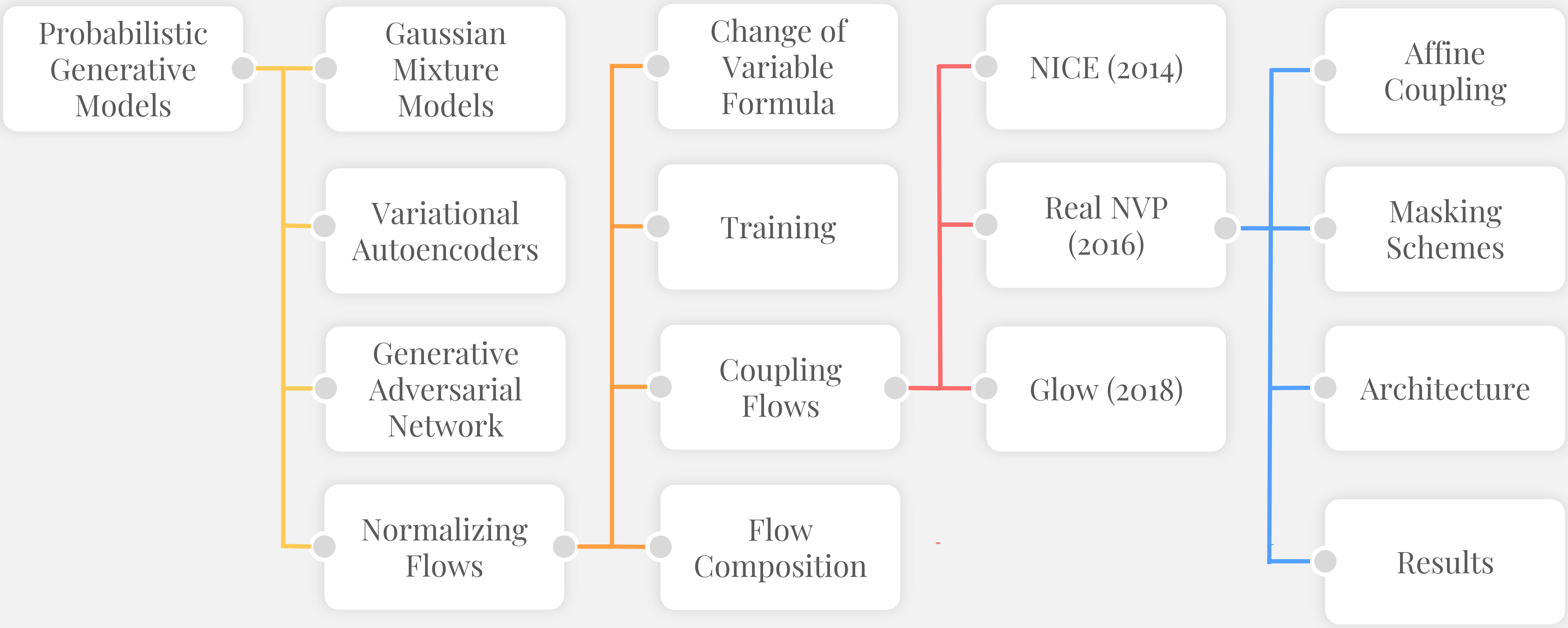
Density Estimation Using Real NVP

Laurent Dinh, et al. · 2016

By Flavio Schneider · March 2021 · ETH zürich

Map



Probabilistic Generative Models
Gaussian Mixture Models
Variational Autoencoders
Generative Adversarial Network
Normalizing Flows
Change of Variable Formula
Training
Coupling Flows
Flow Composition
NICE (2014)
Real NVP (2016)
Glow (2018)
Affine Coupling
Masking Schemes
Architecture
Results

Probabilistic Generative Model

Def 1. A **probabilistic generative model (PGM)** is a model that attempts to estimate the probability distribution $p_{\boldsymbol{X}}(\boldsymbol{x})$ parametrized by $\boldsymbol{\theta}$ over a random variable $\boldsymbol{X}$ given a dataset $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ with $\boldsymbol{x}_i \in \mathbb{R}^D$.

Probabilistic Generative Model

Def 1. A **probabilistic generative model (PGM)** is a model that attempts to estimate the probability distribution $p_X(\boldsymbol{x})$ parametrized by $\boldsymbol{\theta}$ over a random variable $\boldsymbol{X}$ given a dataset $\mathcal{D} = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_N\}$ with $\boldsymbol{x}_i \in \mathbb{R}^D$.

Density Estimation

$p_X(\boldsymbol{x})$

$$\boldsymbol{x} \in \mathbb{R}^D \rightarrow \boxed{\boldsymbol{\theta}} \rightarrow p \in [0,1]$$

$$\mathcal{D}$$

Sampling

$\boldsymbol{x} \sim p_X(\cdot)$

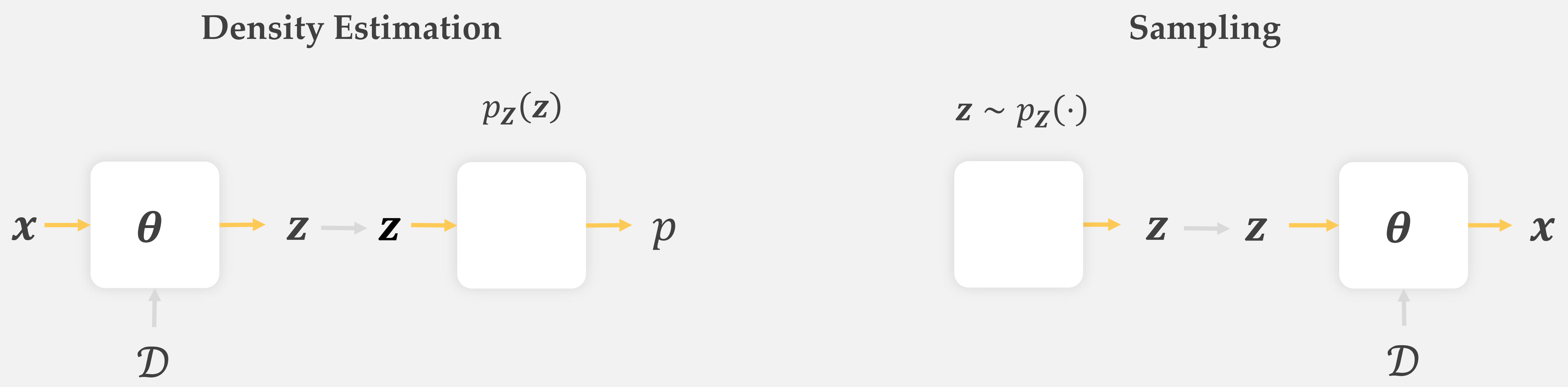
$$\boxed{\boldsymbol{\theta}} \rightarrow \boldsymbol{x} \in \mathbb{R}^D$$

$$\mathcal{D}$$

Probabilistic Generative Model

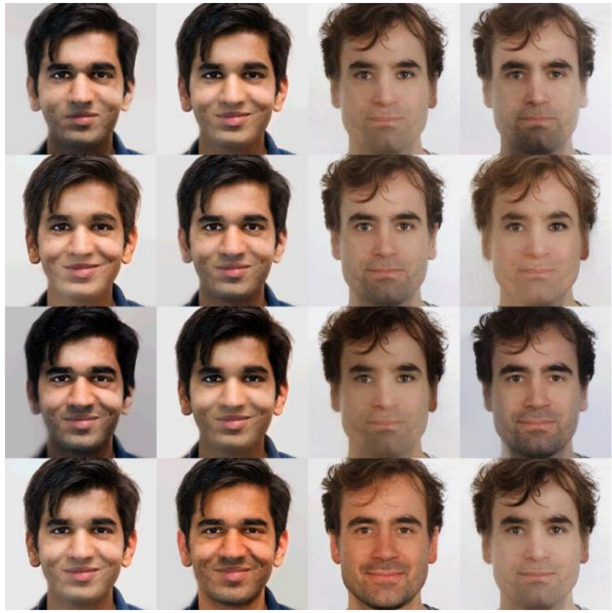
Definition · 2 Models · **Adding Latent $\boldsymbol{z}$** · Uses

Add an additional variable $\boldsymbol{z} \in \mathbb{R}^{D'}$ with some known simple distribution $p_Z(\boldsymbol{z})$ (e.g. $p_Z(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}|\boldsymbol{0}, \boldsymbol{I})$) which is easy to evaluate and sample from.



Probabilistic Generative Model

- Generate data (e.g., images) like the samples in $\mathcal{D}$
- Interpolate between samples in a meaningful way
- Unsupervised clustering
- Denoising
- Data reconstruction
- Build a prior $p_{\boldsymbol{X}}(\boldsymbol{x})$ to some other model (Bayesian setting)



(Gaussian) Mixture Model

- Used mostly for unsupervised clustering.
- Parameters $\boldsymbol{\theta} = \{\boldsymbol{\pi}, \boldsymbol{\mu}_{\boldsymbol{z}_1,\dots,\boldsymbol{z}_k}, \boldsymbol{\Sigma}_{\boldsymbol{z}_1,\dots,\boldsymbol{z}_k}\}$ estimated with EM algorithm.

\+ Density estimation is easy: $p_X(\boldsymbol{x}) = \sum_{\boldsymbol{z}} p_{X|Z}(\boldsymbol{x}|\boldsymbol{z})p_Z(\boldsymbol{z})$

\+ Sampling is easy: $\boldsymbol{z} \sim p_Z(\cdot)$ then $\boldsymbol{x} \sim p_{X|Z}(\cdot\,|\boldsymbol{z})$

\- EM requires initial guess for $\boldsymbol{\theta}$.

\- Scales poorly with large D.

\- Number of clusters k is fixed.

$k \in \mathbb{N}$: number of clusters

$\boldsymbol{x} \in \mathbb{R}^D$: data point

$\boldsymbol{z} \in \{0,1\}^k$: cluster membership

$$p_{Z|X}(\boldsymbol{z}|\boldsymbol{x}) = \frac{p_{X|Z}(\boldsymbol{x}|\boldsymbol{z})p_Z(\boldsymbol{z})}{p_X(\boldsymbol{x})}$$

- $p_{X|Z}(\boldsymbol{x}|\boldsymbol{z}) = \boldsymbol{\mathcal{N}}(\boldsymbol{x}|\boldsymbol{\mu}_{\boldsymbol{z}}, \boldsymbol{\Sigma}_{\boldsymbol{z}}))$
- $p_Z(\boldsymbol{z}) = \boldsymbol{Cat}(\boldsymbol{z}|\pi_1, \dots, \pi_k)$

Variational Autoencoder

- Used for compression, data generation/reconstruction/denoising.
- Parameters $\boldsymbol{\theta}$ estimated by optimizing ELBO.

± Density estimation is approximate: $p_X(\boldsymbol{x}) \approx \frac{p_{X|Z}(\boldsymbol{x}|\boldsymbol{z})p_Z(\boldsymbol{z})}{q_{Z|X}(\boldsymbol{z}|\boldsymbol{x})}$

+ Sampling is easy: $\boldsymbol{z} \sim p_Z(\cdot)$ then $\boldsymbol{x} \sim p_{X|Z}(\cdot\,|\boldsymbol{z})$

$\boldsymbol{x} \in \mathbb{R}^D$: data point

$\boldsymbol{z} \in \mathbb{R}^{D'}$: latent data point

$$p_{Z|X}(\boldsymbol{z}|\boldsymbol{x}) = \frac{p_{X|Z}(\boldsymbol{x}|\boldsymbol{z})p_Z(\boldsymbol{z})}{p_X(\boldsymbol{x})}$$

- $p_Z(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{z}|\boldsymbol{0}, \boldsymbol{I})$

$$q_{Z|X}(\boldsymbol{z}|\boldsymbol{x}) \approx p_{Z|X}(\boldsymbol{z}|\boldsymbol{x})$$

Generative Adversarial Networks

- Used mainly for data generation.
- Parameters $\boldsymbol{\theta}$ estimated by playing the minimax game.

\+ Sampling is easy (built for this): $\boldsymbol{z} \sim p_{\boldsymbol{Z}}(\cdot)$ then $\boldsymbol{x} = \boldsymbol{G}(\boldsymbol{z}|\boldsymbol{\theta}_{\boldsymbol{G}})$

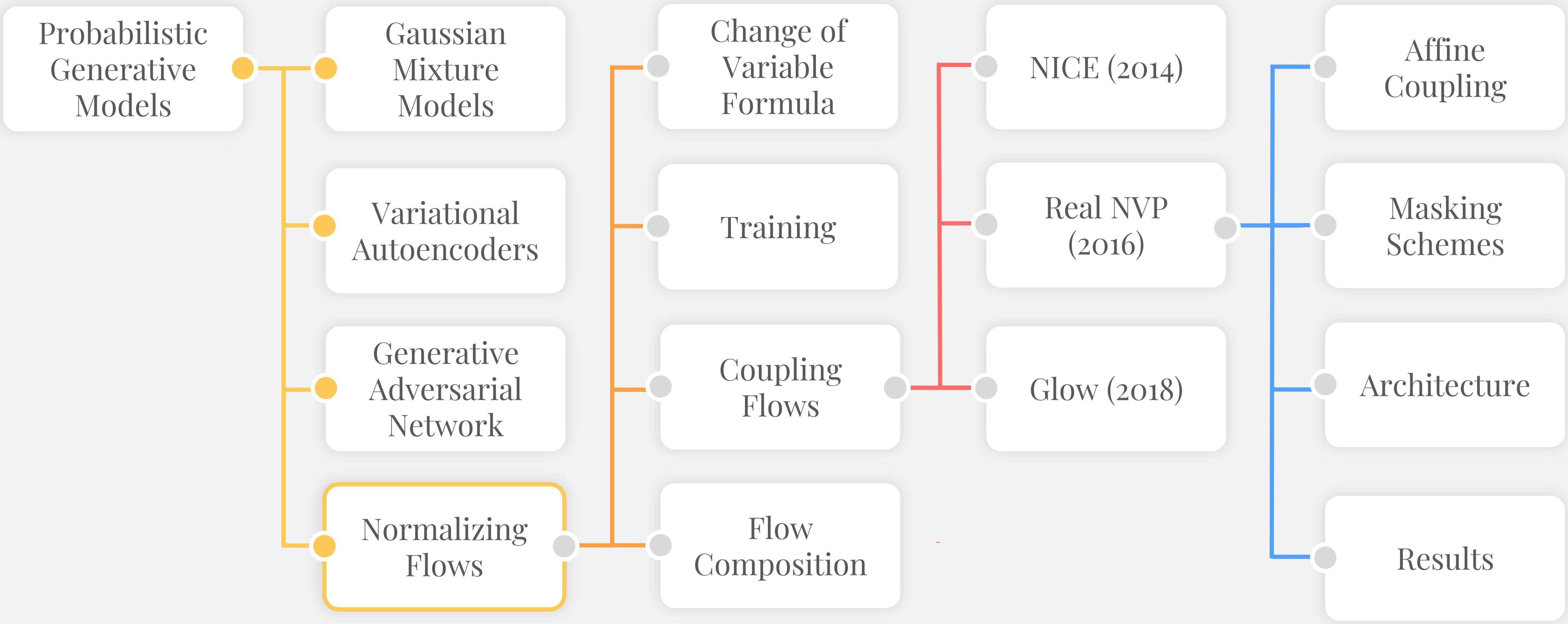
– Density estimation $p_{\boldsymbol{X}}(\boldsymbol{x})$ is not possible.

$\boldsymbol{x} \in \mathbb{R}^D$: data point

$\boldsymbol{z} \in \mathbb{R}^{D'}$: latent data point

$$\min_G \max_D V(D, G)$$

Map


Probabilistic Generative Models
Gaussian Mixture Models
Variational Autoencoders
Generative Adversarial Network
Normalizing Flows
Change of Variable Formula
Training
Coupling Flows
Flow Composition
NICE (2014)
Real NVP (2016)
Glow (2018)
Affine Coupling
Masking Schemes
Architecture
Results

Normalizing Flow

Introduction · Change of Variable · Diagram · Example

- A PGM to directly approximate $p_{\boldsymbol{X}}(\boldsymbol{x})$.
- Turn a simple distribution $p_{\boldsymbol{Z}}(\boldsymbol{z})$ into a complex one in an *invertible manner*.

\+ Easier to train than GANs and VAEs.

\+ Easy to scale.

\+ We can directly compute and optimize the likelihood.

\+ Density estimation $p_{\boldsymbol{X}}(\boldsymbol{x})$ is easy.

\+ Sampling $\boldsymbol{x} \sim p_{\boldsymbol{X}}(\cdot)$ is easy.

Normalizing Flow

Lemma 1. (Change of Variable) Let $\boldsymbol{X}$ and $\boldsymbol{Z}$ be random variables related by an *invertible* and *differentiable* mapping $f: \mathbb{R}^D \to \mathbb{R}^D$ such that $\boldsymbol{Z} = f(\boldsymbol{X})$, then the following equality holds:

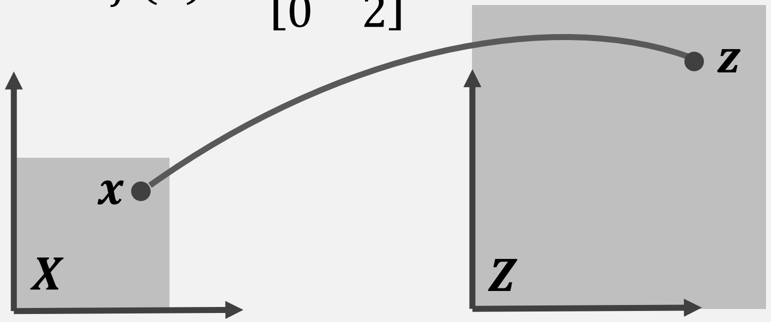
$$p_{\boldsymbol{X}}(\boldsymbol{x}) = p_{\boldsymbol{Z}}\big(f(\boldsymbol{x})\big)\big|\det\big(Df(\boldsymbol{x})\big)\big|$$

$\boldsymbol{x} \in \mathbb{R}^D$: data point

$\boldsymbol{z} \in \mathbb{R}^D$: “latent” variable

$$Df(\boldsymbol{x}) = \begin{bmatrix} \partial_{x_1} f_1(\boldsymbol{x}) & \cdots & \partial_{x_D} f_1(\boldsymbol{x}) \\ \vdots & \ddots & \vdots \\ \partial_{x_1} f_D(\boldsymbol{x}) & \cdots & \partial_{x_D} f_D(\boldsymbol{x}) \end{bmatrix}$$

$$f(\boldsymbol{x}) = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \boldsymbol{x}$$

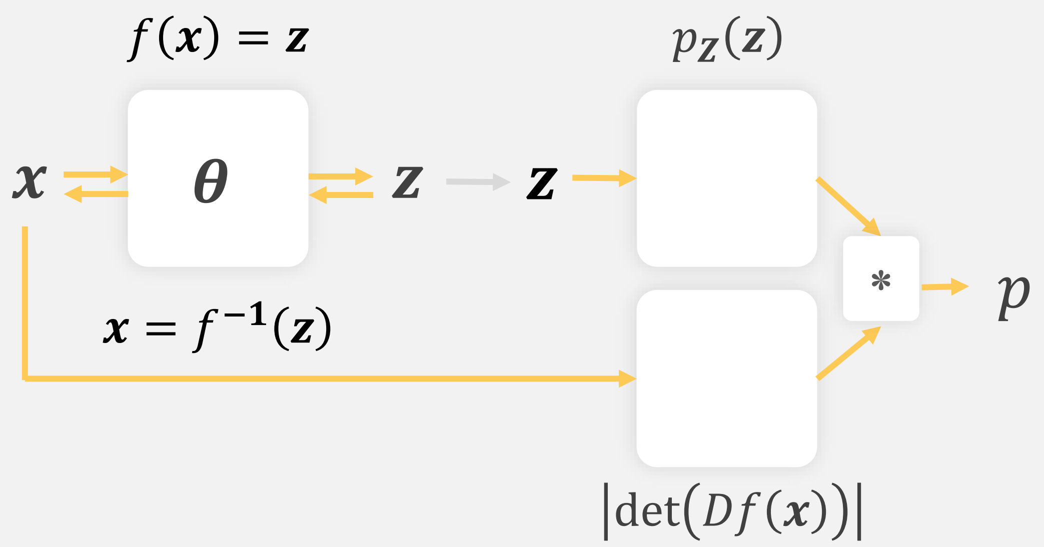


$$p_{\boldsymbol{X}}(\boldsymbol{x}) = p_{\boldsymbol{Z}}\left(\begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \boldsymbol{x}\right) \cdot 4$$

Normalizing Flow

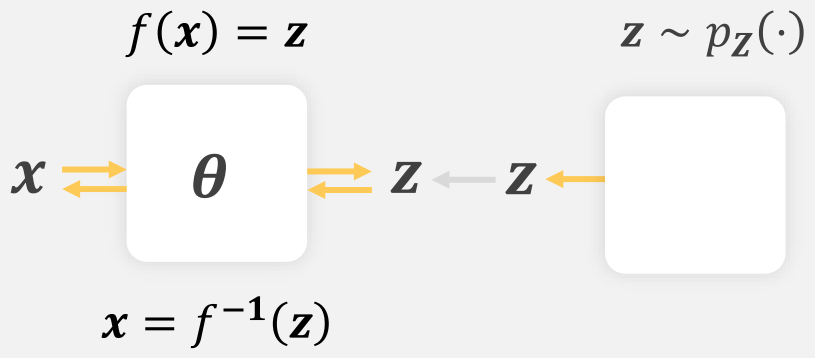
Density Estimation

$$p_X(\boldsymbol{x})$$

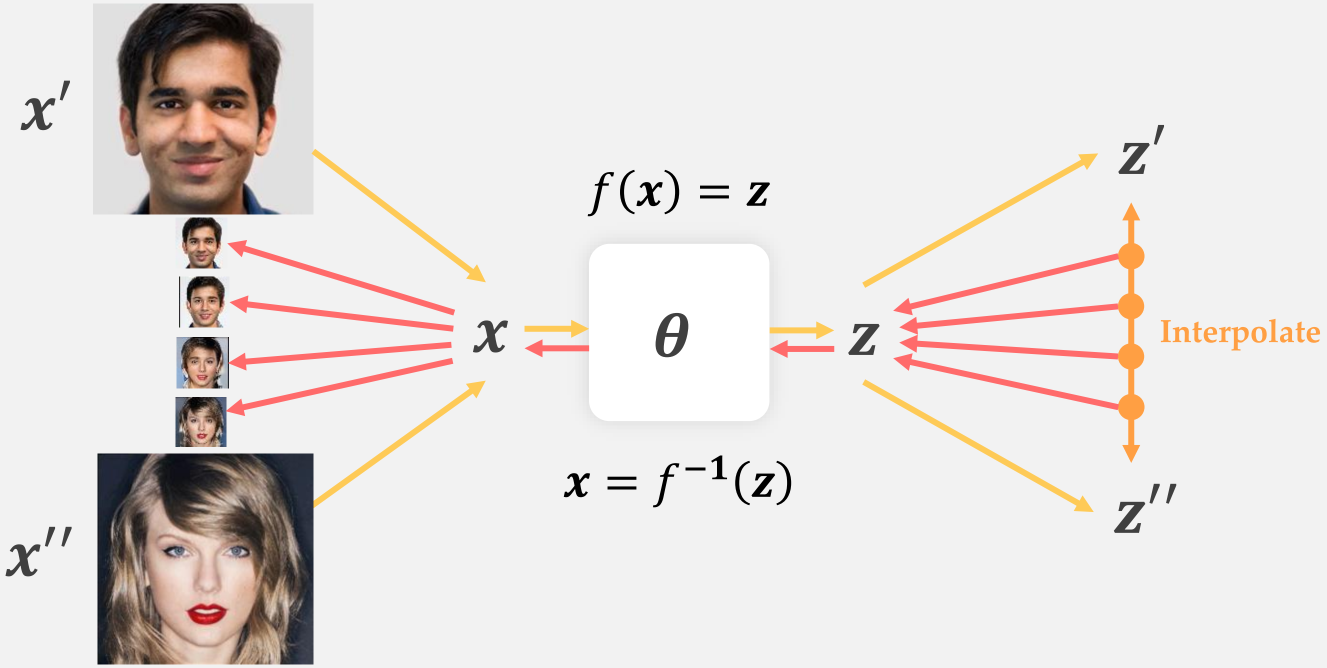


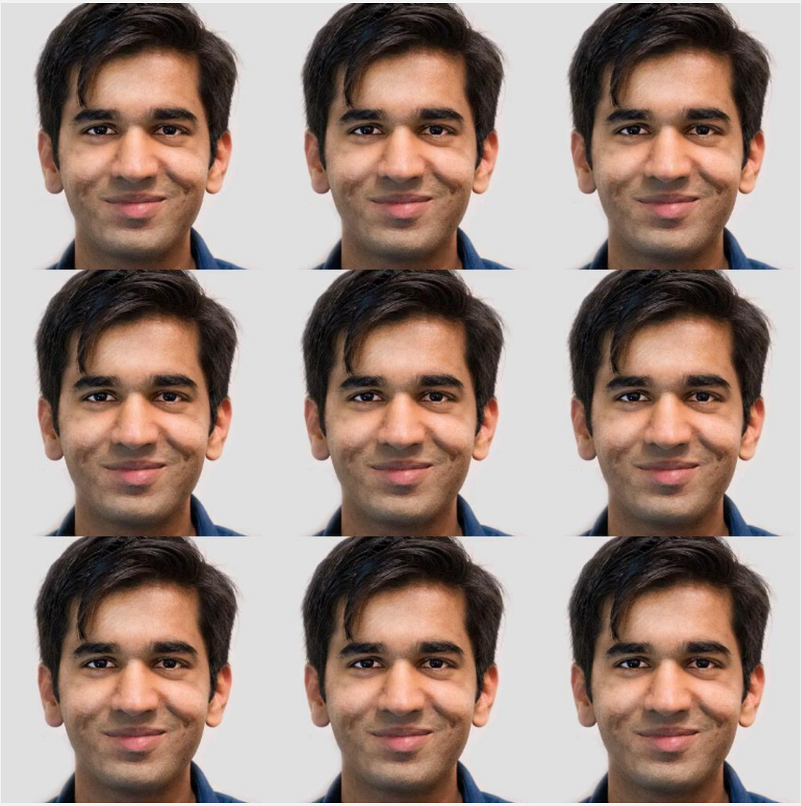
Sampling

$$\boldsymbol{x} \sim p_X(\cdot)$$



Normalizing Flow





"Glow: Generative flow with invertible 1x1 convolutions." (2018).

Training

Maximum Likelihood · Problems

Given a dataset $\mathcal{D} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ where $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \sim_{i.i.d.} p_{\boldsymbol{X}}(\cdot)$ we want to find the best parameters $\boldsymbol{\theta}$ of $f(\boldsymbol{x}|\boldsymbol{\theta})$ that maximize the likelihood of $\mathcal{D}$.

$$
\begin{aligned}
L(\mathcal{D}|\boldsymbol{\theta}) &= L(\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N|\boldsymbol{\theta}) \\
&= \prod_{i=1}^{N} p_{\boldsymbol{X}}(\boldsymbol{x}_i|\boldsymbol{\theta}) \\
&= \prod_{i=1}^{N} p_{\boldsymbol{Z}}\big(f(\boldsymbol{x}_i|\boldsymbol{\theta})\big)\big|\det\big(Df(\boldsymbol{x}_i|\boldsymbol{\theta})\big)\big|
\end{aligned}
$$

$$
\begin{aligned}
\widehat{\boldsymbol{\theta}} &= \arg\max_{\theta} L(\mathcal{D}|\boldsymbol{\theta}) \\
&= \arg\max_{\theta}\ \log L(\mathcal{D}|\boldsymbol{\theta}) \\
&= \arg\max_{\theta}\ \log \prod_{i=1}^{N} p_{\boldsymbol{Z}}\big(f(\boldsymbol{x}_i|\boldsymbol{\theta})\big)\big|\det\big(Df(\boldsymbol{x}_i|\boldsymbol{\theta})\big)\big| \\
&= \arg\max_{\theta} \sum_{i=1}^{N} \log p_{\boldsymbol{Z}}\big(f(\boldsymbol{x}_i|\boldsymbol{\theta})\big) + \log\big|\det\big(Df(\boldsymbol{x}_i|\boldsymbol{\theta})\big)\big|
\end{aligned}
$$

Training

Maximum Likelihood · **Problems**

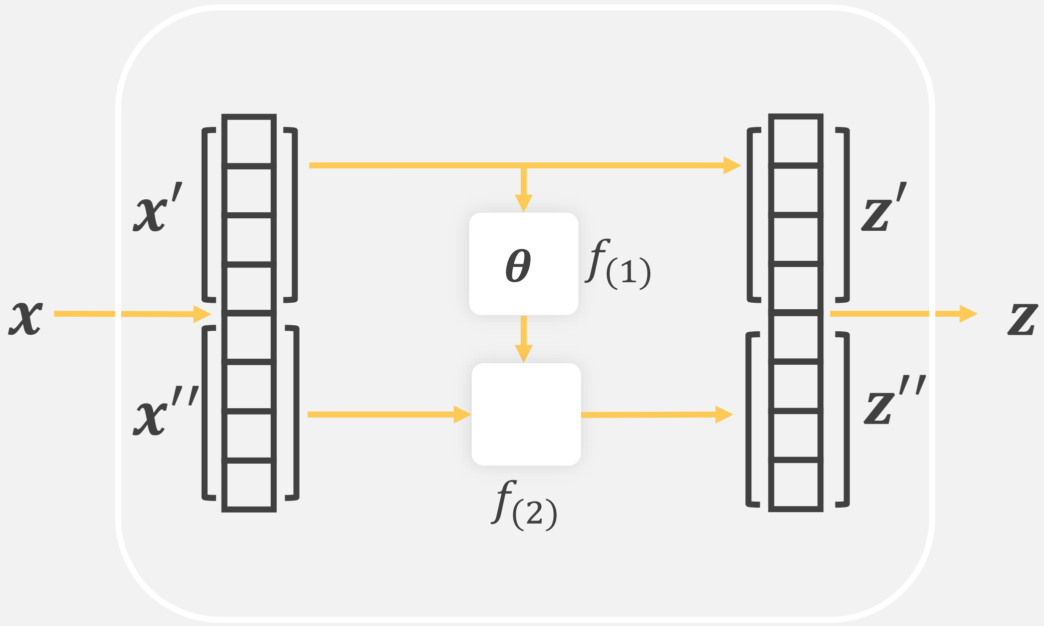
- Problem 1: How do we enforce the fact that f must be invertible (standard NNs aren't) ?
- Problem 2: How do we compute the determinant which is usually computationally prohibitive for large D ?

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \sum_{i=1}^{N} \log p_{\boldsymbol{Z}}\big(f(\boldsymbol{x}_i|\boldsymbol{\theta})\big) + \log\left|\det\big(Df(\boldsymbol{x}_i|\boldsymbol{\theta})\big)\right|$$

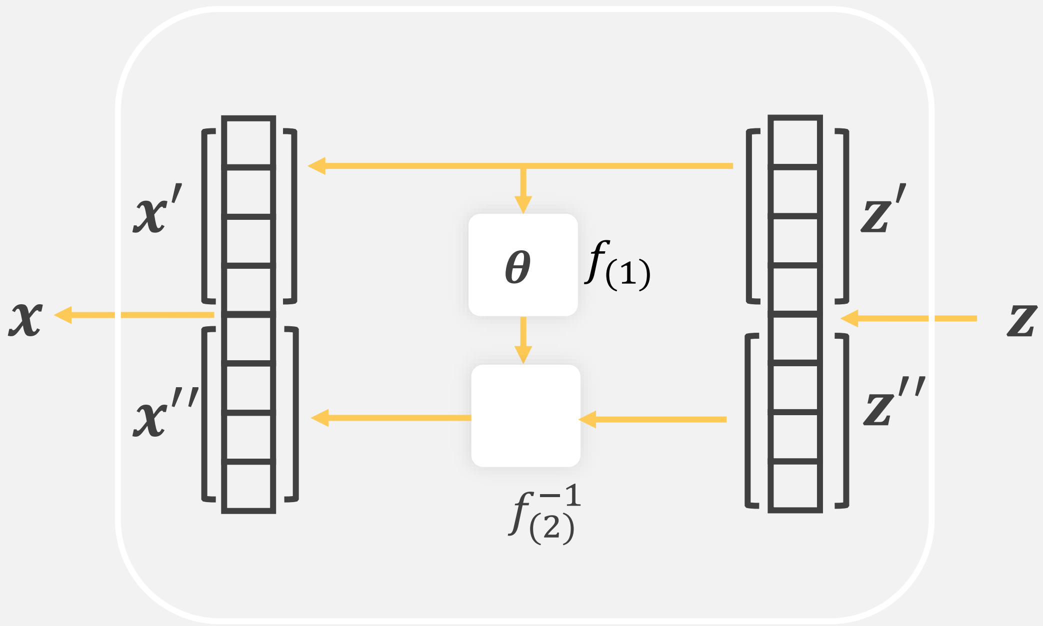
Coupling Flows

Solution To Problem 1 · Solution To Problem 2

$$f(\boldsymbol{x} \mid \boldsymbol{\theta}) = \boldsymbol{z}$$



$$f^{-1}(\boldsymbol{z} \mid \boldsymbol{\theta}) = \boldsymbol{x}$$



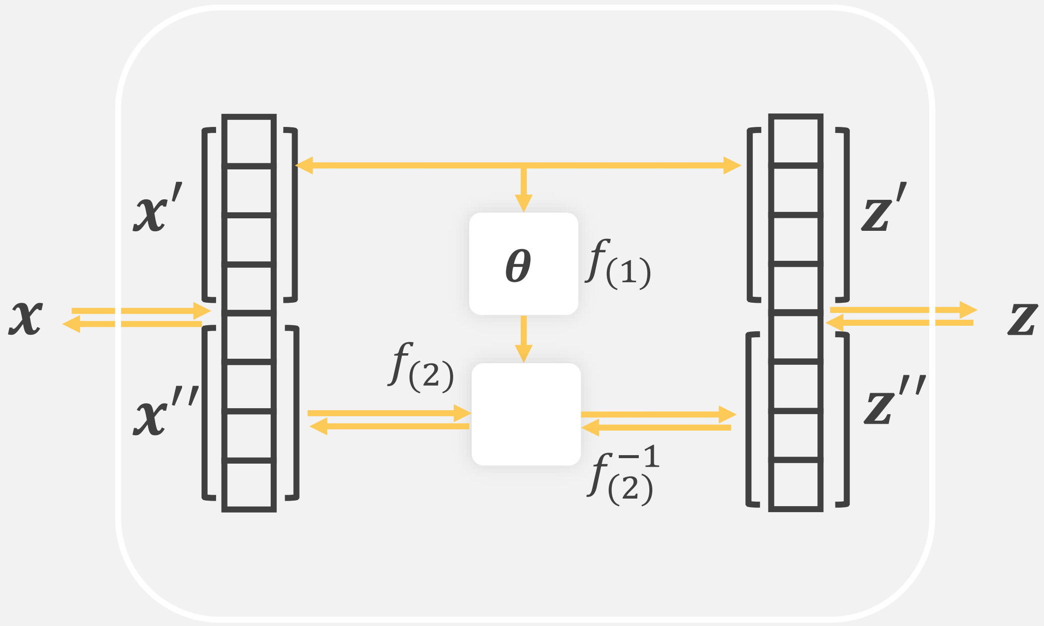
$$f(\boldsymbol{x} \mid \boldsymbol{\theta}) = \left(\boldsymbol{x}', f_{(2)}\left(\boldsymbol{x}'', f_{(1)}(\boldsymbol{x}' \mid \boldsymbol{\theta})\right)\right)$$

$$f^{-1}(\boldsymbol{z} \mid \boldsymbol{\theta}) = \left(\boldsymbol{z}', f_{(2)}^{-1}\left(\boldsymbol{z}'', f_{(1)}(\boldsymbol{z}' \mid \boldsymbol{\theta})\right)\right)$$

Coupling Flows

Solution To Problem 1 · Solution To Problem 2

$$f(\boldsymbol{x} \mid \boldsymbol{\theta}) = \boldsymbol{z}$$



$$f^{-1}(\boldsymbol{z} \mid \boldsymbol{\theta}) = \boldsymbol{x}$$

$$f_{(1)}(\cdot \mid \boldsymbol{\theta}) = \boldsymbol{NN}(\cdot \mid \boldsymbol{\theta}) = \boldsymbol{t}$$

Assuming…

$$f_{(2)}(\boldsymbol{x}'', \boldsymbol{t}) = \boldsymbol{x}'' + \boldsymbol{t}$$

$$f_{(2)}^{-1}(\boldsymbol{z}'', \boldsymbol{t}) = \boldsymbol{z}'' - \boldsymbol{t}$$

We found a way to compute f in an invertible manner!

"NICE: NON-LINEAR INDEPENDENT COMPONENTS ESTIMATION (2015).

Coupling Flows

Solution To Problem 1 · **Solution To Problem 2**

- ~~Problem 1: How do we enforce the fact that f must be invertible (standard NNs aren't) ?~~
- Problem 2: How do we compute the determinant which is usually computationally prohibitive for large D ?

$$\hat{\boldsymbol{\theta}} = \arg\max_{\theta} \sum_{i=1}^{N} \log p_{\boldsymbol{Z}}\big(f(\boldsymbol{x}_i|\boldsymbol{\theta})\big) + \log\left|\det\big(Df(\boldsymbol{x}_i|\boldsymbol{\theta})\big)\right|$$

Coupling Flows

Solution To Problem 1 · **Solution To Problem 2**

Lemma 2. (Determinant LT Matrix) Let $\mathbf{A} \in \mathbb{R}^{D\times D}$ be a lower triangular matrix, then:

$$\det \mathbf{A} = \prod_{i=0}^{D} a_{ii}$$

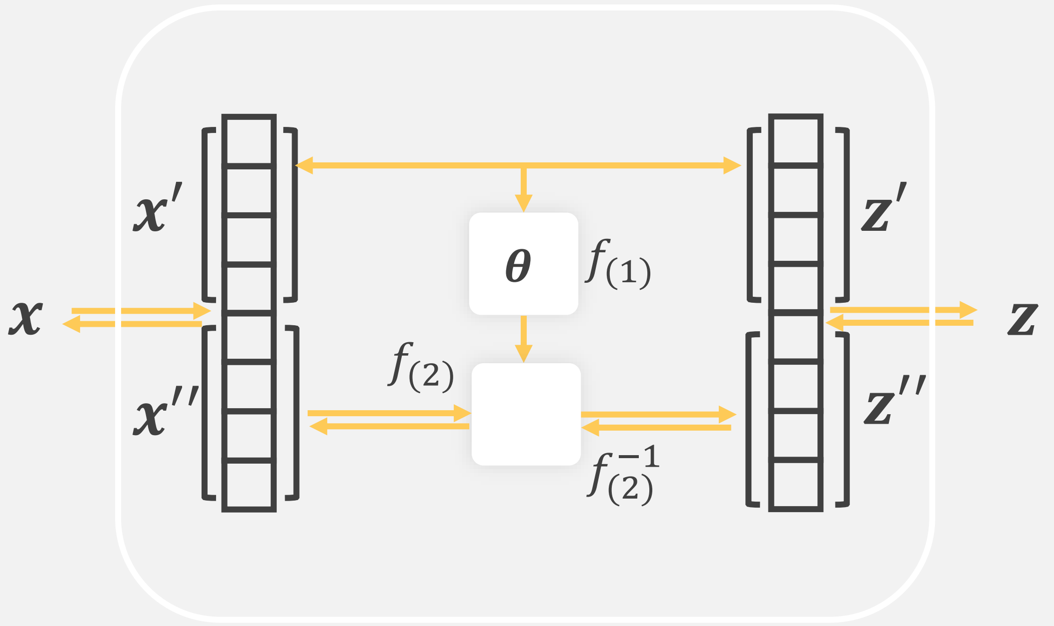
E.g.: $\det \begin{bmatrix} a_{11} & 0 & 0 \\ * & a_{22} & 0 \\ * & * & a_{33} \end{bmatrix} = a_{11}a_{22}a_{33}$

Coupling Flows

$\left|\det\left(Df(\boldsymbol{x}_i|\boldsymbol{\theta})\right)\right|$

Solution To Problem 1 · **Solution To Problem 2**

$$f(\boldsymbol{x} \mid \boldsymbol{\theta}) = \boldsymbol{z}$$



$$f^{-1}(\boldsymbol{z} \mid \boldsymbol{\theta}) = \boldsymbol{x}$$

Derivatives

- How much does $\boldsymbol{z}'$ change if we change $\boldsymbol{x}''$?

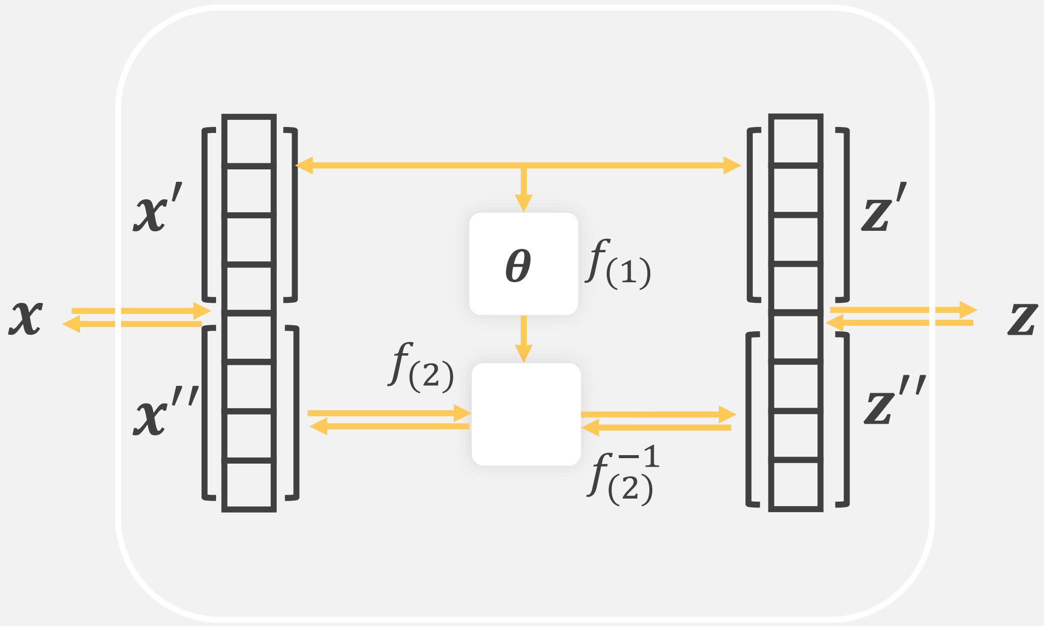
$$Df(\boldsymbol{x}) = \begin{bmatrix} * & * & * & * & * & * \\ * & * & * & * & * & * \\ * & * & * & * & * & * \\ * & * & * & * & * & * \\ * & * & * & * & * & * \\ * & * & * & * & * & * \end{bmatrix}$$

Coupling Flows

$\left|\det\left(Df(\boldsymbol{x}_i|\boldsymbol{\theta})\right)\right|$

Solution To Problem 1 · **Solution To Problem 2**

$$f(\boldsymbol{x} \mid \boldsymbol{\theta}) = \boldsymbol{z}$$



$$f^{-1}(\boldsymbol{z} \mid \boldsymbol{\theta}) = \boldsymbol{x}$$

Derivatives

- How much does $\boldsymbol{z}'$ change if we change $\boldsymbol{x}''$?
- How much does $\boldsymbol{z}'$ change if we change $\boldsymbol{x}'$?

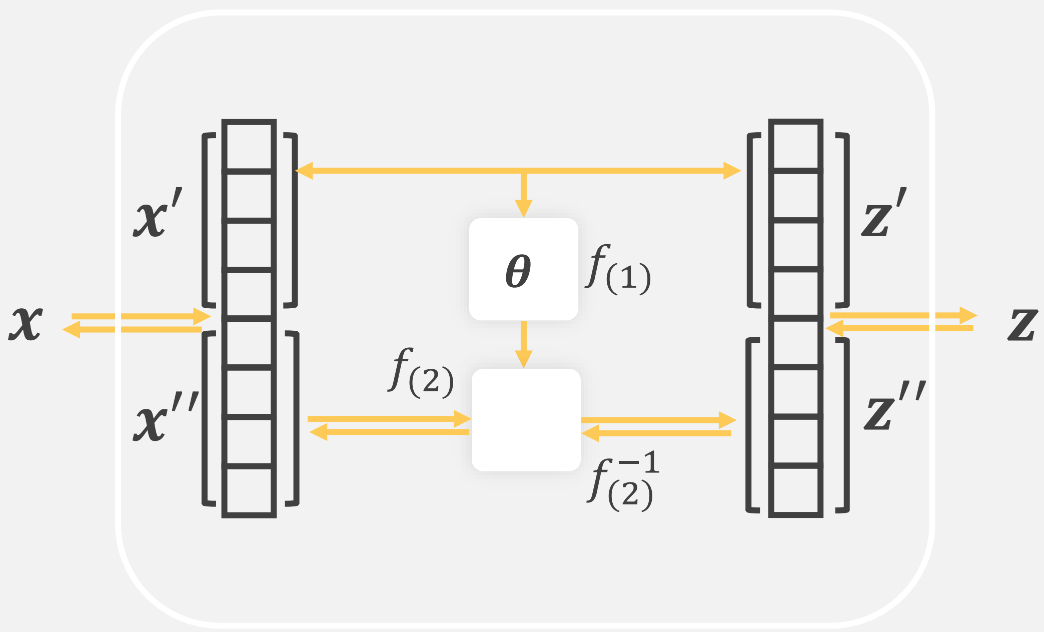
$$Df(\boldsymbol{x}) = \begin{array}{c} \left[\begin{array}{ccc|ccc} * & * & * & 0 & 0 & 0 \\ * & * & * & 0 & 0 & 0 \\ * & * & * & 0 & 0 & 0 \\ \hline * & * & * & * & * & * \\ * & * & * & * & * & * \\ * & * & * & * & * & * \end{array}\right] \begin{array}{l} \\ \boldsymbol{z}' \\ \\ \\ \boldsymbol{z}'' \\ \\ \end{array} \\ \quad\underbrace{\qquad\quad}_{\boldsymbol{x}'} \quad \underbrace{\qquad\quad}_{\boldsymbol{x}''} \end{array}$$

Coupling Flows

$\left|\det\left(Df(\boldsymbol{x}_i|\boldsymbol{\theta})\right)\right|$

Solution To Problem 1 · **Solution To Problem 2**

$$f(\boldsymbol{x} \mid \boldsymbol{\theta}) = \boldsymbol{z}$$



$$f^{-1}(\boldsymbol{z} \mid \boldsymbol{\theta}) = \boldsymbol{x}$$

Derivatives

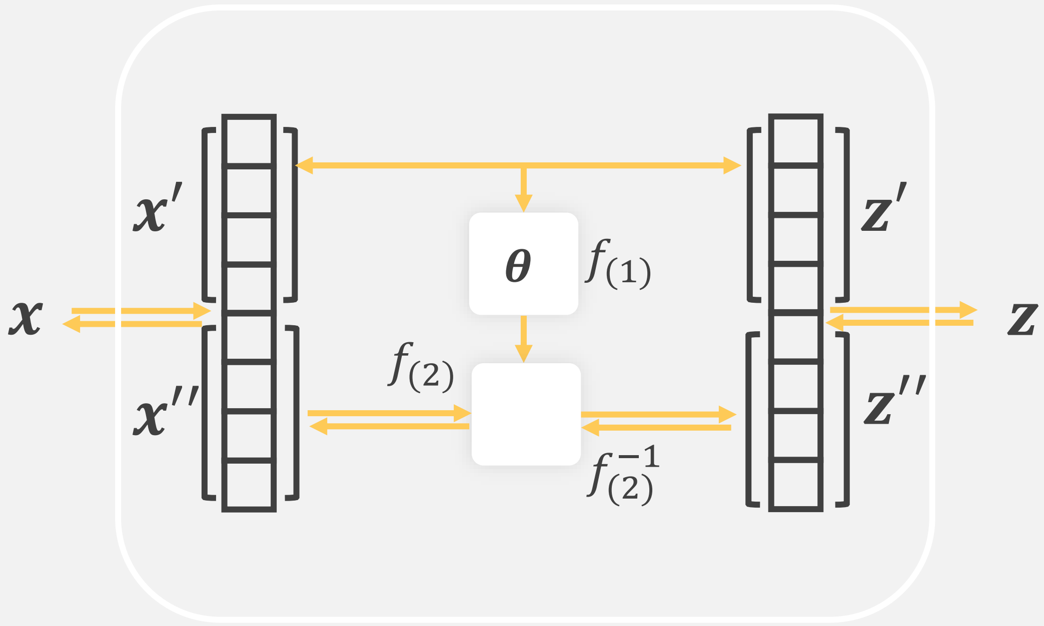
- How much does $\boldsymbol{z}'$ change if we change $\boldsymbol{x}''$?
- How much does $\boldsymbol{z}'$ change if we change $\boldsymbol{x}'$?
- How much does $\boldsymbol{z}''$ change if we change $\boldsymbol{x}''$?

Assuming $f_{(2)}(\boldsymbol{x}'', \boldsymbol{t}) = \boldsymbol{x}'' + \boldsymbol{t}$

$$Df(\boldsymbol{x}) = \begin{array}{c} \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ \hline * & * & * & * & * & * \\ * & * & * & * & * & * \\ * & * & * & * & * & * \end{array}\right] \begin{array}{l} \left.\vphantom{\begin{array}{c}0\\0\\0\end{array}}\right] z' \\ \left.\vphantom{\begin{array}{c}0\\0\\0\end{array}}\right] z'' \end{array} \\ \underbrace{\qquad\quad}_{x'} \quad \underbrace{\qquad\quad}_{x''} \end{array}$$

Coupling Flows

$$f(\boldsymbol{x} \mid \boldsymbol{\theta}) = \boldsymbol{z}$$



$$f^{-1}(\boldsymbol{z} \mid \boldsymbol{\theta}) = \boldsymbol{x}$$

Derivatives

- How much does $\boldsymbol{z}'$ change if we change $\boldsymbol{x}''$?
- How much does $\boldsymbol{z}'$ change if we change $\boldsymbol{x}'$?
- How much does $\boldsymbol{z}''$ change if we change $\boldsymbol{x}''$?

Assuming $f_{(2)}(\boldsymbol{x}'', \boldsymbol{t}) = \boldsymbol{x}'' + \boldsymbol{t}$

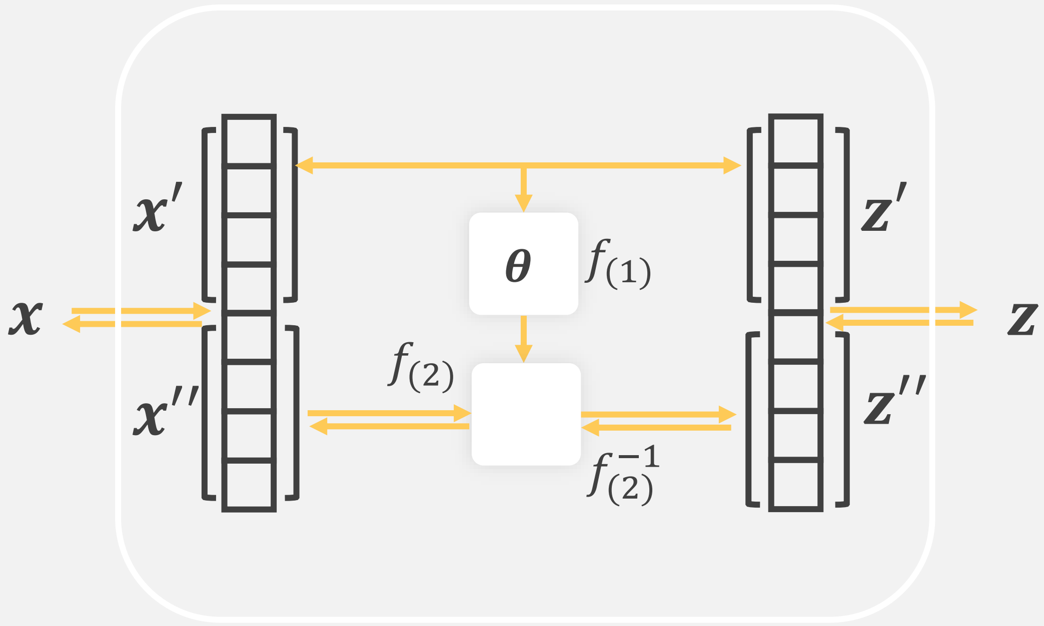
$$Df(\boldsymbol{x}) = \begin{array}{cc} \left[\begin{array}{ccc|ccc} 1&0&0&0&0&0\\ 0&1&0&0&0&0\\ 0&0&1&0&0&0\\ \hline *&*&*&1&0&0\\ *&*&*&0&1&0\\ *&*&*&0&0&1 \end{array}\right] & \begin{array}{l} \boldsymbol{z}' \\ \\ \boldsymbol{z}'' \end{array} \\ \begin{array}{cc} \boldsymbol{x}' & \quad \boldsymbol{x}'' \end{array} & \end{array}$$

Coupling Flows

$\left|\det\left(Df(\boldsymbol{x}_i|\boldsymbol{\theta})\right)\right|$

Solution To Problem 1 · **Solution To Problem 2**

$$f(\boldsymbol{x} \mid \boldsymbol{\theta}) = \boldsymbol{z}$$



$$f^{-1}(\boldsymbol{z} \mid \boldsymbol{\theta}) = \boldsymbol{x}$$

Assuming $f_{(2)}(\boldsymbol{x}'', \boldsymbol{t}) = \boldsymbol{x}'' + \boldsymbol{t}$

$$Df(\boldsymbol{x}) = \begin{array}{c} \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ \hline * & * & * & 1 & 0 & 0 \\ * & * & * & 0 & 1 & 0 \\ * & * & * & 0 & 0 & 1 \end{array}\right] \\ \begin{array}{cc} \underbrace{\qquad\qquad}_{\boldsymbol{x}'} & \underbrace{\qquad\qquad}_{\boldsymbol{x}''} \end{array} \end{array} \begin{array}{l} \left.\vphantom{\begin{array}{c}1\\1\\1\end{array}}\right\} \boldsymbol{z}' \\ \left.\vphantom{\begin{array}{c}1\\1\\1\end{array}}\right\} \boldsymbol{z}'' \\ \vphantom{\underbrace{x}_{x}} \end{array}$$

The Jacobian is lower triangular; hence we can easily compute the determinant!

Coupling Flows

Solution To Problem 1 · **Solution To Problem 2**

- ~~Problem 1: How do we enforce the fact that f must be invertible (standard NNs aren't) ?~~
- ~~Problem 2: How do we compute the determinant which is usually computationally prohibitive for large D ?~~

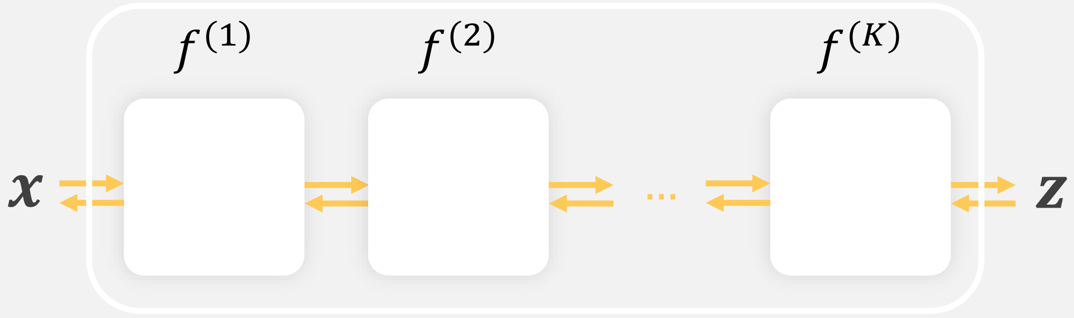
$$\hat{\boldsymbol{\theta}} = \arg\max_{\theta} \sum_{i=1}^{N} \log p_{\boldsymbol{Z}}\left(f(\boldsymbol{x}_i|\boldsymbol{\theta})\right) + \log\left|\det\left(Df(\boldsymbol{x}_i|\boldsymbol{\theta})\right)\right|$$

Flow Composition

Diagram · Determinant Jacobian Lemma

$$\hat{\boldsymbol{\theta}} = \arg\max_{\theta} \sum_{i=1}^{N} \log p_{\boldsymbol{z}}\big(f(\boldsymbol{x}_i|\theta)\big) + \log\left|\det\big(Df(\boldsymbol{x}_i|\boldsymbol{\theta})\big)\right|$$

$$f(\boldsymbol{x} \mid \boldsymbol{\theta}) = \boldsymbol{z}$$



$$f^{-1}(\boldsymbol{z} \mid \boldsymbol{\theta}) = \boldsymbol{x}$$

Flow Composition

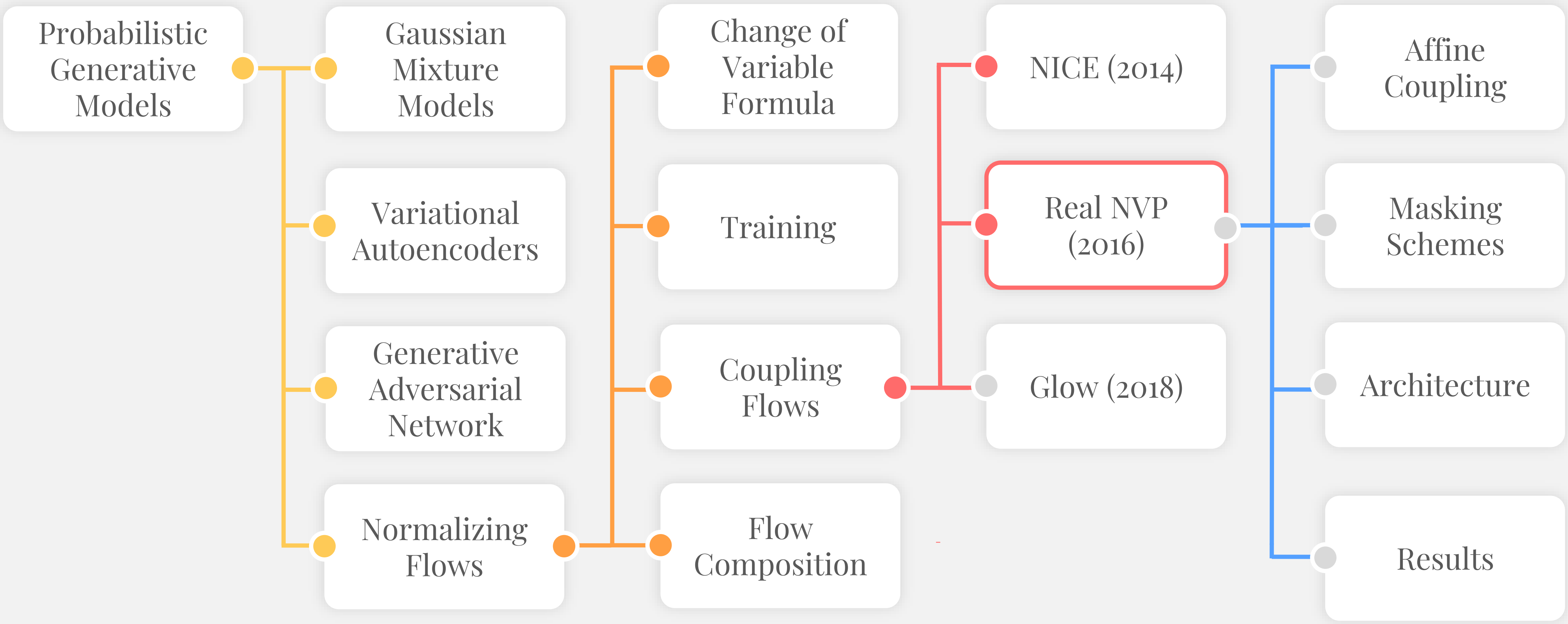
Diagram · **Determinant Jacobian Lemma**

Lemma 3. (Determinant Jacobian Composition) Let $f = f^{(1)} \circ f^{(2)} \circ \cdots \circ f^{(K)}$ with $f^{(j)} : \mathbb{R}^D \to \mathbb{R}^D$, then:

$$\det Df(\boldsymbol{x}) = \det \prod_{j=1}^{K} Df^{(j)}(\boldsymbol{x}) = \prod_{j=1}^{K} \det Df^{(j)}(\boldsymbol{x})$$

$$\begin{aligned}
\widehat{\boldsymbol{\theta}} &= \arg\max_{\theta} \sum_{i=1}^{N} \log p_{\boldsymbol{Z}}\big(f(\boldsymbol{x}_i|\boldsymbol{\theta})\big) + \log\big|\det\big(Df(\boldsymbol{x}_i|\boldsymbol{\theta})\big)\big| \\
&= \arg\max_{\theta} \sum_{i=1}^{N} \log p_{\boldsymbol{Z}}\big(f(\boldsymbol{x}_i|\boldsymbol{\theta})\big) + \log\left|\prod_{j=1}^{K} \det Df^{(j)}(\boldsymbol{x})\right| \\
&= \arg\max_{\theta} \sum_{i=1}^{N} \log p_{\boldsymbol{Z}}\big(f(\boldsymbol{x}_i|\boldsymbol{\theta})\big) + \sum_{j=1}^{K} \log\big|\det Df^{(j)}(\boldsymbol{x})\big|
\end{aligned}$$

Map

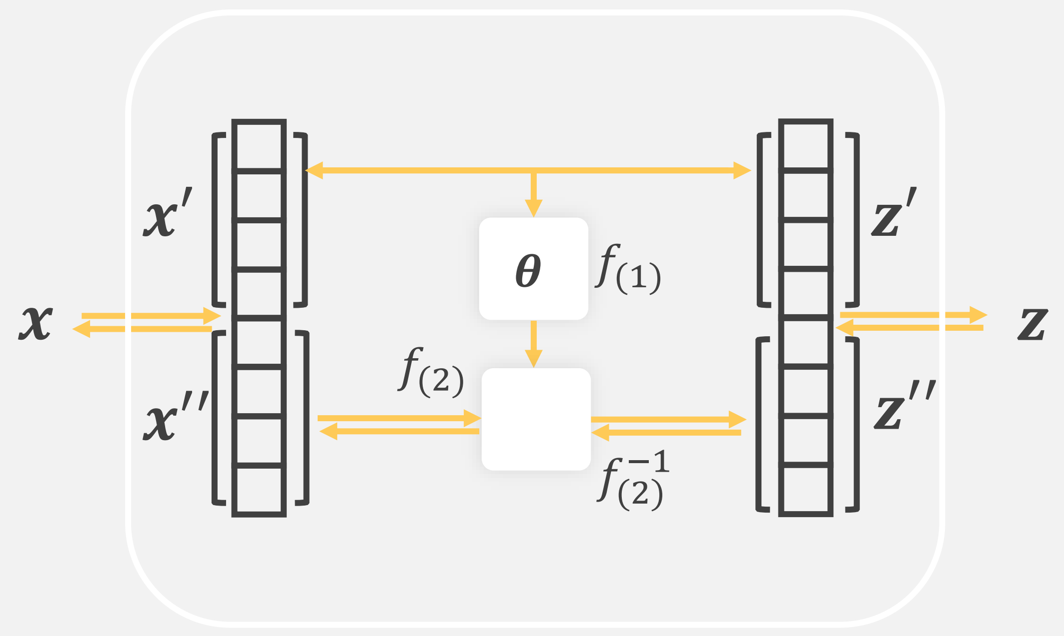


Probabilistic Generative Models
Gaussian Mixture Models
Variational Autoencoders
Generative Adversarial Network
Normalizing Flows
Change of Variable Formula
Training
Coupling Flows
Flow Composition
NICE (2014)
Real NVP (2016)
Glow (2018)
Affine Coupling
Masking Schemes
Architecture
Results

RealNVP (2017)

Affine Coupling Transform · Masking Schemes · Architecture · Results

$$f(\boldsymbol{x} \mid \boldsymbol{\theta}) = \boldsymbol{z}$$



$$f^{-1}(\boldsymbol{z} \mid \boldsymbol{\theta}) = \boldsymbol{x}$$

$$f_{(1)}(\cdot \mid \boldsymbol{\theta}) = \boldsymbol{NN}(\cdot \mid \boldsymbol{\theta}) = (\boldsymbol{t}, \boldsymbol{s})$$

Assuming…

$$f_{(2)}\big(\boldsymbol{x}'', (\boldsymbol{t}, \boldsymbol{s})\big) = \boldsymbol{x}'' \odot \exp(\boldsymbol{s}) + \boldsymbol{t}$$

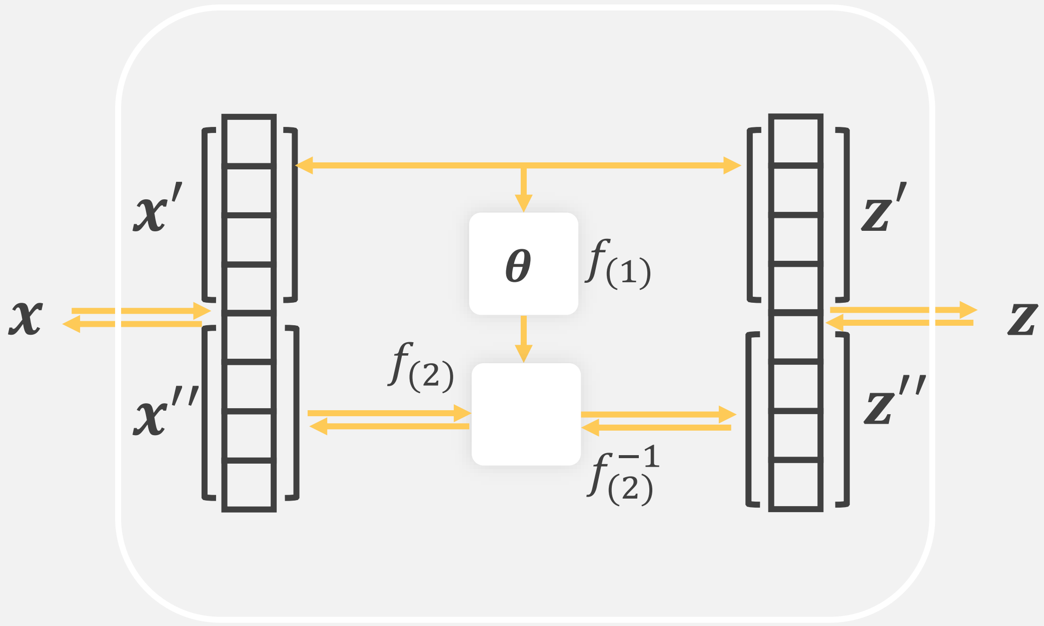
$$f_{(2)}^{-1}\big(\boldsymbol{z}'', (\boldsymbol{t}, \boldsymbol{s})\big) = (\boldsymbol{z}'' - \boldsymbol{t}) \odot \exp(-\boldsymbol{s})$$

$$Df(\boldsymbol{x}) = \begin{array}{cc} \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ \hline * & * & * & \exp(s_1) & 0 & 0 \\ * & * & * & 0 & \ddots & 0 \\ * & * & * & 0 & 0 & \exp(s_D) \end{array}\right] & \begin{array}{c} \boldsymbol{z}' \\ \\ \boldsymbol{z}'' \end{array} \\ \begin{array}{cc} \boldsymbol{x}' & \qquad \boldsymbol{x}'' \end{array} & \end{array}$$

RealNVP (2017)

Affine Coupling Transform · Masking Schemes · Architecture · Results

$$f(\boldsymbol{x} \mid \boldsymbol{\theta}) = \boldsymbol{z}$$



$$f^{-1}(\boldsymbol{z} \mid \boldsymbol{\theta}) = \boldsymbol{x}$$

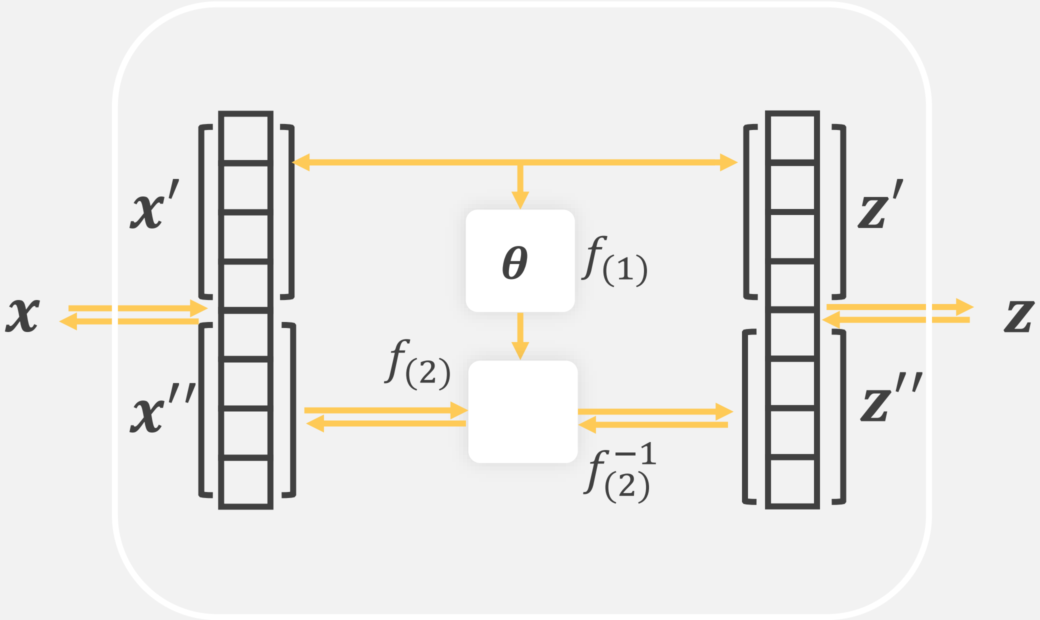
$$\log|\det Df(\boldsymbol{x})| = \log\left|\prod_{i=1}^{D} \exp s_i\right| = \log\left|\exp \sum_{i=1}^{D} s_i\right| = \sum_{i=1}^{D} s_i$$

$$Df(\boldsymbol{x}) = \begin{array}{cc} \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ \hline * & * & * & \exp(s_1) & 0 & 0 \\ * & * & * & 0 & \ddots & 0 \\ * & * & * & 0 & 0 & \exp(s_D) \end{array}\right] & \begin{array}{l} \left.\begin{array}{c} \\ \\ \\ \end{array}\right\} z' \\ \left.\begin{array}{c} \\ \\ \\ \end{array}\right\} z'' \end{array} \\ \underbrace{\hphantom{* \quad * \quad *}}_{x'} \quad \underbrace{\hphantom{\exp(s_1) \quad 0 \quad \exp(s_D)}}_{x''} & \end{array}$$

RealNVP (2017)

Affine Coupling Transform · **Masking Schemes** · Architecture · Results

$$f(\boldsymbol{x} \mid \boldsymbol{\theta}) = \boldsymbol{z}$$

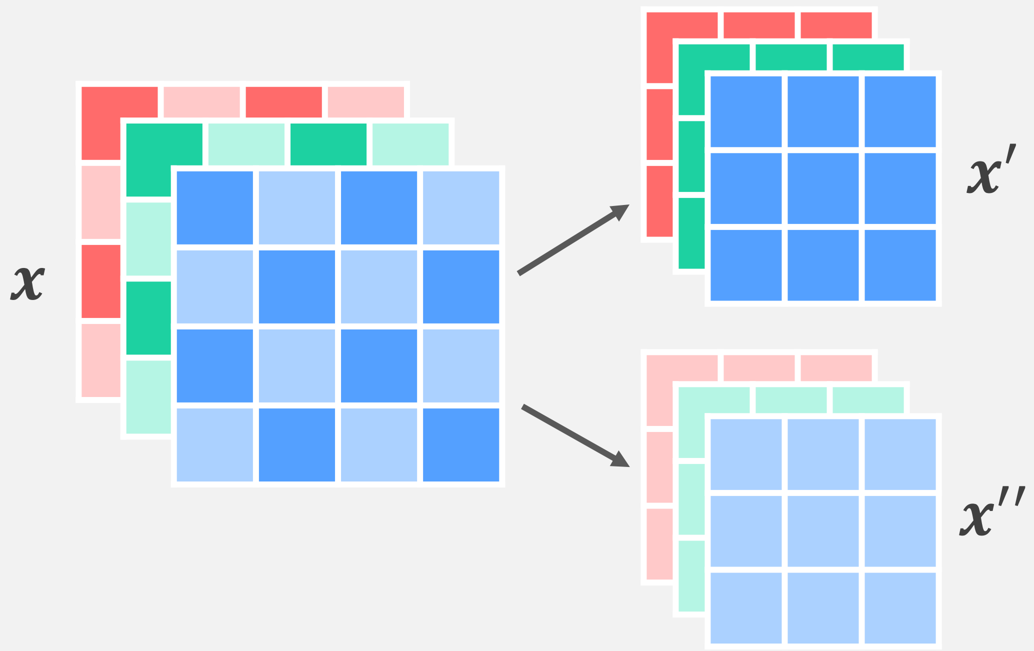


$$f^{-1}(\boldsymbol{z} \mid \boldsymbol{\theta}) = \boldsymbol{x}$$

RealNVP (2017)

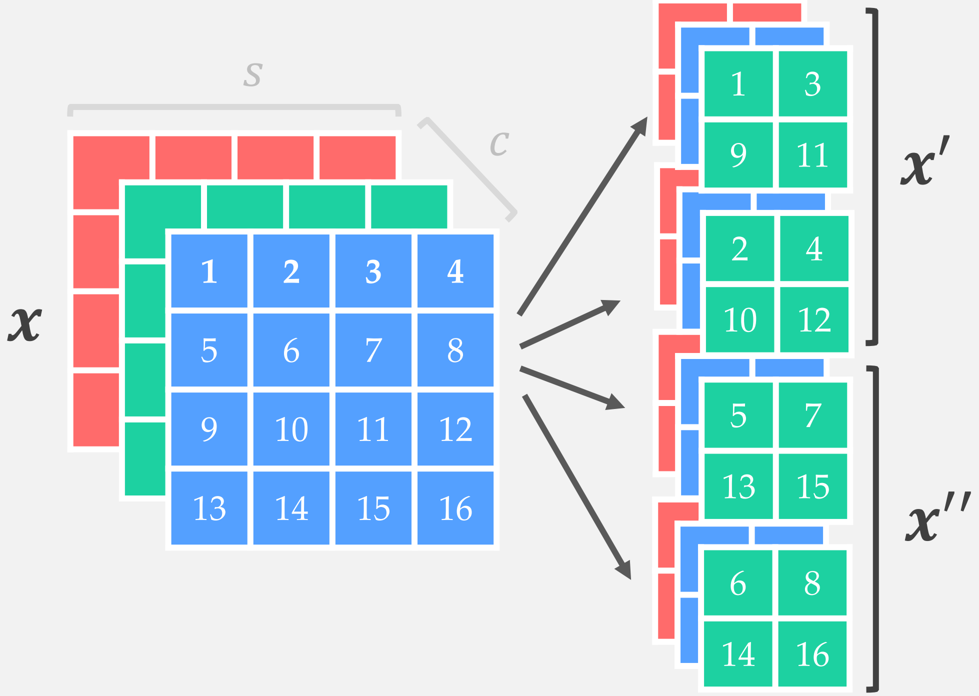
Affine Coupling Transform · **Masking Schemes** · Architecture · Results

Checkboard Masking (M1)



Channel-Wise Masking (M2)

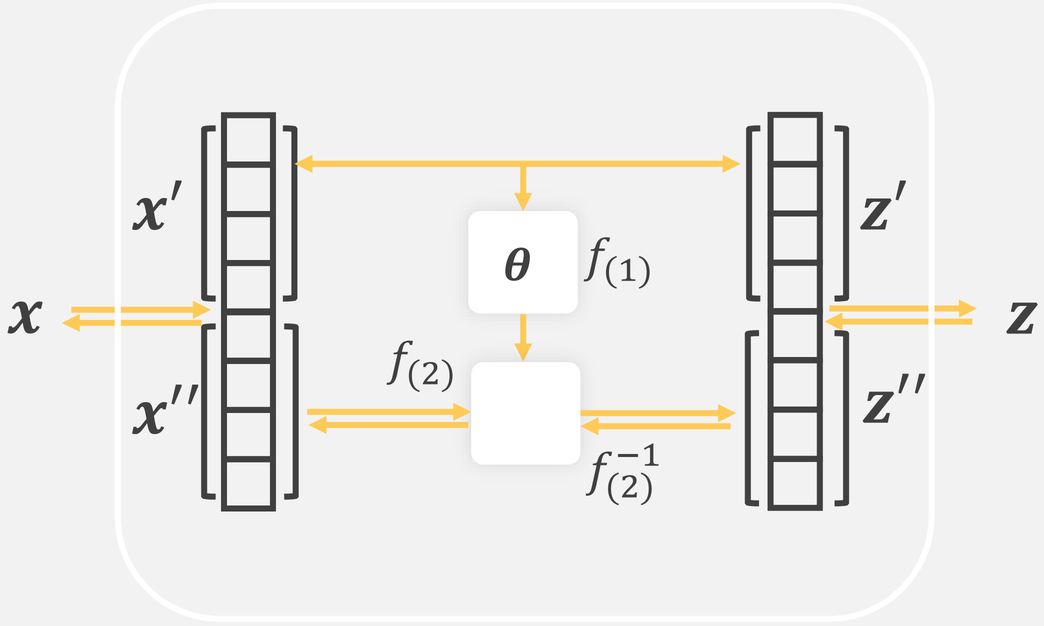
$$s \times s \times c \rightarrow \frac{s}{2} \times \frac{s}{2} \times 4c$$



RealNVP (2017)

"Affine Coupling Layer"

f_{ACL}



$$f_{(1)}(\cdot,\boldsymbol{\theta}) = ResNet(\cdot, n_{blocks}, \boldsymbol{\theta})$$

$$f_{(2)}\big(\boldsymbol{x}'', (\boldsymbol{t},\boldsymbol{s})\big) = \boldsymbol{x}'' \odot \exp(\boldsymbol{s}) + \boldsymbol{t}$$

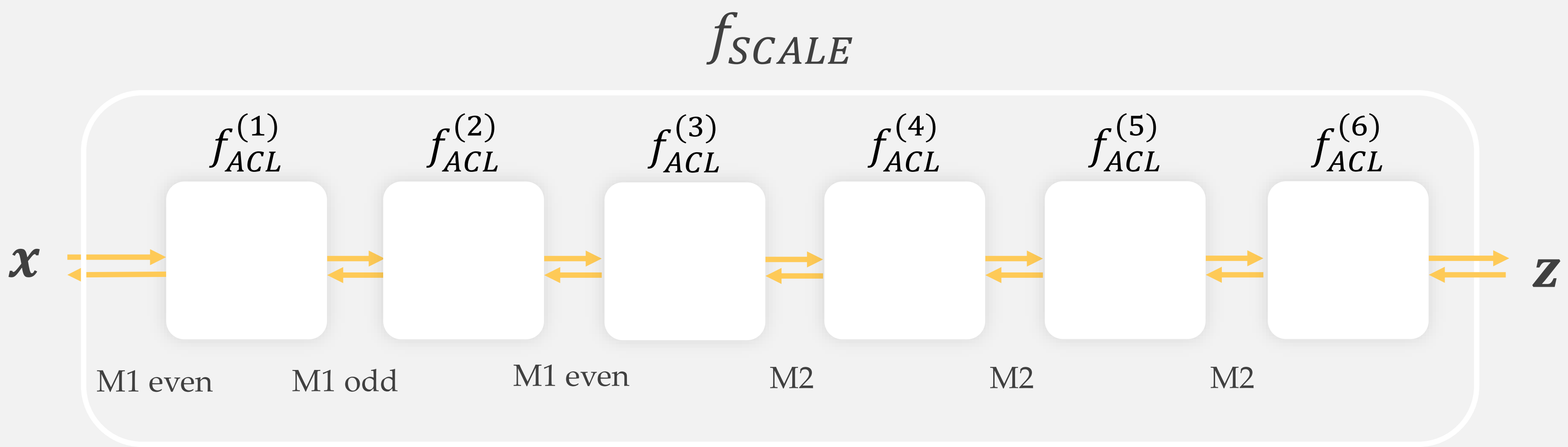
$$f_{(2)}^{-1}\big(\boldsymbol{z}'', (\boldsymbol{t},\boldsymbol{s})\big) = (\boldsymbol{z}'' - \boldsymbol{t}) \odot \exp(-\boldsymbol{s})$$

RealNVP uses:

- $n_{blocks} = 4$ for 32x32 images.
- $n_{blocks} = 2$ for 64x64 images.

RealNVP (2017)

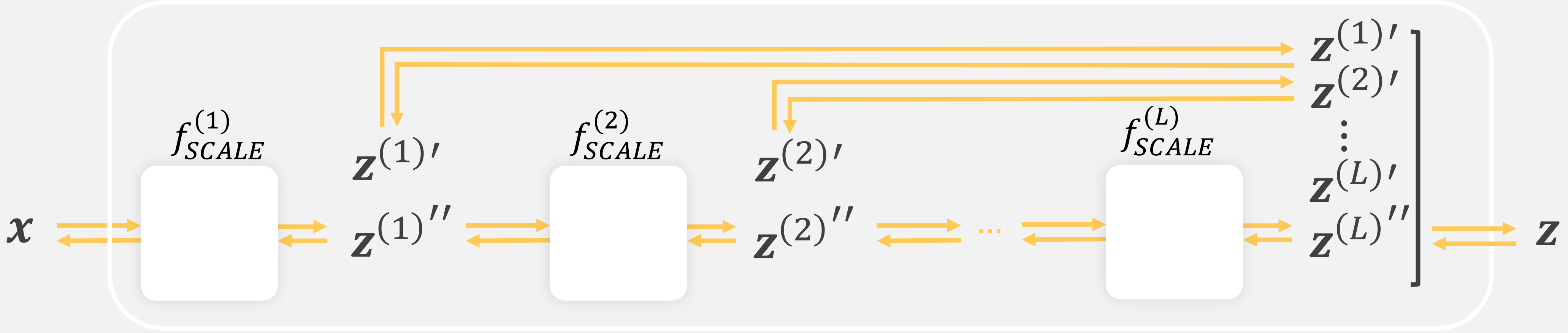
“Scale”



RealNVP (2017)

Final Multi-Scale Architecture

$$f(\boldsymbol{x} \mid \boldsymbol{\theta}) = \boldsymbol{z}$$



$$f^{-1}(\boldsymbol{z} \mid \boldsymbol{\theta}) = \boldsymbol{x}$$

RealNVP (2017)

Sampling: CIFAR-10

RealNVP (2017)

Affine Coupling Transform · Masking Schemes · Architecture · **Results**

Sampling: CelebA

RealNVP (2017)

Interpolating: CelebA

RealNVP (2017)

Affine Coupling Transform · Masking Schemes · Architecture · **Results**

Interpolating: LSUN (Tower)

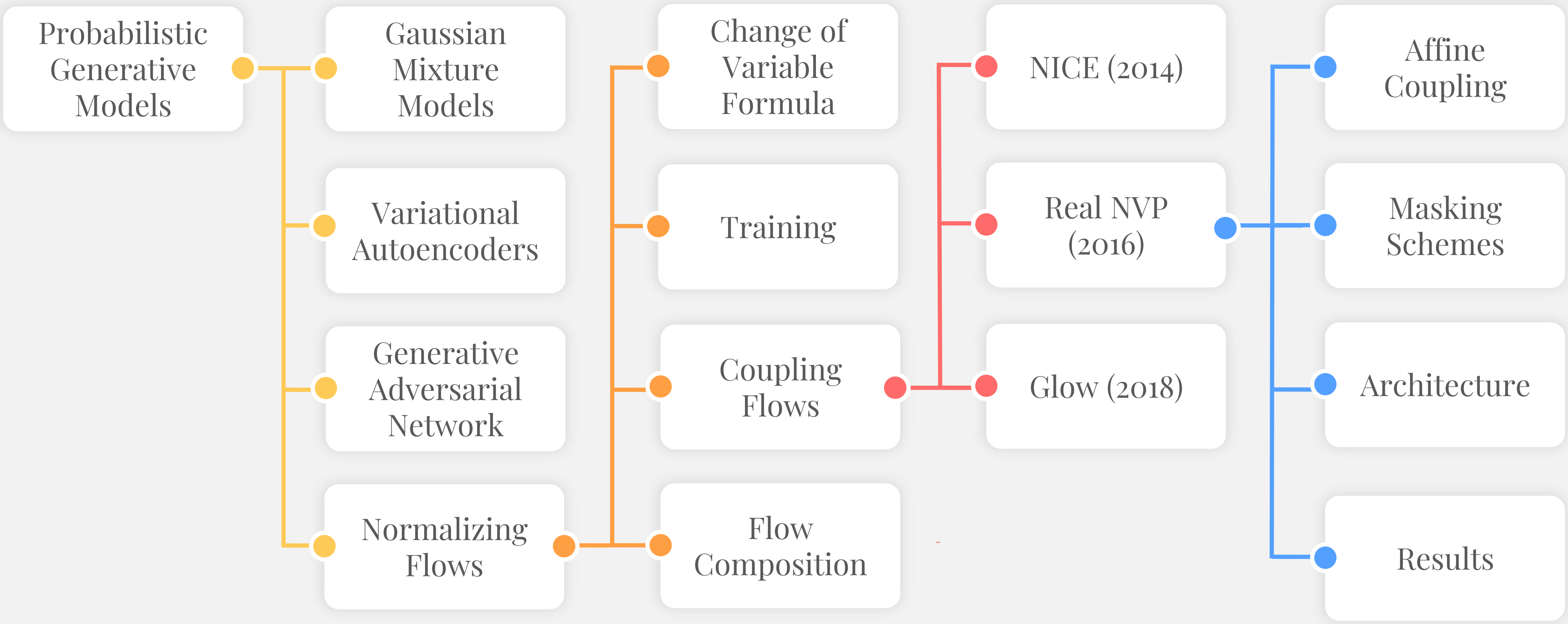
Glow (2018)

Sampling: CelebA

My Take

+ Even if the results are not so good, RealNVP laid the groundwork for papers like GLOW.
+ Invertible NN are a very interesting idea.
– The latent space is hard to interpret and “customize”.
– Some info about RealNVP is only found in the code.

Map



Probabilistic Generative Models
Gaussian Mixture Models
Variational Autoencoders
Generative Adversarial Network
Normalizing Flows
Change of Variable Formula
Training
Coupling Flows
Flow Composition
NICE (2014)
Real NVP (2016)
Glow (2018)
Affine Coupling
Masking Schemes
Architecture
Results

Thank you for your attention.

References

- RealNVP Paper: https://arxiv.org/abs/1605.08803
- NICE Paper: https://arxiv.org/abs/1410.8516
- Glow Paper: https://arxiv.org/abs/1807.03039
- Code Implementations RealNVP: https://paperswithcode.com/paper/density-estimation-using-real-nvp