

Day 5 : DDPM

* Reviewing VAE

- VAE with Single Latent Variable
- A model with single latent variable z
- Decoder : Models $p_\theta(x|z)$. Sample z from fixed distribution ($N(0, I)$) and convert to observed data x . Sampling $\xrightarrow{\text{fix to normal distribution}}$
- Encoder : Models $q_{\phi}(z|x)$. Used for calculating $q_{\phi}(z|x)$ to maximize ELBO (Approximation of $q_{\phi}(z|x)$ to $p_\theta(z|x)$)
 \downarrow
variational approximation / variational bayes
- Encoder & Decoder is implemented using Neural Network.

* VAE with multiple latent variables (hierarchical variable)

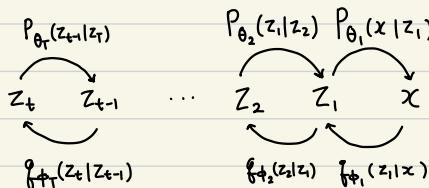
- Make multiple z s hierarchically \Rightarrow 복잡한 구조를 쉽게 이해하는 법!
- Current probabilistic variable is only decided by "previous" probabilistic variable \Rightarrow Markov property

* Markov Property

- 다음 상태($t+1$)은 오직 현재 상태(t)에만 기반, 그 이전의 상태에 대해서는

$$P(x_{t+1} = x_{t+1} | x_t = x_t, x_{t-1} = x_{t-1}, \dots, x_0 = x_0) = P(x_{t+1} = x_{t+1} | x_t = x_t)$$

Hierarchical VAE figure



Markov property assumption prevents parameter increasing?
만약 $Z_2 \leftarrow Z_1, x \leftarrow z_1$ 와 같은 경우를assume, parameter가 ϕ_1, ϕ_2 로 늘어남!
Markov property² $q_{\phi_2}(z_2|z_1)$ 이 ϕ_2 에만 영향을 받을 수 있도록 함.

$$\Theta = \{\theta_1, \theta_2, \dots, \theta_T\}$$

\Rightarrow Model 2T P.D with Neural Net? (What if $T=1000$? too many NNs for modeling probability distributions.)

$$\phi = \{\phi_1, \phi_2, \dots, \phi_T\}$$

Issue: In DDPM all this process is done by "single" neural net!
Does not matter which value the T is.

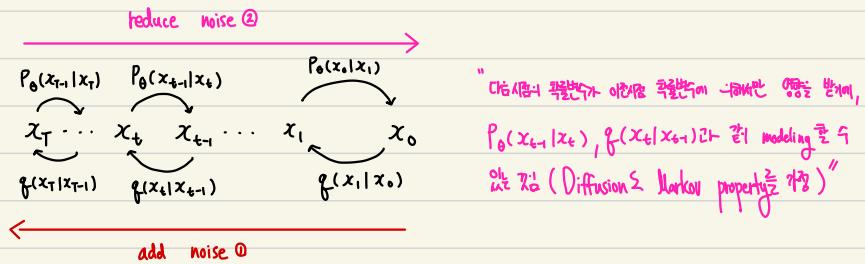
* Hierarchical VAE to Diffusion

Main changes?



1. $X \sim z$ 차원과 $Z \sim z$ 차원을 일치시키기
2. 고정 조건을 따르는 노드를 변화에 추가

Diffusion Model Overview



$- X_0$: observed variable, $x_1 \dots x_T$: latent variable (X dim = Z dim인 경우 X 를 통한)

$- q$ has no ϕ \rightarrow sample x_t from $q(x_t|x_{t-1}) \rightarrow$ fixed gaussian distribution, no need for parameter.

* Two steps of diffusion model : An Overview

① Diffusion process : add noise

- 마지막 확률 X_T (마지막 시점에서 관찰되는)가 $N(x_T; 0, I)$ 를 따르도록 해야함.: complete noise

- How?

$$q(x_t|x_{t-1}) = N(x_t; \sqrt{1-\beta_t} x_{t-1}, \beta_t I), \quad 1 \leq t \leq T, \quad \underbrace{\beta = \{\beta_1, \beta_2, \dots, \beta_T\}}_{\text{Deciding } \beta \text{ is called "noise scheduling"}}, \text{ is hyperparameter}$$

- T 를极大地大きく, β 증가하면 $P(x_T) \approx N(x_T; 0, I)$ 가 됨. (이유는 뒤의)

- How to sample x_t from $g(x_t | x_{t-1}) = N(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I)$? : Reparameterization trick!

$$\epsilon \sim N(\epsilon; 0, I)$$

$$x_t = \sqrt{1-\beta_t} x_{t-1} + \sqrt{\beta_t} \epsilon$$

Scale down x_{t-1} data ↪ add noise ↪

- Scale down? why?

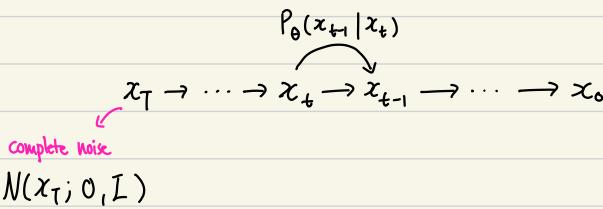
$$\epsilon \sim N(\epsilon; 0, I)$$

$$x_t = x_{t-1} + \sqrt{\beta_t} \epsilon \quad \dots \textcircled{1}$$

$$x_t = \sqrt{1-\beta_t} x_{t-1} + \sqrt{\beta_t} \epsilon \quad \dots \textcircled{2}$$

① ← 봄백대로 scale down으로 풀어보면 $g(x_{t-1} | x_{t-2}) \rightarrow g(x_t | x_{t-1}) \rightarrow g(x_{t+1} | x_t) \rightarrow$ 끝으로 같은 그 뿐의 차이가 있지.
 $g(x_t | x_{t-1})$ ← $g(x_{t+1} | x_t)$ ← 봄백로 scale로 풀어보기 위해 ②와 같이 reparameterization trick Aon x_{t-1} 의 확률 downscaling 한다.

② Reverse Diffusion Process : Denoising



$$N(x_T; 0, I)$$

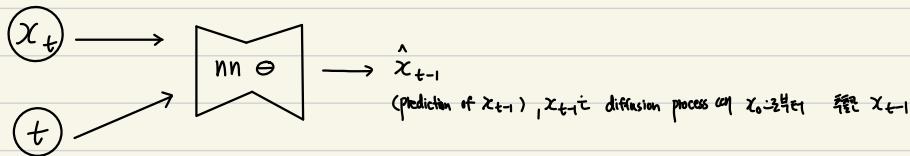
- Denoising for T steps = need T neural nets? ✗, $T=1000$ is too many...

- Use the property that in Diffusion model, $\dim_{\text{(observed variable)}} x_0 = \dim_{\text{(latent variable)}} x_1, x_2, \dots, x_T$

- 각 step을 통해 어떤 step의 input, output이 차이 같고 같은 신경망 구조를 가져올 수 있다.

- 훈련은 신경망 구조를 고지하되 시점 "t" 정보를 제공해줘야 함. (또 사용해서 노드 자체로 신경망 알 수 있도록!)

Fig of NN for Denoising at timestep T.



- NN does not outputs distribution, how to model $P_\theta(x_{t-1}|x_t)$?

- let the output of NN is a "u" of $P_\theta(x_{t-1}|x_t)$ where P is Gaussian Distribution

$$\hat{x}_{t-1} = \text{NN}(x_t, t; \theta)$$

$$P_\theta(x_{t-1}|x_t) = N(x_{t-1}; \hat{x}_{t-1}, I)$$

* Training Diffusion Models : ELBO

- Diffusion model also has latent variable ($Z : z_1, z_2 \dots z_T$) , marginalization for $P_0(x)$ induces \int which leads to log-Sum form for log-likelihood for single data x , making it hard to optimize \Rightarrow Use ELBO for maximizing log-likelihood
- Maximizing $\text{ELBO}(x; \theta, \phi)$ leads to maximizing $\log P_0(x)$ (Jensen's Inequality) ... ①
- Find ELBO's approximation in 3 steps (①, ②, ③)

① Sample $\{x_1, x_2 \dots x_T\}$ from $x_0 : T$

② Sample two latent variables

③ Sample only one

* Step ① : T Sampling method

- ELBO in VAE to ELBO for Diffusion

$$\text{ELBO}(x; \theta, \phi) = \int q_{\phi}(z|x) \log \frac{p_{\theta}(x,z)}{q_{\phi}(z|x)} dz$$

$$= E_{q_{\phi}(z|x)} \left[\log \frac{p_{\theta}(x,z)}{q_{\phi}(z|x)} \right]$$

3 main changes

① observed variable $x \rightarrow x_0$

② latent variable $z \rightarrow \{x_1, x_2, \dots, x_T\}$

③ No need for ϕ since $\{x_1, x_2, \dots, x_T\}$ are sampled from "fixed" Gaussian Distribution

$$\text{ELBO}(x_0; \theta) = E_{q(x_1, x_2, \dots, x_T | x_0)} \left[\log \frac{p_{\theta}(x_0, x_1, x_2, \dots, x_T)}{q(x_1, x_2, \dots, x_T | x_0)} \right]$$

Change of denotation $x_{0:T} = x_0, x_1, \dots, x_T$

$$\text{ELBO}(x_0; \theta) = E_{q(x_{1:T} | x_0)} \left[\log \frac{p_{\theta}(x_{0:T})}{q(x_{1:T} | x_0)} \right] \quad (\text{ELBO of Diffusion model})$$

Ⓐ Jensen's Inequality (why $\log p_{\theta}(x_0) \geq \text{ELBO}(x_0; \theta)$ in diffusion models)

$$\begin{aligned} \log p_{\theta}(x_0) &= \log \int p_{\theta}(x_0, z) dz \quad (\text{marginalization of } z) \\ &= \log \int p_{\theta}(x_0, x_{1:T}) dx_{1:T} \quad (\text{In diffusion, } z = \{x_1, x_2, \dots, x_T\} = x_{1:T}) \\ &= \log \int p_{\theta}(x_{0:T}) dx_{1:T} \end{aligned}$$

$$= \log \int p_{\theta}(x_{0:T}) \frac{q(x_{1:T} | x_0)}{q(x_{1:T} | x_0)} dx_{1:T} \quad \stackrel{?}{>} 1$$

$$= \log E_{q(x_{1:T} | x_0)} \left[\frac{p_{\theta}(x_{0:T})}{q(x_{1:T} | x_0)} \right] \quad \text{Details will be dealt in last page}$$

$$\geq E_{q(x_{1:T} | x_0)} \left[\log \frac{p_{\theta}(x_{0:T})}{q(x_{1:T} | x_0)} \right] \quad (\text{Jensen's Inequality})$$

* Step 1 cont'd : Expansion of ELBO for Diffusion

$$\text{ELBO}(x_0; \theta) = E_{q(x_{1:T}|x_0)} \left[\log \frac{p_\theta(x_0:T)}{q(x_{1:T}|x_0)} \right]$$

↑ ①
↓ ②

"Markov property & chain rule을 이용해서 ①, ② 표시 했지!"

• Reverse-time joint distribution in reverse diffusion $P_\theta(x_{0:T}) \dots \textcircled{1}$

- process of recovering x_0 starting from complete noise $x_T \sim N(x_T; 0, I)$: need to model probability $x_T \rightarrow x_0$ direction

②
- Through markov chain rule $P_\theta(x_{0:T})$ is shown as ..
no θ since $x_T \sim N(x_T; 0, I)$

$$\begin{aligned} P_\theta(x_{0:T}) &= P(x_T) P_\theta(x_{T-1}|x_T) P_\theta(x_{T-2}|x_{T-1}, x_T) \dots P_\theta(x_0|x_{1:T}) \textcircled{A} \\ &= P(x_T) P_\theta(x_{T-1}|x_T) P_\theta(x_{T-2}|x_{T-1}) \dots P_\theta(x_0|x_1) \textcircled{B} \\ &= P(x_T) \prod_{t=1}^T P_\theta(x_{t-1}|x_t) \end{aligned}$$

* Markov property & chain rule

① Joint probability is conditional probability if θ is known. (chain rule) $P(A|B)P(B) = P(A)$
For two variables A, B

$$P(A, B) = P(A|B)P(B) = P(B|A)P(A)$$

For n variables x_1, x_2, \dots, x_n

$$P(x_{1:n}) = P(x_1)P(x_2|x_1)P(x_3|x_2, x_1) \dots P(x_n|x_{n-1}, \dots, x_1)$$

↓
markovassumption markov chain rules

② Reverse process \rightarrow forward process

$$P(x_{t-1}|x_0, x_1, \dots, x_{t-2}) = \underbrace{P(x_{t-1}|x_t)}_{\text{reverse process}} \underbrace{P(x_t|x_0, x_1, \dots, x_{t-2})}_{\text{forward process}}$$

• Forward process probability $q(x_{1:T}|x_0) \dots \textcircled{2}$

$$q(x_{0:T}) = q(x_0) \times q(x_1|x_0) \times q(x_2|x_1, x_0) \dots q(x_T|x_{0:T-1}) \quad (\text{chain rule})$$

$$= q(x_0) \times \underbrace{q(x_1|x_0) \times q(x_2|x_1) \dots q(x_T|x_{T-1})}_{\text{markov property}}$$

Since $q(x_{0:T}) = q(x_0) \times q(x_{1:T}|x_0)$

$$q(x_{1:T}|x_0) = q(x_1|x_0) \times q(x_2|x_1) \dots q(x_T|x_{T-1})$$

$$= \prod_{t=1}^T q(x_t|x_{t-1})$$

$$\text{ELBO}(x_0; \theta) = E_{q(x_{1:T}|x_0)} \left[\log \frac{p_\theta(x_{0:T})}{\pi_{t=1}^T q(x_t|x_{t-1})} \right]$$

$$= E_{q(x_{1:T}|x_0)} \left[\log \frac{p(x_T) \prod_{t=1}^T p_\theta(x_{t-1}|x_t)}{\prod_{t=1}^T q(x_t|x_{t-1})} \right]$$

$$= E_{q(x_{1:T}|x_0)} \left[\underbrace{\log \prod_{t=1}^T p_\theta(x_{t-1}|x_t)}_{\textcircled{1}} + \log \frac{p(x_T)}{\prod_{t=1}^T q(x_t|x_{t-1})} \right] \quad \not \ni \text{not containing } \theta$$

Since only $\textcircled{1}$ is related to θ (reminder: our purpose is finding $\theta = \arg \max_{\theta} \text{ELBO}(x_0; \theta)$)

The objective function $J(\theta)$ is as follows

$$J(\theta) = E_{q(x_{1:T}|x_0)} \left[\log \prod_{t=1}^T p_\theta(x_{t-1}|x_t) \right]$$

$$= E_{q(x_{1:T}|x_0)} \left[\sum_{t=1}^T \log p_\theta(x_{t-1}|x_t) \right]$$

And $J(\theta)$ value can be approximated with "monte carlo"!

how?

- Sample $x_{1:T}$ n times : $N \times (x_{1:T})$

- Mean the value of $\sum_{t=1}^T \log p_\theta(x_{t-1}|x_t)$: $\frac{1}{N} \sum_{n=1}^N \sum_{t=1}^T \log p_\theta(x_{t-1}|x_t)$



$n=1$ (sample just 1 $x_{1:T}$)

$x_{1:T} \sim q(x_{1:T}|x_0)$ (sampling $x_{1:T}$ from x_0 with forward diffusion process)

$J(\theta) \approx \sum_{t=1}^T \log p_\theta(x_{t-1}|x_t)$ (calculate $\log p_\theta(x_{t-1}|x_t)$ for each timestep t and sum.
since $n=1$, no need for averaging)

- How to calculate $\log P_{\theta}(x_{t-1} | x_t)$

$$\hat{x}_{t-1} = \text{NeuralNet}(x_t, t; \theta)$$

$$P_{\theta}(x_{t-1} | x_t) = N(x_{t-1}; \hat{x}_{t-1}, I)$$

\Rightarrow Probability density value of x_{t-1} in Gaussian Distribution where mean vector is \hat{x}_{t-1} and Covariance Matrix is I .

- Expansion of $J(\theta)$

$$J(\theta) \approx \sum_{t=1}^T \log P_{\theta}(x_{t-1} | x_t)$$

$$= \sum_{t=1}^T \log N(x_{t-1}; \hat{x}_{t-1}, I)$$

$$= \sum_{t=0}^{T-1} \log N(x_t; \hat{x}_t, I)$$

$$= \sum_{t=0}^{T-1} \log \frac{1}{\sqrt{(2\pi)^D |I|}} e^{-\frac{1}{2} (x_t - \hat{x}_t)^T I^{-1} (x_t - \hat{x}_t)} \quad (|I| = 1, I^{-1} = I)$$

$$= \sum_{t=0}^{T-1} \left(-\frac{1}{2} (x_t - \hat{x}_t)^T (x_t - \hat{x}_t) + \log \frac{1}{\sqrt{(2\pi)^D}} \right)$$

$$= -\frac{1}{2} \sum_{t=0}^{T-1} (x_t - \hat{x}_t)^T (x_t - \hat{x}_t) + T \log \frac{1}{\sqrt{(2\pi)^D}}$$

\rightarrow does not contain θ

$$\therefore J(\theta) \approx -\frac{1}{2} \sum_{t=0}^{T-1} (x_t - \hat{x}_t)^T (x_t - \hat{x}_t) \quad (\|X\|^2 = X^T X)$$

$$\approx -\frac{1}{2} \sum_{t=0}^{T-1} \|x_t - \hat{x}_t\|^2$$

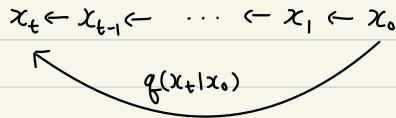
* Summary of ELBO approximation with "T sampling"

- ① Sample $x_{1:T}$ from x_0 (forward diffusion process)
- ② Apply "same" neural network T times for x_t prediction \hat{x}_t (denoising)
- ③ calculate $\|x_t - \hat{x}_t\|^2$ for each time step and sum $\Rightarrow J(\theta)$

issue) Requires T times diffusing and T times denoising? \Rightarrow Cost... what if $T=1000$?!
Need a way to reduce sampling and denoising times!

* Step 2 : 2 times Sampling

- $q(x_t|x_0)$ 을 계산하고 구할 수 있다는 성질 이용 $\Leftrightarrow x_0$ 에 노이즈를 1번만 추가해서 앞부터 시간 t 에 대해 x_t 를 샘플링 할 수 있다.



$$q(x_t|x_0) = N(x_t; \sqrt{a_t}x_0, (I - a_t)I) \quad (\text{where } a_t = 1 - \beta_t, \bar{a}_t = a_t a_{t-1} \dots a_1)$$

* page 20th $P(x_T) = N(x_T; 0, I)$ 인 이유?

for timestep T , $\beta = \{\beta_1, \beta_2, \dots, \beta_T\}$ adjust β well so that $\bar{a}_T \approx 0$

(make $\beta_1, \beta_2, \dots, \beta_{1000}$ linearly increasing from 0.0001 to 0.01, for $T=1000$)

: $\bar{a}_T \rightarrow 0$ 이므로 \Rightarrow noise scheduler가 T 를 증가한다 $\Rightarrow x_T \sim N(x_T; 0, I)$ 를 만족함.

- ELBO expansion in 2 times Sampling

$$\text{ELBO}(x_0; \theta) = E_{q(x_{1:T}|x_0)} \left[\log \frac{P(x_T) \prod_{t=1}^T P_\theta(x_{t-1}|x_t)}{\prod_{t=1}^T q(x_{t-1}|x_{t-1})} \right]$$

→ does not contain θ

$$= E_{q(x_{1:T}|x_0)} \left[\log \prod_{t=1}^T P_\theta(x_{t-1}|x_t) + \log \frac{P(x_T)}{\prod_{t=1}^T q(x_t|x_{t-1})} \right]$$

∴ The objective function $J(\theta)$ is ...

$$\begin{aligned} J(\theta) &= E_{q(x_{1:T}|x_0)} \left[\log \prod_{t=1}^T P_\theta(x_{t-1}|x_t) \right] \\ &= E_{q(x_{1:T}|x_0)} \left[\sum_{t=1}^T \log P_\theta(x_{t-1}|x_t) \right] \end{aligned}$$

\dots linearity of expectation ①

$$= \sum_{t=1}^T E_{q(x_{t-1}, x_t|x_0)} \left[\log P_\theta(x_{t-1}|x_t) \right] \dots \text{Expectation of related variables } ②$$

A Linearity of Expectation

$$\begin{aligned}
 E_{p(x,y)}[ax+by] &= \iint (ax+by) p(x,y) dx dy \\
 &= \iint ax p(x,y) dx dy + \iint by p(x,y) dx dy \\
 &= a \iint x p(x,y) dx dy + b \iint y p(x,y) dx dy \\
 &= a E_{p(x,y)}[x] + b E_{p(x,y)}[y] \quad (\text{넓은 기대값} = \text{기대값의 합}) \\
 &= a E_{p(x)}[x] + b E_{p(y)}[y] \quad - \textcircled{B}
 \end{aligned}$$

for T variables ... $E_{p(x_1:T)}\left[\sum_{t=1}^T x_t\right] = \sum_{t=1}^T E_{p(x_{1:T})}[x_t]$

B Expectation of related Variables

$f(x) \vdash p(x,y)$ on the expectation = $f(x) \vdash p(x)$ on expectation

$$E_{p(x,y)}[f(x)] = E_{p(x)}[f(x)] : \text{기대값의 내용 } f(x) \text{에 대해 } f(x) \text{가 무관한 확률적인 확률 } p(x,y) \text{를 뺀는 것은 수 있다!}$$

$$\begin{aligned}
 \text{pf)} \quad E_{p(x,y)}[f(x)] &= \iint f(x) p(x,y) dx dy \\
 &= \iint f(x) p(x) p(y|x) dx dy \\
 &= \underbrace{\int f(x) p(x) dx}_{1} \int p(y|x) dy \\
 &= \int f(x) p(x) dx \\
 &= E_{p(x)}[f(x)]
 \end{aligned}$$

$$\therefore E_{p(x_{1:T})}[f(x_t)] = E_{p(x_t)}[f(x_t)]$$

$$\therefore E_{p(x_{1:T})}[f(x_{t+1}, x_t)] = E_{p(x_{t+1}, x_t)}[f(x_{t+1}, x_t)]$$

Thus, the objective function is..

$$J(\theta) = \sum_{t=1}^T E_{q(x_{t+1}, x_t | x_0)} [\log P_\theta(x_{t+1} | x_t)]$$

A B

* How to calculate (approximate) $J(\theta)$?

(B) → Sample from $q(x_{t+1}, x_t | x_0)$: sample just two x_{t+1}, x_t in 2 steps.

$$\textcircled{1} \quad x_{t+1} \sim q(x_{t+1} | x_t)$$

$= N(x_{t+1}; \sqrt{\alpha_{t+1}} x_t, (1-\alpha_{t+1})I) \dots x_{t+1}$ sampling

$$\textcircled{2} \quad x_t \sim q(x_t | x_{t+1})$$

$= N(x_t; \sqrt{1-\beta_t} x_{t+1}, \beta_t I) \dots x_t$ sampling

But we still have (A)...

If T is large, calculation cost $\uparrow \Rightarrow$ need a way to approximate \sum_t

How?

- T 까지 합을 구하는법에 대한 기억과 예제 표현할 수 있다

$t \sim U\{1, T\}$ (discrete uniform distribution, $U(t) = \frac{1}{T}$ for $\forall t$)

$$E_{U(t)}[f(t)] = \sum_{t=1}^T U(t) f(t)$$

$$= \sum_{t=1}^T \frac{1}{T} f(t)$$

$$= \frac{1}{T} \sum_{t=1}^T f(t) \dots \textcircled{1}$$

from $\textcircled{1}$,

$f(t)$ 은 정의된 확률변수 $\sum_{t=1}^T f(t)$ 은 ($t \in 1$ 부터 T 까지 어떤 값이든 가능) $t \sim U\{1, T\}$ 인 $f(t)$ 의 expectation은 $E[f(t)]$.

$$\therefore \sum_{t=1}^T f(t) = T E_{U(t)}[f(t)]$$

따라서, 확률변수의 개수를 쉽게把握할 수 있는 것임.

$$J(\theta) = \sum_{t=1}^T E_{q(x_{t-1}, x_t | x_0)} [\log P_\theta(x_{t-1} | x_t)]$$

A B

B를 $f(t)$ 로 놓으면,

$$\begin{aligned} J(\theta) &= \sum_{t=1}^T f(t) \\ &= T E_{u(t)} [f(t)] \\ &= T E_{u(t)} \left[E_{q(x_{t-1}, x_t | x_0)} [\log P_\theta(x_{t-1} | x_t)] \right] \end{aligned}$$

C

②는 2개의 개별값이 중복되어 있음. "Monte carlo"로 (샘플 크기 1개) $u(t)$ 에서 추출, $q(x_{t-1}, x_t | x_0)$ 에서 (x_{t-1}, x_t) 을 Sampling 한다

근사값 계산.

\Leftrightarrow

$$t \sim U\{1, T\}$$

①

$$x_{t-1} \sim q(x_{t-1} | x_0)$$

②

$$x_t \sim q(x_t | x_{t-1})$$

$$J(\theta) \approx T \underbrace{\log P_\theta(x_{t-1} | x_t)}_{\text{③}}$$

Where $P_\theta(x_{t-1} | x_t)$ is as follows :

$$\hat{x}_{t-1} = \text{NeuralNet}(x_t, t; \theta)$$

$$P_\theta(x_{t-1} | x_t) = \mathcal{N}(x_{t-1}; \hat{x}_{t-1}, I)$$

$$\begin{aligned}
 \textcircled{A}: J(\theta) &\approx T \log P_{\theta}(x_{t-1} | x_t) = T \log N(x_{t-1}; \hat{x}_{t-1}, I) \\
 &= T \log \frac{1}{\sqrt{(2\pi)^n |I|}} e^{\left\{-\frac{1}{2}(x_{t-1} - \hat{x}_{t-1})^T I^{-1} (x_{t-1} - \hat{x}_{t-1})\right\}} \quad (|I|=1, I^{-1}=I) \\
 &= T \left\{ -\frac{1}{2}(x_{t-1} - \hat{x}_{t-1})^T (x_{t-1} - \hat{x}_{t-1}) + \log \frac{1}{\sqrt{(2\pi)^n}} \right\} \\
 &\qquad \qquad \qquad \text{constant}
 \end{aligned}$$

$$\therefore J(\theta) \approx -\frac{T}{2} \|x_{t-1} - \hat{x}_{t-1}\|^2$$

$$\left(\underset{\theta}{\operatorname{argmax}} J(\theta) \approx \underset{\theta}{\operatorname{argmin}} \|x_{t-1} - \hat{x}_{t-1}\|^2 \right)$$

Summary for 2 sampling method

$J(\theta)$ can be calculated by $t \sim U\{1, T\}$ (time sampling from uniform distribution)

$$\begin{aligned}
 \textcircled{A} \quad x_{t-1} &\sim f(x_{t-1} | x_0) \quad (x_{t-1} \text{ sampling from } f(x_{t-1} | x_0)) \\
 x_t &\sim f(x_t | x_{t-1}) \quad (x_t \text{ sampling from } f(x_t | x_{t-1})) \quad \rightarrow \text{forward diffusing process} \\
 \hat{x}_{t-1} &= \text{NeuralNet}(x_t, t; \theta) \rightarrow \text{denoising} \\
 \text{get } &\|x_{t-1} - \hat{x}_{t-1}\|^2
 \end{aligned}$$

$$\Rightarrow J(\theta) \approx -\frac{T}{2} \|x_{t-1} - \hat{x}_{t-1}\|^2$$

* $q(x_t | x_0)$ (How can we do A: Sample x_t directly from x_0)

- the sum of gaussian noise is gaussian noise

let x, y are independently generated gaussian noise, where $z = x + y$

$$x \sim N(u_x, \sigma_x^2)$$

$$y \sim N(u_y, \sigma_y^2)$$

$$z = x + y$$

→ This is the reason of downsampling in forward diffusion process!

$$\text{then, } z \sim N(u_x + u_y, \sigma_x^2 + \sigma_y^2)$$

$$q(x_t | x_{t-1}) = N(x_t; \sqrt{1-\beta_t} x_{t-1}, \beta_t I)$$

$$\text{let } a_t = 1 - \beta_t$$

$$= N(x_t; \sqrt{a_t} x_{t-1}, (1-a_t) I)$$

↓
reparametrization trick

$$(t=t) \quad \epsilon_t \sim N(\epsilon_t; 0, I)$$

A

$$x_t = \sqrt{a_t} x_{t-1} + \sqrt{1-a_t} \epsilon_t$$

$$(t=t-1) \quad \epsilon_{t-1} \sim N(\epsilon_{t-1}; 0, I)$$

B

$$x_{t-1} = \sqrt{a_{t-1}} x_{t-2} + \sqrt{1-a_{t-1}} \epsilon_{t-1}$$

from A, B ...

$$x_t = \sqrt{a_t} x_{t-1} + \sqrt{1-a_t} \epsilon_t$$

$$= \sqrt{a_t} (\sqrt{a_{t-1}} x_{t-2} + \sqrt{1-a_{t-1}} \epsilon_{t-1}) + \sqrt{1-a_t} \epsilon_t$$

$$= \underline{\sqrt{a_t a_{t-1}} x_{t-2} + \sqrt{1-a_t a_{t-1}} \epsilon_{t-1}} + \overline{\sqrt{1-a_t} \epsilon_t}$$

B

$\epsilon_{t-1}, \epsilon_t$ are generated from $N(0, I)$ independently, and B is sum of gaussian noise.

Thus, (B) follows gaussian distribution.

$$(B): \sqrt{a_t - a_t a_{t-1}} \epsilon_{t-1} + \sqrt{1-a_t} \epsilon_t$$

- $\epsilon_t \sim N(0, I)$

$$\epsilon_{t-1} \sim N(0, I)$$

$$\sqrt{a_t - a_t a_{t-1}} \epsilon_{t-1} \sim N(0, (a_t - a_t a_{t-1}) I) \quad (\text{Var}(ax) = a^2 \text{Var}(x))$$

$$\sqrt{1-a_t} \epsilon_t \sim N(0, (1-a_t) I)$$

$$\therefore \sqrt{a_t - a_t a_{t-1}} \epsilon_{t-1} + \sqrt{1-a_t} \epsilon_t \sim N(0, (a_t - a_t a_{t-1}) I + (1-a_t) I) \\ = N(0, (1-a_t a_{t-1}) I)$$

Thus,

$$x_t = \underbrace{\sqrt{a_t a_{t-1}} x_{t-2} + \sqrt{a_t - a_t a_{t-1}} \epsilon_{t-1} + \sqrt{1-a_t} \epsilon_t}_{(B)}$$

$$(B) \sim N(0, (1-a_t a_{t-1}) I)$$

$$\therefore x_t \sim N(\sqrt{a_t a_{t-1}} x_{t-2}, (1-a_t a_{t-1}) I)$$



reparameterization trick

$$\epsilon \sim N(0, I)$$

$$x_t = \sqrt{a_t a_{t-1}} x_{t-2} + \sqrt{1-a_t a_{t-1}} \epsilon$$

repeat this process for $x_{t-2}, x_{t-3}, \dots, x_1, x_0$

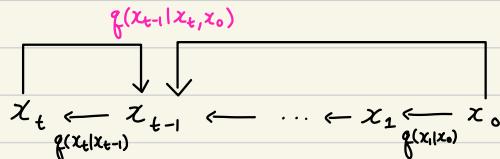


$$x_t = \sqrt{a_t a_{t-1} \dots a_1} x_0 + \sqrt{1-a_t a_{t-1} \dots a_1} \epsilon \\ = \sqrt{a_t} x_0 + \sqrt{1-a_t} \epsilon \quad (a_t = a_t a_{t-1} \dots a_1) \quad (\epsilon \sim N(0, I))$$

$$\therefore f(x_t | x_0) = N(x_t; \sqrt{a_t} x_0, (1-a_t) I)$$

* Step 3 : 1 Sampling for ELBO approx.

$q(x_{t-1} | x_t, x_0)$ will be used for 1 sampling method



$$q(x_{t-1} | x_t, x_0) = \mathcal{N}(x_{t-1}; u_q(x_t, x_0), \mathcal{S}_q^2(t)I)$$

$$\text{where } u_q(x_t, x_0) = \frac{\sqrt{a_t}(1-\bar{a}_{t-1})x_t + \sqrt{\bar{a}_{t-1}}(1-a_t)x_0}{1-\bar{a}_t} \quad (\text{mean vector})$$

$$(a_t = 1 - \beta_t, \bar{a}_t = a_t a_{t-1} \dots a_1)$$

$$\mathcal{S}_q^2(t) = \frac{(1-a_t)(1-\bar{a}_{t-1})}{1-\bar{a}_t} \quad (\text{covariance matrix})$$

$q(x_{t-1} | x_t, x_0)$ 의 mean vector는 x_t 와 x_0 의 선형합으로 표현 (내분점에 가까운) (x_0 은 고정값 x_t 는 Sampling 값이다.)

$\mathcal{S}_q^2(t)$ 는 하이퍼파라미터로 정해지는 scalar임.

forward diffusion process

Sampling

- Transformation of $J(\theta)$ for 1 sampling method
From ELBO we get $J(\theta)$

$$J(\theta) = \frac{\sum_{t=1}^T E_{q(x_{t-1}, x_t | x_0)} [\log P_\theta(x_{t-1} | x_t)]}{\bar{f}_\theta}$$

$$= T E_{u(t)} [E_{q(x_{t-1}, x_t | x_0)} [\log P_\theta(x_{t-1} | x_t)]]$$

$\Leftrightarrow J_0 = T E_{u(t)} [J_0]$: J_0 에 θ 를 포함하지 않은 항은 J_0 에 추가해도 $J(\theta)$ 에 대한 최적화 문제는 변하지 않음.

$q(x_{t-1} | x_t, x_0)$ 은 θ 를 갖기 때문에 이와 관련된 θ 를 포함하지 않은 항은 J_0 에 포함될 것이다!

$$\begin{aligned} \underset{\theta}{\operatorname{argmax}} J_0 &= \underset{\theta}{\operatorname{argmax}} (J_0 - E_{q(x_{t-1}, x_t | x_0)} [\log q(x_{t-1} | x_t, x_0)]) \\ &\quad \textcircled{A} \quad \text{add constant not containing } \theta \\ &= \underset{\theta}{\operatorname{argmax}} (E_{q(x_{t-1}, x_t | x_0)} [\log P_\theta(x_{t-1} | x_t)] - E_{q(x_{t-1}, x_t | x_0)} [\log q(x_{t-1} | x_t, x_0)]) \\ &= \underset{\theta}{\operatorname{argmax}} E_{q(x_{t-1}, x_t | x_0)} [\log P_\theta(x_{t-1} | x_t) - \log q(x_{t-1} | x_t, x_0)] \\ &= \underset{\theta}{\operatorname{argmax}} E_{q(x_{t-1}, x_t | x_0)} \left[\log \frac{P_\theta(x_{t-1} | x_t)}{q(x_{t-1} | x_t, x_0)} \right] \\ &\quad \textcircled{B} \quad (J_0 \text{에 } \textcircled{A} \text{를 추가해 변형한 } J_1) \end{aligned}$$

$\underset{\theta}{\operatorname{argmax}} J_0 = \underset{\theta}{\operatorname{argmax}} J_1$ since the pink term \textcircled{A} does not contain θ .

$\therefore J(\theta) = T E_{u(t)} [J_1]$ 로 놓고 풀어짐!

\Rightarrow 여기서 목적함수에 one sampling ELBO approximation을 구하기 위한 $q(x_{t-1} | x_t, x_0)$ 을 포함시켰다!

• Expansion of J_1 & Maximize J_1

$$J_1 = E_{q(x_{t-1}, x_t | x_0)} \left[\log \frac{P_\theta(x_{t-1} | x_t)}{q(x_{t-1} | x_t, x_0)} \right]$$

$$= \iint q(x_{t-1}, x_t | x_0) \log \frac{P_\theta(x_{t-1} | x_t)}{q(x_{t-1} | x_t, x_0)} dx_{t-1} dx_t$$

$$(q(x_{t-1}, x_t, x_0) = q(x_0) \times q(x_t | x_0) \times q(x_{t-1} | x_t, x_0))$$

$$q(x_{t-1}, x_t | x_0) = q(x_t | x_0) \times q(x_{t-1} | x_t, x_0)$$

$$= \iint q(x_t | x_0) q(x_{t-1} | x_t, x_0) \log \frac{P_\theta(x_{t-1} | x_t)}{q(x_{t-1} | x_t, x_0)} dx_{t-1} dx_t$$

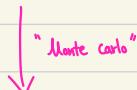
$$= - \int q(x_t | x_0) \int q(x_{t-1} | x_t, x_0) \log \frac{P_\theta(x_{t-1} | x_t)}{q(x_{t-1} | x_t, x_0)} dx_{t-1} dx_t$$



$$D_{KL}(q(x_{t-1} | x_t, x_0) || P_\theta(x_{t-1} | x_t))$$

$$= -E_{q(x_t | x_0)} [D_{KL}(q(x_{t-1} | x_t, x_0) || P_\theta(x_{t-1} | x_t))]$$

: J_1 is expressed as a expectation of D_{KL} of $q(x_t | x_0)$


"Monte carlo"

Sample just one x_t from $q(x_t | x_0)$ and calculate KL to get J_1

$$J_1 = -E_{q(x_{t-1}|x_t, x_o)} [D_{KL}(q(x_{t-1}|x_t, x_o) || P_\theta(x_{t-1}|x_t))]$$

Maximize J_1 ?

$$\Leftrightarrow D_{KL} = 0$$

$$\Leftrightarrow q(x_{t-1}|x_t, x_o) = P_\theta(x_{t-1}|x_t)$$

$$N(x_{t-1}; u_q(x_t, x_o), \sigma^2_q(t)I) \quad \xrightarrow{\text{Def}} \quad N(x_{t-1}; \hat{x}_{t-1}, I)$$

$\downarrow q(x_{t-1}|x_t, x_o) = \sigma^2_q(t) \text{ 를 대체하기 위해 아래와 같은 바꿈}$
 $N(x_{t-1}; \hat{x}_{t-1}, \sigma^2_q(t)I)$
 $(\hat{x}_{t-1} = \text{NeuralNet}(x_t, t; \theta))$

• Change of denotation : $\text{NeuralNet}(x_t, t; \theta) \rightarrow u_\theta(x_t, t)$

$$\therefore P_\theta(x_{t-1}|x_t) = N(x_{t-1} | u_\theta(x_t, t), \sigma^2_\theta(t)I)$$

• Review : D_{KL} between 2 gaussian distribution : 두 개의 노드 KL 계산으로 표현 가능.

$$\text{let } q(z) = N(z; u_1, \sigma^2_1 I), p(z) = N(z; u_2, \sigma^2_2 I), \text{ where } z \in \mathbb{R}^n$$

$$D_{KL}(q||p) = -\frac{1}{2} \sum_{h=1}^H \left(1 + \log \frac{\sigma_{1,h}^2}{\sigma_{2,h}^2} - \frac{(u_{1,h} - u_{2,h})^2}{\sigma_{2,h}^2} - \frac{\sigma_{1,h}^2}{\sigma_{2,h}^2} \right)$$

$$N(x_{t-1}; u_f(x_t, x_0), \sigma_f^2(t)I) \quad N(x_t; u_\theta(x_t, t), \sigma_\theta^2(t)I)$$

$$D_{KL} = D_{KL}(f(x_{t-1}|x_t, x_0) || p_\theta(x_{t-1}|x_t))$$

$$\begin{aligned} &= -\frac{1}{2} \sum_{h=1}^H \left(1 + \log \frac{\sigma_f^2(t)}{\sigma_\theta^2(t)} - \frac{(u_{f,h}(x_t, x_0) - u_{\theta,h}(x_t, t))^2}{\sigma_\theta^2(t)} - \frac{\sigma_\theta^2(t)}{\sigma_f^2(t)} \right) \quad \sigma_f^2(t) = \frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} \Rightarrow \text{"scalar"} \\ &= \frac{1}{2} \sum_{h=1}^H \frac{(u_{f,h}(x_t, x_0) - u_{\theta,h}(x_t, t))^2}{\sigma_f^2(t)} \\ &= \frac{1}{2\sigma_f^2(t)} \sum_{h=1}^H (u_{f,h}(x_t, x_0) - u_{\theta,h}(x_t, t))^2 \end{aligned}$$

$$= \frac{1}{2\sigma_f^2(t)} \|u_\theta(x_t, t) - u_f(x_t, x_0)\|^2 \quad \left(= \frac{1}{2\sigma_f^2(t)} (u_\theta(x_t, t) - u_f(x_t, x_0))^T (u_\theta(x_t, t) - u_f(x_t, x_0)) \right)$$

$$\therefore J(\theta) = -T E_{u(t)} [E_{f(x_t|x_0)} [D_{KL}(f(x_{t-1}|x_t, x_0) || p_\theta(x_{t-1}|x_t))]]$$

$$= -T E_{u(t)} [E_{f(x_t|x_0)} \left[\frac{1}{2\sigma_f^2(t)} \|u_\theta(x_t, t) - u_f(x_t, x_0)\|^2 \right]]$$

J(θ) \rightarrow Loss (- $\frac{1}{2}\| \cdot \|^2$) \oplus remove $\frac{1}{2}, T$

$$\text{Loss}(x_0; \theta) = E_{u(t)} [E_{f(x_t|x_0)} \left[\frac{1}{2\sigma_f^2(t)} \|u_\theta(x_t, t) - u_f(x_t, x_0)\|^2 \right]]$$

"Monte carlo" approximation for Loss function (Sample 1)

only sample x_t (1 time!)

from forward diffusion process...

$$t \sim u(t)$$

$$x_t \sim f(x_t|x_0)$$

$$\text{Loss}(x_0; \theta) \approx \frac{1}{\sigma_f^2(t)} \|u_\theta(x_t, t) - u_f(x_t, x_0)\|^2$$

Train neural network(update θ) to reduce $\text{Loss}(x_0; \theta) \Rightarrow$ make prediction of neural network at timestep t

$$\hat{x}_{t-1} = u_\theta(x_t, t) \text{ closer to } u_f(x_t, x_0).$$

$\hookrightarrow x_t, x_0$ 사이에 내분점

* Summary of 1 sampling method

$$\text{Loss}(x_0; \theta) = E_{u(t)} [E_{q(x_t|x_0)} \left[\frac{1}{\delta_q^2(t)} \|u_\theta(x_t, t) - u_f(x_t, x_0)\|^2 \right]]$$

calculate loss with monte carlo ($n=1$)

for a given observed variable x_0 ,
 $t \sim U\{1, T\}$

$$x_t \sim q(x_t|x_0)$$

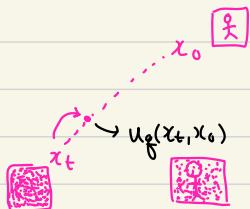
$$\text{Loss}(x_0; \theta) \approx \frac{1}{\delta_q^2(t)} \|u_\theta(x_t, t) - u_f(x_t, x_0)\|^2$$

\Rightarrow update θ to minimize $\text{Loss}(x_0; \theta)$

\Leftrightarrow make $\hat{x}_{t-1} = \text{NeuralNet}(x_t, t; \theta) = u_\theta(x_t, t)$ close to $u_f(x_t, x_0)$
↳ dividing point

Intuition?

$\hat{x}_{t-1} = \text{NeuralNet}(x_t, t)$ (x_t 로부터 훈련 denoising한 값이 x_t, x_0 에 대한(과 비교) 가까워지도록
 $x_t \leftarrow x_{t-1} < \dots < \dots x_0$ Neural Network를 업데이트)



\Rightarrow dividing point should be close to t step denoising output...

① x_0 (Random Sampling x_0 from training sample...)

② 임의 시점 t 쭉

③ x_t sampling ($x_t \sim q(x_t|x_0)$)

④ x_t, t neural net을 통해 예측(denoising) \hat{x}_{t-1} x_{t+1}, x_0 의 dividing point가 가까워지도록 Neural Net parameter θ update

* Training algorithm of Diffusion model

Iterate :

1. Random sample x_0

2. $t \sim U\{1, T\}$: sample t from uniform distribution

3. $\epsilon \sim N(0, I)$

4. $x_t \sim q(x_t | x_0) : x_t = \sqrt{\alpha_t} x_0 + \sqrt{1 - \alpha_t} \epsilon$

5. $U_f(x_t, x_0) = \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})x_t + \sqrt{\bar{\alpha}_{t-1}}(1 - \alpha_t)x_0}{1 - \bar{\alpha}_t}$] mean vector & covariance for $q(x_{t-1} | x_t, x_0)$

6. $S_f^2(t) = \frac{(1 - \alpha_t)(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t}$

7. $\text{Loss}(x_0; \theta) = \frac{1}{S_f^2(t)} \|U_\theta(x_t, t) - U_f(x_t, x_0)\|^2$

8. Get $\frac{\partial}{\partial \theta} \text{Loss}(x_0; \theta)$ and update θ (gradient descent)

⇒ Till now the neural network directly predicts $U_\theta(x_t, t)$... using $U_f(x_t, x_0)$ as training data

Is there other thing that neural network can predict?

\downarrow
 x_t 시점에서 x_0 방향으로
각도 예측가 제작한 내부집.

* Different things that neural network predicts

- Neural Network that predicts x_o (전부 x_o 뿐만 신경망)

신경망을 통해 가깝게 예측해야 하는 값은 $U_f(x_t, x_o)$ 이다.

$$U_f(x_t, x_o) = \frac{\sqrt{a_t}(1-\bar{a}_{t-1})x_t + \sqrt{\bar{a}_{t-1}}(1-a_t)x_o}{1-\bar{a}_t} \quad \text{... ①}$$

A focus here!

$U_\theta(x_t, t) =$ 식은 $U_f(x_t, x_o)$ + formation 를 더해 줘

$$U_\theta(x_t, t) = \frac{\sqrt{a_t}(1-\bar{a}_{t-1})x_t + \sqrt{\bar{a}_{t-1}}(1-a_t)\hat{x}_\theta(x_t, t)}{1-\bar{a}_t} \quad \text{... ②}$$

B make this part what the Neural Network directly predicts

The only difference between ①, ② is ○ part (A,B). Thus, making A,B closer is making ①, ② closer.

$$\therefore D_K(f(x_{t-1}|x_t, x_o) \| P_\theta(x_{t-1}|x_t))$$

$$= \frac{1}{2\delta_f^2(t)} \| U_\theta(x_t, t) - U_f(x_t, x_o) \|^2$$

$$= \frac{1}{2\delta_f^2(t)} \left(\frac{\sqrt{\bar{a}_{t-1}}(1-a_t)}{1-\bar{a}_t} \right)^2 \| \hat{x}_\theta(x_t, t) - x_o \|^2$$

↳ make neural network restore original image from time t (x_t)

- Neural Network that predicts noise

$$q(x_t | x_0) = \mathcal{N}(x_t; \sqrt{\bar{a}_t} x_0, (1-\bar{a}_t) I)$$

↓ reparameterization trick

$$\epsilon \sim \mathcal{N}(0, I)$$

$$x_t = \sqrt{\bar{a}_t} x_0 + \sqrt{1-\bar{a}_t} \epsilon$$

$$\Leftrightarrow x_0 = \frac{x_t - \sqrt{1-\bar{a}_t} \epsilon}{\sqrt{\bar{a}_t}} \dots \textcircled{A}$$

\textcircled{A} 는 앞페이지나 ①에 대입 후 살펴

$$\begin{aligned} u_g(x_t, x_0) &= \frac{\sqrt{\bar{a}_t}(1-\bar{a}_{t-1})x_t + \sqrt{\bar{a}_{t-1}}(1-\bar{a}_t)x_0}{1-\bar{a}_t} \\ &= \frac{1}{\sqrt{\bar{a}_t}} \left(x_t - \frac{1-\bar{a}_t}{\sqrt{1-\bar{a}_t}} \epsilon \right) \dots \textcircled{B} \end{aligned}$$

make $u_\theta(x_t, t)$ format corresponding to ③

$$u_\theta(x_t, t) = \frac{1}{\sqrt{\bar{a}_t}} \left(x_t - \frac{1-\bar{a}_t}{\sqrt{1-\bar{a}_t}} \epsilon_\theta(x_t, t) \right) \dots \textcircled{C}$$

→ the term that Neural Network directly predicts!

The only difference between ②, ③ is ③ part (i, ii). Thus, making i, ii closer is making ③, ④ closer.

$$\begin{aligned} \therefore D_{KL}(q(x_{t-1} | x_t, x_0) || p_\theta(x_{t-1} | x_t)) &= \frac{1}{2\delta_\theta^2(t)} \| u_\theta(x_t, t) - u_g(x_t, x_0) \|^2 \\ &= \frac{1}{2\delta_\theta^2(t) (1-\bar{a}_t) \bar{a}_t} \| \epsilon_\theta(x_t, t) - \epsilon \|^2 \end{aligned}$$

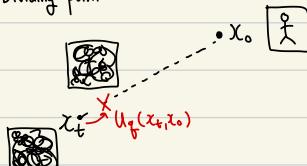
↳ make neural network predict gaussian noise ϵ that has been added from $x_0 \rightarrow x_t$, $x_t \sim q(x_t | x_0)$ for reparameterization trick.

* Summary of what model predicts

Iterate ...

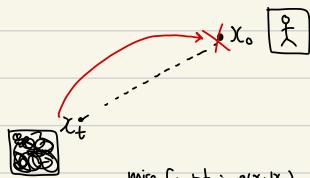
- 1) x_0 sample
- 2) $t \sim U[1, T]$ t sample (uniform distribution)
- 3) $q(x_t | x_0)$ x_t sample ($x_t \sim N(x_t; \sqrt{a_t}x_0, (1-\bar{a}_t)\mathbb{I})$) \rightarrow forward diffusion process
- i) $\epsilon \sim N(0, \mathbb{I})$
- ii) $x_t = \sqrt{a_t}x_0 + \sqrt{1-\bar{a}_t}\epsilon$

① Dividing point

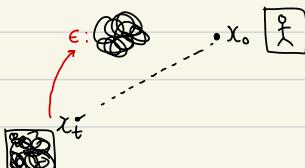


$\rightarrow X$: prediction point (what the neural network predicts)

② x_0 restore



③ Gaussian noise ϵ prediction



"Method ③ is the best method
in experiment"

4) Make Loss function based on these three.

5) Get $\frac{\partial}{\partial \theta} \text{Loss}(x_0, \theta)$ and update θ with gradient descent.

* Sampling (Generating) from Diffusion Models.

$$P_\theta(x_{t-1} | x_t) = N(x_{t-1}; u_\theta(x_t, t), \sigma_f^2(t)I)$$

↓ R.T

$$\epsilon \sim N(0, I)$$

$$x_{t-1} = u_\theta(x_t, t) + \sigma_f(t)\epsilon$$

↳ make $\epsilon = 0$ when $t=1$ (x_1 에서 x_0 예측을 위해 R.T에 사용되는 노드는 $\in A_0^{\frac{1}{2}}$
이지 $X \Rightarrow$ 이어야 결과가 훨씬 더 좋겠대고 큼)

(where

$$u_\theta(x_t, t) = \frac{1}{\sqrt{a_t}} \left(x_t - \frac{1-a_t}{\sqrt{1-\bar{a}_t}} \epsilon_\theta(x_t, t) \right)$$

→ the value that neural network predicts

$$\sigma_f(t) = \sqrt{\frac{(1-a_t)(1-\bar{a}_{t-1})}{1-\bar{a}_t}}$$

Repeat this process starting from $x_T \sim N(0, I)$

• Sample Pseudo code

1) $x_T \sim N(0, I)$ (complete noise)

2) for t in $[T, \dots 1]$:

$$\epsilon \sim N(0, I)$$

if $t=0$: then $\epsilon = 0$

$$\sigma_f(t) = \sqrt{\frac{(1-a_t)(1-\bar{a}_{t-1})}{1-\bar{a}_t}}$$

mean vector for $P_\theta(x_{t-1} | x_t)$

$$x_{t-1} = \frac{1}{\sqrt{a_t}} \left(x_t - \frac{1-a_t}{\sqrt{1-\bar{a}_t}} \epsilon_\theta(x_t, t) \right) + \frac{\sigma_f(t)}{\sqrt{1-\bar{a}_t}} \epsilon$$

Neural Network's output

return x_0

* How to get $q(x_{t-1}|x_t, x_0)$?

"Next" "Time"

*Jensen's Inequality

