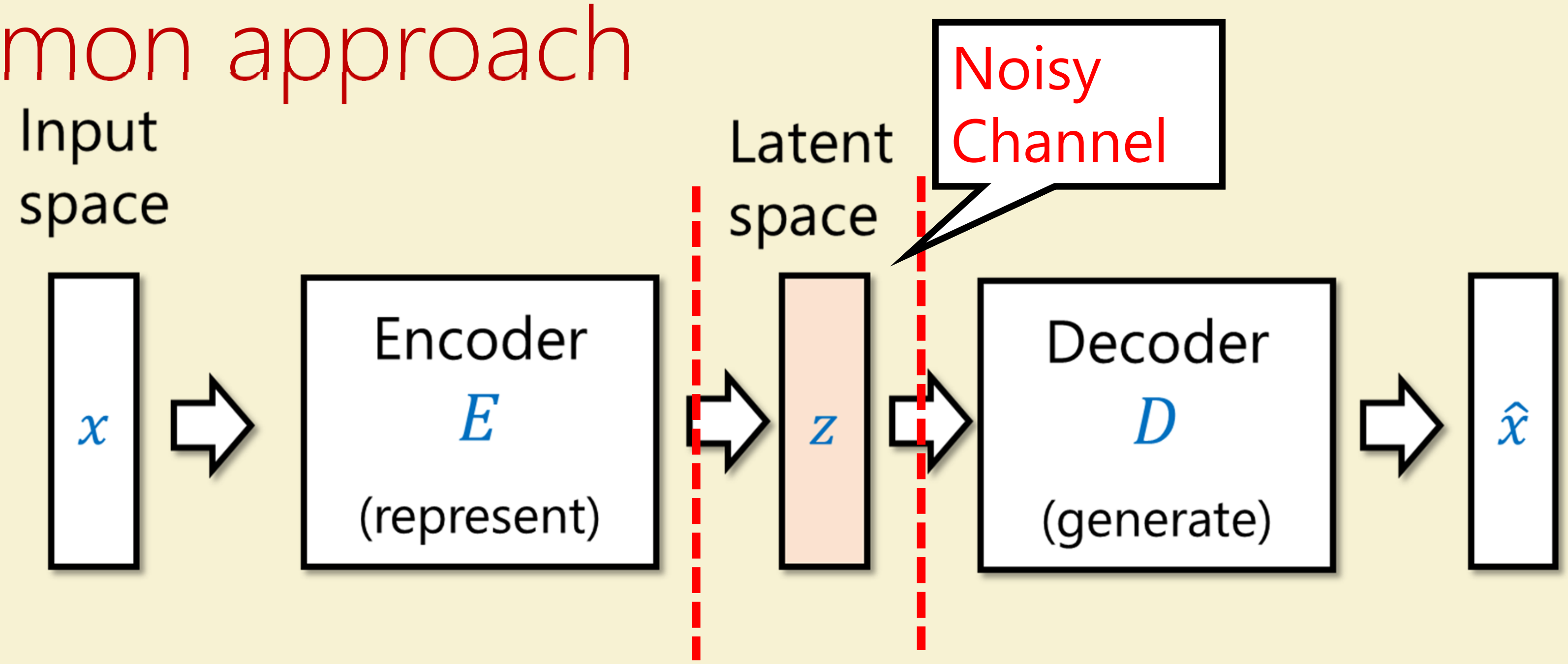


Dream



"dog on the beach"
cat
dog
beach
fur
light
D
z_1
z_2
z_k
E
Input space
Latent space
Encoder
E
(represent)
Decoder
D
(generate)
x
z
$\hat{x}$

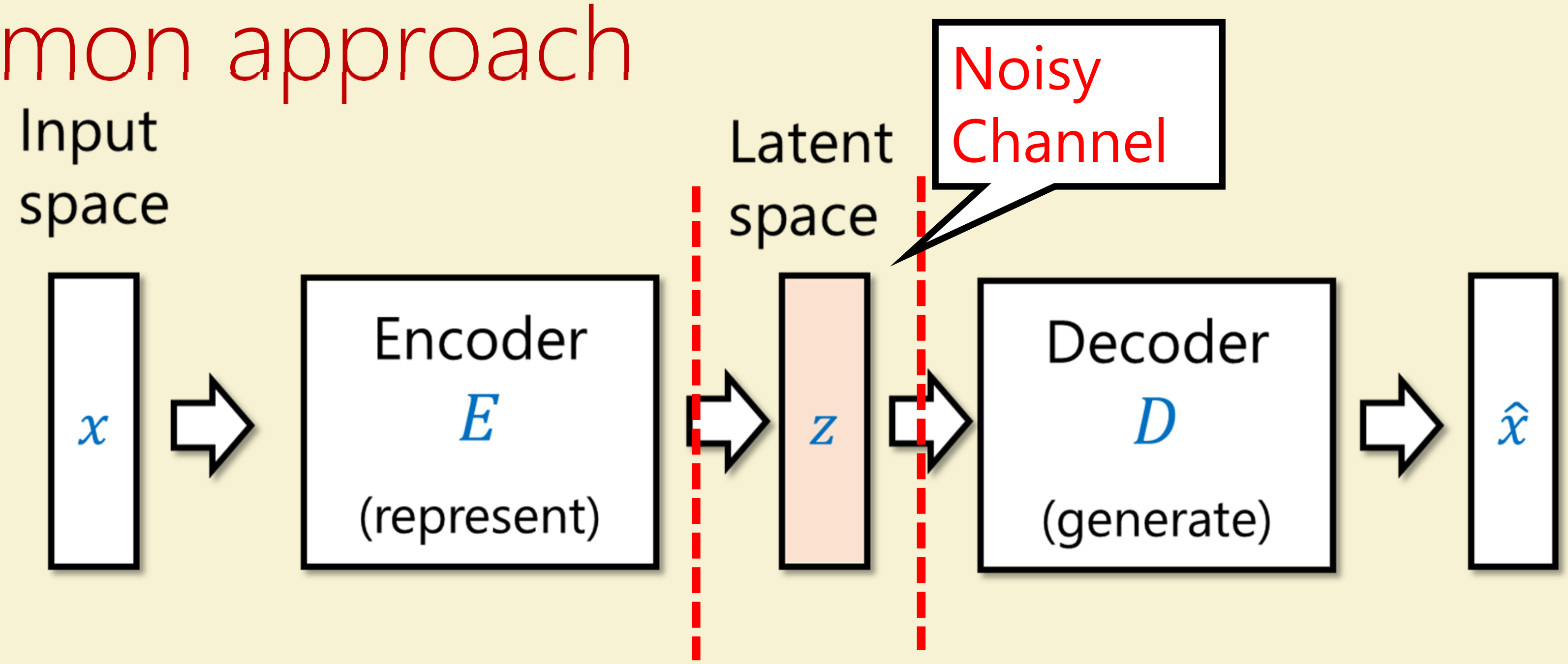
Common approach



Hope: Restricting channel requires “meaningful” latents

- Semantic dimensions
- x “similar” to $x' \Rightarrow z \approx z'$
- Sampleable z (e.g., $z \sim N(0, I)$)

Common approach

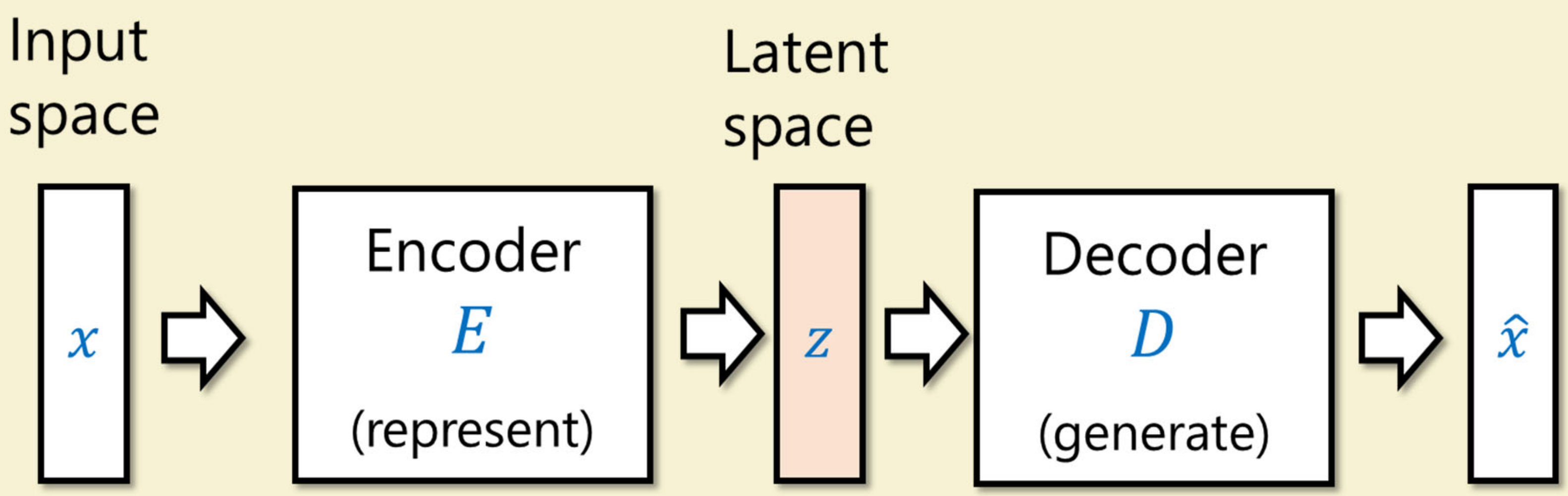


Auto Encoders: Noiseless short z

VAE/Flow: Normal noise (minimize $\Delta_{KL}(N(0,I) \parallel z)$)

VQ-VAE: Other noise model?

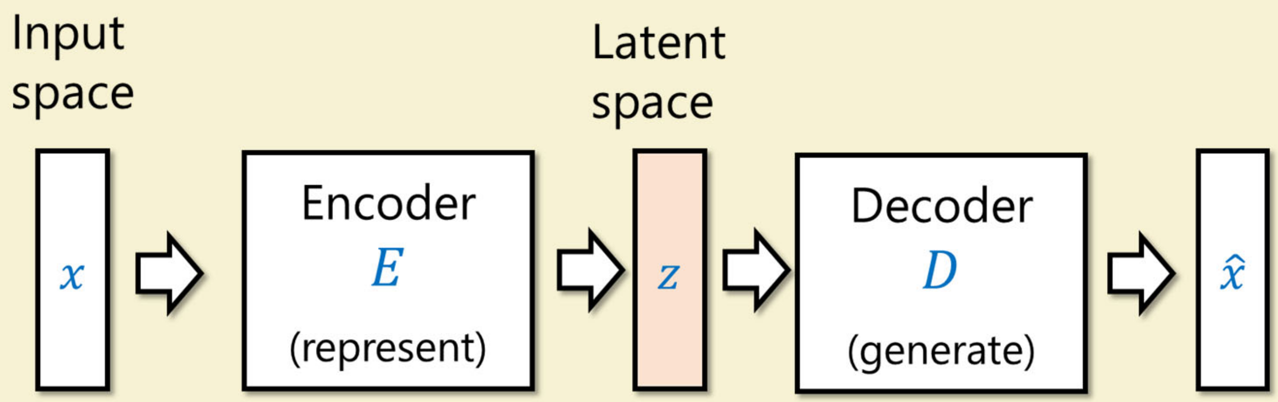
Auto Encoder



Force "understanding" by setting $r = \dim(z) \ll \dim(x) = d$

$$\min \frac{1}{n} \sum \|x_i - D(E(x_i))\|^2$$

Example: PCA

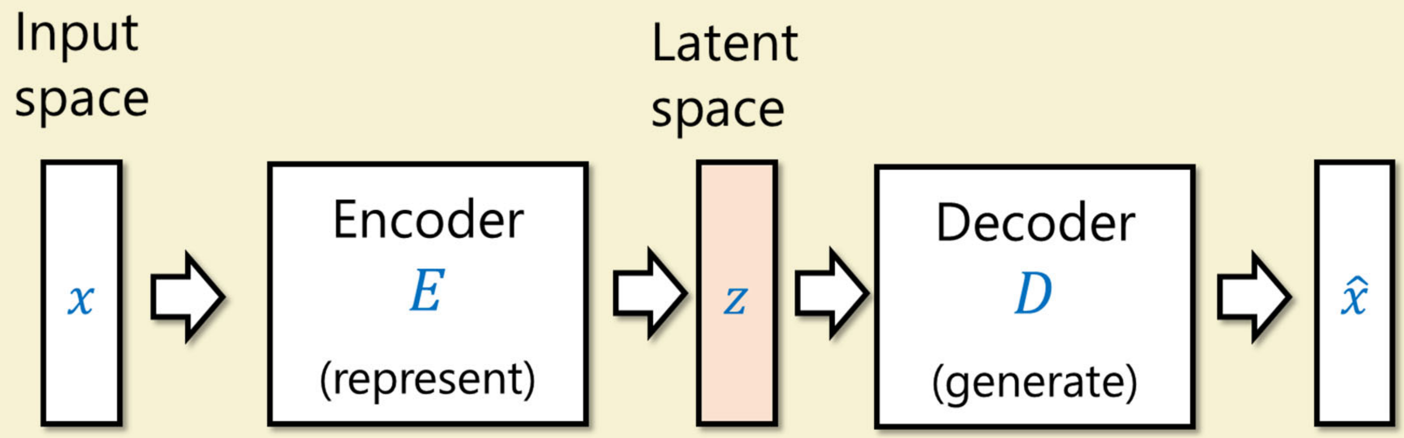


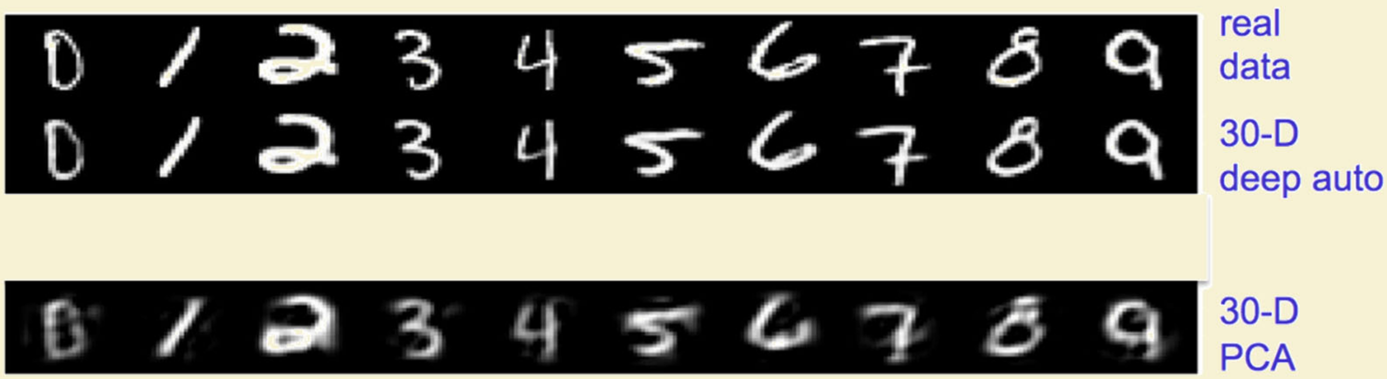
Find $E: \mathbb{R}^d \to \mathbb{R}^r$, $D: \mathbb{R}^r \to \mathbb{R}^d$ minimizing $\sum_i \| x_i - DEx_i \|^2$

Find rank r matrix L minimizing $\| (I - L) X \|^2 = Tr \left((I - L)(I - L)^\top \cdot XX^\top \right)$

$$XX^\top = \begin{pmatrix} \lambda_1 & \cdots & \\ \vdots & \ddots & \vdots \\ & \cdots & \lambda_d \end{pmatrix} \Rightarrow L = 1_{Span\,\{v_1, \ldots, v_R\}}$$

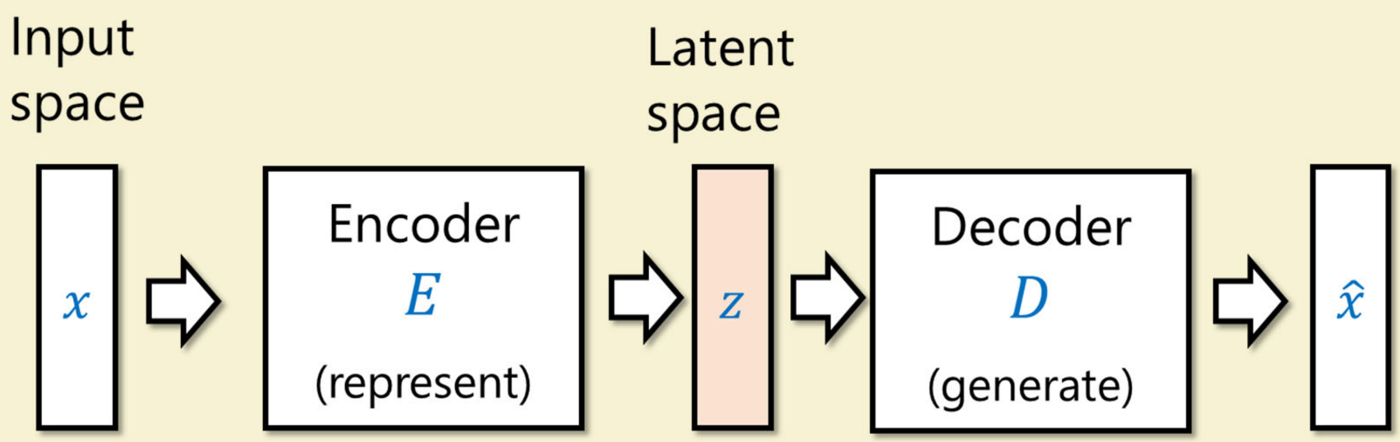
Auto Encoder





Auto Encoder

$$\min\|x - D(E(x))\|^2$$

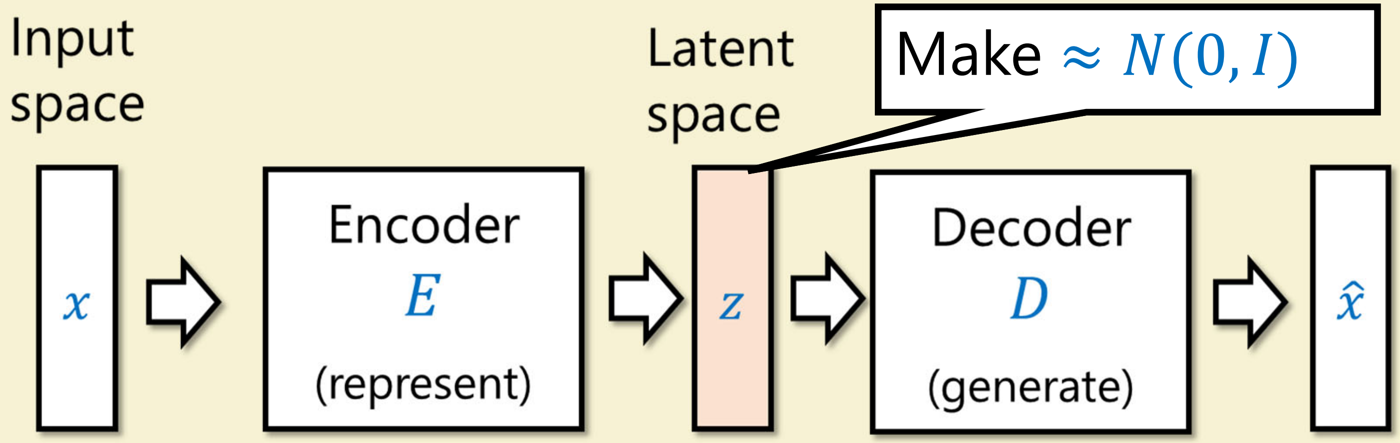


Hope: "Marley's law" $\Rightarrow z$ is informative, $D\big(N(0,I)\big) \approx$ real data

Reality: "Murphy's law" $\Rightarrow z \approx JPEG(x)$

Variational Auto Encoder

$\min \|x - D(E(x))\|^2$



Also $\min \Delta_{KL}(E(x) \parallel N(0, I))$

w.r.t. *fixed* x

$\Rightarrow E$ is *randomized*

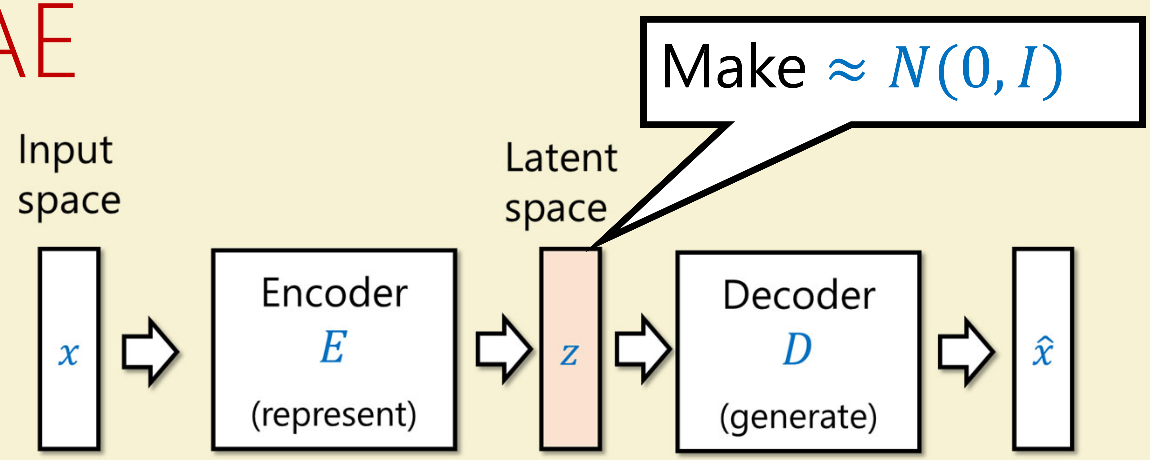
$\mu, \sigma \longrightarrow v \sim N(\mu, \sigma^2)$

$v = \mu + \sigma t \quad t \sim N(0,1)$

Another view on VAE

$$\min \|x - D(E(x))\|^2$$

Also $\min \Delta_{KL}(E(x) \parallel N(0,I))$



Let $p_x = z \sim N(0,I) \mid D(z) = x$

$q_x = E(x)$

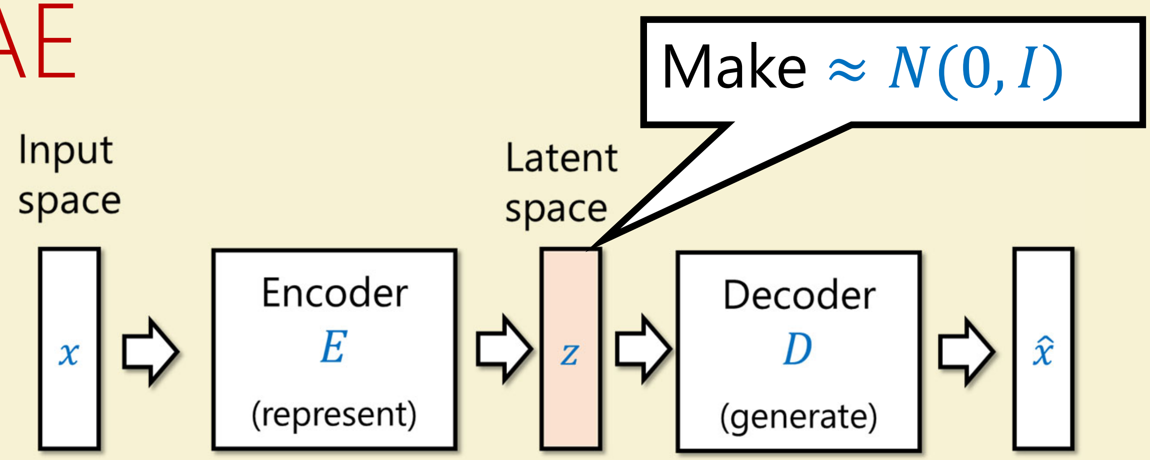
$$0 \leq \Delta_{KL}(q_x \parallel p_x) = H(q_x) - \mathbb{E}_{z \sim q_x}[\log p_x(z)] = H(q_x) - \mathbb{E}_{z \sim q_x}\left[\log\left(\frac{\Pr[N = z \wedge D(z) = x]}{\Pr[D(N) = x]}\right)\right]$$

$$= \underbrace{\log \Pr[D(N) = x]}_{\text{Log likelihood}} - \underbrace{\left(\mathbb{E}_{z \sim q_x}[\log \Pr[N = z \wedge D(z) = x]] - H(q_x)\right)}_{\text{ELBO}}$$

Another view on VAE

$\min \|x - D(E(x))\|^2$

Also $\min \Delta_{KL}(E(x) \parallel N(0,I))$



Let $p_x = z \sim N(0,I) \mid D(z) = x$

$q_x = E(x)$

$$0 \le \Delta_{KL}(q_x \parallel p_x) = H(q_x) - \mathbb{E}_{z\sim q_x}[\log p_x(z)] \;\; = H(q_x) - \mathbb{E}_{z\sim q_x}\left[\log\left(\frac{\Pr[N=z \wedge D(z)=x]}{\Pr[D(N)=x]}\right)\right]$$

$$= \underbrace{\log \Pr[D(N)=x]} - \underbrace{\left(\mathbb{E}_{z\sim q_x}[\log \Pr[N=z \wedge D(z)=x]] - H(q_x)\right)}$$

Log likeliho

$\approx -\|x - D(E(x))\|^2$

reconstruction term

$\approx \mathrm{k} - \Delta_{KL}(E(x) \parallel N(0,I))$

divergence term

In practice (?)

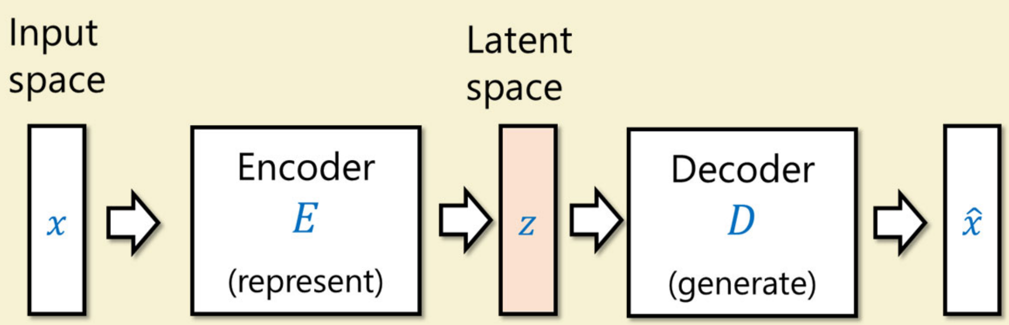
Sunglasses direction

Blond hair direction

Hou, Shen, Sun, Qiu, 2016

See also https://www.compthree.com/blog/autoencoder/

VAE pros & cons



E (and* D) randomized

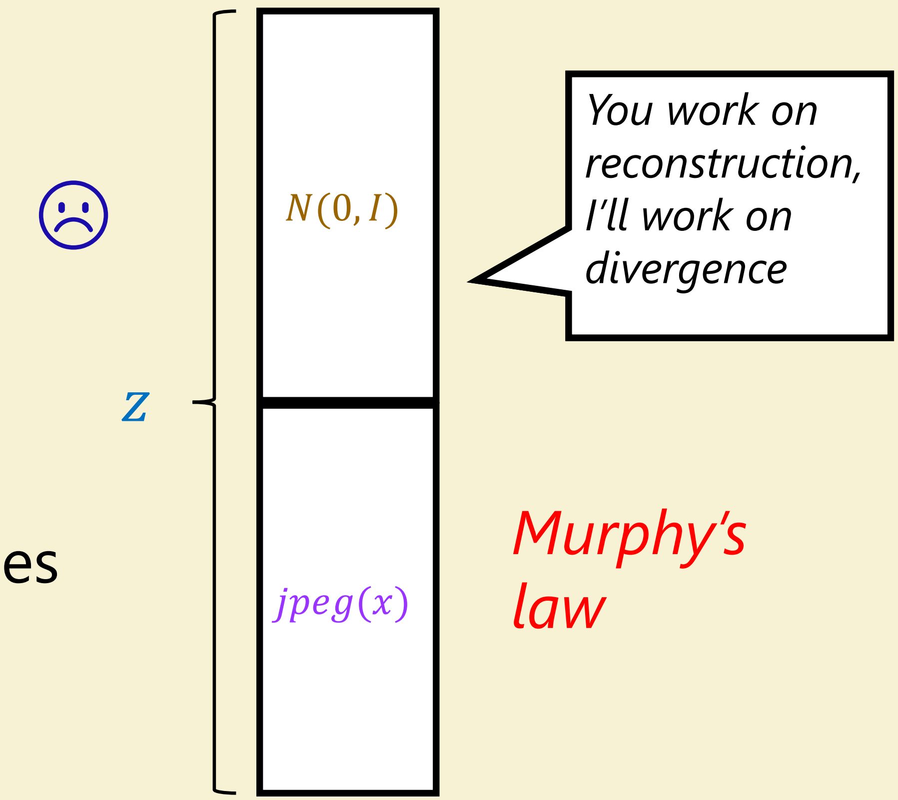
☹ Blurry images

😃 Induces geometry on latent variables

$$z \approx z' \Rightarrow D(z) \approx D(z')$$

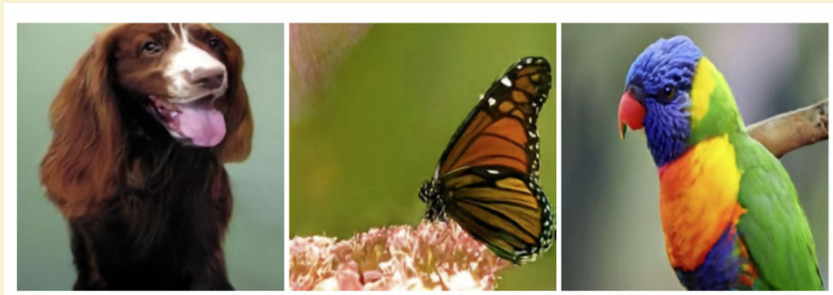
$$\min \|x - D(E(x))\|^2$$

Also $\min \Delta_{KL}(E(x) \parallel N(0, I))$



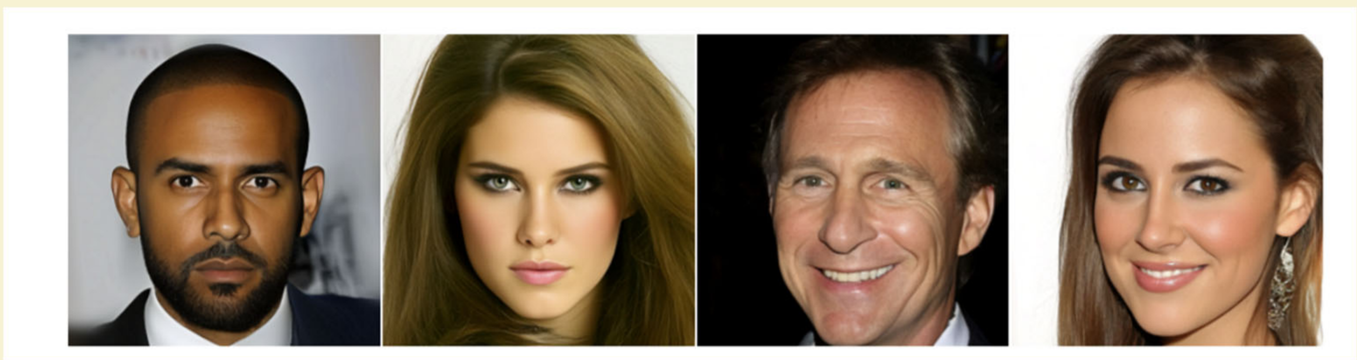
Improved VAEs

Vector quantized VAEs



van den Oord, Vinyals, Kavukcuoglu, 17
Razavi, van den Oord, Vinyals, 19

Hierarchical VAEs



Vahdat, Kautz , 20

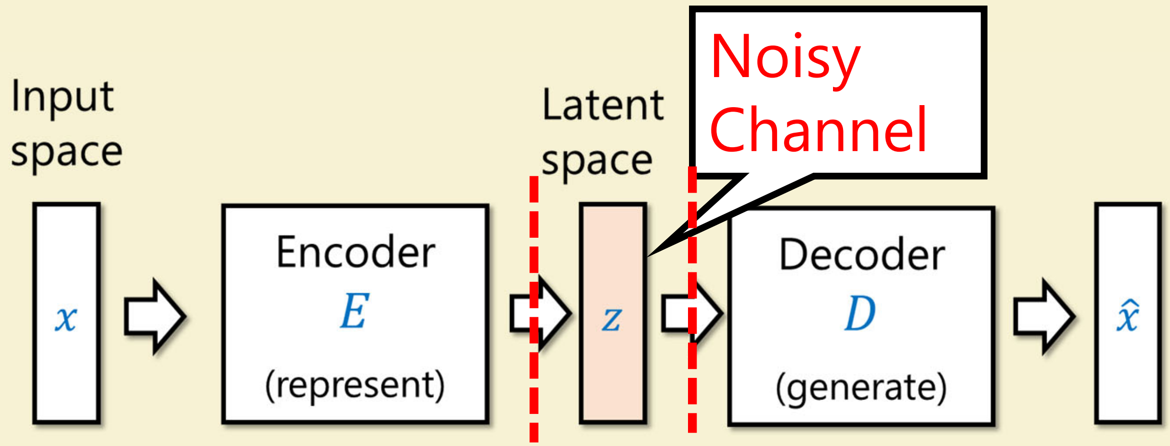
Vector quantization (VQ-VAE, Attention)

Given $S = \{v_1, \dots, v_m\} \in \mathbb{R}^d$

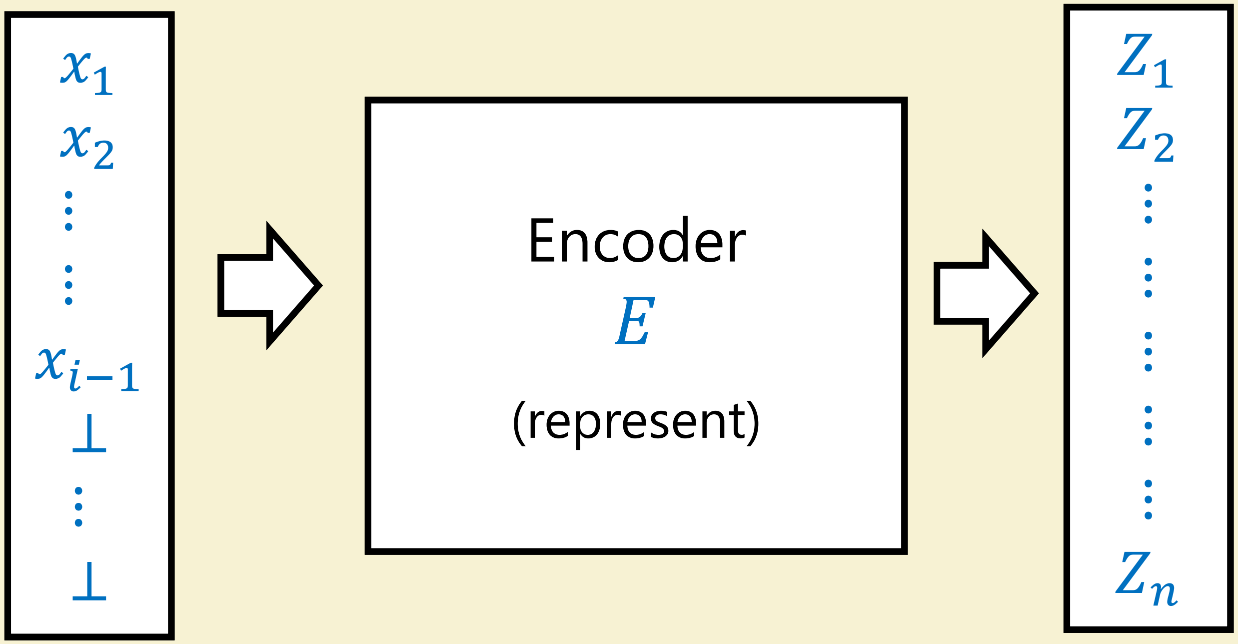
map $w \in \mathbb{R}^d$ to $\arg\max_{v \in S} \langle w, v_i \rangle$

or to $\sum \alpha_i v_i$ where $\vec{\alpha} = \operatorname{soft\,max}(\langle w, v_1 \rangle, \dots, \langle w, v_m \rangle)$

Form of encoding / noise resilience



Auto-regressive models



$x_1, \dots, x_n$ elements in $S \cup \{\perp\}$

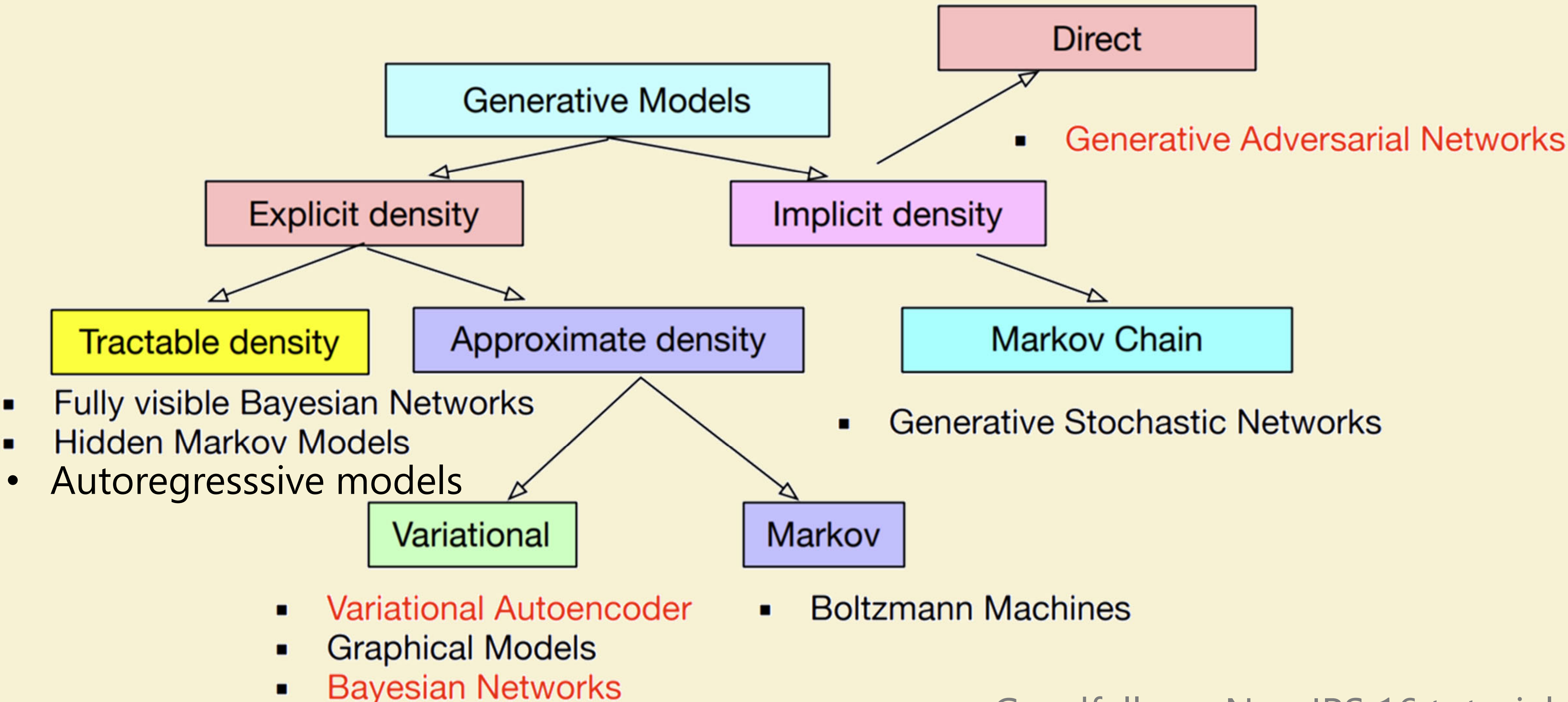
D_i distribution over S

$D_i = D_i(x_1 \dots x_{i-1})$

$D_i \approx D_i | \, x_1 \dots x_{i-1}$

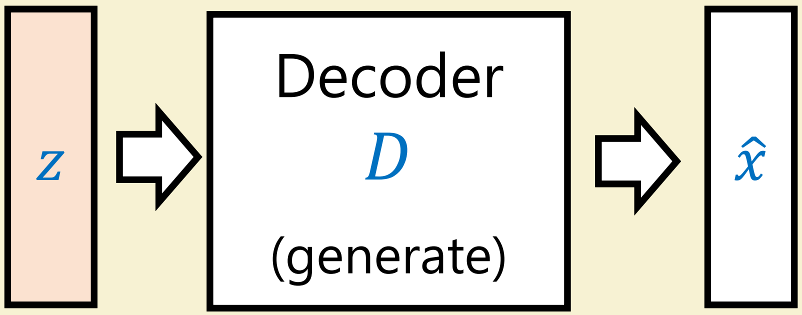
Image GPT https://openai.com/blog/image-gpt/

Metrics for generative models



Goodfellow , NeurIPS 16 tutorial

Metrics for generative models



$$g(x) = \Pr[\hat{x} = x]$$

Negative log likelihood: $-\mathbb{E}_{x\sim X} \log g(x)$

Bits per pixel: $-\dfrac{\mathbb{E}_{x\sim X} \log g(x)}{\mathrm{d}}$

Log Perplexity: $-\dfrac{\log \mathbb{E}_{x\sim X}\, g(x)}{d} = \log\left(\prod_{i=1}^{d} g(x_i | x_{<i})\right)^{1/d}$

Metrics without densit

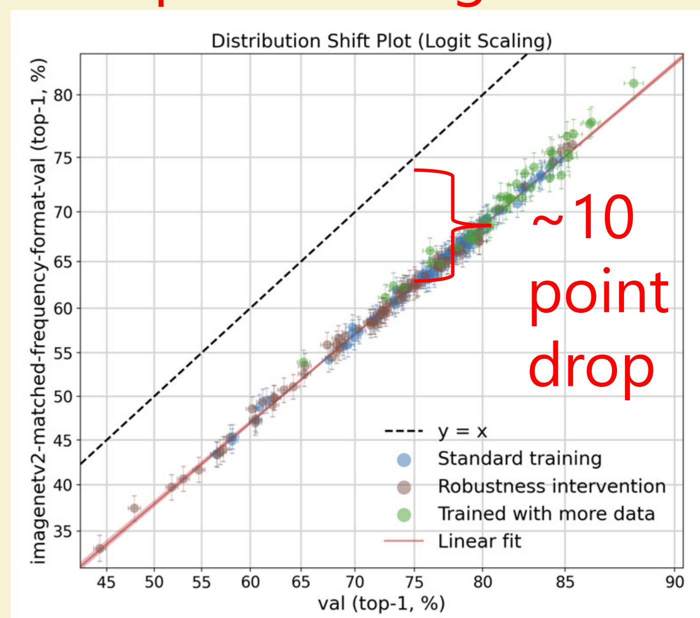
y: random class
$IN(\hat{x})$: probability dist of $y(\hat{x})$
according to Inception v3

$= x]$

Know it when I see it?

Log inception score: $\Delta_{KL}(IN(\hat{x}) \parallel y) = I(\,\hat{x}\,; IN(\hat{x})\,)$

Compare w ImageNet v2



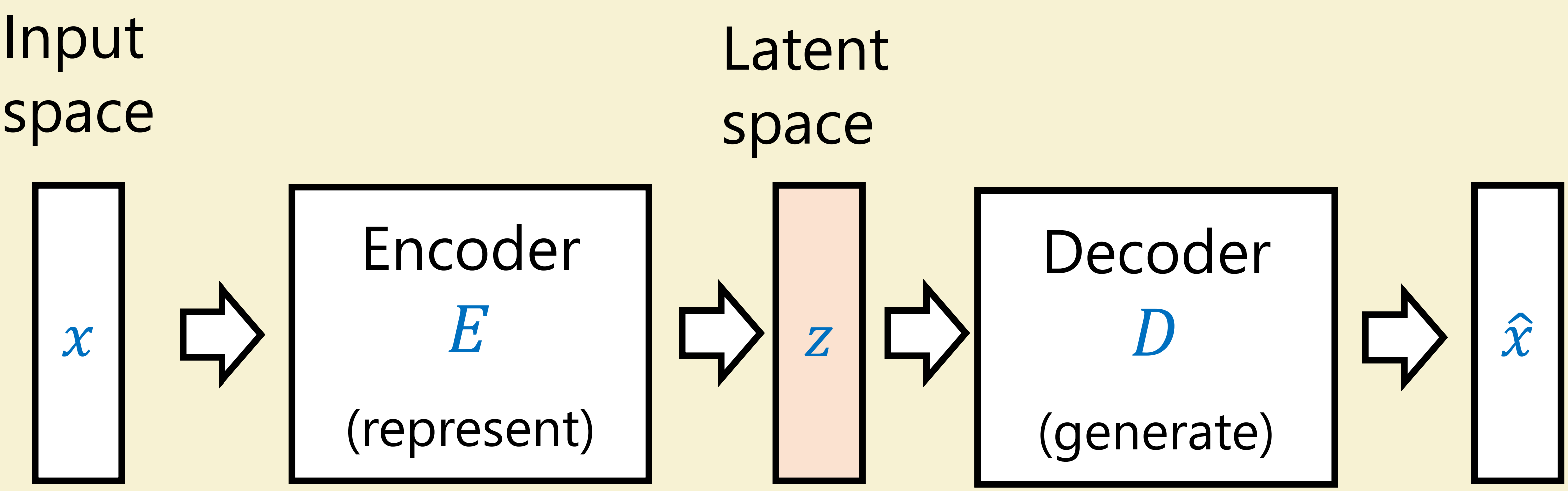
Ravuri-Vinyalis 2019:

Train with BigGAN instead of ImageNet

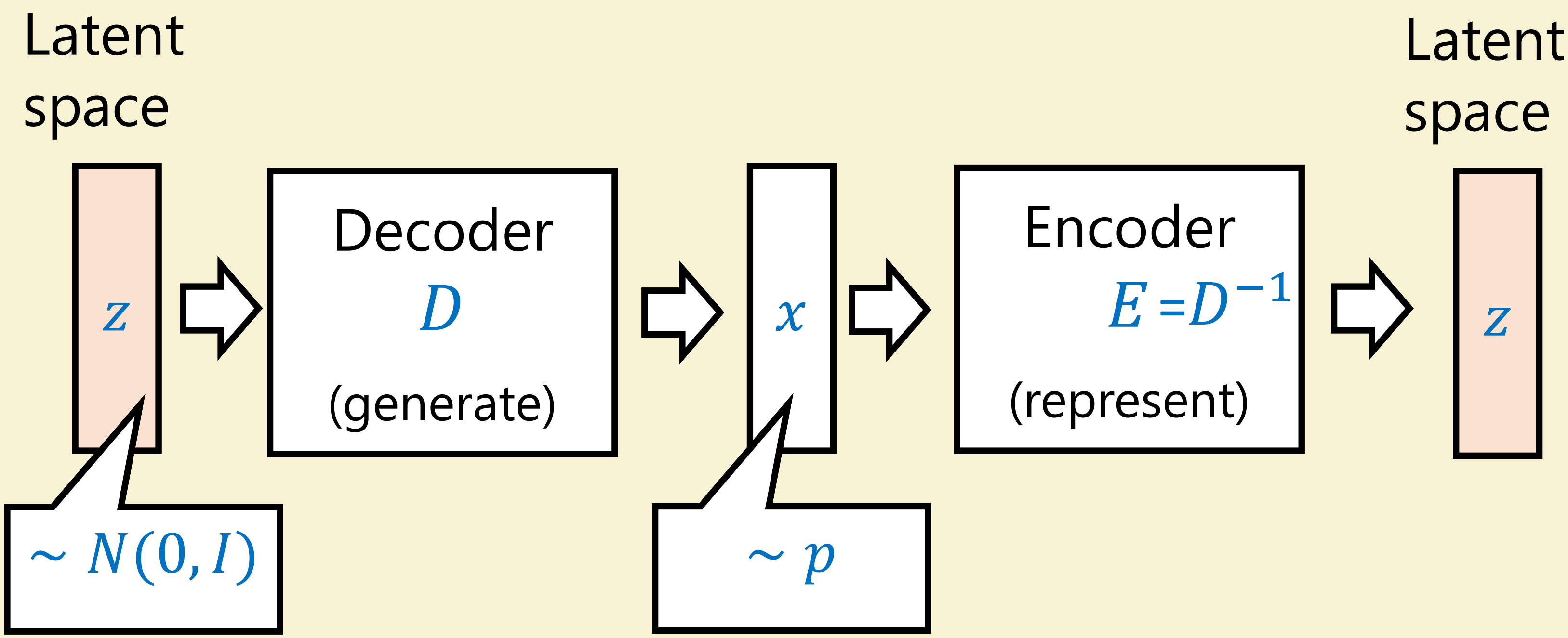
Accuracy drops from 74% to 5%-43%

Uncorrelated with Inception score

Flow models

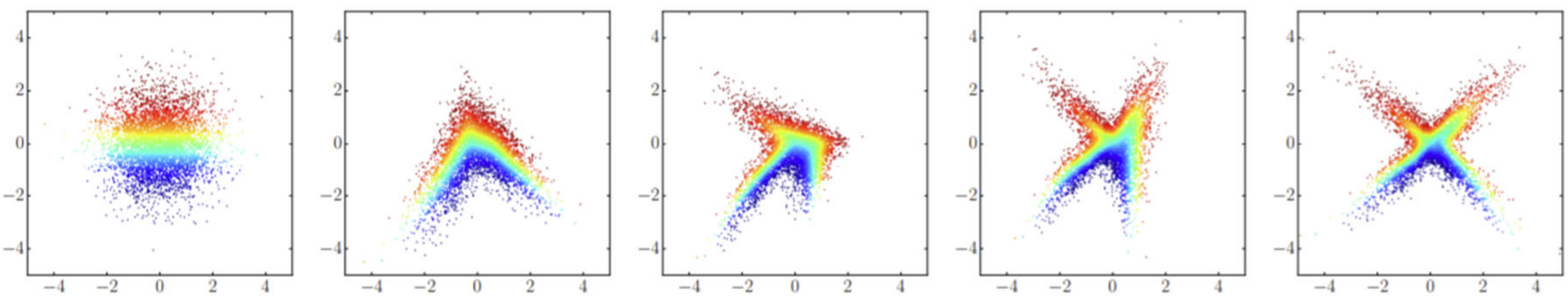

Input space
Latent space
x
Encoder
E
(represent)
z
Decoder
D
(generate)
$\hat{x}$

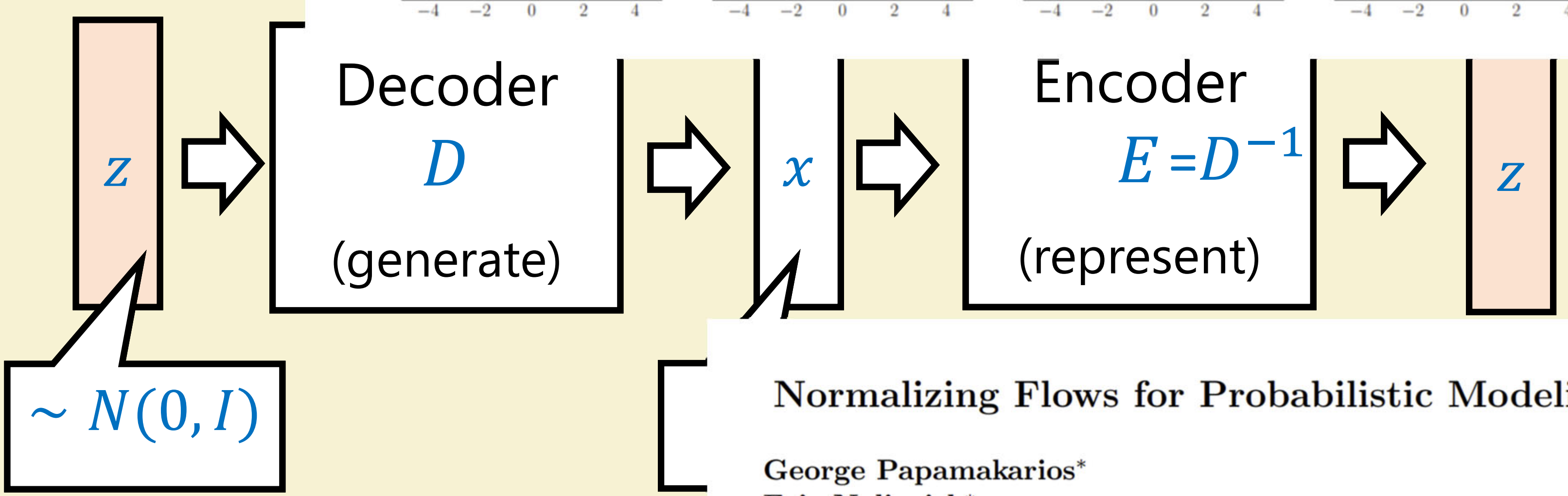
Flow models



Invertible and differentiable map $D: \mathbb{R}^d \to \mathbb{R}^d$ s.t. $D(N) = p$

Flow m



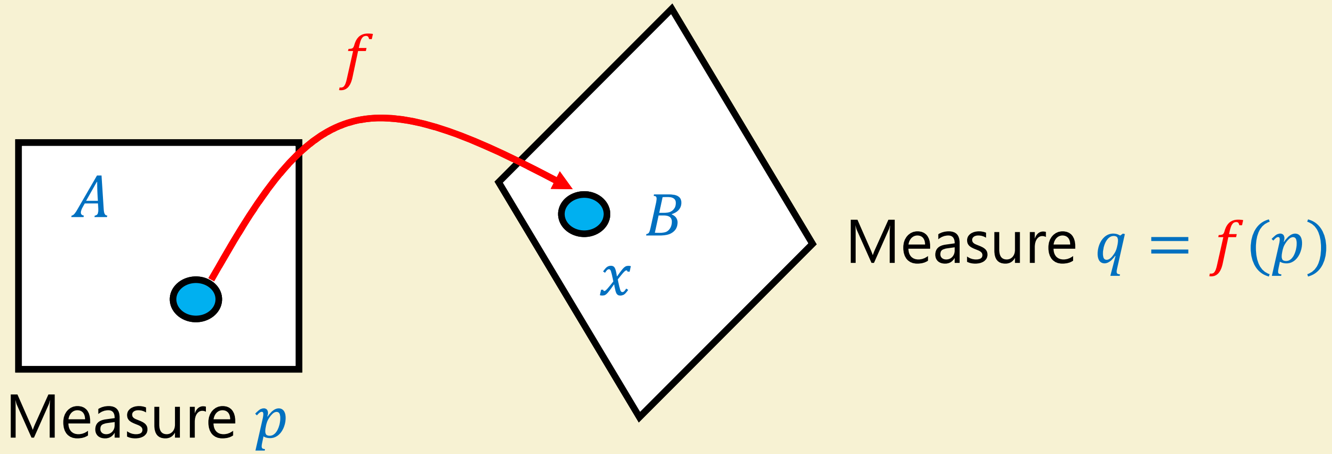


Normalizing Flows for Probabilistic Modeling and Inference

George Papamakarios* GPAPAMAK@GOOGLE.COM
Eric Nalisnick* ENALISNICK@GOOGLE.COM
Danilo Jimenez Rezende DANILOR@GOOGLE.COM
Shakir Mohamed SHAKIR@GOOGLE.COM
Balaji Lakshminarayanan BALAJILN@GOOGLE.COM
DeepMind

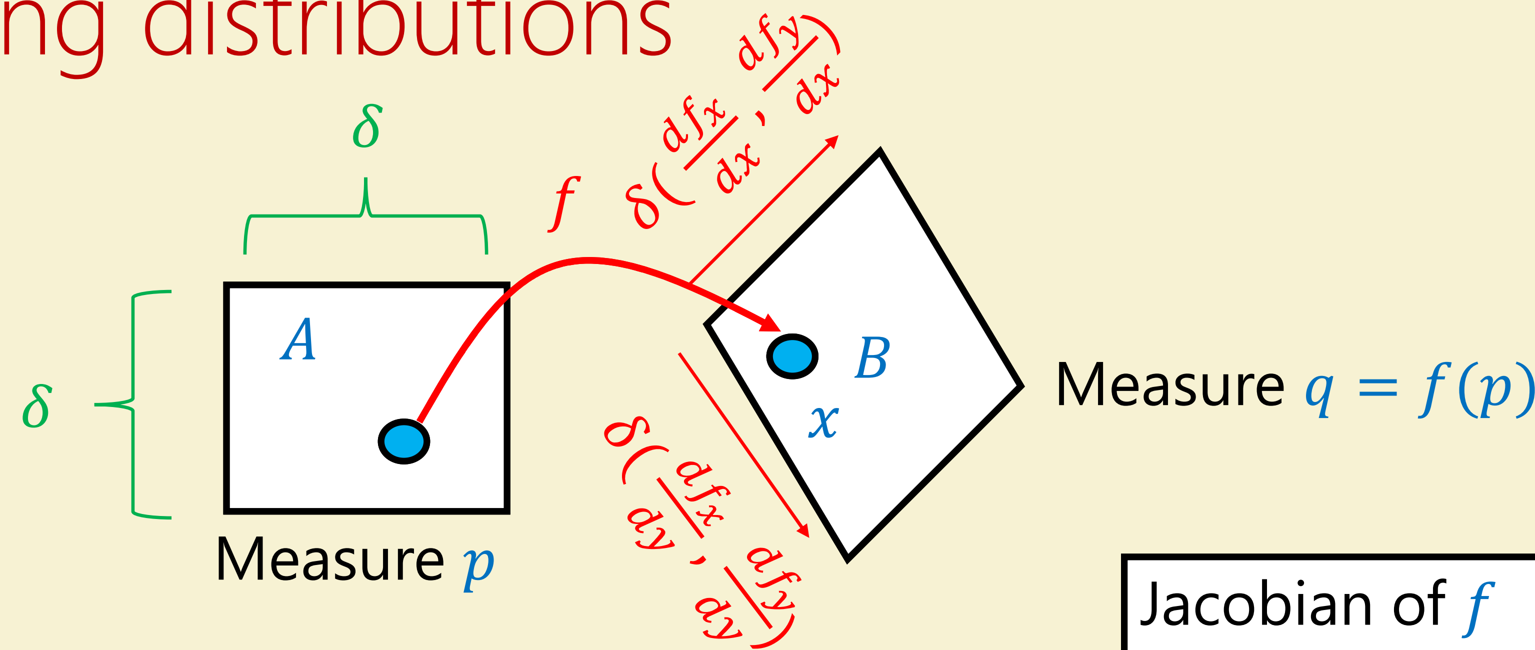
Invertible and differentiable map $D\colon \mathbb{R}^d \to \mathbb{R}^d$ s.t. $D(N) = p$

Mapping distributions



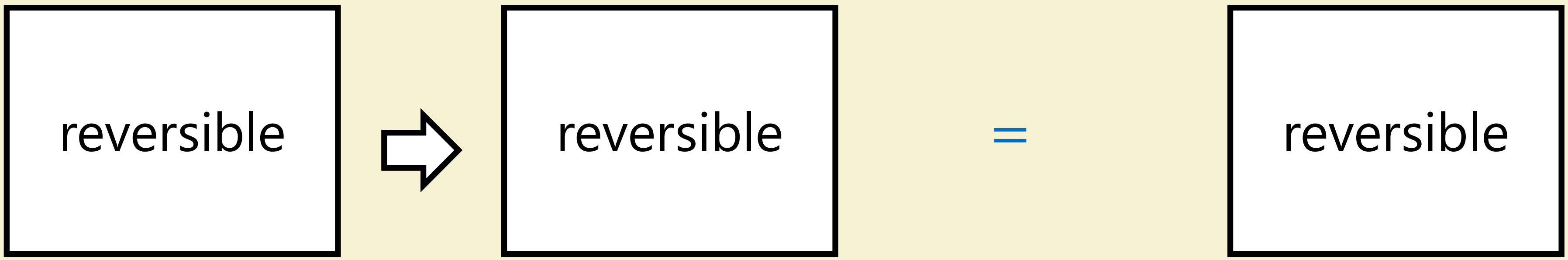
$$q(x) = p\left(f^{-1}(x)\right) \cdot \frac{Vol(A)}{Vol(B)}$$

Mapping distributions

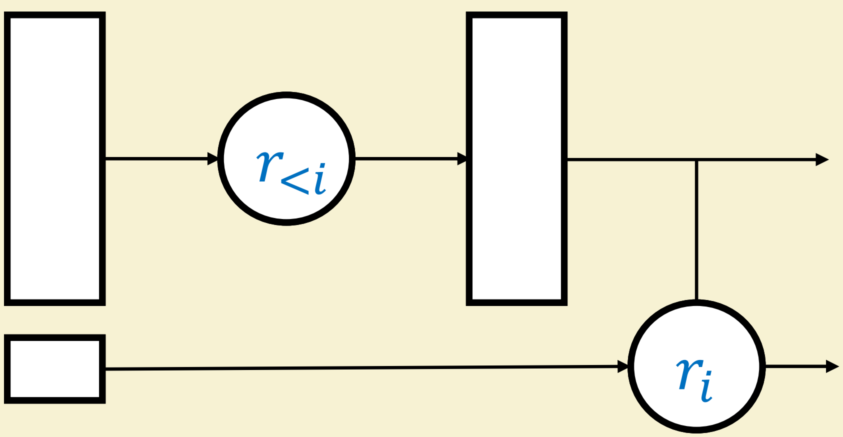


$$q(x) = p\left(f^{-1}(x)\right) \cdot \frac{Vol(A)}{Vol(B)} = p\left(f^{-1}(x)\right) \cdot \frac{\delta^2}{\delta^2} \cdot \left(\det \begin{pmatrix} \frac{df_x}{dx} & \frac{df_y}{dx} \\ \frac{df_x}{dy} & \frac{df_y}{dy} \end{pmatrix} \right)^{-1}$$

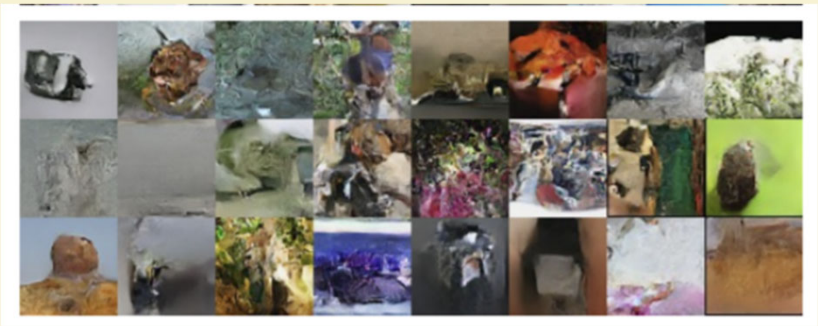
Constructing flow model



Making computation reversible (c.f. quantum, block ciphers)



Flows in practice



Dinh, Sohl-Dickstein, Bengio 17

Ho,Chen, Srinivas, Duan, Abbeel 19

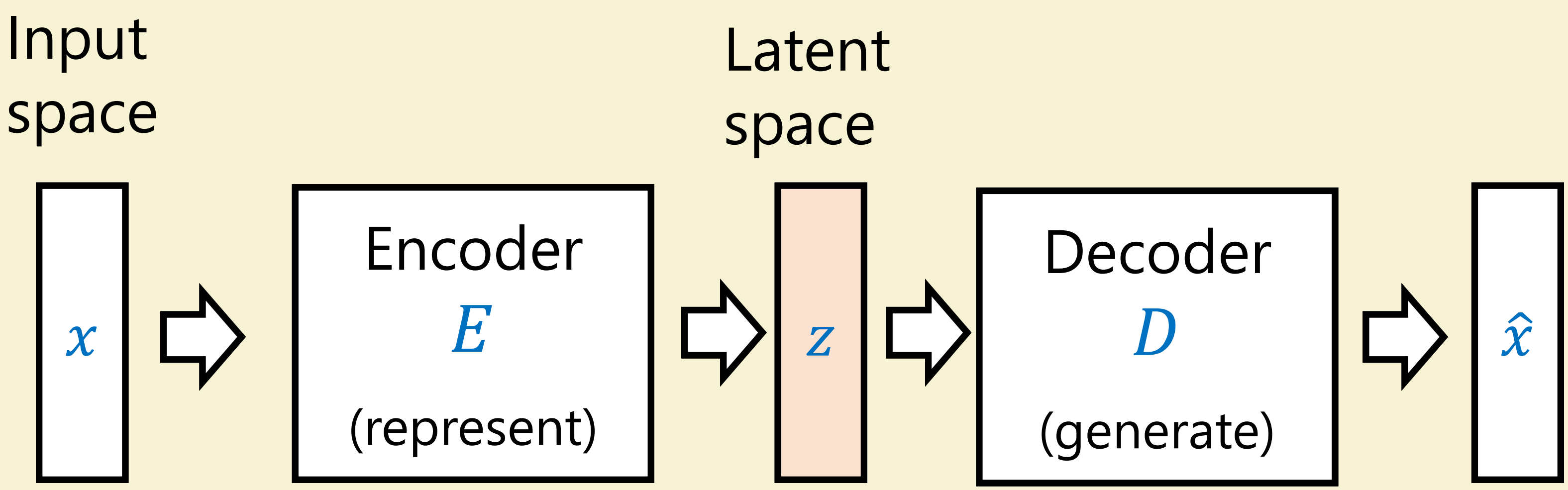
Unsupervised and semi-supervised learning

Input: $x_1, x_2, \ldots, x_n \sim p \subseteq \mathbb{R}^d$

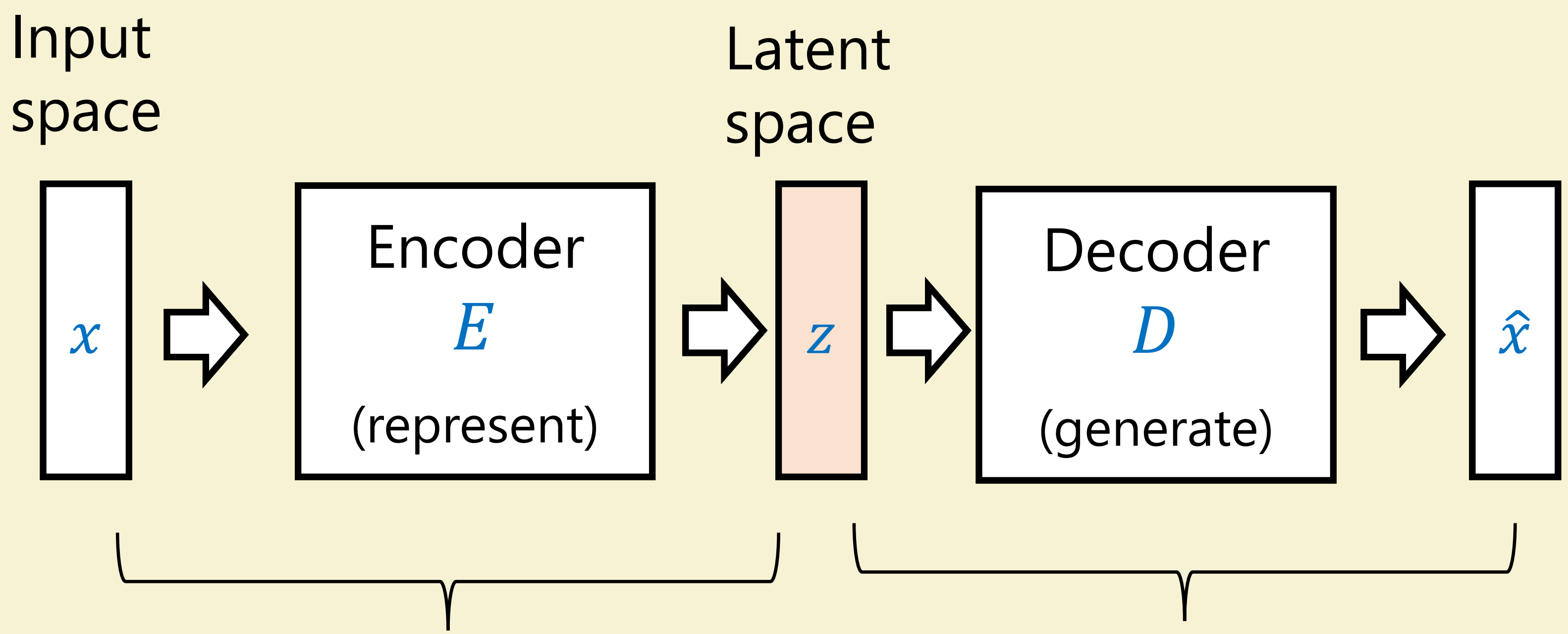
Goal: "understand" p

- Compute/approximate $x \mapsto p(x)$
- Sample fresh $x \sim p$
- Predict x_A from x_B
- Find "good" representation $r: \mathbb{R}^d \to \mathbb{R}^r$

Dream: Solve all via



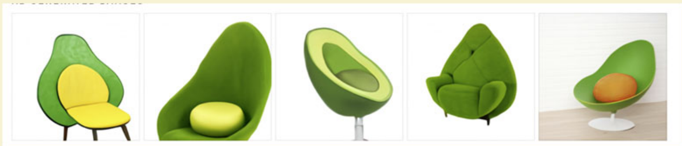
Giving up on (part of) the dream

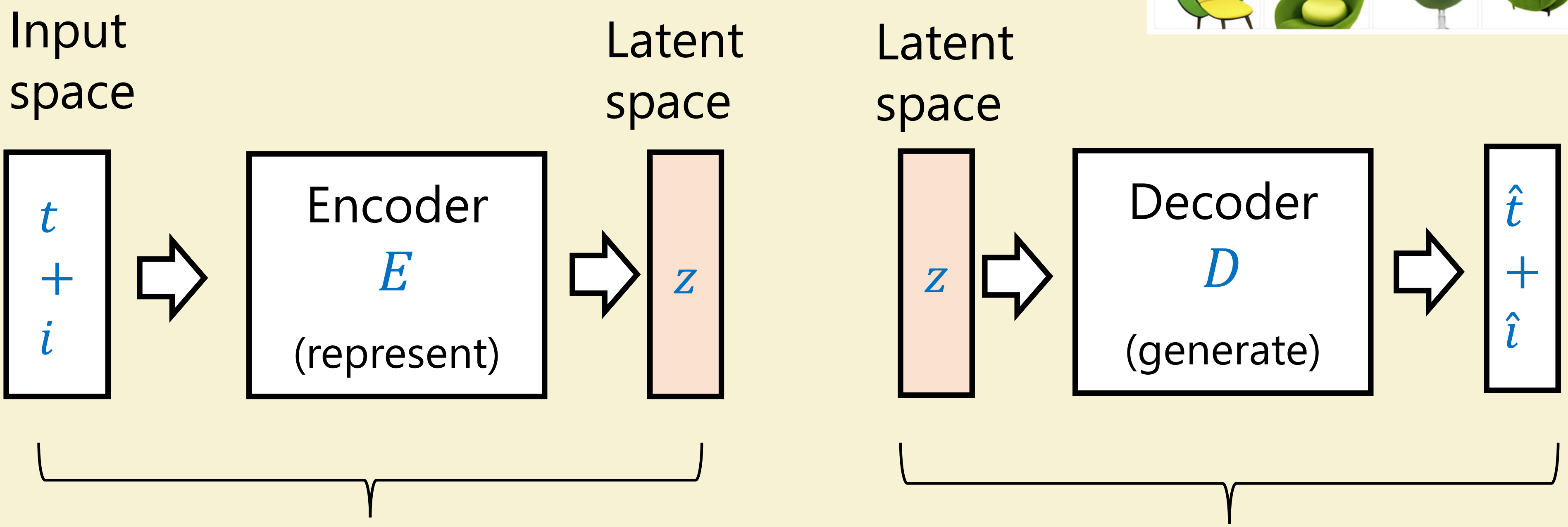


Self-Supervised learning
Representation without generation

GAN
Generation without representation / density estimation

CLIP + DALL-E / Text+ Image





CLIP
Representation for (text,image)

DALL-E
Generation of (text,image) pairs

Contrastive learning

Loss: Representations $u_1, \dots, u_n$ and $v_1, \dots, v_n$

u_i represents "similar object" to v_i

Define $M_{i,j} = f(u_i \cdot v_j)$ for monotone $f: \mathbb{R} \to \mathbb{R}$ (e.g, $f(x) = \exp(\tau \cdot x)$)

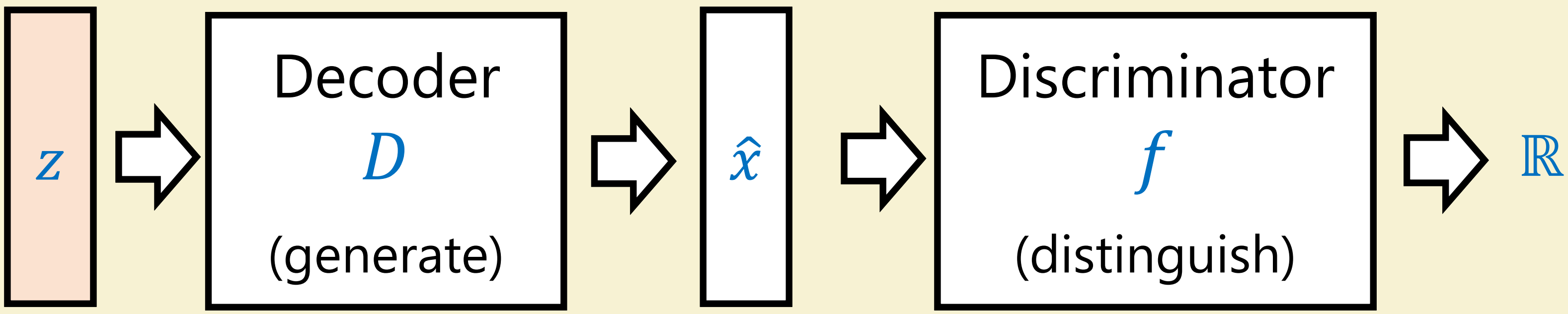
Loss = $\frac{\sum_i M_{i,i}}{\sum_{i \neq j} M_{i,j}}$

Similar objects have nearby representation

SIMCLR: $x_1, \dots., x_n$ images, u_i, v_i independent augmentations of x_i

CLIP: (u_i, v_i) matching text/image pair

Generative Adversarial Networks



$$\text{loss} = \max_{f \in \mathbb{R}^d \to \mathbb{R} \in \mathcal{F}} \left| \mathbb{E}_{\hat{x} \sim D(z)} f(\hat{x}) - \mathbb{E}_{x \sim p} f(x) \right|$$

Trained via best response equilibrium

Performance?