Diffusion Model for Text Generation

Presenter: Jeongwan Shin

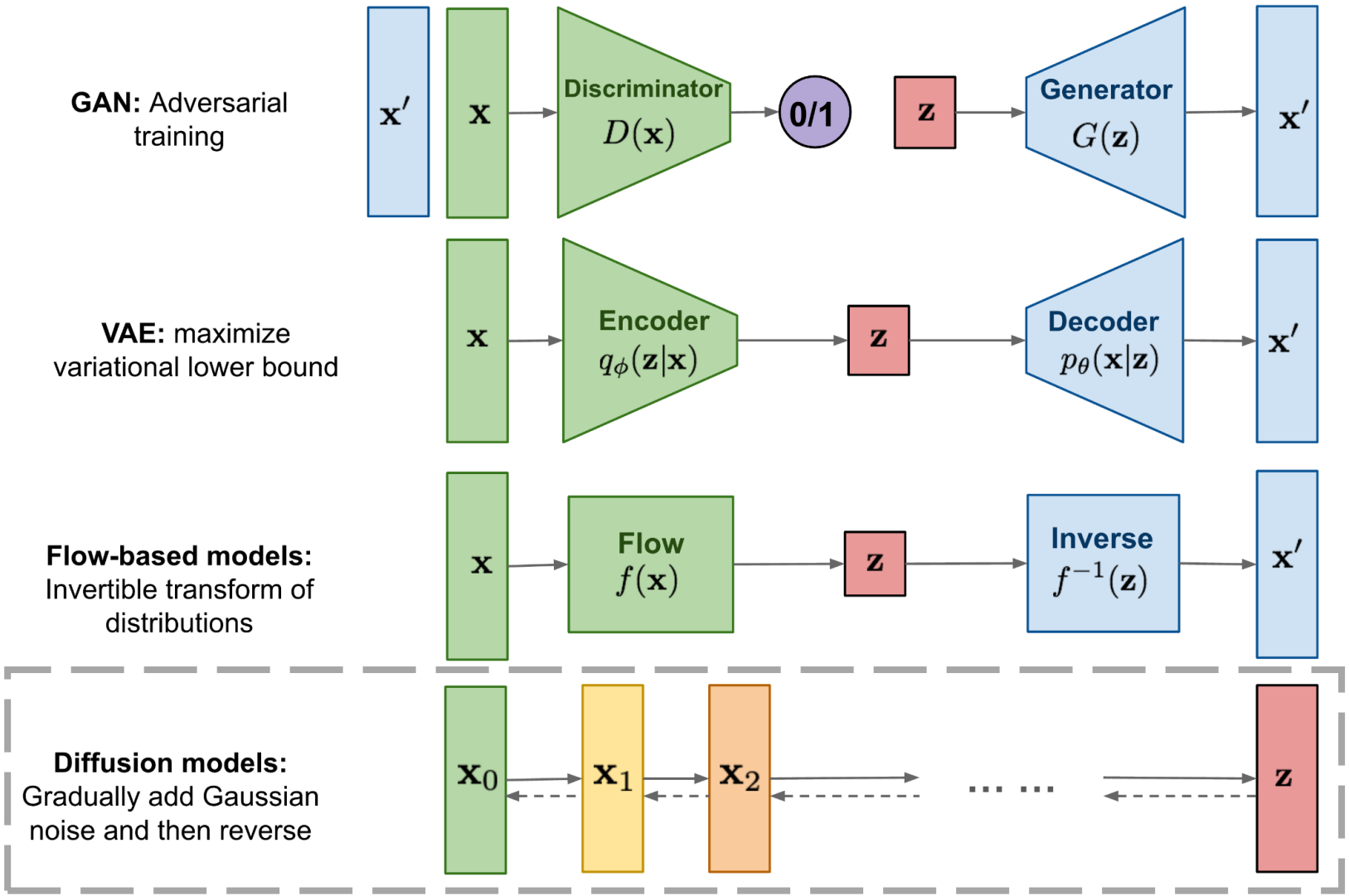
Kyungpook National University

February 13, 2022

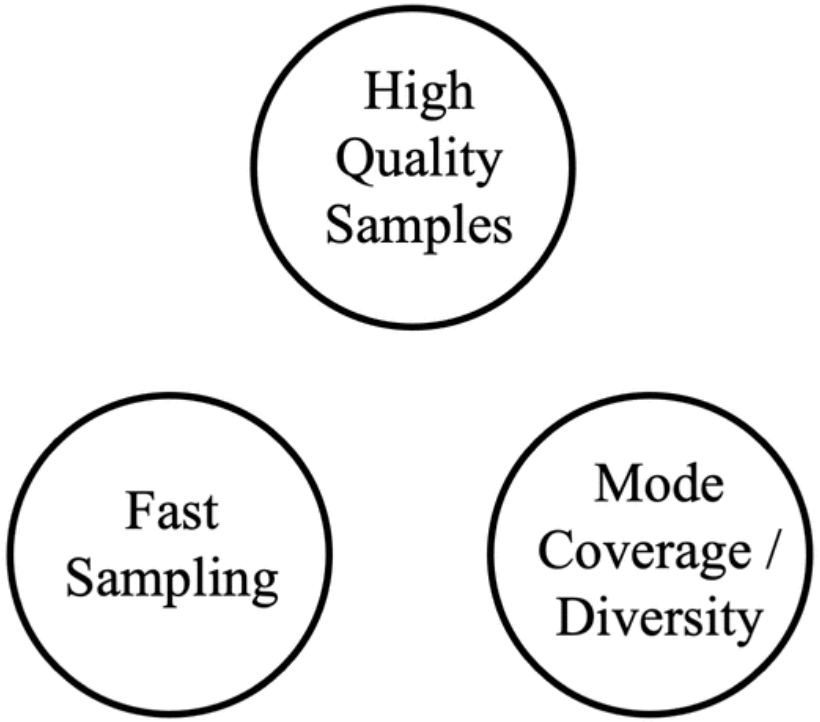
Contents

- **Generative Model**
- **Diffusion Model**
- **Text Diffusion Model**
- **Work in progress**

Generative Models

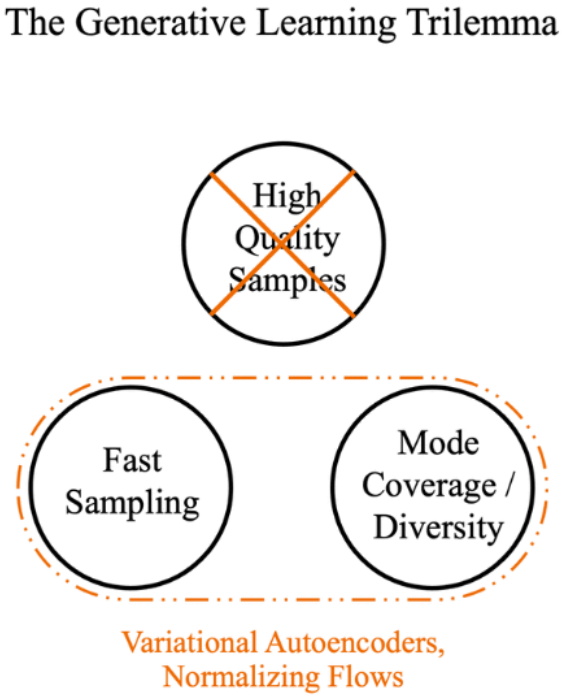


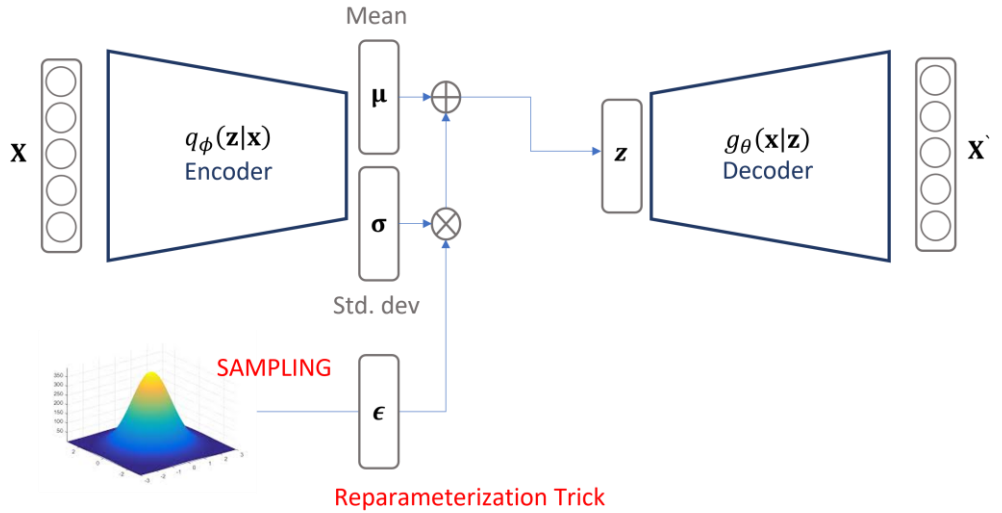
The Generative Learning Trilemma



- 응용 프로그램에 적용하기 위한 생성 모델의 주요 요구 사항

Variational Autoencoder



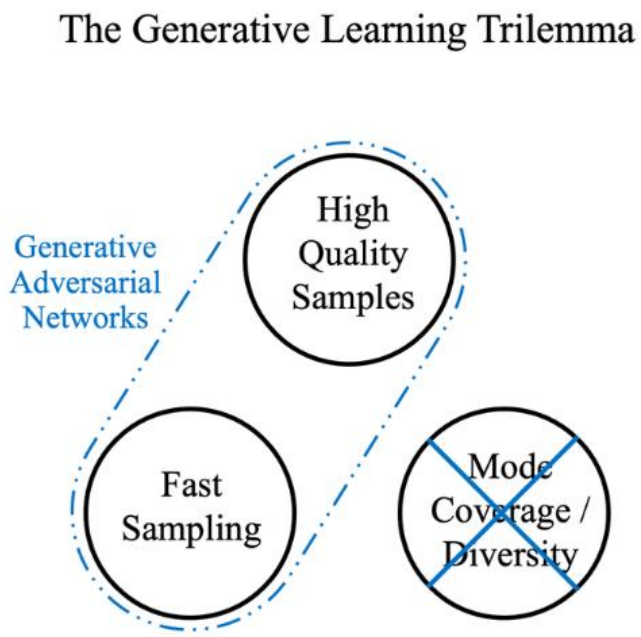


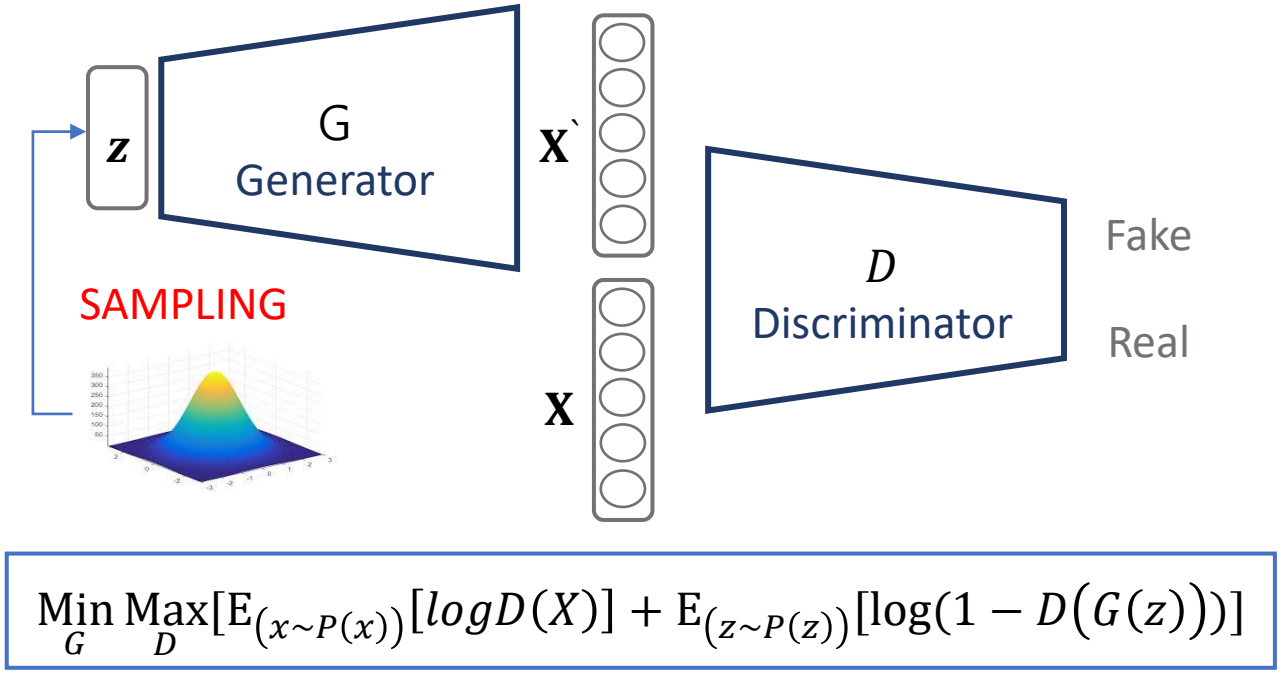
Regularization on Encoder Reconstruction on Decoder

$$L_{VAE} = D_{KL}(q(z|x)||p_\theta(z)) - E_{z\sim q(z|x)}\log P_\theta(x|z)$$

- Reparameterization trick을 통한 샘플링으로 Diversity가 높음
- Regularization과 Reconstruction 의 trade-off

Generative Adversarial Network

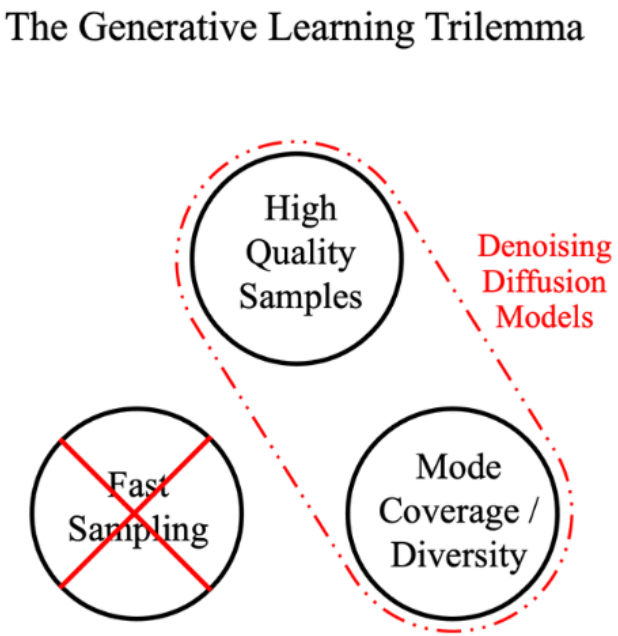


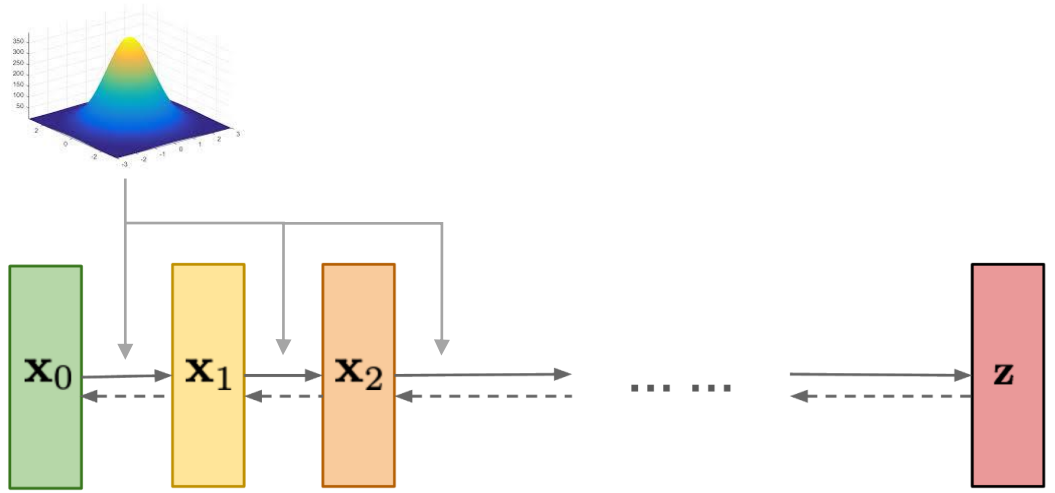


$$\min_G \max_D [\mathrm{E}_{(x\sim P(x))}[logD(X)] + \mathrm{E}_{(z\sim P(z))}[\log(1 - D(G(z)))]$$

- Mode collapse : Generator 는 Discriminator 를 속이기 쉬운 특정 소수 데이터를 생성
- Non-convergence : MinMax 학습의 불안정성(Instability)

Diffusion Model





$$\mathbb{E}_{q(\mathbf{x}_0)}[D_{KL}(q(\mathbf{x}_T|\mathbf{x}_0)|p(\mathbf{x}_T)) + \sum_{t>1} D_{KL}(q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)|p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)) - logp_\theta(\mathbf{x}_0|\mathbf{x}_1)\,]$$

- High quality : Data semantic 을 보존한 latent representation 생성 및 학습, 반복적인 denoising 단계
- 수천 단계의 반복적인 Step으로 인한 느린 생성 속도
- 계층적 latent representation에 의한 diversity 가 높음

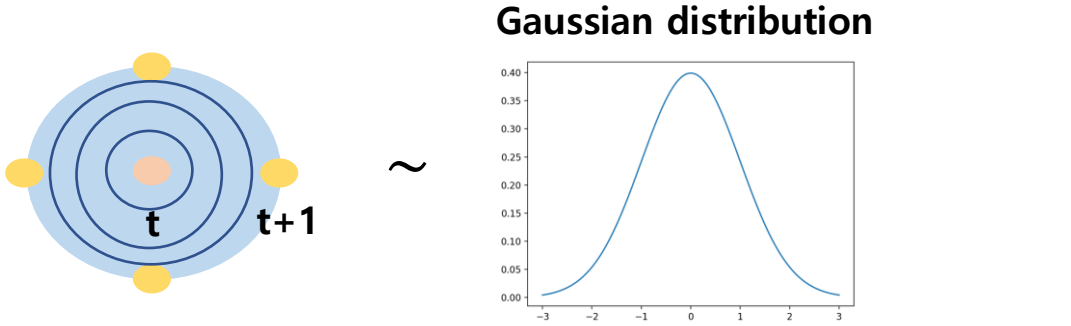
Diffusion Model Papers

- Deep Unsupervised Learning using Nonequilibrium Thermodynamics - ICML2015
- Denoising Diffusion Probabilistic Models(DDPM) - NeurIPS 2020

Diffusion Process

물리학에서 분자의 운동을 Stochastic 문제로 정의

Diffusion Process



- 분자가 움직이는 건 일정하지 않다. 어떤 **Stochastic process**를 따를 것이다.
- 가정 : 아주 짧은 시간 t에서 $t+1$로 움직이는 분자는 가우시안 분포를 따를 것이다.

Diffusion

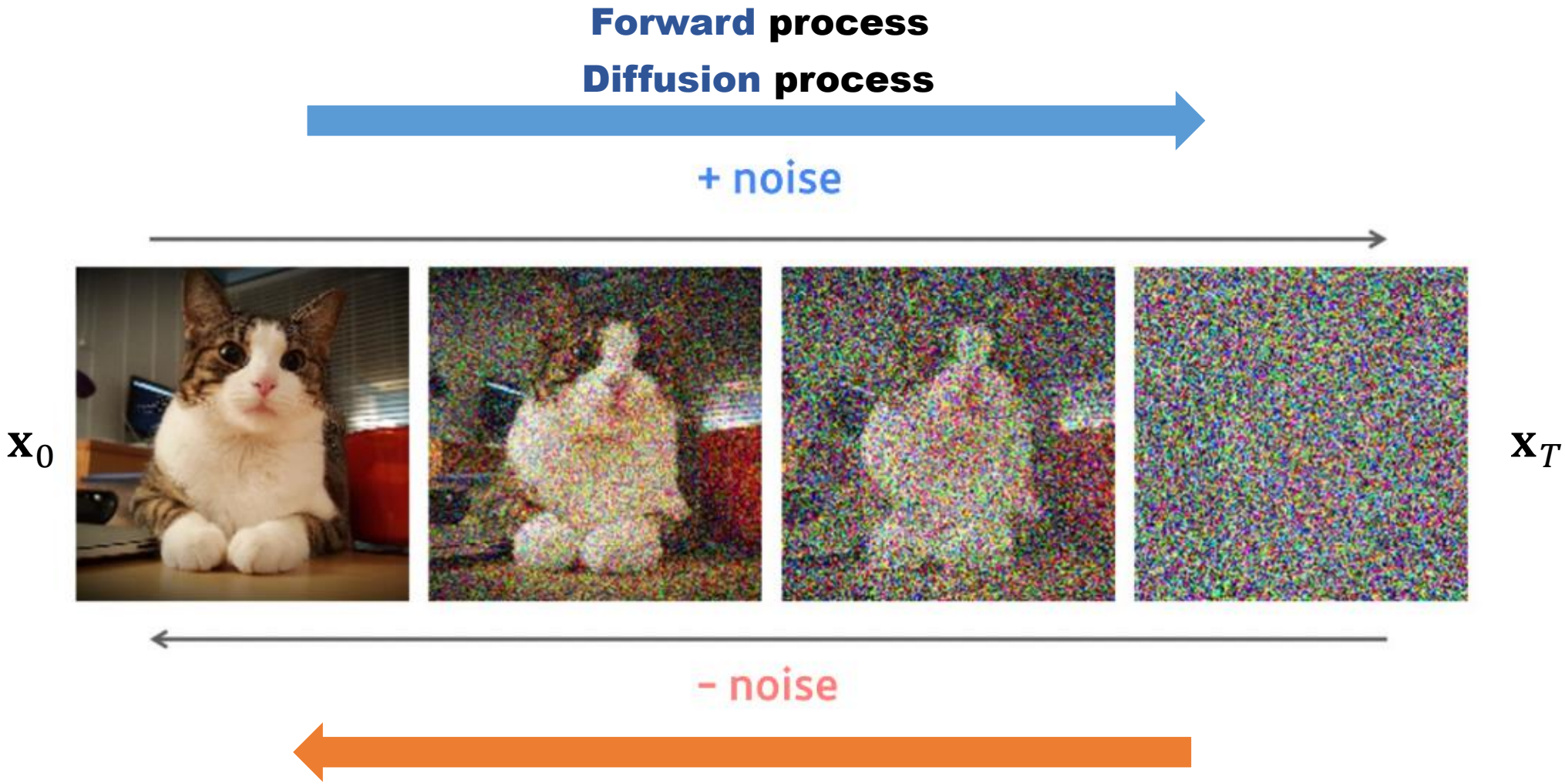
Forward process

Diffusion process

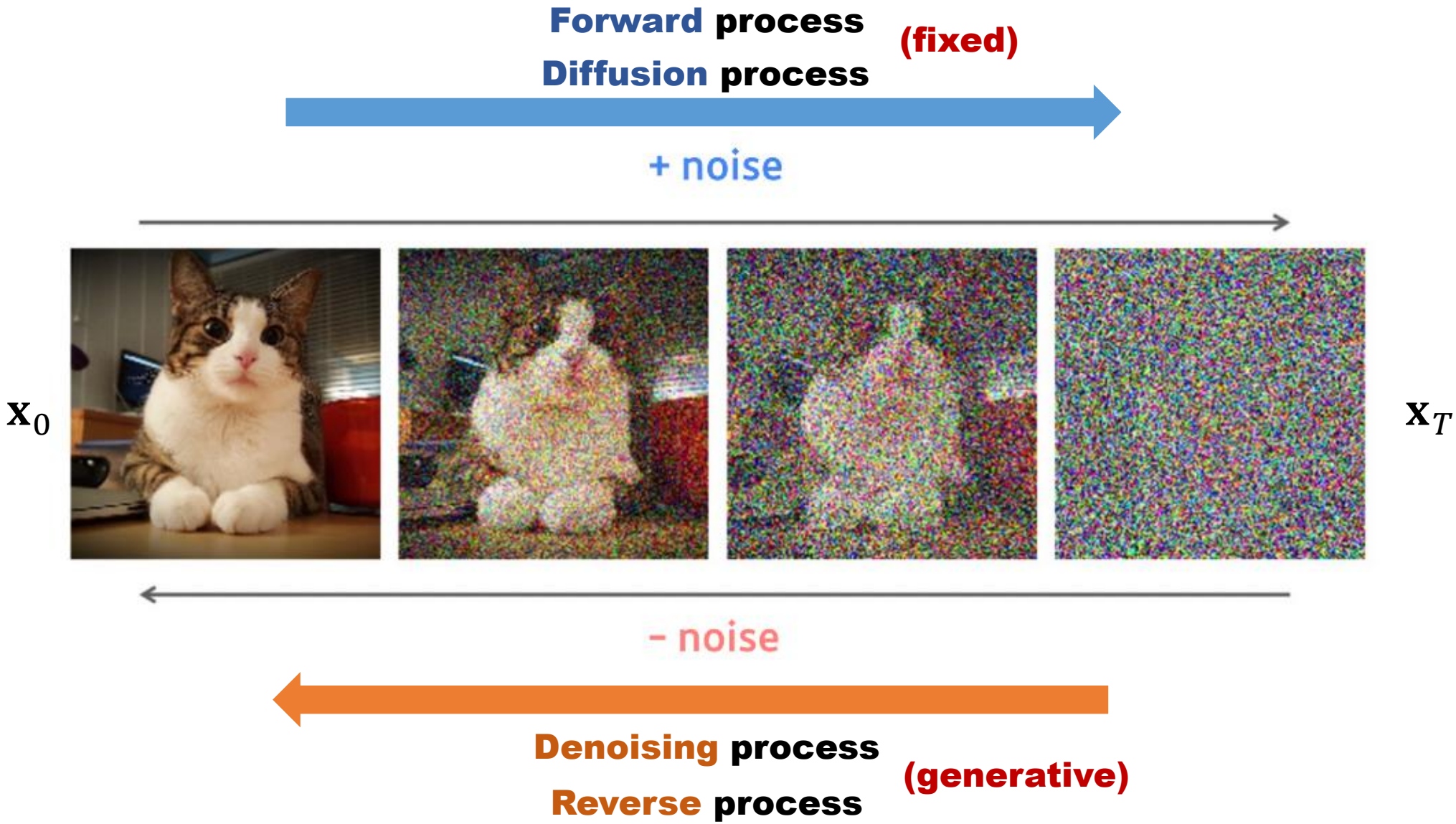
Denoising process

Reverse process

Diffusion


Forward process
Diffusion process
+ noise
$\mathbf{x}_0$
$\mathbf{x}_T$
– noise
Denoising process
Reverse process

Diffusion

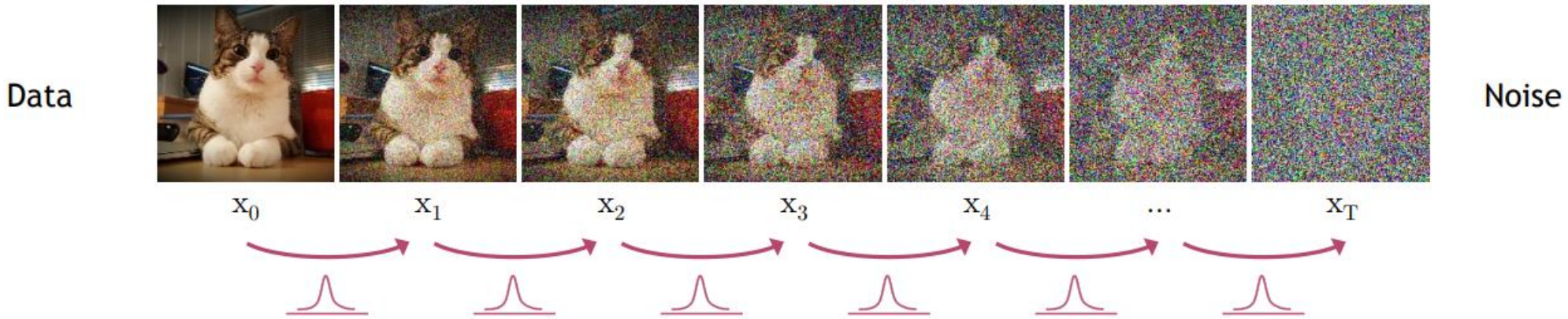

Forward process
Diffusion process
(fixed)
+ noise
$\mathbf{x}_0$
$\mathbf{x}_T$
– noise
Denoising process
Reverse process
(generative)

Forward Diffusion Process

- Markov chain으로 data에 점진적으로 noise를 추가하는 과정

$N(0, \mathrm{I})$

Pure Gaussian distribution



mean variance

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = N(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I})$$

Conditional gaussian distribution

$0 < \beta_1 < \beta_2 < \ ... < \beta_T < 1$
0.0001 ~ 0.02

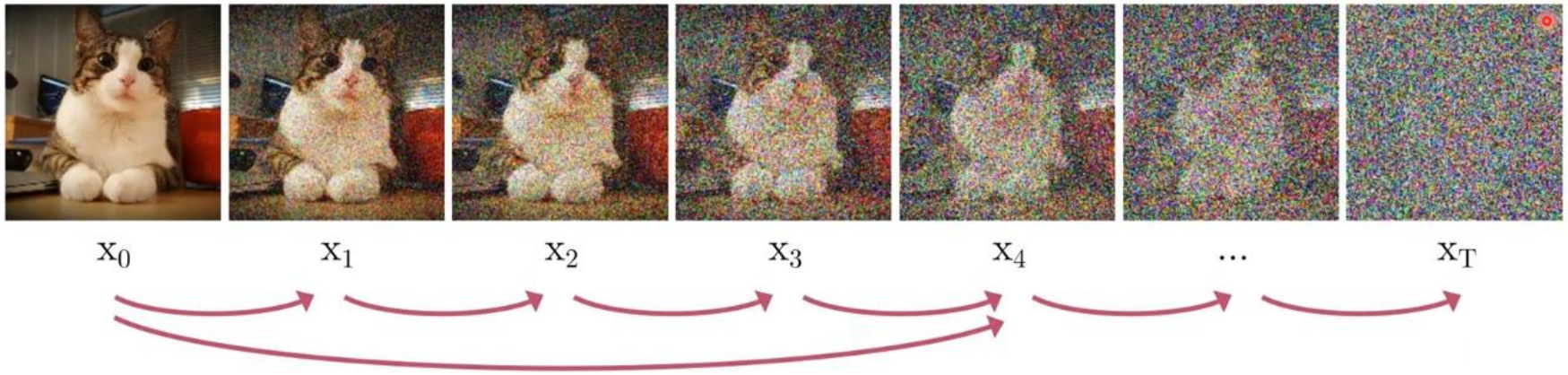
variance schedule

Closed-form ➡ $\mathbf{x}_t = \sqrt{1-\beta_t}\mathbf{x}_{t-1} + \sqrt{\beta_t}\epsilon$ where $\epsilon \sim N(0, \mathbf{I})$

Reparameterization Trick(VAE)

Forward Diffusion Process

- 한 step씩 forward를 실행하면 많은 Memory와 Resource를 요구



$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = N(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathrm{I})$$

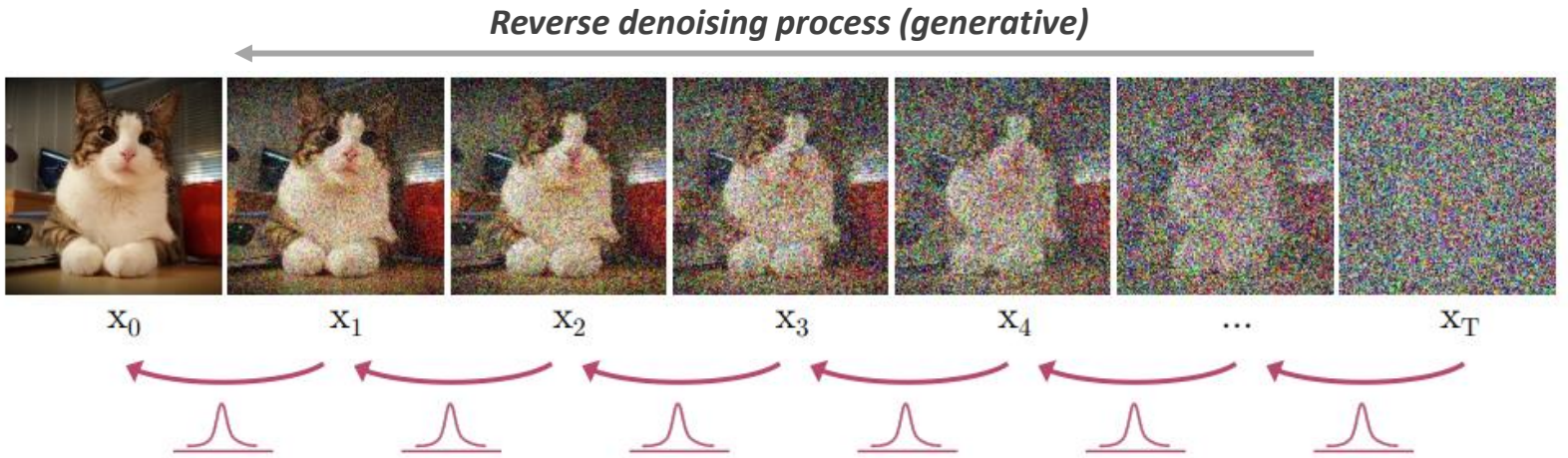
Conditional gaussian distribution

Define $\bar{a}_t = \prod_{s=1}^{t}(1-\beta_s)$ ➡ $q(\mathbf{x}_t|\mathbf{x}_0) = N(\mathbf{x}_t; \sqrt{\bar{a}_t}\mathbf{x}_0, (1-\bar{a}_t)\mathbf{I})$

For sampling: $\mathbf{x}_t = \sqrt{\bar{a}_t}\mathbf{x}_0 + \sqrt{(1-\bar{a}_t)}\,\epsilon$ where $\epsilon \sim N(0, \mathbf{I})$ ← Reparameterization Trick(VAE) [참고자료2]

Reverse Denoising Process

- Gaussian noise $\mathbf{x_T}$ 에서 T step 만큼 Denoising 하면서 이미지 $\mathbf{x_0}$를 만드는 과정



$$p(\mathbf{x}_T) = N(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$$

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = N(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\mathbf{I})$$

Trainable network
(U-net, Denoising Autoencoder)

Objective for Reverse process

$$Maximize\ p_\theta(\mathbf{x}_0)$$

- VAE 학습에서 사용된 variational upper bound를 활용 :

$$\mathbb{E}_{q(\mathbf{x}_0)}[-logP_\theta(\mathbf{x}_0)] \leq \mathbb{E}_{q(\mathbf{x}_0)q(\mathbf{x}_{1:T}|\mathbf{x}_0)}[-log\frac{p_\theta(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}]$$

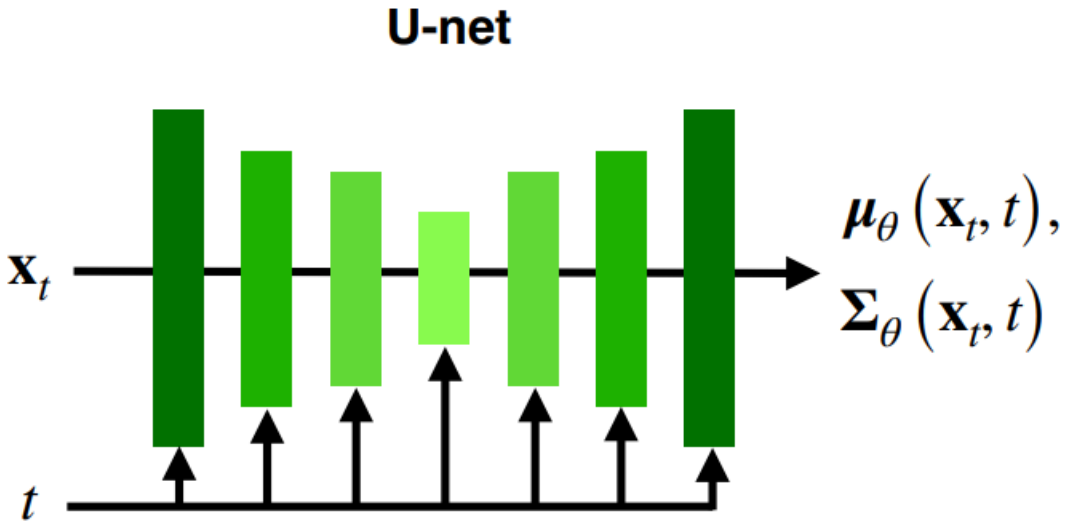
- Sohl-Dickstein et al. ICML2015 (참고자료3)

$$\mathbb{E}_{q(\mathbf{x}_0)}[\underbrace{D_{KL}(q(\mathbf{x}_T|\mathbf{x}_0)|p(\mathbf{x}_T))}_{L_t} + \sum_{t>1}\underbrace{D_{KL}\left(q\left(\mathbf{x}_{t-1}\middle|\mathbf{x}_t,\mathbf{x}_0\right)\middle|p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)\right)}_{L_{t-1}} - \underbrace{logp_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0}\ ]$$

- Ho et al. NeurIPS 2020 (참고자료5)

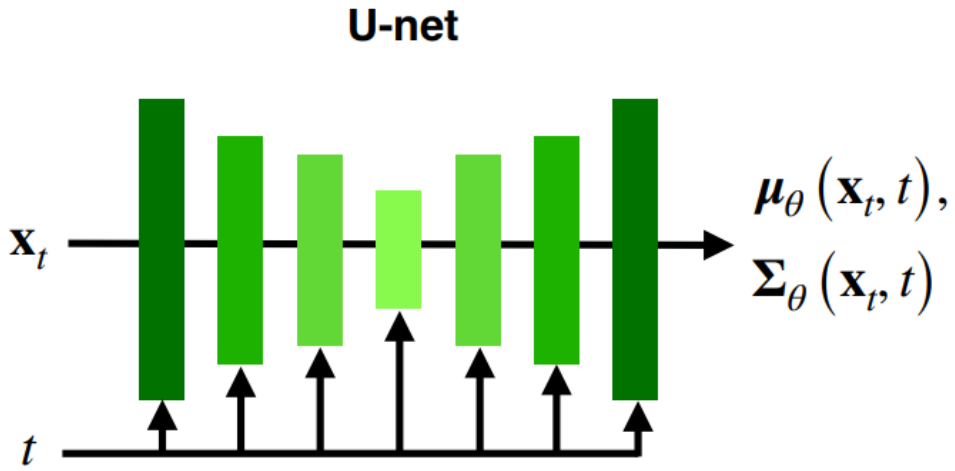
$$L_{t-1} = D_{KL}\left(q\left(\mathbf{x}_{t-1}\middle|\mathbf{x}_t,\mathbf{x}_0\right)\middle|p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)\right) = \mathbb{E}_{q(\mathbf{x}_0)}\left[\frac{1}{2\sigma_t^2}\ ||\tilde{\mu}_t(\mathbf{x}_t,\mathbf{x}_0) - \mu_\theta(\mathbf{x}_t,t)||^2\right] + C$$

Deep Unsupervised Learning using Nonequilibrium Thermodynamics - ICML2015



$$\mathbb{E}_{q(\mathbf{x}_0)}[\underbrace{D_{KL}(q(\mathbf{x}_T|\mathbf{x}_0)|p(\mathbf{x}_T))}_{L_t} + \sum_{t>1} \underbrace{D_{KL}(q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)|p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}} - \underbrace{logp_\theta(\mathbf{x}_0|\mathbf{x}_1)}_{L_0}]$$

Deep Unsupervised Learning using Nonequilibrium Thermodynamics - ICML2015



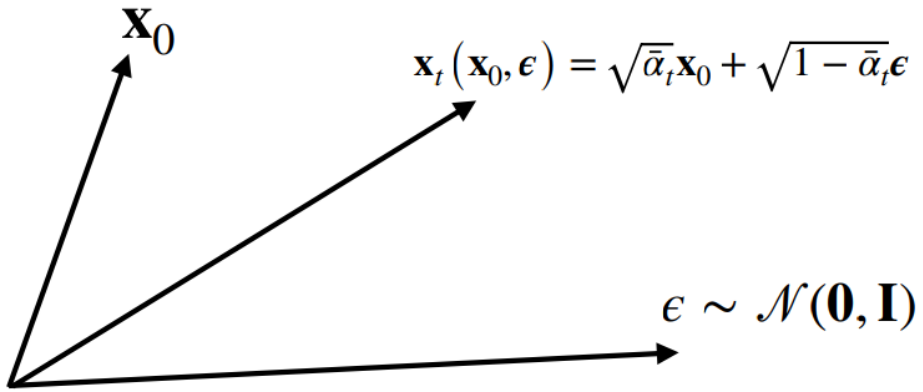
$$L_{t-1} = D_{KL}\left(q\left(\mathbf{x}_{t-1}\middle|\mathbf{x}_t,\mathbf{x}_0\right)\middle|p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)\right) \longrightarrow KL(p,q) = \log\frac{\sigma_2}{\sigma_1} + \frac{\sigma_1^2 + (\mu_1-\mu_2)^2}{2\sigma_2^2} - \frac{1}{2}$$

- *Posterior* : $q\left(\mathbf{x}_{t-1}\middle|\mathbf{x}_t,\mathbf{x}_0\right) = N\left(\mathbf{x}_{t-1}; \tilde{\mu}_t(\mathbf{x}_t,\mathbf{x}_0), \tilde{\beta}_t\mathbf{I}\right)$ [참고자료.4]

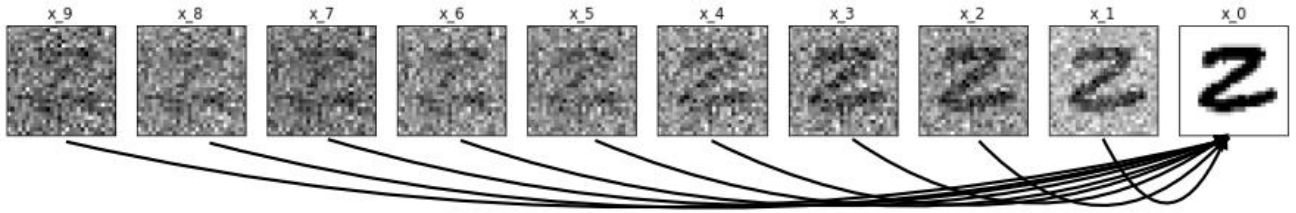
$$where \quad \tilde{\mu}_t(\mathbf{x}_t, \mathbf{x}_0) := \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_t)}{1-\bar{\alpha}_t}\mathbf{x}_t \ and \ \tilde{\beta}_t := \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$$

- *Reverse process* : $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) := N(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, \mathrm{t}), \Sigma_\theta(\mathbf{x}_t, t))$

Denoising Diffusion Probabilistic Models Ho et al. NeurIPS 2020



$\boldsymbol{x}_t$ 는 $\boldsymbol{x}_0$와 ϵ 의 *linear combination*



Predict ϵ (or $\mathbf{x}_t$) at each step

1. $\bar{\alpha}_t$ 만큼 accumulation된 Noise를 denoising하도록 학습 할 수 있음

2. $\mathbf{x}_0$를 바로 예상함으로써 $\mathbf{x}_t$ 시점에서 $\mathbf{x}_0$ 의 스타트 포인트를 예상하여 다음 $\mathbf{x}_{t-1}$ 의 방향을 잘 잡을 수 있다. 즉, $q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)$에서 x_0 가 주어질 때 x_{t-1} 를 더 잘 예측하는 것과 유사 함.

Objective for DDPM

$$KL(p,q) = \log \frac{\sigma_2}{\sigma_1} + \frac{\sigma_1^2 + (\mu_1 - \mu_2)^2}{2\sigma_2^2} - \frac{1}{2}$$

- $q(\mathrm{x}_{t-1}|\mathrm{x}_t, \mathrm{x}_0), p_\theta(\mathrm{x}_{t-1}|\mathrm{x}_t)$ 두 식은 Normal distributions 이기 때문에
- KL divergence는 간단한 형태로 전개 가능

$$L_{t-1} = D_{KL}\left(q(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{x}_0)|p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)\right) = \mathbb{E}_{q(\mathbf{x}_0)}\left[\frac{1}{2\sigma_t^2}\ ||\tilde{\mu}_t(\mathbf{x}_t,\mathbf{x}_0) - \mu_\theta(\mathbf{x}_t,t)||^2\right] + C$$

참고자료.5

$$\tilde{\mu}_t(\mathbf{x}_t,\ \mathbf{x}_0) = \frac{1}{\sqrt{1-\beta_t}}\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon$$

$$\mu_\theta(\mathbf{x}_t,\ t) = \frac{1}{\sqrt{1-\beta_t}}\left(\mathbf{x}_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathbf{x}_t,\ t)\right)$$

$$L_{simple}(\theta) := \mathbb{E}_{t,\mathbf{x}_0,\epsilon}\left[\left|\left|\epsilon - \epsilon_\theta\left(\underbrace{\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon}_{x_t}, t\right)\right|\right|^2\right]$$

Objective for DDPM

$$L_{simple}(\theta) := \mathbb{E}_{t,\mathbf{x}_0,\epsilon}\left[\left\|\epsilon - \epsilon_\theta\left(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t\right)\right\|^2\right]$$

```python
def forward_diffusion_sample(x_0, t, device="cpu"):
    """
    iamge랑 timestep을 input으로 받아와서 noisy된 image를 return
    """
    noise = torch.randn_like(x_0) # x_0와 같은 크기를 갖는 gaussian distribution

    sqrt_alphas_cumprod_t = get_index_from_list(sqrt_alphas_cumprod, t, x_0.shape)
    sqrt_one_minus_alphas_cumprod_t = get_index_from_list(
        sqrt_one_minus_alphas_cumprod, t, x_0.shape
    )
    # mean + variance
    return sqrt_alphas_cumprod_t.to(device) * x_0.to(device) #
    + sqrt_one_minus_alphas_cumprod_t.to(device) * noise.to(device), noise.to(device)
```

For sampling: $\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0 + \sqrt{(1-\bar{\alpha}_t)}\,\epsilon$ where $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

```python
def get_loss(model, x_0, t):
    x_noisy, noise = forward_diffusion_sample(x_0, t, device)
    noise_pred = model(x_noisy, t)
    return F.l1_loss(noise, noise_pred)
```

Objective for Reverse process

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$
6: **until** converged

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

풀이) 참고자료.5

참고자료 1

- Reparameterization trick을 사용하여 closed form으로 임의의 시간 t에서 $\mathbf{x}_t$ 를 샘플링할 수 있다.
- $\alpha_t = 1 - \beta_t, \bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$:

$$\mathbf{x}_t = \sqrt{\alpha_t}\mathbf{x}_{t-1} + \sqrt{1-\alpha_t}\boldsymbol{\epsilon}_{t-1} \quad ; \textit{where } \boldsymbol{\epsilon}_{t-1}, \boldsymbol{\epsilon}_{t-2}, \ldots \sim N(0, I)$$

$$= \sqrt{\alpha_t \alpha_{t-1}}\mathbf{x}_{t-2} + \sqrt{1-\alpha_t\alpha_{t-1}}\bar{\boldsymbol{\epsilon}}_{t-2} \quad ; \textit{where } \bar{\boldsymbol{\epsilon}}_{t-2} \textit{ merges two Gaussians(*)}$$

$$= \ \ldots$$

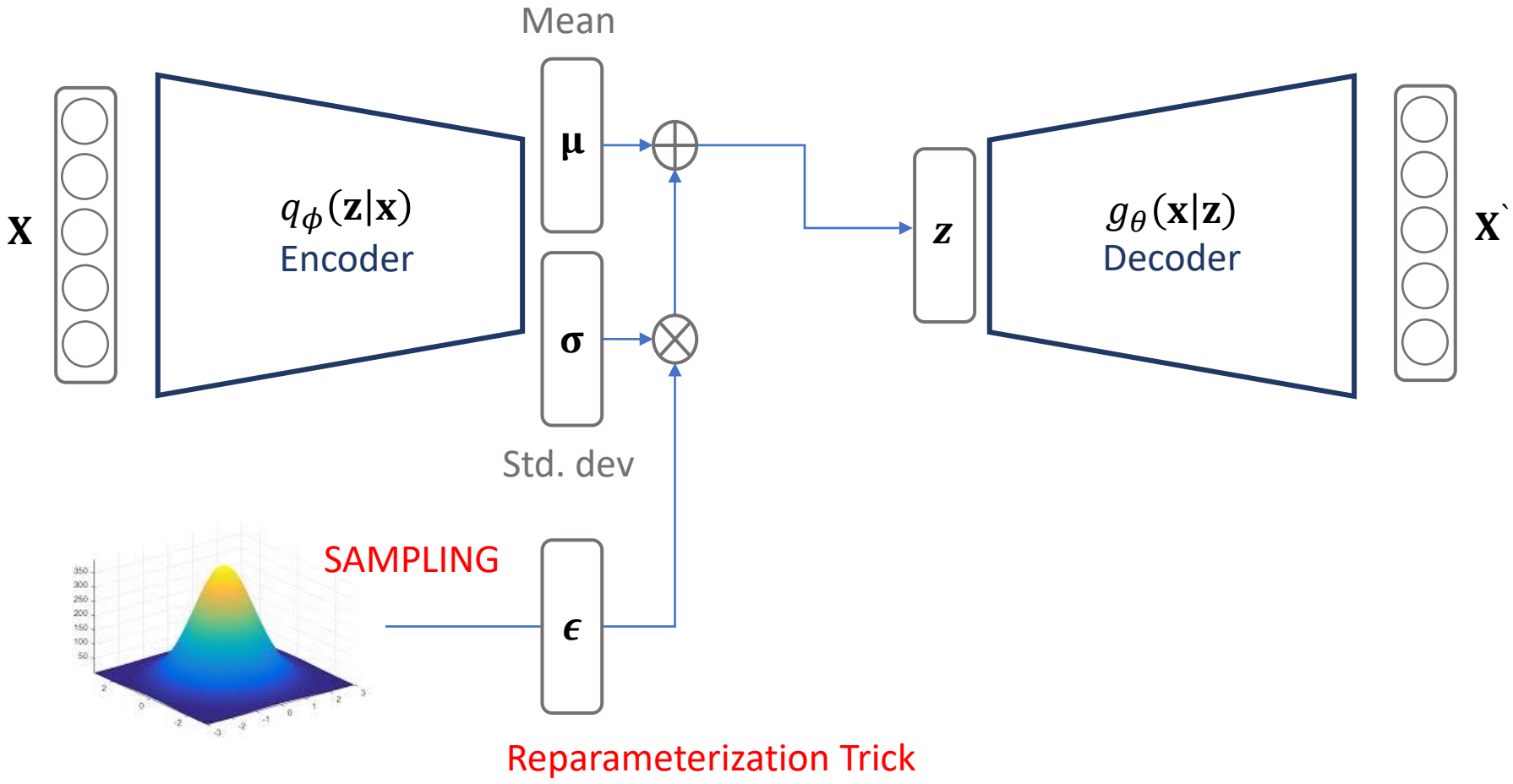
$$= \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}$$

$$q(\mathbf{x}_t|\mathbf{x}_0) = N(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I})$$

- 다른 분산(variance)을 가지는 두개의 가우시안 $(N(\mathbf{0}, \sigma_1^2\mathbf{I}), N(\mathbf{0}, \sigma_2^2\mathbf{I}))$을 병합 $\rightarrow N(\mathbf{0}, (\sigma_1^2 + \sigma_2^2)\mathbf{I})$
- 병합된 표준편차(merged standard deviation) :
  - $\sqrt{(1-\alpha_t) + \alpha_t(1-\alpha_{t-1})} = \sqrt{1-\alpha_t\alpha_{t-1}}$

참고자료 2

- Variational Autoencoder



$$\boldsymbol{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon} \text{ where } \epsilon \sim N(0, \mathbf{I})$$

참고자료.3

$$\boldsymbol{maximize\ p_\theta(x) \rightarrow \mathbb{E}_{q(x_0)}[-log p_\theta(x_0)]}$$

$$\mathbb{E}_{x_T\sim q(x_T|x_0)}[-\log p_\theta(x_0)]$$

① $= \mathbb{E}_{x_T\sim q(x_T|x_0)}\left[-\log \frac{p_\theta(x_0,x_1,x_2,\dots,x_T)}{p_\theta(x_1,x_2,x_3,\dots,x_T|x_0)}\right]$ ∵ bayes rule, $p_\theta(x_T|x_0)=\frac{p_\theta(x_T,x_0)}{p_\theta(x_0)}$

② $= \mathbb{E}_{x_T\sim q(x_T|x_0)}\left[-\log \frac{p_\theta(x_0,x_1,x_2,\dots,x_T)}{p_\theta(x_1,x_2,x_3,\dots,x_T|x_0)}\cdot\frac{q(x_{1:T}|x_0)}{q(x_{1:T}|x_0)}\right]$

③ $\leq \mathbb{E}_{x_T\sim q(x_T|x_0)}\left[-\log \frac{p_\theta(x_0,x_1,x_2,\dots,x_T)}{q(x_{1:T}|x_0)}\right]$ ∵ *KL divergence* > 0, "**ELBO**"

④ $= \mathbb{E}_{x_T\sim q(x_T|x_0)}\left[-\log \frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)}\right]$ ∵ *Notation*

⑤ $= \mathbb{E}_{x_T\sim q(x_T|x_0)}\left[-\log \frac{p_\theta(\boldsymbol{x_T})\prod_{t=1}^{T}p_\theta(\boldsymbol{x_{t-1}}|\boldsymbol{x_t})}{\prod_{t=1}^{T}q(\boldsymbol{x_t}|\boldsymbol{x_{t-1}})}\right]$ ∵ *Below Markov chain property*

⑥ $= \mathbb{E}_{x_{1:T}\sim q(x_{1:T}|x_0)}\left[-\log p_\theta(\boldsymbol{x_T})-\sum_{t=1}^{T}\log\frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})}\right]$ ∵ *separating to summation in logarithm*

$$\mathbb{E}_{x_{1:T}\sim q(x_{1:T}|x_0)}[-\log p_\theta(x_0)]$$

⑦ $\leq \mathbb{E}_{x_{1:T}\sim q(x_{1:T}|x_0)}\left[-\log p_\theta(\boldsymbol{x_T})-\sum_{t=1}^{T}\log\frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})}\right]$

⑧ $= \mathbb{E}_{x_{1:T}\sim q(x_{1:T}|x_0)}\left[-\log p_\theta(\boldsymbol{x_T})-\sum_{t=2}^{T}\log\frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})}-\log\frac{p_\theta(x_0|x_1)}{q(x_1|x_0)}\right]$

⑨ $= \mathbb{E}_{x_{1:T}\sim q(x_{1:T}|x_0)}\left[-\log p_\theta(\boldsymbol{x_T})-\sum_{t=2}^{T}\log\frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t,x_0)}\cdot\frac{q(x_{t-1}|x_0)}{q(x_t|x_0)}-\log\frac{p_\theta(x_0|x_1)}{q(x_1|x_0)}\right]$ ∵*

⑩ $= \mathbb{E}_{x_{1:T}\sim q(x_{1:T}|x_0)}\left[-\log p_\theta(\boldsymbol{x_T})-\sum_{t=2}^{T}\log\frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t,x_0)}-\sum_{t=2}^{T}\log\frac{q(x_{t-1}|x_0)}{q(x_t|x_0)}-\log\frac{p_\theta(x_0|x_1)}{q(x_1|x_0)}\right]$

⑪ $= \mathbb{E}_{x_{1:T}\sim q(x_{1:T}|x_0)}\left[-\log p_\theta(\boldsymbol{x_T})-\sum_{t=2}^{T}\log\frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t,x_0)}-\log\frac{q(x_1|x_0)}{q(x_T|x_0)}-\log\frac{p_\theta(x_0|x_1)}{q(x_1|x_0)}\right]$

⑫ $= \mathbb{E}_{x_{1:T}\sim q(x_{1:T}|x_0)}\left[-\log\frac{p_\theta(\boldsymbol{x_T})}{q(x_T|x_0)}-\sum_{t=2}^{T}\log\frac{p_\theta(x_{t-1}|x_t)}{q(x_{t-1}|x_t,x_0)}-\log p_\theta(x_0|x_1)\right]$

$$p_\theta(\boldsymbol{x_{0:T}}) := p_\theta(\boldsymbol{x_T})\prod_{t=1}^{T}p_\theta(\boldsymbol{x_{t-1}}|\boldsymbol{x_t}) \qquad q(\boldsymbol{x_{1:T}}|\boldsymbol{x_0}) := \prod_{t=1}^{T}q(\boldsymbol{x_t}|\boldsymbol{x_{t-1}})$$

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

$= q(x_t|x_{t-1},x_0)$ ∵ Markov chain property

$= \frac{q(x_t,x_{t-1},x_0)}{q(x_{t-1},x_0)}$ ∵ bayes rule

$= \frac{q(x_{t-1},x_t,x_0)}{q(x_{t-1},x_0)}\cdot\frac{q(x_t,x_0)}{q(x_t,x_0)}$

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

참고자료.3

$$\boldsymbol{maximize}\ p_\theta(x) \rightarrow \mathbb{E}_{q(x_0)}[-\log p_\theta(x_0)]$$

$$= \mathbb{E}_{q(x_0)}\left[-\log \frac{p_\theta(x_0,x_1,x_2,\ldots,x_T)}{p_\theta(x_1,x_2,x_3,\ldots,x_T|x_0)}\right] \quad \because \textit{bayes rule}, p_\theta(x_t|x_0) = \frac{p_\theta(x_T,x_0)}{p_\theta(x_0)}$$

$$= \mathbb{E}_{q(x_0)}\left[-\log \frac{p_\theta(x_0,x_1,x_2,\ldots,x_T)}{p_\theta(x_1,x_2,x_3,\ldots,x_T|x_0)} \frac{q(x_{1:T}|x_0)}{q(x_{1:T}|x_0)}\right] \leq \mathbb{E}_{q(x_0)}\left[-\log \frac{p_\theta(x_0,x_1,x_2,\ldots,x_T)}{p_\theta(x_1,x_2,x_3,\ldots,x_T|x_0)}\right]$$

참고자료.4

Posterior $q\left(\mathbf{x}_{t-1} \mid \mathbf{x}_t, \mathbf{x}_0\right)=\mathcal{N}\left(\mathbf{x}_{t-1} ; \tilde{\boldsymbol{\mu}}_t\left(\mathbf{x}_t, \mathbf{x}_0\right), \tilde{\beta}_t \mathbf{I}\right)$

where $\tilde{\boldsymbol{\mu}}_t\left(\mathbf{x}_t, \mathbf{x}_0\right):=\frac{\sqrt{\bar{\alpha}_{t-1}} \beta_t}{1-\bar{\alpha}_t} \mathbf{x}_0+\frac{\sqrt{\alpha_t}\left(1-\bar{\alpha}_{t-1}\right)}{1-\bar{\alpha}_t} \mathbf{x}_t$ and $\tilde{\beta}_t:=\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t} \beta_t$

Note

$$\boldsymbol{q(x_{t-1} \mid x_t, x_0)=q(x_t \mid x_{t-1}) \frac{q(x_{t-1} \mid x_0)}{q(x_t \mid x_0)}}$$

$$q(x_t|x_{t-1})=\frac{1}{\sqrt{2\pi\beta_t}}\exp\left(-\frac{(x_t-\sqrt{1-\beta_t}x_{t-1})^2}{2\beta_t}\right)$$

$$q(x_t|x_0)=\frac{1}{\sqrt{2\pi(1-\bar{\alpha}_t)}}\exp\left(-\frac{(x_t-\sqrt{\bar{\alpha}_t}x_0)^2}{2(1-\bar{\alpha}_t)}\right)$$

$$q(x_{t-1}|x_0)=\frac{1}{\sqrt{2\pi(1-\bar{\alpha}_{t-1})}}\exp\left(-\frac{(x_{t-1}-\sqrt{\bar{\alpha}_{t-1}}x_0)^2}{2(1-\bar{\alpha}_{t-1})}\right)$$

$$\therefore q(x_{t-1}|x_t,x_0)=\frac{1}{\sqrt{2\pi\beta_t(\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t})}}\exp\left(-\frac{(x_t-\sqrt{1-\beta_t}x_{t-1})^2}{2\beta_t}-\frac{(x_{t-1}-\sqrt{\bar{\alpha}_{t-1}}x_0)^2}{2(1-\bar{\alpha}_{t-1})}+\frac{(x_t-\sqrt{\bar{\alpha}_t}x_0)^2}{2(1-\bar{\alpha}_t)}\right)$$

$$=\frac{1}{\sqrt{2\pi\beta_t(\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t})}}\exp\left(-([\frac{1}{2(1-\bar{\alpha}_{t-1})}+\frac{1-\beta_t}{2\beta_t}]x_{t-1}^2-[\frac{2\sqrt{1-\beta_t}}{2\beta_t}x_t+\frac{2\sqrt{\bar{\alpha}_{t-1}}}{2(1-\bar{\alpha}_{t-1})}x_0]x_{t-1}+C)\right)$$

$$=\frac{1}{\sqrt{2\pi\beta_t(\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t})}}\exp\left(-\frac{1}{2\beta_t(\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t})}[x_{t-1}^2-(\frac{2\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\alpha_t}x_t+\frac{2\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x_0)x_{t-1}+C]\right)$$

$$\approx\frac{1}{\sqrt{2\pi\beta_t(\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t})}}\exp\left(-\frac{1}{2\beta_t(\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t})}[x_{t-1}-(\frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}x_0+\frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_{t-1}}x_t)]^2\right)$$

b a

Note

$$\boldsymbol{P(B \mid A)=P(A \mid B) \frac{P(B)}{P(A)}}$$

$$\boldsymbol{N\left(x ;\ \mu, \sigma^2\right)=\frac{1}{\sigma \sqrt{2 \pi}} e^{-\frac{(x-\mu)^2}{2 \sigma^2}}}$$

- $q(x_t \mid x_{t-1})=N\left(x_t ; \sqrt{1-\beta_t} x_{t-1}, \beta_t * I\right)$
- $q(x_t \mid x_0)=N\left(x_t ; \sqrt{\bar{\alpha}_t} x_0,\left(1-\bar{\alpha}_t\right) * I\right)$
  - $\alpha_t=1-\beta_t$
  - $\bar{\alpha}_t=\prod_{i=1}^t \alpha_i$

- $N\left(X_{t-1} ; \frac{\sqrt{\bar{\alpha}_{t-1}} \beta_t}{1-\bar{\alpha}_t} x_0+\frac{\sqrt{\alpha_t}\left(1-\bar{\alpha}_{t-1}\right)}{1-\bar{\alpha}_{t-1}} x_t, \beta_t \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\right)$

참고자료.5

- Overview
  - $q(X_{t-1} \mid X_t) \rightarrow q(X_{t-1} \mid X_t, X_0)$
    - $N\left(X_{t-1};\ \tilde{\mu}(X_t, X_0), \tilde{\Sigma}(X_t, X_0)\right)$
    - $N\left(X_{t-1}; \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t} x_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_{t-1}} x_t, \beta_t \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\right)$
  - $x_t = \sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t}\varepsilon$ Forward process

Note

- $q(x_t \mid x_{t-1}) = N(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t * I)$
- $q(x_t \mid x_0) = N(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t) * I)$
  - $\alpha_t = 1 - \beta_t$
  - $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$

Note

$$\tilde{\mu}_t = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\frac{1}{\sqrt{\bar{\alpha}_t}}(x_t - \sqrt{1-\bar{\alpha}_t}\epsilon) + \frac{\sqrt{\bar{\alpha}_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}x_t$$

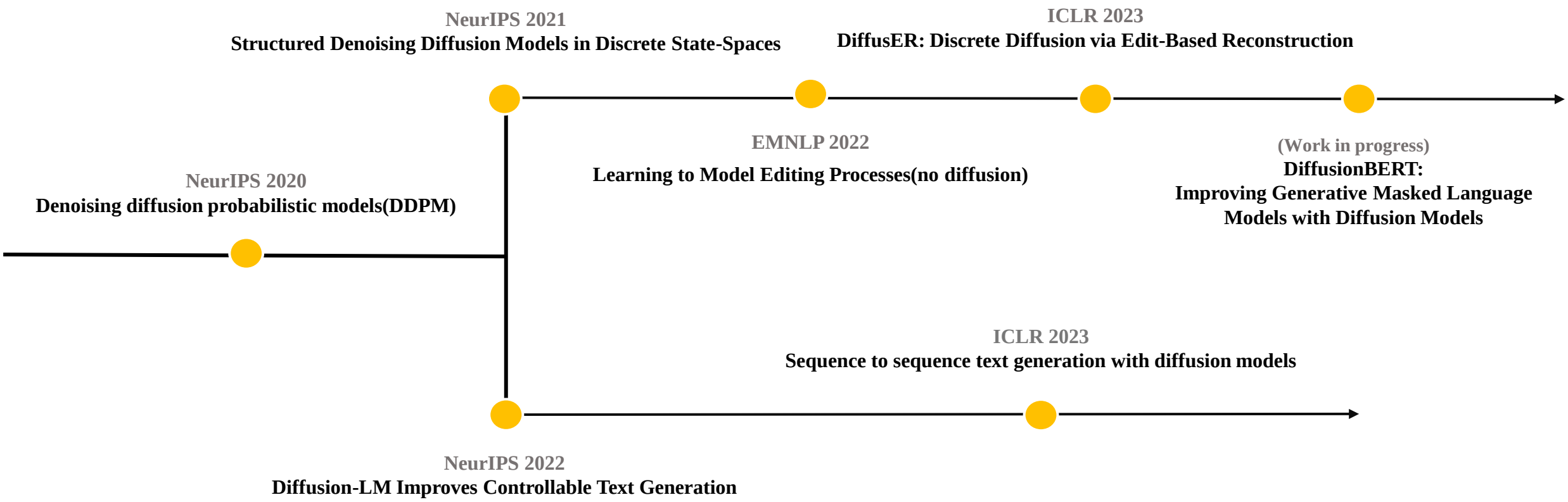
$$= \left(\frac{\beta_t}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}} + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\right)x_t - \frac{\sqrt{1-\bar{\alpha}_t}\beta_t}{(1-\bar{\alpha}_t)\sqrt{\alpha_t}}\epsilon$$

$$= \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon\right)$$

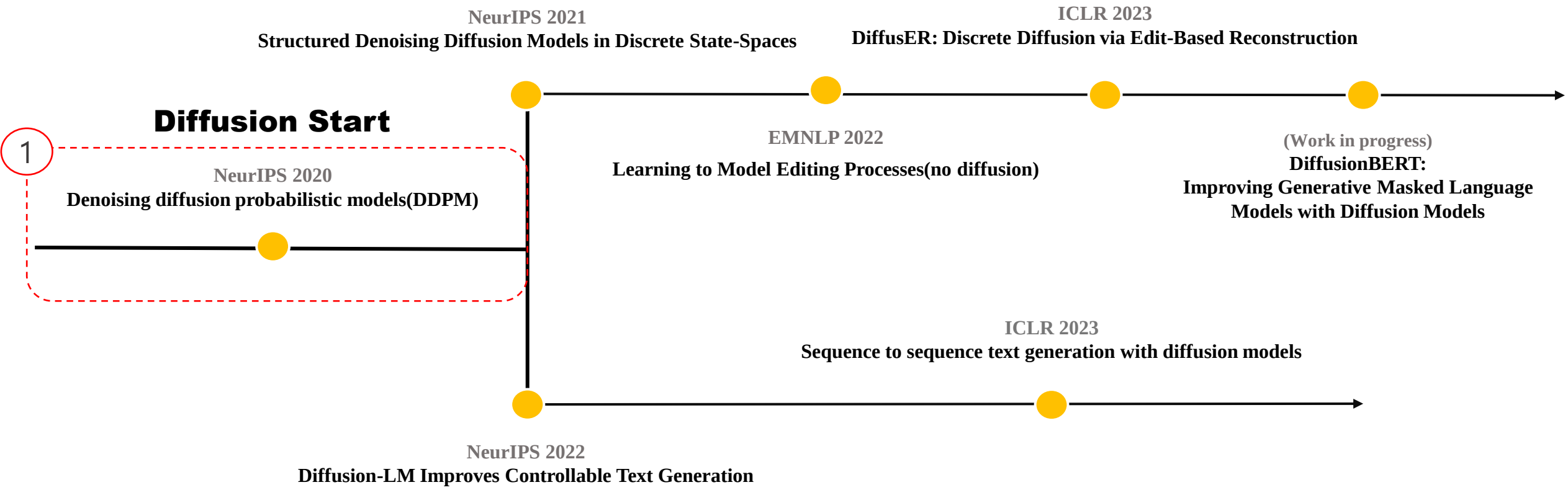
$$\mu_\theta = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta\right)$$

Text Diffusion Progress ??

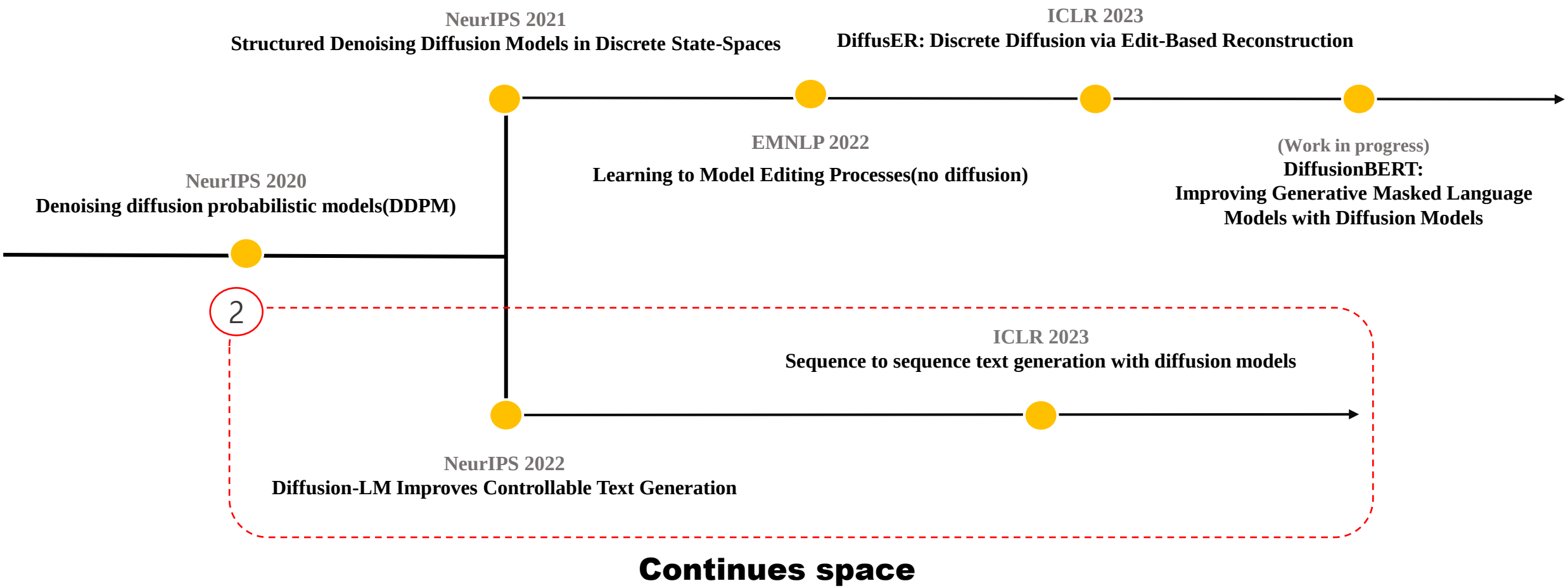
Text Diffusion Progress



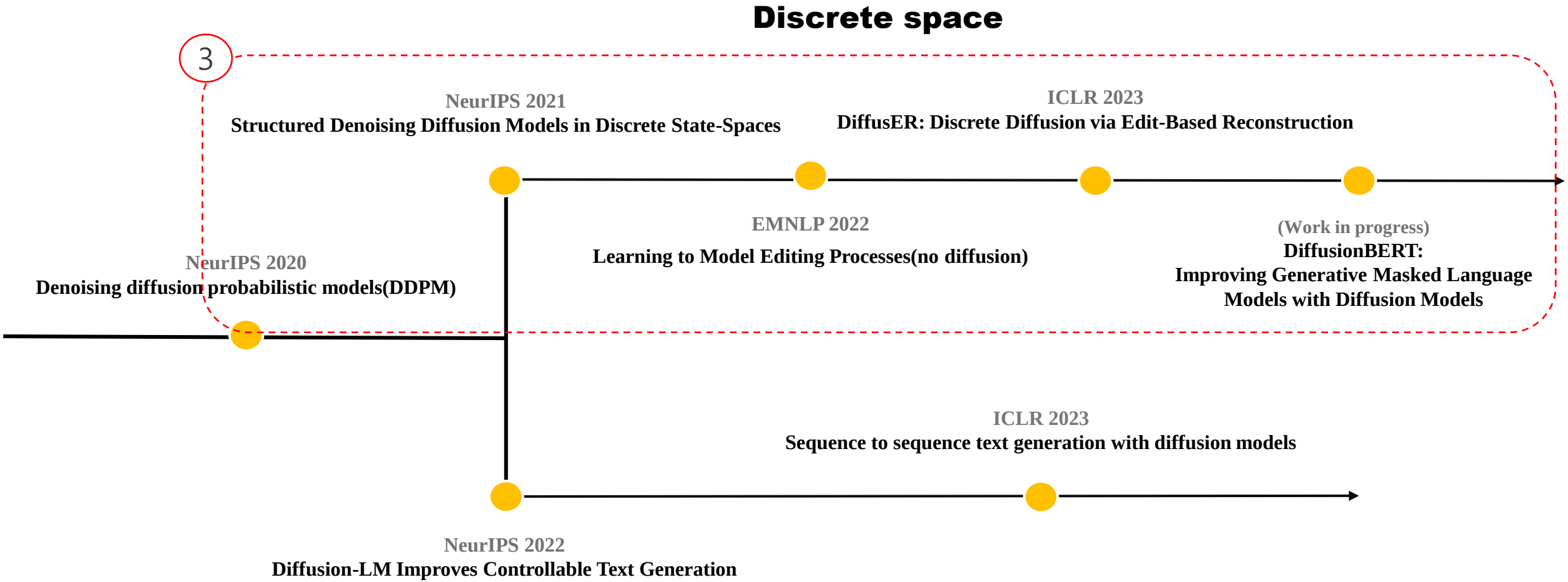
Text Diffusion Progress



Text Diffusion Progress



Text Diffusion Progress



Diffusion-LM Improves Controllable Text Generation

36th Conference on Neural Information Processing Systems (NeurIPS 2022)

Xiang Lisa Li
Stanford University
xlisali@stanford.edu

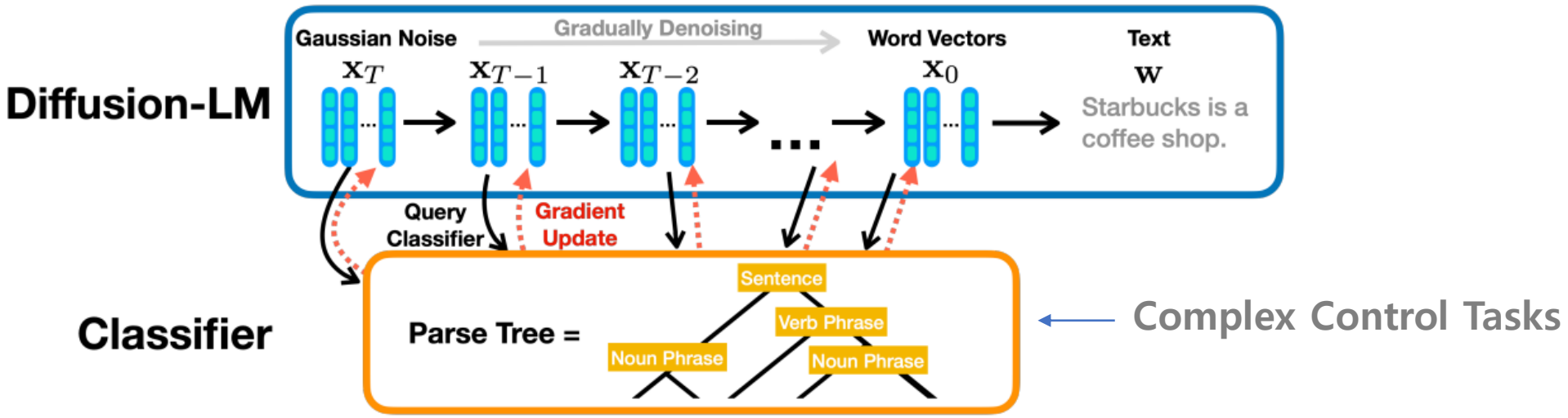
John Thickstun
Stanford University
jthickst@stanford.edu

Ishaan Gulrajani
Stanford Univeristy
igul@stanford.edu

Percy Liang
Stanford Univeristy
pliang@cs.stanford.edu

Tatsunori B. Hashimoto
Stanford Univeristy
thashim@stanford.edu

Introduction

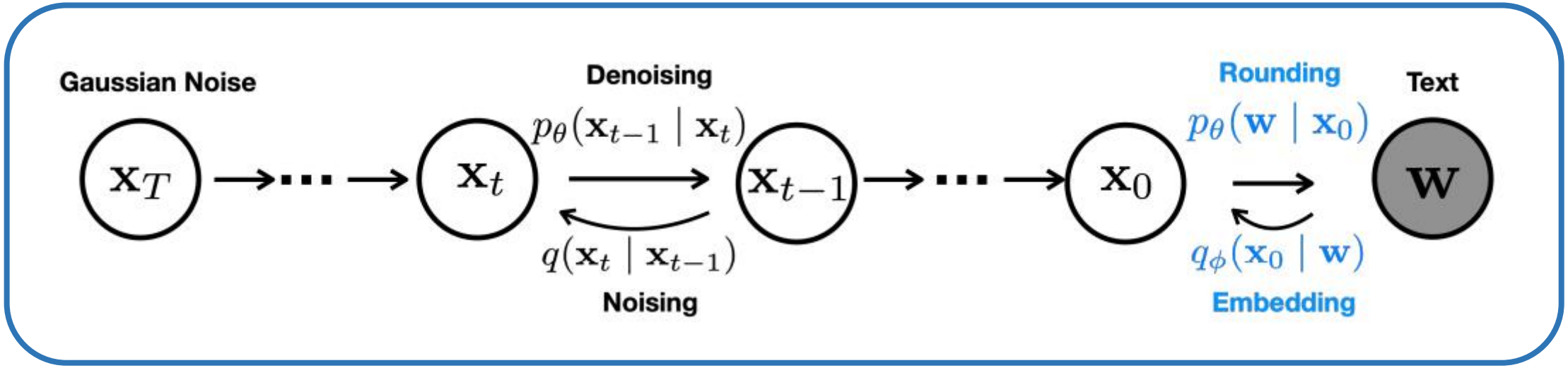


- **Continues domain**
  - Discrete domain의 text를 continues domain로 적용.

- **Controllable Text generation**
  - Gradual denoising steps 은 a hierarchy of continuous latent representations 생성.
  - 이를 통해 조금 더 복잡한 Control(Ex: Parse Tree)을 가능하게 함.

Diffusion-LM: Continuous Diffusion Language Modeling

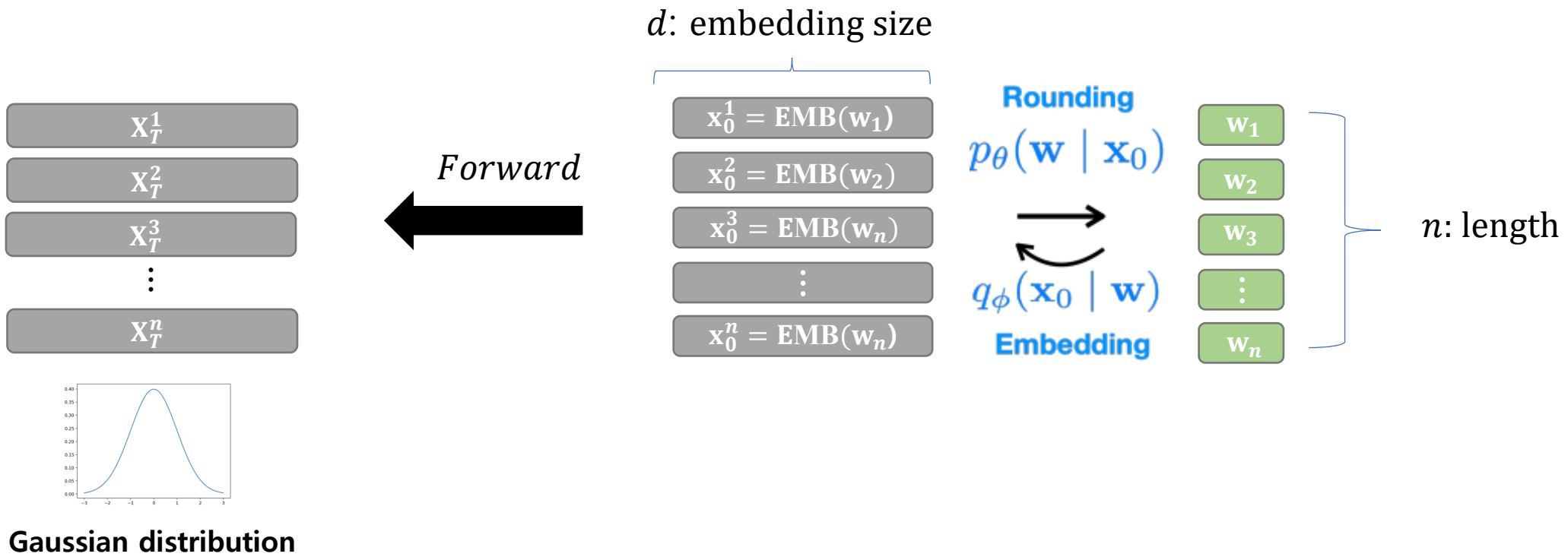
- Embedding step



- 각 토큰 $\mathbf{w}_i$ 을 백터 $\mathbb{R}^d$에 매핑하는 Embedding function $\mathbf{EMB}(\mathbf{w})$ 을 정의하여 continuous diffusion model에 discrete text를 적용.
  - $\mathbf{EMB}(\mathbf{w}) = [\mathbf{EMB}(w_1), \ldots, \mathbf{EMB}(w_n)] \in \mathbb{R}^{nd}$ *where* w: Word *sequence*, n : *Sequence length*

Diffusion-LM: Continuous Diffusion Language Modeling

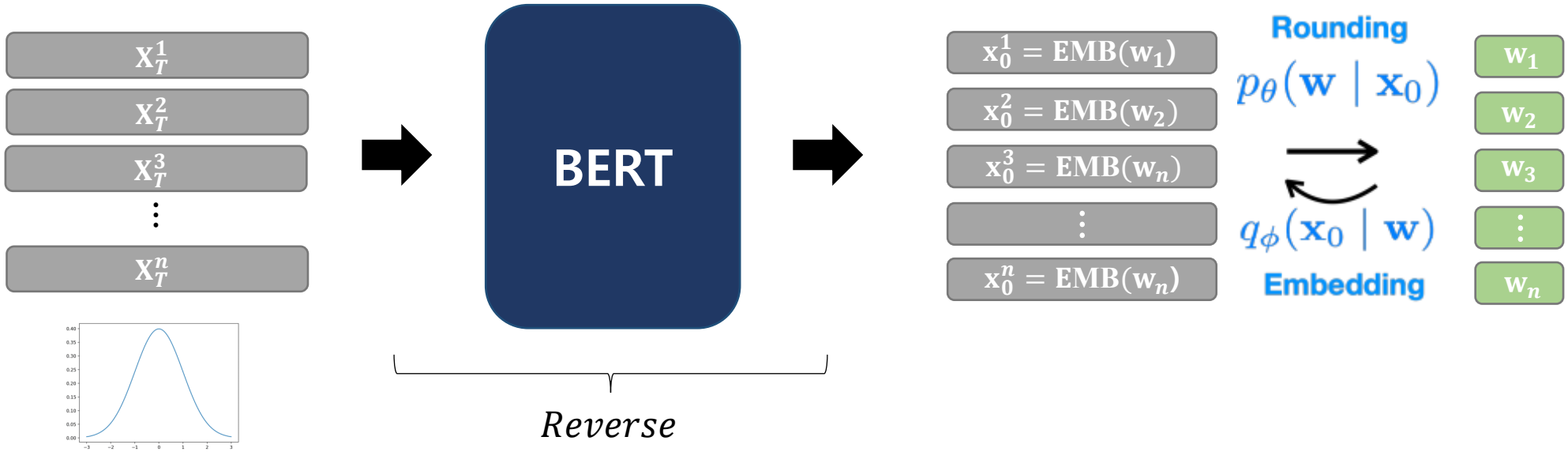
- Forward process



- $\mathbf{EMB}(\mathbf{w}) = [\mathbf{EMB}(\mathbf{w}_1), \ldots, \mathbf{EMB}(\mathbf{w}_n)] \in \mathbb{R}^{nd}$ *where* $\mathbf{w}$: *Word sequence* n *: Sequence length*
- Embedding (the forward process) : $q(\mathbf{x}_0|\mathbf{w}) = N(\mathbf{EMB}(\mathbf{w}), \boldsymbol{\sigma I})$
- $\mathbf{x}_t = \sqrt{\bar{\alpha}}\mathrm{EMB}(w_i) + \sqrt{1-\bar{\alpha}}\varepsilon \sim N(0, I)$

Diffusion-LM: Continuous Diffusion Language Modeling

- Reverse process



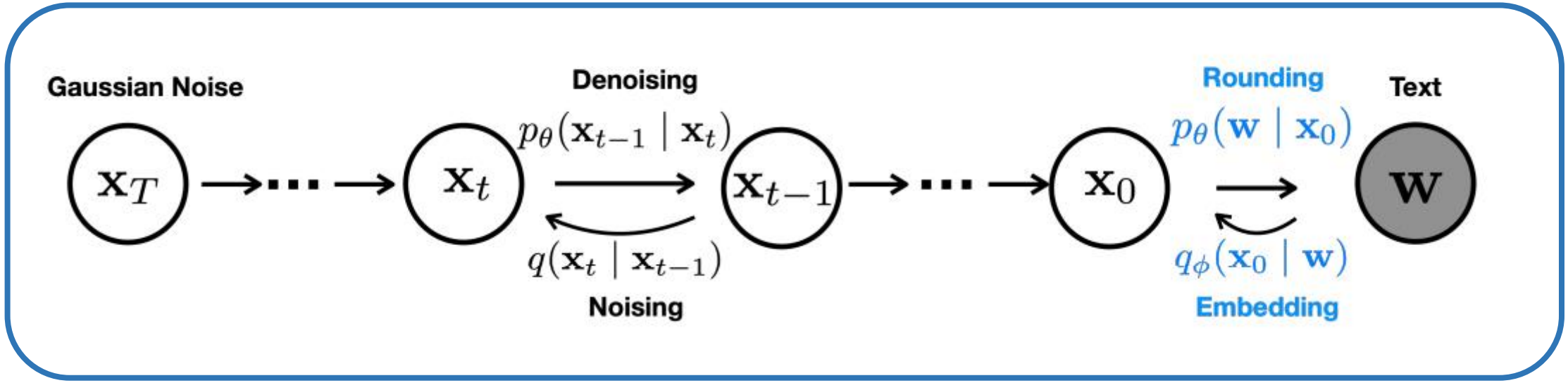
Gaussian distribution

- $\mathbf{x}_t$를 입력 받아 BERT모델은 $\mathbf{x}_0$ 를 predict함.
- Rounding step (the reverse process) : $p_\theta(\mathbf{w}|\mathbf{x}_0) = \prod_{i=1}^{n} p_\theta(\mathbf{w}_i|\mathbf{x}_0^i)$

Softmax distribution

Diffusion-LM: Continuous Diffusion Language Modeling

- Training



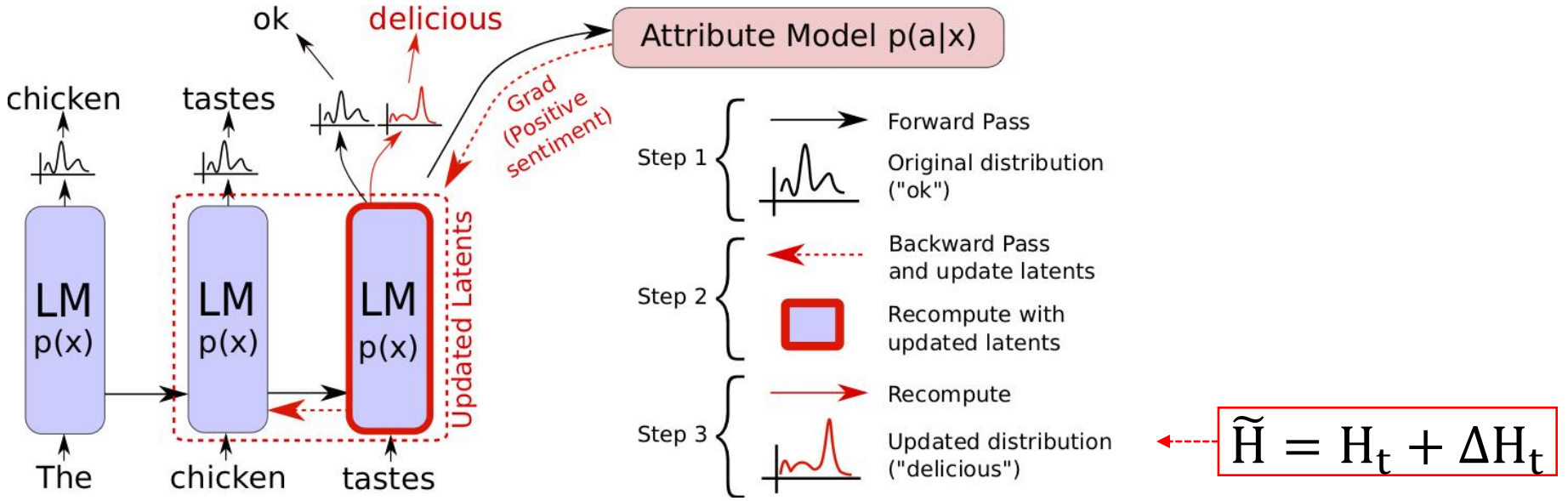
- Training loss function :

$$\mathcal{L}_{vlb}^{e2e}(\mathbf{w}) = \mathbb{E}_{q_\phi}\left[\overbrace{\mathcal{L}_{vlb}(x_0)}^{\text{Diffusion loss}} + \underbrace{\log_{q_\phi}(\mathbf{x}_0|\mathbf{w})}_{\text{Embedding}} - \underbrace{\log_{p_\theta}(\mathbf{w}|\mathbf{x}_0)}_{\text{Rounding}}\right]$$

Controllable Generation with Diffusion-LM

Plug and Play Language Models: A Simple Approach to Controlled Text Generation - ICLR 2020

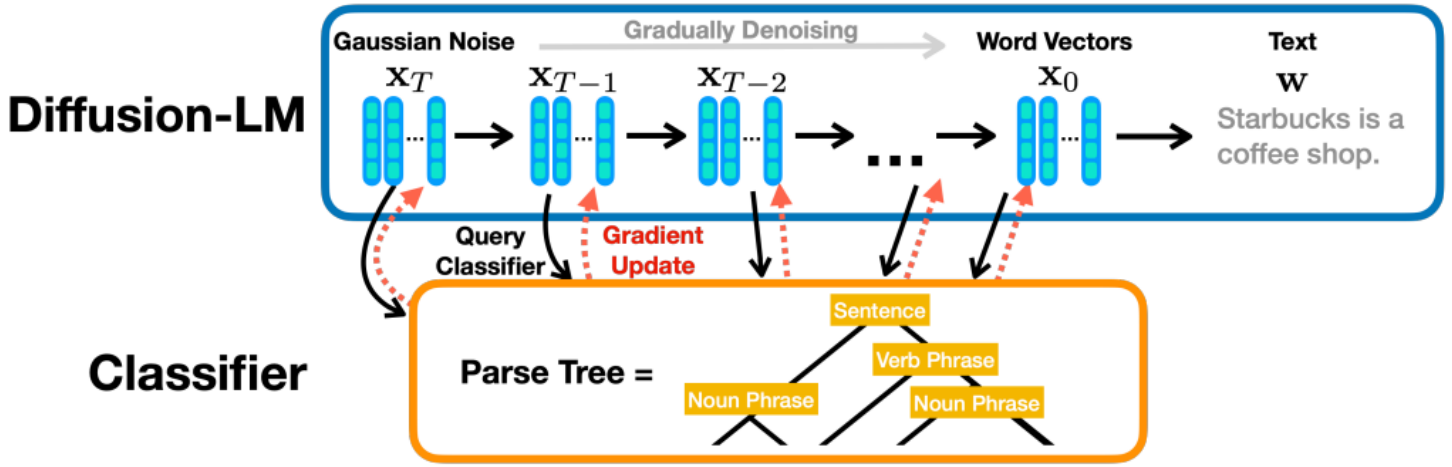
$$P(\mathrm{x}|\mathrm{a}) \propto P(\mathrm{a}|\mathrm{x}) \cdot P(\mathrm{x})$$



- Pretrained LM을 고정하고 생성 과정을 컨트롤 하는 방법.
- Plug and Play Language Model 방식은 아직까지 단순한 attribute 에서만 컨트롤이 가능.

Decoding and Controllable Generation with Diffusion-LM

- Controllable Text Generation



- Reverse 과정에서 Latent variable X_T~ X_0를 컨트롤 :

$$p_\theta(\mathbf{x}_{0:T}|\boldsymbol{c}) = \prod_{t=1}^{T} p(\mathbf{x}_{t-1}|\mathbf{x}_t, c)$$

Control target

$$P(A \mid B,C) = \frac{P(A,B,C)}{P(B,C)}$$
$$= \frac{P(B \mid A,C)\,P(A,C)}{P(B,C)}$$
$$= \frac{P(B \mid A,C)\,P(A \mid C)\,P(C)}{P(B,C)}$$
$$= \frac{P(B \mid A,C)\,P(A \mid C)P(C)}{P(B \mid C)P(C)}$$
$$= \frac{P(B \mid A,C)\,P(A \mid C)}{P(B \mid C)}$$

- 각각의 diffusion step에서 a sequence of control problems 로 decomposition:

$$p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{c}) \propto p(\mathbf{x}_{t-1}|\mathbf{x}_t) \cdot p(\boldsymbol{c}|\mathbf{x}_{t-1}, \mathbf{x}_t)$$

Decoding and Controllable Generation with Diffusion-LM

- Controllable Text Generation

- Sequence of control problems 을 위한 Decomposition:

$$p(\mathbf{x}_{t-1}|x_t, c) \propto p(\mathbf{x}_{t-1}|\mathbf{x}_t) \cdot p(\boldsymbol{c}|\mathbf{x}_{t-1}, \mathbf{x}_t)$$

- Conditional independence assumptions에 의해 $p(\mathbf{c}|\mathbf{x}_{t-1}, \mathrm{x}_t) \rightarrow p(\mathbf{c}|\mathbf{x}_{t-1})$ 간소화

$$p(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{c}) \propto p(\mathbf{x}_{t-1}|\mathbf{x}_t) \cdot p(\boldsymbol{c}|\mathbf{x}_{t-1})$$

- Gradient update 방법(SDE)을 통해 $\mathbf{x}_{t-1}$ 업데이트:

$$\nabla_{\mathbf{x}_{t-1}} logp(\mathbf{x}_{t-1}|\mathbf{x}_t, \boldsymbol{c}) = \nabla_{\mathbf{x}_{t-1}} logp(\mathbf{x}_{t-1}|\mathbf{x}_t) + \nabla_{\mathbf{x}_{t-1}} p(\boldsymbol{c}|\mathbf{x}_{t-1})$$

Diffusion-LM　　Classifier

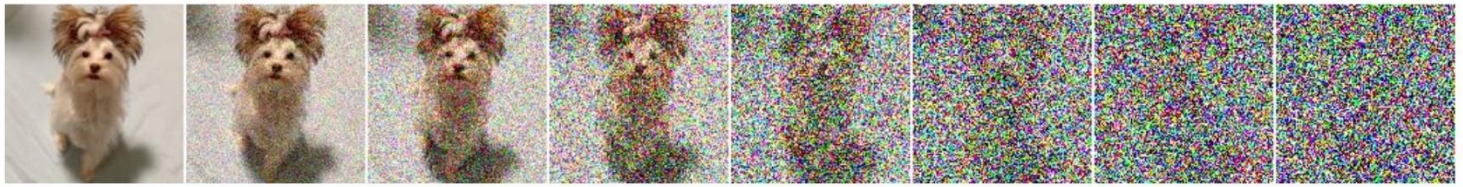
Score-based generative modeling through stochastic differential equations – ICLR2021

SDE : $d\mathbf{x} = \underbrace{f(\mathbf{x}, t)}_{\text{drift}} dt + \underbrace{g(t)}_{\text{diffusion}} d\mathbf{w}$

$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = N(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathbf{I})$

Forward SDE (data → noise)

$\mathbf{x}(0)$ —— $d\mathbf{x} = \mathbf{f}(\mathbf{x}, t)dt + g(t)d\mathbf{w}$ ——→ $\mathbf{x}(T)$



Anderson - 1982

score function

$\mathbf{x}(0)$ ←— $d\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, t) - g^2(t)\nabla_{\mathbf{x}} \log p_t(\mathbf{x})\right] dt + g(t)d\bar{\mathbf{w}}$ — $\mathbf{x}(T)$

Reverse SDE (noise → data)

$p(\mathbf{x}_T) = N(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$

$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = N(\mathbf{x}_{t-1}; \mu_\theta(\mathbf{x}_t, t), \Sigma_{\boldsymbol{\theta}}(\mathbf{x}_t, t)\mathbf{I})$

Score-based generative modeling through stochastic differential equations – Yang Song

- Reverse SDE(noise → data) :

Score function

$$\mathrm{d}\mathbf{x} = [\mathbf{f}(\mathbf{x}, t) - g^2(t)\nabla_x log p_t(\mathbf{x})]\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}$$

- Controllable generation :

$$\mathrm{d}\mathbf{x} = [\mathbf{f}(\mathbf{x}, t) - g^2(t)\nabla_x log p_t(\mathbf{x}|\mathbf{y})]\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}$$

- Bayes' rule :

$$\mathrm{d}\mathbf{x} = [\mathbf{f}(x, t) - g^2(t)\nabla_x log p_t(\mathbf{x}) - g^2(t)\nabla_x log p_t(\mathbf{y}|\mathbf{x})]\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}$$

Decoding and Controllable Generation with Diffusion-LM

- Controllable Text Generation

- Gradient update 방법(SDE)을 통해 $\mathbf{x}_{t-1}$ 업데이트:

$$\nabla_{x_{t-1}} logp(\boldsymbol{x}_{t-1}|x_t, c) = \boxed{\nabla_{x_{t-1}} logp(x_{t-1}|x_t)} + \boxed{\nabla_{x_{t-1}} p(c|x_{t-1})}$$

Diffusion-LM　　Classifier

$$dx = \left[f(x,t) - g^2(t)\nabla_{x_{t-1}} logp_t(x_{t-1}|x_t, c)\right]dt + g(t)dw$$

- $\mathbf{x}_{t-1}$을 sampling 한 다음 두 gradient를 계산한 뒤 $\mathbf{x}_{t-1}$를 업데이트

$$dx = \left[f(x,t) - (g^2(t)\boxed{\nabla_{x_{t-1}} logp_t(x_{t-1}|x_t)} + g^2(t)\boxed{\nabla_{x_{t-1}} logp_t(c|x_{t-1})}\right]dt + g(t)dw$$

Diffusion-LM　　Classifier

$$\nabla_{\mathbf{x}_{t-1}} logp(\mathbf{x}_{t-1}|\mathbf{x}_t, c) = \nabla_{\mathbf{x}_{t-1}} logp(\mathbf{x}_{t-1}|\mathbf{x}_t) + \nabla_{\mathbf{x}_{t-1}} p(\mathbf{c}|\mathbf{x}_{t-1})$$

Diffusion-LM　　Classifier

```
input_embs_param = th.nn.Parameter(sample) #sample을 업데이트 하기 위해 밑에 참조

with th.enable_grad():
    for i in range(K):
        optimizer = th.optim.Adagrad([input_embs_param], lr=step_size)
        optimizer.zero_grad()

        model_out = model_control(input_embs=input_embs_param, parse_chart=label_ids, t=tt)
        coef = coeff # 0.0005

        # 아래 수식 첨부
        if sigma.mean() == 0:
            logp_term = coef * ((mean - input_embs_param) ** 2 / 1.).mean(dim=0).sum()
        else:
            logp_term = coef * ((mean - input_embs_param)**2 / sigma).mean(dim=0).sum()

        loss = model_out.loss + logp_term
        loss.backward()
        optimizer.step()
```

Diffusion model이 sampling 한 $\mathbf{x_{t-1}}$

Pretrained Control Model
(ex: syntactic parser)

참고자료.6

$$logp(\mathbf{x}_{t-1}|\mathbf{x}_t) = -\frac{1}{2}(\mathbf{x}_{t-1} - \mu)^T \Sigma^{-1}(\mathbf{x}_{t-1} - \mu) + C$$

참고 자료.6

Diffusion Models Beat GANs on Image Synthesis – Supplementary Material :
(https://openreview.net/attachment?id=AAWuCvzaVt&name=supplementary_material)

D.2 Deriving Algorithm 1: Conditional Sampling for DDPM

We showed in the previous section that to condition a diffusion process on a label y, it suffices to sample each transition[2] according to

$$p_{\theta,\phi}(x_t|x_{t+1}, y) = Z p_\theta(x_t|x_{t+1}) p_\phi(y|x_t) \tag{48}$$

where Z is a normalizing constant. It is typically intractable to sample from this distribution exactly, but Sohl-Dickstein et al. [63] show that it can be approximated as a perturbed Gaussian distribution. Here, we review this derivation.

Recall that our diffusion model predicts the previous timestep x_t from timestep x_{t+1} using a Gaussian distribution:

$$p_\theta(x_t|x_{t+1}) = \mathcal{N}(\mu, \Sigma) \tag{49}$$

$$\log p_\theta(x_t|x_{t+1}) = -\frac{1}{2}(x_t - \mu)^T \Sigma^{-1}(x_t - \mu) + C \tag{50}$$

Main Results

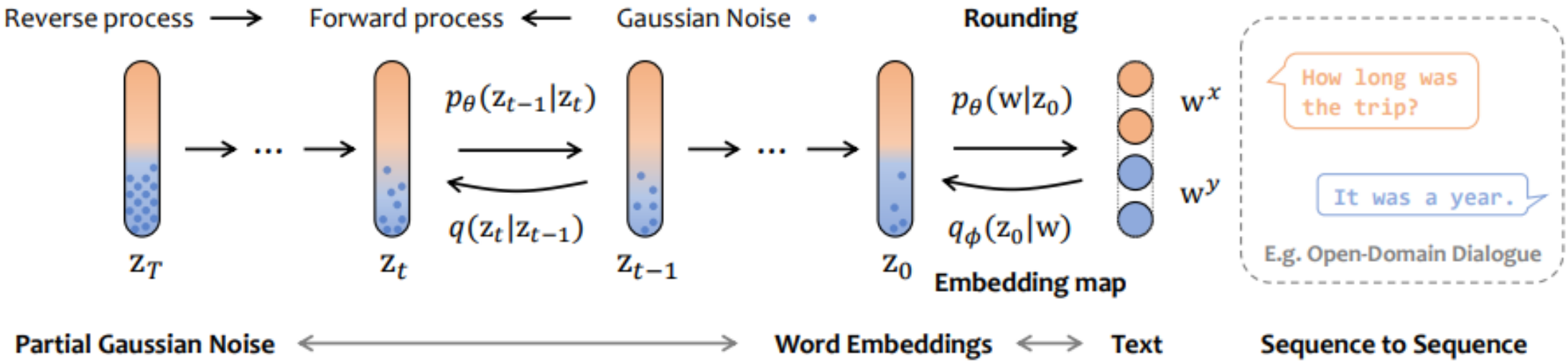
- Syntax span

target span	[4, 16, VP]
FUDGE	The Cambridge Blue pub is near the Café Brazil and offers a high price range for their French food .
Diffusion-LM	On the Ranch there is a children friendly pub called The Cricketers with an average customer rating .
FT	The Travellers Rest Beefeater is an average rated restaurant located in the riverside area near Café Adriatic . Their price range is less than £ 20 .
target span	[0, 2, NP]
FUDGE	The Golden Palace is a cheap , 5 - star coffee shop , located on the river in the north of the city centre .
Diffusion-LM	The Olive Grove is a pub that provides Indian food in the high price range . It is in the city centre .
FT	The Golden Curry is located in city centre near Café Rouge which provides English food . Its customer rating is average and is not family - friendly .
target span	[12, 13, NP]
FUDGE	The Waterman is a family friendly place with a good rating .[missing span]
Diffusion-LM	The Vaults is a high priced , family friendly restaurant that serves Italian food .
FT	Strada is a restaurant which costs less than £ 20 , but is not family - friendly and has an average rating .

DIFFUSEQ : Sequence to sequence text generation with diffusion models

(Under review as a conference paper at ICLR2023)

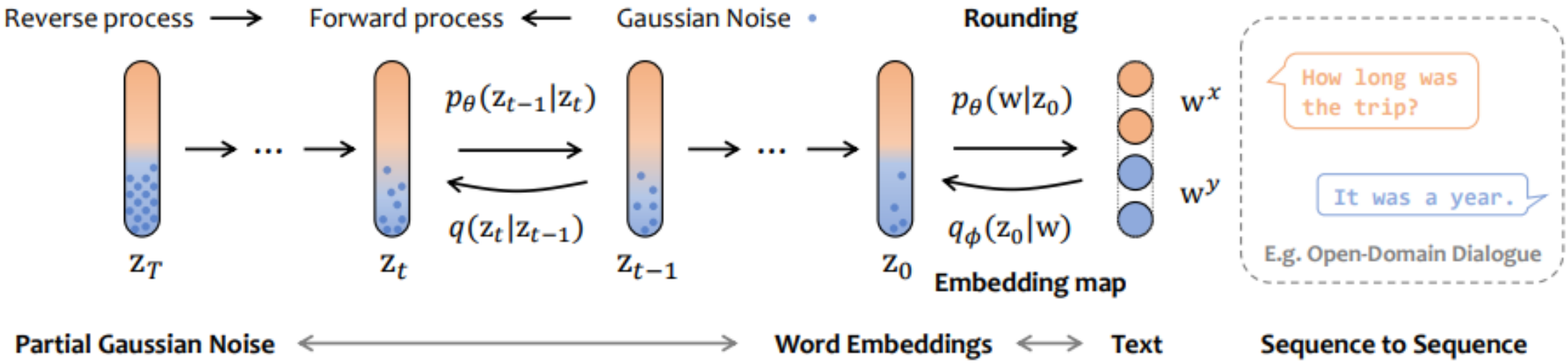
DIFFUSEQ



- Text generation의 대부분은 sequence-to-sequence (Seq2Seq) 문제임
- Diffusion-LM에서 Seq2Seq의 다양한 semantic meaning을 모델링 하는 Classifier를 학습시킬 수 없음

DIFFUSEQ

- Embedding step



- Embedding(Forward process) :

$$\text{EMB}(\mathbf{w}^{x\oplus y}) = [\text{EMB}(w_1^x), \ldots, \text{EMB}(w_m^x), \text{EMB}(w_1^y), \ldots \text{EMB}(w_m^y)] \in \mathbb{R}^{(m+n)\times d}.$$

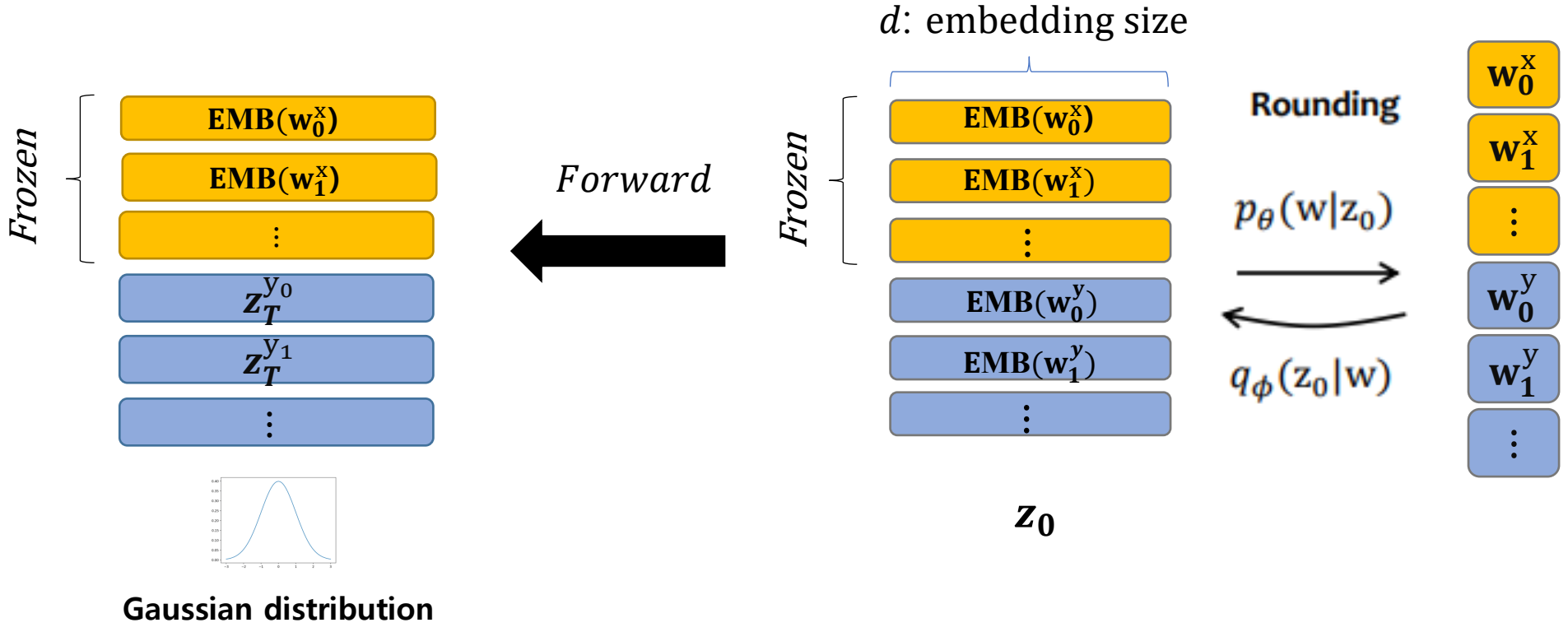
Frozen

- Embedding transformation:

$$q_\phi(\mathbf{z}_0|\mathbf{w}^{x\oplus y}) = \mathcal{N}(\text{EMB}(\mathbf{w}^{x\oplus y}), \beta_0\mathbf{I})$$

DIFFUSEQ

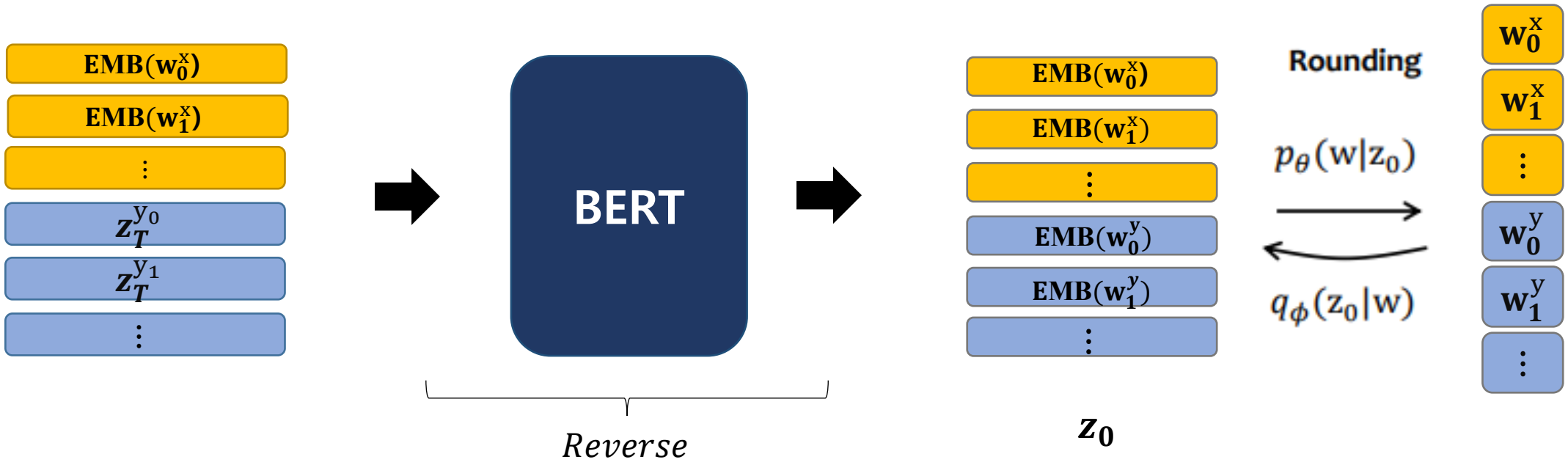
- Forward process



- Forward 를 통해 $\mathbf{EMB(w^y)}$ 만 노이즈를 추가
- $z_t^y = \sqrt{\bar{\alpha}} z_0^y + \sqrt{1-\bar{\alpha}}\varepsilon$

DIFFUSEQ

- Reverse process



- 입력으로 sequence 전체가 입력되며 reverse process를 통해 $\mathbf{w}^y$ 만 복원
- Diffusion-LM과 loss가 같으며, $\mathbf{w}^y$ 만 loss에 사용

DIFFUSEQ

- Generation Results

(Quora Question Pairs)

Table 2: Sample outputs in QQP test set, conditioned on the same **x**.

Original sentence: *How do I make friends.*	***Paraphrase reference***: *How to make friends ?*	
GPT2-large finetune	**GPVAE-T5**	**DIFFUSEQ**
How can I make friends?	How can I make friends?	How can I make friends better?
How can I make friends?	How do I make friends?	How can I make friends?
How can I make friends?	How can I make friends?	How do you make friends?
How can I make friends?	How can I make friends?	What is the best way to make friends?
How do I make friends and keep them?	What's the best way to make friends and make make friends?	How can I make friends and more something?

high diversity

DIFFUSEQ

- Dialogue test set

Table 5: Sample outputs with different random seed in Dialogue test set.

Utterance: *How long does the dye last?*
Response: *Just did this two days ago, not sure how it'll fade yet!*

GPVAE-T5	**GPT2-large finetune**	**DIFFUSEQ**
* I'm not sure, I'm not sure. I've tested it a few times, but I don't know for sure. I've	* Two weeks in my case.	* About an hour, 5 days or so.
* I'm not sure. I'm not sure how long it lasts, I'm sure it 'll get better. It's been a while since	* I've had it for about a year.	* 4 days.
* I've been using it for about a year and a half. I've been using it for about a year and a half.	* The dye can sit around for a month then you can wash it.	* I'm not sure about this, about the same kind of time.

Non-autoregressive
반복 문제

DIFFUSEQ

- Question Generation

Statement: *The Japanese yen is the official and only currency recognized in Japan.* ***Question***: *What is the Japanese currency?*	
GPVAE-T5 * What is the japanese currency * What is the japanese currency * What is the japanese currency	**NAR-LevT** * What is the basic unit of currency for Japan ? * What is the basic unit of currency for Japan ? * What is the basic unit of currency for Japan ?
GPT2-large finetune * What is the basic unit of currency for Japan? * What is the Japanese currency * What is the basic unit of currency for Japan?	**DIFFUSEQ** * What is the Japanese currency * Which country uses the "yen yen" in currency * What is the basic unit of currency?

DIFFUSEQ

- Text Simplification test set

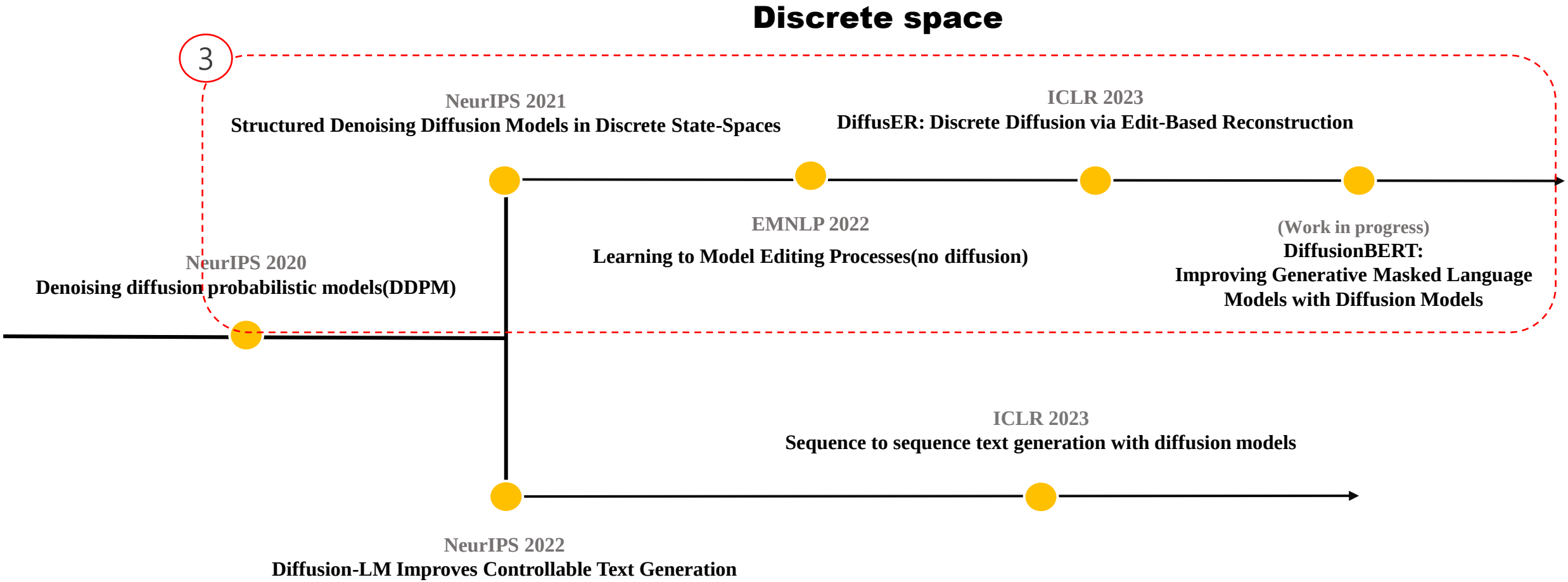
Complex sentence : 사람들은 여러 가지 이유로 외로움을 경험할 수 있으며 어린 시절과 청소년기의 우정 관계 부족 또는 주변에 의미 있는 사람이 물리적으로 없는 것과 같은 많은 삶의 사건이 외로움을 유발할 수 있습니다.
Simplified : 외로움의 원인 중 하나는 어린 시절과 십대 시절에 친구가 없었기 때문입니다.

Table 7: Sample outputs with different random seed in Text Simplification test set.

Complex sentence: *People can experience loneliness for many reasons, and many life events may cause it, such as a lack of friendship relations during childhood and adolescence, or the physical absence of meaningful people around a person.*
Simplified: *One cause of loneliness is a lack of friends during childhood and teenage years.*

GPVAE-T5	**NAR-LevT**	**GPT2-large finetune**
		* Loneliness can be caused by many things.
* People can experience loneliness for many reasons, and many life events may cause it, such as a lack of friendship relations during childhood and adolescence, or the physical absence of meaningful people around a person	* People may experience reashapphap-phapphapphapphappabout life reasit.	* Loneliness can affect people in many ways.
* People can experience loneliness for many reasons, and many life events may cause it, such as a lack of friendship relations during childhood and adolescence, or the physical absence of meaningful people around a person	* People may experience reashapphap-phapphapphapphappabout life reasit.	* Loneliness can be caused by many things.
* People can experience loneliness for many reasons, and many life events may cause it, such as a lack of friendship relations during childhood and adolescence, or the physical absence of meaningful people around a person	* People may experience reashapphap-phapphapphapphappabout life reasit.	**DIFFUSEQ** * Many life events may cause of loneliness * People can also be very experience loneliness for many reasons. * People can experience loneliness for many reasons, and many life events may, cause it.

Text Diffusion Progress



Structured Denoising Diffusion Models in Discrete State-Spaces

NeurIPS 2021

Diffusion models for discrete state spaces

- 기존 forward 의 가우시안 노이즈 추가를 **transition probabilities matrices** 로 변경

$$[Q_t]_{ij} = q(x_t = j | x_{t-1} = i)$$

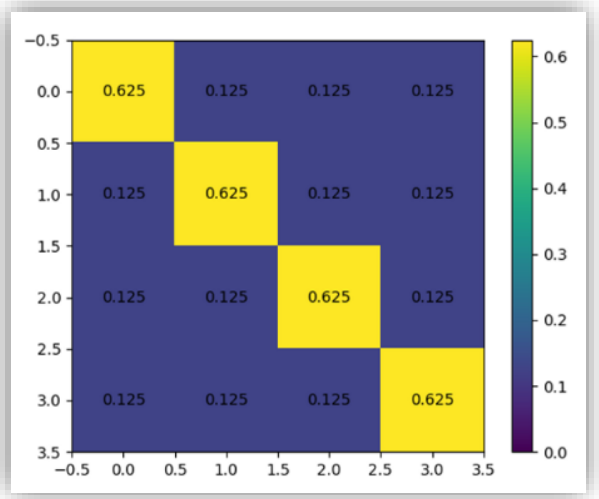
- EX) X_t, X_{t-1}이 K 개의 categories에 포함, K=3이라 가정
  - $q(x_t|x_{t-1}) = Cat(x_t; p = x_{t-1}Q_t)$

$$\mathrm{x_{t-1}} = [0\ 1\ 0], \qquad \mathrm{Q_t} = \begin{bmatrix} 0.6 & 0.2 & 0.2 \\ 0.2 & 0.6 & 0.2 \\ 0.2 & 0.2 & 0.6 \end{bmatrix}$$

$$p = \mathbf{x}_{t-1}Q_t = [0\ \ 1\ \ 0] \begin{bmatrix} 0.6 & 0.2 & 0.2 \\ 0.2 & 0.6 & 0.2 \\ 0.2 & 0.2 & 0.6 \end{bmatrix} = [0.2\ \ 0.6\ \ 0.2]$$

Uniform(Random token)

$$[Q_t]_{ij} = \begin{cases} 1 - \frac{K-1}{K}\beta_t & \text{if} \quad i = j \\ \frac{1}{K}\beta_t & \text{if} \quad i \neq j \end{cases}$$




```python
import numpy as np

x0 = np.array([0.1, 0.5, 0.3, 0.1])
x = x0

Matrix = [[0.625, 0.125, 0.125, 0.125],
          [0.125, 0.625, 0.125, 0.125],
          [0.125, 0.125, 0.625, 0.125],
          [0.125, 0.125, 0.125, 0.625]]

for idx in range(10):
    x = x @ Matrix
    print(idx, np.round(x, 4))
```

```
0 [0.175  0.375  0.275  0.175]
1 [0.2125 0.3125 0.2625 0.2125]
2 [0.2312 0.2812 0.2562 0.2312]
3 [0.2406 0.2656 0.2531 0.2406]
4 [0.2453 0.2578 0.2516 0.2453]
5 [0.2477 0.2539 0.2508 0.2477]
6 [0.2488 0.252  0.2504 0.2488]
7 [0.2494 0.251  0.2502 0.2494]
8 [0.2497 0.2505 0.2501 0.2497]
9 [0.2499 0.2502 0.25   0.2499]
```

sampling

Random token

Diffusion models for discrete state spaces

- Training loss

- *Forward process와 Transition matrix을 정의*

$$q(x_t|x_{t-1}) = Cat(x_t; p = x_{t-1}Q_t)\ ,\ [Q_t]_{ij} = q(x_t = j|x_{t-1} = i)$$

- x_t는 x_0에 Q를 여러 번 곱한 것과 같음

$$q(x_t|x_0) = cat\left(x_t; p = x_0\overline{Q}_t\right), \qquad with\ \overline{Q}_t = Q_1Q_2 \dots Q_t$$

- *Posterior를 정의*

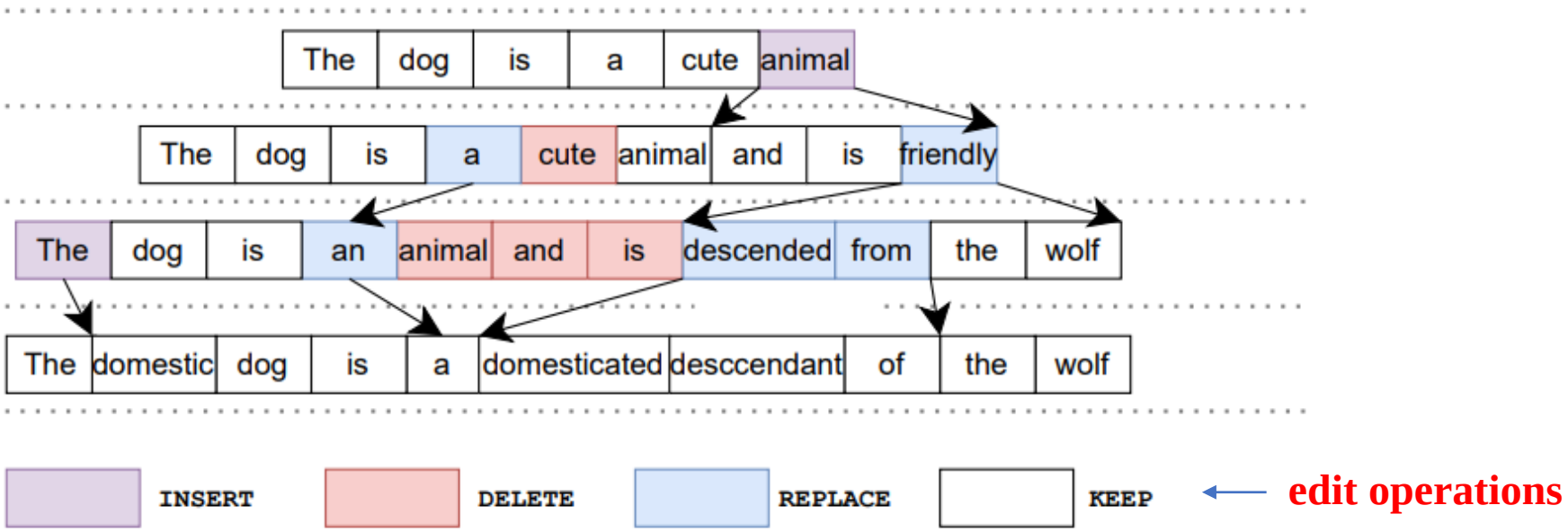
$$q(x_{t-1}|x_t, x_0) = \frac{q(x_t|x_{t-1}, x_0)q(x_{t-1}|x_0)}{q(x_t|x_0)} = cat\left(x_{t-1}; p = \frac{x_tQ_t^{\intercal} \odot x_0\overline{Q}_{t-1}}{x_0\overline{Q}_t x_t^{\intercal}}\right)$$

- *Training loss* : $\mathrm{KL}(q(x_{t-1}|x_t, x_0)||p_\theta(x_{t-1}|x_t)) = \sum_i p_i \log\left(\frac{p_i}{q_i}\right)$

DiffusER: Discrete Diffusion via Edit-based Reconstruction

ICLR 2023

Discrete Diffusion Model using Editing Processes



- Levenshtein(1966)이 제안한 edit operations을 edit process에 적용
- 단순히 이전 문장을 보고 다음 문장을 예측 하는게 아니라 이전 edit process의 context를 바탕으로 다음 문장을 예측

DIFFUSER

- EDIT-BASED CORRUPTION

  - 4가지 Levenshtein 편집 작업을 통해 임의의 토큰 시퀀스를 다른 토큰 시퀀스로 으로 변환

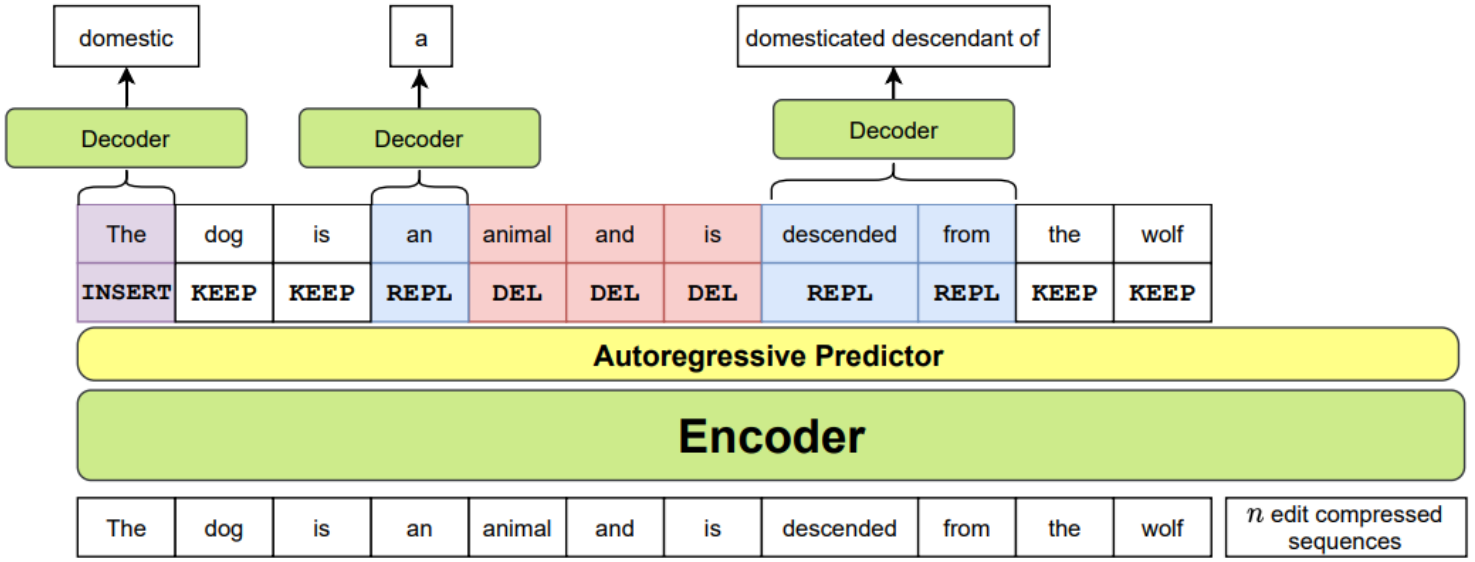
  - t 번째, **forward process** : $q(\mathbf{x}_t|\mathbf{x}_{t-1}; \boldsymbol{\varepsilon_t}, \boldsymbol{\varepsilon_l})$
    - $\boldsymbol{\varepsilon_t}$: 편집 유형에 대한 분포 (e.g. 60% keep, 20% replace, 10% delete, 10% insert)
    - $\boldsymbol{\varepsilon_l}$: 편집 길이에 대한 분포

  - 손상 된 시퀀스 $\boldsymbol{X_T}$가 주어졌을 때, 모델은 손상을 되돌릴 수 있는 프로세스를 학습하는 것이 목표

$$P_\theta(\mathbf{x}_0) = \prod_{t=0}^{T} p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$$

DIFFUSER

- EDIT-BASED RECONSTRUCTION



- 모델은 편집 프로세스를 포함하기 위해 두 단계를 포함
  - Encoder : 편집 판별 프로세스
  - Decoder : 수정 또는 삽입의 경우 token을 생성

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = p_\theta^{\text{tag}}(\mathbf{e}_t|\mathbf{x}_t)p_\theta^{\text{gen}}(\mathbf{x}_{t-1}|\mathbf{x}_t,\mathbf{e}_t)$$

Step 1	elf meantime Nano (¡ Aden Prepare ~~hue mere strictlyrights~~ hueHeat Goalsgeordnet LewisSession beet remindersrights rézes redund boldWisconsin-Port compl rocks@@~~actual Parish norm Lawyers Organisation deprecatedince eradicateewerkschaften~~ oyleingebracht naked Lawyers Organisation von Gewerkschaften al contestants negligible GeneIZE etablieren.HT
Step 2	elf meantime Nano (¡ ~~Aden Prepare hueHeat Goalsgeordnet~~ aggravatedabgeordnet LewisSession beet remindersrights rézes redund ~~boldWisconsinPort compl rocks@@oyleingebracht~~ boldWisconsin eingebracht naked Lawyers Organisation von Gewerkschaften al contestants negligible 2400 CLR GeneIZE etablieren.~~HT~~ isationatar ent
Step 3	~~elf meantime Nano (¡ aggravatedabgeordnet~~ containing Tai Prison Kongressabgeordnet und John LewisSession beet remindersrights ~~rézes redund boldWisconsin~~ rézesvorschlag eingebracht naked Lawyers Organisation von Gewerkschaften ~~al contestants negligible 2400 CLR GeneIZE~~ als Bürgerrecht zu etablieren.isationatar ent
Step 4	~~containing Tai Prison Kongressabgeordnet und John LewisSession beet remindersrights rézesvorschlag eingebrachtnaked Lawyers~~ Die Kongressabgeordneten Keith Ellison und John Lewis haben einen Gesetzesvorschlag eingebracht, um die Organisation von Gewerkschaften als Bürgerrecht zu etablieren. ~~isationatar ent~~
Target	Die Kongressabgeordneten Keith Ellison und John Lewis haben einen Gesetzesvorschlag eingebracht, um die Organisation von Gewerkschaften als Bürgerrecht zu etablieren.

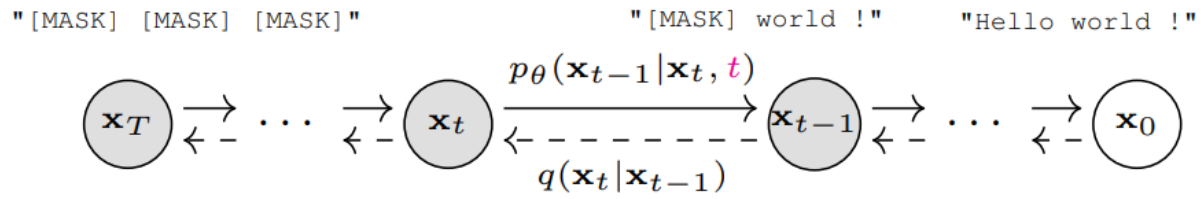
Table 1: Example diffusion process for machine translation from random tokens.

Source Document	(CNN)They're not gonna take it anymore. Really. Twisted Sister says that its 2016 tour will be its last, according to a press release. Next year marks the band's 40th anniversary, and to celebrate, the tour is being titled "Forty and F*ck It." "It's official: Farewell," Twisted Sister singer Dee Snider posted on Facebook. Snider also noted that the band will play with a new drummer, Mike Portnoy of Adrenaline Mob. Portnoy replaces A.J. Pero, who died March 20. The band will also perform two shows in Pero's honor: one at Las Vegas' Hard Rock Hotel and Casino, the other at the Starland Ballroom in Sayreville, New Jersey. The latter is in support of Pero's family. Twisted Sister's biggest hit, "We're Not Gonna Take It," hit the Top Forty in 1984 and was featured in a popular video.
Step 1	~~(CNN)They're not gonna take it anymore. Really.~~ Twisted Sister says that its 2016 tour will be its last, according to a press release. Next year marks the band's 40th anniversary, and to celebrate, the tour is being titled "Forty and F*ck It." "It's official: Farewell," Twisted Sister singer Dee Snider posted on Facebook. Snider also noted that the band will play with a new drummer, Mike Portnoy of Adrenaline Mob. Portnoy replaces A.J. Pero, who died March 20. The band will also perform two shows in Pero's honor: one at Las Vegas' Hard Rock Hotel and Casino, the other at the Starland Ballroom in Sayreville, New Jersey. The latter is in support of Pero's family. Twisted Sister's biggest hit, "We're Not Gonna Take It," hit the Top Forty in 1984 and was featured in a popular video.
Step 2	Twisted Sister says that its 2016 tour will be its last, according to a press release. Next year marks the band's 40th anniversary, and to celebrate, the tour is being titled "Forty and F*ck It." ~~"It's official: Farewell," Twisted Sister singer Dee Snider posted on Facebook. Snider also noted that the band will play with a new drummer, Mike Portnoy of Adrenaline Mob.~~ Portnoy replaces A.J. Pero, who died March 20. The band will also perform two shows in Pero's honor: one at Las Vegas' Hard Rock Hotel and Casino, the other at the Starland Ballroom in Sayreville, New Jersey. The latter is in support of Pero's family. Twisted Sister's biggest hit, "We're Not Gonna Take It," hit the Top Forty in 1984 and was featured in a popular video.
Step 3	Twisted Sister says that its 2016 tour will be its last, according to a press release. Next year marks the band's 40th anniversary, and to celebrate, the tour is being titled "Forty and F*ck It." Portnoy replaces A.J. Pero, who died March 20. The band will ~~also~~ perform two shows in Pero's honor ~~: one at Las Vegas ' Hard Rock Hotel and Casino, the other at the Starland Ballroom in Sayreville, New Jersey. The latter is in support of Pero's family. Twisted Sister's biggest hit, "We're Not Gonna Take It," hit the Top Forty in 1984 and was featured in a popular video~~in Las Vegas and New Jersey.
Step 4	Twisted Sister says that its 2016 tour will be its last, according to a press release. Next year marks the band's 40th anniversary, and to celebrate, the tour is being titled "Forty and F*ck It." ~~Portnoy replaces~~ A.J. Pero, ~~who~~ died March 20. The band will perform two shows in Pero's honor in Las Vegas and New Jersey.
Generated Summary	Twisted Sister says that its 2016 tour will be its last. Next year marks the band's 40th anniversary, and to celebrate, the tour is being titled "Forty and F*ck It." A.J. Pero, died March 20. The band will perform two shows in Pero's honor in Las Vegas and New Jersey.

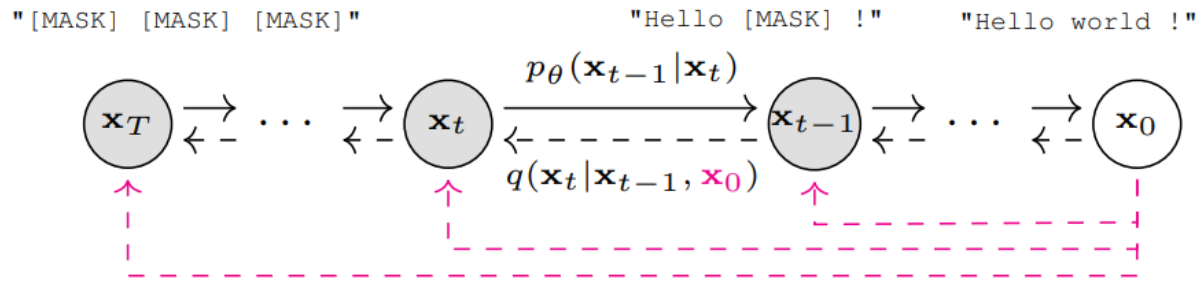
DiffusionBERT: Improving Generative Masked Language Models with Diffusion Models

(Work in progress)

DiffusionBERT



(a) Diffusion models for discrete data



(b) Non-Markovian DiffusionBERT

- **Pre-trained LM 을 사용**
  - Absorbing state를 적용 하는 것이 pre-trained LM을 사용하는데 효과적
- **Spindle Noise Schedule**
  - 새로운 noise schedule 방식을 제안하여 Non-Markovian forward process를 정의

Diffusion models with a Discrete Absorbing state

- **Markov** 프로세서에서 **absorbing state**
  - 각 토큰은 동일하게 유지되거나 약간의 확률로 [MASK]토큰으로 전환(transition)

- $\boldsymbol{t}$ step에서 **Transition matrix:**

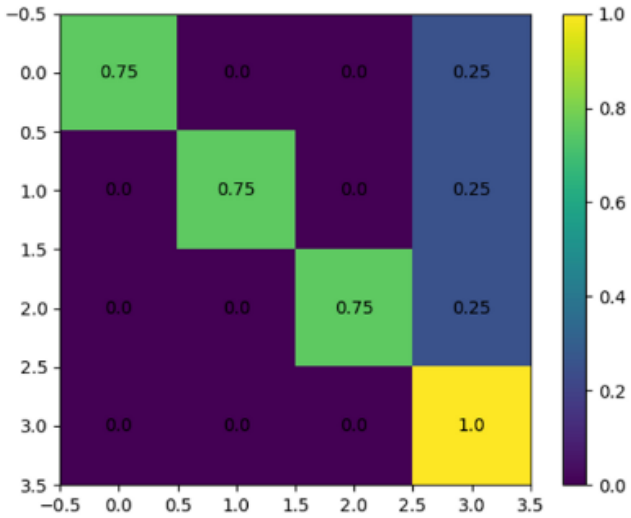
$$[Q_t]_{ij} = \begin{cases} 1 & if \;\; i = j = [M] \\ 1 - \beta_t & if \; i = j \;\; \neq [M] \\ \beta_t & if \; j = [M], i \neq [M] \end{cases}$$

- $\boldsymbol{t}$ step에서 **Marginal** $\boldsymbol{q}(x_i^i|x_0^i)$**:**

$$\boldsymbol{q}\left(x_i^i \middle| x_0^i\right) = \begin{cases} \overline{\boldsymbol{\alpha}}_{\boldsymbol{t}} & \boldsymbol{if} \;\; \boldsymbol{x_t^i = x_0^i} \\ \mathbf{1} - \overline{\boldsymbol{\alpha}}_{\boldsymbol{t}} & \boldsymbol{if} \; \boldsymbol{x_t^i} = [\boldsymbol{M}] \end{cases}$$

Absorbing state

$$[Q_t]_{ij} = \begin{cases} 1, & if\ i = j = [M] \\ 1 - \beta_t, & if\ \ i = j \neq [M] \\ \beta_t, & if\ j = [M], i \neq [M] \end{cases}$$




```python
import numpy as np

x0 = np.array([0, 1.0, 0, 0.])
x = x0

Matrix = [[0.75, 0.00, 0.00, 0.25],
          [0.00, 0.75, 0.00, 0.25],
          [0.00, 0.00, 0.75, 0.25],
          [0.00, 0.00, 0.00, 1.00]]

for idx in range(10):
    x = x @ Matrix
    print(idx, np.round(x, 3))
```

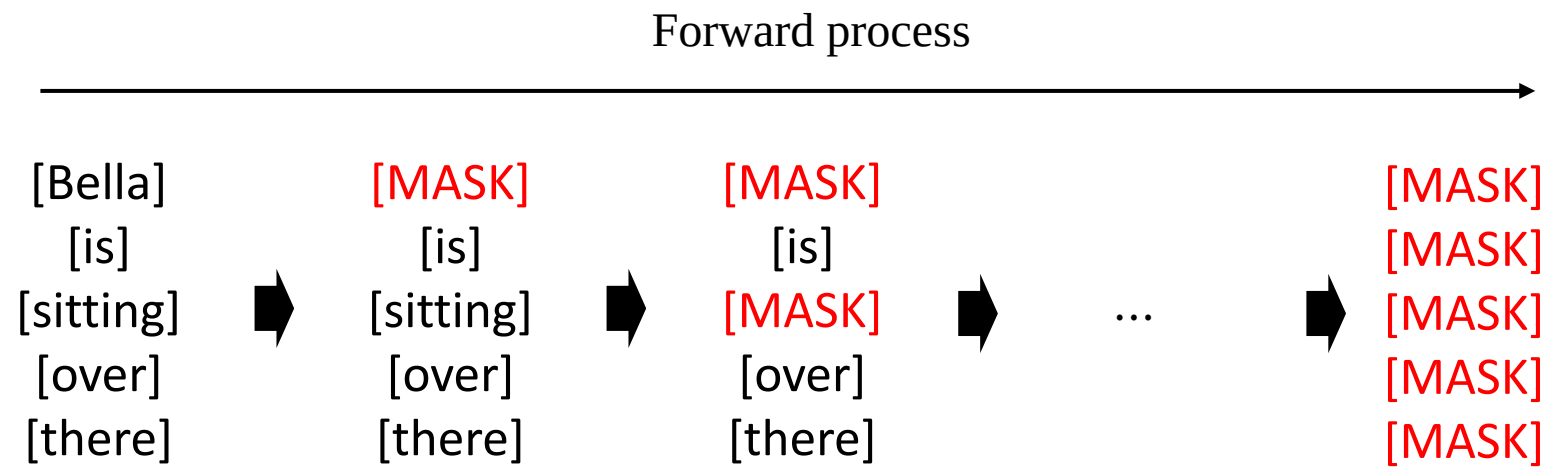
```
0 [0.    0.75 0.    0.25]
1 [0.    0.562 0.    0.438]
2 [0.    0.422 0.    0.578]
3 [0.    0.316 0.    0.684]
4 [0.    0.237 0.    0.763]
5 [0.    0.178 0.    0.822]
6 [0.    0.133 0.    0.867]
7 [0.   0.1 0.   0.9]
8 [0.    0.075 0.    0.925]
9 [0.    0.056 0.    0.944]
```

sampling

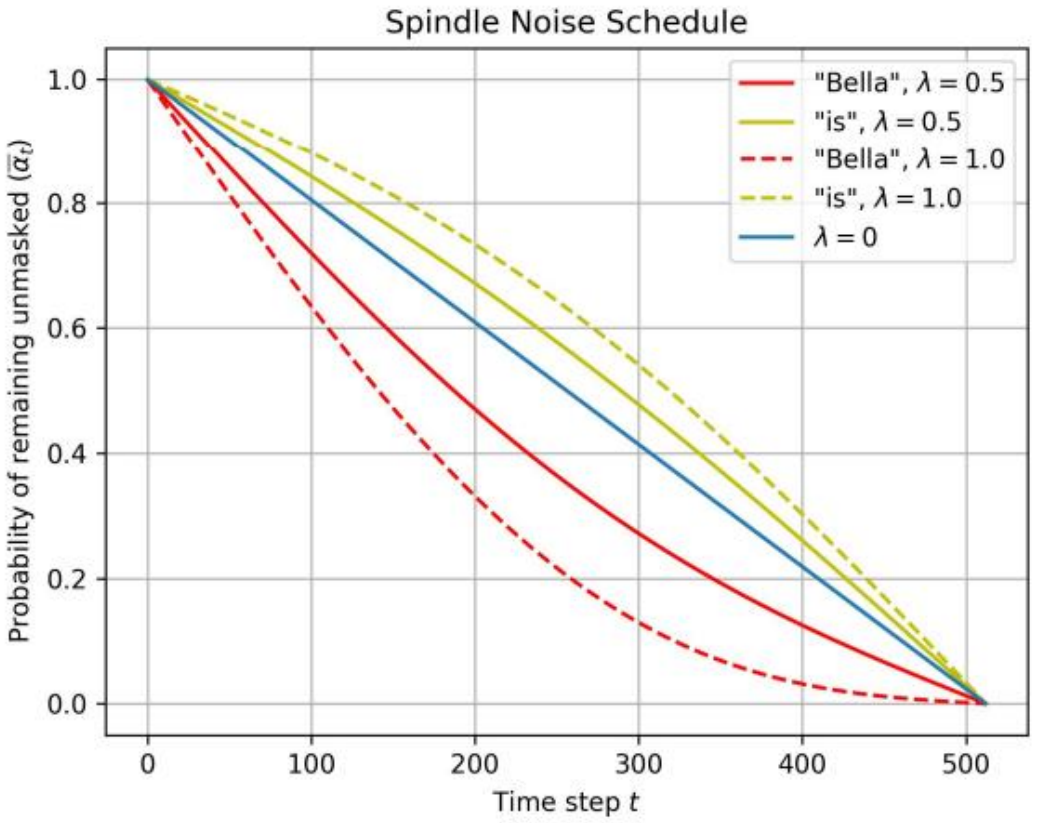
Spindle Noise Schedule

- β_t : **Noise schedule에 관한 고찰**
  - Sequence에서 token간 언어적 차이를 고려하지 않음
  - LM모델은 더 높은 likelihood를 달성하기 위해 자주 등장하는 토큰을 생성하는 경향이 있음

- **Easy-first generative behavior : 가장 정보가 많은 토큰부터 MASK**



Spindle Noise Schedule



- ***t*** step에서 **Marginal** $q(x_i^i|x_0^i)$**:**

$$q(x_i^i|x_0^i) = \begin{cases} \overline{\alpha}_t & if \;\; x_t^i = x_0^i \\ 1 - \overline{\alpha}_t & if \; x_t^i = [M] \end{cases}$$

$$\overline{\alpha}_t^i = 1 - \frac{t}{T} - S(t) \cdot \widetilde{H}(\mathbf{x_0^i}),$$

$$S(t) = \lambda \sin \frac{t\pi}{T},$$

$$\widetilde{H}(\mathbf{x}_0^i) = 1 - \frac{\sum_{j=1}^{n} H(\mathbf{x}_0^j)}{nH(\mathbf{x}_0^i)}$$

Entropy : 학습데이터에서 $\mathbf{x}_0^i$가 나타난 frequency로 계산

- λ 값이 1.0일 때 “Bella”의 unmasked 확률이 빠르게 떨어지며 “is” 의 확률은 천천히 떨어짐
- $H(\mathbf{x}_0^i)$의 값이 커질수록 $\overline{\alpha}_t^i$ 값이 작아지며 MASK 확률이 올라감.

Main results

Method	Pretrained	Schedule	Time Step	PPL ↓	BLEU ↑	Self-BLEU ↓
D3PM (Austin et al., 2021)	✗	$(T-t+1)^{-1}$	LTE	82.34	0.3897	0.2347
			TAD	125.15	0.3390	0.2720
		Spindle	LTE	77.50	0.4241	0.2288
Diffusion-LM (Li et al., 2022)	✗	Cosine	LTE	118.62	0.3553	0.2668
	✓	Cosine	LTE	132.12	0.3562	0.2798
BERT-Mouth (Wang and Cho, 2019)	✓	-	-	142.89	0.2867	**0.1240**
DiffusionBERT	✓	$(T-t+1)^{-1}$	LTE	92.53	0.3995	0.2118
			PTE	79.95	0.3886	0.2156
			TAD	78.76	0.4213	0.2116
		Spindle	TAD	**63.78**	**0.4358**	0.2151

- **D3PM과 비교** : Pretrained LM을 사용 했을 때 전반적으로 좋은 성능을 보임
- Continues domain에서 가우시안 노이즈를 사용하는 Diffusion-LM의 경우 pretrained LM을 사용했을 때 성능이 낮아짐(이유: Continues domain에서는 PLM을 적용하기가 쉽지 않음)