Flow Based Deep Generative Models

Tuesday, November 16, 2021 9:15 AM

Given $z \sim Z$ and $x \sim X$ with $p(z)$ and $p(x)$ respectively.
If z and x are related by

$$\begin{cases} x = f(z) & z \xrightarrow{f} x \\ z = f^{-1}(x) & x \xrightarrow{f^{-1}} z \end{cases}$$

If we know $p(z)$, can we find $p(x)$?

Answer: $$p(x) = p(z = f^{-1}(x)) \left| \frac{df^{-1}(x)}{dx} \right| \quad (\text{Eq } 1)$$

$\left| \frac{df^{-1}(x)}{dx} \right| \leftarrow$ This term is the determinant of the jacobian.

For example: $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ and $z = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$ where $\begin{cases} z_1 = f_a^{-1}(x) \\ z_2 = f_b^{-1}(x) \end{cases}$

then

$$\left| \frac{df^{-1}(x)}{dx} \right| = \det \begin{bmatrix} \frac{\partial}{\partial x_1} f_a^{-1}(x) & \frac{\partial}{\partial x_2} f_a^{-1}(x) \\ \frac{\partial}{\partial x_1} f_b^{-1}(x) & \frac{\partial}{\partial x_2} f_b^{-1}(x) \end{bmatrix} \quad (\text{Eq } 2)$$

<u>Eq 1</u>

$$p(x) = p(z = f^{-1}(x)) \left| \frac{df^{-1}(x)}{dx} \right|$$

↑ This is the key insight for Flow-based Deep Generative models.

- The model assume $\exists$ a bijective map f such that X can be mapped to z and back.

$x = f(z)$ $\quad z \xrightarrow{f} x \quad$ Assume $z \sim N(0,1)$

$z = f^{-1}(x)$ $\quad x \xrightarrow{f^{-1}} z$

- Since $p(z)$ is known ($\mathcal{N}(0,I)$), if we know f and f^{-1} then we can obtain $p(x)$ purely from data via Eq 1.

$$p(x) = p(z = f^{-1}(x)) \left| \frac{df^{-1}(x)}{dx} \right| \qquad (\text{Eq } 1)$$

- We can also generate samples from $p(X)$ by directly sampling from $p(z) \sim \mathcal{N}(0, I)$.

- The key is If we know f and f^{-1}

Since the data X can be very complicated, in reality we don't know f and f^{-1}

$$\boxed{\text{The flow model solves this question.}}$$

The trick

Let's assume that the data X and Z both have 4 dimensions.

$$X = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \quad \text{and} \quad Z = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \end{bmatrix}$$

The mapping from X to Z where $Z = f^{-1}(X)$ is

$$X = \begin{bmatrix} x_a \\ x_b \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \\ \begin{bmatrix} x_3 \\ x_4 \end{bmatrix} \end{bmatrix} \begin{matrix} \longrightarrow \\ \rightarrow \begin{bmatrix} x_3 \\ x_4 \end{bmatrix} e^{s\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right)} + t\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}\right) \longrightarrow \end{matrix} \begin{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \\ \begin{bmatrix} z_3 \\ z_4 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} z_a \\ z_b \end{bmatrix} = z$$

Therefore

$$z_a = x_a$$
$$z_b = x_b e^{s(x_a)} + t(x_a)$$

- Since $s(x_a)$ and $t(x_a)$ are considered as constants z_b is a linear function and "invertable."

- In fact we know the inverse

$$x_a = z_a$$
$$x_b = \frac{z_b - t(z_a)}{e^{s(x_a)}}$$

- while the output of $S(x_a)$ and $t(x_a)$ are constants, they themselves can be very complicated functions.

> Here, we set $S(x_a)$ and $t(x_a)$ as Neural Networks.

- We next return to Eq 1

$$p(x) = p(z = f^{-1}(x)) \left| \frac{df^{-1}(x)}{dx} \right|$$

- We can find f^{-1} by finding the f^{-1} that maxmizes the maximum likelihood of the data.

$$\max_{f^{-1}} \log \prod_{i=1}^{n} p(x_i) = \max_{f^{-1}} \log \prod_{i=1}^{n} p(z = f^{-1}(x)) \left| \frac{df^{-1}(x)}{dx} \right|$$

Here we have

$$z_1 = f_a^{-1}(x)$$

$$z_2 = f$$