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 (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 $p_\theta(z|x)$)
- Encoder & Decoder is implemented using Neural Network.

(↓ variational approximation / variational bayes)

* VAE with multiple latent variables (hierarchical variable)

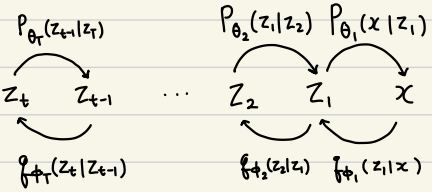
- Make multiple zs hierarchically ⇒ 복잡한 데이터 더욱 표현력 ↑ 표현!
- Current probabilistic variable is only decided by "previous" probabilistic variable ⇒ Markov property

* Markov Property

- 다음 상태($t+1$) 는 2개 전의 상태 (t) 에만 ↑ 1, 그 이전의 상태에는 ↑ 영향

$$P(x_{t+1}=x_{t+1} \mid X_t=x_t, X_{t-1}=x_{t-1} \cdots X_0=x_0) = P(X_{t+1}=x_{t+1} \mid X_t=x_t)$$

- Hierarchical VAE figure



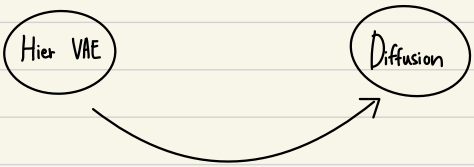
Markov property assumption prevents parameter increasing?
: 만약 z_2가 z_1, x에 대해 모두 영향을 받았다면, parameter가 ϕ_1, ϕ_2로 늘어남!
Markov property로 $q_{\phi_2}(z_2|z_1)$이 ϕ_2에만 영향을 받을 수 있도록 함.

- $\theta = \{\theta_1, \theta_2, \cdots \theta_T\}$
- $\phi = \{\phi_1, \phi_2 \cdots \phi_T\}$

⇒ Model $2T$ P.D with Neural Net? (what if $T=1000$? too many nns for modeling probability distributions.
↳ issue : In DDPM all this process is done by "Single" neural net! Does not matter which value the T is.

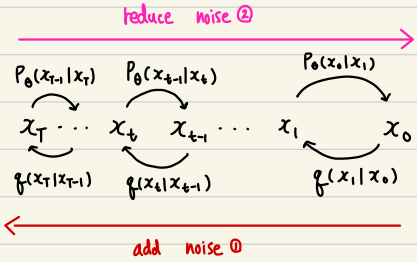
✳ Hierarchical VAE to Diffusion

· Main changes?



1. x의 차원과 z의 차원을 일치시키기
2. 고정된 정규분포를 따르는 노이즈를 인코더에 추가

· Diffusion Model Overview



"다음 시점의 확률분포가 이전시점 확률분포에 너무많은 영향을 받게,
$p_\theta(x_{t-1}|x_t)$, $q(x_t|x_{t-1})$과 같이 modeling 할 수
있을 것임 (Diffusion은 Markov property를 가정)"

- x_0 : observed variable, $x_1 \cdots x_T$: latent variable (x dim = z dim 이기에 표기를 x로 통일)
- q has no ϕ → sample x_t from $q(x_t|x_{t-1})$ → fixed gaussian distribution, no need for parameter.

✳ Two steps of diffusion model : An Overview

① Diffusion process : add noise

- 마지막 시점 x_T (마지막 시점에서의 확률변수)가 $N(x_T; 0, I)$를 따르도록 해야함. : complete noise
- How?

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

Deciding β is called "noise scheduling"

$\beta = \{\beta_1, \beta_2 \cdots \beta_t\}$ is hyperparameter

- T를 충분히 크게 잡고, β 조정을 하면 $p(x_T) \approx N(x_T; 0, I)$가 됨. (이유는 뒤에)

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

$$\epsilon \sim N(\epsilon; 0, I)$$
$$x_t = x_{t-1} + \sqrt{\beta_t}\, \epsilon \cdots ①$$
$$x_t = \sqrt{1-\beta_t}\, x_{t-1} + \sqrt{\beta_t}\, \epsilon \cdots ②$$

$N(\epsilon; 0, I)$ gaussian을 여러번 더하면 ... 위의 식처럼 결국 표준편차 값과 분산이 커짐.

ex) $\epsilon_1 \sim N(\epsilon_1; 0, \alpha I)$ $\epsilon_2 \sim N(\epsilon_2; 0, \beta I)$ / $\epsilon_1 + \epsilon_2 \sim N(\epsilon_1 + \epsilon_2; 0, (\alpha+\beta) I)$

①의 방식대로 scale down하지 않는다면 $q(x_{t-1}|x_{t-2}) \rightarrow q(x_t|x_{t-1}) \rightarrow q(x_{t+1}|x_t)$ 의 분포로 갈수록 그 분산의 값이 커짐.
$q(x_t|x_{t-1})$ 과 $q(x_{t+1}|x_t)$의 분산을 같은 scale로 유지하기 위해 ②과 같이 reparameterization trick 시에 x_{t-1}의 값을 downscaling 함.

② Reverse Diffusion Process : Denoising

$$x_T \rightarrow \cdots \rightarrow x_t \xrightarrow{p_\theta(x_{t-1}|x_t)} x_{t-1} \rightarrow \cdots \rightarrow x_0$$

complete noise

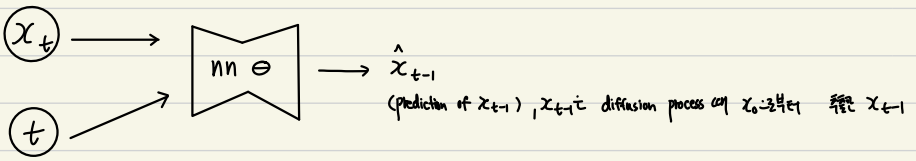
$N(x_T; 0, I)$

- Denoising for T steps = need T neural nets? ✗, T=1000 is too many...
- Use the property that in Diffusion model, dim of x_0 (observed variable) = dim of $x_1, x_2 \cdots x_T$ (latent variable)

- 위의 성질로 인해 여러 step에도 input, output의 차원의 같고 동일한 신경망 구조를 재활용 할 수 있음.

- 동그란 신경망 구조를 거쳐야지 시점 "t" 정보를 제공해줘야함. (어느 시점에서의 노이즈 제거인지 신경망이 알수 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

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

✳ Training Diffusion Models : ELBO

- Diffusion model also has latent variable ($z : x_1, x_2 \cdots x_T$), marginalization for $P_\theta(x)$ induces $\int$ which leads to log-Sum form for log-likelihood for single data x, Making it hard to optimize ⇒ Use ELBO for maximizing log-likelihood
- Maximizing $ELBO(x; \theta, \phi)$ leads to maximizing $\log P_\theta(x)$ (Jensen's Inequality) ··· Ⓐ
- Find ELBO's approximation in 3 steps (①, ②, ③)

① Sample $\{x_1, x_2 \cdots x_T\}$ 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

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

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

3 main changes

① observed variable $x \longrightarrow x_0$

② latent variable $z \longrightarrow \{x_1, x_2 \cdots x_T\}$

③ No need for ϕ since $\{x_1, x_2 \cdots x_T\}$ are sampled from "fixed" Gaussian Distribution, defined by noise scheduling hyperparameter. $\beta = \{\beta_1, \beta_2 \cdots \beta_T\}$

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

Change of denotion $x_{0:T} = x_0, x_1 \cdots x_T$

$$ELBO(x_0;\theta) = E_{q(x_{1:T}|x_0)}\left[\log \frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)}\right]$$ (ELBO of Diffusion model) : proves why x_0's ELBO is a lower evidence bound of x_0's log likelihood.

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

$$\log p_\theta(x_0) = \log \int p_\theta(x_0, z)\, dz \quad \text{(marginalization of } z\text{)}$$

$$= \log \int p_\theta(x_0, x_{1:T})\, dx_{1:T} \quad (\text{In diffusion, } z = \{x_1, x_2 \cdots x_T\} = x_{1:T})$$

$$= \log \int p_\theta(x_{0:T})\, dx_{1:T}$$

$$= \log \int p_\theta(x_{0:T}) \frac{q(x_{1:T}|x_0)}{q(x_{1:T}|x_0)} dx_{1:T} \quad \left(\frac{q(x_{1:T}|x_0)}{q(x_{1:T}|x_0)} \to 1\right)$$

$$= \log E_{q(x_{1:T}|x_0)}\left[\frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)}\right]$$

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

↗ Details will be dealt in last page

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

$$ELBO(x_0;\theta) = E_{q(x_{1:T}|x_0)}\left[\log \frac{P_\theta(x_{0:T})}{q(x_{1:T}|x_0)}\right]$$ (① 분자, ② 분모)

"Markov property & chain rule 을 이용해서 ①, ② 표현 바꾸기"

- **Reverse-time joint distribution in reverse diffusion** $P_\theta(x_{0:T})$ ··· ①
  - process of recovering x_0 starting from complete noise $x_T \sim N(x_T; 0, I)$: need to model probability $x_T \to x_0$ direction
  - Through markov (Ⓑ) chain rule (Ⓐ) $P_\theta(x_{0:T})$ is shown as.. (no θ since $x_T \sim N(x_T; 0, I)$)

$$P_\theta(x_{0:T}) = P(x_T)P_\theta(x_{T-1}|x_T)P_\theta(x_{T-2}|x_{T-1},x_T)\cdots P_\theta(x_0|x_{1:T}) \quad Ⓐ$$

$$= P(x_T)P_\theta(x_{T-1}|x_T)P_\theta(x_{T-2}|x_{T-1})\cdots P_\theta(x_0|x_1) \quad Ⓑ$$

$$= P(x_T)\prod_{t=1}^{T} P_\theta(x_{t-1}|x_t)$$

✳ Markov property & chain rule

Ⓐ Joint probability를 conditional probability의 곱으로 나타낼 수. (chain rule 이용한 시간에 순차적 전개의 x)

For two variables A, B

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

For n variables $X_1, X_2 \cdots X_n$

$P(X_{1:n}) = P(X_1) \times P(X_2|X_1) \times P(X_3|X_2,X_1) \cdots P(X_n|X_{n-1} \cdots X_2, X_1)$

↓ Markov 성질 만족 markov chain rule로 시간에 순차적 관계하는 구조로 표현

Ⓑ 현 시점 state, 전 시점 영향 받음

$P(x_{t-1}|\underbrace{x_t, x_{t+1}\cdots x_T}_{\text{reverse process에서 현 시점 이후 영향 받음}}) = \underbrace{P(x_{t-1}|x_t)}_{\text{Markov 성질 적용됨, 바로 직전 t-1 시점 영향 받음}}$

- **Forward process probability** $q(x_{1:T}|x_0)$ ··· ②

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

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

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

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

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

$$ELBO(x_0;\theta) = E_{q(x_{1:T}|x_0)}\left[\log \frac{P_\theta(x_{0:T})}{q(x_{1:T}|x_0)}\right]$$

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

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

↗ not containing θ

Since only ① is related to θ (reminder: our purpose is finding $\theta = \underset{\theta}{\operatorname{argmax}}\, ELBO(x_0;\theta)$)

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

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

⇒ 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 $J(\theta)$

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

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

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

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

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

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

↳ does not contain θ

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

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

✡ Summary of ELBO approximation with "T sampling"

① Sample $x_{1:T}$ from x_0 (forward diffusion process)

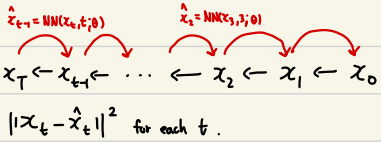
② Apply "same" neural network T times for x_t prediction $\hat{x}_t$ (denoising) from x_{t+1}

③ Calculate $||x_t - \hat{x}_t||^2$ for each time step and sum $\Rightarrow J(\theta)$

Concul) Neural Net이 각 시점의 denoising 예측값과 forward diffusion process의 실제 각 시점의 값이 가까워지게 학습!

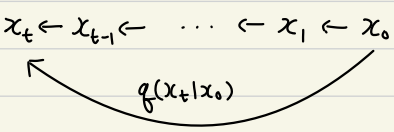
issue) Requires T times diffusing and T times denoising? ⇒ Cost... what if T=1000?!
need a way to reduce sampling and denoising times!

✡ figure



* Step 2 : 2 times Sampling

- $q(x_t|x_0)$를 해석적으로 구할 수 있다는 성질 이용 $\Leftrightarrow$ x_0에 노이즈를 1번만 추가해서 임의 시점 t에 대해 x_t를 샘플링 할 수 있다.



$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\mathbf{I}) \quad (\text{where } \alpha_t = 1-\beta_t,\ \bar{\alpha}_t = \alpha_t\alpha_{t-1}\cdots\alpha_1)$$

* page 2에서 $P(x_T) = \mathcal{N}(x_T; 0, \mathbf{I})$인 이유?

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

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

: $\bar{\alpha}_T$가 0에 가까워지도록 noise scheduler과 T를 결정한다 $\Rightarrow$ $x_T \sim \mathcal{N}(x_T; 0, \mathbf{I})$를 따르게 됨.

- ELBO expansion in 2 times sampling

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

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

↗ does not contain θ

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

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

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

$$= \sum_{t=1}^{T} E_{q(x_{1:T}|x_0)}\left[\log P_\theta(x_{t-1}|x_t)\right] \quad \cdots \text{linearity of expectation } (A)$$

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

Ⓐ Linearity of Expectation

$$E_{p(x,y)}[ax+by] = \iint (ax+by)\,p(x,y)\,dx\,dy$$

$$= \iint ax\,p(x,y)\,dx\,dy + \iint by\,p(x,y)\,dx\,dy$$

$$= a\iint x\,p(x,y)\,dx\,dy + b\iint y\,p(x,y)\,dx\,dy$$

$$= a\,E_{p(x,y)}[x] + b\,E_{p(x,y)}[y]$$ (합의 기대값 = 기대값의 합)

$$= a\,E_{p(x)}[x] + b\,E_{p(y)}[y]$$ — Ⓑ

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

Ⓑ Expectation of related variables

$f(x)$의 $p(x,y)$에 대한 expectation = $f(x)$의 $p(x)$에 대한 expectation

$$E_{p(x,y)}[f(x)] = E_{p(x)}[f(x)]$$: 기댓값의 내용 $f(x)$에 대해 $f(x)$와 무관한 확률분포의 확률 변수는 지울 수 있다!

pf) $E_{p(x,y)}[f(x)] = \iint f(x)\,p(x,y)\,dx\,dy$

$= \iint f(x)\,p(x)\,p(y|x)\,dx\,dy$

$= \int f(x)\,p(x)\underbrace{\int p(y|x)\,dy}_{1}\,dx$

$= \int f(x)\,p(x)\,dx$

$= E_{p(x)}[f(x)]$

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

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

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) $J(\theta)$?

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

① $x_{t-1} \sim q(x_{t-1} | x_0)$

$= \mathcal{N}(x_{t-1}; \sqrt{\bar{\alpha}_{t-1}} x_0, (1-\bar{\alpha}_{t-1}) I)$ ··· x_0로부터 x_{t-1} sampling하고,

② $x_t \sim q(x_t | x_{t-1})$

$= \mathcal{N}(x_t; \sqrt{1-\beta_t} x_{t-1}, \beta_t I)$ ··· x_{t-1}로부터 x_t sampling

But we still have Ⓐ...

If T is large, calculation cost ↑ ⇒ need a way to approximate $\sum_t$ (remember, this is calc of ELBO just for single data x_0. for n datas, sample x_{t-1}, x_t for each data and sum. Repeat this process n times)

How?

— T개의 합을 균등분포에 대한 기댓값으로 표현할 수 있다

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

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

from ①,

$f(t)$를 정의하는 식에서 $\sum_{t=1}^{T} f(t)$는 (t는 1부터 T까지 이산적인 값을 가짐) $t \sim \cup\{1, T\}$로부터의 $f(t)$의 대한 expectation으로 표현된다.

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

정리하면, 균등분포의 기댓값 성질로부터 $\sum_{t=1}^{T}$를 표현할 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,

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

©

©는 2개의 기댓값이 중첩되어 있음. "monte carlo"로 (샘플 크기 1 가정) u(t)에서 하나, ① $q(x_{t-1}, x_t | x_0)$에서 (x_{t-1}, x_t) ② 하나 Sampling 해서 근사값 계산.

<=>

$$t \sim U\{1, T\} \quad ①$$
$$x_{t-1} \sim q(x_{t-1} | x_0)$$
$$x_t \sim q(x_t | x_{t-1}) \quad ②$$
$$J(\theta) \approx T \log P_\theta(x_{t-1} | x_t)$$

Ⓐ

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

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

Ⓐ : $J(\theta) \approx T \log p_\theta(x_{t-1} | x_t) = T \log N(x_{t-1}; \hat{x}_{t-1}, I)$

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

$$= T \left\{ -\frac{1}{2}(x_{t-1}-\hat{x}_{t-1})^T (x_{t-1}-\hat{x}_{t-1}) + \log \frac{1}{\sqrt{(2\pi)^D}} \right\}$$

constant

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

$\left(\underset{\theta}{\operatorname{argmax}} J(\theta) \approx \underset{\theta}{\operatorname{argmin}} \| x_{t-1} - \hat{x}_{t-1} \|^2 \right)$ → 1 method 에서는 매 스텝마다의 제곱오차 최소화인데, 여기서는 1개의 step만!

☆ Summary for 2 sampling method

$J(\theta)$ can be calculated by

$t \sim U\{1, T\}$ (time sampling from uniform distribution)

Ⓐ $x_{t-1} \sim q(x_{t-1} | x_0)$ (x_{t-1} sampling from $q(x_{t-1}|x_0)$)] → forward diffusing process

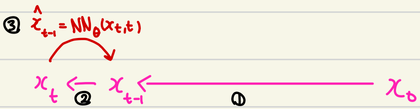
$x_t \sim q(x_t | x_{t-1})$ (x_t sampling from $q(x_t | x_{t-1})$)

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

get $\| x_{t-1} - \hat{x}_{t-1} \|^2$

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

☆ Figure



④ : $\| x_{t-1} - \hat{x}_{t-1} \|^2$

* $q(x_t|x_0)$ (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 $z = x+y$

$$x \sim N(u_x, \delta_x^2)$$
$$y \sim N(u_y, \delta_y^2)$$
$$z = x+y$$

↗ This is the reason of downsampling in forward diffusion process?

then, $z \sim N(u_x+u_y, \delta_x^2+\delta_y^2)$

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

↓ reparametrization trick

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

$$x_t = \sqrt{a_t}\,x_{t-1} + \sqrt{1-a_t}\,\epsilon_t \quad (A)$$

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

$$x_{t-1} = \sqrt{a_{t-1}}\,x_{t-2} + \sqrt{1-a_{t-1}}\,\epsilon_{t-1} \quad (B)$$

from (A), (B) ...

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

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

Thus, Ⓑ follows gaussian distribution.

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

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

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

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

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

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

$= N(0, (1 - a_t a_{t-1})I)$

Thus,

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

Ⓑ $\sim N(0, (1 - a_t a_{t-1})I)$

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

↓ 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 $x_{t-2}, x_{t-3} \cdots x_1, x_0$

↓

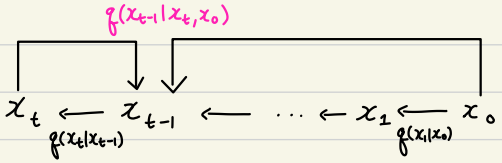
$x_t = \sqrt{a_t a_{t-1} \cdots a_1}\,x_0 + \sqrt{1 - a_t a_{t-1} \cdots a_1}\,\epsilon$

$= \sqrt{\bar{a}_t}\,x_0 + \sqrt{1 - \bar{a}_t}\,\epsilon \quad (\bar{a}_t = a_t a_{t-1} \cdots a_1) \quad (\epsilon \sim N(0, I))$

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

* Step 3 : 1 Sampling for ELBO approx.

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



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

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

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

$$\delta_q^2(t) = \frac{(1-a_t)(1-\bar{a}_{t-1})}{1-\bar{a}_t}$$ (covariance matrix)

$q(x_{t-1}|x_t, x_0)$ 에서 mean vector는 x_t과 x_0의 선형합으로 표현 (내분점의 개념) (x_0은 관측값 x_t는 forward diffusion process 통해 Sampling 해야함)

$\delta_q^2(t)$는 하이퍼파라미터로 정해지는 scalar값.

Concul) x_0, x_t만 알고있다면 Hyperparameter로 정한 $\beta = \{\beta_1, \beta_2 \cdots \beta_T\}$를 통해 Gaussian Distribution $q(x_{t-1}|x_t, x_0)$의 mean vector과 Covariance matrix를 알 수 있다!

• Transformation of $J(\theta)$ for 1 sampling method

From ELBO we get $J(\theta)$

$$J(\theta) = \sum_{t=1}^{T} \underbrace{E_{q(x_{t-1},x_t|x_0)}[\log P_\theta(x_{t-1}|x_t)]}_{J_0}$$

$$= T\, E_{u(t)}\left[\underbrace{E_{q(x_{t-1},x_t|x_0)}[\log P_\theta(x_{t-1}|x_t)]}_{J_0}\right]$$

$\Leftrightarrow$ $J(\theta) = T E_{u(t)}[J_0]$: J_0에 θ를 포함하지 않는 항을 추가해도 $J(\theta)$의 θ에 대한 최적화 문제는 변하지 않음.

$q(x_{t-1}|x_t,x_0)$을 이용해 계산에 아주 관련한 θ를 포함하지 않는 항을 J_0에 추가한 것임!

$$\arg\max_\theta J_0 = \arg\max_\theta \left(J_0 - \underbrace{E_{q(x_{t-1},x_t|x_0)}[\log q(x_{t-1}|x_t,x_0)]}_{\text{Ⓐ}}\right)$$

(add constant not containing θ)

$$= \arg\max_\theta \left(E_{q(x_{t-1},x_t|x_0)}[\log P_\theta(x_{t-1}|x_t)] - \underbrace{E_{q(x_{t-1},x_t|x_0)}[\log q(x_{t-1}|x_t,x_0)]}_{\text{Ⓐ}}\right)$$

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

$$= \arg\max_\theta \underbrace{E_{q(x_{t-1},x_t|x_0)}\left[\log \frac{P_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t,x_0)}\right]}_{J_1 \ (J_0\text{에 Ⓐ를 추가해 변형시킨 } J_1)}$$

$\arg\max_\theta J_0 = \arg\max_\theta J_1$ since the pink term Ⓐ does not contain θ.

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

$\Rightarrow$ 이제서 목적함수에 one sampling ELBO approximation을 수행하기 위한 $q(x_{t-1}|x_t,x_0)$를 포함시켰음!

- Expension of J_1 & Maximize J_1

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

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

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

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

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

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

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

$\therefore J_1$ 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

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

Maximize J_1?

$\Longleftrightarrow \quad D_{KL} = 0$

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

$N(x_{t-1}; u_q(x_t,x_0), \delta_q^2(t)I)$ $\qquad$ $N(x_{t-1}; \hat{x}_{t-1}, I)$

↓ $q(x_{t-1}|x_t,x_0)$의 $\delta_q^2(t)$와 대칭되게 위해 아래와 같이 바꿈

$N(x_{t-1}; \hat{x}_{t-1}, \delta_q^2(t)I)$

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

- Change of denotion : $\text{NeuralNet}(x_t,t;\theta) \longrightarrow u_\theta(x_t,t)$

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

- Review : D_{KL} between 2 gaussian distribution : 두 가우시안의 KL은 해석적으로 표현가능.

let $q(z) = N(z; u_1, \delta_1^2 I)$, $p(z) = N(z; u_2, \delta_2^2 I)$, where $z \in D^H$

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

$N(x_{t-1}; u_q(x_t, x_0), \delta_q^2(t) I)$ $\qquad$ $N(x_{t-1}; u_\theta(x_t, t), \delta_q^2(t) I)$

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

$$= -\frac{1}{2}\sum_{h=1}^{H}\left(1 + \log\frac{\delta_q^2(t)}{\delta_q^2(t)} - \frac{(u_{q,h}(x_t,x_0) - u_{\theta,h}(x_t,t))^2}{\delta_q^2(t)} - \frac{\delta_q^2(t)}{\delta_q^2(t)}\right) \qquad \delta_q^2(t) = \frac{(1-\alpha_t)(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t} \Rightarrow \text{"scalar"}$$

$$\Rightarrow \delta_{q,h}^2(t) \to \delta_q^2(t)$$

$$= \frac{1}{2}\sum_{h=1}^{H}\frac{(u_{q,h}(x_t,x_0) - u_{\theta,h}(x_t,t))^2}{\delta_q^2(t)}$$

$$= \frac{1}{2\delta_q^2(t)}\sum_{h=1}^{H}(u_{q,h}(x_t,x_0) - u_{\theta,h}(x_t,t))^2$$

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

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

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

$J(\theta) \to$ Loss ($-\frac{H}{2}\cdot 171$) $\oplus$ remove $\frac{1}{2}, T$

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

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

Only sample x_t (1 time!) from forward diffusion process...

$$t \sim u(t)$$

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

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

Train neural network (update θ) to reduce $\text{Loss}(x_0;\theta)$ $\Rightarrow$ make prediction of neural network at timestep t $\hat{x}_{t-1} = u_\theta(x_t,t)$ closer to $u_q(x_t,x_0)$.

↳ x_t, x_0 사이의 내분점

✡ Summary of 1 sampling method

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

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

for a given observed variable x_0,

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

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

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

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

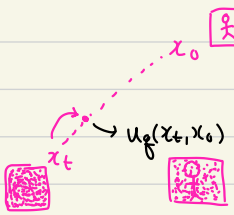
⇔ make $\hat{x}_{t-1} = \text{NeuralNet}(x_t,t;\theta) = u_\theta(x_t,t)$ close to $u_q(x_t,x_0)$

↳ dividing point

Intuition?

$\hat{x}_{t-1} = \text{NeuralNet}(x_t,t)$ (x_t로부터 한 스텝 denoising한 값이 x_t, x_{0-1} 사이점(간 비율 값)과 가까워지도록 Neural Network를 업데이트)

$x_t \leftarrow x_{t-1} < \cdots \cdots x_0$



⇒ dividing point should be close to t step denoising output...

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

② 임의 시점 t 정함

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

④ x_t, t를 neural net을 통해 예측한(denoising) $\hat{x}_{t-1}$이 x_t와 x_0의 dividing point과 가까워지도록 neural net parameter θ update

✳ Traing 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. $x_t \sim q_f(x_t|x_0)$: $x_t = \sqrt{\bar{a}_t}x_0 + \sqrt{1-\bar{a}_t}\,\epsilon$
5. $u_f(x_t, x_0) = \frac{\sqrt{a_t}(1-\bar{a}_{t-1})x_t + \sqrt{\bar{a}_{t-1}}(1-a_t)x_0}{1-\bar{a}_t}$
6. $\sigma_f^2(t) = \frac{(1-a_t)(1-\bar{a}_{t-1})}{1-\bar{a}_t}$

] mean vector & covariance for $q_f(x_{t-1}|x_t, x_0)$

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

↓

x_t 사용하여 x_0 방향으로 약간 모이는가 퍼지는 내분점.

✳ Different things that neural network predicts

- **Neural Network that predicts x_0 (원본 x_0 맞힌 신경망)**

신경망을 통해 가깝게 예측해야 하는 값은 $\mu_q(x_t, x_0)$이다.

$$\mu_q(x_t, x_0) = \frac{\sqrt{a_t}(1-\bar{a}_{t-1})x_t + \sqrt{\bar{a}_{t-1}}(1-a_t)x_0}{1-\bar{a}_t} \quad \text{A} \quad \cdots ①$$

↗ focus here!

$\mu_\theta(x_t, t)$ 식은 $\mu_q(x_t, x_0)$ 식 format에 맞게 조정

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

↗ 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.

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

$$= \frac{1}{2\delta_q^2(t)} \left|\left| \mu_\theta(x_t, t) - \mu_q(x_t, x_0) \right|\right|^2$$

$$= \frac{1}{2\delta_q^2(t)} \left(\frac{\sqrt{\bar{a}_{t-1}}(1-a_t)}{1-\bar{a}_t} \right)^2 \left|\left| \hat{x}_\theta(x_t, t) - x_0 \right|\right|^2$$

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

• Neural Network that predicts noise

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

↓ reparameterization trick

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

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

$$\Leftrightarrow x_0 = \frac{x_t - \sqrt{1-\bar{\alpha}_t}\epsilon}{\sqrt{\bar{\alpha}_t}} \cdots (A)$$

(A)를 앞페이지의 ①에 대입 후 전개

$$\mu_q(x_t, x_0) = \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})x_t + \sqrt{\bar{\alpha}_{t-1}}(1-\alpha_t)x_0}{1-\bar{\alpha}_t}$$

$$= \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon\right) \cdots (B) \quad (\epsilon \rightarrow i)$$

Make $\mu_\theta(x_t, t)$ format corresponding to (B)

$$\mu_\theta(x_t, t) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(x_t, t)\right) \cdots (C) \quad (\epsilon_\theta \rightarrow ii)$$

↳ the term that Neural Network directly predicts!

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

$$\therefore D_{KL}(q(x_{t-1}|x_t, x_0) \| p_\theta(x_{t-1}|x_t)) = \frac{1}{2\delta_q^2(t)}\|\mu_\theta(x_t, t) - \mu_q(x_t, x_0)\|^2$$

$$= \frac{1}{2\delta_q^2(t)}\frac{(1-\alpha_t)^2}{(1-\bar{\alpha}_t)\alpha_t}\|\epsilon_\theta(x_t, t) - \epsilon\|^2$$

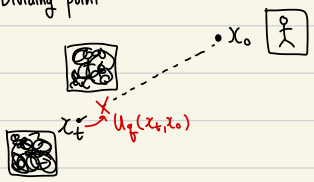
↳ make neural network predict gaussian noise ϵ that has been added from $x_0 \rightarrow x_t$, $x_t \sim q(x_t|x_0)$ for reparameterization trick.

✳ Summary of what model predicts

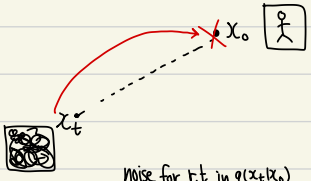
Iterate ...

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

① Dividing point



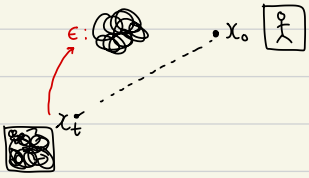
② x_0 restore



X : prediction point (what the neural network predicts)

noise for x_t in $q(x_t|x_0)$

③ Gaussian noise ϵ prediction



"Method ③ is the best method in experiment"

4) Make Loss function based on these three.
5) Get $\frac{\partial}{\partial \theta} Loss(x_0; \theta)$ and update θ with gradient descent.

* Sampling (Generating) from Diffusion Models.

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

↓ R.T

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

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

↳ make $\epsilon = 0$ when $t=1$ (x_1에서 x_0 예측할 때는 R.T에 사용하는 노이즈 ϵ 사용하지 X ⇒ 이래야 결과가 흐릿하다고 함)

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

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

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 $\epsilon = 0$

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

$$x_{t-1} = \underbrace{\frac{1}{\sqrt{a_t}}\left(x_t - \frac{1-a_t}{\sqrt{1-\bar{a}_t}}\epsilon_\theta(x_t,t)\right)}_{\text{mean vector for } P_\theta(x_{t-1}|x_t)} + \underbrace{\delta_f(t)}_{\text{variance for } P_\theta(x_{t-1}|x_t)}\epsilon$$

Neural Network's output

return x_0

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

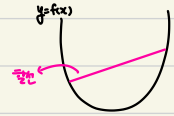
"Next Time"

* Jensen's Inequality

- Use J.I to induce ELBO

• Convex function & Jensen's Inequality

- Convex function?



=> 모든 할선이 항상 위쪽에 f(x)는 convex function

- Mathematical Expression

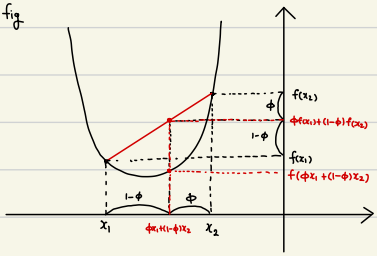
임의의 두 점 x_1, x_2와 $\phi = [0, 1]$ 일 때

$\phi f(x_1) + (1-\phi) f(x_2) \geq f(\phi x_1 + (1-\phi) x_2)$ 이면 $f(x)$는 convex function ····· ©

Ⓐ : $f(x_1)$과 $f(x_2)$의 $1-\phi : \phi$ 내분점

Ⓑ : x_1과 x_2의 $1-\phi : \phi$ 내분점의 $f(x)$값

fig



⊕ Convex function인지를 확인하는 또다른 방법 ⇒ $f''(x) \geq 0$인지 확인하기

ex) $f(x) = x^2 \rightarrow f'(x) = 2x \longrightarrow f''(x) = 2$ (convex function)

Ⓒ는 내분점 1개에 관한 식인데 이를 다수로 확장하면, Jensen's Inequality

$\phi = \{\phi_1, \phi_2 \cdots, \phi_N\}$, $\sum_{n=1}^{N} \phi_n = 1$ 이고 임의의 점 $x = \{x_1, x_2 \cdots x_n\}$에 대해 $f(x)$가 convex function이면 다음이 성립

$$\sum_{n=1}^{N} \phi_n f(x_n) \geq f\left(\sum_{n=1}^{N} \phi_n x_n\right)$$

- Concave function
  - 모든 할선이 $f(x)$ 이하인 경우
  - If $f(x)$ is convex function, then $-f(x)$ is concave function

- Is $f(x) = \log x$ a concave function?

$f(x) = \log x$

$-f(x) = -\log x$

$-f'(x) = -\frac{1}{x}$

$-f''(x) = \frac{1}{x^2}$

Since $-f''(x) \geq 0$, $-f(x)$ is convex function and $f(x)$ is concave function

Since $-f(x) = -\log x$ is convex function,

for $\phi = \{\phi_1, \phi_2 \cdots \phi_N\}$, $\sum_{n=1}^{N} \phi_n = 1$, $x = \{x_1, x_2 \cdots x_N\}$

$$\sum_{n=1}^{N} -\phi_n \log x_n \geq -\log \sum_{n=1}^{N} \phi_n x_n \quad \left(\sum_{n=1}^{N} \phi_n f(x_n) \geq f\left(\sum_{n=1}^{N} \phi_n x_n\right) \text{ for convex func } f(x)\right)$$

$\Leftrightarrow \sum_{n=1}^{N} \phi_n \log x_n \leq \log \sum_{n=1}^{N} \phi_n x_n$: Jensen's Inequality for concave func $\log x$

(보통 J.I는 convex function에 대한 부등식이나 여기서는 기존 J.I에 −를 곱해서 concave func인 log에 대한 J.I를 도출함)

- ELBO from Jensen's Inequality

For probability distribution $q(z)$ and $\sum_z q(z) = 1$, and for arbitrary function $f(z)$,

The $\log x$'s Jensen's Inequality is as follows;

$\sum_z q(z) \log f(z) \leq \log \sum_z q(z) f(z)$ (since $\log x$ is concave function, $\sum_n \phi_n \log x_n \leq \log \sum_n \phi_n x_n$)

For log-likelihood for x in P.D with latent variable z,

$$\log p(x;\theta) = \log \sum_z p(x,z;\theta) \quad \text{(marginalization)}$$

$$= \log \sum_z p(x,z;\theta) \frac{q(z)}{q(z)} \rightarrow \text{multiply 1}$$

$$= \log \sum_z q(z) \frac{p(x,z;\theta)}{q(z)} \rightarrow f(z)$$

$$\geq \sum_z q(z) \log f(z)$$

$$\geq \sum_z q(z) \log \frac{p(x,z;\theta)}{q(z)} \quad \text{(Jensen's Inequality for concave func } \log x \rightarrow \text{ELBO)}$$

↳ ELBO