Why AI needs Deep Generative Modeling?

Jakub M. Tomczak

VU VRIJE UNIVERSITEIT AMSTERDAM

What is **intelligence**?

What is **intelligence**?

…

What is **intelligence**?

…

What is **artificial intelligence**?

What is **intelligence**?

…

What is **artificial intelligence**?

What is **intelligence**?

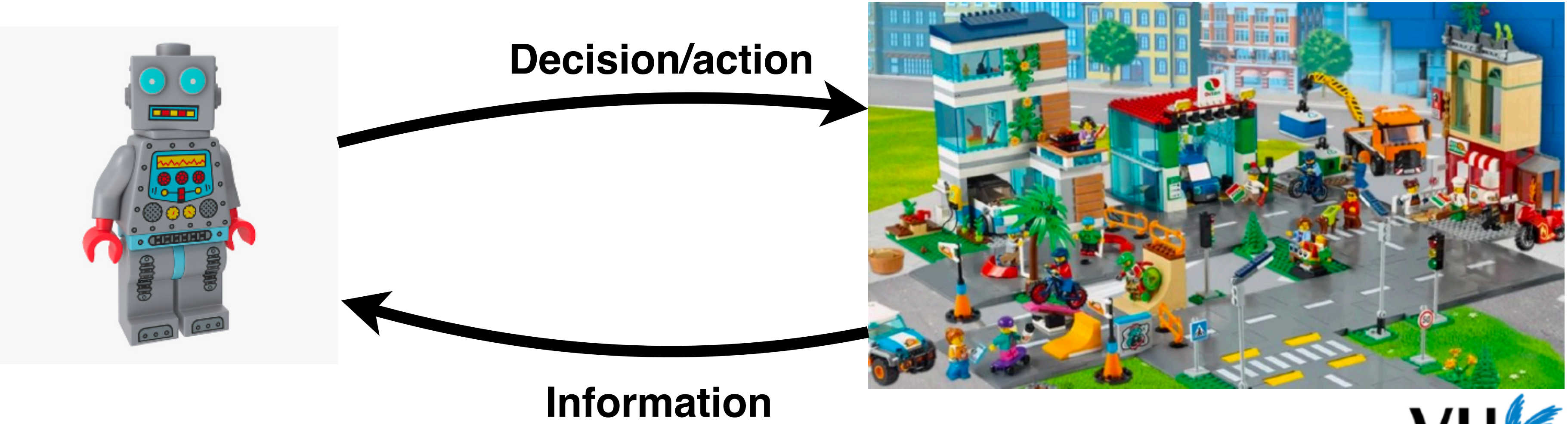
…

What is **artificial intelligence**?

What is **intelligence**?

…

What is **artificial intelligence**?



What is **intelligence**?

…

What is **artificial intelligence**?

Decision/action

Information

0001101010011…

What is **artificial intelligence**?

- **Information** processing
- **Information** storing
- **Information** transmission

What is **artificial intelligence**?

- **Information** processing
- **Information** storing
- **Information** transmission
- **Decision** making

What is **artificial intelligence**?

- **Information** processing
- **Information** storing
- **Information** transmission
- **Decision** making

Learning

Knowledge representation

Models…

What is **artificial intelligence**?

- **Information** processing
- **Information** storing
- **Information** transmission
- **Decision** making

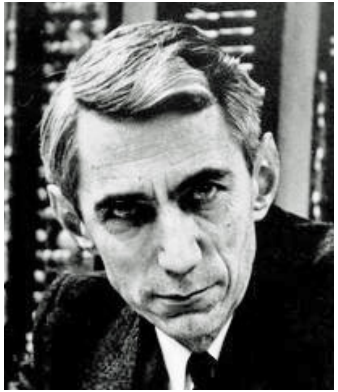
Learning

Knowledge representation

Models…

The question is how to formalize the problem of AI?

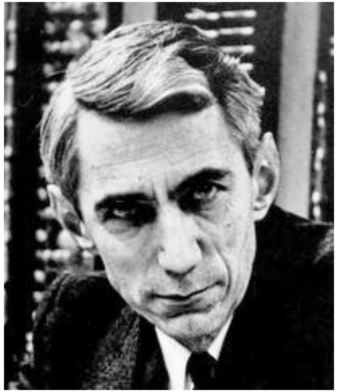
Information (a quick recap)



Claude Shannon

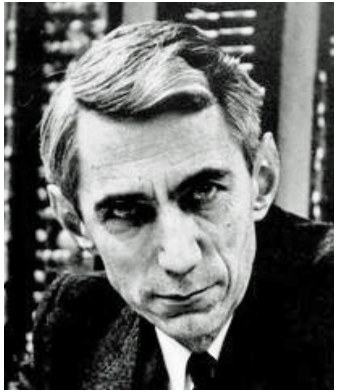
Information (a quick recap)

We have a random source of data x.



Claude Shannon

Information (a quick recap)


Claude Shannon

We have a random source of data x.

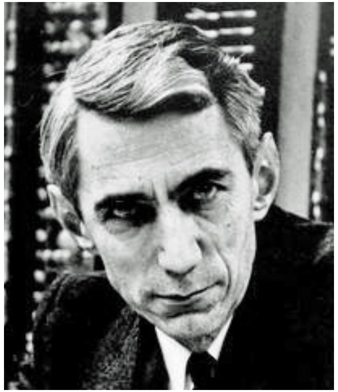
We can quantify the **uncertainty** of this source by calculating **the entropy**:

$$\mathbb{H}[x] = -\sum_{x} p(x) \log p(x)$$

Information (a quick recap)

We have a random source of data x.

We can quantify the **uncertainty** of this source by calculating **the entropy**:



Claude Shannon

$$\mathbb{H}[x] = -\sum_{x} p(x) \log p(x)$$

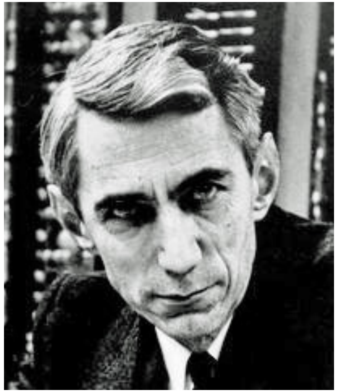
Entropy is max if all x's are equiprobable.

Entropy is min if the probability of one value is 1.

Information (a quick recap)

We have a random source of data x.

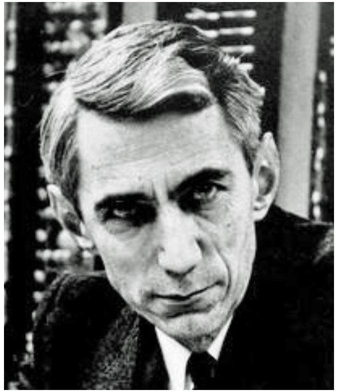
We can quantify the **uncertainty** of this source by calculating **the entropy**:



Claude Shannon

$$\mathbb{H}[x] = -\sum_{x} p(x) \log p(x)$$

Optimal message length ≈ the entropy.



Claude Shannon

Information (a quick recap)

We have two random sources: x and y.

We can quantify the **uncertainty** of them by calculating **the joint entropy**:

$$\mathbb{H}[x, y] = -\sum_{x,y} p(x, y) \log p(x, y)$$

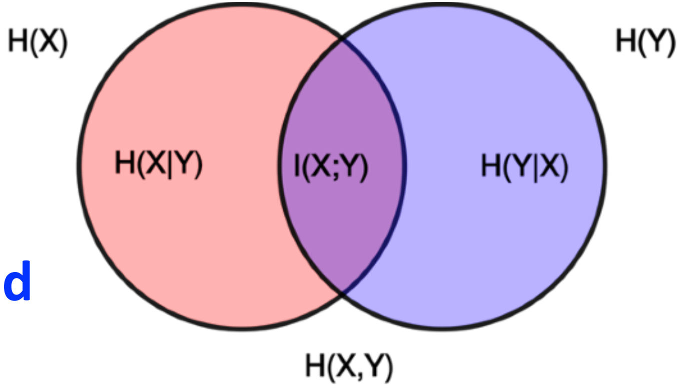
or **the conditional entropy**:

$$\mathbb{H}[y \,|\, x] = -\sum_{x,y} p(x, y) \log p(y \,|\, x)$$

Mutual Information (a quick recap)

We have two random sources: x and y.

Mutual Information (a quick recap)

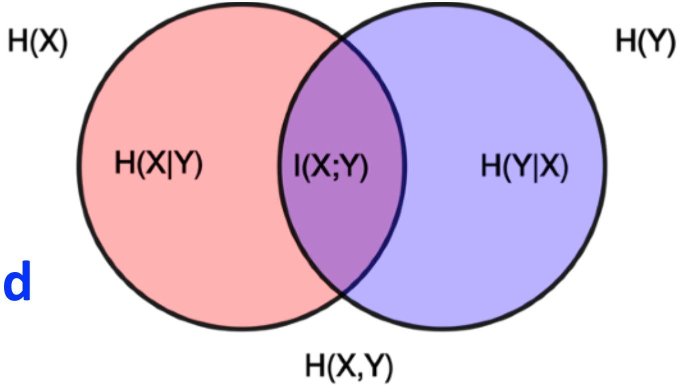


We have two random sources: x and y.

We can quantify how much **information is shared by the two sources:**

$$\mathbb{I}[x; y] = \mathbb{H}[y] - \mathbb{H}[y \,|\, x]$$

Mutual Information (a quick recap)



We have two random sources: x and y.

We can quantify how much **information is shared by the two sources:**

$$\mathbb{I}[x; y] = \mathbb{H}[y] - \mathbb{H}[y \,|\, x]$$

or **how much knowing one source reduces uncertainty about the other.**

We have two random sources: x (e.g., images) and y (e.g., decisions).

We have two random sources: x (e.g., images) and y (e.g., decisions).

We have also a **model** m (a representation of a world).

We have two random sources: x (e.g., images) and y (e.g., decisions).

We have also a **model** m (a representation of a world).

The **goal** of AI is to **maximize** the **mutual information** between (x, y) and m:

$$\mathbb{I}[(x, y); m] = \mathbb{H}[x, y] - \mathbb{H}[x, y \,|\, m]$$

We have two random sources: x (e.g., images) and y (e.g., decisions).

We have also a **model** m (a representation of a world).

The **goal** of AI is to **maximize** the **mutual information** between (x, y) and m:

$$\mathbb{I}[(x, y); m] = \mathbb{H}[x, y] - \mathbb{H}[x, y \,|\, m]$$

Entropy of the world (model has no influence on that)

That's the "real" goal!

The **goal** of AI is to **maximize** the **mutual information** between (x, y) and m

(or minimize $\mathbb{H}[x, y \mid m]$, i.e., minimize uncertainty of the world):

$$\mathbb{H}[x, y \mid m] = \sum_{x,y,m} p(x, y, m)\left[\log p(y \mid x, m) + \log p(x \mid m)\right]$$

The **goal** of AI is to **maximize** the **mutual information** between (x, y) and m

(or minimize $\mathbb{H}[x, y \,|\, m]$, i.e., minimize uncertainty of the world):

$$\mathbb{H}[x, y \,|\, m] = \sum_{x,y,m} p(x, y, m) \left[\log p(y \,|\, x, m) + \log p(x \,|\, m) \right]$$

A model for decision making

A model for understanding the world.

The **goal** of AI is to **maximize** the **mutual information** between (x, y) and m (or minimize $\mathbb{H}[x, y \,|\, m]$, i.e., minimize uncertainty of the world).

In order to achieve that, AI should focus on learning **two models**:

- **A model for decision making:** $p(y \,|\, x, m)$
- **A model for understanding the world:** $p(x \,|\, m)$

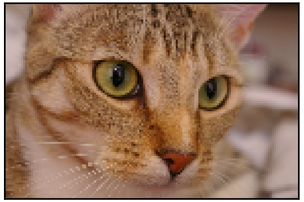
WHAT HAPPENS IF WE LEARN ONLY DECISION MAKING

The bulk of AI is focused on the decision making part **only**!

WHAT HAPPENS IF WE LEARN ONLY DECISION MAKING

The bulk of AI is focused on the decision making part **only**!

Example: Let’s say we have a model that is well trained.

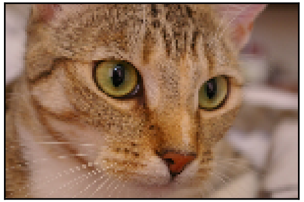


$$p(y = cat|\mathbf{x}) = 0.90$$
$$p(y = dog|\mathbf{x}) = 0.05$$
$$p(y = horse|\mathbf{x}) = 0.05$$

WHAT HAPPENS IF WE LEARN ONLY DECISION MAKING

The bulk of AI is focused on the decision making part **only**!

Example: Let’s say we have a model that is well trained.

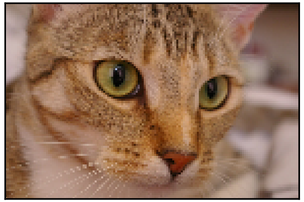


$p(y = cat|\mathbf{x}) = 0.90$
$p(y = dog|\mathbf{x}) = 0.05$
$p(y = horse|\mathbf{x}) = 0.05$

+

noise

=

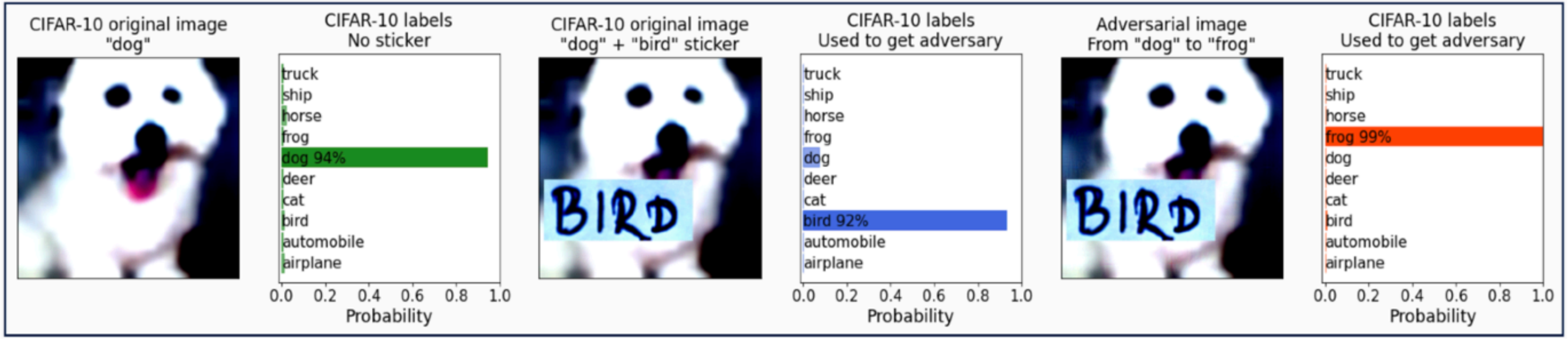


$p(y = cat|\mathbf{x}) = 0.05$
$p(y = dog|\mathbf{x}) = 0.05$
$p(y = horse|\mathbf{x}) = 0.90$

But after adding a little noise it could fail completely…

Let's assume we have a perfectly trained neural net.

What happens if we add (adversarial) noise to an image?

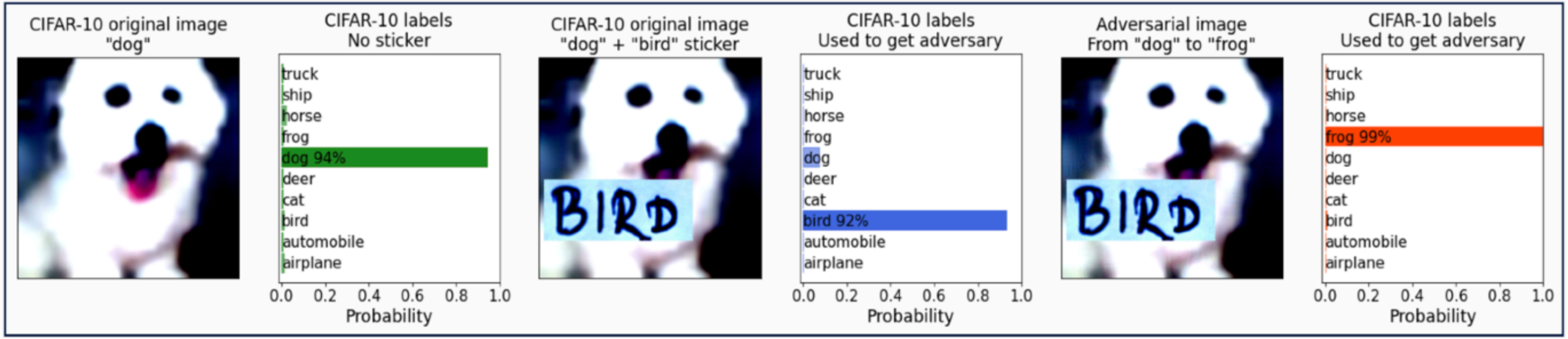


 S. Fort, "Pixels still beat text: Attacking the OpenAI CLIP model with text patches and adversarial pixel perturbations", [Link]

IS LEARNING CLASSIFIERS ENOUGH?

Let’s assume we have a perfectly trained neural net.

What happens if we add (adversarial) noise to an image?



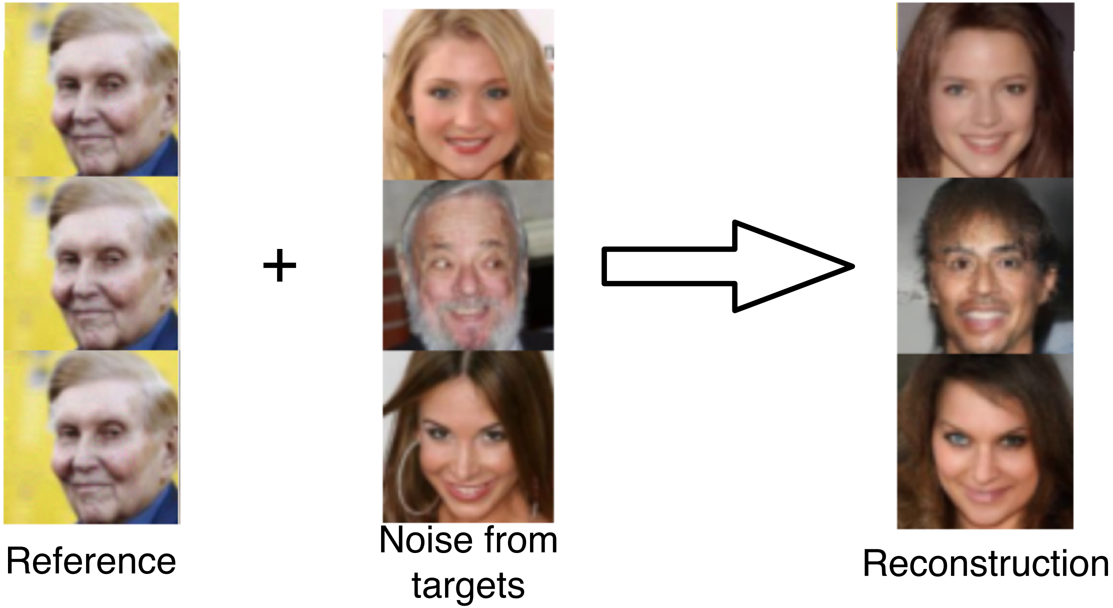
It fails completely…

S. Fort, “Pixels still beat text: Attacking the OpenAI CLIP model with text patches and adversarial pixel perturbations”, [Link]

IS LEARNING CLASSIFIERS ENOUGH?

Let's assume we have a perfectly trained neural net.

What happens if we add (adversarial) noise to an image?



 A. Kuzina, M. Welling, J.M. Tomczak, "Diagnosing Vulnerability of Variational Auto-Encoders to Adversarial Attacks", [Link]

IS LEARNING CLASSIFIERS ENOUGH?

Let’s assume we have a perfectly trained neural net.

What happens if we add (adversarial) noise to an image?

Reference

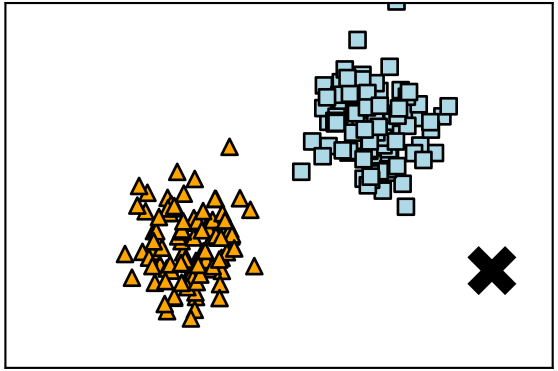
Noise from targets

Reconstruction

It fails completely…

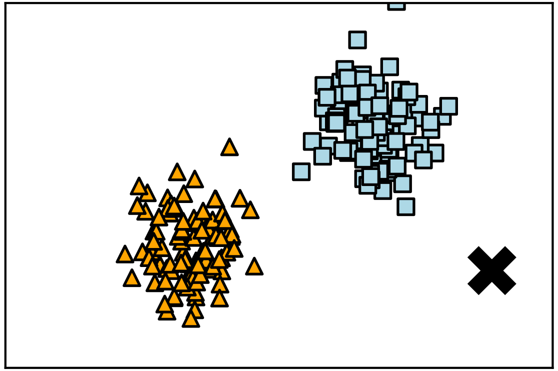
A. Kuzina, M. Welling, J.M. Tomczak, “Diagnosing Vulnerability of Variational Auto-Encoders to Adversarial Attacks”, [Link]

DEEP GENERATIVE MODELING: WHY DO WE NEED THEM?

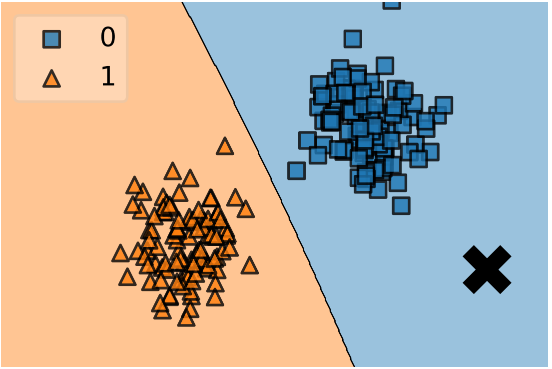


Data

DEEP GENERATIVE MODELING: WHY DO WE NEED THEM?



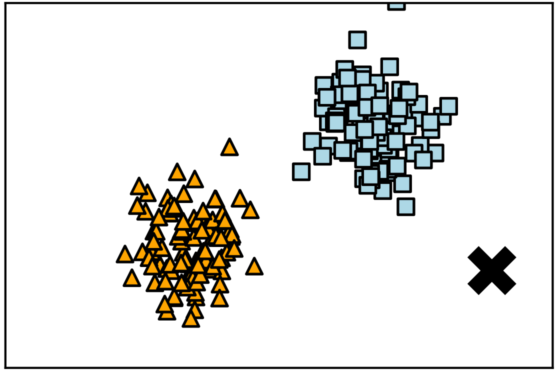
Data



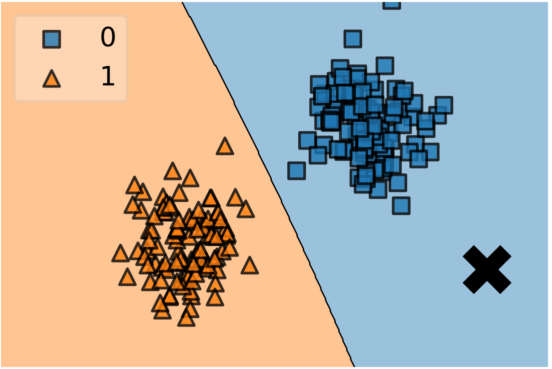
$p(y|\mathbf{x})$

$p(blue|\mathbf{x})$ is high
= certain decision!

DEEP GENERATIVE MODELING: WHY DO WE NEED THEM?

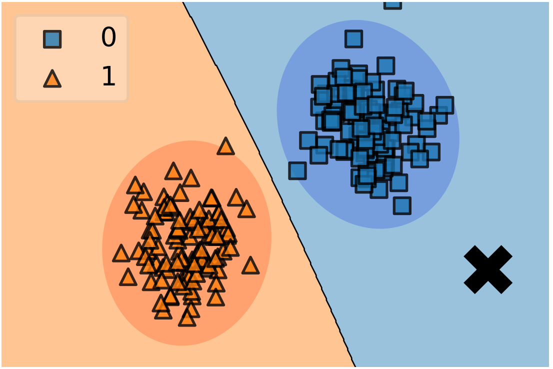


Data



$p(y|\mathbf{x})$

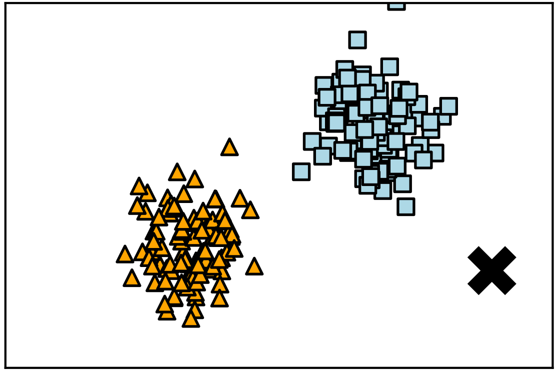
$p(blue|\mathbf{x})$ is high
= certain decision!



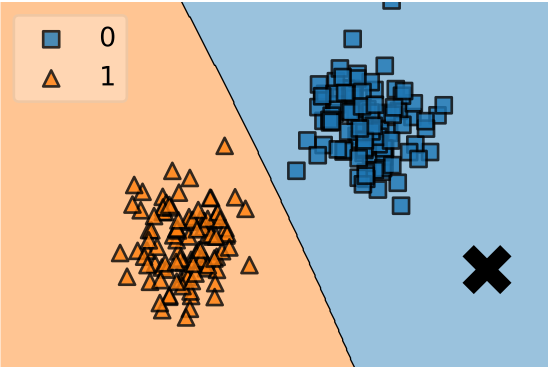
$p(\mathbf{x}, y) = p(y|\mathbf{x})\ p(\mathbf{x})$

$p(blue|\mathbf{x})$ is high
and $p(\mathbf{x})$ is low
= uncertain decision!

DEEP GENERATIVE MODELING: WHY DO WE NEED THEM?

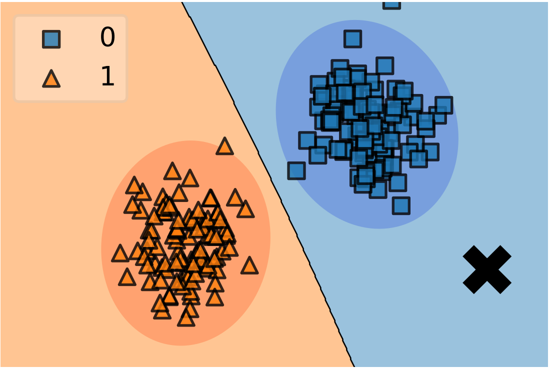


Data



$p(y|\mathbf{x})$

$p(blue|\mathbf{x})$ is high
= certain decision!



$p(\mathbf{x}, y) = p(y|\mathbf{x})\ p(\mathbf{x})$

$p(blue|\mathbf{x})$ is high
and $p(\mathbf{x})$ is low
= uncertain decision!

Thus, learning the conditional is only a part of the story!
How can we learn p(x)?

DEEP GENERATIVE MODELING: WHY DO WE NEED THEM?

We clearly see that training a neural network (i.e., a conditional distribution):

$$p(y \mid \mathbf{x}) = \mathrm{softmax}\left(NN(\mathbf{x})\right)$$

is **not enough**!

Granny Smith	0.1%
iPod	99.7%
library	0.0%
pizza	0.0%
toaster	0.0%
dough	0.0%

DEEP GENERATIVE MODELING: WHY DO WE NEED THEM?

We clearly see that training a neural network (i.e., a conditional distribution):

$$p(y \mid \mathbf{x}) = \mathrm{softmax}\left(NN(\mathbf{x})\right)$$

is **not enough**!

What can we do then?

Or, how to modify the **wrong certainty**?

Granny Smith	0.1%
iPod	99.7%
library	0.0%
pizza	0.0%
toaster	0.0%
dough	0.0%

DEEP GENERATIVE MODELING: HOW WE CAN FORMULATE IT?

Deep Generative Models

- **Autoregressive models** (e.g., PixelCNN)
- **Flow-based models** (e.g., RealNVP)
- **Latent variable models**
  - **Implicit models** (e.g., GANs)
  - **Prescribed models** (e.g., VAEs)

DEEP GENERATIVE MODELING: HOW WE CAN FORMULATE IT?

Deep Generative Models

+ Energy-based models

Autoregressive models
(e.g., PixelCNN)

Flow-based models
(e.g., RealNVP)

Latent variable models

Implicit models
(e.g., GANs)

Prescribed models
(e.g., VAEs)

+ Diffusion models

DEEP GENERATIVE MODELING: HOW WE CAN FORMULATE IT?

Generative models	Training	Likelihood	Sampling	Lossy compression	Lossless compression
Autoregressive models	stable	exact	slow	no	yes
Flow-based models	stable	exact	fast/slow	no	yes
Implicit models	unstable	no	fast	no	no
Prescribed model	stable	approximate	fast	yes	no

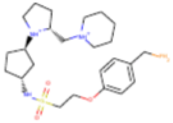
DEEP GENERATIVE MODELING: WHERE CAN WE USE IT?

“ i want to talk to you . ”
“i want to be with you . ”
“i do n’t want to be with you . ”
i do n’t want to be with you .
she did n’t want to be with him .

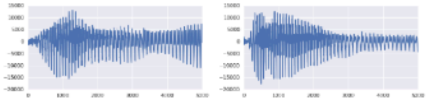
he was silent for a long moment .
he was silent for a moment .
it was quiet for a moment .
it was dark and cold .
there was a pause .
it was my turn .

Text analysis

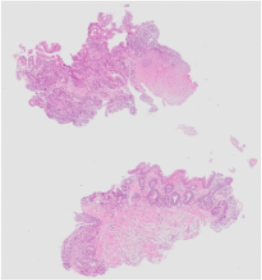
Image analysis



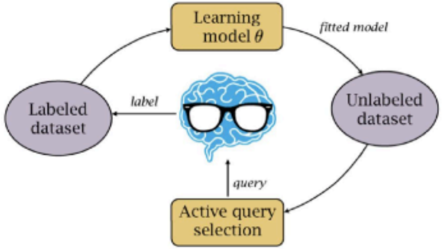
Graph analysis



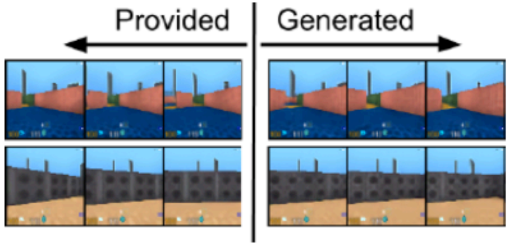
Audio analysis



Medical data



Active Learning



Reinforcement Learning

and more...

DEEP GENERATIVE MODELING: HOW WE CAN FORMULATE IT?

Deep Generative Models

Autoregressive models
(e.g., PixelCNN)

Flow-based models
(e.g., RealNVP)

Latent variable models

Implicit models
(e.g., GANs)

Prescribed models
(e.g., VAEs)

We are interested in modeling

$$p(\mathbf{x})$$

where $\mathbf{x} \in \{0, 1, \ldots, 255\}^{D \times 3}$ is an RGB image (for instance).

We are interested in modeling

$$p(\mathbf{x})$$

where $\mathbf{x} \in \{0,1,\ldots,255\}^{D\times 3}$ is an RGB image (for instance).

We can use the **product rule**:

$$p(\mathbf{x}) = p(x_1) \prod_{d=2}^{D} p(x_d | \mathbf{x}_{<d})$$

where $\mathbf{x}_{<d} = [x_1, x_2, \ldots, x_{d-1}]^\top$

We are interested in modeling

$$p(\mathbf{x})$$

where $\mathbf{x} \in \{0,1,\ldots,255\}^{D\times 3}$ is an RGB image (for instance).

We can use the **product rule**:

$$p(\mathbf{x}) = p(x_1)\prod_{d=2}^{D} p(x_d | \mathbf{x}_{<d})$$

where $\mathbf{x}_{<d} = [x_1, x_2, \ldots, x_{d-1}]^\top$

Example:

$$p(\mathbf{x}) = p(x_1)p(x_2 | x_1)p(x_3 | x_1, x_2)$$

AUTOREGRESSIVE MODELS (ARMS)

We are interested in modeling

$$p(\mathbf{x})$$

where $\mathbf{x} \in \{0,1,\ldots,255\}^{D\times 3}$ is an RGB image (for instance).

We can use the **product rule**:

$$p(\mathbf{x}) = p(x_1)\prod_{d=2}^{D} p(x_d | \mathbf{x}_{<d})$$

where $\mathbf{x}_{<d} = [x_1, x_2, \ldots, x_{d-1}]^{\top}$

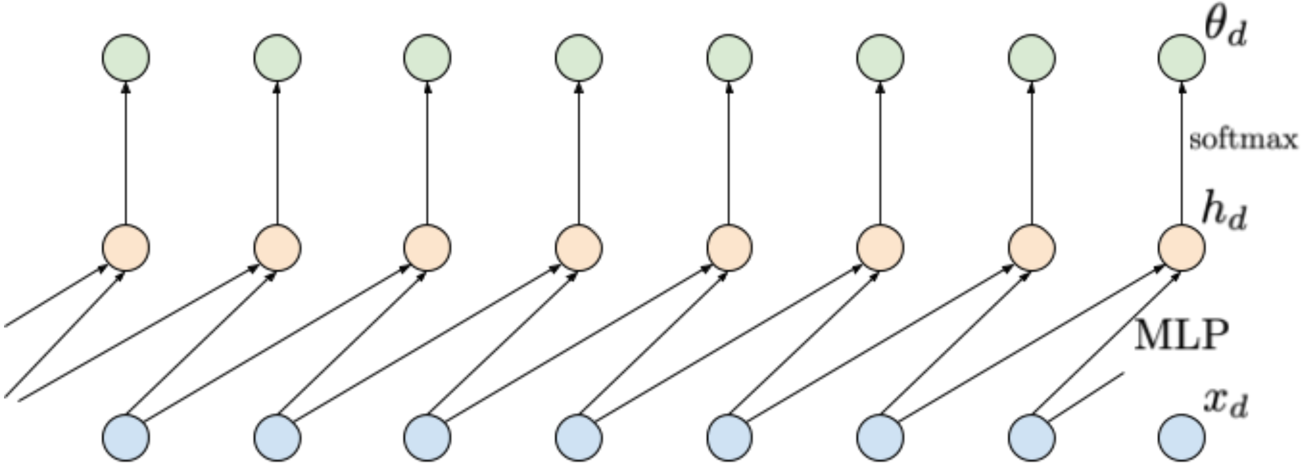
Training objective:

$$\ln p(\mathbf{x}) = \ln p(x_1) + \sum_{d=2}^{D} \ln p(x_d | \mathbf{x}_{<d})$$

Question: How we can model the conditionals $p(x_d | \mathbf{x}_{<d})$ **efficiently**?

Question: How we can model the conditionals $p(x_d | \mathbf{x}_{<d})$ efficiently?

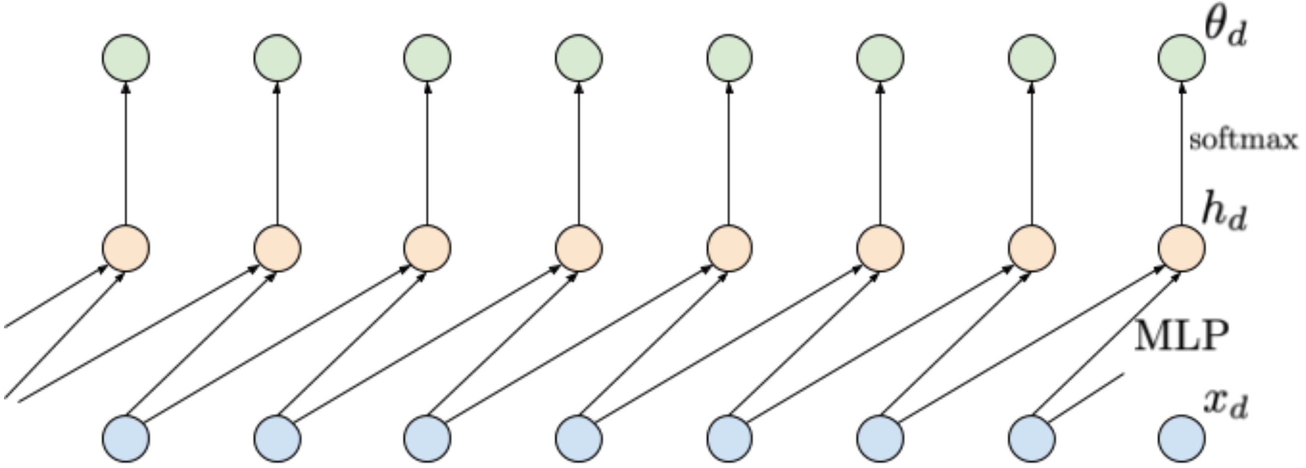
Approach 1: Finite memory



AUTOREGRESSIVE MODELS (ARMS)

Question: How we can model the conditionals $p(x_d \mid \mathbf{x}_{<d})$ efficiently?

Approach 1: Finite memory

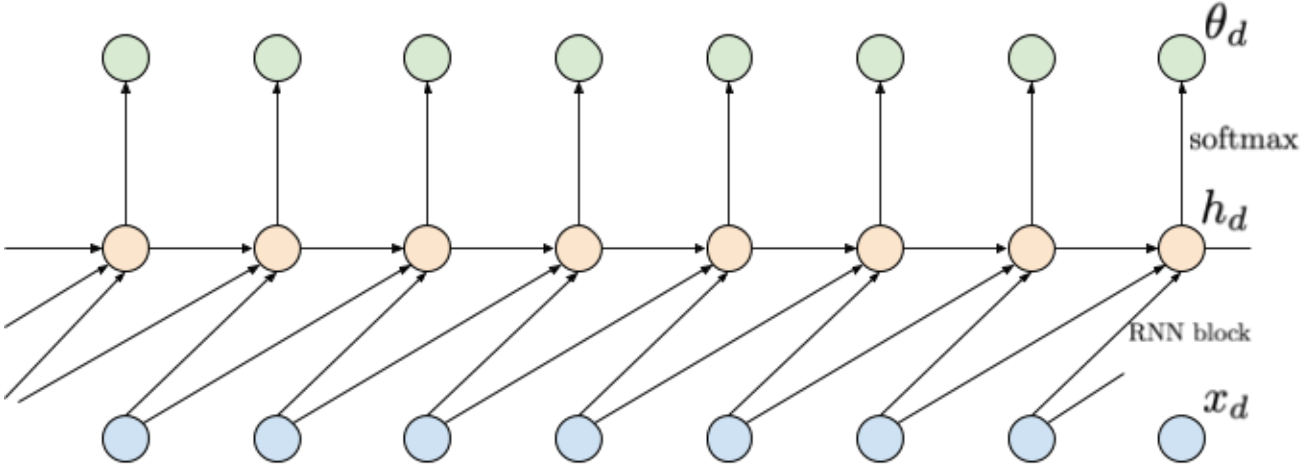


Easy!

Limited dependencies!
How many we should take?

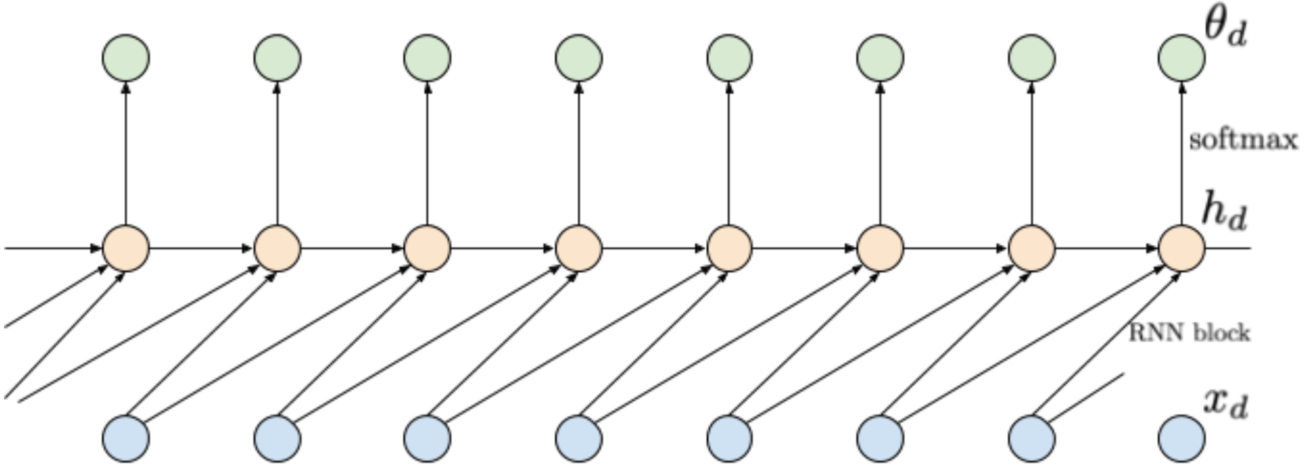
Question: How we can model the conditionals $p(x_d | \mathbf{x}_{<d})$ **efficiently**?

Approach 2: **Long-range memory with RNNs**



Question: How we can model the conditionals $p(x_d | \mathbf{x}_{<d})$ **efficiently**?

Approach 2: **Long-range memory with RNNs**

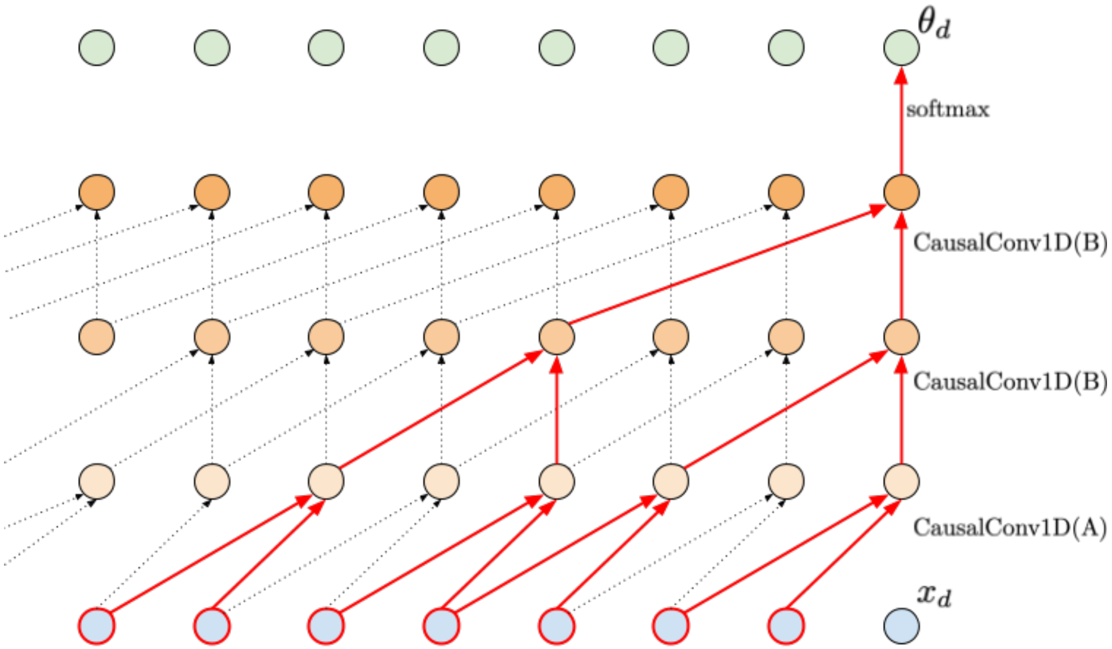


Easy!
Long-range dependencies!

Sequential -> slow
Vanishing gradient problem

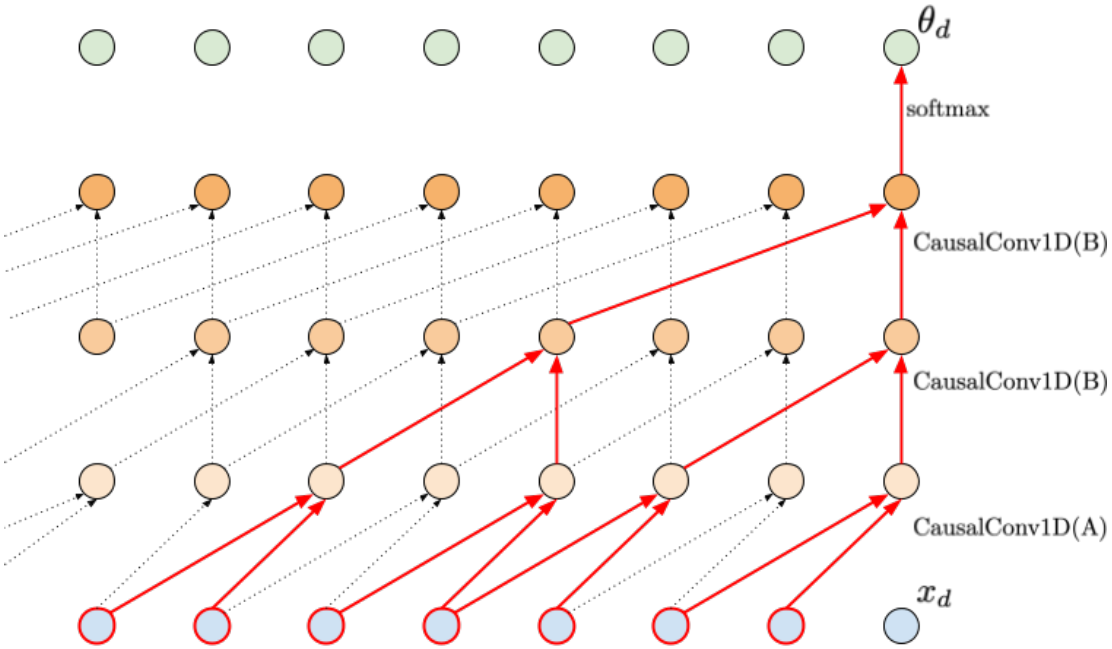
Question: How we can model the conditionals $p(x_d \mid \mathbf{x}_{<d})$ **efficiently**?

Approach 3: **Long-range memory with CNNs**



Question: How we can model the conditionals $p(x_d \mid \mathbf{x}_{<d})$ efficiently?

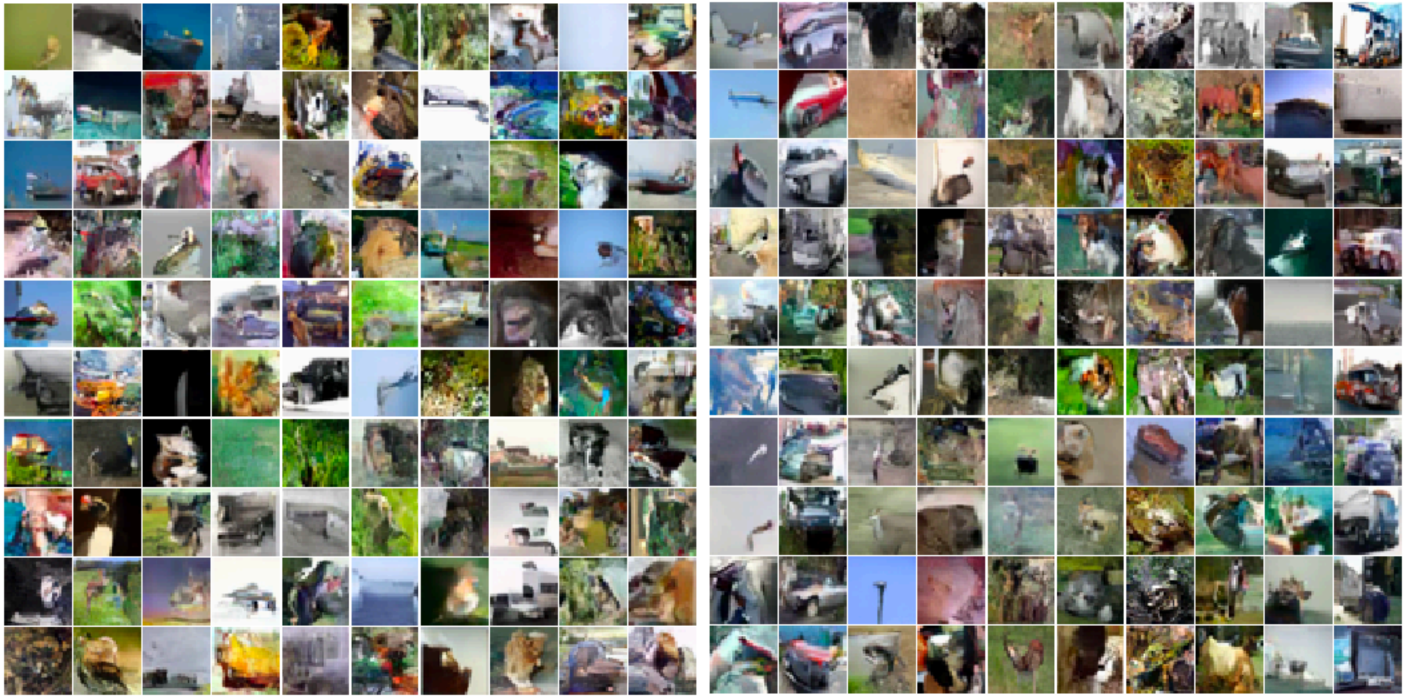
Approach 3: **Long-range memory with CNNs**



Easy!
Long-range dependencies!
No training issues!

Slow generation

AUTOREGRESSIVE MODELS (ARMS)



Samples from a PixelCNN

 Chen, Xi, et al. "Pixelsnail: An improved autoregressive generative model." *ICML 2018*

DEEP GENERATIVE MODELING: HOW WE CAN FORMULATE IT?

Deep Generative Models

- **Autoregressive models** (e.g., PixelCNN)
- **Flow-based models** (e.g., RealNVP)
- **Latent variable models**
  - **Implicit models** (e.g., GANs)
  - **Prescribed models** (e.g., VAEs)

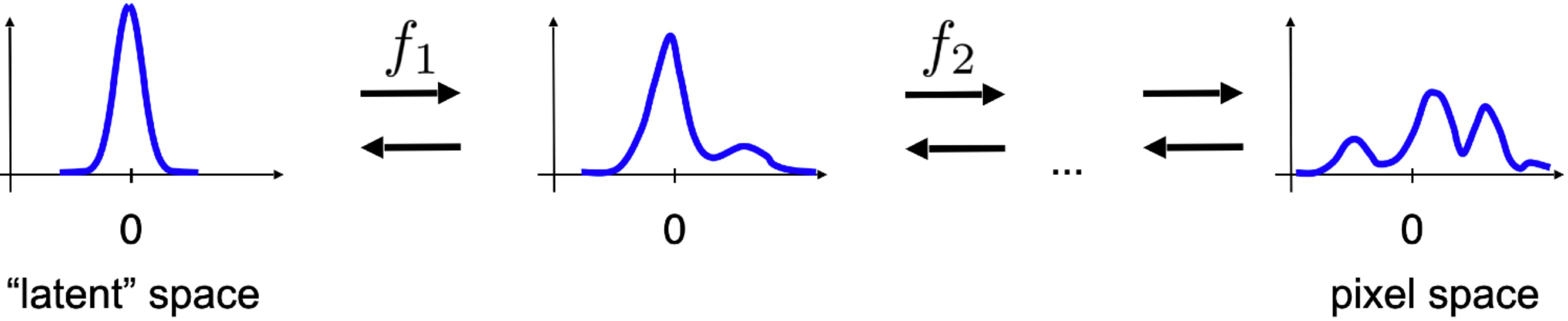
FLOWS (FLOW-BASED MODELS)

We change a random variable **x** to another random variable **z** using **invertible** transformations, $\mathbf{x}, \mathbf{z} \in \mathbb{R}^D$:

$$p(\mathbf{x}) = \pi\left(\mathbf{z}_0 = f^{-1}(\mathbf{x})\right) \prod_{i=1}^{K} \left| \mathbf{J}_{f_i}(z_{i-1}) \right|^{-1}$$

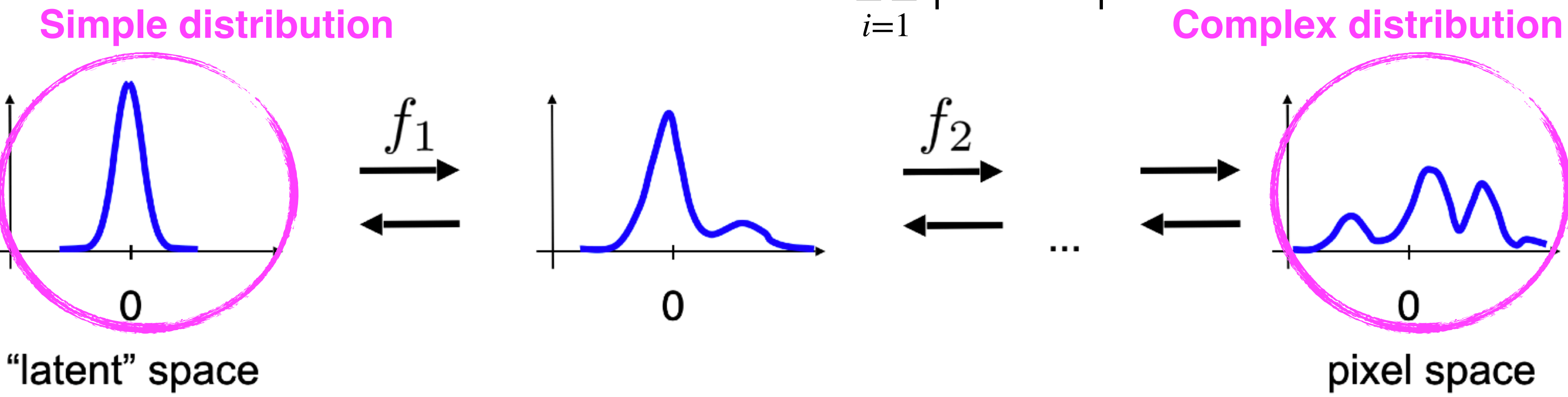
We change a random variable **x** to another random variable **z** using **invertible** transformations, $\mathbf{x}, \mathbf{z} \in \mathbb{R}^D$:

$$p(\mathbf{x}) = \pi\left(\mathbf{z}_0 = f^{-1}(\mathbf{x})\right) \prod_{i=1}^{K} \left| \mathbf{J}_{f_i}(z_{i-1}) \right|^{-1}$$



We change a random variable **x** to another random variable **z** using **invertible** transformations, $\mathbf{x}, \mathbf{z} \in \mathbb{R}^D$:

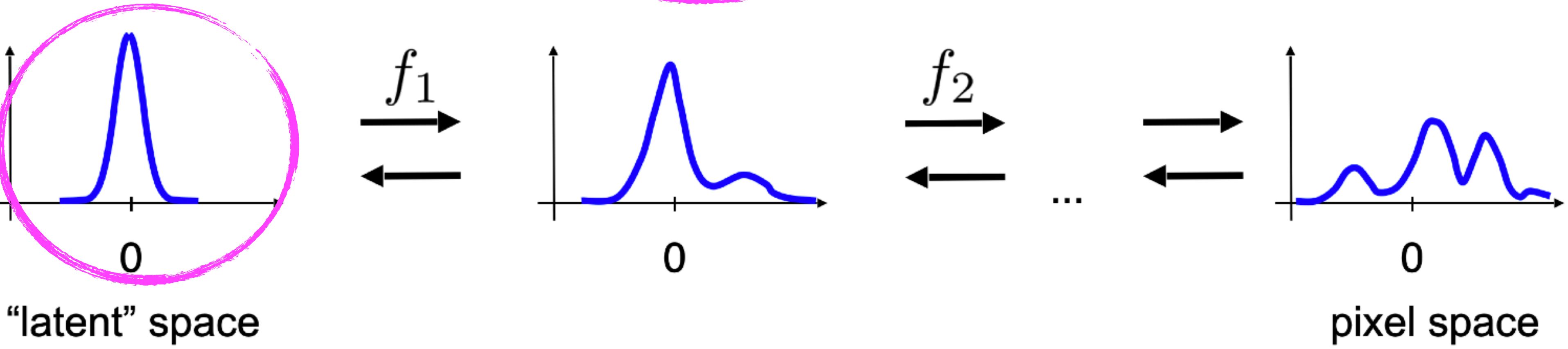
$$p(\mathbf{x}) = \pi\left(\mathbf{z}_0 = f^{-1}(\mathbf{x})\right) \prod_{i=1}^{K} \left| \mathbf{J}_{f_i}(z_{i-1}) \right|^{-1}$$



We change a random variable **x** to another random variable **z** using **invertible** transformations, $\mathbf{x}, \mathbf{z} \in \mathbb{R}^D$:

Known, e.g., Gaussian

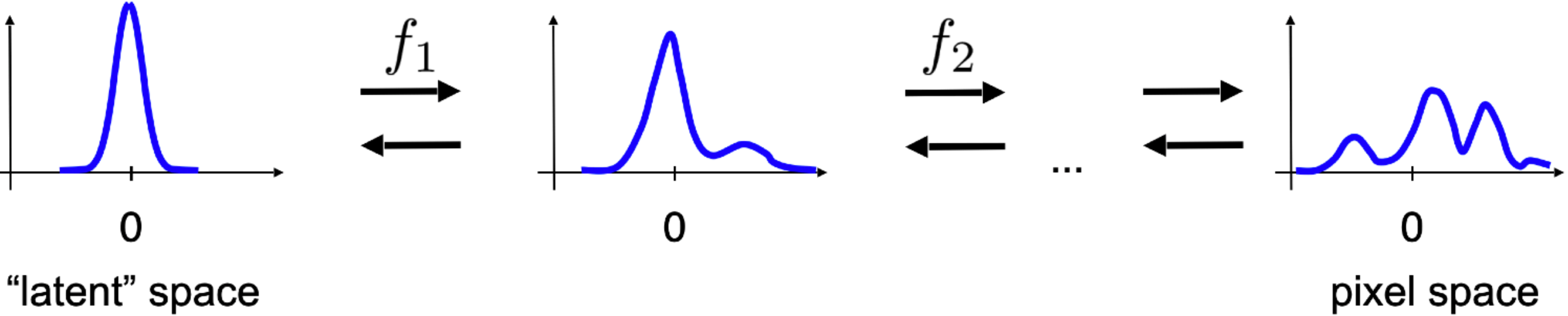
$$p(\mathbf{x}) = \pi\left(\mathbf{z}_0 = f^{-1}(\mathbf{x})\right) \prod_{i=1}^{K} \left| \mathbf{J}_{f_i}(z_{i-1}) \right|^{-1}$$



We change a random variable **x** to another random variable **z** using **invertible** transformations, $\mathbf{x}, \mathbf{z} \in \mathbb{R}^D$:

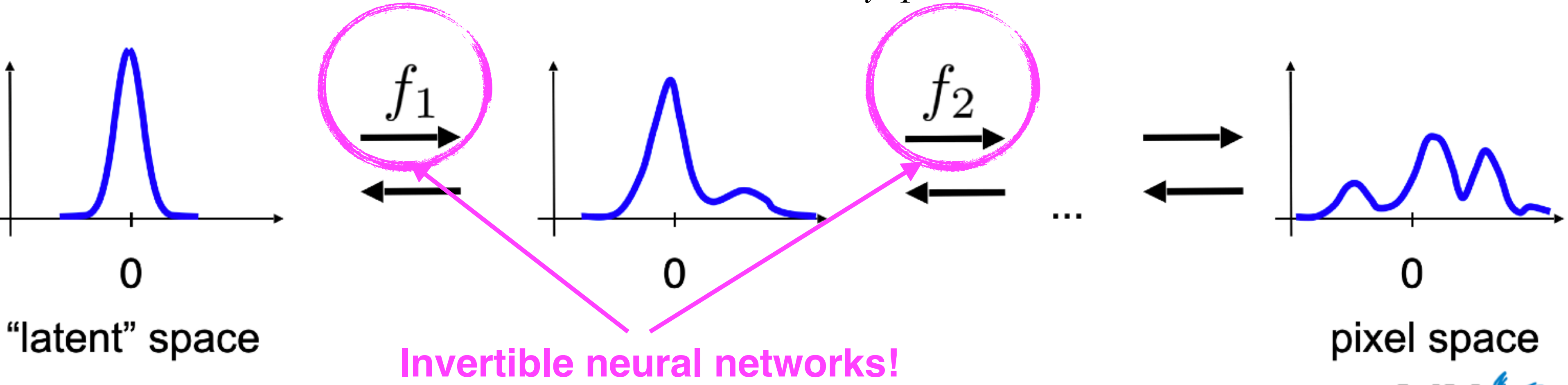
Jacobian must be tractable

$$p(\mathbf{x}) = \pi\left(\mathbf{z}_0 = f^{-1}(\mathbf{x})\right) \prod_{i=1}^{K} \left|\mathbf{J}_{f_i}(z_{i-1})\right|^{-1}$$



We change a random variable **x** to another random variable **z** using **invertible** transformations, $\mathbf{x}, \mathbf{z} \in \mathbb{R}^D$:

$$p(\mathbf{x}) = \pi\left(\mathbf{z}_0 = f^{-1}(\mathbf{x})\right) \prod_{i=1}^{K} \left| \mathbf{J}_{f_i}(z_{i-1}) \right|^{-1}$$



We change a random variable **x** to another random variable **z** using **invertible** transformations, $\mathbf{x}, \mathbf{z} \in \mathbb{R}^D$:

$$p(\mathbf{x}) = \pi\left(\mathbf{z}_0 = f^{-1}(\mathbf{x})\right) \prod_{i=1}^{K} \left| \mathbf{J}_{f_i}(z_{i-1}) \right|^{-1}$$

Training objective:

$$\ln p(\mathbf{x}) = \ln \pi\left(\mathbf{z}_0 = f^{-1}(\mathbf{x})\right) - \sum_{i=1}^{K} \ln \left| \mathbf{J}_{f_i}(z_{i-1}) \right|$$

Two main components

1) **Coupling layer**:

$$\mathbf{y}_a = \mathbf{x}_a$$
$$\mathbf{y}_b = \exp\Big(s\left(\mathbf{x}_a\right)\Big) \odot \mathbf{x}_b + t\left(\mathbf{x}_a\right)$$

is invertible by design:

$$\mathbf{x}_b = \big(\mathbf{y}_b - t(\mathbf{y}_a)\big) \odot \exp\big(-s(\mathbf{y}_a)\big)$$
$$\mathbf{x}_a = \mathbf{y}_a$$

2) **Permutation layer**

Two main components

1) **Coupling layer**:

$$\mathbf{y}_a = \mathbf{x}_a$$
$$\mathbf{y}_b = \exp\Big(s\left(\mathbf{x}_a\right)\Big) \odot \mathbf{x}_b + t\left(\mathbf{x}_a\right)$$

Jacobian is tractable!

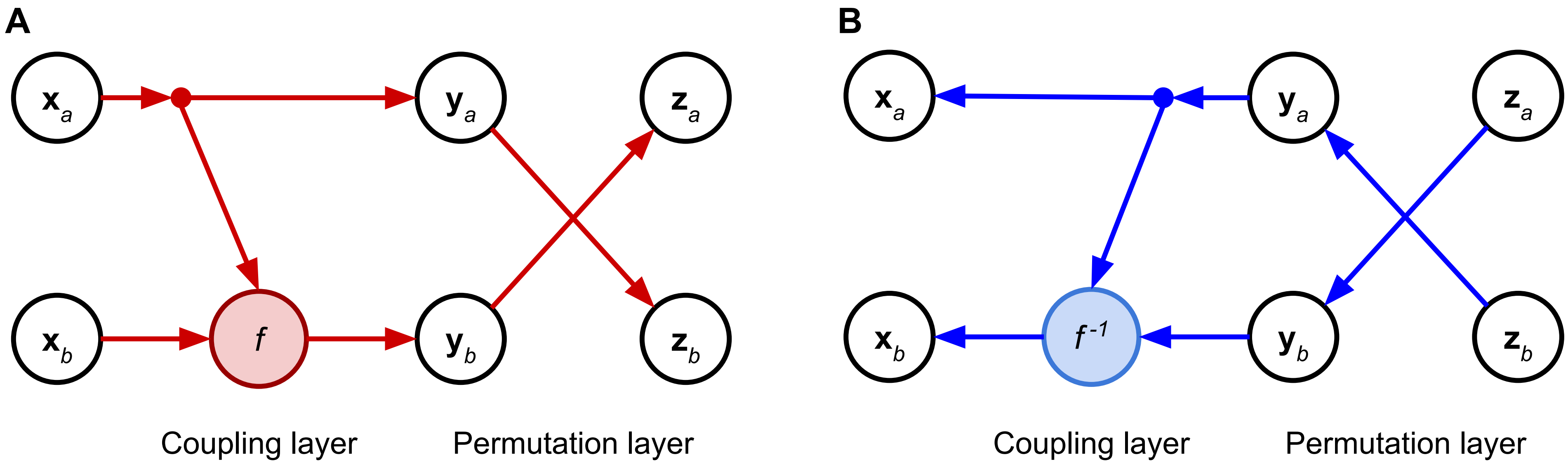
$$\det(\mathbf{J}) = \prod_{j=1}^{D-d} \exp\Big(s\left(\mathbf{x}_a\right)\Big)_j = \exp\left(\sum_{j=1}^{D-d} s\left(\mathbf{x}_a\right)_j\right)$$

is invertible by design:

$$\mathbf{x}_b = \Big(\mathbf{y}_b - t(\mathbf{y}_a)\Big) \odot \exp\Big(-s(\mathbf{y}_a)\Big)$$
$$\mathbf{x}_a = \mathbf{y}_a$$

2) **Permutation layer** $\det(\mathbf{J}) = 1$

FLOWS (FLOW-BASED MODELS): INVERTIBLE LAYERS



A: Forward pass. **B**: Inverse pass.

FLOWS (FLOW-BASED MODELS)

 Kingma, D.P., and Prafulla D. "Glow: Generative flow with invertible 1x1 convolutions." *NeurIPSx 2018*

DEEP GENERATIVE MODELING: HOW WE CAN FORMULATE IT?

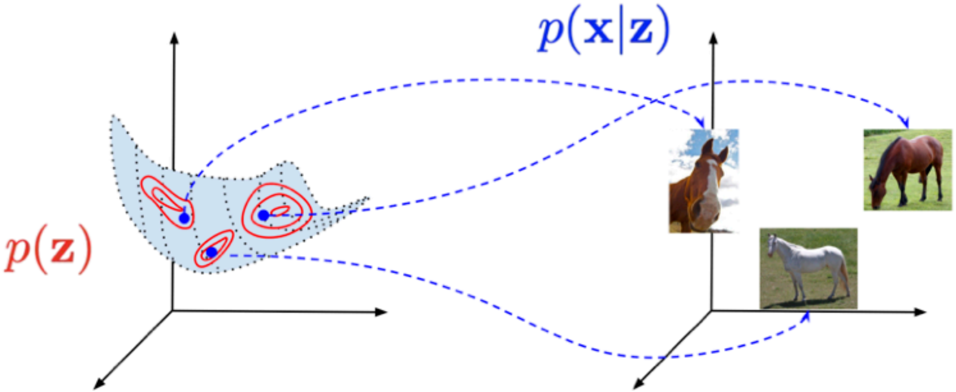
Deep Generative Models

- **Autoregressive models**
  (e.g., PixelCNN)
- **Flow-based models**
  (e.g., RealNVP)
- **Latent variable models**
  - **Implicit models**
    (e.g., GANs)
  - **Prescribed models**
    (e.g., VAEs)

Let's consider a **latent variable model** where we distinguish:

- **latent variables** $\mathbf{z} \in \mathcal{Z}^M$
- **observable variables** $\mathbf{x} \in \mathcal{X}^D$

Latent variables lie on a **low-dimensional manifold**.



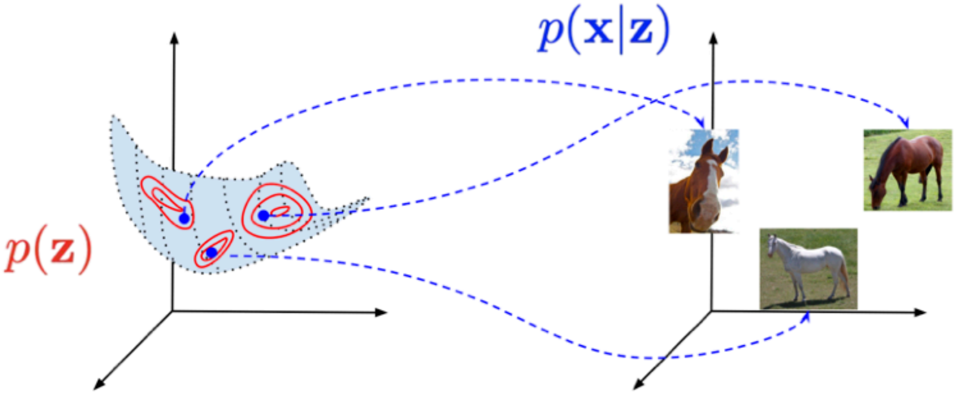
Generative process:

1. $\mathbf{z} \sim p(\mathbf{z})$
2. $\mathbf{x} \sim p(\mathbf{x}|\mathbf{z})$

Let's consider a **latent variable model** where we distinguish:

- **latent variables** $\mathbf{z} \in \mathcal{Z}^M$
- **observable variables** $\mathbf{x} \in \mathcal{X}^D$

Latent variables lie on a **low-dimensional manifold**.



Generative process:

1. $\mathbf{z} \sim p(\mathbf{z})$
2. $\mathbf{x} \sim p(\mathbf{x}\,|\,\mathbf{z})$

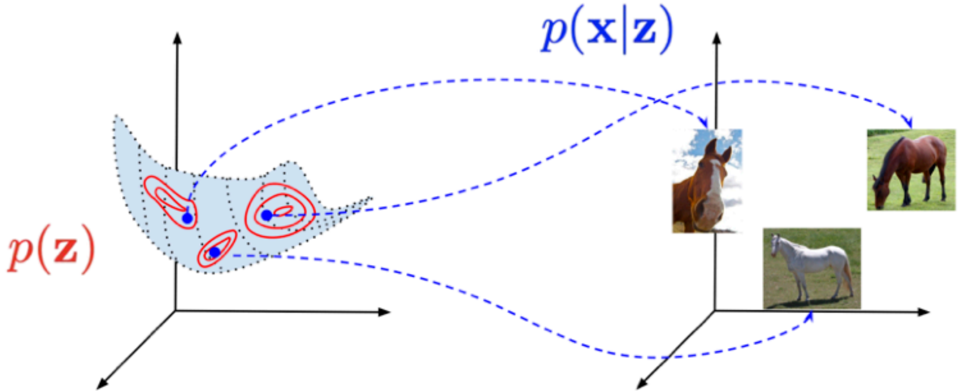
The objective function:

$$\ln p(\mathbf{x}) = \ln \int p(\mathbf{x}\,|\,\mathbf{z}) p(\mathbf{z}) \, \mathrm{d}\mathbf{z}$$

Let's consider a **latent variable model** where we distinguish:

- **latent variables** $\mathbf{z} \in \mathcal{Z}^M$
- **observable variables** $\mathbf{x} \in \mathcal{X}^D$

Latent variables lie on a **low-dimensional manifold**.



Generative process:

1. $\mathbf{z} \sim p(\mathbf{z})$
2. $\mathbf{x} \sim p(\mathbf{x}|\mathbf{z})$

The objective function:

$$\ln p(\mathbf{x}) = \ln \int p(\mathbf{x}|\mathbf{z}) p(\mathbf{z}) \, \mathrm{d}\mathbf{z}$$

The integral is intractable…

$$\ln p(\mathbf{x}) = \ln \int p(\mathbf{x} \,|\, \mathbf{z}) p(\mathbf{z}) \; \mathrm{d}\mathbf{z}$$

$$= \ln \int \frac{q_\phi(\mathbf{z})}{q_\phi(\mathbf{z})} p(\mathbf{x} \,|\, \mathbf{z}) p(\mathbf{z}) \; \mathrm{d}\mathbf{z}$$

$$= \ln \mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z})} \left[\frac{p(\mathbf{x} \,|\, \mathbf{z}) p(\mathbf{z})}{q_\phi(\mathbf{z})} \right]$$

$$\geq \mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z})} \ln \left[\frac{p(\mathbf{x} \,|\, \mathbf{z}) p(\mathbf{z})}{q_\phi(\mathbf{z})} \right]$$

$$= \mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z})} \left[\ln p(\mathbf{x} \,|\, \mathbf{z}) + \ln p(\mathbf{z}) - \ln q_\phi(\mathbf{z}) \right]$$

$$= \mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z})} \left[\ln p(\mathbf{x} \,|\, \mathbf{z}) \right] - \mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z})} \left[\ln q_\phi(\mathbf{z}) - \ln p(\mathbf{z}) \right]$$

$$\ln p(\mathbf{x}) = \ln \int p(\mathbf{x}\,|\,\mathbf{z})p(\mathbf{z})\ \mathrm{d}\mathbf{z}$$

$$= \ln \int \frac{q_\phi(\mathbf{z})}{q_\phi(\mathbf{z})} p(\mathbf{x}\,|\,\mathbf{z})p(\mathbf{z})\ \mathrm{d}\mathbf{z}$$

Variational posteriors

$$= \ln \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})}\left[\frac{p(\mathbf{x}\,|\,\mathbf{z})p(\mathbf{z})}{q_\phi(\mathbf{z})}\right]$$

$$\geq \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})} \ln \left[\frac{p(\mathbf{x}\,|\,\mathbf{z})p(\mathbf{z})}{q_\phi(\mathbf{z})}\right]$$

$$= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})}\left[\ln p(\mathbf{x}\,|\,\mathbf{z}) + \ln p(\mathbf{z}) - \ln q_\phi(\mathbf{z})\right]$$

$$= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})}\left[\ln p(\mathbf{x}\,|\,\mathbf{z})\right] - \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})}\left[\ln q_\phi(\mathbf{z}) - \ln p(\mathbf{z})\right]$$

VARIATIONAL AUTO-ENCODERS

$$
\begin{aligned}
\ln p(\mathbf{x}) &= \ln \int p(\mathbf{x}\,|\,\mathbf{z})p(\mathbf{z})\ \mathrm{d}\mathbf{z} \\
&= \ln \int \frac{q_\phi(\mathbf{z})}{q_\phi(\mathbf{z})} p(\mathbf{x}\,|\,\mathbf{z})p(\mathbf{z})\ \mathrm{d}\mathbf{z} \\
&= \ln \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})}\left[\frac{p(\mathbf{x}\,|\,\mathbf{z})p(\mathbf{z})}{q_\phi(\mathbf{z})}\right] \\
&\geq \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})} \ln \left[\frac{p(\mathbf{x}\,|\,\mathbf{z})p(\mathbf{z})}{q_\phi(\mathbf{z})}\right] \\
&= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})}\left[\ln p(\mathbf{x}\,|\,\mathbf{z}) + \ln p(\mathbf{z}) - \ln q_\phi(\mathbf{z})\right] \\
&= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})}\left[\ln p(\mathbf{x}\,|\,\mathbf{z})\right] - \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})}\left[\ln q_\phi(\mathbf{z}) - \ln p(\mathbf{z})\right]
\end{aligned}
$$

$$
\begin{aligned}
\ln p(\mathbf{x}) &= \ln \int p(\mathbf{x}\,|\,\mathbf{z})p(\mathbf{z})\ \mathrm{d}\mathbf{z} \\
&= \ln \int \frac{q_\phi(\mathbf{z})}{q_\phi(\mathbf{z})} p(\mathbf{x}\,|\,\mathbf{z})p(\mathbf{z})\ \mathrm{d}\mathbf{z} \\
&= \ln \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})} \left[\frac{p(\mathbf{x}\,|\,\mathbf{z})p(\mathbf{z})}{q_\phi(\mathbf{z})} \right] \\
&\geq \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})} \ln \left[\frac{p(\mathbf{x}\,|\,\mathbf{z})p(\mathbf{z})}{q_\phi(\mathbf{z})} \right] \qquad \text{Jensen's inequality} \\
&= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})} \left[\ln p(\mathbf{x}\,|\,\mathbf{z}) + \ln p(\mathbf{z}) - \ln q_\phi(\mathbf{z}) \right] \\
&= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})} \left[\ln p(\mathbf{x}\,|\,\mathbf{z}) \right] - \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})} \left[\ln q_\phi(\mathbf{z}) - \ln p(\mathbf{z}) \right]
\end{aligned}
$$

$$
\begin{aligned}
\ln p(\mathbf{x}) &= \ln \int p(\mathbf{x}\,|\,\mathbf{z})p(\mathbf{z})\ \mathrm{d}\mathbf{z} \\
&= \ln \int \frac{q_\phi(\mathbf{z})}{q_\phi(\mathbf{z})} p(\mathbf{x}\,|\,\mathbf{z})p(\mathbf{z})\ \mathrm{d}\mathbf{z} \\
&= \ln \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})}\left[\frac{p(\mathbf{x}\,|\,\mathbf{z})p(\mathbf{z})}{q_\phi(\mathbf{z})}\right] \\
&\geq \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})} \ln \left[\frac{p(\mathbf{x}\,|\,\mathbf{z})p(\mathbf{z})}{q_\phi(\mathbf{z})}\right] \qquad \text{Jensen's inequality} \\
&= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})}\left[\ln p(\mathbf{x}\,|\,\mathbf{z}) + \ln p(\mathbf{z}) - \ln q_\phi(\mathbf{z})\right] \\
&= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})}\left[\ln p(\mathbf{x}\,|\,\mathbf{z})\right] - \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})}\left[\ln q_\phi(\mathbf{z}) - \ln p(\mathbf{z})\right]
\end{aligned}
$$

$$\begin{aligned}
\ln p(\mathbf{x}) &= \ln \int p(\mathbf{x} \mid \mathbf{z}) p(\mathbf{z}) \; \mathrm{d}\mathbf{z} \\
&= \ln \int \frac{q_\phi(\mathbf{z})}{q_\phi(\mathbf{z})} p(\mathbf{x} \mid \mathbf{z}) p(\mathbf{z}) \; \mathrm{d}\mathbf{z} \\
&= \ln \mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z})} \left[\frac{p(\mathbf{x} \mid \mathbf{z}) p(\mathbf{z})}{q_\phi(\mathbf{z})} \right] \\
&\geq \mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z})} \ln \left[\frac{p(\mathbf{x} \mid \mathbf{z}) p(\mathbf{z})}{q_\phi(\mathbf{z})} \right] \\
&= \mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z})} \left[\ln p(\mathbf{x} \mid \mathbf{z}) + \ln p(\mathbf{z}) - \ln q_\phi(\mathbf{z}) \right] \\
&= \mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z})} \left[\ln p(\mathbf{x} \mid \mathbf{z}) \right] - \mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z})} \left[\ln q_\phi(\mathbf{z}) - \ln p(\mathbf{z}) \right]
\end{aligned}$$

$$
\begin{aligned}
\ln p(\mathbf{x}) &= \ln \int p(\mathbf{x}\,|\,\mathbf{z})p(\mathbf{z})\ \mathrm{d}\mathbf{z} \\
&= \ln \int \frac{q_\phi(\mathbf{z})}{q_\phi(\mathbf{z})} p(\mathbf{x}\,|\,\mathbf{z})p(\mathbf{z})\ \mathrm{d}\mathbf{z} \\
&= \ln \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})} \left[\frac{p(\mathbf{x}\,|\,\mathbf{z})p(\mathbf{z})}{q_\phi(\mathbf{z})} \right] \\
&\geq \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})} \ln \left[\frac{p(\mathbf{x}\,|\,\mathbf{z})p(\mathbf{z})}{q_\phi(\mathbf{z})} \right] \\
&= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})} \left[\ln p(\mathbf{x}\,|\,\mathbf{z}) + \ln p(\mathbf{z}) - \ln q_\phi(\mathbf{z}) \right] \\
&= \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})} \left[\ln p(\mathbf{x}\,|\,\mathbf{z}) \right] - \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z})} \left[\ln q_\phi(\mathbf{z}) - \ln p(\mathbf{z}) \right]
\end{aligned}
$$

$$\ln p(\mathbf{x}) = \ln \int p(\mathbf{x}\,|\,\mathbf{z}) p(\mathbf{z}) \, \mathrm{d}\mathbf{z}$$

$$= \ln \int \frac{q_\phi(\mathbf{z})}{q_\phi(\mathbf{z})} p(\mathbf{x}\,|\,\mathbf{z}) p(\mathbf{z}) \, \mathrm{d}\mathbf{z}$$

$$= \ln \mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z})} \left[\frac{p(\mathbf{x}\,|\,\mathbf{z}) p(\mathbf{z})}{q_\phi(\mathbf{z})} \right]$$

$$\geq \mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z})} \ln \left[\frac{p(\mathbf{x}\,|\,\mathbf{z}) p(\mathbf{z})}{q_\phi(\mathbf{z})} \right]$$

Reconstruction error

“Regularization” term

$$= \mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z})} \left[\ln p(\mathbf{x}\,|\,\mathbf{z}) + \ln p(\mathbf{z}) - \ln q_\phi(\mathbf{z}) \right]$$

$$= \mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z})} \left[\ln p(\mathbf{x}\,|\,\mathbf{z}) \right] - \mathbb{E}_{\mathbf{z} \sim q_\phi(\mathbf{z})} \left[\ln q_\phi(\mathbf{z}) - \ln p(\mathbf{z}) \right]$$

$$\ln p(\mathbf{x}) \geq \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\ln p(\mathbf{x}\,|\,\mathbf{z})\right] - \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\ln q_\phi(\mathbf{z}\,|\,\mathbf{x}) - \ln p(\mathbf{z})\right]$$

ELBO: Evidence Lower Bound

 Kingma, D.P., and Welling, M.. "Auto-encoding variational bayes." *ICLR 2014*

$$\ln p(\mathbf{x}) \geq \mathbb{E}_{\mathbf{z}\sim q_{\phi}(\mathbf{z}|\mathbf{x})}\left[\ln p(\mathbf{x}\,|\,\mathbf{z})\right] - \mathbb{E}_{\mathbf{z}\sim q_{\phi}(\mathbf{z}|\mathbf{x})}\left[\ln q_{\phi}(\mathbf{z}\,|\,\mathbf{x}) - \ln p(\mathbf{z})\right]$$

We consider **amortized inference**: $q_{\phi}(\mathbf{z}\,|\,\mathbf{x})$

In other words, a single parameterization for each new input **x**.

 Kingma, D.P., and Welling, M.. "Auto-encoding variational bayes." *ICLR 2014*

VARIATIONAL AUTO-ENCODERS

$$\ln p(\mathbf{x}) \geq \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\ln p(\mathbf{x}\,|\,\mathbf{z})\right] - \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}\left[\ln q_\phi(\mathbf{z}\,|\,\mathbf{x}) - \ln p(\mathbf{z})\right]$$

We consider **amortized inference**: $q_\phi(\mathbf{z}\,|\,\mathbf{x})$

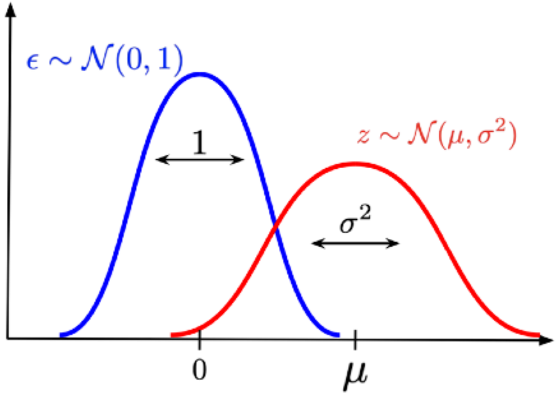
In other words, a single parameterization for each new input **x**.

Moreover, we use **reparameterization trick**:

Every Gaussian variable could be defined as:

$z = \mu + \sigma \cdot \varepsilon$

where $\varepsilon \sim \mathcal{N}(0,1)$



 Kingma, D.P., and Welling, M.. "Auto-encoding variational bayes." *ICLR 2014*

$$\ln p(\mathbf{x}) \geq \mathbb{E}_{\mathbf{z}\sim q_{\phi}(\mathbf{z}|\mathbf{x})}\left[\ln p(\mathbf{x}\,|\,\mathbf{z})\right] - \mathbb{E}_{\mathbf{z}\sim q_{\phi}(\mathbf{z}|\mathbf{x})}\left[\ln q_{\phi}(\mathbf{z}\,|\,\mathbf{x}) - \ln p(\mathbf{z})\right]$$

We consider **amortized inference**: $q_{\phi}(\mathbf{z}\,|\,\mathbf{x})$

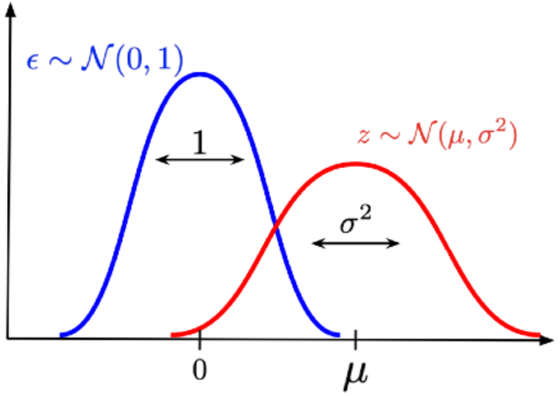
In other words, a single parameterization for each new input **x**.

Moreover, we use **reparameterization trick**:

It reduces the variance of the gradients.

It allows to get randomness outside **z**.

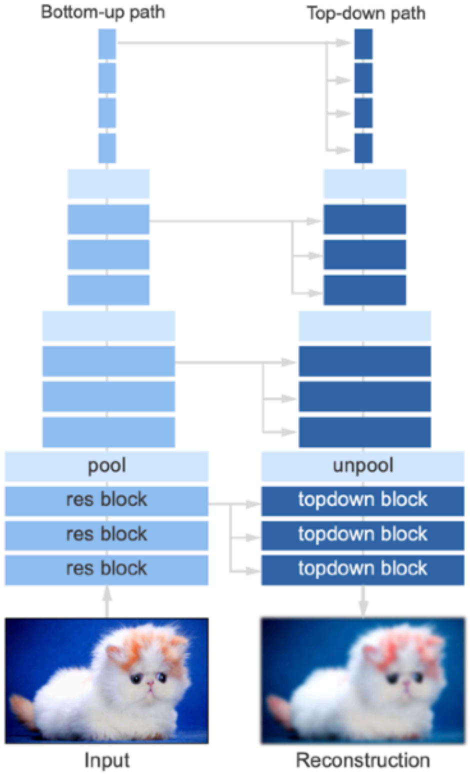
$$z = \mu + \sigma \cdot \varepsilon$$



 Kingma, D.P., and Welling, M.. "Auto-encoding variational bayes." *ICLR 2014*

VARIATIONAL AUTO-ENCODERS

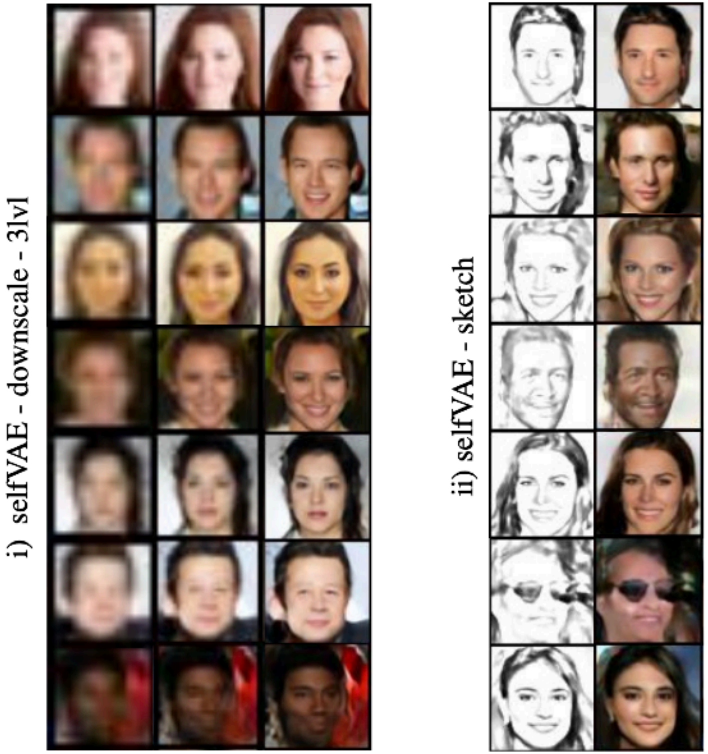
Generations



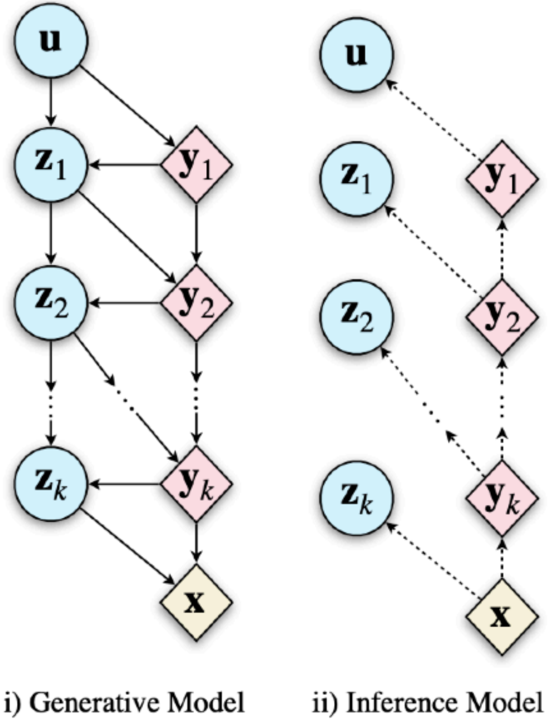
Very Deep VAE

 Child, R. "Very Deep VAEs Generalize Autoregressive Models and Can Outperform Them on Images." *ICLR 2021*

VARIATIONAL AUTO-ENCODERS



Generations



Hierarchical VAE

 Gatopoulos, I., and Tomczak, J.M., "Self-Supervised Variational Auto-Encoders." *arXiv preprint arXiv:2010.02014* (2020).

CONCLUSION

- Here: **the likelihood-based generative models**.

CONCLUSION

- Here: **the likelihood-based generative models**.
- We **skipped** Generative Adversarial Nets & others.

CONCLUSION

- Here: **the likelihood-based generative models**.
- We **skipped** Generative Adversarial Nets & others.
- Why generative modeling?

$$p(\mathbf{x}, y) = p(y \mid \mathbf{x})\, p(\mathbf{x})$$

CONCLUSION

- Here: **the likelihood-based generative models**.
- We **skipped** Generative Adversarial Nets & others.
- Why generative modeling?

$$p(\mathbf{x}, y) = p(y \mid \mathbf{x})\, p(\mathbf{x})$$

- Important directions:
  - ➡ Better uncertainty quantification
  - ➡ New parameterization (new neural networks)
  - ➡ Out-of-Distribution
  - ➡ Continual learning

BLOG ABOUT DEEP GENERATIVE MODELING

If you are interested in going deeper into deep generative modeling, please take a look at my blog: **[Blog]**

- **Intro**: [Link]
- **ARMs**: [Link]
- **Flows**: [Link], [Link]
- **VAEs**: [Link], [Link]
- **Hybrid modeling**: [Link]

THANK YOU FOR YOUR ATTENTION

Jakub M. Tomczak
Computational Intelligence group
Vrije Universiteit Amsterdam

Webpage: https://jmtomczak.github.io/

Github: https://github.com/jmtomczak

Twitter: https://twitter.com/jmtomczak