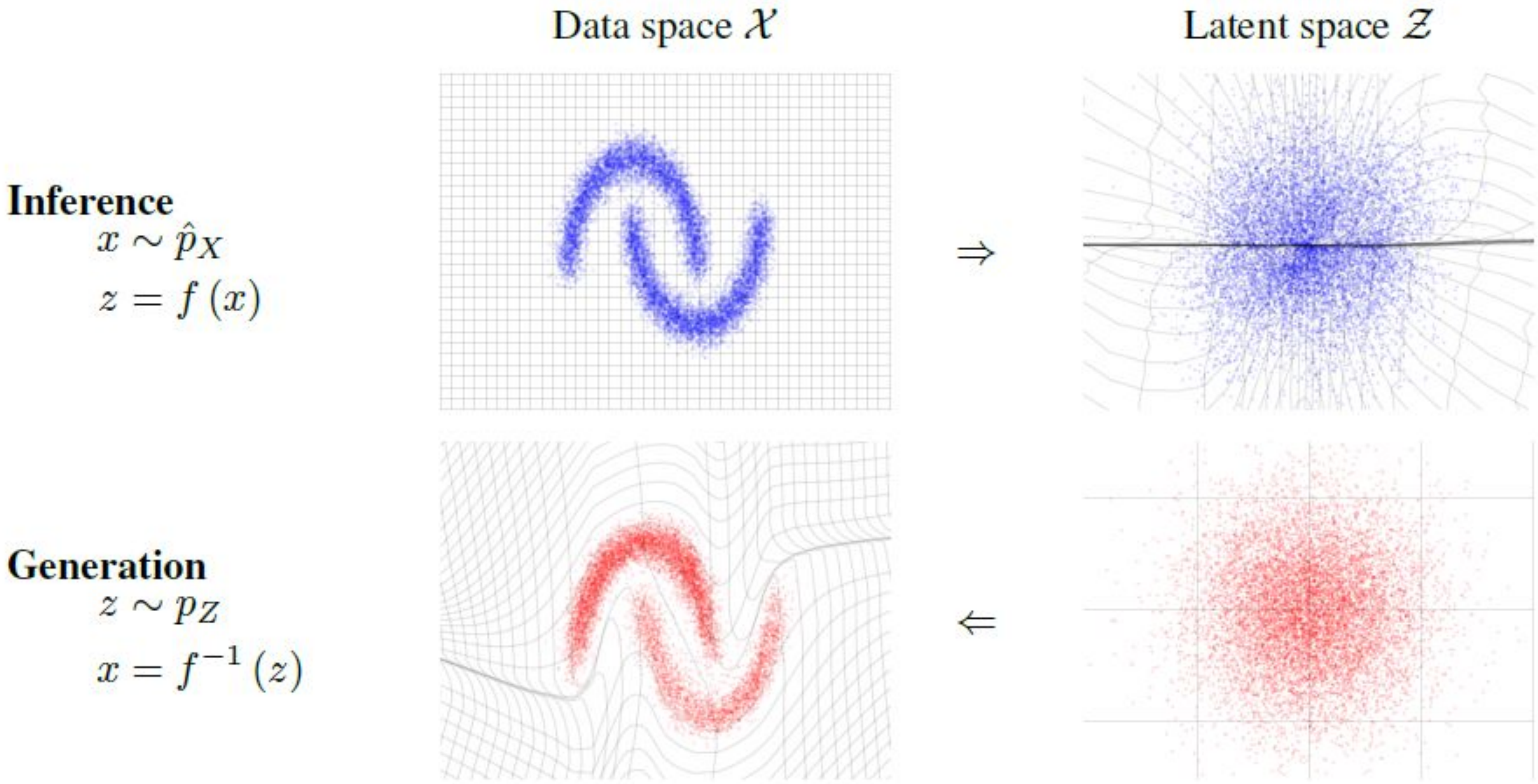
Density Estimation Using Real NVP

Presenter: Zhengyuan Cui


Data space $\mathcal{X}$
Latent space $\mathcal{Z}$
Inference
$x \sim \hat{p}_X$
$z = f(x)$
$\Rightarrow$
Generation
$z \sim p_Z$
$x = f^{-1}(z)$
$\Leftarrow$

$$
\begin{aligned}
p_X(x) &= p_Z(f(x)) \left|\det\left(\frac{\partial f(x)}{\partial x^T}\right)\right| \\
\log(p_X(x)) &= \log\Big(p_Z(f(x))\Big) + \log\left(\left|\det\left(\frac{\partial f(x)}{\partial x^T}\right)\right|\right),
\end{aligned}
$$

Coupling layer

$$y_{1:d} = x_{1:d}$$
$$y_{d+1:D} = x_{d+1:D} \odot \exp\big(s(x_{1:d})\big) + t(x_{1:d}),$$

$$\frac{\partial y}{\partial x^T} = \begin{bmatrix} \mathbb{I}_d & 0 \\ \frac{\partial y_{d+1:D}}{\partial x_{1:d}^T} & \operatorname{diag}\big(\exp\left[s\left(x_{1:d}\right)\right]\big) \end{bmatrix}$$

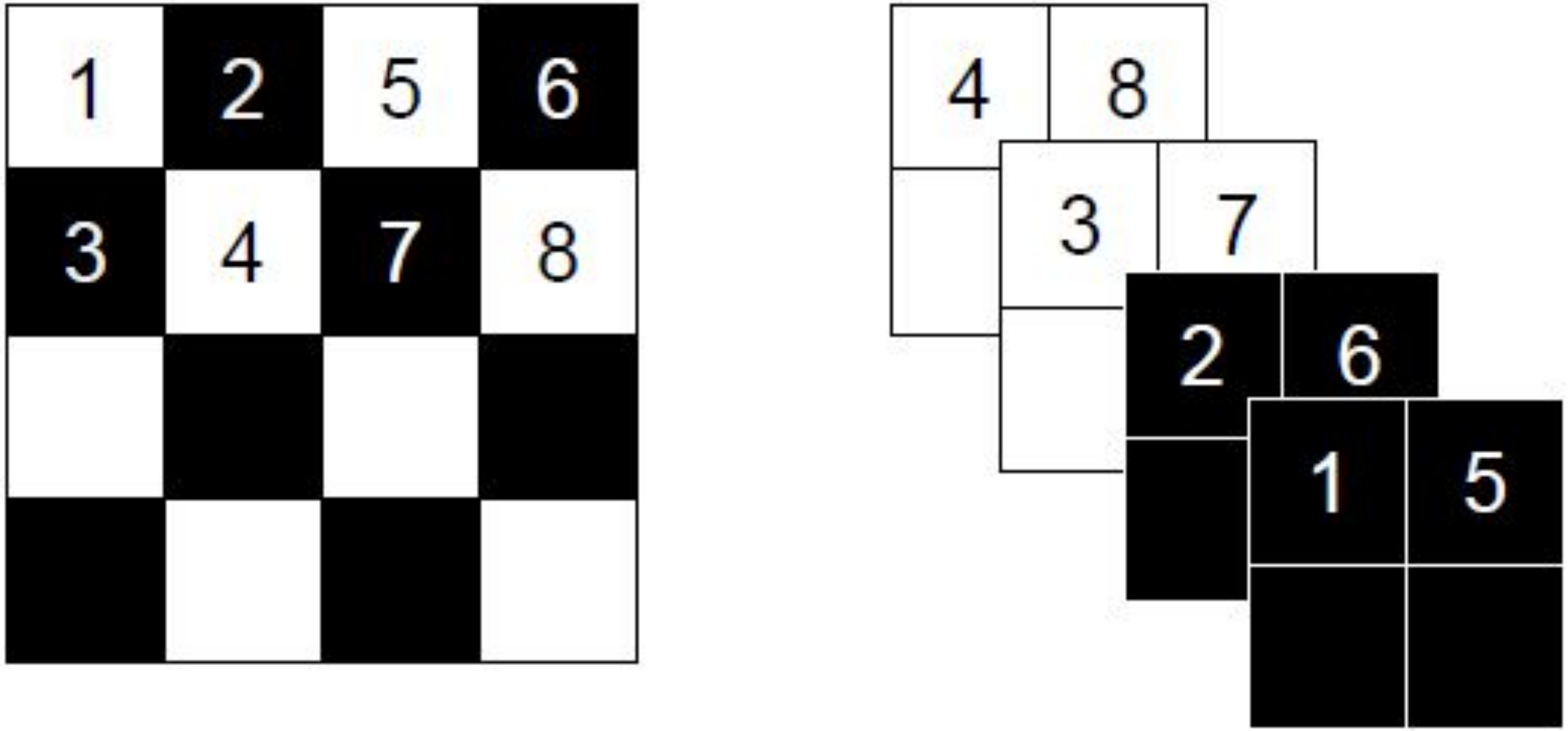
Coupling Layer

$$\begin{cases} y_{1:d} &= x_{1:d} \\ y_{d+1:D} &= x_{d+1:D} \odot \exp\big(s(x_{1:d})\big) + t(x_{1:d}) \end{cases}$$

$$\Leftrightarrow \begin{cases} x_{1:d} &= y_{1:d} \\ x_{d+1:D} &= \big(y_{d+1:D} - t(y_{1:d})\big) \odot \exp\big(-s(y_{1:d})\big), \end{cases}$$

Another expression

$$y = b \odot x + (1 - b) \odot \Big(x \odot \exp\left(s(b \odot x)\right) + t(b \odot x) \Big).$$

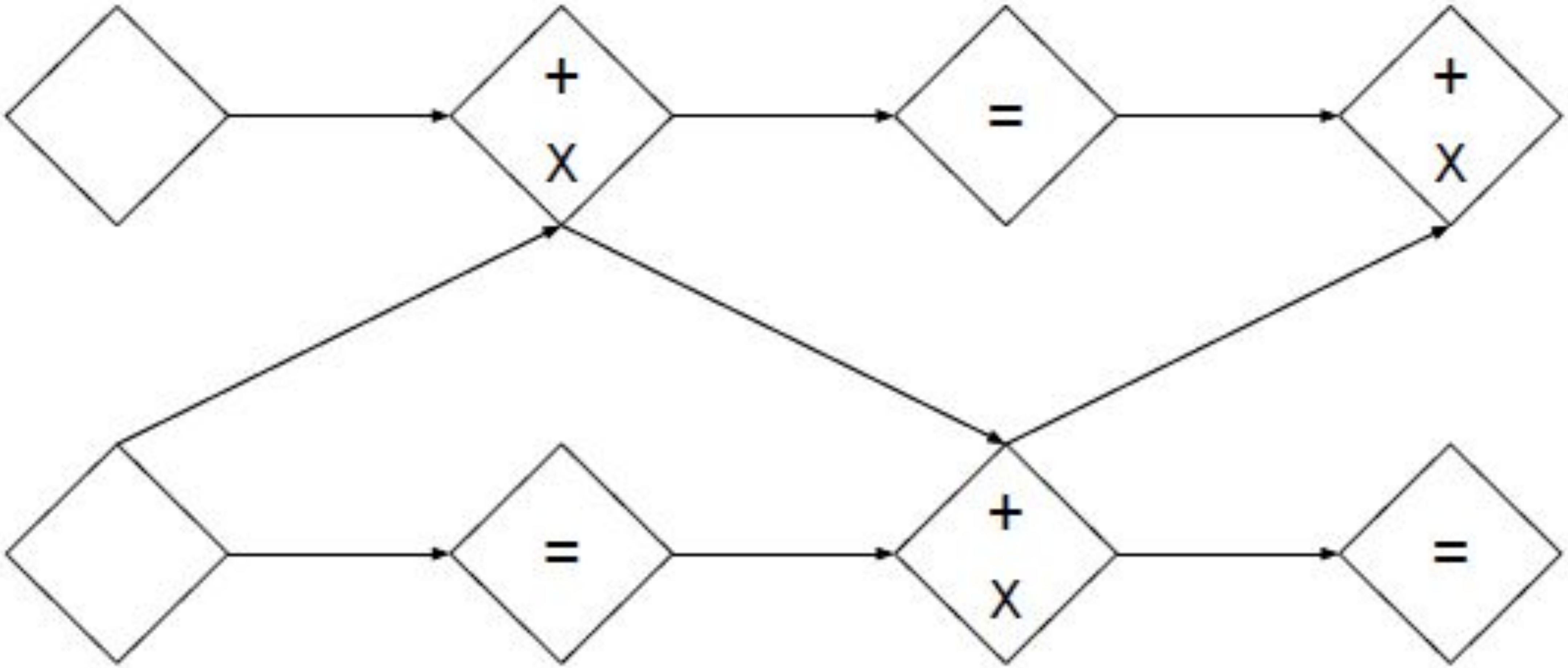

1
2
5
6
3
4
7
8
4
8
3
7
2
6
1
5

Combining coupling layers

$$\frac{\partial(f_b \circ f_a)}{\partial x_a^T}(x_a) = \frac{\partial f_a}{\partial x_a^T}(x_a) \cdot \frac{\partial f_b}{\partial x_b^T}(x_b = f_a(x_a))$$

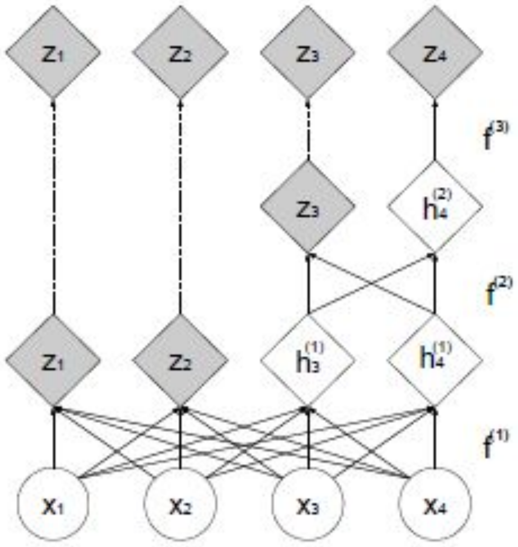
$$\det(A \cdot B) = \det(A)\det(B).$$

$$(f_b \circ f_a)^{-1} = f_a^{-1} \circ f_b^{-1}.$$



+
X
=
+
X
=
+
X
=

Multi-scale architecture



$$
\begin{aligned}
h^{(0)} &= x \\
(z^{(i+1)}, h^{(i+1)}) &= f^{(i+1)}(h^{(i)}) \\
z^{(L)} &= f^{(L)}(h^{(L-1)}) \\
z &= (z^{(1)}, \dots, z^{(L)}).
\end{aligned}
$$

Batch Normalization

$$x \mapsto \frac{x - \tilde{\mu}}{\sqrt{\tilde{\sigma}^2 + \epsilon}}$$

$$\left(\prod_i (\tilde{\sigma}_i^2 + \epsilon) \right)^{-\frac{1}{2}}$$

Experiment

- Multi-scale architecture (repeated recursively until the input of the last recursion is a 4 * 4 * c tensor)
- Deep convolutional residue network in coupling layer
- Optimizer: ADAM
- L2 regularization on weight scale parameters

Result

Dataset	PixelRNN [46]	Real NVP	Conv DRAW [22]	IAF-VAE [34]
CIFAR-10	3.00	3.49	< 3.59	< 3.28
Imagenet (32×32)	3.86 (3.83)	4.28 (4.26)	< 4.40 (4.35)	
Imagenet (64×64)	3.63 (3.57)	3.98 (3.75)	< 4.10 (4.04)	
LSUN (bedroom)		2.72 (2.70)		
LSUN (tower)		2.81 (2.78)		
LSUN (church outdoor)		3.08 (2.94)		
CelebA		3.02 (2.97)		

Results

Results

Smooth semantically consistent meaning

$$z = \cos(\phi)\left(\cos(\phi')z_{(1)} + \sin(\phi')z_{(2)}\right) + \sin(\phi)\left(\cos(\phi')z_{(3)} + \sin(\phi')z_{(4)}\right)$$

Pros and cons

- Performance: Competitive but has lots of room for improvement
- Pros:
  - Allow tractable and exact log likelihood
  - Flexible functional form allows fast and exact sampling from model distribution
  - Does not require the use of fixed form reconstruction cost
  - Able to learn a semantically meaningful latent space
- Cons:
  - Latent space does not reduce number of dimensions
  - Need much more space, too much networks

Discussion points

- In the multi-scale architecture, the factoring operation factors out lots of positions without going through enough transformation. Would this be a factor that hurts the final performance of the model?
- What other masks we can use besides the checkerboard mask and the channel-wise mask?

Useful resources

- https://www.youtube.com/watch?v=6GUSrmo9Qpw&t=458s The author of this paper talking about the model
- https://www.youtube.com/watch?v=u3vVyFVU_lI Introduction to normalizing flow
- https://arxiv.org/abs/1908.09257 Introduction to lots of flow methods: “Normalizing Flows: An Introduction and Review of Current Methods”